Supplemental Material

A. Mathematical derivation of alignment loss

Alignment loss, as one of the optimization objectives, aims to improve the representation consistency of individual interactions by forcing the user embedding to be close to the embedding of its positive sample items. This loss works in conjunction with the dual-tower encoder with shared parameters in the Siamese network structure, on the one hand preserving the semantic space of users and items, and on the other hand avoiding semantic confusion through local alignment. Unlike global alignment in contrastive learning, our method does not involve negative samples, so it does not interfere with the preservation of user personality and ensures semantic independence. In addition, we give the gradient calculation and parameter update formula of the alignment loss below to help readers intuitively understand its working process from an optimization perspective. To further enhance the intuitiveness, we also add a visual example of movie recommendation to illustrate how the alignment process optimizes the distance between users and their preferred items and maintains the objectivity of shared item embeddings.

1) Definition and derivation of alignment loss function: To enhance the representation consistency between users and their interactive items, we introduce an optimization-level alignment loss function $\mathcal{L}_{align}$, which is defined as follows:

$$\mathcal{L}_{align} = \frac{1}{|B|} \sum_{(u,i)\in B} \parallel f_U(e_u) - f_I(e_i) \parallel^2 \tag{1}$$

Where B represents the user-item positive sample pairs in the training mini-batch; $f_U(e_u)$ and $f_I(e_i)$ are the l_2 normalized user and item embeddings. The loss term aims to minimize the distance between the positive sample pairs so that the user embeddings gradually approach their corresponding positive sample items.

Gradient derivation and convergence trend

Taking the derivative of the user embedding $f_U(e_u)$, we get:

$$\frac{\partial \mathcal{L}_{align}}{\partial f_U(e_u)} = \frac{2}{|B|} \sum_{(u,i)\in B} (f_U(e_u) - f_I(e_i)) \tag{2}$$

Taking the derivative of the item embedding $f_I(e_i)$, we get:

$$\frac{\partial \mathcal{L}_{align}}{\partial f_I(e_i)} = \frac{2}{|B|} \sum_{(u,i)\in B} (f_I(e_i) - f_U(e_u)) \tag{3}$$

Finally, the embedding vector is updated by gradient descent:

$$f_U(e_u) \leftarrow f_U(e_u) - \eta \cdot \frac{2}{|B|} \sum_{(u,i)\in B} (f_U(e_u) - f_I(e_i)) \tag{4}$$

$$f_I(e_i) \leftarrow f_I(e_i) - \eta \cdot \frac{2}{|B|} \sum_{(u,i)\in B} (f_I(e_i) - f_U(e_u)) \tag{5}$$

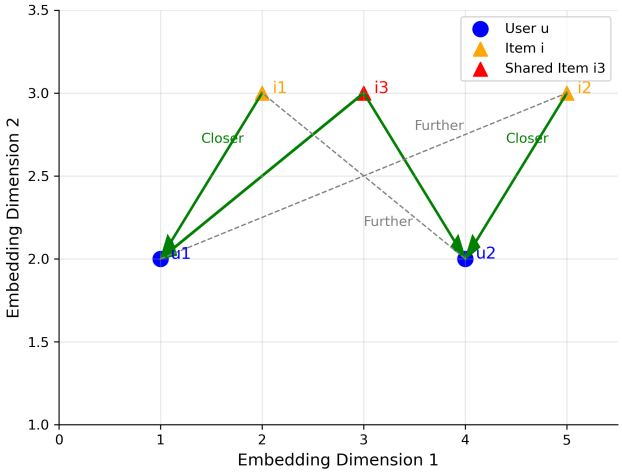


Fig. 1: A typical example of movie recommendation.

Through continuous iterative optimization, the embedding distance between the user and its positive sample items will gradually narrow, achieving an alignment effect.

2) Alignment loss combined with SiamGCL network: SiamGCL adopts the Siamese network structure, which consists of two sub-encoders with shared parameters, respectively processing the representation of the same batch of interaction samples under different enhanced views. Assume that the input is two sets of views, the first view gets $f_U^{(1)}(e_u)$ and $f_I^{(1)}(e_i)$, and the second view gets $f_U^{(2)}(e_u)$ and $f_I^{(2)}(e_i)$.

