Pattern Recognition 155 (2024) 110692

Contents lists available at ScienceDirect

Pattern Recognition

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

DGRM: Diffusion-GAN recommendation model to alleviate the mode collapse problem in sparse environments

Deng Jiangzhou [a,b], Wang Songli [a], Ye Jianmei [a], Ji Lianghao [a], Wang Yong [a,*]

[a] Key Laboratory of Big Data Intelligent Computing, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
[b] Mashang Consumer Finance Co., Ltd., Chongqing 400065, China

ARTICLE INFO

Keywords:
Diffusion model
Generative adversarial network
Mode collapse
Data sparsity
Recommender systems

ABSTRACT

Generative adversarial network (GAN) has been widely adopted in recommender systems (RSs) to improve the recommendation accuracy. However, existing GAN-based models often suffer from the mode collapse problem in sparse environments and fail to adequately capture the complexity of user preferences and behaviors, which affects recommendation performance. To address these issues, we introduce a diffusion model (DM) into the GAN framework, proposing an efficient Diffusion-GAN recommendation model (DGRM) to achieve mutual enhancement between the two generative models. This model first utilizes the forward process of DM to generate conditional vectors that guide the training of the GAN generator. Subsequently, the backward process of DM assists the GAN discriminator using Wasserstein distance during adversarial training. The Wasserstein distance is adopted to solve the asymmetry of Kullback-Leibler (KL) divergence as a loss function in traditional GANs. Experiments on multiple datasets demonstrate that the proposed model effectively alleviates mode collapse and surpasses other state-of-the-art (SOTA) methods in various evaluation metrics.

1. Introduction

The rapid development of the digital economy has led to an explosive growth of data. Efficiently storing, processing, and applying these data have become increasingly important. Particularly, a large amount of homogeneous information floods the network, making it challenging for online users to discover content that truly matches their interests. This leads to a severe problem of information overload [1-4]. Recommender systems (RSs) [5,6], as a prominent information filtering technique, offer personalized recommendation services. By analyzing users' historical preference behaviors deeply, RSs effectively enhance user experience and satisfaction.

In recent years, generative adversarial network (GAN) [7], as an effective deep learning model, has achieved tremendous success in various fields such as image generation [8-11] and speech recognition [12,13]. GANs excel at learning high-dimensional complex data distributions without relying on prior assumptions [14], making them highly effective in RSs to enhance recommendation quality. For instance, Chae et al. [15] introduced a collaborative filtering-based GAN (CFGAN) that uses vector-wise adversarial training to overcome the limitations of traditional GAN-based models [16,17], which often struggle with generating and distinguishing discrete item indices, thereby improving recommendation accuracy. Wang et al. [18] augmented GANs with user review information to enhance label quality, significantly boosting performance in sparse environments. Wen et al. [19] combined user and item embedding vectors through a memory module to help the GAN generator capture personalized user features more effectively. Yang et al. [20] enhanced the interpretability and efficacy of negative sampling strategies by integrating them with GANs on knowledge graphs. Lastly, Chen et al. [21] developed a GAN-based end-to-end recommendation model that addresses the cold-start problem.

However, several studies [22,23] have found that GAN-based models have a high probability of collapsing multiple modes into a few modes, resulting in the loss of certain patterns. This issue is particularly acute in RSs due to the diversity of user rating preferences—ranging from "strongly dislike" to "strongly like". If training GANs omits some preference features, it becomes difficult for the prediction results to accurately reflect users' complex interest preferences and behavior patterns, thereby reducing the reliability of the model. Take the MovieLens dataset as an example, suppose in a rating scale of 1–5, a GAN model generates prediction ratings that are almost exclusively concentrated at the extremes of the scale (1 and 5) with negligible occurrence of

* Corresponding author.
E-mail address: wangyong1@cqupt.edu.cn (W. Yong).

https://doi.org/10.1016/j.patcog.2024.110692
Received 15 March 2024; Received in revised form 30 May 2024; Accepted 13 June 2024
Available online 17 June 2024

intermediate values, this leads to rating mode loss. Such results significantly deviates from the actual distribution of users' original rating preferences and fail to fully capture the spectrum of user preferences, which greatly affects recommendation quality and user satisfaction. The main reasons for mode collapse in GAN-based models include, on the one hand, the use of Kullback-Leibler (KL) divergence as a loss function. Its asymmetry forces the generator to sacrifice certain modes to maintain training accuracy [22-24]. On the other hand, in sparse environments, the discriminator accelerates model convergence. As the discriminator reaches optimum, the gradient of the generator may tend toward zero, leading to gradient disappearance and mode collapse, thus impacting the prediction results [25].

To alleviate the mode collapse problem in GANs, existing studies have primarily focused on constructing new loss functions. For instance, Arjovsky et al. [26] proposed the Wasserstein GAN (WGAN), which analyzes the mode collapse problem by minimizing the Earth-Mover distance. This model effectively balances the discriminator and generator during traditional GAN training. Subsequently, Gulrajani et al. [27] improved upon the WGAN by replacing weight clipping with a gradient penalty, further stabilizing the training process. Despite these advancements, these methods still tend to perform poorly in sparse environments. Inspired by studies [28,29], we explored incorporating a diffusion model (DM) into the GAN framework as a potential solution to this issue.

DM [30,31] mainly consists of two parts: forward and backward processes. It begins by sampling Gaussian noise from a prior distribution and gradually removes the noise through learned noise distributions. Through these processes, DM can gradually capture high-quality samples from the initial noise distribution. The parameters are then optimized using maximum likelihood estimation to generate data that matches the real distribution. Currently, DM is widely used in image processing [31,32], with limited applications in the recommendation domain. To the best of our knowledge, Wang et al. [33] were the first to apply DM in RSs, using it to learn the generation process of user interactions through denoising, thereby achieving good recommendation performance. Following this, Zhao et al. [34] proposed a plug-in DM for embedded denoising that employs a multi-step approach to manage implicit noise in RSs, thereby enhancing recommendation accuracy. However, these methods assume that the denoising distribution is Gaussian, which makes the denoising process need to be carried out step by step or in a small step size; otherwise, if a large denoising step size is used, it is difficult to ensure that the denoising distribution is always close to a Gaussian distribution [28]. Moreover, if the number of denoising steps is too large, the efficiency of the model will be seriously affected [28,31]. Based on this issue, we think of using the generator in GAN for a one-step denoising in the latent vector space, instead of the multi-step denoising approach of DM. This strategy addresses the issue of noise distribution changes during large steps denoising and improves the efficiency of the model while ensuring the diversity of the distribution.

Based on the above analysis and discussion, in this paper strategically integrates a DM into the GAN framework, proposing an efficient Diffusion-GAN Recommendation Model (DGRM) to facilitate mutual enhancement between the two models, thereby enhancing the performance of RSs. This model effectively mitigates the mode collapse problem typically seen in traditional GANs. Moreover, the forward process of DM generates dense vectors at different steps, providing real samples for the discriminator in GAN and as conditional vectors to guide the training of the generator. This process helps preserve different preference features of users by controlling the noise scale. Subsequently, guided by these conditional vectors, the generator performs denoising in the latent space,effectively predicting the original rating matrix. Finally, the prediction matrix facilitates the generation of a corresponding noise matrix through the backward process, which is then used for adversarial training. This approach addresses the problem of data sparsity and achieves the effective reconstruction of users' personalized interests.

The main contributions of this paper are outlined as follows:

- By integrating DM with GAN, we propose an effective generative interactive recommendation model. This model not only alleviates mode collapse commonly encountered in GANs in sparse environments but also resolves the problems of low operational efficiency caused by a single noise distribution in DM. It effectively fosters mutual enhancement between the two generative models.
- In the proposed model, the generator employs the forward process to obtain conditional vectors that enable more accurate predictions. The discriminator measures the Wasserstein distance between fake samples, generated by the backward process, and real samples, produced by the forward process under the same conditional vector. This approach not only simplifies the training task but also effectively prevents the discriminator from overfitting, thereby enhancing the stability of model training.
- Experiments conducted on three public datasets with varying levels of sparsity demonstrate that our model alleviates the mode collapse problem, compared with a traditional GAN model (CFGAN), and outperforms other comparison methods in terms of recommendation accuracy.

The remaining parts of this paper are structured as follows. Section 2 introduces some preliminaries concerning DM and improved GAN (called WGAN). In Section 3, we provide a detailed description of the proposed model DGRM. Section 4 presents a series of experiments to verify the effectiveness of our model. Finally, Section 5 concludes the study and discusses future work.

2. Preliminaries

In this section, we introduce the relevant knowledge regarding diffusion model (DM) and Wasserstein GAN (WGAN).

