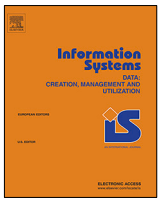
Information Systems 109 (2022) 102051

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/is



Handling information loss of graph convolutional networks in collaborative filtering☆

Xin Xiong [a], XunKai Li [b], YouPeng Hu [b], YiXuan Wu [c], Jian Yin [b,*]

[a] Nanjing University, School of Artificial Intelligence, No.163 Xianlin Avenue, Qixia District, Nanjing City, Jiangsu Province, China
[b] Shandong University, School of Mechanical, Electrical and Information Engineering, No.180 Wenhua West Road, Weihai City, Shandong Province, China
[c] Zhejiang University, Polytechnic Institute, No. 269 Shixiang Road, Hangzhou City, Zhejiang Province, China

ARTICLE INFO

Article history:
Received 26 August 2021
Received in revised form 20 March 2022
Accepted 13 April 2022
Available online 21 April 2022
Recommended by Yannis Manolopoulos

Keywords:
Collaborative filtering
Graph convolutional network
Recommender system
Variational autoencoder

ABSTRACT

Collaborative filtering (CF) methods based on graph convolutional network (GCN) and autoencoder (AE) achieve outstanding performance. But the GCN-based CF methods suffer from information loss problems, which are caused by information lossy initialization and using low-order Chebyshev Polynomial to fit the graph convolution kernel. And the AE-based CF methods obtain the prediction results by reconstructing the user-item interaction matrix, which does not conduct deep excavation of the behavior patterns, resulting in the limited-expression ability.

To solve the above problems, we propose Variational AutoEncoder-Enhanced Graph Convolutional Network (VE-GCN) for CF. Specifically, we use a variational autoencoder (VAE) to compress the interactive behavior patterns as the prior information of GCN to achieve sufficient learning, thus alleviating the information lossy initialization problem. And then the generalized graph Laplacian convolution kernel is proposed in GCN to handle the high-frequency information loss problem caused by Chebyshev Polynomial fitting in the GCN-based CF. To the best of our knowledge, VE-GCN is a feasible method to handle the information loss problems mentioned above in GCN-based CF for the first time. Meanwhile, the structure of GCN is optimized by removing redundant feature transformation and nonlinear activation function, and using DenseGCN to complete multi-level information interaction. Experiments on four real-world datasets show that the VE-GCN achieves state-of-the-art performance.

1. Introduction

Recommender system, as the key technology of personalized service in the age of the Internet, has attracted wide attention because it can manage the information explosion phenomenon effectively and generate commercial value. Collaborative filtering (CF) is a classic recommendation method that filters tons of information according to the historical interaction data and uses the collaborative information to screen out the items that the users may be interested in. The original collaborative filtering methods calculate the similarity between users and items according to the user-item historical interaction matrix by cosine similarity [1] and Pearson correlation coefficient [2] and so forth. After obtaining the scores ranked by the similarity of users and items, the final recommendation results are generated. But there is a problem called the "head effect", which means popular items will be recommended more frequently than unpopular items because they have participated in more interactions. That is to say, popular items will be more popular, while unpopular items will be more unpopular. To solve the problems of obvious head effect and weak generalization ability of the original collaborative filtering, some matrix factorization (MF) [3] methods for CF are proposed. They obtain embeddings of users and items by MF of the historical interaction matrix, and make the inner product of the embeddings to generate recommendation results, such as FunkSVD [4], LFM [5], and NMF [6], etc. However, MF at this time is not expressive enough in the way of obtaining user and item representations, therefore limiting the performance.

With the rapid development of computing resources, graph neural network (GNN)-based CF has been widely studied and has achieved outstanding performance. CF is essentially the link prediction problem of the bipartite graph of users and items, and this kind of graph-based task can be solved well by GNN. The core idea of GNN-based CF is to use GNN to learn user behavior patterns and item co-occurrence relationships into embeddings, and obtain the final prediction results through feature interaction

☆ This work was supported by the National Natural Science Foundation of China under Grant 61971268.
* Corresponding author.
E-mail addresses: xiongxin@smail.nju.edu.cn (X. Xiong), lxk_yb@163.com (X. Li), yooooooohu@gmail.com (Y. Hu), beixuan_wh@163.com (Y. Wu), yinjian@sdu.edu.cn (J. Yin).

https://doi.org/10.1016/j.is.2022.102051
0306-4379/

functions. Graph Convolutional Network (GCN) [7–9] is a kind of GNN, which has a strong ability to learn non-Euclidean structure data while maintaining computational efficiency. Based on graph topology, GCN uses spectrum theory to complete feature propagation and information aggregation through graph convolution. Some GCN-based CF methods have achieved state-of-the-art performance, and the learning ability of GCN-based CF: NGCF [10], LightGCN [11], LR-GCCF [12] and so forth, on the node embeddings exceeds that of the DNN- or CNN- based CF: NCF [13], ONCF [14], etc. The widely used GCN [7] uses the first-order Chebyshev Polynomial to fit the graph convolution kernel in order to maintain the simplicity of the calculation, which leads to the loss of high-order eigenvalues. Inspired by the related work of graph node clustering AGC [15], in the eigendecomposition of the graph Laplacian matrix, the eigenvectors corresponding to smaller eigenvalues have lower frequencies and are smoother. Therefore, the Chebyshev Polynomial fitting process makes the graph convolution operation essentially a process of applying low-pass filters to the signals on the graph, and correspondingly leads to the loss of high-frequency information in the node signal.

GCN was first proposed for node classification tasks [16,17] and was applied to CF later, but there is an essential difference between node classification and CF, so the existing GCN cannot be directly migrated to CF in parallel. FAGCN [18] emphasizes that the low-frequency information and the high-frequency information describe the similarities and the differences of the nodes in the network respectively. Therefore, the low-frequency information obtained through GCN is an advantage for node classification tasks, because the essence of node classification tasks is to aggregate similar nodes. However, for CF, the bipartite graph of users and items is a heterogeneous graph. The user's latent representations and the item's latent representations are obtained by aggregating the item collections and the user collections respectively. The similarity and difference measures of nodes are unknown on different types of nodes. It is difficult for us to know whether the CF requires low-frequency or high-frequency information specifically. Then, without knowing whether low-frequency or high-frequency information is needed, the previous GCN-based CF uniformly discarded high-frequency information. If we can retain this kind of high-frequency information effectively, we will be able to learn more suitable embeddings, thereby improving the prediction results of CF.

There are various studies on the information loss problems of GCN-based recommender systems at present. In [19], the problem of temporal information loss in recommender systems is studied, focusing on incremental GCN how to save temporal information from the last period and the current period. [20] focuses on the information loss problems in the session-based recommendation, namely the lossy session encoding problem and the ineffective long-range dependency capturing problem, and these two problems are both about the ineffective conversion of sessions and are closely related to sequence information. In [21], it is mentioned that compressing the neighbor information of all orders into a fixed-size vector in GCN will cause the loss of important information in the transmission process. However, we focus on the problems of information loss in GCN-based CF, which pays attention to the problem of migration of GCN into CF in parallel, and is different from the above literature.

To summarize, the existing GCN-based CF has achieved the SOTA performance, but there are problems of information loss uniformly. If we can solve these problems, the recommendation effect will be greatly improved. To be specific, there are two information loss problems in GCN-based CF: 1. Using ID embeddings with weak semantic information to initialize GCN input leads to information loss initialization; 2. Using the first-order approximation of Chebyshev Polynomial to fit graph convolution kernel leads to information loss learning.

To solve the above problems, we propose VE-GCN for CF. On the one hand, VE-GCN uses variational autoencoder (VAE) [22] to learn robust users' and items' embeddings from the initial interaction matrix as the prior information to enrich the learning process of the GCN. The general GCN-based CF is initialized with ID embeddings obeying the standard normal distribution. This initialization method based on the central limit theorem is universal. Firstly, non-universal normal distributions can be learned by VAE to personalize the initialization of nodes in the network. Also, the node distributions learned by VAE have better generalization attributes. Secondly, the user-item interaction information extracted by VAE should contain high-frequency and low-frequency information at the same time, because of the not frequency-specific learning method, so it can be more conducive to the subsequent GCN learning process. On the other hand, a generalized graph Laplacian convolution kernel is used for processing the graph signals while keeping the computational simplicity of the model. The generalized graph Laplacian convolution kernel can process low-frequency and high-frequency information at the same time. Therefore, the information loss problem in GCN caused by Chebyshev Polynomial fitting is handled. Meanwhile, we optimize the structure of the existing GCN to simplify the calculation and improve the performance. Specifically, we remove redundant feature transformation and nonlinear activation function, and then introduce DenseGCN to connect each layer. Experiments prove that VE-GCN has achieved state of the art performance.

In summary, the main contributions of this paper are as follows:

- We summarize the problems of information loss in GCN-based CF for the first time. To solve the problems, we propose a decoupling model VE-GCN.
- We use VAE to compress historical interactive data as the prior information to learn the interactive information sufficiently and use generalized graph Laplacian convolution kernel to process the low- and high-frequency information. We use the above methods to solve the problems of information loss in GCN. Also, GCN is optimized by removing feature transformation and nonlinear activation function to simplify the calculation, and introducing DenseGCN to achieve multi-level information interaction.
- Experiments on four real-world datasets show that the VE-GCN achieves state-of-the-art performance. What is more, VE-GCN alleviates the problems of information loss in GCN-based CF to some extent.

2. Related work

In this section, we will briefly summarize the relevant work of GCN-based CF and the relevant work of VAE.

2.1. GCN-based collaborative filtering

GCN is based on the graph topology structure and uses spectrum theory to apply convolution operation on the graph structure data to complete data mining. Data mining refers to the process of searching key information hidden in a large amount of data through algorithms, what we want to emphasize here is that GCN can find in the topology graph the important information we need. The input of CF is bipartite graph data, so GCN can be widely used in CF and can be roughly divided into Graph-Autoencoder-based and Graph-Deep-Learning-based methods.

GC-MC [23] regards the recommendation prediction problem as a link prediction problem on bipartite graphs, which uses graph autoencoders to learn the embeddings of users and items based on the interaction matrix, then uses bilinear decoder to get

the reconstructed interaction matrix as the result of the model. STAR-GCN [24] is based on the framework of GC-MC, which proposes to stack multiple GC-MC graph autoencoder blocks, then uses task loss and reconstruction loss to complete the recommended tasks novelly. And the dropout method proposed by STAR-GCN provides a new idea to solve the cold start problem of the recommender system.

Graph-Deep-Learning-based CF has developed rapidly in recent years. NGCF [10] uses multi-layer GCNs to get the information embeddings of users and items and uses the inner product of embeddings to complete the prediction task. However, the complex information propagation method in GCN limits the expression ability of the model. The appearance of SGC [25] simplifies the structure of GCN, eliminates the nonlinear activation function between GCN layers, and multiple feature transforms are folded into a linear transformation function. There is no semantic information in the node features of GCN-based CF, while the collaborative information hidden in the topology structure is the focus of learning, turning SGC into a kind of information transmission framework that is very suitable for CF. LR-GCCF [12] utilizes a linear embedding propagation framework similar to SGC and combines residual connections to strengthen the feature interaction between layers to improve the recommendation effect of the model and alleviate the over-smoothing problem in GCN. LightGCN [11] finds that the nonlinear activation function and feature transformation process in GCN is meaningless, so only the information aggregation in GCN is retained in LightGCN, which not only reduced the complexity of the model but also improved the recommendation effect. LCF [26] discusses the impact of simplified graph convolution on CF and designs to eliminate the noise caused by exposure and quantization in the observation data, therefore improving the performance of the model.

2.2. Variational autoencoder

The latent representation of the autoencoder can compress the input information well, which can obtain the dense feature representations from the sparse inputs, and it is very suitable for getting embeddings. But the original autoencoder-based models are essentially point-to-point reconstruction methods, then the random sampling results of the latent representation are unreasonable. Therefore VAE [22] comes in. VAE aims to obtain a latent distribution rather than a latent representation through point-to-point reconstruction. The latent distribution of VAE has better generalization performance and can describe the characteristics of the input better. β-VAE [27] introduces hyperparameter β to the KL divergence term of VAE and makes $\beta > 1$, then the enhancement of the KL term can increase the model's disentanglement. Disentanglement means that the attributes of different elements are separated from each other. The increase in disentanglement makes the various attributes of the input more separated so that the model has a better generation effect. In β-TCVAE [28], the increase in the disentanglement of β-VAE is explained in principle, and the KL divergence is divided into three sub-divergences. β-TCVAE decreases the total correlation to increase the disentanglement without introducing redundant hyperparameters. The fusion of autoregressive distribution, multi-scale architecture, separable convolution, swish activation function, and flow model in NVAE [29] makes an unprecedented improvement of the generation effect of VAE.