We calculate the alignment loss in two views separately:

$$\mathcal{L}_{align}^{(1)} = \frac{1}{|B|} \sum_{(u,i)\in B} \left\| f_U^{(1)}(e_u) - f_I^{(1)}(e_i) \right\|^2 \tag{6}$$

$$\mathcal{L}_{align}^{(2)} = \frac{1}{|B|} \sum_{(u,i)\in B} \left\| f_U^{(2)}(e_u) - f_I^{(2)}(e_i) \right\|^2 \tag{7}$$

The final alignment loss is:

$$\mathcal{L}_{align} = \frac{\gamma}{2} \left(\mathcal{L}_{align}^{(1)} + \mathcal{L}_{align}^{(2)} \right) \tag{8}$$

This loss will participate in the training objective function together with BPR loss and contrast loss, thereby guiding the learning of representation consistency and discriminability at the same time.

3) Examples and functions of alignment mechanisms: We use an example to explain the alignment problem more clearly, A typical example: Take the movie recommendation scenario as an example (The blue circle represents the user, the yellow triangle represents the item, and the red triangle represents the shared item, which is the common preference of the users. In addition, the green line represents the process optimized by alignment, and the gray dotted line represents the process optimized by BPR).

Assumptions:

1) User $u1$ mainly prefers science fiction movies, such as item $i1$.

2) User $u2$ prefers action movies, such as item $i2$.

3) Both user $u1$ and user $u2$ have watched the anime movie $i3$ (Shared Item $i3$).

4) The negative samples of user $u1$ and user $u2$ are $i2$ and $i1$ respectively (That is to say, user $u1$ does not interact with $i2$, and user $u2$ does not interact with $i1$.)

In the alignment design:

1) User $u1$'s embedding $f(u1)$ will be optimized to be closer to its positive sample item $f(i1)$.

2) User $u2$'s embedding $f(u2)$ will be optimized to be closer to its positive sample item $f(i2)$.

3) For the shared item $i3$, its embedding $f(i3)$ remains objective, reflecting only its general attributes as a movie, and will not directly interfere with the personalized expression of $u1$ and $u2$.

In addition, the embedding $f(u1)$ of user $u1$ and the embedding $f(i2)$ of item $i2$ will be pulled apart BPR, and similarly, the embedding $f(u2)$ of user $u2$ and the embedding $f(i1)$ of item $i1$ will also be pulled apart by BPR.

Therefore, aligning user and item embeddings can achieve the following effects:

Personalization preservation: Although $u1$ and $u2$ have both seen $i3$, their different preferences for other items ($i1$ and $i2$) ensure that user embeddings remain personalized in space.

Semantic non-interference: The embedding of the shared item $i3$ only reflects its general properties as a movie, and does not cause the embedding to deviate due to its relationship with a specific user.

B. Datasets

- **ml-1M**[1]: A widely used movie recommendation dataset, collected by the GroupLens research team from their movie website. It includes 1,000,209 ratings generated by 6,040 users on 3,900 movies. Each rating data contains the user ID, movie ID, rating (on a scale of 1-5), and timestamp (representing the time when the rating was made).
- **LastFM**[2]: It originates from the Last.fm website, which provides music social networking services, and has been widely used in music recommendation research. The dataset records user information, song and artist details, and user listening history. In our study, we mainly focus on the interaction between users and artists to recommend artists to users. The dataset used in our study contains 92,834 user-artist interactions involving 1,892 users and 17,632 artists.
- **Yelp2018**[3]: It originates from the Yelp Challenge and contains a large number of active interactions between many users and business entities. It has become a widely adopted benchmark for evaluating the performance of various models in RS. The dataset we use includes 31,668 users, 38,048 business entities, and a total of 1,561,406 interactions between them.
- **Gowalla**[4]: This dataset comes from a location-based social network platform where users share their locations through check-ins. It consists of 29,859 users, 40,989 items, and 1,027,464 interactions.
- **Alibaba-iFashion**[5]: This dataset from Alibaba's e-commerce platform focuses on fashion clothing. It is a particularly sparse dataset in the RS, with a density of only 0.00007. It involves 300,000 users, 81,614 items, and a total of 1,607,813 interactions.