2.1. Diffusion model

DM is a probabilistic generative model that is designed to generate clean samples from noisy data. It achieves this by learning the backward process of data generation, which is essential a step-by-step denoising procedure. The ultimate goal of DM is to approximate the distribution of real data as closely as possible. Typically, DM consists of two main processes: forward noise-adding and backward denoising processes. The forward process works by gradually adding T steps of Gaussian noise to the raw data sample $\mathbf{x}_0 \sim q(\mathbf{x}_0)$:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}\right) \quad t \in \{1, \cdots, T\} \tag{1}$$

where $q(\mathbf{x}_0)$ represents a generating-data distribution, t denotes the diffusion step, and $\beta_t \in (0, 1)$ controls the noise scales added at the step t. The backward process is to remove the added noise at the step t from $\mathbf{x}_t$ and recover $\mathbf{x}_{t-1}$ with parameters θ:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_t^2\mathbf{I}\right) \tag{2}$$

where $\mu_\theta(\mathbf{x}_t, t)$ and $\sigma_t^2\mathbf{I}$ represent the mean and variance of the predicted Gaussian distribution, respectively. The purpose of process is to capture slight changes in the complex distribution to reconstruct $\mathbf{x}_0$ from $\mathbf{x}_T$. DM is optimized by maximizing the evidence lower bound (ELBO) of the likelihood of data $\mathbf{x}_0$:

$$\begin{aligned}\log p_\theta(\mathbf{x}_0) &= \log \int p_\theta(\mathbf{x}_{0:T}) d\mathbf{x}_{1:T} \\ &= \log \mathrm{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] \\ &\geq -\sum_{t\geq 1}^{T} \mathrm{E}_{q(\mathbf{x}_t|\mathbf{x}_0)}[D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))] + C\end{aligned} \tag{3}$$

where C is a constant, D_{KL} represents the KL distance between the true denoising distribution $q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)$ and the parameterized denoising distribution $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$. The Eq. (3) can be simplified as [31]:

$$\sum_{t\geq 1}^{T} \mathrm{E}_{t,\epsilon}\left[\parallel \epsilon - \epsilon_\theta(\mathbf{x}_t, t) \parallel_2^2 \right] \tag{4}$$

where $\epsilon \sim \mathcal{N}(\mathbf{0},\mathbf{I})$; and $\epsilon_\theta(\mathbf{x}_t,t)$ represents a neural network used to predict the noises added from $\mathbf{x}_0$ to $\mathbf{x}_t$ in the forward process. After iterative training θ, DM can utilize the backward process $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ to denoise $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0},\mathbf{I})$ repeatedly to $\mathbf{x}_0$, viz. $\mathbf{x}_T \to \mathbf{x}_{T-1} \to \cdots \to \mathbf{x}_0$.

2.2. Wasserstein GAN

GAN is a remarkable generative model that can learn complex, high-dimensional data distributions without relying on any prior assumptions. It consists of a generator G and a discriminator D, achieving Nash equilibrium through continual adversarial optimization training. Formally, the G and D are playing the following two-player minimax game with the objective function V:

$$\min_{\theta}\max_{\phi} V(G,D) = \mathbb{E}_{\mathbf{x}\sim p_{data}}[\log D(\mathbf{x})] + \mathbb{E}_{\hat{\mathbf{x}}\sim p_\phi}[\log(1-D(\hat{\mathbf{x}}))] \tag{5}$$

where θ and ϕ denote the model parameters of G and D, respectively; $\mathbf{x}$ is a real sample from the true distribution p_{data}, while $\hat{\mathbf{x}}$ is a fake sample from the model distribution p_ϕ; $D(\cdot)$ denotes the estimated probability that its input sample is real. However, some studies [22,23] have found that GAN training is unstable and suffers from mode collapse due to the asymmetry of KL distance as a loss function in GANs. The theoretical explanation is presented Appendix C.

To address these issues, Wasserstein distance is introduced into GAN, called Wasserstein GAN (WGAN) [26], to improve the quality of generated samples. The Wasserstein distance is defined as follows:

$$W(\mathrm{P}_r,\mathrm{P}_g) = \inf_{\gamma\in\prod(\mathrm{P}_r,\mathrm{P}_g)} \mathrm{E}_{(x,y)\sim\gamma} \parallel x-y \parallel \tag{6}$$

where $\Pi(\mathbf{P}_r,\mathbf{P}_g)$ represents the set of all joint distributions $\gamma(x,y)$ whose marginals are $\mathbf{P}_r$ and $\mathbf{P}_g$, respectively.

However, because the infimum in Eq. (6) is difficult to obtain, Eq. (6) is transformed, as follows:

$$W(\mathrm{P}_r,\mathrm{P}_g) = \frac{1}{K}\sup_{\|f\|_L\leq K}\left(\mathrm{E}_{x\sim\mathrm{P}_r}[f(x)] - \mathrm{E}_{x\sim\mathrm{P}_g}[f(x)]\right) \tag{7}$$

where the supremum is over all the K-Lipschitz functions f. Then, a set of parameters w is constructed to define functions $f_{(w)}$ that may satisfy certain conditions. Thus, Eq. (7) can be approximated as:

$$K\cdot W(\mathrm{P}_r,\mathrm{P}_g) \approx \sup_{w:\|f_{(w)}\|_L\leq K}\left(\mathrm{E}_{x\sim\mathrm{P}_r}[f(x)] - \mathrm{E}_{x\sim\mathrm{P}_g}[f(x)]\right) \tag{8}$$

Subject to the condition $\| f_{(w)} \|_L \leq K$, solving the Wasserstein distance is equivalent to solving the following L:

$$L = \mathrm{E}_{x\sim\mathrm{P}_r}\left[f_{(w)}(x)\right] - \mathrm{E}_{x\sim\mathrm{P}_g}\left[f_{(w)}(x)\right] \tag{9}$$

Since the first term in L is independent of the generator G, the loss functions of the generator G and the discriminator D are respectively given by

$$L^{\mathrm{G}} = -\mathrm{E}_{x\sim\mathrm{P}_g}\left[f_{(w)}(x)\right],\ L^{\mathrm{D}} = -\mathrm{E}_{x\sim\mathrm{P}_r}\left[f_{(w)}(x)\right] + \mathrm{E}_{x\sim\mathrm{P}_g}\left[f_{(w)}(x)\right] \tag{10}$$

3. Proposed model

To alleviate the problem of mode collapse in traditional GAN-based models caused by the data sparsity problem in RSs, we propose an effective denoising recommendation model that integrates DM and GAN, called DGRM. The overall framework of the proposed model is illustrated in Fig. 1. The training process of our model DGRM is introduced as follows. The first is the forward process of DM. Gaussian noise is regularly added to the user-item rating matrix $\mathbf{R}$ to perturb the original rating records and denser the sparse matrix. Then, we utilize the backward process and WGAN at denoising steps to recover the perturbed dense matrix, and user interaction interests are reconstructed to recommend items for active users through iterative training of the model. The advantages of this fusion in our model are that the forward process can alleviate the GAN mode collapse caused by highly sparse input data, and the time-independent WGAN training framework can be employed to further stabilize the model training.

3.1. Forward and backward processes of DGRM

Step 1: The forward process of our model. The forward process is used to regularly add Gaussian noise to perturb the original rating distribution. The specific substeps are as follows. Firstly, the conditional vector $\mathbf{R}^t$ at step t as input to the generator G and the discrimination vector $\mathbf{R}^{t-1}$ at step t-1 as input to the discriminator D are two dense matrices that have been introduced to noise, and $\mathbf{R}^{t-1}$ can be cleverly applied to the training task of D to stabilize the training process of the model. Then, the noise scale and its corresponding step t can freely control the level of personalized information in the user rating vector, which provides our model with conditional vectors of different densities to enhance the robustness of the model training process.

Let the rating vector of a user u be $\mathbf{R}_u = [r_{u1},\cdots,r_{ui},\cdots,r_{um}]$, where m is the number of all items, and r_{ui} denotes the rating value of user u on item i. According to Eq. (1), given an initial rating vector $\mathbf{R}_u = \mathbf{R}_u^0 \sim q(\mathbf{R}^0)$ at step $t = 0$, the noise-addition process can be parameterized by

$$q(\mathbf{R}_u^t|\mathbf{R}_u^{t-1}) = \mathcal{N}\left(\mathbf{R}_u^t; \sqrt{1-\beta_t}\mathbf{R}_u^{t-1}, \beta_t\mathbf{I}\right)\quad t\in\{1,\cdots,T\} \tag{11}$$

where β_t is used to control the Gaussian noise scales at the step t. Based on reparameterization technique and independent Gaussian distributions mentioned in [31], we can derive any $\mathbf{R}_u^t$ directly from $\mathbf{R}_u^0$:

$$q(\mathbf{R}_u^t|\mathbf{R}_u^0) = \mathcal{N}\left(\mathbf{R}_u^t; \sqrt{\overline{\alpha}_t}\mathbf{R}_u^0, (1-\overline{\alpha}_t)\mathbf{I}\right) \tag{12}$$

where $\alpha_t = 1-\beta_t$ and $\overline{\alpha}_t = \prod_{t'=1}^{t}\alpha_{t'}$. Thus, we have:

$$\mathbf{R}_u^t = \sqrt{\overline{\alpha}_t}\mathbf{R}_u^0 + \sqrt{1-\overline{\alpha}_t}\epsilon \tag{13}$$

where $\epsilon \in \mathcal{N}(\mathbf{0},\mathbf{I})$.To adjust the noise scales at different t, a linear noise schedule method [33] is adopted, as follows:

$$1-\overline{\alpha}_t = c\cdot\left[\alpha_{\min} + \frac{t-1}{T-1}(\alpha_{\max}-\alpha_{\min})\right]\quad t\in\{1,\cdots,T\} \tag{14}$$

where $c\in[0,1]$ is used to control the scale of the noise, and $\alpha_{\max}$ and $\alpha_{\min}$ represent the upper and lower bounds of the added noises, respectively. Thus, we can obtain $\mathbf{R}_u^t$ and $\mathbf{R}_u^{t-1}$ from the above forward process, and then they will be used for the training process of WGAN model as Step 2 mentioned in Section 3.2.

Step 3: The backward process of our model. Instead of the noise prediction, we perform the backward denoising process directly by predicting $\mathbf{R}_u^0$. Actually, predicting the noise is equivalent to predicting $\mathbf{R}_u^0$. The explanation and proof are as follows.