These variants of VAE pay more attention to the improvement of VAE's generation ability. The increase of disentanglement in hidden layer distribution can improve the generation ability of VAE [28]. However, we expect the distribution sampled from VAE to meet a variety of user behavior patterns, the increase of disentanglement is of no use to us, and we will discuss this in the experiment part.

3. Model

VE-GCN is mainly an improvement for the problem of information loss in GCN-based CF. The formal definition of the information loss problem in GCN-based CF is as follows. First, GCN-based CF is often initialized with ID embeddings with weak semantic information, ignoring the rich interactive information between users and items, namely the information lossy initialization. Second, since the proposal of GCN is essentially to solve the task of node classification and so forth that requires low-frequency information, problems arise when using GCN directly in CF. The lost high-frequency information, which is caused by the fitting of Chebyshev Polynomial, may be useful for CF and the lost information lead to a decrease in performance, namely the high-frequency information loss problem in GCN-based CF.

The target of VE-GCN is to predict whether a user will interact with an item based on the existing user-item interaction matrix. VE-GCN is a decoupling model, which consists of the VAE part and the GCN part. Firstly, VE-GCN proposes to use the bottleneck vectors, which mean the vectors obtained under the middle layer of the encoder–decoder framework, of the VAE part as the prior information of the GCN part to solve the problem of information lossy initialization. The VAE part can extract the node distributions of users and items from the user-item interaction matrix containing complex behavior patterns. It should be noted that the information extracted by VAE includes both high-frequency and low-frequency information. Secondly, in the GCN part, the generalized graph Laplacian convolution kernel and the optimized graph information propagation framework are used to complete the signal processing process. The generalized graph Laplacian convolution kernel can preserve high-frequency information to some extent. In this way, the information loss calculation of GCN in CF caused by Chebyshev Polynomial fitting is handled. We get the predicted ratings through the feature interaction function based on the users' and items' embeddings got by the GCN part. The final recommendation results are obtained by sorting the predicted ratings. Fig. 1 shows the overall framework of VE-GCN.

3.1. Preliminary

To describe VE-GCN, we make some definitions as follows. The interaction matrix between users and items is defined as $\mathbf{R} \in \mathbb{R}^{|M|\times|N|}$, where $|M|$ and $|N|$ represent the number of users and items respectively. For each element $r_{ij} \in \mathbf{R}$, if $r_{ij} = 1$, it means that there is a positive interaction between user i and item j, such as click, purchase, browse, otherwise $r_{ij} = 0$. The input of User-VAE and Item-VAE is the item sequences that the users have interacted with and the user sequences that the items have interacted with, respectively. That is to say the user-item interaction matrix $\mathbf{R}$ and the item-user interaction matrix $\mathbf{R}^T$ are used as the input of the User-VAE and the Item-VAE, respectively. The input of the GCN part are the bottleneck vectors $\boldsymbol{z}_u$ of User-VAE and the bottleneck vectors $\boldsymbol{z}_i$ of Item-VAE, which are used to initialize the users' and items' embeddings of the GCN part's first layer: $\boldsymbol{e}_u^{(0)}$, $\boldsymbol{e}_i^{(0)}$.

An undirected simple graph can be represented as $G = (V, E)$, where V and E represent the set of vertices and edges in the graph, respectively. Based on this, the adjacency matrix of graph G can be defined as $\mathbf{A} \in \mathbb{R}^{T\times T}$, where T represents the number of nodes in the graph. $\mathbf{D}$ is the degree matrix of $\mathbf{A}$. As a discrete Laplacian operator on a graph, the graph Laplacian matrix reflects the gradient differences between the center point and the surrounding points. The graph Laplacian matrix is the core of performing the graph convolution operation, which aggregates the neighbor information of the central node so that each node can obtain a representation of the local topology.

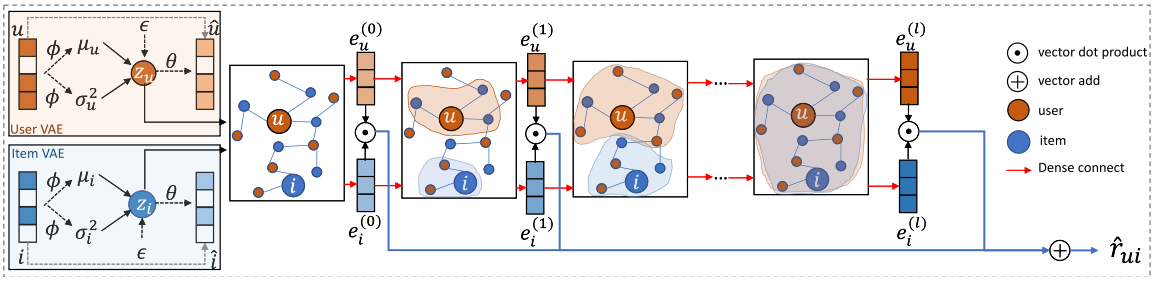


Fig. 1. The overall framework of VE-GCN. μ_u, μ_i and σ_u^2, σ_i^2 represents the mean and the variance of the bottleneck vectors of the User-VAE and Item-VAE respectively. ϵ represents the standard normal distribution $N(0, 1)$, ϕ and θ represent the inference and the generate parameters in the VAE part respectively, and the ϕ and θ in User-VAE and Item-VAE are not shared. $\boldsymbol{e}_u^{(l)}$ and $\boldsymbol{e}_i^{(l)}$ represent the embeddings of users and items at the lth layer in the GCN part respectively.

It is often used in signal processing on graph, and the most frequently used symmetric normalized graph Laplacian matrix is defined as $\mathbf{L}_{\text{sym}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$. Because of the semi-positive definite property of the symmetric graph Laplacian matrix, the graph Laplacian matrix can be decomposed as $\mathbf{L}_{\text{sym}} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$. $\mathbf{U}$ represents a set of completely orthogonal bases and can be used as a set of bases in the frequency domain to complete Fourier Transform of spatial domain signals on the graph. $\mathbf{\Lambda}$ is a diagonal matrix composed of the eigenvalues of the graph nodes, each element is not negative and its values from small to large reflect the low-frequency information to high-frequency information in the graph. The above concepts are the theoretical bases for completing the matrix multiplication operation in the frequency domain and then completing the convolution operation in the corresponding spatial domain.

3.1.1. Widely used graph convolution form

In order to avoid the huge complexity of calculation for eigen decomposition of the Laplacian matrix, the commonly used layer-wise graph convolution process is defined as follows:

$$\boldsymbol{Q}^{(l+1)} = \sigma(\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{Q}^{(l)}\boldsymbol{W}^{(l)}), \tag{1}$$

where $\tilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}$, $\boldsymbol{A}$ represents the adjacency matrix of the graph, $\boldsymbol{I}$ represents the identity matrix, $\tilde{\boldsymbol{D}}$ is the degree matrix of $\boldsymbol{A}$. $\boldsymbol{Q}^{(l)}$ represents the feature embedding matrix of the lth layer, $\boldsymbol{Q}^{(0)}$ represents the initial node feature matrix. $\boldsymbol{W}^{(l)}$ represents the lth layer feature transformation matrix, and $\sigma(\cdot)$ represents the activation function, such as ReLU [30], etc. $\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-\frac{1}{2}}$ is used for neighbor information aggregation, and $\boldsymbol{W}^{(l)}$ is used to perform feature space transformation.

3.2. The VAE part

In VE-GCN, there are two independent VAEs based on users and items: User-VAE and Item-VAE, which use historical interaction to generate prior information. The prior information is used to achieve sufficient learning. The process of VAE can be divided into two steps: the inference process and the generate process. To show details in the VAE part, we take User-VAE as an example, and the process of Item-VAE is similar.

3.2.1. The inference process

The inference process refers to the process of generating random variables, from the random process, which will be further used by the generate process.

$$g_\phi(x_u) = \text{MLP}(x_u) \equiv \left[\mu_\phi(x_u), \sigma_\phi^2(x_u)\right] \in \mathbb{R}^{2K}. \tag{2}$$

Multi-Layer Perceptron (MLP) can approximate any complex polynomial theoretically. For user input x_u (the sequence of items that user u interacts with), we use MLP$(\cdot)$ to fit $\mu_\phi(x_u) \in \mathbb{R}^K$ and $\sigma_\phi^2(x_u) \in \mathbb{R}^K$ respectively, which are used to form the user's hidden layer feature distribution. ϕ represents the set of inference parameters, $g_\phi(x_u)$ represents the variational parameter equation learned by MLP$(\cdot)$. $[\cdot, \cdot]$ represents $\mu_\phi(x_u)$ and $\sigma_\phi^2(x_u)$ are stored in $g_\phi(x_u)$ in sequence, the first K terms of $g_\phi(x_u)$ are $\mu_\phi(x_u)$, and the last K terms are $\sigma_\phi^2(x_u)$.

Suppose that z_u is a random variable sampled from the standard normal distribution, i.e., $z_u \sim N(0, 1)$, which is used to ensure that the noise of the VAE part always exists and prevents the VAE part from degenerating into an ordinary autoencoder, so as to ensure the generation ability of the VAE part. The formation process of the posterior distribution $q_\phi(z_u|x_u)$ is as follows:

$$q_\phi(z_u|x_u) = N\left(\mu_\phi(x_u), \text{diag}\left(\exp(\sigma_\phi^2(x_u))\right)\right) \in \mathbb{R}^K, \tag{3}$$

where $\text{diag}\left(\exp(\sigma_\phi^2(x_u))\right)$ represents the diagonal matrix with $\exp(\sigma_\phi^2(x_u))$ as the element. $q_\phi(z_u|x_u)$ depicts the distribution of user's hidden layer features and reflects the attributes of user u in K dimensions. The distribution of item's hidden layer features is as follows:

$$q_\phi(z_i|x_i) = N\left(\mu_\phi(x_i), \text{diag}\left(\exp(\sigma_\phi^2(x_i))\right)\right) \in \mathbb{R}^K. \tag{4}$$

3.2.2. The generate process

The generate process refers to the process of generating data through the mapping equation based on the variable parameters obtained from the above-mentioned inference process.

$$\pi(x_u|z_u) = \text{softmax}\left(\exp(f_\theta(q_\phi(z_u \mid x_u)))\right), \tag{5}$$

where $f_\theta(\cdot)$ represents a non-linear MLP, θ represents the set of generate parameters. The probability vector $\pi(x_u|z_u)$ represents the probability of obtain x_u after given the potential distribution z_u, which is generated by sending $\exp(f_\theta\left(q_\phi(z_u \mid x_u)\right))$ through softmax$(\cdot)$.

3.2.3. The optimize process

First of all, the optimization goal of VAE is to get the maximum likelihood estimation (MLE) [31] of x_u, and the formula is as follows:

$$L = \max_\theta \sum_{x_u} \log\ p(x_u; \theta). \tag{6}$$

After formula reasoning in [22], the optimization goal has become to maximize the evidence lower-bound ELBO $\equiv \mathcal{L}(x_u; \theta, \phi)$:

$$\begin{aligned} \log\ p(x_u; \theta) &\geq \mathbb{E}_{q_\phi(z_u|x_u)}[\log\ p_\theta(x_u \mid z_u)] \\ &- \text{KL}\left(q_\phi(z_u \mid x_u) \parallel p(z_u)\right) \\ &\equiv \mathcal{L}(x_u; \theta, \phi). \end{aligned} \tag{7}$$

$$\log\ p_\theta(x_u \mid z_u) = \sum_i x_{ui} \log \pi_i(x_u \mid z_u). \tag{8}$$

We use the multinomial distribution to estimate x_u as in Multi-VAE [32], and regard the KL $(\cdot)$ as a regularization term, then the formula can be improved into the following form:

$$\begin{aligned}\mathcal{L}(x_u;\theta,\phi) \equiv & \mathbb{E}_{q_\phi(z_u|x_u)}[\log\ p_\theta(x_u \mid z_u)] \\ & -\ \beta \cdot \mathrm{KL}\left(q_\phi(z_u \mid x_u) \parallel p(z_u)\right),\end{aligned} \tag{9}$$

where β represents the hyperparameter, indicating the degree of regularization. We use the same annealing trick as in [32] for β.

