
SUPERVISED CONTRASTIVE LEARNING FOR RECOMMENDATION

Chun Yang

School of Automation Engineering, University of Electronic Science and Technology of China
Chengdu, Sichuan, China
beiluo@std.uestc.edu.cn

ABSTRACT

Compared with the traditional collaborative filtering methods, the graph convolution network can explicitly model the interaction between the nodes of the user-item bipartite graph and effectively use higher-order neighbors, which enables the graph neural network to obtain more effective embeddings for recommendation, such as NGCF And LightGCN. However, its representations is very susceptible to the noise of interaction. In response to this problem, SGL explored the self-supervised learning on the user-item graph to improve the robustness of GCN. Although effective, we found that SGL directly applies SimCLR's comparative learning framework. This framework may not be directly applicable to the scenario of the recommendation system, and does not fully consider the uncertainty of user-item interaction.

In this work, we aim to consider the application of contrastive learning in the scenario of the recommendation system adequately, making it more suitable for recommendation task. We propose a supervised contrastive learning framework to pre-train the user-item bipartite graph, and then fine-tune the graph convolutional neural network. Specifically, we will compare the similarity between users and items during data preprocessing, and then when applying contrastive learning, not only will the augmented views be regarded as the positive samples, but also a certain number of similar samples will be regarded as the positive samples, which is different from SimCLR who treats other samples in a batch as negative samples. We term this learning method as Supervised Contrastive Learning(SCL) and apply it on the most advanced LightGCN. In addition, in order to consider the uncertainty of node interaction, we also propose a new data augment method called node replication. Empirical research and ablation study on a data set prove the effectiveness of SCL and node replication, which improve the accuracy of recommendations and robustness to interactive noise.

Keywords Supervised contrastive Learning · Graph convolution network · Recommended system · Representational learning

1 INTRODUCTION

In order to meet the diverse and personalized information needs of users, the personalized recommendation system has emerged [1, 2, 3, 4]. A recommendation system can help users find items of interest in a large amount of candidate items. It recommends similar items to users with similar behaviors based on interactive content such as purchases, clicks, and ratings. This method is called collaborative filtering. Collaborative filtering generally uses embedding to represent users and items, and predicts scores based on embedding. In recent years, a large number of collaborative filtering methods have been successfully implemented, such as [5, 6, 7, 8, 9].

Although these collaborative filtering methods are effective, they only consider the explicit interaction between a single user and a single item, and do not encode the collaborative signal explicitly. When the interaction data is very sparse, it is difficult to learn embedded representations with sufficient knowledge. Can not effectively characterize the similarities between users or items [10, 11].