C. Baselines

- **BPR-MF** [1]: A general recommendation moel to solve the sorting problem in RS. It selects and arranges items based on the user's behavioral data to ensure the front row items are more likely to appeal to the user. This algorithm is based on the largest posterior estimation, derived from Bayesian analysis.
- **NCF** [2]: A neural collaborative filtering model applies neural networks to model implicit feedback. Differing from matrix factorization, NCF replaces inner products with neural networks and utilizes multi-layer perceptrons to learn user-item interactions. This approach enhances the nonlinear capabilities of the modeling process.
- **NGCF** [3]: A recommendation model that employs GNN to directly model the user-item interaction graph. This approach is capable of capturing high-order connections between users and items, thereby enhancing the accuracy and relevance of the recommendations.
- **LightGCN** [4]: A simplified and enhanced recommendation model based on Graph Convolution Network (GCN) retains the core neighbor aggregation operation and linearly propagates user and item embedding vectors on the user-item interaction graph.
- **SGL** [5]: A GCL-based recommendation model employs three augmentation versions — node dropout, edge dropout, and random walk — to generate multiple enhanced views of a node. By bringing the consistency closer between different views of the same node, while simultaneously pushing the consistency further between views of different nodes, it guides the representation learning of nodes.
- **SimpleX** [6]: A simple baseline based on collaborative filtering emphasizes that the performance of collaborative filtering-based recommendation models depends not only on the encoder but also on the loss function and negative sampling. To address this, the approach proposes the use of cosine contrastive loss in current recommendation models.
- **NCL** [7]: A neighborhood-enhanced contrastive learning method is introduced that leverages neighbors within the graph structure and semantic space to mitigate data sparsity in graph collaborative filtering.

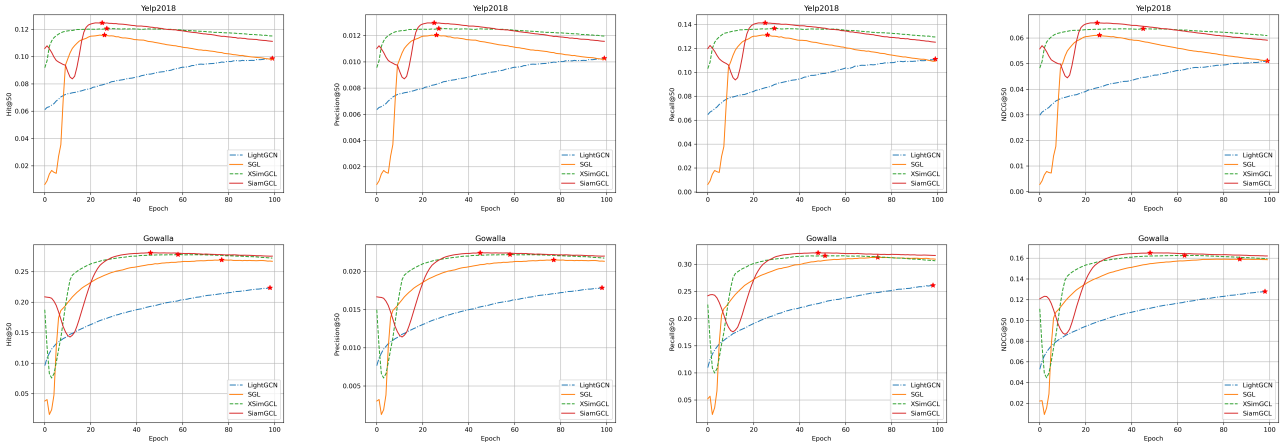
[1]http://files.grouplens.org/datasets/movielens/

[2]http://ir.ii.uam.es/hetrec2011

[3]https://www.yelp.com/dataset

[4]https://snap.stanford.edu/data/loc-gowalla.html

[5]https://www.alibabagroup.com/



(a) Hit (b) Precision (c) Recall (d) NDCG

Fig. 2: The training cures in the first 100 epochs of LightGCN, SGL, XSimGCL, and SiamGCL on Yelp2018 and Gowalla.

- **XSimGCL** [8]: An extremely simple graph contrastive learning method is introduced, which does not rely on complex graph augmentations, but instead uses simple random noise embedding augmentations.
- **DirectAU** [9]: A novel loss function is introduced, designed to quantify alignment and uniformity from the perspectives of representation alignment and uniformity. This approach aims to significantly improve recommendation performance.
- **AdaGCL** [10]: Two learnable view generators—a graph generation model and a graph denoising model—are introduced to adaptively generate contrastive views. This method enhances the scalability of the augmentation strategy, aiming to better address data sparsity and noise issues.