For a detailed description of the algorithms, you can refer to Algorithm 1.

Algorithm 1 User-VAE Training Algorithm

Input: $\mathbf{R} \in \mathbb{R}^{|M|\times|N|}$: the user-item rating matrix; Ep: the epochs; β: the regularization hyperparameters;

```
Randomly initialize θ_u, φ_u;
for iter ∈ 0,1,..., Ep do
    Sample a batch of users 𝒰;
    for all u ∈ 𝒰 do
        Sample ϵ ∈ 𝒩(0, I_K);
        Compute z_u via Eq.(3-3);
        Compute loss ℒ and gradient Δ_{θ_u}ℒ and Δ_{φ_u}ℒ with z_u;
    end for;
    Average gradients from batch;
    Update θ_u, φ_u by optimizer;
end for
```

Output: the hidden distribution of users $\boldsymbol{z}_u$ computed by Eq.(3-10).

The prior information $\boldsymbol{z}_u, \boldsymbol{z}_i$ provided to the GCN part are defined as follows:

$$\boldsymbol{z}_u = \frac{\mu_u + \exp(\sigma_u^2)\times\frac{\epsilon}{\tau}}{\|\mu_u + \exp(\sigma_u^2)\times\frac{\epsilon}{\tau}\|_\infty}, \boldsymbol{z}_i = \frac{\mu_i + \exp(\sigma_i^2)\times\frac{\epsilon}{\tau}}{\|\mu_i + \exp(\sigma_i^2)\times\frac{\epsilon}{\tau}\|_\infty}, \tag{10}$$

where μ_u, μ_i and ${\sigma_u}^2$, ${\sigma_i}^2$ represent the mean and variance fitted by the User-VAE and the Item-VAE inference process respectively, ϵ represents the standard normal distribution $N(0, 1)$, τ represents the ϵ's scaling hyperparameter, $\|\cdot\|_\infty$ represents the infinite norm.

3.3. The GCN part

GCN-based CF usually uses the IDs of users and items with weak semantic information as initial nodes features. We use VAE enhancement to solve this kind of initialization problem with information loss. More importantly, the GCN as a low-pass filter is beneficial for tasks such as node classification, but the lost high-frequency information may be useful for CF. GCN-based CF methods do not consider that the GCN process has the high-frequency information lossy problem, so in addition to VAE-Enhanced, we further introduce a generalized graph Laplacian convolution kernel. We will discuss this in detail in Section 3.5.

In order to give the matrix form of the GCN calculation process, the adjacency matrix form of the user-item interaction graph is defined as follows:

$$\mathbf{A} = \begin{pmatrix}\mathbf{0} & \mathbf{R} \\ \mathbf{R}^T & \mathbf{0}\end{pmatrix}. \tag{11}$$

The Laplacian matrix is defined as: $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{D}$ represents the degree matrix of $\mathbf{A}$: $\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}$. The Laplacian matrix can be eigen-decomposed as: $\mathbf{L} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$, where $\mathbf{U} = [u_1, \ldots, u_n]$, $\boldsymbol{\Lambda} = \mathrm{diag}([\lambda_1, \ldots, \lambda_n])$. $\{u_l\}_{l=1}^n$ represents all orthogonal eigenvectors, and $\{\lambda_l\}_{l=1}^n$ represents all non-negative eigenvalues.

The Normalized Laplacian matrix is defined as follows, we directly use $\mathbf{L}$ to represent it for simplicity:

$$\begin{aligned}\mathbf{L} &= \mathbf{D}^{-\frac{1}{2}}(\mathbf{D}-\mathbf{A})\mathbf{D}^{-\frac{1}{2}} \\ &= \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}.\end{aligned} \tag{12}$$

The purpose of normalizing the Laplacian matrix is to prevent the features of nodes with large degrees from becoming larger and larger, while the features of nodes with small degrees become smaller and smaller, thereby preventing the problem of vanishing gradient or exploding gradient.

In order to introduce the generalized graph Laplacian convolution kernel, we define the generalized Laplacian smoothing filter $\mathbf{H}$ as follows:

$$\mathbf{H} = \mathbf{I} - \kappa\mathbf{L}. \tag{13}$$

$\mathbf{H}$ can be seen as a special smoothing filter. Assuming the input signal is $\boldsymbol{x}$, the filtered signal $\widetilde{\boldsymbol{x}}$ is defined as:

$$\widetilde{\boldsymbol{x}} = \mathbf{H}\boldsymbol{x} = \mathbf{U}(\mathbf{I} - \kappa\boldsymbol{\Lambda})\mathbf{U}^T\boldsymbol{x} = \sum_{i=1}^{n}(1-\kappa\lambda_i)\boldsymbol{x}, \tag{14}$$

where $1 - \kappa\lambda$ is the frequency response function, $\kappa \in \mathbb{R}$. The selection of the frequency response function will affect the filtering effect of $\mathbf{H}$ on different frequencies. When using the t-order generalized Laplacian smoothing filter, which is actually a multiple smoothing process, we get the filtered signal $\widetilde{\boldsymbol{X}}$ as follows:

$$\widetilde{\boldsymbol{X}} = \mathbf{H}^t\boldsymbol{X}. \tag{15}$$

The formula reasoning of $\mathbf{H}$ in matrix form:

$$\mathbf{H} = \mathbf{I} - \kappa\left(\mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\right) = (1-\kappa)\mathbf{I} + \kappa\left(\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\right). \tag{16}$$

Then we get the complete generalized graph Laplacian convolution kernel form as follows:

$$\boldsymbol{\xi} = \mathbf{H}^t = [(1-\kappa)\mathbf{I} + \kappa\left(\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\right)]^t. \tag{17}$$

$\mathbf{H}$ is essentially a low-pass filter, so the high-order Laplacian convolution kernel will increase the loss of high-frequency information, and we will verify this in the experimental part, then the generalized graph Laplacian convolution kernel with $t = 1$ is used as the filter in our GCN process. For the selection of the parameter κ, the distribution of the eigenvalue $\boldsymbol{\Lambda}$ should be considered, assuming the maximum value of the eigenvalue to be λ_{max}. According to AGC [15], when $\kappa < \frac{1}{\lambda_{max}}$, $\boldsymbol{H}$ is still a low-pass filter, but some high-frequency components are retained. So we can design a low-pass filter that retains some of the high-frequency information. Since obtaining λ_{max} through matrix factorization will add huge computational complexity to our model, we set κ as a hyperparameter.

The GCN part aims to mine the topological relationship of the graph, so as to obtain the interactive behavior patterns of users and items. Inspired by LightGCN [25], the feature transformation and nonlinear activation function in GCN are redundant operations for CF, which will increase the computational complexity and affect the expressive ability of the model. In order to reduce model complexity and improve model performance, we remove the redundant feature transformation and nonlinear activation function in the GCN. Meanwhile, DenseGCN [33] is used to achieve dense connections between layers, which strengthens the feature interaction and improves the robustness (see Fig. 2). The aggregation operation of the user and item nodes at the $(l+1)$-layer in VE-GCN is defined as follows:

$$\boldsymbol{e}_u^{(l+1)} = \boldsymbol{\xi}\gamma\left(\boldsymbol{e}_u^{(0)}, \boldsymbol{e}_u^{(1)}, \ldots, \boldsymbol{e}_u^{(l)}\right),$$

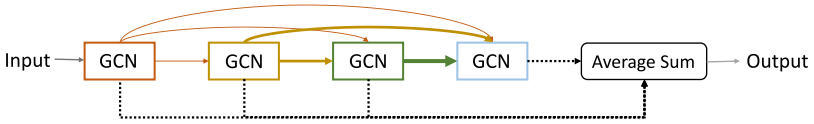


Fig. 2. DenseGCN method. The dashed line represents that the output of each layer of GCN will pass through another GCN layer, where the Average Sum represents the mean cascade function.

$$\boldsymbol{e}_i^{(l+1)} = \boldsymbol{\xi}\gamma\left(\boldsymbol{e}_i^{(0)}, \boldsymbol{e}_i^{(1)}, \ldots, \boldsymbol{e}_i^{(l)}\right), \tag{18}$$

where $\boldsymbol{e}_u^{(l+1)}$ and $\boldsymbol{e}_i^{(l+1)}$ represent the embeddings of users and items in the $(l+1)$th layer respectively, $\boldsymbol{\xi}$ represents the generalized graph Laplacian convolution kernel, $\boldsymbol{z}_u$ and $\boldsymbol{z}_i$ are the hidden layer vectors obtained through User-VAE and Item-GCN. $\boldsymbol{e}_u^{(0)}$ and $\boldsymbol{e}_i^{(0)}$ are initialized with $\boldsymbol{z}_u$ and $\boldsymbol{z}_i$, and $\gamma(\cdot)$ represents the cascade function to connect different layers. The mean function is used as the cascade function in VE-GCN. The final feature embeddings of users and items are defined as the mean values of each layer's embeddings after passing through another generalized graph Laplacian convolution kernel:

$$\boldsymbol{E}_u = \frac{1}{L}\sum_{l=0}^{L-1}\boldsymbol{\xi}(\boldsymbol{e}_u^{(l)}),\ \boldsymbol{E}_i = \frac{1}{L}\sum_{l=0}^{L-1}\boldsymbol{\xi}(\boldsymbol{e}_i^{(l)}), \tag{19}$$

where L is the total number of graph convolution layers. The inner product is used as the feature interaction function, and then the final rating prediction $\hat{\boldsymbol{R}}_{ui}$ is got through the users' and items' feature embeddings:

$$\hat{\boldsymbol{R}}_{ui} = \boldsymbol{E}_u^T\boldsymbol{E}_i. \tag{20}$$

3.4. Model training

It should be noted that VE-GCN is not an end-to-end model. The VAE part and GCN part are trained separately so that user-item interaction information can be learned in different manners through VAE and GCN, then the combination of the interaction information can boost the performance of VE-GCN.

The training of the GCN part uses the BPR loss [34], which is defined as follows:

$$L_{BPR} = -\sum_{u=1}^{M}\sum_{i\in N_u}\sum_{j\notin N_u} ln\sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \lambda\|\Delta^{(0)}\|^2, \tag{21}$$

where λ represents the regularization parameter, $\hat{y}$ represents the predicted rating between users and items, N_u represents the neighbor nodes of user u. It should be noted that only the parameters of the first embedding layer $\Delta^{(0)}$ need to be learned during the information propagation process of GCN. For more details refer to Algorithm 2.

3.5. Information loss problem

Regarding the information lossy initialization, we will give empirical proof in the experimental part, and here we will provide theoretical proof for the high-frequency information loss problem. The experimental part also proves that the loss of high-frequency information will cause a sharp decrease in the CF.

The Graph Laplacian matrix can be eigen-decomposed as $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T$. The conversion between the spatial domain signal $x \in \mathbb{R}^n$ and the frequency domain signal $\hat{x} \in \mathbb{R}^n$ can be completed by Fourier Transform and inverse Fourier Transform. The implementation process is as follows:

$$\hat{x} = \boldsymbol{U}^T x, \quad x = \boldsymbol{U}\hat{x}. \tag{22}$$

Algorithm 2 GCN Part Training Algorithm

Input: κ: the Laplacian convolution kernel coefficient; t: the Laplacian convolution kernel order; **R**: the user-item rating matrix; Ep: the epochs; **L**: the number of GCN layers;

```
Compute the hidden vector z_u, z_i by Eq.(3-10);
Initialize the first embedding parameters of users and items: e_u^(0), e_i^(0) by z_u, z_i;
Compute the generalized graph Laplacian convolution kernel by Eq.(3-17);
for iter ∈ 0,1,..., Ep do
    while k < L do
        Compute the embeddings e_u^(k), e_i^(k) by Eq.(3-18);
    end while
    Compute the final embeddings E_u, E_i by Eq.(3-19);
    Compute the rating: R̂ by Eq.(3-20);
    Compute loss ℒ and gradient Δℒ with R,R̂ by Eq.(3-21);
    Update e_u^(0), e_i^(0) by optimizer;
end for
```

Output: the final rating $\hat{\boldsymbol{R}}$ computed by Eq.(3-20);

Given the signal x in the spatial domain and the convolution kernel y, the graph convolution $*_G$ is defined as:

$$x *_G y = \boldsymbol{U}((\boldsymbol{U}^T x) \odot (\boldsymbol{U}^T y)). \tag{23}$$

$\boldsymbol{U}^T y$ is the convolution kernel in the frequency domain. Let $\boldsymbol{U}^T y = [\theta_0, \ldots, \theta_{n-1}]$, $g_\theta = \mathrm{diag}([\theta_0, \ldots, \theta_{n-1}])$:

$$x *_G y = \boldsymbol{U} g_\theta \boldsymbol{U}^T x. \tag{24}$$

The above formula is approximated by Chebyshev Polynomial to reduce the number of parameters of g_θ, and then we obtain ChebyNet [35], the number of parameters drops from n to K:

$$g_\beta(\Lambda) = \sum_{k=0}^{K-1}\beta_k\boldsymbol{\Lambda}^k. \tag{25}$$

$$x *_G y = \boldsymbol{U} g_\beta(\boldsymbol{\Lambda})\boldsymbol{U}^T x = \sum_{k=0}^{K-1}\beta_k\boldsymbol{L}^k x. \tag{26}$$

In GCN [7], it has K = 2, then they reduce the number of parameters through the first-order Chebyshev Polynomial. However, only the low-order eigenvalues are fitted at this time, making the essence of GCN a kind of low-pass filter and causing GCN to be a calculation process with a high-frequency information loss problem.