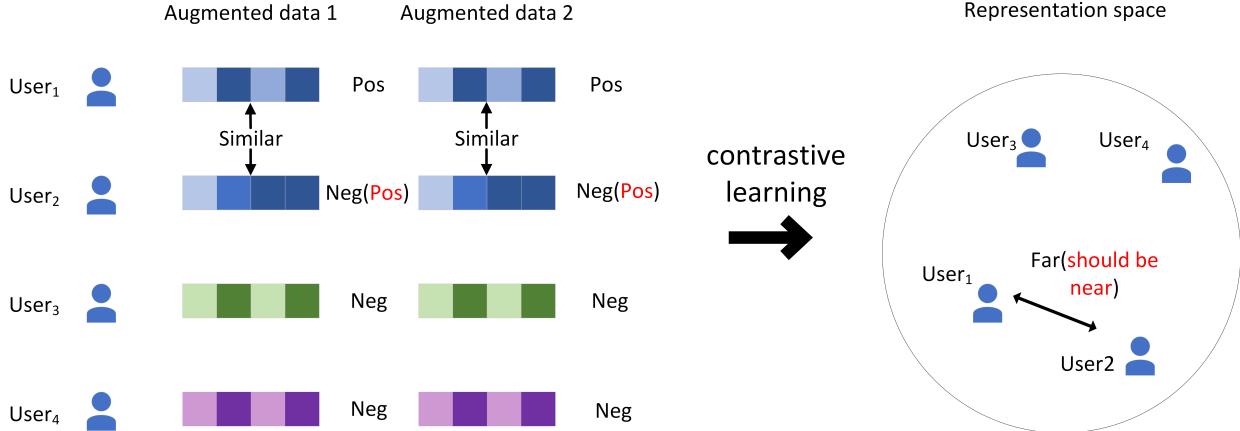


Figure 1: The limitation of contrastive learning framework used by SGL. SGL treats other samples in a batch as negative samples, which makes the representation of similar users further in the representation space, and it should be closer to be more effective for subsequent recommendation task. Our proposed SCL introduces supervised information and labels similar users as positive samples, as shown by the red word in the figure. After contrastive learning, similar users are closer in the representation space, which can obtain representations that are more conducive to recommendation.

In order to make use of the user-item bipartite graph high-hop neighbor information fully, more effectively represent the embedding of users and items, Wang et al. recently proposed graph convolution collaborative filtering, and the best experimental results were obtained on three data sets [12]. They transferred the concept of graph convolution network to the field of recommendation systems, and regarded the user-item bipartite graph as a isomorphic graph, and carried out feature embedding, information dissemination and feature aggregation on the graph to obtain the final representation of users and items. Their method encoded the high-hop neighbors between the user and the item explicitly, so they get more reliable and more informative embedding representations and achieve better results.

Based on NGCF, He et al. proposed lightGCN [13]. They believe that many operations of NGCF are directly inherited from GCN. These operations are not necessarily suitable for collaborative filtering tasks in the recommendation field. Therefore, they simplify many operations of NGCF, reduce the difficulty of model training, and achieve the current state-of-the-art on multiple data sets.

GCN provides a most advanced and efficient solution for solving user-item bipartite graph sparsity problems and interactive modeling problems. However, Wu et al. believe that the representations learned by GCN are easily biased towards high-frequency items or users and are also easily affected by noise of interactions. In order to address the above limitations, they proposed Self-supervised Graph Learning(SGL) for recommendation, which uses unlabeled data space by augmenting the input data, thereby achieving a significant improvement in downstream tasks [14]. SGL refer to the contrastive learning of SimCLR and use the framework of simCLR as a paradigm of self-supervised learning [15].

Although SGL has achieved a certain degree of effect improvement, we believe that the contrastive learning framework of SimCLR may not be very suitable for recommendation tasks. Specifically, the application background of SimCLR is a computer vision(CV) task. It treats two augmenting samples of a input data as the positive samples and treats other samples in the batch as the negative samples. Due to the diversity of samples in CV tasks, this design is reasonable and helps to mine hard negative samples which improve the quality of the learned representations. However, in the recommendation system, the core of the comparison is the users or items node and the ultimate goal is to compare the cosine similarity of the data. This means that there is a high probability that there are similar users or items in a batch. Regarding these samples as the negative samples, similar users or items will become farther away in the representation space, which violates the optimization purpose of the recommendation system, affects the final representation learned by GCN, and reduces the performance of the recommendation system.

In order to solve the above limitations, we believe that it is necessary to improve the application of contrastive learning. We propose a Supervised Contrastive Learning(SCL) for recommendation. We hope to design a framework of contrastive learning with the recommendation task as the goal, so as to obtain the representations that is more in line with the requirements of the recommendation task and improve the performance of downstream tasks. Specifically, it consists of two steps: (1) data augmentation, which generates multiple views for each node; (2) contrastive learning, which makes similar samples are closer and different samples are further in the representation space. Since the Bayesian Personalized