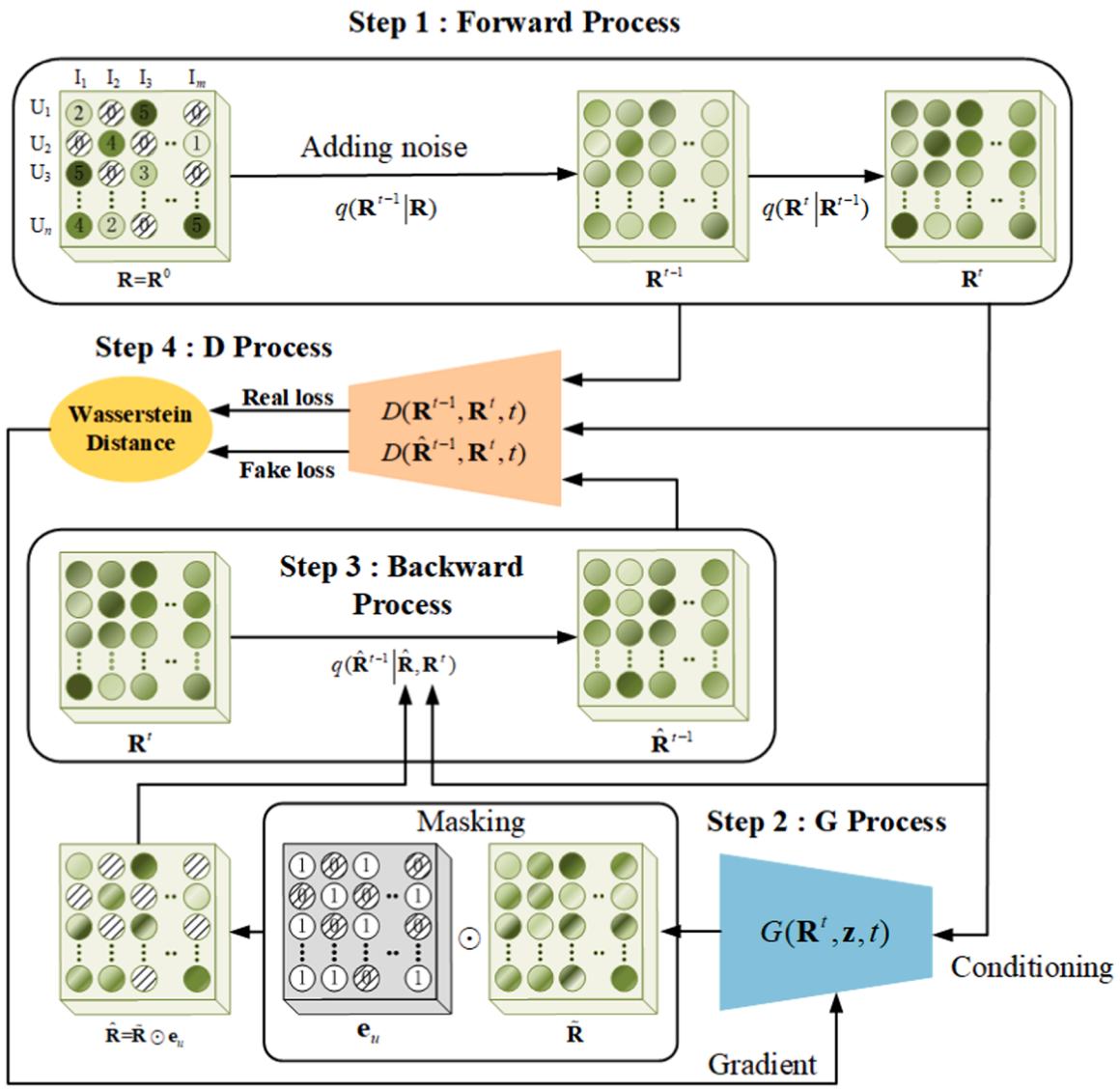

Fig. 1. The overall framework of the proposed DGRM.

Firstly, the parameterized denoising distribution $p_\theta(\mathbf{R}_u^{t-1}|\mathbf{R}_u^t)$ in the traditional backward process needs to be fitted to $q(\mathbf{R}_u^{t-1}|\mathbf{R}_u^t,\mathbf{R}_u^0)$. According to the Bayes theorem, we can obtain $q(\mathbf{R}_u^{t-1}|\mathbf{R}_u^t,\mathbf{R}_u^0)$:

$$q(\mathbf{R}_u^{t-1}|\mathbf{R}_u^t,\mathbf{R}_u^0)=\frac{q(\mathbf{R}_u^t|\mathbf{R}_u^{t-1})q(\mathbf{R}_u^{t-1}|\mathbf{R}_u^0)}{q(\mathbf{R}_u^t|\mathbf{R}_u^0)} \tag{15}$$

where the Eq. (15) follows the forward process. Since all three terms obey the Gaussian distribution, the posterior distribution $q(\mathbf{R}_u^{t-1}|\mathbf{R}_u^t,\mathbf{R}_u^0)$ also obeys the Gaussian distribution. Thus, it can be defined as

$$q(\mathbf{R}_u^{t-1}|\mathbf{R}_u^t,\mathbf{R}_u^0)=\mathcal{N}(\mathbf{R}_u^{t-1};\widetilde{\mu}_t(\mathbf{R}_u^t,\mathbf{R}_u^0),\widetilde{\beta}_t\mathbf{I}) \tag{16}$$

where $\widetilde{\mu}_t(\mathbf{R}_u^t,\mathbf{R}_u^0)$ and $\widetilde{\beta}_t$ represent the mean and variance of the posterior distribution, respectively. Based on Eqs. (11) and (12), we obtain:

$$\widetilde{\mu}_t(\mathbf{R}_u^t,\mathbf{R}_u^0):=\frac{\sqrt{\overline{\alpha}_{t-1}}\beta_t}{1-\overline{\alpha}_t}\mathbf{R}_u^0+\frac{\sqrt{\alpha_t}(1-\overline{\alpha}_{t-1})}{1-\overline{\alpha}_t}\mathbf{R}_u^t,\ \widetilde{\beta}_t:=\frac{1-\overline{\alpha}_{t-1}}{1-\overline{\alpha}_t}\beta_t \tag{17}$$

Then, the noise ϵ added in the forward process is predicted by a neural network $\epsilon_\theta(\mathbf{R}_u^t,t)$ in DM, and $p_\theta(\mathbf{R}_u^{t-1}|\mathbf{R}_u^t)$ is obtained by [31]

$$\mathbf{R}_u^{t-1}=\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{R}_u^t-\frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(\mathbf{R}_u^t,t)\right)+\sigma_t z \tag{18}$$

where $z\sim\mathcal{N}(\mathbf{0},\mathbf{I})$; and $\sigma_t=\sqrt{\widetilde{\beta}_t}$ is the standard deviation of the posterior distribution. Further, we have $\mathbf{R}_u^t=\sqrt{\overline{\alpha}_t}\mathbf{R}_u^0+\sqrt{1-\overline{\alpha}_t}\epsilon$. After using $\epsilon_\theta(\mathbf{R}_u^t,t)$ to predict the noise ϵ, we have $\mathbf{R}_u^0=\frac{1}{\sqrt{\overline{\alpha}_t}}(\mathbf{R}_u^t-\sqrt{1-\overline{\alpha}_t}\epsilon_\theta(\mathbf{R}_u^t,t))$ as a prediction of $\mathbf{R}_u^0$. Then, we replace $\mathbf{R}_u^0$ in the mean of the posterior distribution in Eq. (17):

$$\widetilde{\mu}_t(\mathbf{R}_u^t,\mathbf{R}_u^0)=\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{R}_u^t-\frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(\mathbf{R}_u^t,t)\right) \tag{19}$$

Compared with Eq. (18), Eq. (19) simply corresponds to sampling from the Gaussian posterior distribution. Thus, the denoising posterior distribution $p_\theta(\mathbf{R}_u^{t-1}|\mathbf{R}_u^t)$ can be parameterized as

$$p_\theta(\mathbf{R}_u^{t-1}|\mathbf{R}_u^t):=q(\mathbf{R}_u^{t-1}|\mathbf{R}_u^t,\widetilde{\mathbf{R}}_u^0=f_\theta(\mathbf{R}_u^t,t)) \tag{20}$$

where $\widetilde{\mathbf{R}}_u^0$ can be predicted by $f_\theta(\mathbf{R}_u^t,t)$. Here, we use the generator G to replace $f_\theta(\mathbf{R}_u^t,t)$ for prediction. As a result, the backward denoising

process can be implemented by sampling $\mathbf{R}_u^{t-1}$ using the posterior distribution $q(\mathbf{R}_u^{t-1}|\mathbf{R}_u^t, \widetilde{\mathbf{R}}_u^0)$ given $\mathbf{R}_u^t$ and the predicted $\widetilde{\mathbf{R}}_u^0$.

3.2. Generation and discrimination processes of DGRM

To address the problem that the traditional GAN models are difficult to simulate the complex and sparse distribution of the user-item rating matrix in one shot, the time independent WGAN framework is adopted to mitigate training instability and mode collapse of the model. In this framework, the discriminator D needs to guide the generator G to generate fake samples similar to the real sample distribution based on Wasserstein distance. Here, both the D and G are trained by a neural network called the multi-layer perceptron (MLP). In addition, since the information contained in the user condition vector $\mathbf{R}_u^t$ at different t is different, G needs to generate fake vectors guided by $\mathbf{R}_u^t$ at different t. Accordingly, the D also discriminates the Wasserstein distance between the fake vectors and the real vectors at different t, thus further enhancing the robustness of the model.

