Variational Graph Auto-Encoder Driven Graph Enhancement for Sequential Recommendation

Yuwen Liu[1,2] , **Lianyong Qi**[1,2,3*] , **Xingyuan Mao**[1,2] , **Weiming Liu**[4] , **Shichao Pei**[5] , **Fan Wang**[4] , **Xuyun Zhang**[6] , **Amin Beheshti**[6] , **Xiaokang Zhou**[7,8]
[1]College of Computer Science and Technology, China University of Petroleum (East China)
[2]Shandong Key Laboratory of Intelligent Oil and Gas Industrial Software
[3]State Key Laboratory for Novel Software Technology, Nanjing University
[4]College of Computer Science and Technology, Zhejiang University
[5]Department of Computer Science, University of Massachusetts Boston
[6]School of Computing, Macquarie University
[7]Faculty of Business and Data Science, Kansai University
[8]RIKEN Center for Advanced Intelligence Project
yuwenliu97@gmail.com, lianyongqi@upc.edu.cn, hsingyuanmao@gmail.com, 21831010@zju.edu.cn, shichao.pei@umb.edu, fanwang97@zju.edu.cn, {xuyun.zhang, amin.beheshti}@mq.edu.au, zhou@kansai-u.ac.jp

Abstract

Recommender systems play a critical role in many applications by providing personalized recommendations based on user interactions. However, it remains a major challenge to capture complex sequential patterns and address noise in user interaction data. While advanced neural networks have enhanced sequential recommendation by modeling high-order item dependencies, they typically assume that the noisy interaction data as the user's preferred preferences. This assumption can lead to suboptimal recommendation results. We propose a Variational Graph Auto-Encoder driven Graph Enhancement (VGAE-GE) method for robust augmentation in sequential recommendation. Specifically, our method first constructs an item transition graph to capture higher-order interactions and employs a Variational Graph Auto-Encoder (VGAE) to generate latent variable distributions. By utilizing these latent variable distributions for graph reconstruction, we can improve the item representation. Next, we use a Graph Convolutional Network (GCN) to transform these latent variables into embeddings and infer more robust user representations from the updated item embeddings. Finally, we obtain the reconstructed user check-in data, and then use a Mamba-based recommender to make the recommendation process more efficient and the recommendation results more accurate. Extensive experiments on five public datasets demonstrate that our VGAE-GE model improves recommendation performance and robustness.

*Corresponding author

1 Introduction

Recommender systems are extensively utilized in e-commerce, social media, and content streaming to satisfy the users' requirements. On these platforms, understanding the temporal patterns of user interactions can significantly improve the quality of personalized recommendations [Liu *et al.*, 2023a; Ma *et al.*, 2024; Liao *et al.*, 2023]. For the sequential recommendation task, the essential problem is how to utilize sequential patterns of user interactions to provide users with recommendations that satisfy their dynamic preference needs. In the past, a number of attempts [You *et al.*, 2024; Wu *et al.*, 2024] have been made to develop powerful sequential models based on deep learning. Many works [Hidasi *et al.*, 2016; Yue *et al.*, 2024] utilize Recurrent Neural Networks (RNNs) to capture the sequential patterns among user interaction sequences. Additionally, some works [Qi *et al.*, 2022; Ahmed *et al.*, 2023] combine RNNs with attention mechanisms to focus on important information within the interaction sequences. Although above methods have achieved significant results, however, they primarily focus on the sequential patterns of user interactions and fail to capture the more complex item transition patterns in user sequences. Furthermore, user interaction behavior can be influenced by many factors (e.g., benefit incentives, accidental clicks, over-recommendation of popular items, etc.), it can result in user sequences containing some noise. RNN-based methods [Yue *et al.*, 2024; Bach *et al.*, 2020; Ahmed *et al.*, 2023] typically assume user interaction sequences as the primary indicator of user preferences, which may lead to suboptimal recommendation results. Recently, Graph Neural Networks (GNNs) [Scarselli *et al.*, 2008; Nguyen and Tran, 2023; Okamura *et al.*, 2023] have attracted attention for their ability to capture higher-order interactions. In addition, some simplified GNNs (e.g., NGCF [Wang *et al.*, 2019], LightGCN [He *et al.*, 2020], PriGCN [Liu *et al.*, 2023b]) reduce the model complexity by omitting linear transformers and activation functions and are becoming popular in recommender systems. However, these methods still face the challenge of data noise. Directly utilizing all user interaction sequences and performing multi-loop embedding propagation can worsen the noise effect, leading to skewed results.

To address above challenge, recent methods attempt to use self-supervised learning techniques to enhance the noise robustness of recommendation models [Wu *et al.*, 2021; Liao *et al.*, 2023]. These methods create contrastive views for self-supervised learning but rely heavily on the laborious trial-and-error process for selecting augmentation methods. In this paper, we propose a Variational Graph Auto-Encoder driven Graph Enhancement method (VGAE-GE) for robust augmentation in sequential recommendation. We first construct an item transition graph to capture the higher-order interaction information between items. Then we use the inference model of Variational Graph Auto-Encoder (VGAE) to obtain the latent variable distribution of items. The distribution distance between items with interactions should be closer, while those without interactions should be farther apart. Then we use the generative model and latent variables to reconstruct the item transition graph. Finally, we employ a Graph Convolutional Network (GCN) to transform the latent variables back into node embeddings. By combining the reconstructed item embeddings with user embeddings, we utilize a Mamba-based model for sequential recommendation.

- We propose a novel method called VGAE-GE, which can convert item embeddings into potential vectors and capture higher-order information in the potential space. To enhance the noise immunity, we reconstruct the graph structure by using generative model.
- We propose to use GCN to revert the reconstructed graph structure to embedding, and then input the user interaction data into a Mamba-based recommendation model. By combining the basic Mamba blocks with various techniques such as layer normalization and feed-forward networks, the sequence modeling capability is further improved without sacrificing inference efficiency.
- Comprehensive experiments on five public datasets confirm the effectiveness of our VGAE-GE model.

2 Related Work

2.1 Graph-based Sequential Recommendation

Prominent works, such as SR-GNN [Wu *et al.*, 2019], GCN-GNN [Wang *et al.*, 2020] and ITGCN [Liu *et al.*, 2022], utilize GNNs to model user-item interaction sequences, demonstrating significant improvements in recommendation accuracy. Simplified GNN models, such as NGCF [Wang *et al.*, 2019], LightGCN [He *et al.*, 2020] and UltraGCN [Mao *et al.*, 2021], have also become popular in sequential recommendation by reducing model complexity through the omission of linear transformers and activation functions. In addition, some works have utilized graph-based methods for sequential recommendation based on multiple aspects to overcome multiple challenges [Zhu *et al.*, 2021]. Peintner [Peintner, 2023] attempts to reduce the recommendation bias problem when utilizing graph-based models for sequential recommendation. Focusing on the continuous time modeling problem in sequential recommendation, GDERec [Qin *et al.*, 2024] proposes a new GNN based on ordinary differential equations to simulate implicitly the time evolution. Some works utilize GNNs combined with Graph Masked Auto-Encoder to combat data noise and sparsity, such as MAERec [Ye *et al.*, 2023], AutoCF [Xia *et al.*, 2023] and S-CIEE [Wang *et al.*, 2025b]. Despite these advancements, these methods still face challenges related to data noise. Directly using all user interaction sequences and conducting multi-layer information propagation can amplify noise effects, leading to biased recommendation results.