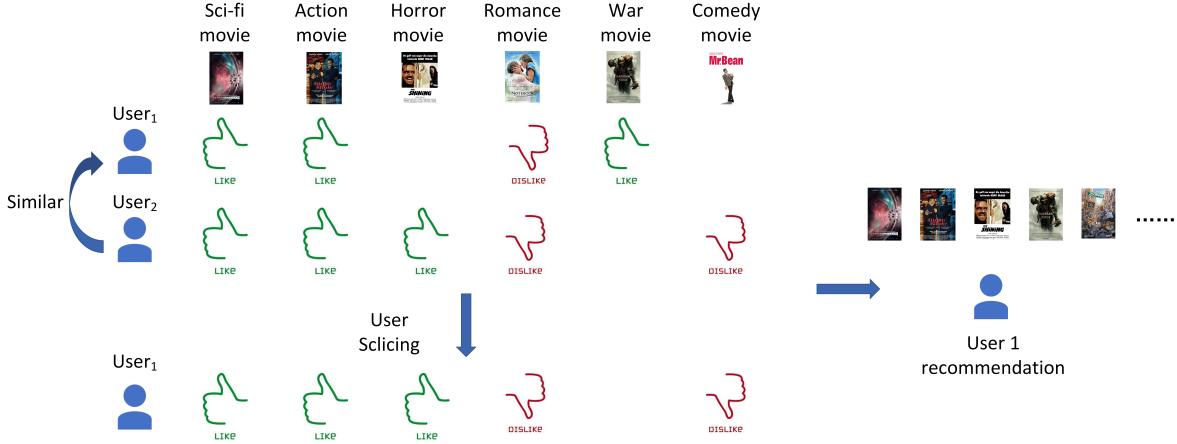


Figure 2: Motivation for node replication. By replacing part of the data of user 1 with the corresponding data of similar user 2, the augmented sample are more diverse, and the obtained representation can better express the user’s preferences, thereby making the recommendation results more diverse.

Ranking(BPR) loss of the recommendation task takes node interaction as input, a batch may contain a large number of similar users or nodes [10]. Directly treating other samples as the negative samples may not be conducive to the learning of representation, which is different from SGL directly taking contrast learning as In the way of auxiliary tasks, SCL use comparative learning as a pre-training task to obtain a preliminary representation, and then use BPR loss for fine-tuning. In addition, in order to be more adaptable to the recommendation task, we will not simply treat other samples in the same batch as negative samples when we perform comparative learning, but treat all similar samples as positive samples, and dissimilar samples as negative samples. This comparative learning method introduces supervised information, which makes similar users or items more inclined to learn similar representations, which is beneficial to downstream recommendation tasks. This supervised and contrasted representation learning method starts from the goal of the recommendation system, and its design is more suitable for the recommendation system and can further improve the performance of the recommendation task.

In addition, we found that the BPR loss is simply optimized by constructing interactive and non-interactive triples. However, the user-item bipartite graph may have noisy interaction or similar user item pairs without interaction. From the purpose of the recommendation system, in order to further improve the recommendation performance, we believe that the edge drop and node drop on the user-item bipartite graph does not bring enough diversity information. In order to improve the model’s adaptability to noise interaction and the diversity of the representations, we propose a new data augmentation method called node replication. Specifically, we replace part of the interaction of the current node with the corresponding interaction of similar nodes according to a certain probability. This data augmentation method can effectively improve recommendation performance and help increase the diversity of recommendations.

It is worth noting that our SCL can be applied to any collaborative filtering network based on the user-item bipartite graph. In this work, we choose to implement it on the most advanced LightGCN, and prove the effectiveness of SCL on a benchmark data set, which can significantly improve the recommendation performance.

To summarize, The main contributions of this paper are summarized as follows:

- We propose a new supervised comparative learning paradigm, which considers the purpose of the recommendation system and provides supervised information for representation learning, so that similar nodes are closer in the representation space, which helps to improve the recommendation performance.
- In order to improve the representation ability of nodes and obtain more diverse information, we propose a new data augmentation method called node replication, which improves the robustness and diversity of the representations, and makes the recommendation results more diverse and has better performance.
- We verified the effect of SCL on a benchmark dataset and proved the effectiveness of SCL.

2 PRELIMINARIES

2.1 PROBLEM STATEMENT

The set of user-item interactions can be easily modeled as a bipartite graph $\mathcal{G} = (\mathcal{U}, \mathcal{V}, \mathcal{E})$, where \mathcal{U} represents the user set, \mathcal{V} represents the item set, and \mathcal{E} represents the edge set. If there is an interaction between a user and an item, then there is an edge between them [16]. The collaborative filtering based on GCN needs to embed each user $i \in \mathcal{U}$ and item $j \in \mathcal{V}$ into a unified d -dimensional representation space, and the corresponding representations are expressed by $u \in \mathbb{R}^d$ and $v \in \mathbb{R}^d$. By calculating the inner product between the representations of user and the item we can know whether the item should be recommended to the user.