The propagation rules of GCN [7] are as follows:

$$x *_G y = \beta_0 x + \beta_1 \boldsymbol{L} x. \tag{27}$$

Assume $\beta = \beta_0 = -\beta_1$:

$$x *_G y = \beta(\boldsymbol{I} - \boldsymbol{L})x. \tag{28}$$

Continue to give the definition of t-order graph convolution, which is used to verify the low-pass filtering effect of the GCN convolution kernel in the experimental part:

$$x *_G y = [\beta(\boldsymbol{I} - \boldsymbol{L})]^t x = [\beta(\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}})]^t x. \tag{29}$$

In summary, the information lossy initialization of GCN-based CF due to ignoring user-item interaction information can be solved by VAE enhancement, VAE is used to model a variety of interaction patterns as the prior information to strengthen the learning process of GCN. And the problem of high-frequency information loss caused by Chebyshev Polynomial fitting can be solved by using the generalized Laplacian convolution kernel that can preserve high-frequency information. It should be mentioned that due to the non-frequency-specific learning method of VAE, VAE can acquire both low- and high-frequency information. Then the generalized graph Laplacian convolution kernel, which introduces frequency response function, is used to perform convolution to process the low- and high-frequency information simultaneously. The current information is obtained from both VAE and graph topology through graph convolution.

4. Experiment

To verify the effectiveness of VE-GCN, we conduct a series of experiments to verify the following research questions. It should be noted that the experimental figures on different datasets that are not shown in this paper show consistent results.

- **RQ1:** Is there an information loss problem in GCN-based CF?
- **RQ2:** Is the addition of the VAE part helping the sufficient learning of the interactive information?
- **RQ3:** How does the generalized graph Laplacian convolution kernel affect VE-GCN?
- **RQ4:** Which information propagation mode is suitable for VE-GCN?
- **RQ5:** How does the disentanglement in VAE affect VE-GCN?

4.1. Baselines and datasets

The datasets used in the experiments include AMusic, LastFM,[1] Ml-1M,[2] and Yelp2018, of which the AMusic and Yelp2018 are from DeepCF [36] and LightGCN [11]. For Ml-1M, we discard the last item in the list of items that interacted by users, and the discarded items do not participate in training and testing. For LastFM, we use "User-Listened Artist Relations" of "Hetrec2011-lastfM-2K" as the user-item interaction matrix, where "artist" is used as the item here. The ratio between the training set and testing set of AMusic, LastFM, and ML-1M is 9:1. For larger dataset: Yelp2018, is 8:1. We randomly divide the dataset twice, conduct three experiments on each randomly divided dataset, and take the mean and variance of the final experimental results. The specific details of the dataset are shown in Table 1. All datasets are with explicit ratings, and we regard all user ratings of items as positive interactions, even low ratings are considered positive interactions. The reason for this is that rating information is difficult to obtain under realistic conditions, interaction logs are the source of information in most cases. To verify the models under realistic conditions, we only regard the dataset as interaction logs and do not care about the specific ratings. However, we will perform a translation experiment between ratings and positive interactions in a later section.

- NGCF [10]: NGCF learns users' and items' embeddings by using complex GCN on the user-item interaction graph, finally uses the inner product of the learned embeddings to predict the ratings. However, the process of redundant feature transformation and nonlinear activation function in NGCF increases the complexity and degrades the performance of the model.

[1] https://grouplens.org/datasets/hetrec-2011/.

[2] https://grouplens.org/datasets/movielens/.

Table 1
Details of the datasets used in the experiments.

Dataset	User	Item	Interactions	Density
AMusic	1776	12,929	46,087	0.00201
LastFM	1892	17,632	92,834	0.00278
Ml-1M	6040	3706	994,169	0.04441
Yelp2018	31,668	38,048	1,561,406	0.00130

- LightGCN [11]: LightGCN migrates GCN to the collaborative filtering, in which GCN only retains the most important part for CF: neighbor aggregation. Nevertheless, LightGCN has the problem of information loss caused by fitting the convolution kernel with the first-order Chebyshev Polynomial.
- MultiVAE [32]: MultiVAE uses VAE for collaborative filtering, which can surpass the modeling capabilities of linear models. MultiVAE is a polynomial likelihood generation model that uses variational Bayesian inference to infer parameters. However, MultiVAE's VAE-based infrastructure is too simple to capture the potential user-item relationship well.
- LR-GCCF [12]: LR-GCCF combines simple graph convolution and residual connection. The residual connection can alleviate the over-smoothing problem in the process of graph convolution. Nevertheless, LR-GCCF has the problem of using ID embeddings to initialize and the information loss caused by the improper selection of the convolution kernel.

4.2. Experimental metrics

- RECALL @Y [37]:

$$\text{RECALL}_u @Y = \frac{|\text{Rel}_u \cap \text{Rec}_u|}{\text{Rel}_u}, \tag{30}$$

where Rel_u represents the item set related to user u, and Rec_u represents the top Y items recommended to user u. The intersection of Rel_u and Rec_u is divided by the number of Rel_u ($\text{Rel}_u = Y$) to get $\text{RECALL}_u @Y$. Average the $\text{RECALL}_u @Y$ of each user to get the final result RECALL @Y.

- NDCG @Y [38]:

$$\begin{aligned} \text{DCG}_Y &= \sum_{i=1}^{Y} \frac{2^{\text{rel}_i} - 1}{\log_2(i+1)}, \\ \text{IDCG}_Y &= \sum_{i=1}^{Y} \frac{1}{\log_2(i+1)}, \\ \text{NDCG}_u @Y &= \frac{\text{DCG}_Y}{\text{IDCG}_Y}, \end{aligned} \tag{31}$$

where rel_i represents the relevance of the recommended item to user u at the top i position. Average the $\text{NDCG}_u @Y$ of each user to get the final result NDCG @Y.

For the metrics of our experiments, we set topY to top20, which is a balanced choice. Such a choice can take into account the recommendation effect of a sufficient number of items, and will not make the items at the end of the recommendation list unnecessary because of the excessive number of Y.

4.3. Parameters setting

In MultiVAE, the learning rate is $1e-3$, the decoder dims are $(200, 600)$, the optimizer is Adam. In LR-GCCF, the learning rate is $1e-4$, the embedding size is 64, the optimizer is Adam. In LightGCN, the learning rate is $1e-3$, the embedding size is 64, the GCN layer is 3, the optimizer is Adam. In NGCF, the learning

Table 2
Comparison of the models on LastFM, AMusic, Ml-1M, and Yelp2018. VE-GCN-nD represents VE-GCN without using DenseGCN. The best results are indicated in bold, and the second-best results are underlined. The calculation of Improv. is based on Light-GCN.

		NGCF	LightGCN	MultiVAE	LR-GCCF
LastFM	RECALL@20	0.2700 ± 1.02e−05	0.2971 ± 6.63e−06	0.2573 ± 8.06e−07	0.2736 ± 1.19e−04
	NDCG@20	0.2203 ± 3.60e−06	0.2468 ± 2.76e−06	0.2000 ± 1.23e−06	0.2245 ± 7.82e−05
AMusic	RECALL@20	0.0716 ± 2.37e−05	0.0944 ± 6.62e−06	0.0836 ± 1.90e−06	0.0924 ± 9.16e−06
	NDCG@20	0.0442 ± 1.43e−05	0.0624 ± 3.18e−06	0.0522 ± 2.81e−06	0.0587 ± 4.49e−06
Ml-1M	RECALL@20	0.2728 ± 2.85e−06	0.2855 ± 4.88e−06	0.2714 ± 4.39e−06	0.2688 ± 1.51e−06
	NDCG@20	0.2839 ± 9.20e−06	0.2975 ± 1.10e−05	0.2662 ± 1.33e−05	0.2844 ± 1.24e−05
Yelp2018	RECALL@20	0.0834 ± 9.34e−08	0.0945 ± 4.12e−07	0.0807 ± 1.08e−07	0.0849 ± 2.89e−08
	NDCG@20	0.0718 ± 4.61e−08	0.0821 ± 6.82e−08	0.0690 ± 5.16e−07	0.0719 ± 1.56e−08
		VE-GCN-nD	VE-GCN	Improv.	
LastFM	RECALL@20	0.3054 ± 7.95e−08	**0.3067 ± 4.42e−07**	3.23%	
	NDCG@20	0.2528 ± 4.42e−06	**0.2540 ± 2.11e−07**	2.92%	
AMusic	RECALL@20	**0.1055 ± 3.81e−06**	0.1043 ± 1.88e−06	11.76%	
	NDCG@20	**0.0665 ± 8.39e−07**	0.0659 ± 6.10e−07	6.57%	
Ml-1M	RECALL@20	0.2911 ± 9.39e−06	**0.2929 ± 1.57e−10**	2.59%	
	NDCG@20	0.3010 ± 1.04e−05	**0.3051 ± 6.19e−08**	2.55%	
Yelp2018	RECALL@20	0.0974 ± 3.67e−07	**0.0998 ± 1.27e−07**	5.61%	
	NDCG@20	0.0844 ± 2.33e−08	**0.0863 ± 1.47e−08**	5.12%	

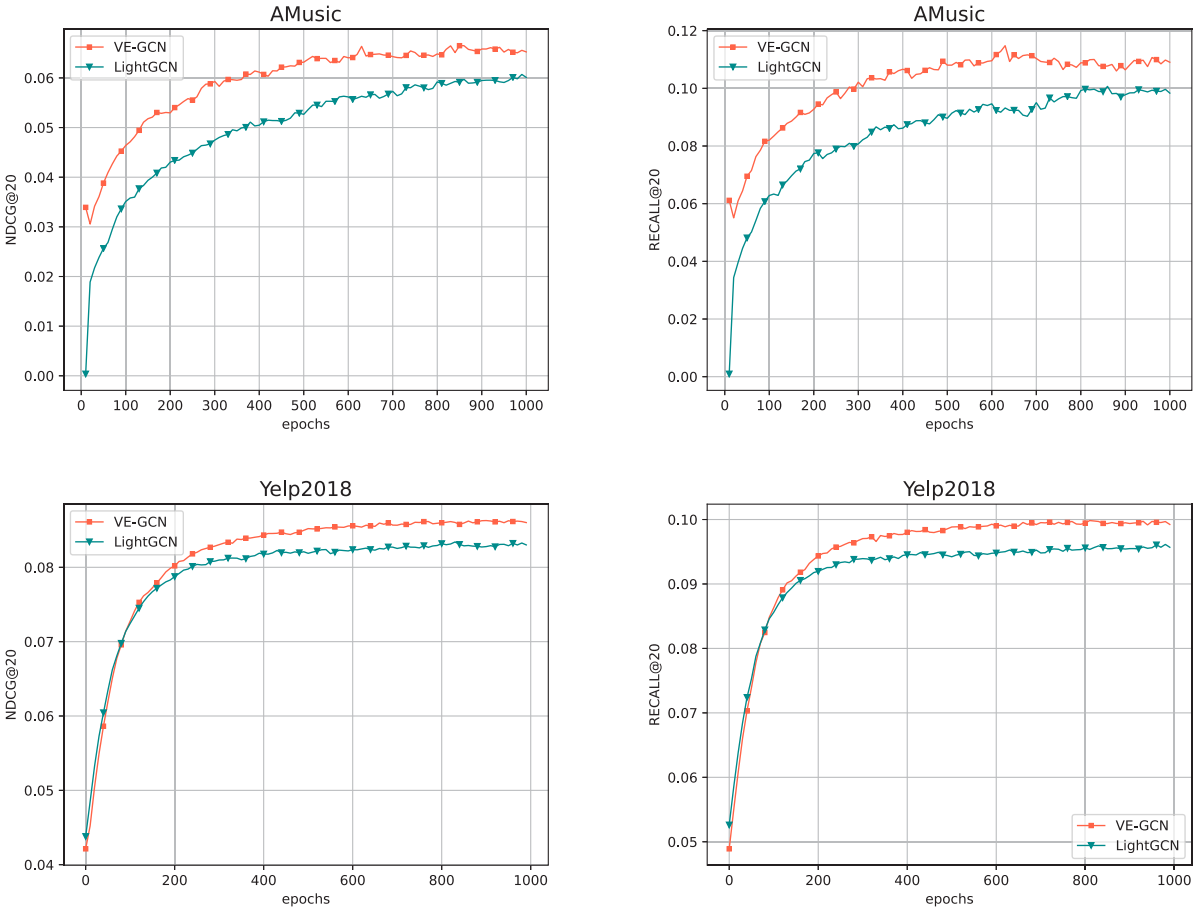


Fig. 3. Comparative experiments between VE-GCN on AMusic and Yelp2018.