2.2 Variational Graph Auto-Encoders

VGAE has proven effective for learning latent representations of graph-structured data, providing a probabilistic approach to encoding graph information. MVGAE [Yi and Chen, 2021] is a model that utilizes VGAE to generate representations of nodes (i.e., mean vectors for semantic information and variance vectors for the noise level of the corresponding modality), which are then applied to recommender systems. However, MVGAE overlooks the temporal relevance of user-item interaction. Similarly, Altaf et al. [Altaf *et al.*, 2019] simply apply VGAE to a dataset recommendation system. SCVG [Ding *et al.*, 2021] is a semi-deterministic and contrastive VGAE for item recommendation. STR-VGAE [Zhu *et al.*, 2023] uses VGAE to learn travel packages that contain users' implicit preferences from their behavioral sequences to make travel packages recommendation. In addition, there are many models such as DR-VAE [Wang *et al.*, 2025a] and MvDGAE [Zheng *et al.*, 2021] that try to utilize VGAE to exploit the latent distribution for enhancing robustness and accuracy, but currently these methods do not consider both the latent distribution and the embedding representation, and ignore the application in sequential recommender systems.

3 Methodology

Let $\mathcal{U} = \{u_1, u_2, ..., u_{|\mathcal{U}|}\}$ and $\mathcal{V} = \{v_1, v_2, ..., v_{|\mathcal{V}|}\}$ denote the set of users and items, respectively. Each user $u \in \mathcal{U}$ is associated with a sequence of interactions with items over time. We denote historical check-ins for user u as $S^u = \{s_1^u, s_2^u, ..., s_m^u\}$, where m is the length of u's item sequence S^u, $s_t^u \in \mathcal{V}$ represents the item accessed by the user at t-th, and t denotes the corresponding timestamp of interaction with $1 \leq t \leq m$. Our goal is to predict the item s_{m+1}^u that the target user u is most likely to interact with.

3.1 Item Transition Graph Construction

To obtain item dependencies, we use user behavior sequences to generate an item transition graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $N = |\mathcal{V}|$ nodes to represent the transition relationships between different items. $e_{ij} \in \mathcal{E}$ indicates that there is an edge between the i^{th} and j^{th} items. $A = (a_{ij}) \in \mathbb{R}^{N \times N}$ is the adjacency matrix of $\mathcal{G}$ and it indicates the implicit relationships between items. $D \in \mathbb{R}^{N \times N}$ is the degree matrix of A. For the edge set $\mathcal{E}$, we construct it by the following form:

$$\mathcal{E} = \left\{ \left(s_i^u, s_j^u\right) : u \in \mathcal{U}, |i-j| \leq c, 1 \leq i<j \leq m \right\}. \quad (1)$$

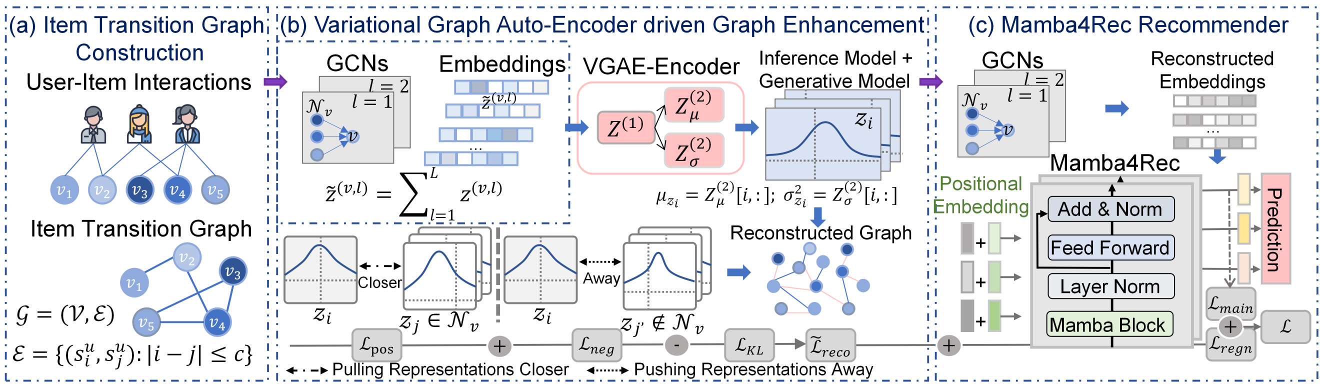
(a) Item Transition Graph Construction
User-Item Interactions
Item Transition Graph
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$
$\mathcal{E} = \{(s_i^u, s_j^u) : |i-j| \le c\}$
(b) Variational Graph Auto-Encoder driven Graph Enhancement
GCNs
Embeddings
VGAE-Encoder
Inference Model + Generative Model
$\tilde{z}^{(v,l)} = \sum_{l=1}^{L} z^{(v,l)}$
$\mu_{z_i} = Z_\mu^{(2)}[i,:]$; $\sigma_{z_i}^2 = Z_\sigma^{(2)}[i,:]$
Reconstructed Graph
Closer
Away
$\mathcal{L}_{pos}$
$\mathcal{L}_{neg}$
$\mathcal{L}_{KL}$
$\mathcal{L}_{reco}$
Pulling Representations Closer
Pushing Representations Away
(c) Mamba4Rec Recommender
Reconstructed Embeddings
Mamba4Rec
Positional Embedding
Add & Norm
Feed Forward
Layer Norm
Mamba Block
Prediction
$\mathcal{L}_{main}$
$\mathcal{L}_{regn}$
$\mathcal{L}$

Figure 1: Overall architecture of our proposed VGAE-GE. (a) Item transition graph construction. (b) Variational Graph Auto-Encoder driven graph enhancement. (c) Mamba4Rec as the sequence recommender.

Here, we create an edge between each item and its c-hop neighbor across all user sequences, computing duplicated edges only once. Thus, $a_{ij} = 1$ if $(v_i, v_j) \in \mathcal{E}$ and $a_{ij} = 0$ otherwise.

