Generative-Contrastive Graph Learning for Recommendation

Yonghui Yang
Key Laboratory of Knowledge Engineering with Big Data, Hefei University of Technology
yyh.hfut@gmail.com

Zhengwei Wu
Ant Group
zejun.wzw@antfin.com

Le Wu*
Key Laboratory of Knowledge Engineering with Big Data, Hefei University of Technology
lewu.ustc@gmail.com

Kun Zhang
Key Laboratory of Knowledge Engineering with Big Data, Hefei University of Technology
zhang1028kun@gmail.com

Richang Hong
Key Laboratory of Knowledge Engineering with Big Data, Hefei University of Technology
hongrc.hfut@gmail.com

Zhiqiang Zhang
Ant Group
lingyao.zzq@antfin.com

Jun Zhou
Ant Group
jun.zhoujun@antfin.com

Meng Wang
Key Laboratory of Knowledge Engineering with Big Data, Hefei University of Technology
Institute of Artificial Intelligence, Hefei Comprehensive National Science Center
eric.mengwang@gmail.com

ABSTRACT

By treating users' interactions as a user-item graph, graph learning models have been widely deployed in Collaborative Filtering (CF) based recommendation. Recently, researchers have introduced Graph Contrastive Learning (GCL) techniques into CF to alleviate the sparse supervision issue, which first constructs contrastive views by data augmentations and then provides self-supervised signals by maximizing the mutual information between contrastive views. Despite the effectiveness, we argue that current GCL-based recommendation models are still limited as current data augmentation techniques, either structure augmentation or feature augmentation. First, structure augmentation randomly dropout nodes or edges, which is easy to destroy the intrinsic nature of the user-item graph. Second, feature augmentation imposes the same scale noise augmentation on each node, which neglects the unique characteristics of nodes on the graph.

To tackle the above limitations, we propose a novel *Variational Graph Generative-Contrastive Learning (VGCL)* framework for recommendation. Specifically, we leverage variational graph reconstruction to estimate a Gaussian distribution of each node, then generate multiple contrastive views through multiple samplings from the estimated distributions, which builds a bridge between generative and contrastive learning. The generated contrastive views can well reconstruct the input graph without information distortion. Besides, the estimated variances are tailored to each node, which regulates the scale of contrastive loss for each node on optimization. Considering the similarity of the estimated distributions, we propose a cluster-aware twofold contrastive learning, a node-level to encourage consistency of a node's contrastive views and a cluster-level to encourage consistency of nodes in a cluster. Finally, extensive experimental results on three public datasets clearly demonstrate the effectiveness of the proposed model.

CCS CONCEPTS

• **Information systems** → **Collaborative filtering**; **Recommender systems**.

KEYWORDS

Collaborative Filtering, Recommendation, Generative Learning, Graph Contrastive Learning

ACM Reference Format:
Yonghui Yang, Zhengwei Wu, Le Wu, Kun Zhang, Richang Hong, Zhiqiang Zhang, Jun Zhou, and Meng Wang. 2023. Generative-Contrastive Graph Learning for Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23), July 23–27, 2023, Taipei, Taiwan.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3539618.3591691

1 INTRODUCTION

CF-based recommendation relies on the observed user-item interactions to learn user and item embeddings for personalized preference

This work is done when Yonghui Yang works as an intern at Ant Group.
Le Wu is the Corresponding author.

SIGIR '23, July 23–27, 2023, Taipei, Taiwan

ACM ISBN 978-1-4503-9408-6/23/07…$15.00
https://doi.org/10.1145/3539618.3591691

prediction, and has been pervasive in real-world applications [30]. Early works leverage the matrix factorization technique to obtain user and item embeddings, and then compute users' preferences by inner product [22, 23] or neural networks [8]. As users' interactions can be naturally formulated as a user-item graph, borrowing the success of Graph Neural Networks (GNNs), graph-based CF models have been widely studied with superior performances [2, 7, 38]. These models iteratively propagate the neighborhood information for embedding updates, such that the higher-order collaborative signals can be incorporated for better user and item embedding learning.

Despite the effectiveness, graph-based CF models suffer from the sparse supervision issue for model learning. As an alternative, self-supervised learning leverages the input data itself as the supervision signal and has attracted many researchers [14, 42]. Among all self-supervised learning, contrastive learning is a popular paradigm that constructs data augmentations to teach the model to compare similar data pairs, and has shown competitive performance in computer vision, natural language processing, graph mining, and so on [14, 21]. Some recent studies have introduced contrastive learning in graph-based CF [20, 39, 51]. In addition to the supervised recommendation task, GCL-based models first construct multiple contrastive views through data augmentation, and then maximize the mutual information to encourage the consistency of different views. Existing GCL-based CF methods can be classified into two categories: structure augmentation and feature augmentation. Specifically, structure augmentation randomly dropout graph nodes or edges to obtain subgraph structure, and then feeds the augmented graphs into an encoder for contrastive representations [39]. Feature augmentation adds random noises to node embeddings as contrastive views [51]. These GCL-based CF models learn self-supervised signals based on data augmentation and significantly improve recommendation performances.

Although data augmentation is the key to the performance of GCL-based CF models, we argue that current solutions are still limited by current data augmentation strategies, either structure augmentation or feature augmentation. Firstly, structure augmentation randomly dropout nodes or edges, which is easy to destroy the intrinsic nature of the input graph. The reason is that all nodes are connected on the graph and don't satisfy the IID assumption. Secondly, feature augmentation adds the same scale noise to each node, which neglects the unique characteristics of nodes on the graph. In real-world recommender systems, different users (items) have different characteristics, and the data augmentation techniques should be tailored to each user. E.g., some users have more item links in the user-item graph, which contains more supervised signals compared to users with only very few links. How to better exploit the user-item graph structure to design more sophisticated contrastive view construction techniques is still open.

In this paper, we exploit the potential of the generative model to facilitate contrastive view generation without data augmentation. Specifically, we propose a *Variational Graph Generative-Contrastive Learning (VGCL)* framework for recommendation. Instead of data augmentation, we leverage variational graph inference [17] to estimate a Gaussian distribution of each node, then generate multiple contrastive views through multiple samplings from the estimated distributions. As such, we build a bridge between the generative and contrastive learning models for recommendation. The generated contrastive views can well reconstruct the input graph without information distortion. Besides, the estimated variances are tailored to each node, which can adaptively regulate the scale of contrastive loss of each node for optimization. We consider that similar nodes are closer in the representation space, and then propose cluster-aware contrastive learning with twofold contrastive objectives. The first one is a node-level contrastive loss that encourages the consistency of each node's multiple views. The second one is a cluster-level contrastive loss that encourages the consistency of different nodes in a cluster, with the cluster learned from the estimated distributions of nodes. The major contributions of this paper are summarized as follows:

- We introduce a novel generative-contrastive graph learning paradigm from the perspective of better contrastive view construction, and propose a novel *Variational Graph Generative-Contrastive Learning (VGCL)* framework for recommendation.
- We leverage variational graph reconstruction to generate contrastive views, and a design cluster-aware twofold contrastive learning module, such that the self-supervised signals can be better mined at different scales for GCL-based recommendation.
- Extensive experiments on three public datasets clearly show the effectiveness of the proposed framework, our *VGCL* consistently outperforms all baselines.