D. Comparison of the different training curves

In our study, we compare the performance of our method with that of LightGCN, SGL, and XSimGCL during training on the Yelp2018 and Gowalla datasets using Hit, Precision, Recall, and NDCG metrics. The results, depicted in Figure 2 where stars denote optimal values, indicate that models incorporating contrastive learning exhibit superior performance and require fewer epochs to train compared to traditional graph-based recommendation models. It is worth noting that LightGCN did not converge within 100 epochs on both Yelp2018 and Gowalla datasets. Our experiments show that it may take about 300-500 epochs to converge. In contrast, SGL, XSimGCL, and our model SiamGCL all converged within 60 epochs. In addition, compared to the current state-of-the-art graph-based contrastive learning recommendation models such as SGL and XSimGCL, our method SiamGCL achieves the best performance in fewer epochs. For example, on the Gowalla dataset, SiamGCL only needs 49 epochs to achieve the best performance, which is less than 52 cycles of XSimGCL and 78 epochs of SGL.

E. Model robustness study

1) Addressing data sparsity: In recommendation scenarios, the sparsity of user-item interactions affects recommendation performance. To assess our proposed method under different levels of sparsity, we divided the total interactions from the Alibaba-iFashion dataset into five equal groups. These groups were ordered based on the number of users, from the lowest to the highest, resulting in five datasets labeled G1 to G5. It's worth noting that the G5 has the most users, but also the sparsest. The density of these groups were 0.00043, 0.00018, 0.00011, 0.00008, and 0.00005, respectively.

As shown in Figure 3, the results illustrate the impact of different sparsity levels in user-item interactions on the performance of the SGL, DirectAU, XSimGCL, and SiamGCL methods. For SGL, the Recall@20 results show a downward trend from G1 to G5, starting from 0.0782 for G1 and decreasing to 0.0693 for G5. This shows that the performance of SGL decreases significantly as the sparsity increases. In contrast, SiamGCL shows better resistance to sparsity interactions. Its performance peaked in the middle group, with G3's result of 0.0974. Even in the sparsest group G5, it only drops to 0.0805, indicating relative stability under high sparsity conditions.

2) Handling noise interactions: In this study, we conducted a series of noise injection experiments to explore the impact of noise data on user-item interaction models. Specifically, we introduced different proportions of noise data into the original training set: 0%, 5%, 10%, 15%, and 20%. For each proportion, we randomly selected the corresponding percentage of user interaction records. For each selected record, we constructed a new interaction instance involving a product not previously interacted with the user. These newly constructed interaction instances were then added to the training set, while the test set remained unchanged.

Table I illustrate the impact of injecting different proportions of noise interactions on the performance of recommendation methods. It is observed that contrastive learning methods SGL and SiamGCL maintain a decline rate of approximately 11% in Recall@50 and NDCG@50, even as the noise injection ratio

TABLE I: Performance comparison of Recall@50 and NDCG@50 with clean data and with 5%, 10%, 15%, and 20% noisy interactions on Gowalla. The number in parentheses represents the percentage decrease compared to the same method without noise injection.

Methods	Noisy interactions									
	0%		5%		10%		15%		20%	
	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG
LightGCN	0.2955	0.1492	0.2730 (↓7.61%)	0.1380 (↓7.50%)	0.2566 (↓13.16%)	0.1295 (↓13.20%)	0.2438 (↓17.49%)	0.1240 (↓16.89%)	0.2355 (↓20.30%)	0.1190 (↓20.24%)
SGL	0.3117	0.1584	0.3016 (↓3.24%)	0.1533 (↓3.21%)	0.2922 (↓6.25%)	0.1483 (↓6.37%)	0.2844 (↓8.75%)	0.1446 (↓8.71%)	0.2786 (↓10.61%)	0.1415 (↓10.66%)
XSimGCL	0.3177	0.1627	0.3129 (↓1.51%)	0.1605 (↓1.35%)	0.3080 (↓3.05%)	0.1575 (↓3.20%)	0.3023 (↓4.84%)	0.1552 (↓4.60%)	0.2988 (↓5.94%)	0.1531 (↓5.90%)
DirectAU	0.2869	0.1437	0.2831 **(↓1.32%)**	0.1421 **(↓1.11%)**	0.2827 **(↓1.46%)**	0.1417 **(↓1.39%)**	0.2823 **(↓1.60%)**	0.1412 **(↓1.73%)**	0.2798 **(↓2.47%)**	0.1402 **(↓2.43%)**
SiamGCL	**0.3224**	**0.1653**	**0.3178** (↓1.42%)	**0.1631** (↓1.33%)	**0.3122** (↓3.16%)	**0.1606** (↓2.84%)	**0.3075** (↓4.62%)	**0.1579** (↓4.47%)	**0.3040** (↓5.70%)	**0.1562** (↓5.50%)