3.2 VGAE-driven Graph Enhancement

To encode interaction patterns among items, we map them into a d-dimensional latent space. Specifically, we create an embedding vector $\mathbf{e_j} \in \mathbb{R}^d$ for each item v_j. Additionally, we use $\mathbf{E^{(v)}} \in \mathbb{R}^{N\times d}$ to represent the embedding of items and map the input graph to low-dimensional representations:

$$Z^{(v)} = \tilde{A} \cdot \mathbf{E^{(v)}}. \tag{2}$$

Here, $\tilde{A}$ is calculated by $\tilde{A} = \mathrm{D}^{-1/2} A \mathrm{D}^{-1/2}$. Next, multiple embedding propagation layers are used to aggregate the node neighborhood information of the item:

$$z^{(v,l+1)} = z^{(v,l)} + \sum_{v' \in \mathcal{N}_v} z^{(v',l)}, \tag{3}$$

where $z^{(v,l+1)}$ and $z^{(v,l)}$ denote the the embedding representations of item v at the $(l+1)$-th and l-th layers, respectively. $\mathcal{N}_v$ denotes the neighboring item of item v, $\sum_{v' \in \mathcal{N}_v} z^{(v',l)}$ denotes the neighbor aggregation information for item v at l-th layer. We aggregate the embedding representations of each layer of the item and get the representation.

$$\tilde{z}^{(v,l)} = \sum_{l=1}^{L} z^{(v,l)}, \tag{4}$$

where $\tilde{z}^{(v,l)}$ is the final representation of an item v. Subsequently, we utilize $\tilde{Z}$ to represent the embedding representation of all items.

After obtaining each node representation and the graph structure of the item transformation graph $\mathcal{G}$, we designed a VGAE-based encoder to convert the node representations into a distributed form. The encoder consists of two encoding heads, represented as follows: $Z^{(1)} = f_{\text{ReLU}}\left(\tilde{Z}, A \mid W^{(0)}\right)$; $Z_\mu^{(2)} = f_{\text{Linear}}\left(Z^{(1)}, A \mid W_\mu^{(1)}\right) \in \mathbb{R}^{N\times d}$, $Z_\sigma^{(2)} = f_{\text{Linear}}\left(Z^{(1)}, A \mid W_\sigma^{(1)}\right) \in \mathbb{R}^{N\times d}$. Here, $W_\mu^{(1)}$ is the trainable parameter matrix used to generate each node's mean, $W_\sigma^{(1)}$ is the trainable parameter matrix used to generate each node's variance.

Inference model. The inference model is defined by the distribution $q(Z \mid \tilde{Z}, A)$ parameterized by two encoding layers:

$$q(Z \mid \tilde{Z}, A) = \prod_{i=1}^{N} q\left(z_i \mid \tilde{Z}, A\right),$$

$$\text{with } \prod_{i=1}^{N} q\left(z_i \mid \tilde{Z}, A\right) = \prod_{i=1}^{N} \mathcal{N}\left(z_i \mid \mu_{z_i}, \operatorname{diag}\left(\sigma_{z_i}^2\right)\right), \tag{5}$$

where $q(Z \mid \tilde{Z}, A)$ denotes joint distribution of latent variables for all nodes, $\prod_{i=1}^{N} q\left(z_i \mid \tilde{Z}, A\right)$ denotes the product of the distribution of latent variables for each node z_i, $q\left(z_i \mid \tilde{Z}, A\right)$ denotes the distribution of latent variables for node z_i, $\mu_{z_i} = Z_\mu^{(2)}[i,:]$ is the mean vector of a multivariate Gaussian distribution associated within z_i, and $\sigma_{z_i}^2 = Z_\sigma^{(2)}[i,:]$ is the corresponding variance vector.

Generative model. In the generative model, we use the latent variables Z to generate the distribution of A.

$$p(A \mid Z) = \prod_{i=1}^{N}\prod_{j=1}^{N} p\left(a_{ij} \mid z_i, z_j\right),$$

$$\text{with } p\left(a_{ij} = 1 \mid z_i, z_j\right) = \sigma\left(z_i^\top z_j\right), \tag{6}$$

where $p(A \mid Z)$ denotes the probability of generating a connection between each a_{ij} given the latent variable Z, a_{ij} are the elements of A.

After completing the design of the inference model and the generative model, consistent with [Kipf and Welling, 2016], we formulate a variational lower bound on the log-likelihood of the input graph as: $\mathcal{L}_{\text{reco}} = \mathcal{L}_{\text{pos}} + \mathcal{L}_{\text{neg}}$. Here, $\mathcal{L}_{\text{reco}}$ denotes the reconstruction loss, which consists of the cross-entropy loss for positive (i.e., $\mathcal{L}_{\text{pos}}$) and negative (i.e., $\mathcal{L}_{\text{neg}}$) samples, respectively. Here, the positive sample reconstruction loss calculates the binary cross-entropy loss between the predicted values of the positive sample edges and the true labels:

$$\begin{aligned}\mathcal{L}_{\text{pos}} = -&\sum_{(i,j)\in\text{pos}} [y_{ij}\log\sigma(z_i\cdot z_j)\\ &+(1-y_{ij})\log(1-\sigma(z_i\cdot z_j))],\end{aligned}\tag{7}$$

where (i,j) denotes the edge between item v_i and v_j, y_{ij} is the true label of edge (i,j), $\sigma(z_i\cdot z_j)$ is the probability of the existence of the predicted label, z_i and z_j are the potential representations of the items v_i and v_j, respectively. The potential representation z_i can be computed using mean and variance: $z_i=\mu_i+\sigma\odot\epsilon_i$, where $\epsilon_i\sim\mathcal{N}(0,1)$ is the noise matrix generated by the standard normal distribution, $\odot$ denotes the Hadamard product. Considering that $y_{ij}=1$ for positive samples and $y_{ij}=0$ for negative samples remain constant, the reconstruction loss for positive and negative samples can be simplified as:

$$\begin{aligned}\mathcal{L}_{\text{pos}} &= -\sum_{(i,j)\in\text{Pos}}\log(p(a_{ij}=1\mid z_i,z_j))\\ &=-\sum_{(i,j)\in\text{Pos}}\log(\sigma(z_i\cdot z_j)).\end{aligned}\tag{8}$$

$$\begin{aligned}\mathcal{L}_{\text{neg}} &= -\sum_{(i,j)\in\text{Neg}}\log(p(a_{uv}=0\mid z_i,z_j))\\ &=-\sum_{(i,j)\in\text{Neg}}\log(1-\sigma(z_i\cdot z_j)).\end{aligned}\tag{9}$$

In addition, we utilize the Kullback-Leibler Divergence (KLD) to measure the difference between the distribution $q(Z\mid\tilde{Z},A)$ generated in the variational auto-encoder and the prior distribution $p(Z)$, so that the distribution is as similar as possible to the standard Gaussian in the hypothesis. The KLD loss is as follows:

$$\begin{aligned}\mathcal{L}_{KL} &= \mathrm{KL}[q(Z\mid\tilde{Z},A)\|p(Z)]\\ &=-\frac{1}{2}\sum_{i=1}^{N}\left(1+\log\left(\sigma_i^2\right)-\mu_{z_i}^2-\sigma_{z_i}^2\right).\end{aligned}\tag{10}$$