2.2 GRAPH CONVOLUTIONAL NETWORK

The core of GCN is to aggregate the domain representations of each node on the bipartite graph \mathcal{G} , and update the representation of current node by considering the information of high-order neighbors, and finally obtain an effective representation. For user i , its representation is calculated as follows:

$$u_i^{(l)} = \delta(W^{(l)}[u^{(l-1)}_i | A(v_j \in \mathcal{N}(u_i))]) \quad (1)$$

Where $v_j \in \mathcal{N}(u_i)$ represents the domain node of user i , A denotes the node aggregation function, which can be average, weighted sum, linear mapping, etc [17, 18, 19]. $[\cdot | \cdot]$ is a concatenation operation and $W^{(l)}$ is the weight matrix of the l -th layer, δ is the activation function of the l -th layer, which can be *sigmoid*, *Relu*, or *None*, etc [20, 18, 21]. $u_i^{(l)}$ denotes the representation of user i at layer l , $u^{(l-1)}_i$ is that of previous layer, and $u^{(0)}_i$ is the ID embeddings which are trainable parameters.

Specifically, GCN first aggregates the representations of all domain nodes of user i at layer $l - 1$, then combines it with its own representation, and then obtains the representation of layer l through linear mapping. For a certain item j , the calculation method of its representation is the same as that on the user side. The obtained representation of the l layer corresponds to the information aggregation of the node's l hop neighbor. For different layers of information, there are also different ways to generate the final node representation, such as using the last layer only, splicing, averaging, weighted sum, attention and so on [12, 13, 22, 23, 24].

After obtaining the final representation of each node, the prediction layer can be used to calculate a certain user i 's preference for item j . In order to ensure the calculation efficiency of the GCN, the vector inner product is widely used as the prediction layer at present [13]:

$$\hat{y}_{ij} = u_i^T v_j \quad (2)$$

When optimizing model parameters, GCN generally use BPR loss, and BPR loss chooses a user, an interactive item and a non-interactive item to form a triple, and the predicted value of interaction is expected to be greater than the predicted value of no interaction:

$$L_{BPR} = - \sum_{u=1}^M \sum_{i \in \mathcal{N}_u} \sum_{j \notin \mathcal{N}_u} \ln \sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \lambda L_2 \quad (3)$$

Where λ is the parameter that controls the intensity of L2 regularization, and M is the total number of interactions. BPR loss is the key to achieving the final recommendation result, and we apply it in the fine-tuning stage.

2.3 BACKGROUND CONTRASTIVE LEARNING

The framework of contrastive learning is shown in the figure 3. It maximizes the similarity between augmented samples generated by the same data and minimizes the similarity with other samples in the same batch to learn representations. The framework includes four components: data augmentation, encoder, projection head, and contrast loss function [15].

Specifically, the data augmentation module generates related views from the original data X_1 through some random data augmentation methods, denoted as X_1^1 and X_1^2 . In the bipartite graph, the data augmentation methods we used include node drop, edge drop, and node replication. Different data augmentation methods have a great impact on the performance of comparative learning.

The encoder generates a representation vector H_1^1 and H_1^2 from the augmented view. In the user-item bipartite graph problem, the encoder is the GCN network.

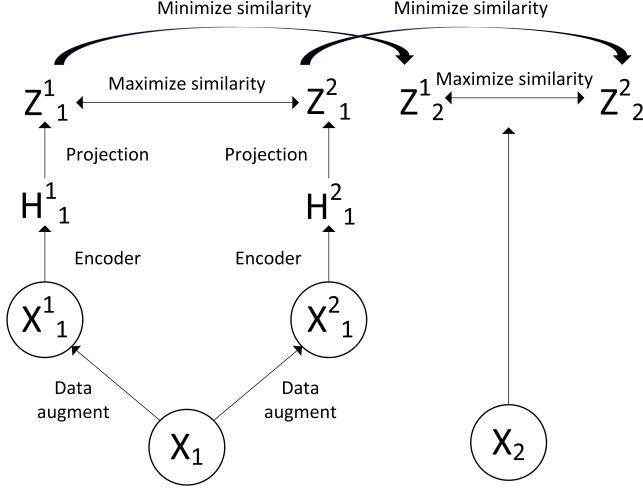


Figure 3: The framework of contrastive learning, it aim to minimize the distance of the augmented views generated by the same sample in the contrast loss space, while maximizing the distance of the augmented views generated by other samples in the same batch in that space.