Step 2: Fake sample is generated by the generator G. For a specific user u, let the conditional rating vector $\mathbf{R}_u^t$ after noise obtained in Step 1 and its corresponding embedding vector $\mathbf{v}_t$ of the step t be concatenated as the conditional vector, and inputting random noise $\mathbf{z}_u$ into the G, viz. $G(\mathbf{R}_u^t, \mathbf{z}_u, \mathbf{v}_t)$. This enables the G to denoise in the latent vector space and obtain a initial fake rating vector $\widetilde{\mathbf{R}}_u^0$ which is a completely dense matrix.

To simulate the sparsity level of the original rating matrix, a masking layer is constructed, denoted by $\widetilde{\mathbf{R}}_u^0 \odot \mathbf{e}_u$, where $\mathbf{e}_u$ has the same dimension as $\widetilde{\mathbf{R}}_u^0$, and it is used to show whether a set of items have been rated by the user u. If an item i has been rated by the user u, the e_{ui} is equal to 1, otherwise 0; $\odot$ represents the element-wise product. Thus, the generator G accepts only the loss gradient of rated items, ensuring that the G learns more efficiently. Moreover, a fake predicted rating matrix $\widehat{\mathbf{R}}_u^0 = \widetilde{\mathbf{R}}_u^0 \odot \mathbf{e}_u$ with the same sparsity level as the original matrix is also obtained for the discriminator D training.

Step 4: The authenticity of dense samples is determined by the D based on Wasserstein distance. Firstly, the D needs to know the distribution of the real samples $\mathbf{R}_u^{t-1}$ that are obtained by the forward process (Step 1) given $\mathbf{R}_u^0$ and $\mathbf{R}_u^t$. Then, the D also needs to identify the distribution of the fake samples $\widehat{\mathbf{R}}_u^{t-1}$ through the backward process (Step 3). That is, sampling $\widehat{\mathbf{R}}_u^{t-1}$ using the posterior distribution $q(\widehat{\mathbf{R}}_u^{t-1}|\mathbf{R}_u^t, \widehat{\mathbf{R}}_u^0)$ given $\mathbf{R}_u^t$ and $\widehat{\mathbf{R}}_u^0$. Here, it is worth noting that the masking layer plays a crucial role in filtering out unrated item vectors for obtaining the distribution of the fake samples $\widehat{\mathbf{R}}_u^{t-1}$. Because the noise of the position of unrated item vector will affect the calculation of the posterior distribution $q(\mathbf{R}_u^{t-1}|\mathbf{R}_u^t, \widetilde{\mathbf{R}}_u^0)$ in Eq. (20), resulting in the inaccurate distribution of $\widehat{\mathbf{R}}_u^{t-1}$. Thus, we replace $\widetilde{\mathbf{R}}_u^0$ in $q(\mathbf{R}_u^{t-1}|\mathbf{R}_u^t, \widetilde{\mathbf{R}}_u^0)$ with $\widehat{\mathbf{R}}_u^0 = \widetilde{\mathbf{R}}_u^0 \odot \mathbf{e}_u$. To overcome the overfit of D, an optimization model of D is constructed to guide the G to be better learned. Formally, the D can be optimized by

$$\min_{\phi} \sum_u \sum_{t\geq 1} \left[\mathbb{E}_{q(\mathbf{R}_u^0) q(\mathbf{R}_u^{t-1}|\mathbf{R}_u^0) q(\mathbf{R}_u^t|\mathbf{R}_u^{t-1})} D_\phi(\mathbf{R}_u^{t-1}, \mathbf{R}_u^t, t) - \mathbb{E}_{q(\mathbf{R}_u^t) p_\theta(\widehat{\mathbf{R}}_u^{t-1}|\mathbf{R}_u^t, \widehat{\mathbf{R}}_u^0)} D_\phi(\widehat{\mathbf{R}}_u^{t-1}, \mathbf{R}_u^t, t) \right] \tag{21}$$

where $p_\theta(\widehat{\mathbf{R}}_u^{t-1}|\mathbf{R}_u^t, \widehat{\mathbf{R}}_u^0 = \widetilde{\mathbf{R}}_u^0 \odot \mathbf{e}_u)$ represents the distribution of the fake sample produced from $\mathbf{R}_u^t$ and $\widetilde{\mathbf{R}}_u^0$ generated by the G, and θ is a set of parameters of the G. Since the first term in Eq. (21) is independent of the parameters of the G, the G is trained by

Table 1
Detailed description of all used datasets.

Dataset	#User	#Item	#Rating	Sparsity level
ML-100K	943	1,682	100,000	6.3 %
ML-1M	6,040	3,952	1,000,209	4.2 %
AA	87,271	13,209	752,937	0.065 %

$$\min_{\theta} \sum_u \sum_{t\geq 1} \mathbb{E}_{q(\mathbf{R}_u^t) p_\theta(\widehat{\mathbf{R}}_u^{t-1}|\mathbf{R}_u^t, \widehat{\mathbf{R}}_u^0)} D_\phi(\widehat{\mathbf{R}}_u^{t-1}, \mathbf{R}_u^t, t) + \lambda \sum_u \left((\widetilde{\mathbf{R}}_u^0 - \mathbf{R}_u^0) \odot \mathbf{e}_u \right)^2 \tag{22}$$

where λ is used to control the importance of reconstructing rating loss.

3.3. Model prediction

After the model converges, we only use the forward process (Step 1), the generation process (Step 2), and the backward process (Step 3) for predictions. Firstly, since there may be some noise, e.g., natural noise, in the user-item rating matrix, we use the Step 1 to add T times of noise to the sparse rating vectors of users to get $\mathbf{R}^T$. Then, the initial predictions $\widetilde{\mathbf{R}}^0$ are generated by the Step 2 in the latent vector space with a conditional vector $\mathbf{R}^T$. Finally, we set the denoising times in the Step 3 to be $T' < T$ to obtain the final prediction results $\mathbf{R}^{T-T'}$. We do this for two main reasons, as follows: i) This can preserve the users' personalized information to some extent. ii) Preserving some of the noise introduced by in the latent vector space helps to simulate the real uncertain behaviors of users, which has a positive effect on capturing more potential preferences of uses and effectively improves the personalized performance of RSs.

4. Experiments

This section describes multiple datasets and indicators to evaluate the performance of our method DGRM. To compare the proposed DGRM with other SOTA methods, various experiments were conducted. The open source code of DGRM is available at https://github.com/OOHXWS/DGRM/tree/master.

4.1. Dataset preparation

Three public datasets, MovieLens 100k (ML-100K), MovieLens 1M (ML-1M), and Apps for Android (AA), with different sparsity levels are used for experimental design and analysis. A detailed description of all used datasets is provided in Table 1. Moreover, to fully cover all users in each dataset, we randomly selected 80 % of each user's rating records as the training set, and the remaining 20 % as the testing set for model validation.

4.2. Evaluation metrics

Two widely used evaluation metrics, Recall and NDCG (Normalized Discounted Cumulative Gain), are adopted to measure the performance of Top-N recommendations.

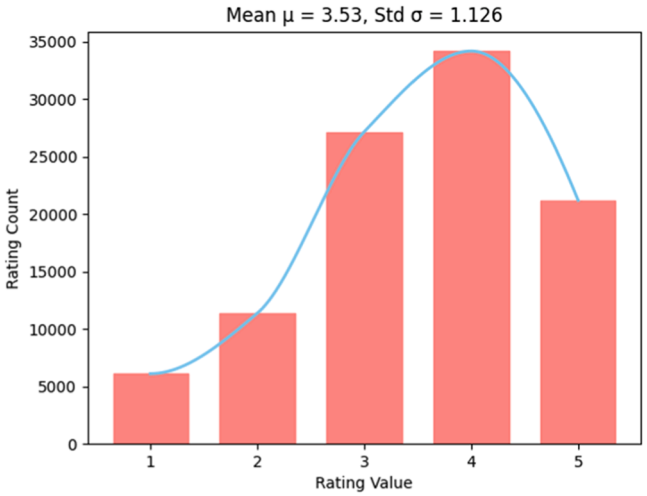

Fig. 2. The distribution of the original rating matrix.

Recall: It is used to measure the degree of item matching between predicted and actual recommendation lists, indicating the accuracy of the recommendation model.

$$\text{Recall@}N = \frac{|I_p \cap I_a|}{|I_a|} \tag{23}$$

where N represents the length of the recommendation list, I_p and I_a refer to the predicted and actual recommendation lists, respectively. Note that the ultimate goal of most RSs is to recommend a set of items that the active user is most likely interested in, thus we assume that if the predicted or actual rating of an item is greater than or equal to the median of the rating scale, the item is added to the recommended list. Then, an internal ranking is performed based on the rating value, and the top N items are selected as the final recommendation results.

NDCG: It is used to measure the degree of relevance of the ranking of recommended items in the predicted recommendation list to the actual recommendation list, indicating the quality of the recommendation model.

$$\text{NDCG@}N = \frac{\text{DCG@}N}{\text{IDCG@}N} \tag{24}$$

$$\text{DCG@}N = \sum_{p=1}^{N} \frac{2^{rel_p} - 1}{\log_2^{(p+1)}} \tag{25}$$

where DCG represents the discounted cumulative gain of recommending the top N items to the target user, and IDCG is the ideal DCG; rel_p denotes the relevance of the recommended item at position p in the sorted list. Here, rel_p is 1 if an item is in the actual recommendation list, and 0 otherwise. It is worth noting that considering the partitioning method and sparsity level of the dataset, we fix the length of the recommendation list N for each user to 5 through statistical analysis.

4.3. Comparison method

To validate the effectiveness of our model, several representative recommendation models are selected for a comparative analysis of performance metrics.