rate is $1e-4$, the embedding size is 64, the optimizer is Adam. All baseline parameters are consistent with the official ones. In VE-GCN, we set KL regularization weight $\beta = 0.2$ of VAE part; L2 regularization weight $\lambda = 1e-4$ of GCN part; scaling value $\tau = 10$; generalized graph Laplacian convolution kernel order $t = 1$; generalized graph Laplacian convolution kernel coefficient κ are [0.5, 0.9, 0.9, 0.8] on AMusic, LastFM, Ml-1M and Yelp2018 respectively. User-VAE encoder dimension: [N-800-64*2]; Item-VAE encoder dimension: [M-800-64*2], which means that K is set to be 64; User-VAE training epoch is 200; Item-VAE training epochs are [1000,500,500,500] for AMusic, LastFM, Ml-1M and Yelp2018 respectively; the number of GCN layers are [4,5,5,5] on AMusic, LastFM, Ml-1M and Yelp2018 respectively. The dimension of the VAE decoder is symmetrical: if the dimension of the encoder is [N-800-64*2], then the dimension of the decoder is [64-800-N]. The weight initialization method of the VAE part is the Xavier initialization [39], and the bias initialization method is the normal distribution initialization. The cascade function $\gamma(\cdot)$ of the GCN part is the mean function. The optimizer of the VAE part and the GCN part is Adam [40], and the learning rate is set to 1e–3. The selection of hyperparameters is the result of a small-scale grid search. Better experimental results can be obtained if a more

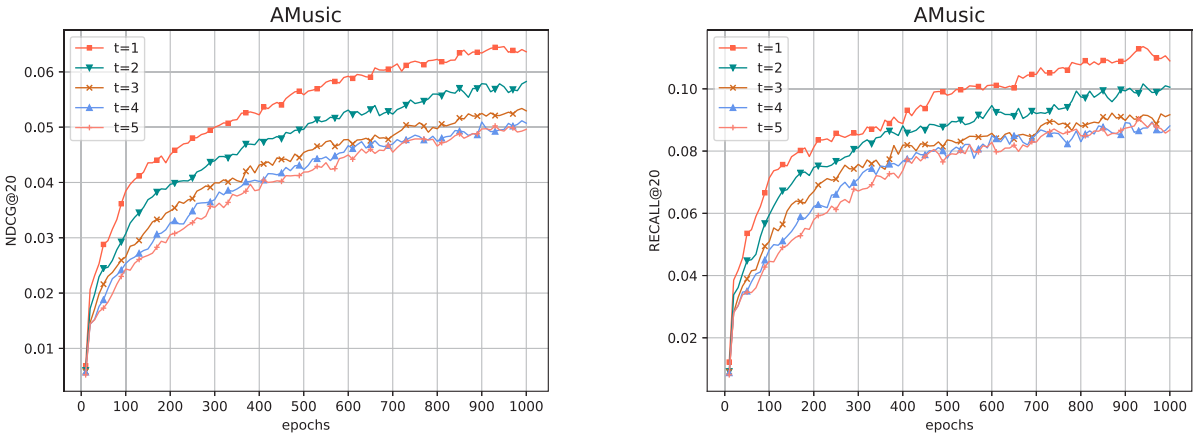


Fig. 4. VE-GCN's high-order convolution kernel experiment on AMusic, where t represents the order of the convolution kernel, and κ is set to 1 uniformly.

comprehensive grid search is conducted. Experiments are run for 1000 epochs or stopped after 50 epochs with no performance improvement.

4.4. Performance analysis

4.4.1. Model comparison experiment

The comparative experimental results of the model are shown in Table 2. We compare the performances of VE-GCN with other baseline models. The experimental results of Table 2 show that the performance of VE-GCN on each dataset surpasses other baseline models. It also proves that the addition of DenseGCN can improve the performance of VE-GCN generally, because DenseGCN integrates the embeddings of multiple layers to achieve multi-layer information interaction. LightGCN is currently the most advanced collaborative filtering model based on graph convolutional network. Experiments (see Fig. 3) show that the performance of VE-GCN on each dataset is better than LightGCN.

4.4.2. GCN-based CF information loss problem (RQ1)

There are two information loss problems in GCN-based CF. Firstly, the use of ID embeddings with weak semantic information to initialize network nodes causes information lossy initialization. The ID embeddings used for GCN-based CF initialization are generally set to obey the standard normal distribution. The universality assumption of this standard normal distribution initialization may not be consistent with every dataset. We will continue to discuss this problem in Section 4.4.3. Secondly, the use of first-order Chebyshev Polynomial to fit the convolution kernel causes the loss of high-frequency information. About the loss of high-frequency information, Fig. 4 shows that high-pass signals are suppressed in the graph convolution process, when the order t increases, the filtering ability of the low-pass filter increases, and more signals are suppressed, resulting in further degradation of model performance.

If we can propose solutions to the two information loss problems verified above, we will be able to obtain more appropriate node representations, thereby obtaining better recommendation prediction results.

4.4.3. VAE information mining experiment (RQ2)

The existing GCN-based CF is generally initialized with the standard normal distribution $N(0, 1)$, which is on account of the central limit theorem. But in specific situations, we can further mine the interaction matrix to provide prior information for the information aggregation process of GCN. So we think of using VAE to enhance the network.

The bottleneck vectors of the VAE part are actually normal distributions fitted by the network, which also conform to the central limit theorem. VAE captures the interactive behavior patterns between users and items as the initial prior information, and the prior information represents the node characteristics of users and items in the graph to some extent.

It can be seen from the experimental results in Fig. 5 that the spatial distributions of the information formed by the VAE part of VE-GCN are more stable and more convergent than the standard normal distribution. The experimental results of Fig. 6 prove that the bottleneck vectors of the VAE part provide different hypothetical distributions for each user and item. It shows that the VAE part can provide non-universal prior hypotheses for VE-GCN. The characteristic information of users and items learned from the hidden layer distributions is provided to the GCN part for the nodes' embeddings learning.

The experimental results in Fig. 7 verify our above conclusions. Normal-GCN has no VAE part and uses the standard normal distribution for initialization. Fig. 7 proves the initialization information loss problem of GCN-based CF can be solved well by VAE enhancement. The addition of the VAE part improves the performance of the model significantly, indicating that the VAE part can extract the interactive mode of users and items effectively.

4.4.4. Generalized graph Laplacian convolution kernel (RQ3)

Using a generalized graph Laplacian convolution kernel adapted to the data distribution of different datasets can improve the performance of VE-GCN (see Table 3), and can retain the topology information to a greater extent, then alleviate the information loss of GCN-based CF.

After experimental verification, the maximum eigenvalues λ_{max} obtained by the Laplacian matrix decomposition are [1, 1, 1] on the Amusic, LastFM, and Ml-1M separately, and the selected optimal Laplacian convolution kernel coefficients κ are [0.5,0.9,0.9] respectively. When $\kappa < \frac{1}{\lambda_{max}}$, making the generalized graph Laplacian convolution kernel a low-pass filter essentially but still retains part of the high-frequency information, which further reduces the information loss problem caused by the first-order Chebyshev Polynomial approximation.

According to Fig. 8, it can be seen that the performance of the cooperative work of using VAE and generalized Laplacian graph convolution kernel on VE-GCN is the best. And Fig. 8 also reflects that the performance of nV+K is not necessarily better than nV+nK. On AMusic, Ml-1M, and Yelp2018, the performance of nV+K is better than nV+nK, while the result is slightly worse on LastFM. It shows that the effectiveness of the high-frequency

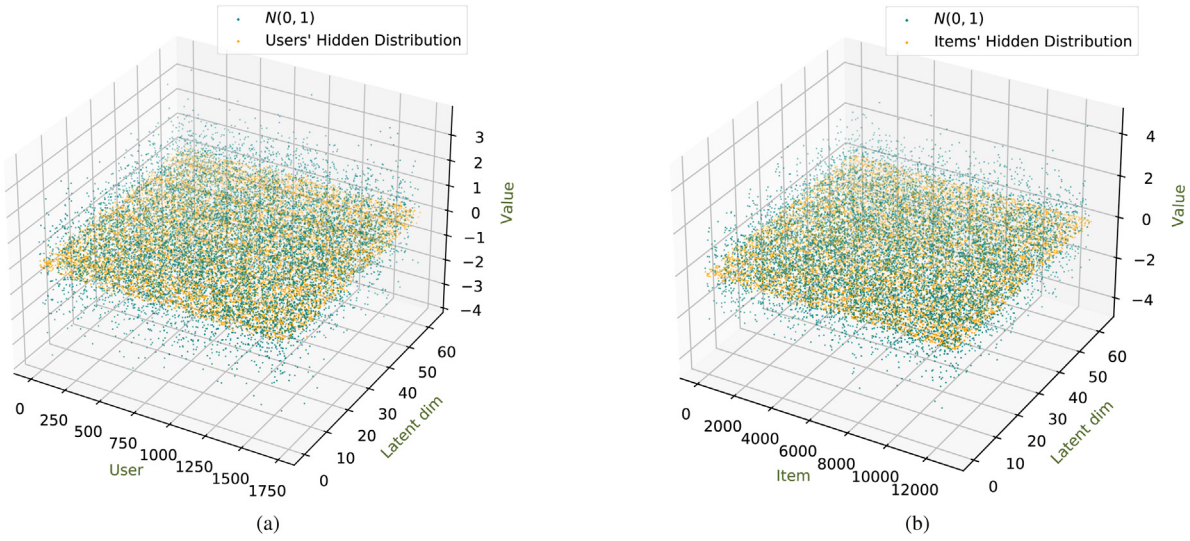
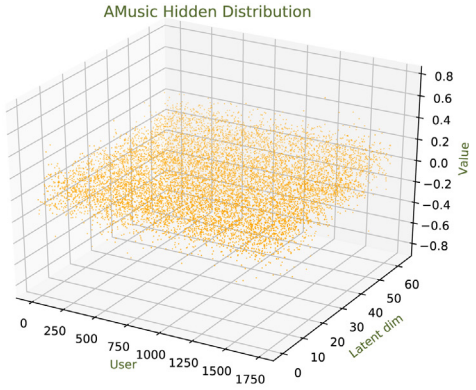
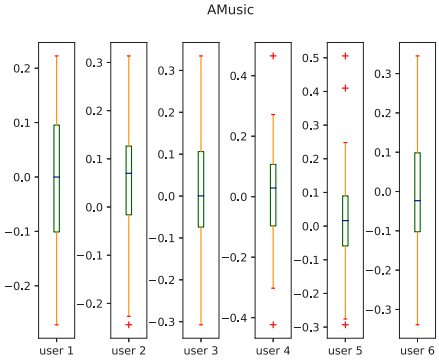


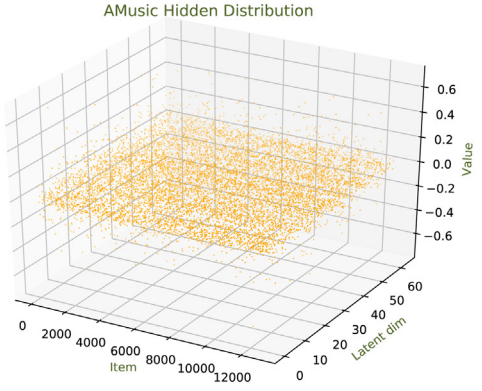
Fig. 5. Comparison of the distributions obtained through the VAE part of VE-GCN and the standard normal distribution on AMusic.