2 PRELIMINARIES

2.1 Graph based Collaborative Filtering

In fundamental collaborative filtering, there are two kinds of entities: a userset U ($|U| = M$) and an itemset V ($|V| = N$). Considering the recommendation scenarios with implicit feedback, we use matrix $\mathbf{R} \in \mathbb{R}^{M \times N}$ to describe user-item interactions, where each element $\mathbf{r}_{ai} = 1$ if user a interacted with item i, otherwise $\mathbf{r}_{ai} = 0$. Graph-based CF methods [2, 7, 38] formulate the available data as a user-item bipartite graph $\mathcal{G} = \{U \cup V, \mathbf{A}\}$, where $U \cup V$ denotes the set of nodes, and $\mathbf{A}$ is the adjacent matrix defined as follows:

$$\mathbf{A} = \begin{bmatrix} \mathbf{0}^{M \times M} & \mathbf{R} \\ \mathbf{R}^T & \mathbf{0}^{N \times N} \end{bmatrix}. \tag{1}$$

Given the initialized node embeddings $\mathbf{E}^0$, graph-based CF methods update node embeddings through multiple graph convolutions:

$$\mathbf{E}^l = \mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}} \mathbf{E}^{l-1}, \tag{2}$$

where $\mathbf{D}$ is the degree matrix of graph $\mathcal{G}$, $\mathbf{E}^l$ and $\mathbf{E}^{l-1}$ denote node embeddings in l^{th} and $(l-1)^{th}$ graph convolution layer, respectively. When stacking L graph convolution layers, the final node representations can be obtained with a readout operation:

$$\mathbf{E} = Readout(\mathbf{E}^0, \mathbf{E}^1, ..., \mathbf{E}^L). \tag{3}$$

The pairwise ranking [23] loss is adopted to optimize model parameters:

$$\mathcal{L}_{rec} = \sum_{a=0}^{M-1} \sum_{(i,j) \in D_a} -log\sigma(\hat{r}_{ai} - \hat{r}_{aj}) + \lambda ||\mathbf{E}^0||^2, \tag{4}$$

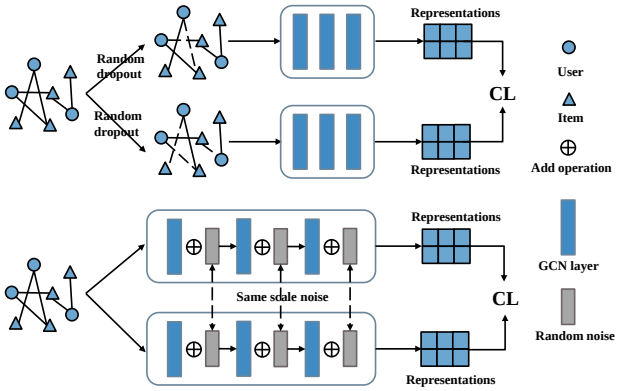


Figure 1: Graph contrastive learning paradigms with structure and feature data augmentation.

where $\sigma(\cdot)$ is the sigmoid activation function, λ is the regularization coefficient. $D_a = \{(i,j)|i \in R_a \wedge j \notin R_a\}$ denotes the pairwise training data for user a. R_a represents the item set that user a has interacted.

2.2 Graph Contrastive Learning for Recommendation

GCL usually as an auxiliary task to complement recommendation with self-supervised signals [20, 39, 51], which performs multi-task learning:

$$\mathcal{L} = \mathcal{L}_{rec} + \alpha \mathcal{L}_{cl}, \tag{5}$$

where α is a hyper-parameter that controls the contrastive task weight, $\mathcal{L}_{cl}$ is the typical InfoNCE loss function [3]:

$$\mathcal{L}_{cl} = \sum_{i \in \mathcal{B}} -log \frac{exp(\mathbf{e}_i'^T \mathbf{e}_i'' / \tau)}{\sum_{j \in \mathcal{B}} exp(\mathbf{e}_i'^T \mathbf{e}_j'' / \tau)}, \tag{6}$$

where $\mathcal{B}$ denote a batch users (items), τ is the contrastive temperature. For node i, e_i' and e_i'' denote the corresponding contrastive representations with L_2 normalization, the same as node j. This objective encourages consistency of contrastive representations for each node.

Revisiting GCL-based recommendation models from a data augmentation perspective, there are two popular strategies: structure augmentation [39] and feature augmentation [51]. As illustrated in the upper part of Figure1, structure augmentation randomly perturb graph structure to obtain two augmented views $\mathcal{G}', \mathcal{G}''$, then generate contrastive representations as follows:

$$\mathbf{E}' = \mathcal{E}(\mathcal{G}', \mathbf{E}^0), \mathbf{E}'' = \mathcal{E}(\mathcal{G}'', \mathbf{E}^0), \tag{7}$$

where $\mathcal{E}(\cdot)$ denotes graph encoder. Because nodes do not satisfy the IID assumption on the graph, random structure perturbation easy to destroys the intrinsic nature of the input graph, then can't fully make use of GCL for recommendation. Another is feature augmentation [51], which is illustrated in the lower part of Figure1. Feature augmentation adds random noises into node embeddings, then generate contrastive representations with GNNs:

$$\mathbf{E}' = \mathcal{E}(\mathbf{E}^0, \epsilon\delta'), \mathbf{E}'' = \mathcal{E}(\mathbf{E}^0, \epsilon\delta''), \tag{8}$$

where $\delta', \delta'' \sim U(0,1)$ are uniform noises, ϵ is the amplitude that controls noise scale. Although this noise-based augmentation is controllable and constrains the deviation, we argue that a fixed and generic ϵ is not generalized for nodes with unique characteristics. For example, user-item interactions usually perform the long-tail distribution, the head nodes have more supervision signals than tails, then a small ϵ maybe satisfy the tail nodes while not sufficient to the head nodes. The above flaws drive us to find a better graph augmentation that maintains graph information and is adaptive to each node.

3 METHODOLOGY

In this section, we present our proposed *Variational Graph Generative-Contrastive Learning (VGCL)* framework for recommendation. As shown in Figure 2, *VGCL* consists of two modules: a variational graph reconstruction module and a cluster-aware contrastive learning module. Specifically, we first use variational graph reconstruction to estimate the probability distribution of each node, then design cluster-aware twofold contrastive learning objectives to encourage the consistency of contrastive views which are generated by multiple samplings from the estimated distribution. Next, we introduce each component in detail.

3.1 Variational Graph Reconstruction

VAE Brief. Given the training data $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$, VAE assumes that each sample $\mathbf{x}_i$ is constructed from a generative process: $\mathbf{x} \sim p_\theta(\mathbf{x}|\mathbf{z})$. Thus, it's natural to maximize the likelihood function:

$$logp(\mathbf{x}) = log \int p_\theta(\mathbf{x}|\mathbf{z}) p(\mathbf{z}) d\mathbf{z}, \tag{9}$$

where $p(\mathbf{z})$ is the prior distribution of latent variable $\mathbf{z}$. However, it's intractable to compute Eq.(9) because we don't know all possible latent variables $\mathbf{z}$. Thus, VAE adopts a variational inference technique and uses an inference model $q_\phi(\mathbf{z}|\mathbf{x})$ to approximate the posterior distribution $p_\theta(\mathbf{x}|\mathbf{z})$. Then, VAE is optimized by minimizing the Evidence Lower Bound (ELBO) based objective:

$$\mathcal{L}_{ELBO} = -\mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})}[log(p_\theta(\mathbf{x}|\mathbf{z}))] + KL[q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})], \tag{10}$$

where $q_\phi(\mathbf{z}|\mathbf{x})$ and $p_\theta(\mathbf{x}|\mathbf{z})$ also denote the encoder and decoder which are parameterized by neural networks. $KL[q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})]$ is the Kullback-Leibler divergence between the approximate posterior $q_\phi(\mathbf{z}|\mathbf{x})$ and prior $p(\mathbf{z})$, which is used to constrain $q_\phi(\mathbf{z}|\mathbf{x})$ closer to the prior Gaussian distribution.