1) MF [35]: Matrix factorization is a classic model-based collaborative filtering that uses ALS method to update latent feature matrices for users and items.
2) NCF [36]: Neural collaborative filtering is a classic nonlinear recommendation model that utilizes a fully connected neural network to capture complex interactions between users and items.
3) ICDAE [37]: Improved autoencoder recommendation model utilizes active user interaction information to optimize model training and alleviate the vanishing gradient problem. It is used to verify the recommendation accuracy of our model.

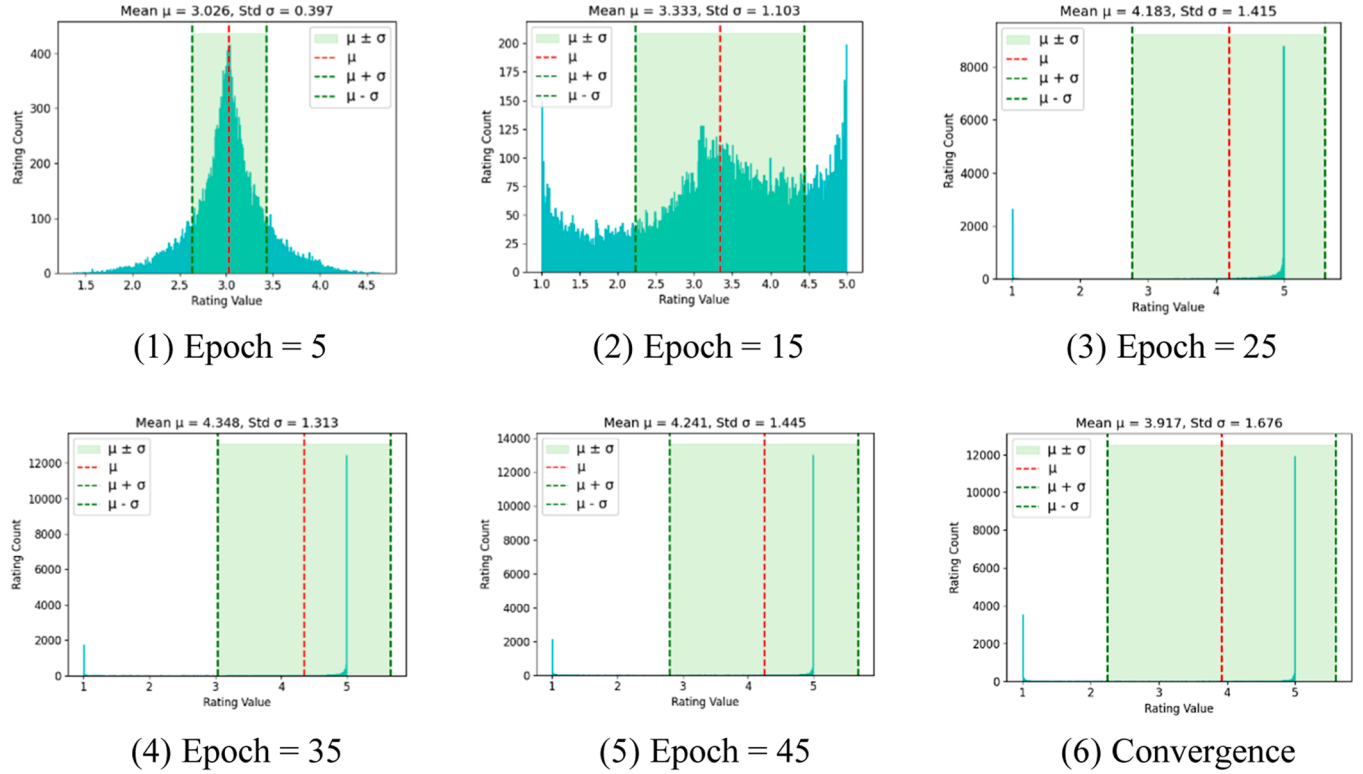

Fig. 3. Predicted rating distribution of CFGAN at different epochs (Labeled).

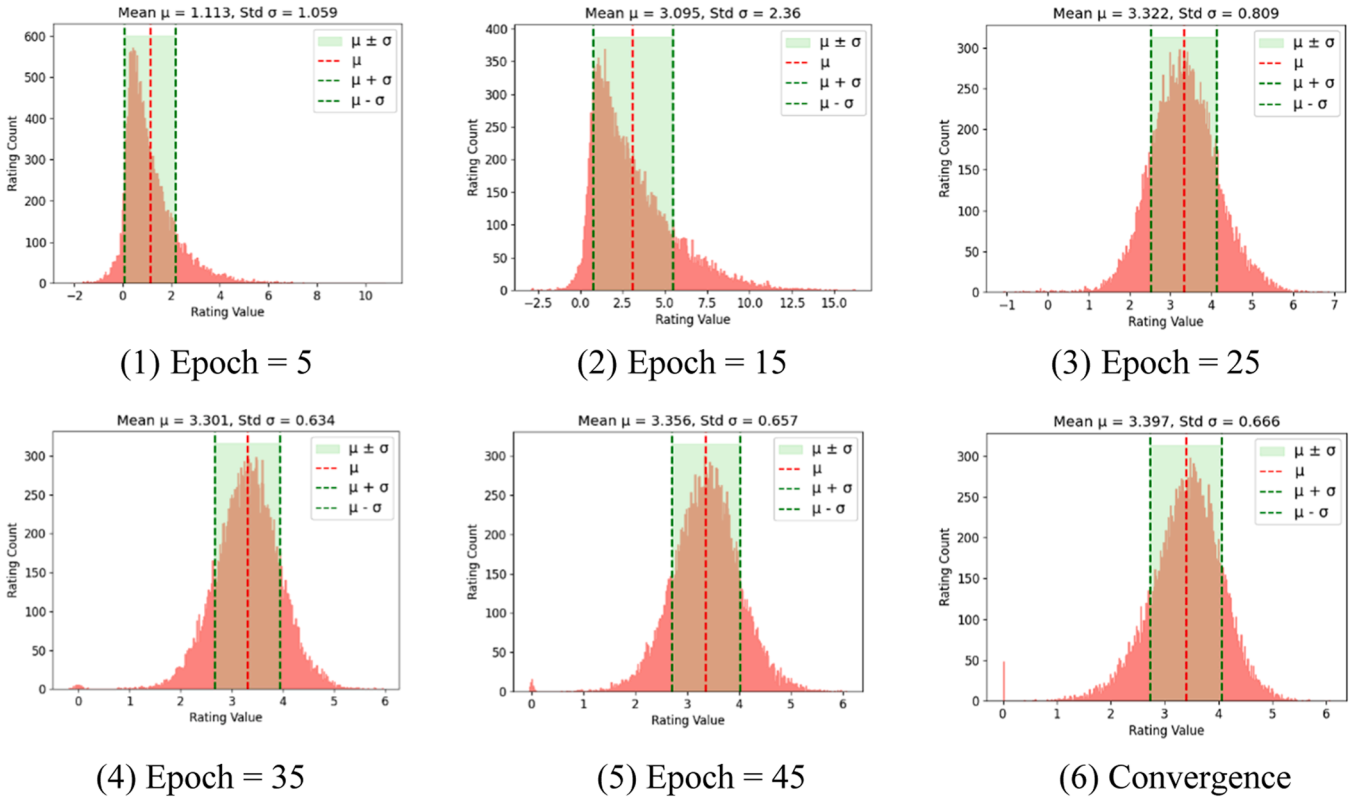

Fig. 4. Predicted rating distribution of DGRM at different epochs (Labeled).

4) CFGAN [15]: GAN-based collaborative filtering employs vector-wise training to predict user-item interaction probabilities. It is used to verify the effect of introducing DM in our model.
5) PRGAN [19]: Conditional GANs-based recommendation model uses user-item feature interactions as conditional vectors for predictions. It is used to examine the effect of conditional vectors on recommendation effectiveness.
6) DiffRec [33]: Diffusion recommender model reconstructs user interest preferences to capture the temporal patterns in users' interactions. It is used to test the effect of GAN in our model on recommendation performance.
7) DGRM-KL: A variant of our model DGRM, which adopts KL distance instead of Wasserstein distance. It is used to verify the performance of the proposed model using different distance metrics.

The hyper-parameter settings of all comparison methods satisfy the principle of grid search method and their detailed settings are presented in Appendix A.

4.4. Result analysis and discussion

4.4.1. Mode collapse validation

To verify that our model DGRM can effectively alleviate the mode collapse problem in traditional GANs, we take the ML-100K dataset as an example and investigate the distribution of prediction results of DGRM for labeled (rated) and unlabeled (unrated) items [22]. Fig. 2 shows the distribution of the original rating matrix. It can be seen from Fig. 2 that the mean of the rating distribution is 3.53, and it is right-skewed. This indicates that users in the dataset tend to give higher ratings ($\geq$4).

For labeled items, we display the distribution of predictions at different epochs to test the distribution consistency of predicted and original ratings during the training process of CFGAN and DGRM, as shown in Figs. 3 and 4, respectively. From Fig. 3, it can be observed that the CFGAN, in the early stages ($\leq$15), does not exhibit mode collapse, but it gradually tends to generate high ratings at epoch equals 25. Finally, it can only predict extreme rating values, viz. nearby 0 and 5. In contrast, it can be seen from Fig. 4 that our model DGRM does not have the mode collapse problem until it converges. This is because it can continuously reconstruct and learn user rating preferences during model training. Moreover, we also find that the distribution of our prediction results gradually approaches the original distribution, indicating its ability to capture user real interest preferences. In addition, we calculate the KL distance between the predicted rating distribution after model convergence and its corresponding original rating distribution for the testing set. A smaller KL value indicates a closer distance between two distributions. The result is presented in Table 2. From Table 2, it is evident that the predicted rating distribution of the DGRM is closer to the original rating distribution compared with the CFGAN, which indicates that our model has a better prediction effect.