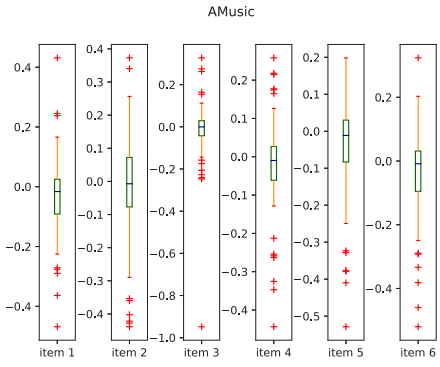
(a) Users' hidden distribution on AMusic.



(b) Hidden distribution of users: $\{u_1, u_2, u_3, u_4, u_5, u_6\}$ on AMusic.



(c) Items' hidden distribution on AMusic.



(d) Hidden distributions of items: $\{i_1, i_2, i_3, i_4, i_5, i_6\}$ on AMusic.

Fig. 6. Hidden distributions learned by the VAE part of VE-GCN on AMusic.

information and the low-frequency information for CF in different datasets is inconsistent. However, high-frequency information is still beneficial in general. According to Fig. 9, it can be seen that no matter what the value of k is, the effect of VE-GCN is better. The user and item representations learned by VAE can be better operated by the generalized graph Laplacian convolution.

4.4.5. Ablation experiment of GCN propagation mode in VE-GCN (RQ4)

The results of ablation experiments (see Fig. 10) prove that GCN without feature transformation and nonlinear activation function is more suitable for VE-GCN. This is also observed in LightGCN, which is initialized with ID embedding with weak

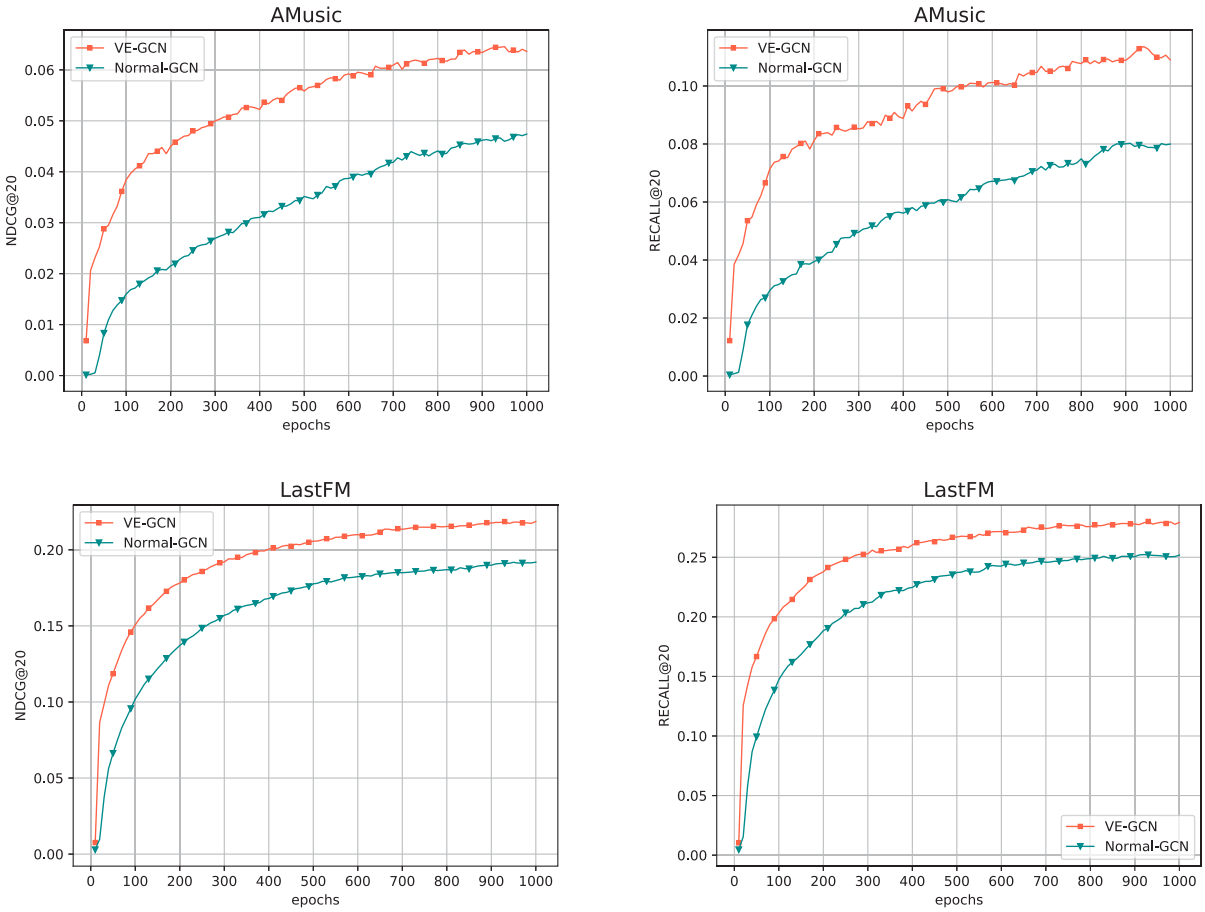


Fig. 7. Comparison experiment of VE-GCN and Normal-GCN on AMusic and LastFM, κ is set to 1 uniformly.

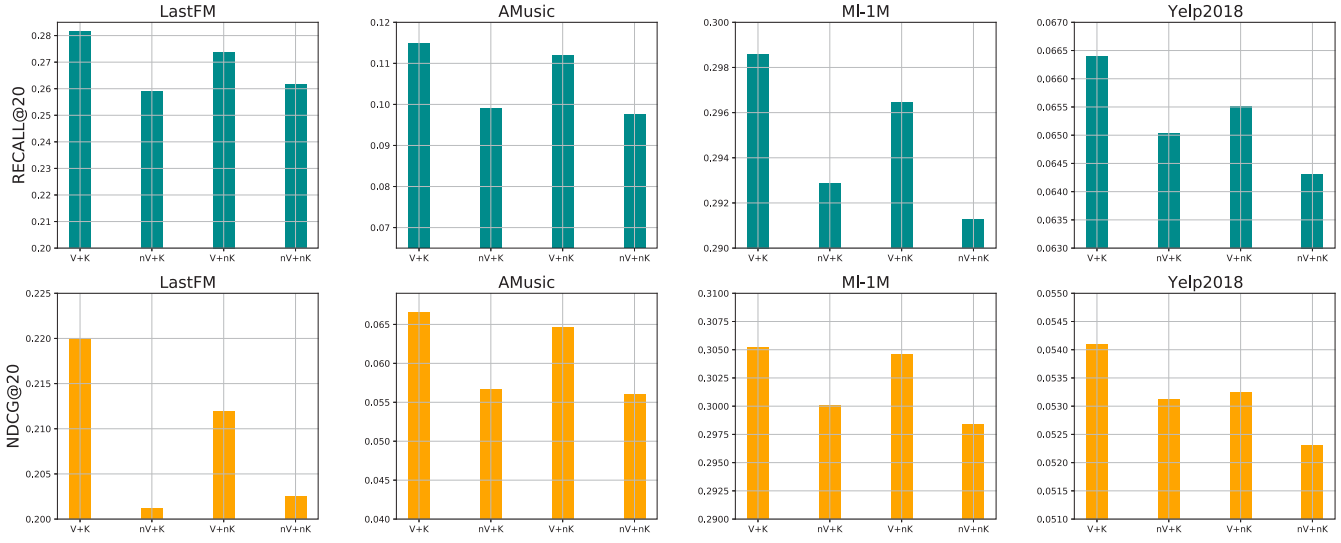


Fig. 8. Comparison of ablation experiments of VE-GCN on LastFM, AMusic, Ml-1M, Yelp2018. V+K represents VE-GCN, nV+K represents VE-GCN without enhancing by VAE, V+nK represents VE-GCN without using generalized graph convolution kernel, nV+nK represents VE-GCN without enhancing by VAE and using generalized graph convolution kernel. Experiments related to k have taken the best results.

semantics. In this way, the feature transformation and nonlinear activation function in GCN are only suitable for data with rich semantic information. If used in CF where the input data has no actual semantics, it will bring higher model complexity and lead to performance degradation. In addition, the experimental results also prove that our research direction for graph convolution filters is correct. For CF, we should focus on processing the signals on the graph instead of feature space conversion and nonlinear activation function, and it also provides new thinkings and directions for future work.

4.4.6. Explanation of disentanglement in VE-GCN (RQ5)

According to Fig. 11, we can see that when β increases, the latent distribution of the VAE part of VE-GCN becomes more

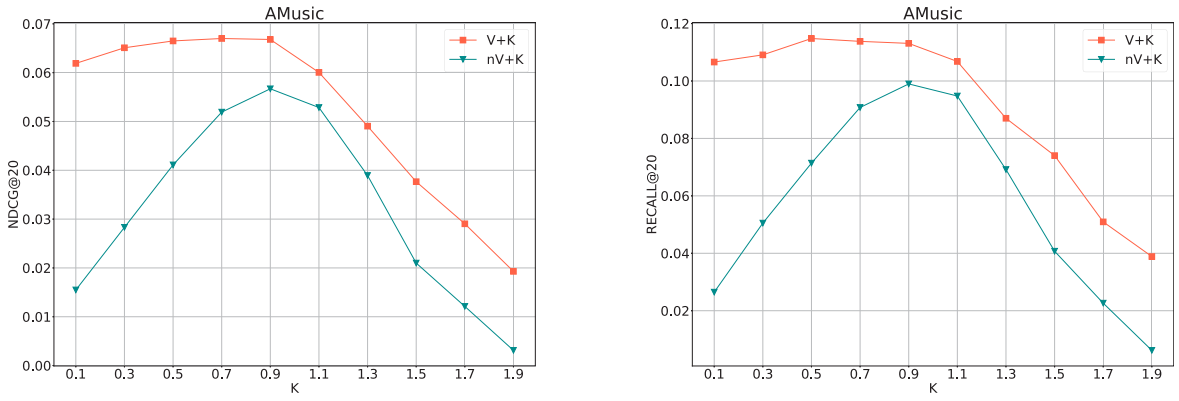


Fig. 9. Comparison experiment of VE-GCN and Normal-GCN with different k on AMusic.

Table 3
Influence of different generalized graph Laplacian convolution kernel coefficient k on VE-GCN. The best results are indicated in bold.