Graph Inference. Given the observed user-item interaction graph $\mathcal{G} = \{U \cup V, \mathbf{A}\}$, and initialized node embeddings $\mathbf{E}^0$. Graph inference aims to learn probability distributions $\mathbf{Z}$ which can reconstruct the input graph structure: $\hat{\mathbf{A}} \sim p_\theta(\mathbf{A}|\mathbf{Z})$. Same to VAE, we also adopt variational inference $q_\phi(\mathbf{Z}|\mathbf{A}, \mathbf{E}^0) = \prod_{i=0}^{M+N-1} q_\phi(\mathbf{z}_i|\mathbf{A}, \mathbf{E}^0)$ to approximate the posterior $p_\theta(\mathbf{A}|\mathbf{Z})$. To be specific, we encode each node i into a multi-variate Gaussian distribution $q_\phi(\mathbf{z}_i|\mathbf{A}, \mathbf{E}^0) = \mathcal{N}(\mathbf{z}_i|\mu_\phi(i), diag(\sigma_\phi^2(i)))$, where $\mu_\phi(i)$ and $\sigma_\phi^2(i)$ denote the mean and variance of node i's distribution, respectively. To better exploit high-order user-item graph structure, we adopt GNNs to estimate the parameters of node distributions:

$$\mu = GNN(\mathbf{A}, \mathbf{E}^0, \phi_\mu), \sigma = GNN(\mathbf{A}, \mathbf{E}^0, \phi_\sigma), \tag{11}$$

where ϕ_μ and ϕ_σ denote learnable parameters on graph inference. Following the previous research on graph-based collaborative filtering, we select LightGCN [7] as the encoder to deploy the above graph inference process. For each node i, the corresponding means

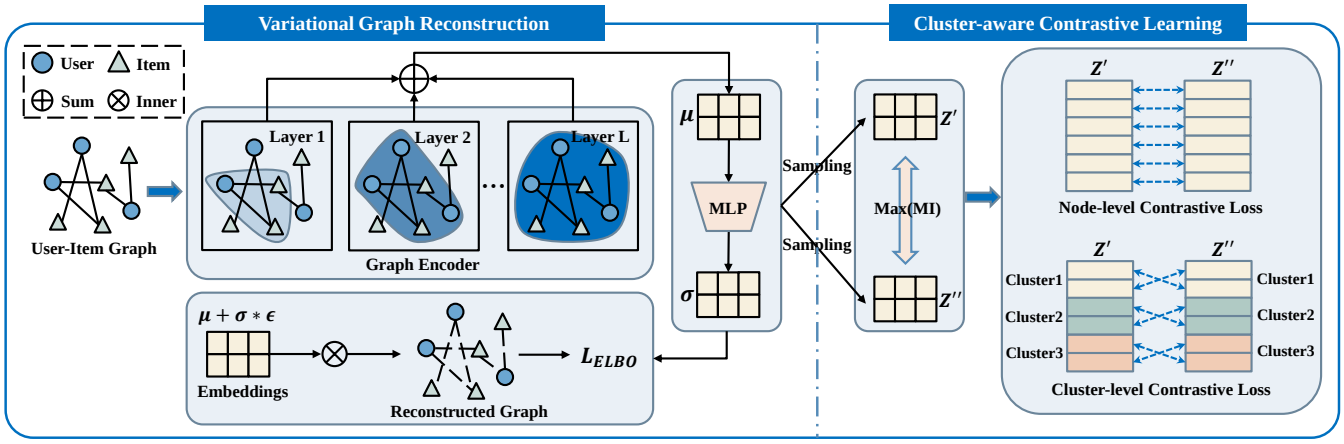


Figure 2: An Illustration of our proposed *Variational Graph Generative-Contrastive Learning (VGCL)* framework, which consists of a variational graph reconstruction module and a cluster-aware contrastive learning module. The variational graph reconstruction module generates contrastive views by multiple samplings from the estimated distributions. The Cluster-aware contrastive learning module provides self-supervised signals, which include node-level and cluster-level contrastive objectives.

are updated as follows:

$$\mu_i^l = \sum_{j \in \mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_i|}\sqrt{|\mathcal{N}_j|}} \mu_i^{l-1}, \tag{12}$$

where μ_i^l and μ_i^{l-1} are corresponding means on l^{th} and $(l-1)^{th}$ graph convolution layer, $\mathcal{N}_i$ and $\mathcal{N}_j$ denote the connected neighbors for node i and node j. We initialize the means $\mu^0 = \mathbf{E}^0$. When stacking L graph convolution layers, we have $L+1$ outputs $[\mu^0, \mu^1, ..., \mu^L]$, then we fuse all layers' outputs and compute the means and variances as follows:

$$\mu = \frac{1}{L} \sum_{l=1}^{L} \mu^l, \sigma = MLP(\mu), \tag{13}$$

where the variances are learned from an MLP, which feeds the means as the input. In practice, we find that one-layer MLP achieves the best performance, then $\sigma = exp(\mu \mathbf{W} + \mathbf{b})$, where $\mathbf{W} \in \mathbb{R}^{d \times d}$ and $\mathbf{b} \in \mathbb{R}^d$ are two learnable parameters. After obtaining the mean and variance of the approximate posterior, we generate the latent representation $\mathbf{z}_i$ by sampling from $\mathcal{N}(\mu_i, \sigma_i^2)$. However, it can not be directed optimized because the sampling process is non-differentiable. We employ the reparameterization trick instead of the sampling process [16]:

$$\mathbf{z}_i = \mu_i + \sigma_i \cdot \varepsilon, \tag{14}$$

where $\varepsilon \sim \mathcal{N}(0, \mathbf{I})$ is a normal Gaussian noise.

Graph Generation. After estimating the probability distribution of the latent variables $\mathbf{Z}$, the objective of graph generation is to reconstruct the original user-item graph:

$$p(\mathbf{A}|\mathbf{Z}) = \prod_{i=0}^{M+N-1} \prod_{j=0}^{M+N-1} p(\mathbf{A}_{ij}|\mathbf{z}_i, \mathbf{z}_j). \tag{15}$$

There are many choices to realize the graph generation process, such as inner product, factorization machine, and neural networks. As suggested in [17], we use an inner product to compute the propensity score that node i connected with node j:

$$p(\mathbf{A}_{ij} = 1|\mathbf{z}_i, \mathbf{z}_j) = \sigma(\mathbf{z}_i^T \mathbf{z}_j), \tag{16}$$

where $\sigma(\cdot)$ is the sigmoid function.

3.2 Cluster-aware Contrastive Learning

Contrastive View Construction. Given the estimated probability distribution of latent representation $\mathbf{Z} \sim \mathcal{N}(\mu, \sigma^2)$, we introduce a novel contrastive learning paradigm based on the estimated distribution. Different from previous GCL-based recommendation methods [39, 51], we construct contrastive views through multiple samplings from the estimated distribution instead of data augmentation. Specifically, for each node i, we generate contrastive representations $\mathbf{z}'$ and $\mathbf{z}''$ as follows:

$$\mathbf{z}_i' = \mu_i + \sigma_i \cdot \varepsilon', \tag{17}$$

$$\mathbf{z}_i'' = \mu_i + \sigma_i \cdot \varepsilon'', \tag{18}$$

where $\varepsilon', \varepsilon'' \sim \mathcal{N}(0, \mathbf{I})$ are two random normal noise. Compared to structure or feature augmentations, our method is more efficient and effective for contrastive view construction. Firstly, all contrastive representations are sampled from the estimated distributions, which can well reconstruct the input graph without any information distortion. Secondly, the estimated variances are tailored to each node, which can be adaptive to regulate the scale of contrastive loss.