After considering the KLD, the reconstruction loss can be updated as follows:

$$\begin{aligned}\tilde{\mathcal{L}}_{\text{reco}} &= \mathcal{L}_{\text{pos}}+\mathcal{L}_{\text{neg}}+\mathcal{L}_{KL}\\ &=\sum_{i,j=1}^{N}\mathbb{E}_{z_i,z_j\sim q(.|\tilde{Z},A)}\left[\log\left(p\left(a_{ij}\mid z_i,z_j\right)\right)\right]\\ &+KL\left(q\left(z_i\mid\tilde{Z},A\right)\|p\left(z_i\right)\right).\end{aligned}\tag{11}$$

3.3 Mamba4Rec as Sequence Encoder

After obtaining the reconstructed graph structure, we encode the generated graph using GCN to obtain the item embedding. We still ignore the weight transformations and activation functions in the message passing process. The representation of item is generated as follows:

$$e^{(v,l+1)}=e^{(v,l)}+\sum_{v'\in\mathcal{N}_v}e^{(v',l)},\tag{12}$$

$$\tilde{e}^{(v,l)}=\sum_{l=1}^{L}e^{(v,l)},\tag{13}$$

where $\mathcal{N}_v$ denotes the 1-hop neighborhood of v, $e^{(v,l)}$ and $e^{(v,l+1)}$ denote the reconstructed embedding representations of item v at the l-th and $(l+1)$-th layers, respectively. $\tilde{e}^{(v,l)}$ is the final reconstructed representation of a item v. Here, we design a reconstruction rate hyperparameter β. We get the probability of existence of the edge based on the hidden variables, if the probability is greater than or equal to β, then this edge is retained in the reconstructed graph; otherwise, this edge is eliminated from the reconstructed graph.

In the user's interaction sequence, we add positional embedding p_i to enhance the temporal information of the sequence:

$$E_u=\left[\left(\tilde{e}_{s_1^u}+p_1\right),\left(\tilde{e}_{s_2^u}+p_2\right),\cdots\left(\tilde{e}_{s_m^u}+p_m\right)\right].\tag{14}$$

With the user representation and item representation already obtained, we utilize the user interaction data E_u as input data into the Mamba4Rec model and perform sequence recommendation.

$$\mathbf{E}_{u,m}=\mathrm{Conv1d}(E_uW^{(1)}+b^{(1)}),\tag{15}$$

where $W^{(1)}$ and $b^{(1)}$ are the weights and biases of the linear projection, the $\mathrm{Conv1d}(\cdot)$ is a 1D convolution operation. Next, the output $\tilde{\mathbf{E}}_{u,m}$ is obtained after State Space Model (SSM) and residual connection:

$$\hat{\mathbf{E}}_{u,m}=\mathrm{SSM}(\mathbf{E}_{u,m}),\tag{16}$$

$$\tilde{\mathbf{E}}_{u,m}=(\mathrm{SiLU}(\hat{\mathbf{E}}_{u,m})+r_m)W^{(2)}+b^{(2)},\tag{17}$$

where $\hat{\mathbf{E}}_{u,m}$ is the is the intermediate value processed by the SSM, r_m denotes the residual connection, $\mathrm{SiLU}(\cdot)$ is the activation function. The final interaction probabilities are obtained as follows:

$$\hat{y}=\tilde{\mathbf{E}}_{u,m}\cdot\tilde{\mathbf{e}}_{s_u^{m+1}}.\tag{18}$$

Through the above stages, the Mamba4Rec model is able to efficiently process the user's sequential interaction data and generate high quality recommendation results.

3.4 Model Training

We employ the cross-entropy loss function to optimize the recommendation task. All subsequences of S^u are used as training data, i.e., $\{(s_1^u),(s_1^u,s_2^u),...,(s_1^u,...,s_{m-1}^u)\}$. The main loss function $\mathcal{L}_{\text{main}}$ for the recommendation task is given as follows:

$$\begin{aligned}\mathcal{L}_{\text{main}} &= -\sum_{u\in\mathcal{U}}\sum_{1\le t\le m}\log\sigma\left(\tilde{\mathbf{E}}_{u,t}\cdot\tilde{\mathbf{e}}_{s_u^{t+1}}\right)\\ &+\log\left(1-\sigma\left(\tilde{\mathbf{E}}_{u,t}\cdot\tilde{\mathbf{e}}_{v_t^-}\right)\right),\end{aligned}\tag{19}$$

where $\tilde{\mathbf{E}}_{u,t}$ denotes the embedding of the sequence $(s_1^u,...,s_t^u)$, and $v_t^-\notin S^u$ is the t-th item randomly sampled from the negative samples. In addition, we use regularization

Dataset	#Users	#POIs	#Interactions	Ave.len.	Density
Books	93,043	54,756	506,637	5.45	$9.94e^{-5}$
Retailrocket	91,655	43,886	452,546	4.94	$7.50e^{-5}$
Toys	116,429	54,784	478,460	4.11	$1.12e^{-4}$
NYC	1,083	9,989	179,468	165.71	$1.66e^{-2}$
TKY	2,293	15,177	494,807	215.79	$1.42e^{-2}$

Table 1: Statistics of datasets.

loss for preventing model overfitting, obtained by computing the L2 norm of the model parameters:

$$\mathcal{L}_{\text{reg}} = \lambda \left(\|\theta_{\text{encoder}}\|_2^2 + \|\theta_{\text{decoder}}\|_2^2 + \|\theta_{\text{recommender}}\|_2^2 \right), \tag{20}$$

where θ_{encoder}, θ_{decoder} and $\theta_{\text{recommender}}$ are the parameters of the encoder, decoder and the recommender model, respectively, and λ is the regularization factor. Finally, the total loss $\mathcal{L}$ is the weighted sum of the reconstruction loss, the main loss and the regularization loss:

$$\mathcal{L} = \lambda_1 \tilde{\mathcal{L}}_{\text{reco}} + \lambda_2 \mathcal{L}_{\text{main}} + \mathcal{L}_{\text{reg}}, \tag{21}$$

where λ_1 and λ_2 are hyperparameters that control the strengths of reconstruction loss and main loss, respectively.

4 Experiments

Our experiments focused on validating the performance of VGAT-GE and answering the following key questions: **RQ1:** Can our proposed VGAE-GE outperform the baselines for sequential recommendation? **RQ2:** How does the VGAE affect model performance? **RQ3:** How to demonstrate the effectiveness of Mamba4Rec Recommender? **RQ4:** How is the performance of VGAE-GE affected by different parameter settings?