The purpose of the projection head is to project the representation into the contrast loss space. The projection head is generally an MLP with a hidden layer. Many experiments on contrastive learning have proved that the projection head is beneficial to improve the performance of contrastive learning [15, 25, 26, 27].

The contrast loss function which called InfoNCE aims to minimize the distance of the augmented views generated by the same sample in the contrast loss space, while maximizing the distance of the augmented views generated by other samples in the same batch in that space [15, 28, 29, 30]:

$$\ell_{i,j} = -\log \frac{\exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_j) / \tau)}{\sum_{k=1}^{2N} 1_{[k \neq i]} \exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_k) / \tau)} \quad (4)$$

Where $\text{sim}(\mathbf{u}, \mathbf{v}) = \mathbf{u}^\top \mathbf{v} / \|\mathbf{u}\| \|\mathbf{v}\|$ denote the dot product between ℓ_2 normalized \mathbf{u} and \mathbf{v} . $1_{[k \neq i]}$ is an indicator function, when k is not equal to i , the value is 1, otherwise it is 0. τ denotes a temperature parameter.

3 METHODOLOGY

We present the proposed paradigm of Supervised Contrastive Learning(SCL) for recommendation system. The learning process is divided into two stages, as shown in the figure 4. First, we use supervised constrastive learning as the pre-training part to learn preliminary representations, and then apply the BPR loss to fine-tune the model.

In this section, we first introduce how to perform data augmentation to generate multiple views, and then introduce the design details of SCL. The two-stage training method is adopted because BPR loss uses interactions as input, while contrastive learning uses nodes as input.

SCL focuses on recommended task, hoping to make similar nodes similar in the representation space, and address the limitations of the comparative learning framework used in SGL, which makes SCL more suitable for recommendation systems.

3.1 Data Augmentation

In SimCLR, the authors discusses various augmentation methods for images [15], but for recommendation task, the data structure is different so that the data augmentation methods cannot be directly migrated. In the user-item bipartite graph recommendation task, the input data is a sparse matrix composed of interactions. We regard this data as the graph structure data as shown in the figure 4. In order to learn the representation of each node fully, we need to design new data augmentation schemes for graph structure data.

In SGL, the authors has designed various augmentation methods including node drop and edge drop for graph structure data [14]. In order to further improve the diversity of the recommendation results and ensure that the representation of