Node-level Contrastive Loss. After constructing contrastive views of each node, we maximize the mutual information to provide self-supervised signals to improve recommendation performance. Considering that similar nodes are closer in the representation, we propose cluster-aware twofold contrastive objectives for optimization: a node-level contrastive loss and a cluster-level contrastive loss. Among them, node-level contrastive loss encourages consistency of contrastive views for each node, and cluster-level contrastive loss encourages consistency of contrastive views of nodes in a cluster. The objective of node-level contrastive learning is $\mathcal{L}_N = \mathcal{L}_N^U + \mathcal{L}_N^V$, where $\mathcal{L}_N^U$ and $\mathcal{L}_N^V$ denote user side and item side losses:

$$\mathcal{L}_N^U = \sum_{a \in \mathcal{B}_u} -log \frac{exp({\mathbf{z}_a'}^T \mathbf{z}_a''/\tau_1)}{\sum_{b \in \mathcal{B}_u} exp({\mathbf{z}_a'}^T \mathbf{z}_b''/\tau_1)}, \tag{19}$$

$$\mathcal{L}_N^I = \sum_{i \in \mathcal{B}_i} -log \frac{exp({\mathbf{z}_i'}^T \mathbf{z}_i''/\tau_1)}{\sum_{j \in \mathcal{B}_i} exp({\mathbf{z}_i'}^T \mathbf{z}_j''/\tau_1)}, \tag{20}$$

where τ_1 is the contrastive temperature, $\mathcal{B}_u$ and $\mathcal{B}_i$ denote users and items in a batch training data.

Cluster-level Contrastive Loss. Considering the similarity of the estimated distributions of nodes, we design cluster-level contrastive loss to further distinguish the positive and negative contrastive pairs in batch training data. Overall, our aim is to maximize the consistency of node pairs with the same cluster and minimize the consistency of node pairs with different clusters. Suppose there are K_u user prototypes $\mathbf{C}^u \in \mathbb{R}^{d\times K_u}$ and K_i item cluster prototypes $\mathbf{C}^i \in \mathbb{R}^{d\times K_i}$, we use $p(c_k^u|z_a)$ to denote the conditional probability that user a belongs to k^{th} user cluster, and $p(c_h^i|z_i)$ denote the conditional probability that item i belongs to h^{th} item cluster. Given the estimated distributions as input, we implement the clustering process by the K-Means algorithm [6]. Then, we compute the probability that two users (items) are assigned to the same prototype:

$$p(a,b) = \sum_{k=0}^{K_u-1} p(\mathbf{c}_k^u|\mathbf{z}_a)p(\mathbf{c}_k^u|\mathbf{z}_b), \tag{21}$$

$$p(i,j) = \sum_{h=0}^{K_i-1} p(\mathbf{c}_h^i|\mathbf{z}_i)p(\mathbf{c}_h^i|\mathbf{z}_j), \tag{22}$$

where $p(a,b)$ denote the probability that user a and user b belong to the same cluster, and $p(i,j)$ denote the probability that item i and item j belong to the same cluster. Next, we present the cluster-level contrastive loss $\mathcal{L}_C = \mathcal{L}_C^U + \mathcal{L}_C^I$, where $\mathcal{L}_C^U$ and $\mathcal{L}_C^I$ denote user side and item side losses:

$$\mathcal{L}_C^U = \sum_{a\in\mathcal{B}_u} \frac{-1}{SP(a)} log\left(\frac{\sum_{b\in\mathcal{B}_u,b!=a} p(a,b)exp(\mathbf{z'}_a^T\mathbf{z''}_b/\tau_2)}{\sum_{b\in\mathcal{B}_u,b!=a} exp(\mathbf{z'}_a^T\mathbf{z''}_b/\tau_2)}\right), \tag{23}$$

$$\mathcal{L}_C^I = \sum_{i\in\mathcal{B}_i} \frac{-1}{SP(i)} log\left(\frac{\sum_{j\in\mathcal{B}_i,j!=i} p(i,j)exp(\mathbf{z'}_i^T\mathbf{z''}_j/\tau_2)}{\sum_{j\in\mathcal{B}_i,j!=i} exp(\mathbf{z'}_i^T\mathbf{z''}_j/\tau_2)}\right), \tag{24}$$

where $SP(a) = \sum_{b\in\mathcal{B}_u,b!=a} p(a,b)$ and $SP(i) = \sum_{j\in\mathcal{B}_i,j!=i} p(i,j)$, τ_2 is the temperature to control the mining scale of hard negatives. The final contrastive loss is the weighted sum of the node-level loss and the cluster-level contrastive loss:

$$\mathcal{L}_{cl} = \mathcal{L}_N + \gamma\mathcal{L}_C, \tag{25}$$

where γ is the coefficient to balance two level contrastive losses.

3.3 Model Optimization.

For the variational graph reconstruction part, we optimize the parameters of graph inference and graph generation with ELBO:

$$\mathcal{L}_{ELBO} = -\mathbb{E}_{\mathbf{Z}\sim q_\phi(\mathbf{Z}|\mathbf{A},\mathbf{E}^0)}[log(p_\theta(\mathbf{A}|\mathbf{Z}))] + KL[q_\phi(\mathbf{Z}|\mathbf{A},\mathbf{E}^0)||p(\mathbf{Z})]. \tag{26}$$

Among them, the first term is the reconstruction error between the original graph and the generated graph. We employ a pairwise learning strategy to minimize the reconstruction error:

$$\mathbb{E}_{\mathbf{Z}\sim q_\phi(\mathbf{Z}|\mathbf{A},\mathbf{E}^0)}[log(p_\theta(\mathbf{A}|\mathbf{Z}))] = \sum_{a=0}^{M-1}\sum_{(i,j)\in D_a} -log\sigma(\hat{r}_{ai} - \hat{r}_{aj})), \tag{27}$$

$D_a = \{(i,j)|i \in R_a \wedge j \notin R_a\}$ denotes the pairwise training data for user a. R_a represents the item set that user a has interacted. Overall, we optimize the proposed *VGCL* with a multi-task learning framework:

$$min\mathcal{L} = \mathcal{L}_{ELBO} + \alpha\mathcal{L}_{cl} + \lambda||\mathbf{E}^0||^2, \tag{28}$$

Algorithm 1: The Algorithm of *VGCL*

Input: user-item bipartite graph $\mathcal{G}$;
Output: Parameters $\Theta_{GNN} = \mathbf{E}^0$ and $\Theta_{MLP} = [\mathbf{W}, \mathbf{b}]$;

```
Randomly initialize parameters Θ_GNN and Θ_MLP ;
while not converged do
    Sample a batch of training data;
    Calculate graph inference parameters μ and σ (Eq.(12) to Eq.(13));
    Estimate node distribution Z by parameterization (Eq.(14));
    Generate contrastive instances Z' and Z'' by multiple samplings (Eq.(17), Eq.(18));
    Compute prototypes C_u and C_v based on K-Means clustering algorithm;
    Compute node-level contrastive loss L_N (Eq.(19));
    Compute cluster-level contrastive loss L_C (Eq.(23), Eq.(24));
    Compute variational graph reconstruction loss L_ELBO (Eq.(26));
    Update all parameters according to (Eq.(28));
end while
Return Θ_GNN = E^0 and Θ_MLP = [W, b].
```

where α is the balance parameter of contrastive loss, and λ is the regularization coefficient. After the model training process, we use Eq.(16) to predict the unknown preferences for the recommendation.

3.4 Model Analysis

Space Complexity. As shown in Algorithm 1, the model parameters are composed of two parts: node embeddings $\mathbf{E}^0$ and MLP parameters $\mathbf{W}, \mathbf{b}$. Compared to traditional embedding-based collaborative filtering, the additional parameters only have $\mathbf{W}, \mathbf{b}$ which are shared among all nodes. So the additional storage space is very small and can be neglected.