4.1 Experimental Setup

Datasets. We consider five challenging recommendation datasets: Amazon Books (i.e., Books) and Amazon Toys (i.e., Toys) collected from Amazon platform (*https://www.amazon.com/*), Retailrocket (i.e., Retail) collected from an e-commerce website (*https://www.kaggle.com/retailrocket/ecommerce-dataset/*), NYC and TKY [Yang *et al.*, 2013] collected from Foursquare, of which the statistics are shown in Table 1. We consider items with fewer than three interactions in the first three datasets as outliers and remove them, and similarly, check-ins with fewer than five interactions in the last two datasets are considered as outliers and removed.

Evaluation Metrics. We use leave-one-out strategy for evaluation and use two widely adopted ranking-based metrics to evaluate the performance of all methods, namely Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) with top 5/10/20 recommended candidates.

Baselines. We compare VGAE-GE with 9 competitive methods, including RNN-based methods like **GRU4Rec** [Hidasi *et al.*, 2016] (Abbreviated as G4REC in Table 2), **NARM** [Li *et al.*, 2017], Transformer-based methods like **SASRec** [Kang and McAuley, 2018], **BERT4Rec** [Sun *et al.*, 2019] (Abbreviated as B4Rec in Table 2), **CORE** [Hou *et al.*, 2022],

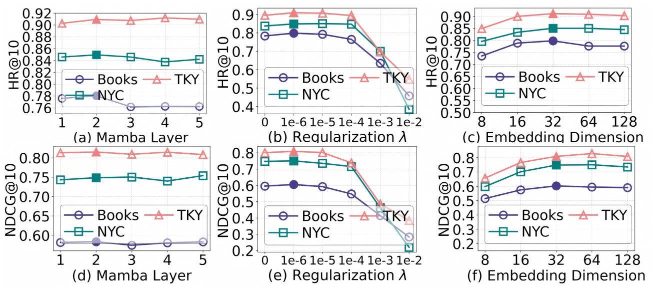

Figure 2: Performance w.r.t. different hyper-parameters.

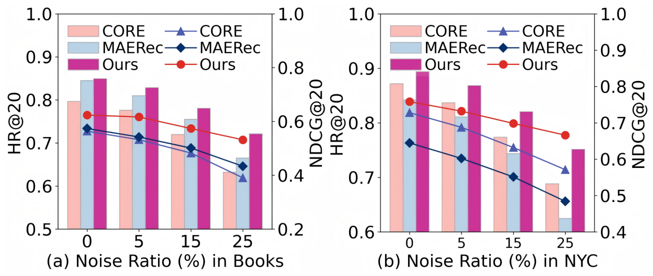

Figure 3: Performance w.r.t noise ratio on Books and NYC.

GNN-based methods such as **SRGNN** [Wu *et al.*, 2019], attention-based methods like **SINE** [Tan *et al.*, 2021], and self-supervised learning models like **CL4SRec** [Xie *et al.*, 2022] (Abbreviated as C4Rec in Table 2), **MAERec** [Ye *et al.*, 2023].

Implementation Details. Our method is implemented in PyTorch and experiments are run on an NVIDIA 4090 GPU. The Adam optimizer is utilized for parameter inference with a learning rate of 1e-2. For the GNN component, we set the number of layers to 2. The embedding dimension is fixed at 32, with a dropout rate of 0.3 to prevent overfitting. We apply a regularization coefficient of 1e-6 to improve model generalization. The graph is constructed using a distance parameter of 3. For the parameters of the Mamba block, the SSM state expansion factor is 32, the kernel size for 1D convolution is 4, and the block expansion factor for linear projections is 2. For the Books, Retail and Toys datasets, we train the model for 150 epochs; the batch size is set to 2048; the reconstruction rate is 0.3; the maximum user sequence length is restricted to 50. For the NYC and TKY, we train the model for 300 epochs; the batch size is set to 256; the reconstruction rate is 0.6; the maximum user sequence length is restricted to 200.

4.2 Performance Analysis (RQ1)

As can be seen in Table 2, our model consistently achieves the best recommendation results. While other models with better performance (e.g., NARM and MAERec) are also able to achieve better results, their excellent performance cannot be maintained on all datasets, and the robustness of the recommendation is not good as the size of the dataset or the interaction scenario changes. Due to the different sizes and features of the datasets, the MAERec model can present better results on datasets such as shopping platforms, where the average user interaction sequences are shorter. However, its perfor-