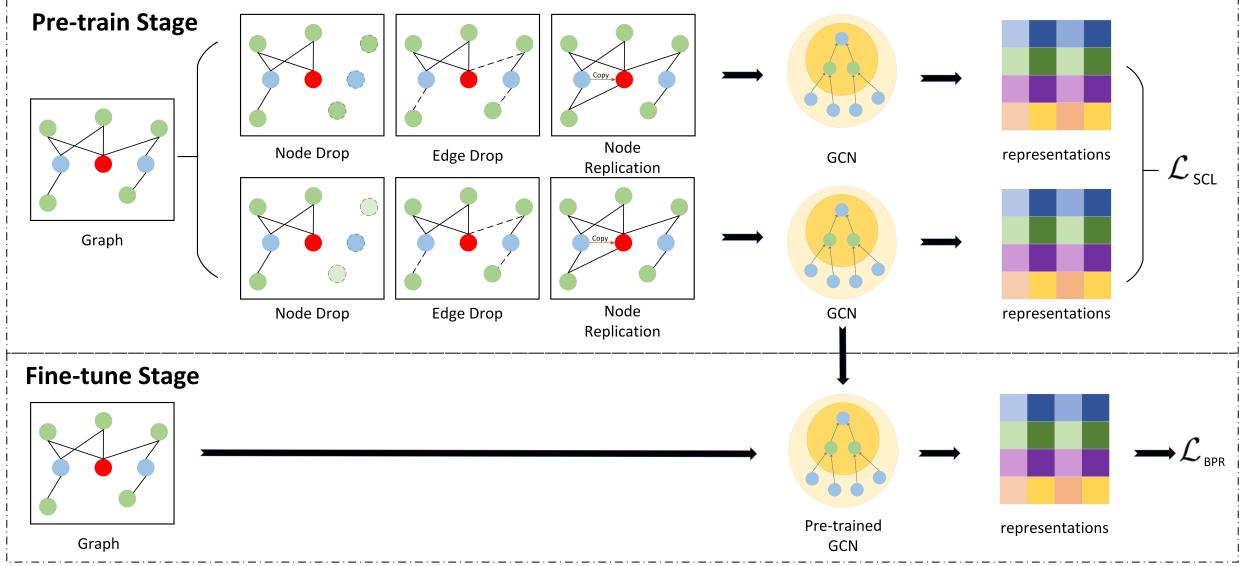


Figure 4: The framework of contrastive learning, it aim to minimize the distance of the augmented views generated by the same sample in the contrast loss space, while maximizing the distance of the augmented views generated by other samples in the same batch in that space.

each node contains the user's possible interests as much as possible, we proposed a new data augmentation method called node replication. We detail the augmentation methods as follows:

3.1.1 Node Drop(ND)

Node drop discards a certain node in the graph and any interactions(i.e. edge in graph) related to it according to probability ρ_1 :

$$V(\mathcal{G}) = (\mathbf{M}_1 \odot \mathcal{N}, \mathcal{E}) \quad (5)$$

Where $V(\mathcal{G})$ is the view generated by \mathcal{G} as input. \mathbf{M}_1 is the masking vectors which apply on the node set, where 0 indicates that the corresponding node is discarded. \mathcal{N} represents the set of nodes, including user nodes set \mathcal{U} and item nodes set \mathcal{V} .

When performing data augmentation, we can generate multiple views by applying formula 8 multiple times on the original graph data. This augmentation can reduce the impact of high-frequency nodes on the representation by randomly drop some nodes, so that the representations can learn more from the long tail node.

3.1.2 Edge Drop(ED)

Edge drop discards an certain interaction in the graph (i.e. edge in graph) according to the probability ρ_2 :

$$V(\mathcal{G}) = (\mathcal{N}, \mathbf{M}_2 \odot \mathcal{E}) \quad (6)$$

Where \mathbf{M}_2 is the masking vectors which apply on the edge set, where 0 indicates that the corresponding edge is discarded. \mathcal{E} represents the set of edges.

This augmentation randomly discards some edges, and it is expected that the representation learned by contrastive learning will not be affected by specific interactions, and the robustness of the representation will be improved.

3.1.3 Node Replication(NR)

Node replication will replace part of the interactions of the current node with the corresponding interactions of similar nodes according to probability ρ_3 . The similarity between the nodes is represented by the cosine, that is, the more similar is the interaction history, the more similar are the nodes. The matrix form of similarity calculation is as follows:

$$S = \mathcal{G}\mathcal{G}^T \quad (7)$$

Table 1: Statistics of the datasets

Dataset	Users#	Items#	Interactions#	Density
ML-100K	943	1682	100000	6.30%

Where \mathcal{G} is the sparse matrix represents the interaction history of each node. Then for each node we will sort S and select the first N nodes that can be used for replacement. It is worth noting that when calculating similarity, user-side nodes and item-side nodes are calculated separately.

We divide the interaction history of each node into k segments, and randomly replace one of them when performing node replication. Node replication is as follows:

$$V(\mathcal{G}) = (\mathbf{M}_3 \odot \mathcal{N}, \mathcal{E}) \quad (8)$$

Where \mathbf{M}_3 is the masking vectors which apply on the node set, where 0 indicates that the corresponding node is applied node replication.

This augmentation randomly replaces part of the interactions, and it is expected that the representation can contain more information, increase the diversity of recommendation results, and improve the performance of the recommendation system.