Time Complexity. We compare the time complexity of *VGCL* with other GCL-based recommendation methods based on data augmentation. Let $|E|$ denote the edge number of the graph, d be the embedding size, and S denote the average neighbor number. For the graph convolution part, *VGCL* costs $O(2|E|dS)$, where $2|E|$ denotes the number of non-zero elements on the adjacent matrix. However, SGL and SimGCL need to repeat graph convolution three times, which generates main embeddings for recommendation and two auxiliary embeddings for contrastive learning. Therefore, SGL and SimGCL all cost $O(6|E|dS)$ while *VGCL* only need $O(2|E|dS)$. For the contrastive learning part, *VGCL* additionally has a clustering process, we implement the K-means clustering algorithm with Faiss-GPU [1], and the time cost can be neglected compared to model learning in practice. Therefore, *VGCL* is more time-efficient than current GCL-based recommendation methods based on data augmentation.

[1] https://faiss.ai/

Table 1: The statistics of three datasets.

Datasets	Users	Items	Interactions	Density
Douban-Book	13,024	22,347	792,062	0.272%
Dianping	59,426	10,224	934,334	0.154%
Movielens-25M	92,901	8,826	2,605,952	0.318%

4 EXPERIMENTS

4.1 Experimental Settings

4.1.1 Datasets. To compare the recommendation performance of our *VGCL* with other state-of-the-art models, we select three benchmarks to conduct the empirical analysis: Douban-Book [51], Dianping [41] and Movielens-25M [5]. For Movielens-25M, we convert ratings equal to 5 as positive feedback, and other ratings as negative feedback. We filter users with less than 10 interactions for all datasets, and randomly sample 80% interactions as training data, and the remaining 20% as test data. The statistics of three datasets are summarized in Table 1.

4.1.2 Baselines and Evaluation Metrics. We compare our model with the following baselines, including matrix factorization based method: BPR-MF [23], graph based method: LightGCN [7], VAE based methods: Multi-VAE [19], CVGA [52], and graph contrastive learning based methods: SGL [39], NCL [20], SimGCL [51].

We employ two widely used metrics: Recall@N and NDCG@N to evaluate all recommendation models. Specifically, Recall@N measures the percentage of recalled items on the Top-N ranking list, while NDCG@N further assigns higher scores to the top-ranked items. To avoid selection bias in the test stage, we use the full-ranking strategy [53] that views all non-interacted items as candidates. All metrics are reported with average values with 5 times repeated experiments.

4.1.3 Parameter Settings. We implement our *VGCL* model and all baselines with Tensorflow[2]. We initialize all models parameter with a Gaussian distribution with a mean value of 0 and a standard variance of 0.01, embedding size is fixed to 64. We use Adam as the optimizer for model optimization, and the learning rate is 0.001. The batch size is 2048 for the Douban-Book and Dianping datasets and 4096 for the Movielens-25M dataset. For our *VGCL* model, we turn the contrastive temperature τ in $[0.10, 0.25]$, contrastive regularization coefficient λ in $[0.01, 0.05, 0.1, 0.2, 0.5, 1.0]$, and clustering number k_1, k_2 in $[100, 1000]$. Besides, we carefully search the best parameter of γ, and find *VGCL* achieves the best performance when $\gamma = 0.4$ on Douban-Book, $\gamma = 0.5$ on Dianping dataset, and $\gamma = 1.0$ on Movielens-25M dataset. As we employ the pairwise learning strategy for graph reconstruction, we randomly select one unobserved item as a candidate negative sample to compose triple data for model training. For all baselines, we search the parameters carefully for fair comparisons. We repeat all experiments 5 times and report the average results.

4.2 Overall Performance Comparisons

As shown in Table 2, we compare our model with other baselines on three datasets. We have the following observations:

- Our proposed *VGCL* consistently outperforms all baselines under different settings. Specifically, *VGCL* improves LightGCN $w.r.t$ NDCG@20 by 28.17%, 14.70% and 8.61% on Douban-Book, Dianping and Movielens-25M dataset, respectively. Compared to the strongest baseline (SimGCL), *VGCL* also achieves better performance, e.g., about 6.36% performance improvement of NDCG@20 on the Douban-Book dataset. Besides, we find that *VGCL* achieves higher improvements on the small-length ranking task, which is more suitable for real-world recommendation scenarios. Extensive empirical studies verify the effectiveness of the proposed *VGCL* , which benefits from combining the strength of generative and contrastive graph learning for recommendation.
- Graph-based methods achieve better performance than their counterparts, which shows the superiority that capturing users' preferences by modeling high-order user-item graph structure. To be specific, LightGCN always outperforms BPR and CVGA consistently outperforms Multi-VAE, which proves that graph learning can effectively capture the high-order user-item interaction signals to improve recommendation performance, whether in embedding-based or VAE-based recommendation methods.
- All GCL-based methods (SGL, NCL, SimGCL) significantly improve LightGCN on three datasets. It verifies the effectiveness of incorporating self-supervised learning into collaborative filtering. SimGCL achieves the best performance among these baselines, demonstrating that feature augmentation is more suitable for collaborative filtering than structure augmentation, which can maintain sufficient invariants of the original graph. It's worth noting that, our method also can be regarded as feature augmentation, but we rely on multiple samplings from the estimated distribution and the scales of augmentations are adaptive to different nodes. Therefore, *VGCL* achieves better performance compared to SimGCL.

4.3 Ablation Study

To exploit the effectiveness of each component of the proposed *VGCL* , we conduct the ablation study on three datasets. As shown in Table 3, we compare *VGCL* and corresponding variants on Top-20 recommendation performances. VGCL-w/o C denotes that remove the cluster-level contrastive loss of *VGCL* , we only use the general node-level contrastive loss. VGCL-w/o V denotes that remove the variational graph reconstruction part of *VGCL* , then we use feature augmentation the same as SimGCL to generate contrastive views. From Table 3, we observe that VGCL-C consistently improves SimGCL on three datasets, which verifies that the proposed variational graph reconstruction module can provide better contrastive views for contrastive learning. Besides, VGCL-V also shows better performances than SimGCL, it demonstrates the effectiveness of cluster-aware contrastive pair sampling on contrastive learning. Finally, *VGCL* consistently outperforms two variants, demonstrating the effectiveness of combining the variational graph reconstruction and cluster-aware sampling strategy. Based on the above analysis, we can draw the conclusion that variational graph reconstruction can provide better contrastive views than simple data augmentation and cluster-aware sampling is better than random sampling

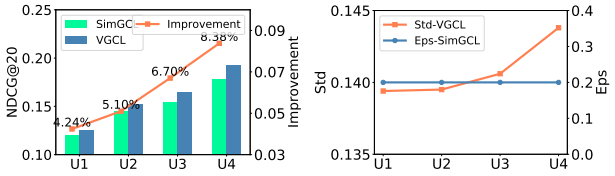
[2]https://www.tensorflow.org

Table 2: Recommendation performances on three datasets. The best-performing model on each dataset and metrics are highlighted in bold, and the second-best model is underlined.