Datasets	Metric	G4Rec	NARM	SASRec	B4Rec	SRGNN	SINE	CORE	C4Rec	MAERec	Ours	*Improv.*
Books	HR@5	0.5732	0.5958	0.5547	0.4759	0.5386	0.5807	0.6211	0.5689	0.6460	**0.7023**	8.72%
	NDCG@5	0.4609	0.4845	0.4541	0.3691	0.4346	0.4587	0.5133	0.4550	0.5156	**0.5803**	12.54%
	HR@10	0.6754	0.6942	0.6491	0.5824	0.6361	0.6863	0.7099	0.6713	0.7527	**0.7859**	4.41%
	NDCG@10	0.4940	0.5164	0.4846	0.4036	0.4662	0.4929	0.5421	0.4882	0.5502	**0.6075**	10.41%
	HR@20	0.7787	0.7901	0.7452	0.6933	0.7400	0.7886	0.7963	0.7743	0.8448	**0.8494**	0.54%
	NDCG@20	0.5201	0.5406	0.5089	0.4316	0.4924	0.5188	0.5639	0.5141	0.5736	**0.6237**	8.73%
Retail	HR@5	0.7660	0.7736	0.7578	0.6430	0.6916	0.7911	0.8297	0.7236	0.8478	**0.8541**	0.74%
	NDCG@5	0.6689	0.6853	0.6715	0.5398	0.5828	0.7026	0.7552	0.6241	0.7602	**0.7870**	3.53%
	HR@10	0.8313	0.8309	0.8167	0.7241	0.7730	0.8471	0.8682	0.7896	0.8843	**0.8877**	0.38%
	NDCG@10	0.6901	0.7039	0.6906	0.5660	0.6091	0.7208	0.7677	0.6444	0.7736	**0.7979**	3.14%
	HR@20	0.8905	0.8849	0.8728	0.8033	0.8471	0.8969	0.9052	0.8550	0.9154	**0.9190**	0.39%
	NDCG@20	0.7051	0.7176	0.7048	0.5860	0.6279	0.7334	0.7770	0.6597	0.7811	**0.8059**	3.18%
Toys	HR@5	0.4065	0.4220	0.3642	0.3196	0.3629	0.4275	0.4380	0.4054	0.4832	**0.5211**	7.84%
	NDCG@5	0.3090	0.3264	0.2882	0.2352	0.2713	0.3250	0.3492	0.3123	0.3696	**0.4030**	9.02%
	HR@10	0.5194	0.5306	0.4575	0.4284	0.4744	0.5369	0.5356	0.5110	0.5932	**0.6320**	6.54%
	NDCG@10	0.3454	0.3614	0.3183	0.2702	0.3072	0.3604	0.3807	0.3464	0.4052	**0.4389**	8.30%
	HR@20	0.6498	0.6569	0.5770	0.5650	0.6055	0.6590	0.6527	0.6338	0.7097	**0.7318**	3.11%
	NDCG@20	0.3783	0.3933	0.3484	0.3046	0.3402	0.3912	0.4102	0.3773	0.4346	**0.4642**	6.82%
NYC	HR@5	0.7147	0.7599	0.7276	0.7248	0.7479	0.6842	0.7461	0.5891	0.6851	**0.7839**	3.16%
	NDCG@5	0.6583	0.7058	0.6701	0.6660	0.6891	0.6224	0.6911	0.5115	0.5985	**0.7265**	2.93%
	HR@10	0.7913	0.8098	0.7895	0.7775	0.8006	0.7442	0.8172	0.6704	0.7830	**0.8458**	3.50%
	NDCG@10	0.6829	0.7219	0.6902	0.6829	0.7058	0.6416	0.7142	0.5378	0.6300	**0.7462**	3.36%
	HR@20	0.8707	0.8670	0.8578	0.8476	0.8587	0.8153	0.8717	0.7673	0.8421	**0.8938**	2.53%
	NDCG@20	0.7029	0.7363	0.7076	0.7005	0.7204	0.6595	0.7280	0.5623	0.6447	**0.7582**	2.97%
TKY	HR@5	0.8055	0.8408	0.8046	0.8064	0.8330	0.7745	0.8439	0.7418	0.7366	**0.8644**	2.43%
	NDCG@5	0.7256	0.7774	0.7324	0.7300	0.7659	0.6976	0.7699	0.6465	0.6252	**0.7915**	1.81%
	HR@10	0.8644	0.8892	0.8666	0.8561	0.8783	0.8382	0.8901	0.8125	0.8173	**0.9128**	2.55%
	NDCG@10	0.7447	0.7931	0.7527	0.7463	0.7805	0.7184	0.7850	0.6693	0.6516	**0.8120**	2.38%
	HR@20	0.9198	0.9315	0.9167	0.9154	0.9189	0.8936	0.9289	0.8766	0.8853	**0.9442**	1.37%
	NDCG@20	0.7587	0.8041	0.7653	0.7614	0.7907	0.7322	0.7948	0.6855	0.6688	**0.8203**	2.02%

Table 2: Overall performance. The best performing baseline and best performer in each row are underlined and boldfaced, respectively.

mance significantly declines when applied to check-in scenarios with longer average user interaction sequences. Conversely, NARM performs better in check-in scenarios. This all reflects that the current approach is not stable enough to consistently maintain better performance in multiple scenarios. In contrast, VGAE-GE has better performance and robustness and always maintains the best performance in multiple scenarios. On the other hand, our model mines the user's preference representation more accurately through distributional transformation and combines with the Mamba4Rec recommender to maintain good performance on all datasets, which demonstrates that our proposed VGAE-GE model has good robustness and excellent recommendation performance.

Performance Against Data Noise. We replace 5%, 15%, and 25% of user interactions with random negative items to test robustness. Figure 3 shows that VGAE-GE consistently outperforms state-of-the-art model (i.e., CORE and MAERec) under all noise levels on two datasets, with lower performance degradation, demonstrating strong noise resilience.

Ablation Studies (RQ2 and RQ3)

To better understand the contributions of different components in VGAE-GE, we conduct ablation studies by removing specific parts to form the following architectures: **w.o. (without) Graph Enhancement (GE)**, **w.o. Mamba4Rec (mam)** and **w.o. Position Embedding (pos)**. The results on the five datasets are shown in Table 3. To evaluate the enhancement effect of the graph transformation based on VGAE (**RQ2**), we conduct ablation studies and propose a corresponding variant, w.o. GE. From Table 3, it can be inferred that the performance of the model significantly decreases across all five datasets when the graph transformation module is removed. This evidence strongly supports the effectiveness of VGAE-GE module. To evaluate the effectiveness of our recommender (**RQ3**), we proposed a variant, w.o. mam, in which we replace our Mamba4Rec with a widely recognized strong recommender, SASRec. The experimental results show a performance decline across all five datasets. This also demonstrates that our proposed recommender outperforms the widely used SASRec. Furthermore, to measure the role of position encoding in the Mamba4Rec recommender, we remove the position encoding and propose the corresponding variant, w.o. pos. The model with position coding removed still produces a small degradation in performance relative to the VGAE-GE model. This indicates that while the position encoding contributes to improved performance, the core strength of our model does not solely rely on it. Therefore, we retain the position encoding in the Mamba4rec recommender.

4.3 Parameter Sensitivity Analyses (RQ4)

Mamba Layer. Considering that the number of Mamba layers in the recommender can affect the model's performance, we evaluate candidate values from $\{1, 2, 3, 4, 5\}$ and conduct experiments. The results for HR@10 and NDCG@10 are summarized in subfigures (a) and (d) of Figure 2, respective-

Datasets	Model	HR@5	NDCG@5	HR@10	NDCG@10
Books	w.o. GE	0.6859	0.5591	0.7710	0.4759
	w.o. mam	0.6697	0.5430	0.7634	0.5735
	w.o. pos	0.6892	0.5612	0.7742	0.5889
	w. all	**0.7023**	**0.5803**	**0.7859**	**0.6075**
Toys	w.o. GE	0.4989	0.3862	0.6010	0.4192
	w.o. mam	0.4804	0.3719	0.5838	0.4054
	w.o. pos	0.5105	0.6198	0.3854	0.4249
	w. all	**0.5211**	**0.4030**	**0.6320**	**0.4389**
Retail	w.o. GE	0.8457	0.7582	0.8767	0.7695
	w.o. mam	0.8364	0.7660	0.8658	0.7756
	w.o. pos	0.8522	0.7831	0.8797	0.7925
	w. all	**0.8541**	**0.7870**	**0.8877**	**0.7979**
NYC	w.o. GE	0.7553	0.6608	0.8310	0.6852
	w.o. mam	0.7581	0.6835	0.8264	0.7065
	w.o. pos	0.7747	0.7109	0.8412	0.7322
	w all	**0.7839**	**0.7265**	**0.8458**	**0.7462**
TKY	w.o. GE	0.8334	0.7620	0.8997	0.7804
	w.o. mam	0.8434	0.7644	0.9023	0.7836
	w.o. pos	0.8561	0.7770	0.9075	0.7937
	w. all	**0.8644**	**0.7915**	**0.9128**	**0.8120**

Table 3: Ablation study with key modules.

ly. For most datasets, the model performance improved when the number of layers increased from 1 to 2, but subsequently declined with additional layers. Therefore, to standardize the model, we set the number of layers to 2.