3.2 Supervised Comparative Learning

After establishing multiple views of the node, we input them into the GCN respectively, and then perform contrastive learning loss on the generated representations. Different from InfoNCE, which treats the views of the same node as a positive sample pair, and treats the views of any other different nodes as the negative samples, we propose a supervised contrastive learning loss, which treat similar nodes as the same type and all are regarded as positive sample. We call this loss supervised InfoNCE(S-InfoNCE):

$$\ell_{i,j} = -\log \frac{\sum_{k=1}^{2N} 1_{[j \in i]} \exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_j) / \tau)}{\sum_{k=1}^{2N} 1_{[k \notin i]} \exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_k) / \tau)} \quad (9)$$

Where $1_{[k \notin i]}$ denotes an indicator function, when k is not similar to i , the value is 1, otherwise it is 0. While $1_{[j \in i]}$ denotes that function, when j is similar to i . $\text{sim}(i, j)$ and τ is same as infoNCE.

Our S-InfoNCE encourages the representations of similar nodes to be close to each other in the representation space to ensure their consistency. In addition, dissimilar samples are taken as negative samples to ensure that the representations of these nodes are significantly different. We take the recommendation task as the goal, and believe that similar users should have similar representations to facilitate collaborative filtering.

3.3 Complexity Analyses of SCL

It is worth noting that our SCL learning paradigm will only perform pre-training with contrastive learning on the basis of GCN without introducing any additional parameters. In other words, the learning paradigm we proposed will not bring any parameter burden to the original model, and it will improve the recommendation performance while ensuring its high efficiency.

4 EXPERIMENT

To verify the effectiveness of the SCL for recommendation paradigm proposed in this paper, a dataset named MovieLens-100k(ML-100K) is adopted for top-K recommendation and the results are compared with other state-of-the-art methods. The specific statistics of ML-100K are shown in the table ??.

4.1 Experimental Settings

4.1.1 Experimental Metrics

In top-k recommendation, we follow the strategy described in [12] and the ranking protocol described in [12], and choose MAP (Mean Average Precision), MRR (Mean Reciprocal Rank) and MNDCG (Mean Normalized Discounted Cumulative Gain) to evaluate the recommendation performance. And consider the case where K is equal to 3, 5, 10 respectively.

The proposed method is compared with the pure MLP, LSTM, BiLSTM and IDCNN to verify validity and efficiency. Since the core innovation of this paper is the proposed time relationship unit and decoupling position embedding unit that can be regarded as a new feature extractor, for fairness, the same classification layer is applied when comparing the performance among the those methods.

4.1.2 Compared Baselines

We compare the proposed SCL with the following strong baselined CF models: DeepWalk [31], Node2vec [32], LINE [33], PinSage [34], GC-MC [35], IGMC [36], NeuMF [6], NGCF [12], LightGCN [15] and SGL [14]. On the basis of LightGCN, we implement three variants of SCL, namely SCL-ND, SCL-ED, and SCL-NR, which respectively applied the data augmentation method of Node Drop, Edge drop, Node Replication.

4.1.3 Implementation Details

The above methods are all using the official implementation. For fairness, all methods use the same training and testing data. The embedding size is uniformly fixed as 128 and the learning rate is 0.001. IGMC uses 1-hop subgraphs with an encoder depth of 2. The adam is used as optimizer and the batch size is set to 1024.

4.2 Performance Comparison

Table 2: Performance comparison of top-K recommendation on ML-100K

Method	MAP@3	MAP@5	MAP@10	MRR@3	MRR@5	MRR@10	NDCG@3	NDCG @5	NDCG @10
DeepWalk	2.72	3.54	4.92	43.86	46.83	48.75	7.17	9.32	13.13
Node2vec	3.07	3.9	5.19	44.8	48.02	49.78	7.69	9.91	13.41
LINE	2.45	3.26	4.67	44.8	48.02	49.78	7.69	9.91	13.41
PinSage	4.52	6.18	9.13	62.56	64.77	65.76	10.95	14.51	20.27
GC-MC	4.41	5.84	8.43	60.6	62.21	63.53	10.88	13.87	19.21
IGMC	3.5	4.82	7.18	56.89	59.13	60.46	9.21	12.2	17.27
NeuMF	3.46	4.54	6.45	54.42	56.39	57.79	8.87	11.38	15.89
NGCF	4.49	6.15	9.11	62.56	64.62	65.55	11.03	14.49	20.29
LightGCN	4.8	6.52	9.32	63.26	67.4	68.89	13.64	14.84	21.82
SGL	4.92	6.75	9.44	63.11	67.52	68.92	13.78	15.12	22.13
SCL-ND	5.02	6.88	9.83	64.56	67.48	68.91	13.92	15.43	22.31
SCL-ED	5.04	6.84	9.61	63.15	67.45	68.88	13.84	13.27	22.16
SCL-NR	5.13	6.92	9.95	64.67	67.42	68.92	13.97	15.51	22.43