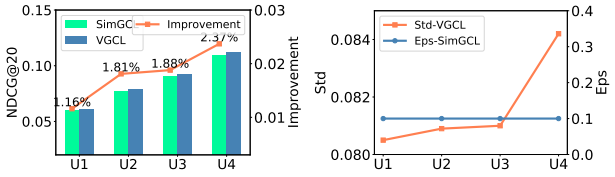
Models	Douban-Book				Dianping				Movielens-25M			
	R@10	N@10	R@20	N@20	R@10	N@10	R@20	N@20	R@10	N@10	R@20	N@20
BPR-MF	0.0869	0.0949	0.1296	0.1045	0.0572	0.0443	0.0934	0.0557	0.2152	0.2011	0.3163	0.2343
LightGCN	0.1042	0.1195	0.1516	0.1278	0.0679	0.0536	0.1076	0.0660	0.2258	0.2192	0.3263	0.2509
Multi-VAE	0.0941	0.1073	0.1376	0.1155	0.0645	0.0508	0.1046	0.0632	0.2188	0.2101	0.3185	0.2418
CVGA	0.1058	0.1305	0.1492	0.1359	0.0719	0.0562	0.1128	0.0690	0.2390	0.2306	0.3454	0.2641
SGL-ED	0.1103	0.1357	0.1551	0.1419	0.0719	0.0560	0.1111	0.0686	0.2298	0.2239	0.3274	0.2541
NCL	0.1121	0.1377	0.1576	0.1439	0.0727	0.0571	0.1124	0.0701	0.2281	0.2222	0.3274	0.2531
SimGCL	0.1218	0.1470	0.1731	0.1540	0.0768	0.0606	0.1208	0.0743	0.2428	0.2356	0.3491	0.2690
VGCL	**0.1283**	**0.1564**	**0.1829**	**0.1638**	**0.0778**	**0.0616**	**0.1234**	**0.0757**	**0.2463**	**0.2400**	**0.3507**	**0.2725**

Table 3: Ablation study of VGCL, VGCL-w/o C denotes without cluster-level contrastive loss and VGCL-w/o V denotes without the variational graph reconstruction part.

Models	Douban-Book		Dianping		Movielens-25M	
	R@20	N@20	R@20	N@20	R@20	N@20
LightGCN	0.1512(-)	0.1271(-)	0.1076(-)	0.0660(-)	0.3263(-)	0.2509(-)
SimGCL	0.1731(14.48%)	0.1540(+21.16%)	0.1208(+12.27%)	0.0743(+12.58%)	0.3491(+6.99%)	0.2690(+7.21%)
VGCL-w/o C	0.1776(+17.46%)	0.1575(+23.92%)	0.1222(+13.57%)	0.0750(+13.64%)	0.3477(+6.56%)	0.2705(+7.81%)
VGCL-w/o V	0.1722(+13.89%)	0.1547(+21.72%)	0.1218(+13.20%)	0.0746(+13.03%)	0.3493(+7.05%)	0.2702(+7.69%)
VGCL	**0.1829(+20.97%)**	**0.1638(+28.87%)**	**0.1233(+14.59%)**	**0.0756(+14.55%)**	**0.3507(+7.48%)**	**0.2725(+8.61%)**



(a) Douban-Book Dataset



(b) Dianping Dataset

Figure 3: Performance comparisons under different user groups.

for contrastive learning. All of the proposed modules are beneficial to GCL-based recommendations.

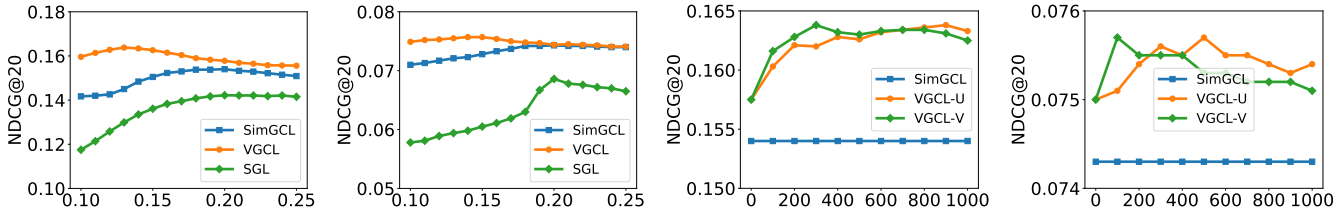
4.4 Investigation of the Estimated Distribution

As we introduced in methodology, our proposed *VGCL* can adaptively learn variances for various nodes. To investigate the effect of personalized variances, we conduct comparisons of different user groups. Specifically, we first split all users into 4 groups according to their interactions, then analyze recommendation performances under different user groups. Figure3 illustrates NDCG@20 values of various groups on Douban-Book and Dianping datasets. We observe that all models show better performances in the denser user group, which conforms to the intuition of CF. Besides, our proposed *VGCL* achieves better performances on all user groups, demonstrating that *VGCL* is general to users with different interactions. Further, we plot the relative improvements that *VGCL* over SimGCL on Figure3. We find that *VGCL* achieves a more significant improvement in the denser groups, e.g., 8.4% improvement in U4 while 4.2% improvement in U1 on the Douban-Book dataset. To exploit this phenomenon, we compared the standard variances of the estimated distribution of different users. From the right part of Figure3, we can observe that the inferred standard variances vary from each group, and increase by group ID. Compared with SimGCL which set fixed eps (noise scale) for all users, our method can learn personalized contrastive scales for different users. What's more, *VGCL* can adaptively learn larger variances to those users with amounts of interactions, it's important to provide sufficient self-supervised signals to improve recommendation performance. Experimental results effectively demonstrate the effectiveness of our proposed adaptive contrastive objectives.

4.5 Hyper-Parameter Sensitivities

In this part, we analyze the impact of hyper-parameters in *VGCL* . We first exploit the effect of temperature τ, which plays an important role in contrastive learning. Next, we investigate the influence of graph inference layer L. Finally, we study the impact of clustering prototype numbers K_u, K_v and contrastive loss weights α and γ.

Effect of Graph Inference Layer L. To exploit the effect of different graph inference layers, we search the parameter L in the range of $\{1, 2, 3, 4\}$. As shown in Table 4, we compare experimental results of different graph inference layers on Douban-Book and

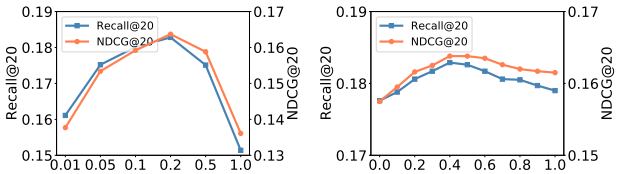


(a) τ on Douban-Book (b) τ on Dianping (c) K_u, K_i on Douban-Book (d) K_u, K_i on Dianping

Figure 4: Performance comparisons w.r.t different temperature τ, prototype number K_u and K_i.

Table 4: Performance on different graph inference layer L.

Layers	Douban-Book		Dianping	
	Recall@20	NDCG@20	Recall@20	NDCG@20
L=1	0.1750	0.1555	0.1197	0.0733
L=2	**0.1829**	**0.1638**	0.1229	0.0751
L=3	0.1808	0.1618	**0.1234**	**0.0757**
L=4	0.1793	0.1605	0.1233	0.0752



(a) α on Douban-Book (b) γ on Douban-Book

Figure 5: Performance comparisons under different contrastie loss weights α and γ.

Dianping datasets. From Table 4, we observe that recommendation performances increase first and then perform slightly drop when the graph inference layer increases. Specifically, *VGCL* achieves the best performance with $L = 2$ on the Douban-Book dataset and $L = 3$ on the Dianping dataset, respectively. This suggests that shallow graph inference layers can't well capture graph structure for node distribution estimation, but too deep graph inference layers also decrease the estimation quality due to the over-smoothing issue.

Effect of Temperature τ. As introduced in the previous works, temperature τ controls the mining scale of hard negatives [15]. Specifically, a low temperature will highlight the gradient contributions of hard negatives that are similar to positive nodes. In *VGCL* , there are two temperatures τ_1 and τ_2 in node-level and cluster-level contrastive losses, respectively. As suggested in the previous work, we fix the temperature $\tau_1 = 0.2$ on node-level contrastive loss, then analyze the impact of $\tau = \tau_2$ of the cluster-level contrastive loss. From Figure 4(a) and Figure 4(b), we have the following observations. First, too high or low temperature will decrease recommendation performance on all methods. A too-high temperature drops the ability to mine hard negative samples, while a too-low temperature will over-highlight hard negatives which are usually false negatives. Second, SGL and SimGCL achieve the best performances when temperature $\tau = 0.2$ as suggested in the original paper, while *VGCL* achieves better performance on a smaller temperature, e.g., $\tau = 0.13$ on Douban-Book and $\tau = 0.15$ on Dianping dataset. The reason is that our proposed cluster-aware contrastive learning further encourages the consistency of nodes in a cluster, then a lower temperature will help the model better mine hard negatives.