Table 2
KL distance between the predicted and original rating distributions.

Method	KL distance
DGRM	0.041
CFGAN	0.229

For unlabeled items, we also collect the predicted rating distributions generated at different epochs during the training processes of the two models, as shown in Figs. 5 and 6. Since unlabeled items have no rating value, we are unable to evaluate their prediction accuracy. However, starting from the definition of mode collapse, it can be observed from Figs. 5 and 6 that compared with the CFGAN, our model DGRM covers almost all modes, viz. all rating interval values, with a significantly higher coverage of prediction ratings. Meanwhile, based on user rating preferences, we aim for the model to predict the distribution of unrated items in accordance with the preferences of users in the dataset. As

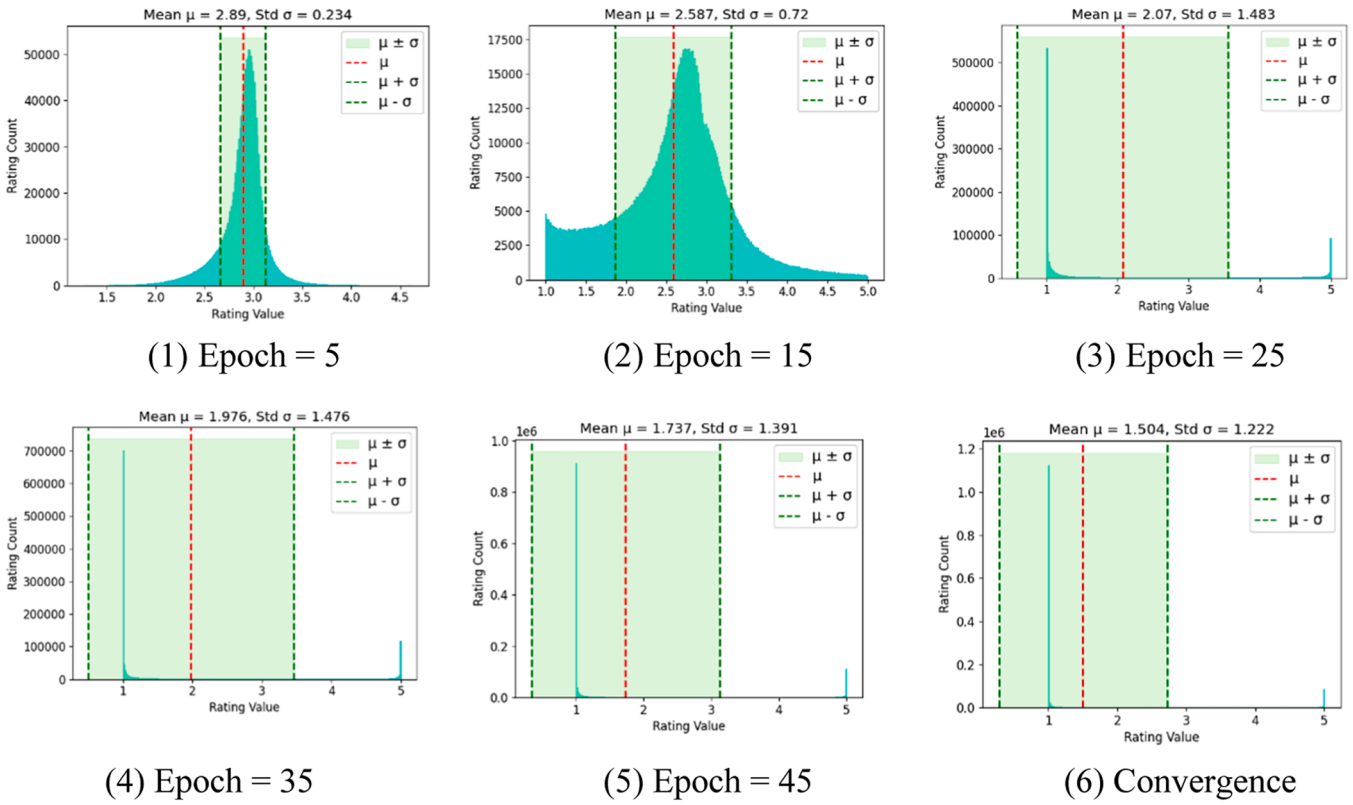

Fig. 5. Predicted rating distribution of CFGAN at different epochs (Unlabeled).

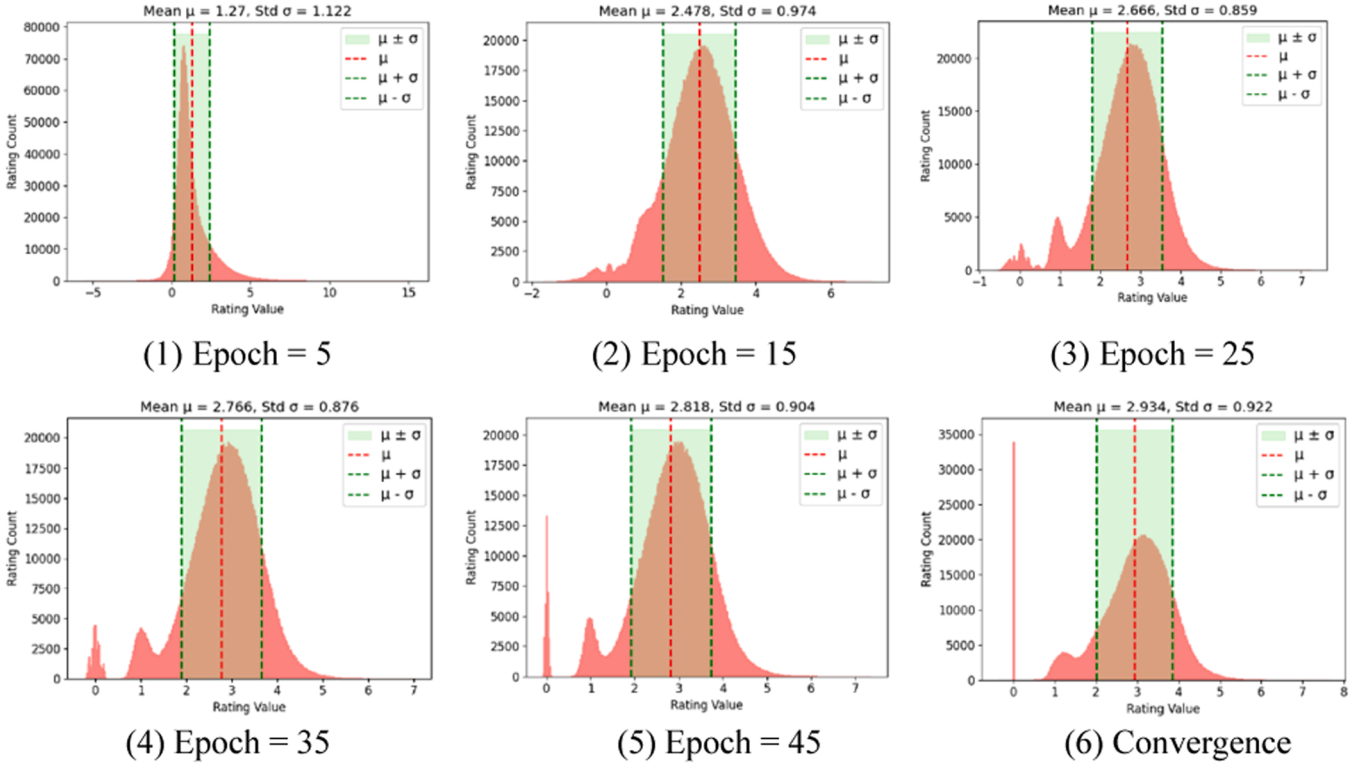

Fig. 6. Predicted rating distribution of DGRM at different epochs (Unlabeled).

illustrated in Fig. 6, with the gradual convergence of the model, the mean of the rating distribution predicted by the DGRM is closer and closer to that of the original rating distribution, and its distribution is also right-skewed. Moreover, the standard deviation of our prediction distribution has remained in a stable interval, which indicates that our model has strong stability.