Regularization Parameter. For the regularization parameter, we give the candidate values {0, 1e-6,1e-5,1e-4,1e-3,1e-2}, and the experimental results of three representative datasets are selected and summarized in subfigure (b) and subfigure (e) of Figure 2. There is an improvement in the model performance when the regularization parameter λ=1e-6, and the subsequent performance gradually decreases. Therefore we set the value of this parameter to 1e-6.

Embedding Dimension. The embedding dimension is a crucial parameter that affects the model's performance. We conducte experiments with candidate values {8, 16, 32, 64, 128} across datasets of different sizes, and the results are summarized in subfigures (c) and (f) of Figure 2. When the dimensionality is too small, it is difficult to accommodate sufficient feature information, and at this time, boosting the dimension size is beneficial to increase the model's information representation capability. However, as the dimension is excessively raised, the computational cost of the model increases and the performance of the model tends to stabilize. For most datasets, the performance began to stabilize once the embedding dimension reached 32. To balance computational cost and performance, we set the embedding dimension to 32 for the datasets used in our experiments.

Reconstruction Rate. The reconstruction rate relates to the amount of information in the reconstructed graph. We specify candidate values {0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8} for the reconstruction rate and then conduct experiments,visualizing the results for two representative datasets: Books and NYC in Figure 4. Since different application scenarios have their unique characteristics, we need to select the best suitable reconstruction rate for different datasets. For the datasets with larger scales (i.e., Books, Retail, and Toys), we chose $\beta = 0.3$, and for the datasets with relatively smaller scales (i.e., NYC and TKY), we chose a reconstruction rate of $\beta = 0.6$.

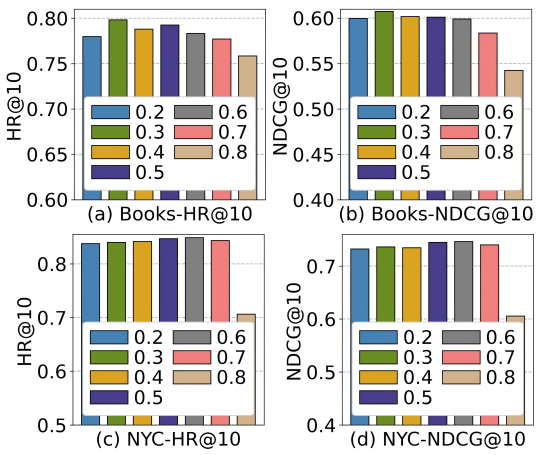

Figure 4: Performance w.r.t. different reconstruction rate.

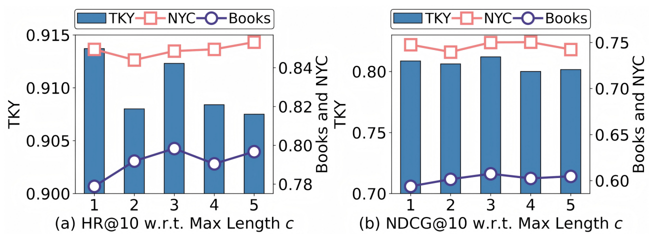

Figure 5: Performance w.r.t. different c-hop neighbor.

c-hop Neighbor. To select the optimal distance for constructing the graph, we consider candidate values for c-hop from {1, 2, 3, 4, 5} and conduct comprehensive experiments on all five datasets. We visualize the results for three datasets and summarize them in Figure 5. When $c = 3$, the majority of the datasets achieve the best performance. When the value of c is other values, such as $c = 1$ or $c = 5$, the performance of the model is not stable enough on different datasets. This indicates that when the graph construction distance is too short, it is difficult to capture higher-order information between items, whereas an excessively long distance incorporates too much noise. In most scenarios, 3-hop neighbors basically provide sufficient neighbor information. And in order to ensure the generality of the model, we make the same settings on all datasets. Therefore, we set $c = 3$ for all datasets.

5 Conclusion

This paper explores a graph augmentation method based on distribution transformation, utilizing VGAE to enhance graph structure information. The embedding is converted into latent variables using VGAE, and then the hidden information is mined and then reduced to the representation of embedding using GCN. Finally, the embeddings are combined with the Mamba4Rec recommender to enhance the sequential recommendation system. In future work, we plan to design more adaptive graph structure augmentation criteria to further improve the model's adaptability. In addition, we will explore capturing complex hierarchical information between users and items in non-Euclidean space.

Acknowledgments

This work is supported by the Natural Science Foundation of Shandong Province (ZR2023MF007), Open Project of State Key Laboratory for Novel Software Technology (K-FKT2024B50, KFKT2024A03), Innovation fund project for graduate students of China University of Petroleum (East China) and the Fundamental Research Funds for the Central Universities (No.25CX04046A).

References

[Ahmed *et al.*, 2023] Imran Ahmed, Misbah Ahmad, Abdellah Chehri, and Gwanggil Jeon. A heterogeneous network embedded medicine recommendation system based on lstm. *Future Generation Computer Systems*, 149:1–11, 2023.

[Altaf *et al.*, 2019] Basmah Altaf, Uchenna Akujuobi, Lu Yu, and Xiangliang Zhang. Dataset recommendation via variational graph autoencoder. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 11–20. IEEE, 2019.

[Bach *et al.*, 2020] Ngo Xuan Bach, Dang Hoang Long, and Tu Minh Phuong. Recurrent convolutional networks for session-based recommendations. *Neurocomputing*, 411:247–258, 2020.

[Ding *et al.*, 2021] Yue Ding, Yuxiang Shi, Bo Chen, Chenghua Lin, Hongtao Lu, Jie Li, Ruiming Tang, and Dong Wang. Semi-deterministic and contrastive variational graph autoencoder for recommendation. In *Proceedings of the 30th ACM international conference on information & knowledge management*, pages 382–391, 2021.

[He *et al.*, 2020] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 639–648, 2020.

[Hidasi *et al.*, 2016] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. 2016.

[Hou *et al.*, 2022] Yupeng Hou, Binbin Hu, Zhiqiang Zhang, and Wayne Xin Zhao. Core: simple and effective session-based recommendation within consistent representation space. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*, pages 1796–1801, 2022.

[Kang and McAuley, 2018] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, pages 197–206. IEEE, 2018.

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

[Li *et al.*, 2017] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1419–1428, 2017.

[Liao *et al.*, 2023] Xinting Liao, Weiming Liu, Xiaolin Zheng, Binhui Yao, and Chaochao Chen. Ppgencdr: A stable and robust framework for privacy-preserving cross-domain recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 4453–4461, 2023.