Effect of Prototype Number K_u and K_i. To investigate the effect of prototype numbers, we set the prototype numbers from zero to hundreds. We illustrate the experimental results in Figure4(c) and Figure4(d). Please note that when $K_u = K_i = 0$, *VGCL* degenerates to VGCL-w/o C without the cluster-level objective. From this Figure, we find that *VGCL* consistently outperforms VGCL-C, which demonstrates that our proposed cluster-aware twofold contrastive learning strategy effectively improves the recommendation performance. For the Douban-Book dataset, *VGCL* reaches the best performances when $K_u = 900$ and $K_i = 300$. For the Dianping dataset, *VGCL* reaches the best performance when $K_u = 500$ and $K_i = 100$. It shows that precise clustering can provide pseudo-labels to distinguish contrastive samples.

Effect of Contrastive Loss Weights α and γ. As illustrated in Figure5, we carefully tune the contrastive loss weights α and γ on the Douban-Book dataset. We observe that *VGCL* achieves the best performance when $\alpha = 0.2$ and $\gamma = 0.4$ on the Douban-Book dataset. As the space limit, we don't present analysis on the other two datasets, the best parameters are $\alpha = 0.05, \gamma = 0.5$ and $\alpha = 0.1, \gamma = 1.0$ on Dianping and Movielens-25M datasets, respectively. Besides, the performance increases first and then drops quickly while α and γ increase. It indicates that proper contrastive loss weights could effectively improve the sparse supervision issue, however, a too-strong self-supervised loss will lead to model optimization neglecting the recommendation task.

5 RELATED WORK

5.1 Graph based Collaborative Filtering

Collaborative filtering is a popular technique widely used in recommender systems. The key is to learn user and item embeddings relying on historical interactions [11, 26, 40]. Early works leverage the matrix factorization technique to project users' and items' IDs into latent embeddings, and compute preferences with the inner product or neural networks [8, 22, 23]. Recently, borrowing the success of Graph Neural Networks (GNNs) [4, 18, 34], a series of graph-based models have been widely studied on various recommendation scenarios [10, 33, 38, 43, 44, 48]. As users' behavior can be naturally formulated as a user-item graph, graph-based CF methods formulate the high-order user-item graph structure on representation learning and achieve great performance improvements [2, 7, 36–38]. NGCF is the first attempt that introduces GNNs

to collaborative filtering, which injects high-order collaborative signals for embedding learning [38]. LR-GCCF proposes linear residual networks for user-item graph learning, which can effectively alleviate the over-smoothing issue in deep graph neural networks [2]. LightGCN is a representative work and proposes a simplified graph convolution layer for CF which only has neighbor aggregation [7]. Despite effectiveness, graph-based CF methods also suffer from sparse supervision. In this work, we investigate collaborative filtering with self-supervised learning to tackle the above issues.

5.2 Contrastive Learning based Recommendation

As one of the popular self-supervised learning paradigms, contrastive learning aims to learn the representational invariants by data augmentation [14, 21]. In general, contrastive learning first generates contrastive views from data augmentation, then maximize the mutual information to encourage the consistency of different contrastive views. Recently, some research successfully apply the CL technique to graph representation learning, either local-global scale contrast [28, 31, 35] or global-global scale contrast [49, 50]. For instance, DGI learns node representations by maximizing the mutual information between the local and global representations [35]. GraphCL proposes four random graph augmentation strategies to multiple subgraphs for contrastive learning [50], AutoGCL further proposes an automated GCL method with learnable contrastive views [47]. Inspired by these works, some GCL-based CF methods have been proposed [1, 20, 39, 46, 51]. BiGI maximizes the local-global mutual information on user-item bipartite graph [1]. EGLN proposes to learn the enhanced graph structure and maximize the mutual information maximization with a local-global objective [46]. Besides, data augmentation based CL techniques are usually applied to CF, aiming to deal with sparse supervision and noise interaction problems [20, 39, 51]. SGL designs three structure graph augmentations to generate contrastive views and improve recommendation accuracy and robustness by maximizing the consistency of different views [39]. NCL proposes neighborhood-enriched contrastive learning to improve performance, it uses the correlated structure neighbors and semantic neighbors as contrastive objects [20]. SimGCL revisits structure augmentation methods and proposes a simple feature augmentation to enhance GCL-based recommendations [51].

Despite the effectiveness, we argue that current GCL-based recommendation methods are still limited by data augmentation strategies whatever structure or feature augmentation. First, structure augmentation randomly deletes nodes or edges of the input graph to generate subgraphs for contrastive learning. However, random structure augmentation is easy to destroy the intrinsic nature of the original user-item graph. Besides, feature augmentation adds the same scale noise to all nodes, ignoring the unique characteristics of nodes (such as degree on the graph), thus can't satisfy all nodes. In this work, we propose a novel contrastive paradigm without data augmentation and implement adaptive contrastive loss learning for different nodes.

5.3 VAE and Applications on Recommendation

Variational Auto-Encoder (VAE) is a generative method widely used in machine learning [16, 24]. It assumes that the input data can be generated from variables with some probability distribution. Following, some extensions of VAE are proposed to improve performance from different perspectives [9, 12, 13, 29]. CVAE considers complex condition distribution on inference and generation process [29], β-VAE proposes to learn the disentangled representations by adding the loss of KL-term [9], and DVAE reconstructs the input data from its corrupted version to enhance the robustness [12]. The basic idea of applying VAEs to the recommendation is to reconstruct the input users' interactions. Mult-VAE proposes that multinomial distribution is suitable for modeling user-item interactions, and parameterizes users by neural networks to enhance the representation ability [19]. RecVAE further improves Mult-VAE by introducing a novel composite prior distribution for the latent encoder [27]. Bi-VAE proposes bilateral inference models to estimate the user-item distribution and item-user distribution [32]. CVGA combines GNNs and VAE and proposes a novel collaborative graph auto-encoder recommendation method, which reconstructs user-item bipartite graph using variance inference [52]. Besides, some works attempt to leverage VAEs for sequential recommendation [45] and cross-domain recommendation [25]. Different from the above VAE-based recommendation models, our *VGCL* introduces the variational inference technique to generate multiple contrastive views for GCL-based recommendation, which build a bridge between generative and contrastive learning models for recommendation.

6 CONCLUSION

In this work, we investigate GCL-based recommendation from the perspective of better contrastive view construction, and propose a novel *Variational Graph Generative-Contrastive Learning (VGCL)* framework. Instead of data augmentation, we leverage the variational graph reconstruction technique to generate contrastive views to serve contrastive learning. Specifically, we first estimate each node's probability distribution by graph variational inference, then generate contrastive views with multiple samplings from the estimated distribution. As such, we build a bridge between the generative and contrastive learning models for recommendation. The advantages have twofold. First, the generated contrastive representations can well reconstruct the original graph without information distortion. Second, the estimated variances vary from different nodes, which can adaptively regulate the scale of contrastive loss for each node. Furthermore, considering the similarity of the estimated distributions of nodes, we propose a cluster-aware twofold contrastive learning, a node-level to encourage consistency of a node's contrastive views and a cluster-level to encourage consistency of nodes in a cluster. Empirical studies on three public datasets clearly show the effectiveness of the proposed framework.