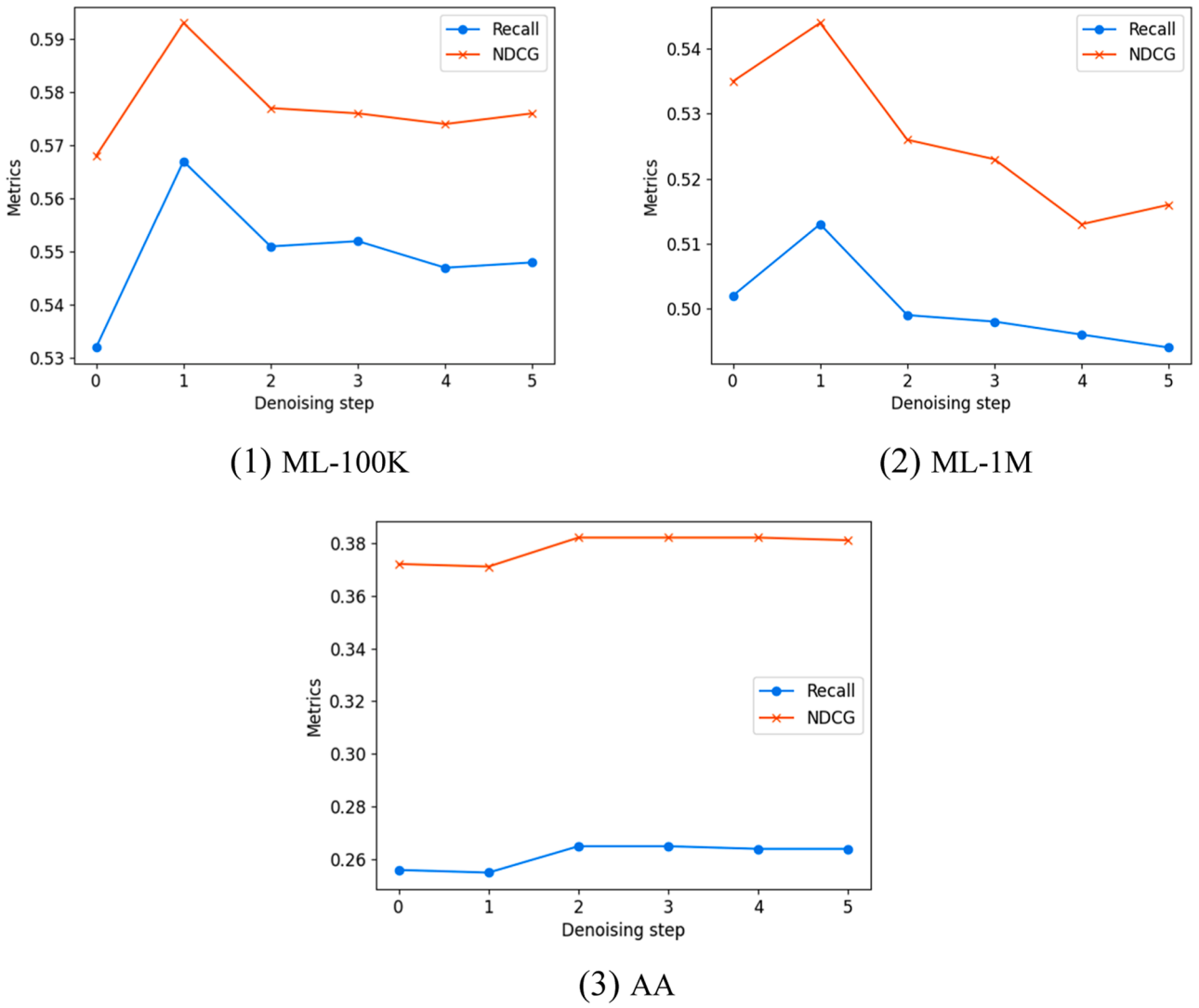

Fig. 7. Effects of denoising step T' on the model performance in three datasets.

Here, we will give an in-depth analysis of the reasons for the above results. Firstly, due to the loss function of the CFGAN being based on KL distance, its penalty for lack of diversity in prediction results is much smaller than the penalty for lack of accuracy. This makes the generator G prefer sacrificing sample diversity solely to pursue accuracy, resulting in mode collapse [26]. Secondly, due to the sparse nature of the training data, even with the masking layer simulating the sparsity of the original rating matrix, for the discriminator D, the changes of rating values at individual positions in the sparse matrix significantly impact the distribution. This sensitivity enables the D to easily discern the authenticity of sample matrices, leading to rapid convergence as the optimal discriminator and causing gradient vanishing for the G, thereby exacerbating the process of mode collapse. In contrast, the DGRM initially adopts the WGAN framework to optimize the traditional GAN loss function, ensuring that Wasserstein distance provides valuable gradients in all circumstances. Additionally, by employing the forward and backward processes of diffusion model, we ingeniously alter the task of the D. The D is required to calculate the Wasserstein distance between two dense matrices at different step t, rather than identifying the authenticity of two sparse matrices. Although this increases the learning difficulty of the D, and it also improves the stability of the model training process.

Table 3
Performance of all comparison methods on different datasets (the best is in bold).

Method	ML-100K		ML-1M		AA	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
MF	0.390	0.418	0.389	0.396	0.209	0.305
NCF	0.513	0.532	0.472	0.496	0.221	0.318
ICDAE	0.523	0.541	0.496	0.521	0.224	0.325
CFGAN	0.494	0.527	0.475	0.501	0.223	0.326
PRGAN	0.527	0.537	0.455	0.469	0.218	0.324
DiffRec	0.538	0.560	0.491	0.522	0.230	0.337
DGRM-KL	0.558	0.573	0.492	0.501	0.259	0.376
DGRM	**0.567**	**0.593**	**0.513**	**0.544**	**0.265**	**0.382**

4.4.2. *Effects of denoising step T'*

During the model training process, we need to utilize the forward and backward diffusion processes to assist the WGAN training. Here, two hyper-parameters, i.e., diffusion step T and denoising step T', are involved to investigate the effects on the performance of our scheme DGRM in two metrics. For the diffusion step T, based on previous studies [28,33], its value cannot be set too large, otherwise (1) it disturbs the user's personalized information, resulting in the conditional vector ineffective; (2) it reduces the efficiency of model training. Thus, we do not test effects of the diffusion step T on the performance of our model in the experiment, and the step T is fixed to 5. For the denoising step T', to

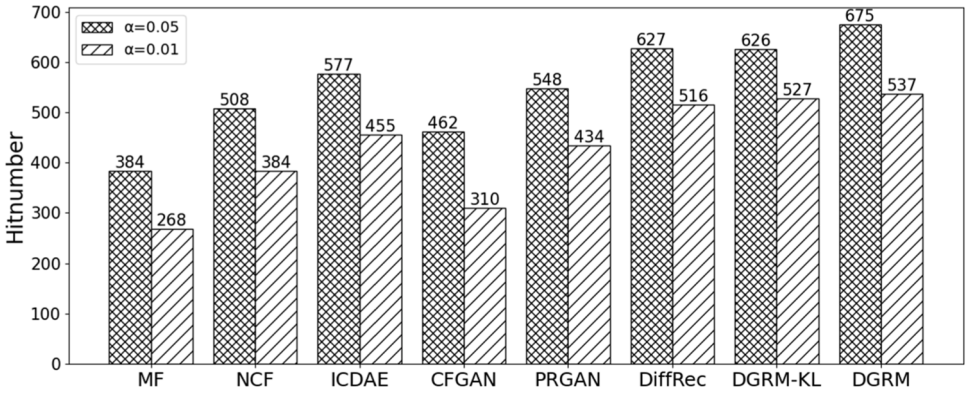

Fig. 8. The outcomes of KS test at different significance levels of α.

maximize the retention of user's personalized information for recommendations, the step of backward process is set to be $T' < T$and the process $\mathbf{R}^{T} \rightarrow \mathbf{R}^{T-1} \rightarrow \cdots \rightarrow \mathbf{R}^{T-T'}$ is executed. Note that the denoising step varies from different datasets. Therefore, we test it in three datasets, and the results are shown in Fig. 7.

It can be observed from Fig. 7 that the best performance of our model is at $T' = 1$ on the datasets ML, and the step T' is 2 on the AA dataset. We find that the model is more sensitive to changes in the denoising step T' on the relatively dense datasets ML; while on the relatively sparse dataset AA, it shows little change in recommendation performance when $T' \neq 0$. It is noteworthy that the backward process does not work when $T' = 0$, and the recommendation results are directly generated by the G. As a result, we claim that too few or even zero executions of the backward process may retain unnecessary noise and disturb the user's original preferences, leading to inaccurate prediction results. Conversely, due to the inherent noise in the original rating matrix, too many denoising steps may make the model over-sensitive to noise, thus affecting the predictive ability of the model.

4.4.3. Comparison validation

To validate the recommendation effectiveness of the proposed model DGRM, the evaluation results of several SOTA methods mentioned in Section 4.3 are compared on different datasets. The comparison results are shown in Table 3. It can be seen from Table 3 that the performance of DGRM is superior to other comparison models on all datasets, especially with outstanding performance on the dataset AA for both metrics. For GAN-based models, the PRGAN achieves relatively better results compared with CFGAN, as it can capture more user preference features through the conditional vector in the memory module. While our model integrates DM to optimize GAN learning, resulting in a performance improvement of at least 3.4 % and 4.2 % for metrics Recall and NDCG on the dataset ML-1M, respectively; and up to 15.2 % and 13.4 % on the AA dataset. For the linear-based model MF, it relies on the amount of interaction information between users and items, usually performing poorly on the sparser datasets, especially on the AA, where the results are the worst, with the Recall and NDCG being 0.209 and 0.305, respectively, far inferior to deep learning-based models. On the other hand, the ICDAE is a non-linear model that can capture complex features between users and items even in situations with limited information. Compared with the NCF, it effectively utilizes rating information from active users, thus demonstrating superior performance among all comparison methods. The DiffRec performs the best among other non-linear models, especially on the sparser dataset AA, where it is our closest competitor. Because our model uses GAN to denoise complex distribution noise in the rating matrix rather than relying on Gaussian distribution noise, obtaining the best performance. Compared with the sub-optimal model DiffRec, our method achieves an average performance improvement of at least 7.97 % and 7.83 % in Recall and NDCG metrics, respectively. Therefore, we introduce Gaussian noise into the original rating matrix through the diffusion forward process, and use the G to reconstruct and learn user preference information, while combining the diffusion backward process to optimize the learning of the D, which can effectively improve the quality of RSs.

In addition, an ablation study is conducted to verify the effectiveness of DGRM using different distance metrics. The results are presented in Table 3. From Table 3, it can be seen that our model DGRM (using Wasserstein distance) is superior to that using KL distance in various metrics. This is because using Wasserstein distance as a loss function can better deal with the problem of GAN model training instability compared with using KL distance that is asymmetric, which effectively alleviates mode collapse. Moreover, we also find that the DGRM-KL outperforms other comparison methods on most datasets (except ML-1M), indicating the superiority of our framework.