Dataset	Metrics	k = 0.1	k = 0.2	k = 0.3	k = 0.4	k = 0.5	k = 0.6	k = 0.7	k = 0.8	k = 0.9	k = 1.0
AMusic	RECALL@20	0.1066	0.1094	0.1091	0.1101	**0.1148**	0.1125	0.1138	0.1128	0.1131	0.1136
	NDCG@20	0.0619	0.0648	0.0651	0.0660	0.0665	0.0669	**0.0670**	0.0665	0.0668	0.0646
LastFM	RECALL@20	0.2652	0.2687	0.2713	0.2753	0.2766	0.2807	0.2815	**0.2816**	0.2815	0.2801
	NDCG@20	0.2047	0.2092	0.2117	0.2140	0.2156	0.2179	0.2189	0.2195	**0.2200**	0.2187
Ml-1M	RECALL@20	0.2605	0.2661	0.2707	0.2757	0.2762	0.2836	0.2858	**0.2922**	0.2918	0.2920
	NDCG@20	0.2613	0.2662	0.2696	0.2731	0.2760	0.2826	0.2854	**0.3098**	0.2883	0.2893

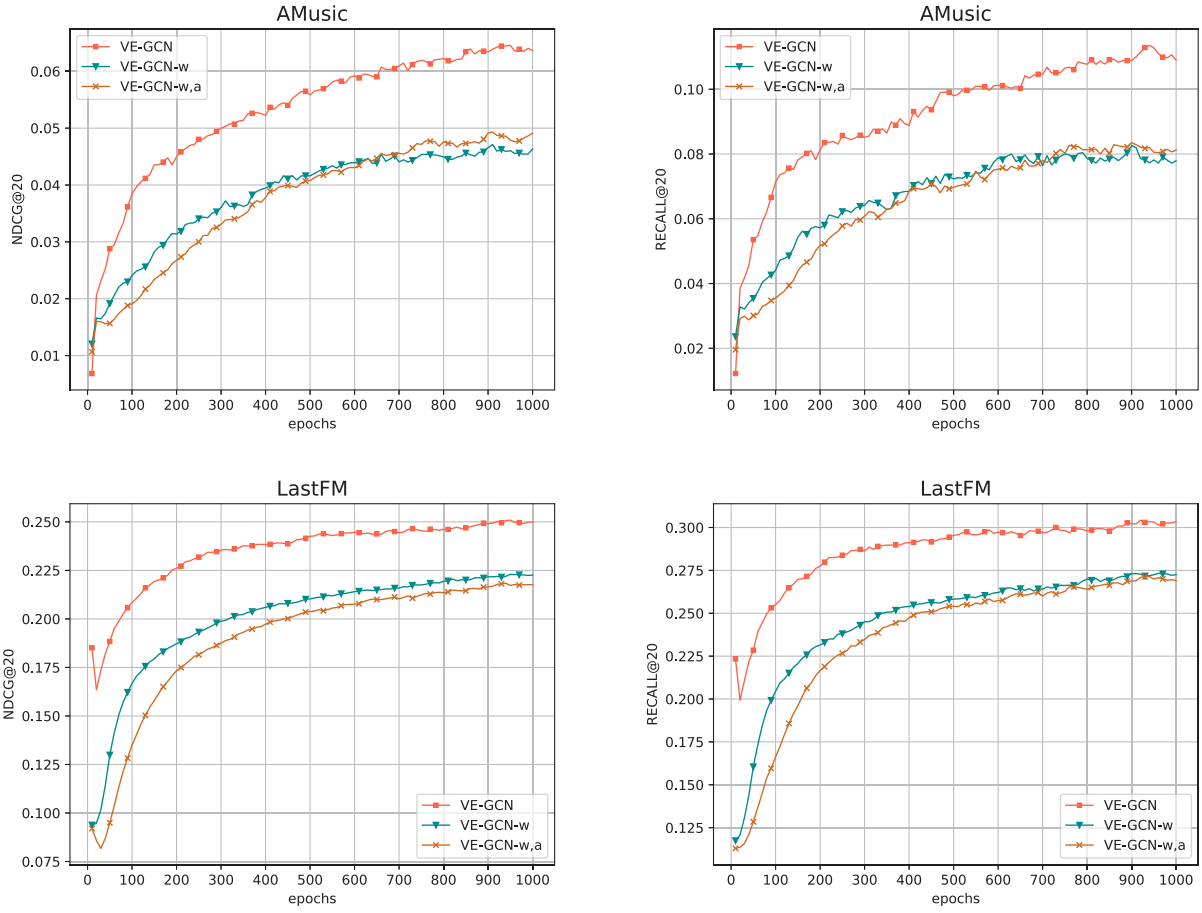
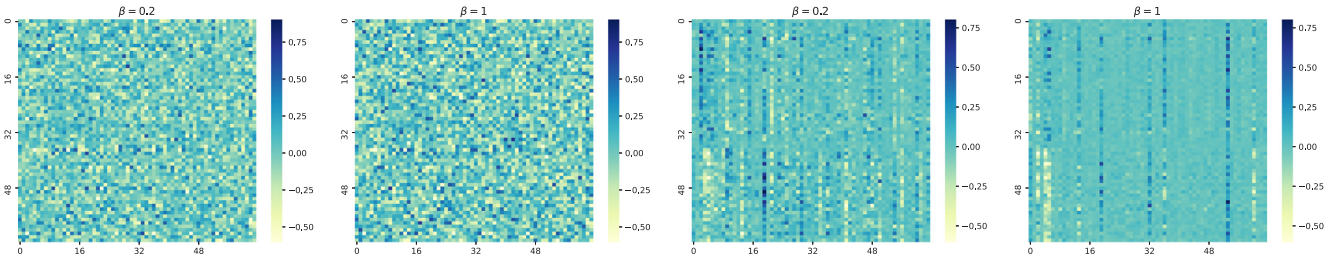


Fig. 10. VE-GCN ablation experiment on AMusic and LastFM. Where VE-GCN-w means VE-GCN with feature transformation; VE-GCN-w,a means VE-GCN with feature transformation and nonlinear activation function. κ is set to 1 uniformly.

separated, indicating that the disentanglement of the distribution increases. Experiments (see Fig. 12) show that we can achieve different effects by controlling hyperparameter β. When $\beta < 1$ and the KL term is weakened, the disentanglement decreases,



(a) Users-LastFM (b) Users-LastFM (c) Items-LastFM (d) Items-LastFM

Fig. 11. The heat map of the latent distributions of the VE-GCN's VAE part with different KL term coefficient β on LastFM. Users-LastFM refers to Users' hidden distribution of LastFM, and Items-LastFM refers to Items' hidden distribution of LastFM.

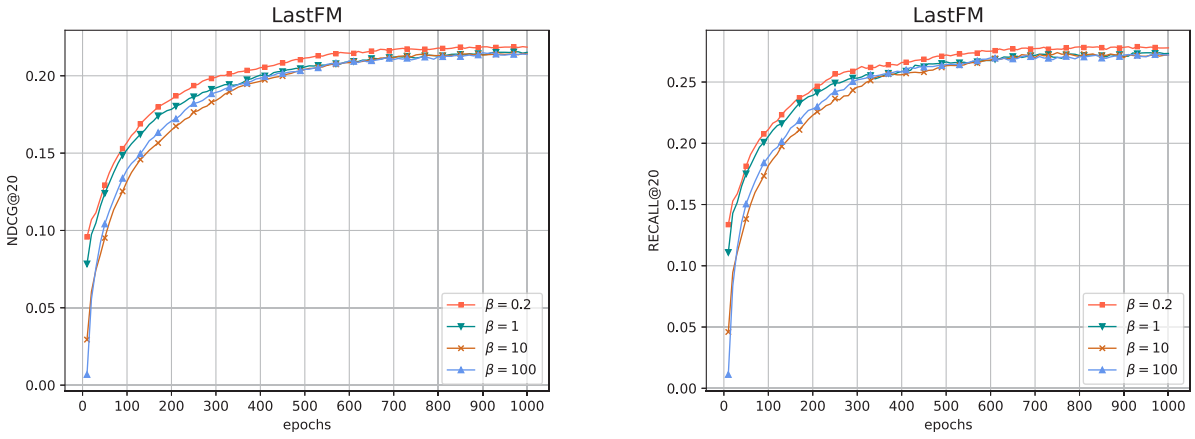


Fig. 12. Influence of different KL term coefficient β on VE-GCN on LastFM.

Table 4
Comparison of training and testing time of the models on LastFM, AMusic, Ml-1M, and Yelp2018.

		Train(s)/epoch	Test(s)/epoch
NGCF	LastFM	1.1954	0.0701
	AMusic	0.4981	0.0507
	Ml-1M	40.0697	0.0929
	Yelp2018	89.5185	2.6725
LightGCN	LastFM	0.2111	0.0014
	AMusic	0.1096	0.0012
	Ml-1M	11.3632	0.0047
	Yelp2018	24.1212	0.0168
MultiVAE	LastFM	0.0398	0.0016
	AMusic	0.0216	0.02681
	Ml-1M	0.0297	0.0044
	Yelp2018	0.8973	0.2191
LR-GCCF	LastFM	0.0679	0.1300
	AMusic	0.0374	0.0668
	Ml-1M	0.7918	0.0997
	Yelp2018	1.1646	3.2501
VE-GCN	LastFM	0.4244	0.0095
	AMusic	0.1897	0.0148
	Ml-1M	27.3771	0.1878
	Yelp2018	57.5635	1.4153

which means that the degree of separation of the attributes in the latent distribution is low, and the degree of fusion is high. When $\beta > 1$, the increase of the disentanglement means the degree of separation of the attributes in the latent distribution is high, and the attributes are independent mainly. This increase of disentanglement is indeed beneficial to improve the generation ability of models, but we pay more attention to the learning of the latent distribution in VE-GCN. We hope to extract the distribution that better describes the multiple interaction patterns of users and items. Therefore, the increase of disentanglement is of no use to us and will affect the effect of VE-GCN.

4.4.7. Numerical ratings and positive interactions conversion experiment

In previous experiments, we regard all user ratings of items as positive interactions, even low ratings are considered positive interactions. We now conduct experiments with different ratings as positive interactions.

Datasets used contains ratings from one to five, so we conduct experiments with rating $\geq$1, …, rating $\geq$5 are regarded as positive interactions (see Fig. 13). It can be seen that the rating conversion experiments on the two datasets show different results, indicating the results depend on the specific dataset. NDCG and RECALL of Ml-1M show two opposite performances: for RECALL, we have $5 > 4 > 3 > 2 > 1$, while for NDCG, we have $1 > 2 > 3 > 4 > 5$. NDCG and RECALL of AMusic are not exactly the opposite: for RECALL, we have $3 > 5 > 1 > 2 > 4$, while for NDCG, we have $1 > 3 > 2 > 4 > 5$. As a general result, low-rating conversion is more favorable for NDCG, which focuses on the sorting order, while high-rating conversion is more favorable for RECALL. High-rating means users have great willingness to choose the corresponding items, which has great advantages for RECALL focusing on the overall results.

4.4.8. Time analysis of methods

The equipment for our experiments is *NVIDIA GeForce RTX 2080 Ti*. Referring to Table 4, we can see the results of training time and testing time: NGCF > VE-GCN > LightGCN.

We mention that VE-GCN is a decoupling model, where the training of User-VAE and Item-VAE will bring additional time consumption. In Fig. 14, we show the percent of the additional

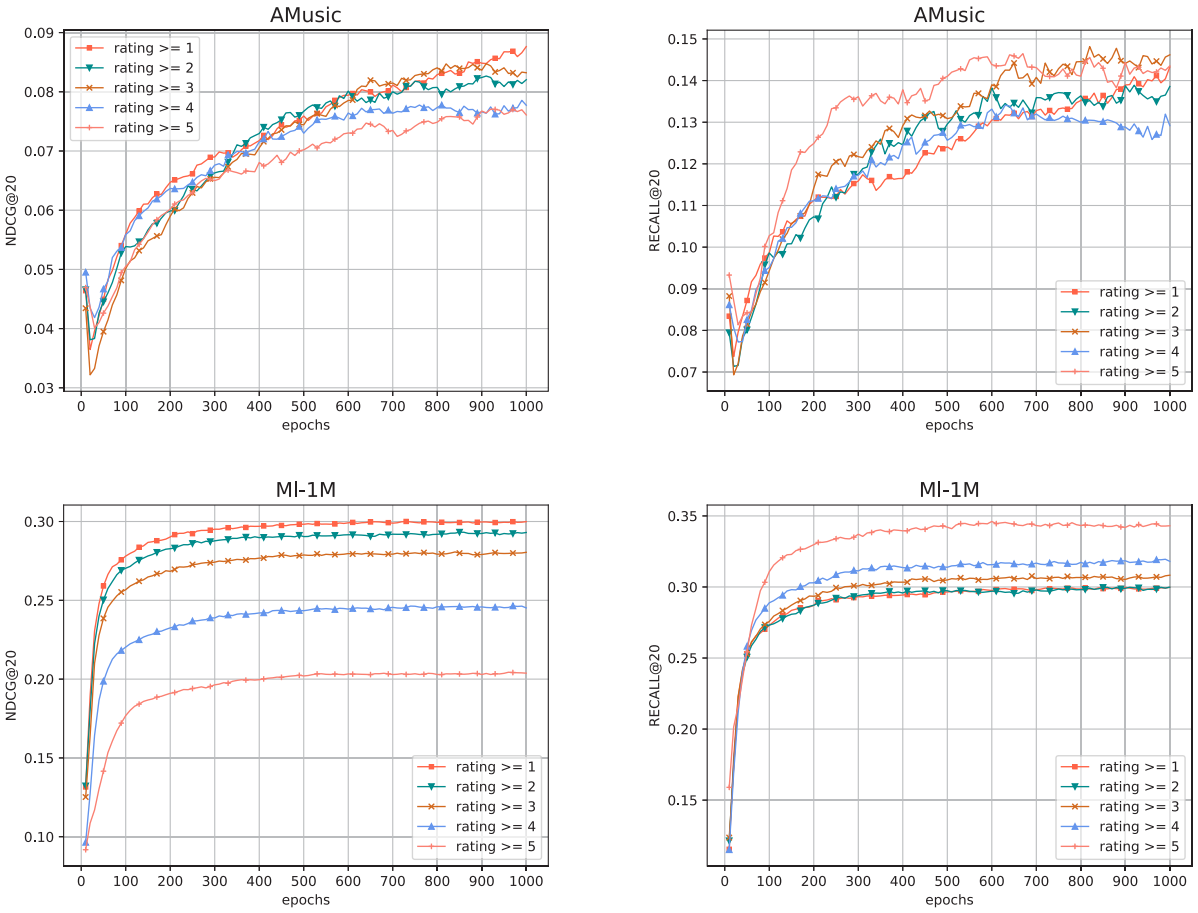


Fig. 13. Numerical ratings and positive interactions conversion experiments on AMusic and Ml-1M.