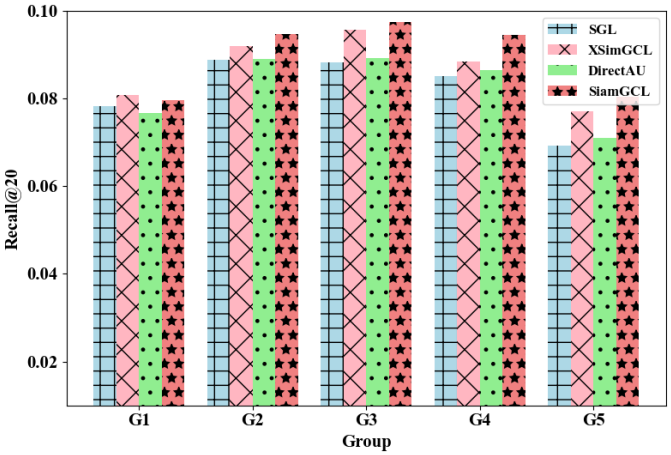


Fig. 3: Performance of different sparsity-level users. G5 has the highest sparsity level.

increases. In contrast, the graph model LightGCN, which does not incorporate contrastive learning, experiences a significant performance decrease with even a small amount of noise injection (only 10%). Thus, it is evident that contrastive learning enhances noise resistance in the recommendation domain. Furthermore, our method outperforms the classical contrastive method SGL in noise resistance. Specifically, with 5% noise interaction injection, our method only experiences a 1.42% and 1.33% decrease in Recall@50 and NDCG@50, respectively, compared to over a 3% decrease in the SGL method. Additionally, when the noise interaction injection ratio reaches 20%, the performance of the SGL method significantly declines by over 10%, while our method remains around a 5% decrease. However, we also observed that under all noise injection rates, DirectAU maintained the best performance and SiamGCL achieved the second best performance, which also revealed that studying denoising recommendation from the perspective of representation alignment and uniformity is a promising direction.

REFERENCES

[1] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: bayesian personalized ranking from implicit feedback," in *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 2009, pp. 452–461.

[2] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. Chua, "Neural collaborative filtering," in *Proceedings of the 26th International Conference on World Wide Web*. ACM, 2017, pp. 173–182.

[3] X. Wang, X. He, M. Wang, F. Feng, and T. Chua, "Neural graph collaborative filtering," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2019, pp. 165–174.

[4] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "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*. ACM, 2020, pp. 639–648.

[5] J. Wu, X. Wang, F. Feng, X. He, L. Chen, J. Lian, and X. Xie, "Self-supervised graph learning for recommendation," in *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2021, pp. 726–735.

[6] K. Mao, J. Zhu, J. Wang, Q. Dai, Z. Dong, X. Xiao, and X. He, "Simplex: A simple and strong baseline for collaborative filtering," in *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management*. ACM, 2021, pp. 1243–1252.

[7] Z. Lin, C. Tian, Y. Hou, and W. X. Zhao, "Improving graph collaborative filtering with neighborhood-enriched contrastive learning," in *WWW '22: The ACM Web Conference*. ACM, 2022, pp. 2320–2329.

[8] J. Yu, X. Xia, T. Chen, L. Cui, N. Q. V. Hung, and H. Yin, "Xsimgcl: Towards extremely simple graph contrastive learning for recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 2, pp. 913–926, 2024.

[9] C. Wang, Y. Yu, W. Ma, M. Zhang, C. Chen, Y. Liu, and S. Ma, "Towards representation alignment and uniformity in collaborative filtering," in *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, 2022, pp. 1816–1825.

[10] Y. Jiang, C. Huang, and L. Huang, "Adaptive graph contrastive learning for recommendation," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*, A. K. Singh, Y. Sun, L. Akoglu, D. Gunopulos, X. Yan, R. Kumar, F. Ozcan, and J. Ye, Eds. ACM, 2023, pp. 4252–4261. [Online]. Available: https://doi.org/10.1145/3580305.3599768