As can be seen from the table, our proposed SCL learning paradigm is more effective on ML-100K data, and has achieved significant performance improvements in all three indicators. In addition, comparing different data augmentation method, it can be seen that NR is significantly effective. This data augmentation method can effectively improve the diversity of recommendation results, thereby improving recommendation performance.

5 Conclusion

We have only conducted experiments on small datasets at present, and we will conduct more adequate experiments on large datasets such as gowalla in the future. In addition, we will further conduct ablation experiments on SCL to illustrate its performance. We also hope to further analyze the principle of SCL.

References

- [1] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, pages 191–198, 2016.
- [2] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. Deep interest evolution network for click-through rate prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 5941–5948, 2019.
- [3] Qi Pi, Weijie Bian, Guorui Zhou, Xiaoqiang Zhu, and Kun Gai. Practice on long sequential user behavior modeling for click-through rate prediction. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2671–2679, 2019.
- [4] Mihajlo Grbovic and Haibin Cheng. Real-time personalization using embeddings for search ranking at airbnb. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 311–320, 2018.
- [5] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434, 2008.
- [6] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pages 173–182, 2017.
- [7] Travis Ebisu, Bin Shen, and Yi Fang. Collaborative memory network for recommendation systems. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 515–524, 2018.
- [8] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. Variational autoencoders for collaborative filtering. In *Proceedings of the 2018 world wide web conference*, pages 689–698, 2018.
- [9] Antonio Hernando, Jesús Bobadilla, and Fernando Ortega. A non negative matrix factorization for collaborative filtering recommender systems based on a bayesian probabilistic model. *Knowledge-Based Systems*, 97:188–202, 2016.
- [10] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618*, 2012.
- [11] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. Latent relational metric learning via memory-based attention for collaborative ranking. In *Proceedings of the 2018 World Wide Web Conference*, pages 729–739, 2018.
- [12] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*, pages 165–174, 2019.
- [13] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng 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*, pages 639–648, 2020.
- [14] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. Self-supervised graph learning for recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 726–735, 2021.
- [15] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.
- [16] Jianing Sun, Yingxue Zhang, Wei Guo, Huifeng Guo, Ruiming Tang, Xiuqiang He, Chen Ma, and Mark Coates. Neighbor interaction aware graph convolution networks for recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1289–1298, 2020.
- [17] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.
- [18] William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 1025–1035, 2017.
- [19] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

- [20] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.
- [21] Hongmin Zhu, Fuli Feng, Xiangnan He, Xiang Wang, Yan Li, Kai Zheng, and Yongdong Zhang. Bilinear graph neural network with neighbor interactions. *arXiv preprint arXiv:2002.03575*, 2020.
- [22] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.
- [23] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 335–344, 2017.
- [24] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Thirty-Second AAAI conference on artificial intelligence*, 2018.
- [25] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.
- [26] Khoi Nguyen, Yen Nguyen, and Bao Le. Semi-supervising learning, transfer learning, and knowledge distillation with simclr. *arXiv preprint arXiv:2108.00587*, 2021.
- [27] Vikas Verma, Thang Luong, Kenji Kawaguchi, Hieu Pham, and Quoc Le. Towards domain-agnostic contrastive learning. In *International Conference on Machine Learning*, pages 10530–10541. PMLR, 2021.
- [28] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems*, pages 1857–1865, 2016.
- [29] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018.
- [30] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.
- [31] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.
- [32] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.
- [33] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, pages 1067–1077, 2015.
- [34] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 974–983, 2018.
- [35] Rianne van den Berg, Thomas N Kipf, and Max Welling. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263*, 2017.
- [36] Muhan Zhang and Yixin Chen. Inductive matrix completion based on graph neural networks. *arXiv preprint arXiv:1904.12058*, 2019.