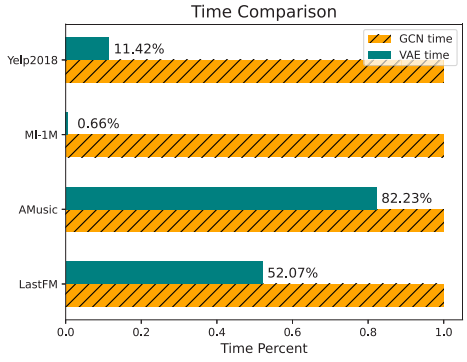


Fig. 14. The percent of User-VAE+Item-VAE training time to the training time of GCN part of VE-GCN on LastFM, AMusic, Ml-1M, Yelp2018.

time consuming on average to each epoch. Especially on Ml-1M and Yelp2018, which take a long time to train, almost no additional time consuming is brought. For AMusic and LastFM, the training time of both is relatively short, so the increased training time would not cause a computational burden.

5. Conclusion and future work

We propose the information loss problems of GCN-based CF for the first time. Then we propose a novel framework named VE-GCN, which combines GCN and VAE to optimize the information propagation framework of CF. Specifically, we use the embeddings of VAE as the prior information to learn the interactive information sufficiently. Then we use the generalized graph Laplacian convolution kernel for the graph convolution to process the low- and high-frequency information simultaneously. Therefore, the information loss problems in GCN-based CF are alleviated, which is caused by insufficient learning of the interactive information and fitting graph convolution kernel with the low-order Chebyshev Polynomial. At the same time, we optimize the information propagation method of GCN. Specifically, we remove redundant feature transformation and nonlinear activation function to simplify the calculation and to improve the performance of VE-GCN, and DenseGCN is introduced to achieve multi-level information interaction. Experiments show that our method achieves state-of-the-art performance, and a feasible method to handle the information loss problems in GCN-based CF is proposed for the first time.

Time complexity is always an issue, we will continue to figure out the optimization method of the algorithm in future work. And there may be other undiscovered information loss problems of GCN-based CF, if they can be found and solved, better recommendation results will be obtained. And we will consider other collaborative methods based on AE and GCN. In the tightly coupled working mode, there are two directions of information flow interaction, so a better information modeling effect can be obtained. We will consider the tightly coupled working mode of AE and GCN for collaborative filtering. In addition, the pre-training of GCN is one of the potential methods to solve the sparse data problem in graph learning tasks, and we will further consider pre-training methods based on the generative model or based on comparative learning to improve the performance of the GCN-based collaborative filtering methods.

Table 5
Notations.

Notation	Meaning
$\mathbf{R}$	Interaction matrix between users and items
$\|M\|$	Number of users
$\|N\|$	Number of items
z_u	Bottleneck vector of User-VAE
z_i	Bottleneck vector of Item-VAE
$e_u^{(l)}$	Users' embedding of the GCN part's lth layer
$e_i^{(l)}$	Items' embedding of the GCN part's lth layer
V	Set of vertices
E	Set of edges
$G = (V, E)$	Undirected simple graph
T	Number of nodes in the graph
$\mathbf{A}$	Adjacency matrix of the graph
$\mathbf{D}$	Degree matrix of $\mathbf{A}$
$\mathbf{L}_{\text{sym}}$	Symmetric normalized graph laplacian matrix
$\mathbf{I}$	Identity matrix
$\mathbf{U}$	Eigenvector matrix
Λ	Eigenvalue matrix
$\tilde{\mathbf{A}}$	$\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$
$\tilde{\mathbf{D}}$	Degree matrix of $\tilde{\mathbf{A}}$
$\mathbf{Q}^{(l)}$	Feature embedding matrix of lth layer of GCN
$\mathbf{W}^{(l)}$	Feature transformation matrix of lth layer of GCN
σ	Activation function
x_u	User input
x_i	Item input
ϕ	Set of inference parameters
θ	Set of generate parameters
β	Regularization hyperparameter
μ_u	Mean fitted by User-VAE
σ_u^2	Variance fitted by User-VAE
μ_i	Mean fitted by Item-VAE
σ_i^2	Variance fitted by Item-VAE
ϵ	Standard normal distribution $N(0, 1)$
τ	Scaling hyperparameter of ϵ
$\mathbf{H}$	Generalized Laplacian smoothing filter
$\boldsymbol{\xi}$	Complete generalized graph Laplacian convolution kernel
$\mathbf{E}_u$	Final feature embedding of users
$\mathbf{E}_i$	Final feature embedding of items
$\hat{\mathbf{R}}_{ui}$	Final rating prediction

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix

We provide a symbol table you may want to consult (see Table 5).

References

[1] Badrul Sarwar, George Karypis, Joseph Konstan, John Riedl, Item-based collaborative filtering recommendation algorithms, in: Proceedings of the 10th International Conference on World Wide Web, 2001, pp. 285–295.

[2] Manh Cuong Pham, Yiwei Cao, Ralf Klamma, Matthias Jarke, A clustering approach for collaborative filtering recommendation using social network analysis, J. UCS 17 (2011) 583–604.

[3] Yehuda Koren, Robert Bell, Chris Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (8) (2009) 30–37.

[4] Wang Yun, Ni Jing, Ma Gang, Recommendation algorithm based on funksvd matrix decomposition and similarity matrix, Jisuanji Yingyong Yuruanjian 36 (12) (2019) 245–250.

[5] Rodolphe Jenatton, Nicolas L. Roux, Antoine Bordes, Guillaume R Obozinski, A latent factor model for highly multi-relational data, in: Advances in Neural Information Processing Systems 25, Vol. 25, 2012, pp. 3167–3175.

[6] Daniel D. Lee, H. Sebastian Seung, Learning the parts of objects by non-negative matrix factorization, Nature 401 (6755) (1999) 788–791.

[7] Thomas N. Kipf, Max Welling, Semi-supervised classification with graph convolutional networks, 2016, arXiv preprint arXiv:1609.02907.

[8] William L. Hamilton, Rex Ying, Jure Leskovec, Inductive representation learning on large graphs, 2017, arXiv preprint arXiv:1706.02216.

[9] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, Graph attention networks, 2017, arXiv preprint arXiv:1710.10903.

[10] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, Tat-Seng Chua, Neural graph collaborative filtering, in: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 165–174.

[11] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, 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, 2020, pp. 639–648.

[12] Lei Chen, Le Wu, Richang Hong, Kun Zhang, Meng Wang, Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 27–34.

[13] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, Tat-Seng Chua, Neural collaborative filtering, in: Proceedings of the 26th International Conference on World Wide Web, 2017, pp. 173–182.

[14] Xiangnan He, Xiaoyu Du, Xiang Wang, Feng Tian, Jinhui Tang, Tat-Seng Chua, Outer product-based neural collaborative filtering, 2018, arXiv preprint arXiv:1808.03912.

[15] Ganqu Cui, Jie Zhou, Cheng Yang, Zhiyuan Liu, Adaptive graph encoder for attributed graph embedding, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2020, pp. 976–985.

[16] Yu Rong, Wenbing Huang, Tingyang Xu, Junzhou Huang, DropEdge: TOwards deep graph convolutional networks on node classification, in: ICLR 2020 : Eighth International Conference on Learning Representations, 2020.

[17] Fenyu Hu, Yanqiao Zhu, Shu Wu, Liang Wang, Tieniu Tan, Hierarchical graph convolutional networks for semi-supervised node classification, in: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, 2019, pp. 4532–4539.

[18] Deyu Bo, Xiao Wang, Chuan Shi, Huawei Shen, Beyond low-frequency information in graph convolutional networks, 2021.

[19] Jiafeng Xia, Dongsheng Li, Hansu Gu, Tun Lu, Peng Zhang, Ning Gu, Incremental graph convolutional network for collaborative filtering, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, Association for Computing Machinery, New York, NY, USA, 2021, pp. 2170–2179, http://dx.doi.org/10.1145/3459637.3482354.

[20] Tianwen Chen, Raymond Chi-Wing Wong, Handling information loss of graph neural networks for session-based recommendation, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 1172–1180.

[21] Shaowen Peng, Tsunenori Mine, A robust hierarchical graph convolutional network model for collaborative filtering, 2020, arXiv preprint arXiv:2004.14734.

[22] Diederik P. Kingma, Max Welling, Auto-encoding variational bayes, 2013, arXiv preprint arXiv:1312.6114.

[23] Rianne van den Berg, Thomas N. Kipf, Max Welling, Graph convolutional matrix completion, 2017, arXiv preprint arXiv:1706.02263.

[24] Jiani Zhang, Xingjian Shi, Shenglin Zhao, Irwin King, STAR-GCN: Stacked and reconstructed graph convolutional networks for recommender systems, 2019, arXiv preprint arXiv:1905.13129.

[25] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, Kilian Weinberger, Simplifying graph convolutional networks, in: International Conference on Machine Learning, PMLR, 2019, pp. 6861–6871.

[26] Wenhui Yu, Zheng Qin, Graph convolutional network for recommendation with low-pass collaborative filters, in: ICML 2020: 37th International Conference on Machine Learning, Vol. 1, 2020, pp. 10936–10945.

[27] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, Alexander Lerchner, Beta-VAE: Learning basic visual concepts with a constrained variational framework, in: ICLR 2017 : International Conference on Learning Representations 2017, 2017.

[28] Tian Qi Chen, Xuechen Li, Roger B. Grosse, David Duvenaud, Isolating sources of disentanglement in variational autoencoders, in: ICLR (Workshop), 2018.

[29] Arash Vahdat, Jan Kautz, NVAE: A Deep hierarchical variational autoencoder, in: Advances in Neural Information Processing Systems, 33, 2020, pp. 19667–19679.

[30] X. Glorot, A. Bordes, Y. Bengio, Deep sparse rectifier neural networks, in: Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS), 2011, pp. 315–323.

[31] F.W. Scholz, Maximum likelihood estimation, Wiley StatsRef: Statist. Ref. Online (2006).
[32] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, Tony Jebara, Variational autoencoders for collaborative filtering, in: Proceedings of the 2018 World Wide Web Conference, 2018, pp. 689–698.
[33] Guohao Li, Matthias Muller, Ali Thabet, Bernard Ghanem, Deepgcns: Can GCNs go as deep as CNNs? in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 9266–9275.
[34] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, Lars BPR Schmidt-Thieme, Bayesian personalized ranking from implicit feedback, in: Proc. of Uncertainty in Artificial Intelligence, 2014, pp. 452–461.
[35] Michaël Defferrard, Xavier Bresson, Pierre Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, 2016, arXiv preprint arXiv:1606.09375.
[36] Zhi-Hong Deng, Ling Huang, Chang-Dong Wang, Jian-Huang Lai, S Yu Philip, Deepcf: A unified framework of representation learning and matching function learning in recommender system, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 61–68.
[37] David Martin Powers, Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation, 2011, pp. 37–63, arXiv preprint arXiv:2010.16061 2, 1.
[38] R. Herbrich, Large margin rank boundaries for ordinal regression, Adv. Large Margin Classif. (2000) 115–132.
[39] Xavier Glorot, Yoshua Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, 2010, pp. 249–256.
[40] Diederik P. Kingma, Jimmy Ba, Adam: A method for stochastic optimization, 2014, arXiv preprint arXiv:1412.6980.