[Liu *et al.*, 2022] Yuwen Liu, Huiping Wu, Khosro Rezaee, Mohammad R Khosravi, Osamah Ibrahim Khalaf, Arif Ali Khan, Dharavath Ramesh, and Lianyong Qi. Interaction-enhanced and time-aware graph convolutional network for successive point-of-interest recommendation in traveling enterprises. *IEEE Transactions on Industrial Informatics*, 19(1):635–643, 2022.

[Liu *et al.*, 2023a] Weiming Liu, Xiaolin Zheng, Jiajie Su, Longfei Zheng, Chaochao Chen, and Mengling Hu. Contrastive proxy kernel stein path alignment for cross-domain cold-start recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 35(11):11216–11230, 2023.

[Liu *et al.*, 2023b] Yuwen Liu, Xiaokang Zhou, Huaizhen Kou, Yawu Zhao, Xiaolong Xu, Xuyun Zhang, and Lianyong Qi. Privacy-preserving point-of-interest recommendation based on simplified graph convolutional network for geological traveling. *ACM Transactions on Intelligent Systems and Technology*, 2023.

[Ma *et al.*, 2024] Gehua Ma, He Wang, Jingyuan Zhao, Rui Yan, and Huajin Tang. Successive poi recommendation via brain-inspired spatiotemporal aware representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 574–582, 2024.

[Mao *et al.*, 2021] Kelong Mao, Jieming Zhu, Xi Xiao, Biao Lu, Zhaowei Wang, and Xiuqiang He. Ultragcn: ultra simplification of graph convolutional networks for recommendation. In *Proceedings of the 30th ACM international conference on information & knowledge management*, pages 1253–1262, 2021.

[Nguyen and Tran, 2023] Loc Tan Nguyen and Tin T Tran. Combigcn: An effective gcn model for recommender system. In *International Conference on Computational Data and Social Networks*, pages 111–119. Springer, 2023.

[Okamura *et al.*, 2023] Hiroki Okamura, Keisuke Maeda, Ren Togo, Takahiro Ogawa, and Miki Haseyama. Improving dropout in graph convolutional networks for recommendation via contrastive loss. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[Peintner, 2023] Andreas Peintner. Sequential recommendation models: A graph-based perspective. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 1295–1299, 2023.

[Qi *et al.*, 2022] Lianyong Qi, Yuwen Liu, Yulan Zhang, Xiaolong Xu, Muhammad Bilal, and Houbing Song. Privacy-aware point-of-interest category recommendation in internet of things. *IEEE Internet of Things Journal*, 9(21):21398–21408, 2022.

[Qin *et al.*, 2024] Yifang Qin, Wei Ju, Hongjun Wu, Xiao Luo, and Ming Zhang. Learning graph ode for continuous-time sequential recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[Scarselli *et al.*, 2008] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[Sun *et al.*, 2019] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 1441–1450, 2019.

[Tan *et al.*, 2021] Qiaoyu Tan, Jianwei Zhang, Jiangchao Yao, Ninghao Liu, Jingren Zhou, Hongxia Yang, and Xia Hu. Sparse-interest network for sequential recommendation. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 598–606, 2021.

[Wang *et al.*, 2019] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*, pages 165–174, 2019.

[Wang *et al.*, 2020] Ziyang Wang, Wei Wei, Gao Cong, Xiao-Li Li, Xian-Ling Mao, and Minghui Qiu. Global context enhanced graph neural networks for session-based recommendation. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, pages 169–178, 2020.

[Wang *et al.*, 2025a] Fan Wang, Chaochao Chen, Weiming Liu, Minye Lei, Jintao Chen, Yuwen Liu, Xiaolin Zheng, and Jianwei Yin. Dr-vae: Debiased and representation-enhanced variational autoencoder for collaborative recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 12703–12711, 2025.

[Wang *et al.*, 2025b] Fan Wang, Lianyong Qi, Weiming Liu, Bowen Yu, Jintao Chen, and Yanwei Xu. Inter- and intra-similarity preserved counterfactual incentive effect estimation for recommendation systems. *ACM Trans. Inf. Syst.*, 2025.

[Wu *et al.*, 2019] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 346–353, 2019.

[Wu *et al.*, 2021] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. Self-supervised graph learning for recommendation. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pages 726–735, 2021.

[Wu *et al.*, 2024] Yiqing Wu, Ruobing Xie, Yongchun Zhu, Fuzhen Zhuang, Xu Zhang, Leyu Lin, and Qing He. Personalized prompt for sequential recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3376–3389, 2024.

[Xia *et al.*, 2023] Lianghao Xia, Chao Huang, Chunzhen Huang, Kangyi Lin, Tao Yu, and Ben Kao. Automated self-supervised learning for recommendation. In *Proceedings of the ACM Web Conference 2023*, pages 992–1002, 2023.

[Xie *et al.*, 2022] Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. Contrastive learning for sequential recommendation. In *2022 IEEE 38th international conference on data engineering (ICDE)*, pages 1259–1273. IEEE, 2022.

[Yang *et al.*, 2013] Dingqi Yang, Daqing Zhang, Zhiyong Yu, and Zhu Wang. A sentiment-enhanced personalized location recommendation system. In *Proceedings of the 24th ACM conference on hypertext and social media*, pages 119–128, 2013.

[Ye *et al.*, 2023] Yaowen Ye, Lianghao Xia, and Chao Huang. Graph masked autoencoder for sequential recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 321–330, 2023.

[Yi and Chen, 2021] Jing Yi and Zhenzhong Chen. Multi-modal variational graph auto-encoder for recommendation systems. *IEEE Transactions on Multimedia*, 24:1067–1079, 2021.

[You *et al.*, 2024] Xiaoyu You, Jianwei Xu, Mi Zhang, Zechen Gao, and Min Yang. Rrl: Recommendation reverse learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 9296–9304, 2024.

[Yue *et al.*, 2024] Zhenrui Yue, Yueqi Wang, Zhankui He, Huimin Zeng, Julian McAuley, and Dong Wang. Linear recurrent units for sequential recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 930–938, 2024.

[Zheng *et al.*, 2021] Jiawei Zheng, Qianli Ma, Hao Gu, and Zhenjing Zheng. Multi-view denoising graph auto-encoders on heterogeneous information networks for cold-start recommendation. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 2338–2348, 2021.

[Zhu *et al.*, 2021] Tianyu Zhu, Leilei Sun, and Guoqing Chen. Graph-based embedding smoothing for sequential recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 35(1):496–508, 2021.

[Zhu *et al.*, 2023] Guixiang Zhu, Jie Cao, Lei Chen, Youquan Wang, Zhan Bu, Shuxin Yang, Jianqing Wu, and Zhiping Wang. A multi-task graph neural network with variational graph auto-encoders for session-based travel packages recommendation. *ACM Transactions on the Web*, 17(3):1–30, 2023.