ACKNOWLEDGEMENTS

This work was supported in part by grants from the National Key Research and Development Program of China (Grant No. 2021ZD0111802), the National Natural Science Foundation of China (Grant No. 72188101, 61932009, 61972125, U19A2079, 62006066, U22A2094), Major Project of Anhui Province (Grant No. 202203a05020011), and the CCF-AFSG Research Fund (Grant No. CCF-AFSG RF20210006).

REFERENCES

[1] Jiangxia Cao, Xixun Lin, Shu Guo, Luchen Liu, Tingwen Liu, and Bin Wang. 2021. Bipartite Graph Embedding via Mutual Information Maximization. In *WSDM*. 635–643.

[2] Lei Chen, Le Wu, Richang Hong, Kun Zhang, and Meng Wang. 2020. Revisiting Graph Based Collaborative Filtering: A Linear Residual Graph Convolutional Network Approach. In *AAAI*. 27–34.

[3] Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. JMLR Workshop and Conference Proceedings, 297–304.

[4] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *NeurIPS*. 1024–1034.

[5] F Maxwell Harper. 2015. The movielens datasets: History and context. *TIIS* (2015), 1–19.

[6] John A Hartigan and Manchek A Wong. 1979. Algorithm AS 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)* 28, 1 (1979), 100–108.

[7] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *SIGIR*. 639–648.

[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*. 173–182.

[9] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2016. beta-vae: Learning basic visual concepts with a constrained variational framework. (2016).

[10] Jun Hu, Shengsheng Qian, Quan Fang, Youze Wang, Quan Zhao, Huaiwen Zhang, and Changsheng Xu. 2021. Efficient graph deep learning in tensorflow with tf_geometric. In *MM*. 3775–3778.

[11] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *ICDM*. 263–272.

[12] Daniel Im Im, Sungjin Ahn, Roland Memisevic, and Yoshua Bengio. 2017. Denoising criterion for variational auto-encoding framework. In *AAAI*, Vol. 31.

[13] Oleg Ivanov, Michael Figurnov, and Dmitry Vetrov. 2019. Variational autoencoder with arbitrary conditioning. In *ICLR*.

[14] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. 2020. A survey on contrastive self-supervised learning. *Technologies* 9, 1 (2020), 2.

[15] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *NeurIPS* 33 (2020), 18661–18673.

[16] Diederik P Kingma and Max Welling. 2014. Auto-encoding variational bayes. *ICLR* (2014).

[17] Thomas N Kipf and Max Welling. 2016. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308* (2016).

[18] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.

[19] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. 2018. Variational autoencoders for collaborative filtering. In *WWW*. 689–698.

[20] Zihan Lin, Changxin Tian, Yupeng Hou, and Wayne Xin Zhao. 2022. Improving Graph Collaborative Filtering with Neighborhood-enriched Contrastive Learning. In *WWW*. 2320–2329.

[21] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. 2021. Self-supervised learning: Generative or contrastive. *TKDE* (2021).

[22] Andriy Mnih and Russ R Salakhutdinov. 2008. Probabilistic matrix factorization. In *NeurIPS*. 1257–1264.

[23] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*. 452–461.

[24] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*. 1278–1286.

[25] Aghiles Salah, Thanh Binh Tran, and Hady Lauw. 2021. Towards Source-Aligned Variational Models for Cross-Domain Recommendation. In *Recsys*. 176–186.

[26] Pengyang Shao, Le Wu, Lei Chen, Kun Zhang, and Meng Wang. 2022. FairCF: fairness-aware collaborative filtering. *Science China Information Sciences* 65, 12 (2022), 1–15.

[27] Ilya Shenbin, Anton Alekseev, Elena Tutubalina, Valentin Malykh, and Sergey I Nikolenko. 2020. Recvae: A new variational autoencoder for top-n recommendations with implicit feedback. In *WSDM*. 528–536.

[28] Jie Shuai, Kun Zhang, Le Wu, Peijie Sun, Richang Hong, Meng Wang, and Yong Li. 2022. A review-aware graph contrastive learning framework for recommendation. In *SIGIR*. 1283–1293.

[29] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. *NeurIPS* 28 (2015).

[30] Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence* (2009).

[31] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. 2020. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. *ICLR* (2020).

[32] Quoc-Tuan Truong, Aghiles Salah, and Hady W Lauw. 2021. Bilateral variational autoencoder for collaborative filtering. In *WSDM*. 292–300.

[33] Rianne van den Berg, Thomas N Kipf, and Max Welling. 2017. Graph Convolutional Matrix Completion. *STAT* 1050 (2017), 7.

[34] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. *ICLR* (2018).

[35] Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep graph infomax. *ICLR* (2019).

[36] Wenjie Wang, Fuli Feng, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2021. Denoising implicit feedback for recommendation. In *WSDM*. 373–381.

[37] Wenjie Wang, Fuli Feng, Xiangnan He, Xiang Wang, and Tat-Seng Chua. 2021. Deconfounded recommendation for alleviating bias amplification. In *KDD*. 1717–1725.

[38] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *SIGIR*. 165–174.

[39] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised graph learning for recommendation. In *SIGIR*. 726–735.

[40] Le Wu, Xiangnan He, Xiang Wang, Kun Zhang, and Meng Wang. 2022. A survey on accuracy-oriented neural recommendation: From collaborative filtering to information-rich recommendation. *TKDE* (2022).

[41] Le Wu, Junwei Li, Peijie Sun, Richang Hong, Yong Ge, and Meng Wang. 2020. Diffnet++: A neural influence and interest diffusion network for social recommendation. *TKDE* (2020).

[42] Lirong Wu, Haitao Lin, Cheng Tan, Zhangyang Gao, and Stan Z Li. 2021. Self-supervised learning on graphs: Contrastive, generative, or predictive. *TKDE* (2021).

[43] Le Wu, Yonghui Yang, Lei Chen, Defu Lian, Richang Hong, and Meng Wang. 2020. Learning to transfer graph embeddings for inductive graph based recommendation. In *SIGIR*. 1211–1220.

[44] Le Wu, Yonghui Yang, Kun Zhang, Richang Hong, Yanjie Fu, and Meng Wang. 2020. Joint item recommendation and attribute inference: An adaptive graph convolutional network approach. In *SIGIR*. 679–688.

[45] Zhe Xie, Chengxuan Liu, Yichi Zhang, Hongtao Lu, Dong Wang, and Yue Ding. 2021. Adversarial and contrastive variational autoencoder for sequential recommendation. In *WWW*. 449–459.

[46] Yonghui Yang, Le Wu, Richang Hong, Kun Zhang, and Meng Wang. 2021. Enhanced graph learning for collaborative filtering via mutual information maximization. In *SIGIR*. 71–80.

[47] Yihang Yin, Qingzhong Wang, Siyu Huang, Haoyi Xiong, and Xiang Zhang. 2022. Autogcl: Automated graph contrastive learning via learnable view generators. In *AAAI*, Vol. 36. 8892–8900.

[48] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *SIGKDD*. 974–983.

[49] Yuning You, Tianlong Chen, Yang Shen, and Zhangyang Wang. 2021. Graph contrastive learning automated. In *ICML*. 12121–12132.

[50] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. *NeurIPS* 33 (2020).

[51] Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Quoc Viet Hung Nguyen. 2022. Are graph augmentations necessary? simple graph contrastive learning for recommendation. In *SIGIR*. 1294–1303.

[52] Yi Zhang, Yiwen Zhang, Dengcheng Yan, Shuiguang Deng, and Yun Yang. 2022. Revisiting Graph-based Recommender Systems from the Perspective of Variational Auto-Encoder. *TIST* (2022).

[53] Wayne Xin Zhao, Junhua Chen, Pengfei Wang, Qi Gu, and Ji-Rong Wen. 2020. Revisiting alternative experimental settings for evaluating top-n item recommendation algorithms. In *CIKM*. 2329–2332.