4.4.4. Kolmogorov-Smirnov test

To further demonstrate the validity of the proposed model, the two-sample Kolmogorov-Smirnov (KS) test [38] is adopted to count the number of the cumulative distributions of predicted and actual ratings on the test set that are identical. In this experiment, we organize the predicted ratings of each user into a treatment group and the actual ratings into a control group, analyzing a total of 943 pairs from the ML-100K dataset. Each group pair is tested at different significance levels of α. The outcomes of the KS test are counted using a cumulative test value, denoted as q, which serves as the hit number. If $q = 1$, supporting for the null hypothesis; otherwise, $q = 0$. The statistic is depicted in Fig. 8. It can be found from Fig. 8 that the hit number of DGRM significantly surpasses those of comparison methods at different α values, indicating the superior prediction accuracy and robustness of our method.

5. Conclusion and future work

To address the problem of mode collapse in traditional GAN-based models, particularly in sparse environments, we propose an efficient generative recommendation model that integrates DM with GAN, named DGRM. This model facilitates mutual enhancement between the two generative models. Firstly, the introduction of DM helps avoid the gradient disappearance problem in the GAN generator, effectively addressing the mode collapse problem. Secondly, employing the generator for a one-step denoising replaces the need for stepwise denoising in the backward process of DM, which greatly improves the efficiency of the model. In addition, we adopt the Wasserstein distance as a loss function in GAN to ensure stable model training. Experiments on three datasets show that our model is better than other comparison methods in terms of recommendation accuracy. Furthermore, compared with a classic GAN-based model (CFGAN), our model can identify a

broader range of rating patterns and generate a better prediction distribution, thus effectively alleviating mode collapse and enhancing the predictive ability.

Although the application of DGRM in the field of RSs opens up new insights for the fusion of DM and GAN, our model still encounters several limitations. On the one hand, for users with varying levels of interaction, determining the appropriate conditional vector remains challenge due to the sensitivity of the added noise to parameter changes. On the other hand, the use of random Sampling noise can lead to the loss of personalized user information, which may blur the preferences captured by the model and impact the effectiveness of the recommendations. To investigate these concerns, a validation experiment was conducted on the ML-100K dataset (see Appendix B) to explore these limitations further. Looking ahead, we have identified several promising directions for future research. One approach is to integrate an attention mechanism into the model to generate more personalized conditional vectors for users, rather than obtaining them randomly. Additionally, incorporating more advanced structures, such as graph neural networks and transformers, could enhance the complexity of the generator and discriminator, potentially improving the performance of RSs.

CRediT authorship contribution statement

Deng Jiangzhou: Writing – review & editing, Validation, Resources, Project administration, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Wang Songli:** Writing – original draft, Visualization, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Ye Jianmei:** Writing – review & editing, Validation, Resources, Funding acquisition, Formal analysis. **Ji Lianghao:** Writing – review & editing, Validation, Software, Resources, Funding acquisition. **Wang Yong:** Writing – review & editing, Validation, Supervision, Resources, Project administration, Investigation, Funding acquisition, Conceptualization.

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.

Data availability

Data will be made available on request.

Acknowledgments

This work has been supported by the National Natural Science Foundation of China (Grant No. 62272077 and 72301050), the Science and Technology Research Program of Chongqing Municipal Education Commission (Grant No. KJQN202300605 and KJQN202100603), and the Humanities and Social Science project of the Ministry of Education of China (Grant No. 23YJC630215).

Appendices

A. Parameter settings

The hyper-parameter settings of all comparison methods satisfy the principle of grid search method. For the DGRM, in the forward process, the noise steps are set to 5, and the noise scale, upper and lower bounds of noise are searched in {0, $1e^{-4}$, $1e^{-2}$, $1e^{-1}$, 1}, {$1e^{-4}$, $1e^{-3}$, $1e^{-2}$}, and {$5e^{-3}$, $1e^{-1}$, 1}, respectively. For the generator and discriminator, the number of iterations is chosen from {2, 4} and {1, 2}, respectively; regularization coefficient is set to 0.1, the step embedding size is fixed to 10, and the learning rate is selected from {$1e^{-4}$, $1e^{-3}$, $1e^{-2}$}. In the backward process, the denoising steps are searched in {0, 1, 2, 3, 4, 5}.

For the MF, L2 regularization is determined by {$1e^{-1}$, $1e^{-2}$}, and the number of latent factors is chosen from {8, 10, 12}. For the NCF, the learning rate is selected from {$1e^{-3}$, $1e^{-2}$}, and the number of latent factors is chosen from {12, 14, 16}. For the ICDAE, the learning rate is searched in {$1e^{-4}$, $1e^{-3}$, $1e^{-2}$}, the number of latent vectors is chosen from {24, 32, 64}, and the weight decay is selected from {$1e^{-4}$, $1e^{-2}$}. For the CFGAN, the generator and discriminator alternate training times are searched in {2, 4} and {1, 2}, respectively, the learning rate is from {$1e^{-4}$, $1e^{-3}$, $1e^{-2}$}, the number of negative samples are determined by {70, 200, 400}, and the reconstruction regularization is set to 0.1. For the PRGAN, training times of the G and D are searched in {2, 4} and {3, 5}, respectively; the learning rate is chosen from {$1e^{-3}$, $1e^{-2}$}, and the gradient penalty is selected from {$1e^{-2}$, $1e^{-1}$}. For the DiffRec, the learning rate is determined by {$1e^{-4}$, $1e^{-3}$, $1e^{-2}$}, the number of latent factors is searched in {256, 512, 1024}, and the noise-adding and denoising steps are chosen from {2, 5, 10} and {0, 1, 4}, respectively.

B. Exploratory experiment

A validation experiment is conducted on the ML-100 K dataset to explore the limitations of our method. Initially, the number of steps in the forward process is fixed at $T = 5$ and $T = 1$, representing the addition of maximum and minimum noise levels to all user vectors during the training process, while keeping the backward process unchanged. The result is illustrated in Table 4.

Table 4
Effects of different fixed steps on our model.

Method	Recall	NDCG
DGRM ($T = 5$)	0.547	0.572
DGRM ($T = 1$)	0.557	0.583
DGRM	0.567	0.593

As shown in Table 4, indiscriminately adding relatively high noise to all user vectors, without considering the number of user interaction data, leads to the loss of user personalized information. This approach makes it difficult for the model to accurately capture the true preferences and interests of users. Moreover, while reducing the noise level can enhance the performance of recommendations, too low a noise level can break the robustness of the model, resulting in poor performance.

In addition, although random noise addition does not guarantee the optimal noise level for users with varying amounts of information, it has a certain probability of appropriately assisting the training process. Our future work will also focus on how to incorporate a reasonable amount of noise that not only preserves user personalized information but also enhances the accuracy and the robustness of the model.

C. Theoretical explanation of using Wasserstein distance instead of KL divergence

Suppose that $\mathbf{P}_g(x)$ and $\mathbf{P}_r(x)$ represent the generator's distribution and the unknown real distribution, for a discrete rating dataset, KL divergence is defined by

$$KL\left[\mathbf{P}_g(X)\|\ \mathbf{P}_r(X)\right] = \sum_{x\in X}\left[\mathbf{P}_g(x)\log\frac{\mathbf{P}_g(x)}{\mathbf{P}_r(x)}\right] = E_{x\sim P_g(x)}\left[\log\frac{\mathbf{P}_g(x)}{\mathbf{P}_r(x)}\right] \tag{26}$$

By definition, KL divergence is an asymmetric measure of distributions, viz. $KL\left[\mathbf{P}_g(X)\ \|\mathbf{P}_r(X)\right] \neq KL\left[\mathbf{P}_r(X)\ \|\mathbf{P}_g(X)\right]$. Especially, KL divergence as a loss function of GAN have completely different penalties in the following two extreme cases.

$$KL\left[\mathbf{P}_g(X)\|\mathbf{P}_r(X)\right] = E_{x\sim\mathbf{P}_g(x)}\left[\log\frac{\mathbf{P}_g(x)}{\mathbf{P}_r(x)}\right] \to \begin{cases} 0, & \text{if } \mathbf{P}_g(x)\to 0, \mathbf{P}_r(x)\to 1, \\ \infty, & \text{if } \mathbf{P}_g(x)\to 1, \mathbf{P}_r(x)\to 0. \end{cases} \tag{27}$$

The explanation is as follows: for a generated sample x,

1) If $\mathbf{P}_g(x)\to 0$ and $\mathbf{P}_r(x)\to 1$, $\mathbf{P}_g(x)\log\frac{\mathbf{P}_g(x)}{\mathbf{P}_r(x)}\to 0$, which means the sample x generated by G has a higher probability of coming from the real sample rather than the generated sample. Namely, the larger the $\mathbf{P}_g(x)$, the less real the generated sample x. This indicates that G generates samples not covering parts of real data, leading to lack of diversity in the generated samples. In this case, as $KL\to 0$, the penalty on G is little.

2) If $\mathbf{P}_g(x)\to 1$ and $\mathbf{P}_r(x)\to 0$, $\mathbf{P}_g(x)\log\frac{\mathbf{P}_g(x)}{\mathbf{P}_r(x)}\to\infty$, which means the sample x generated by G has a higher probability of coming from the generated sample rather than the real sample. Namely, the larger the $\mathbf{P}_r(x)$, the more real the generated sample x. This indicates that G generates fake samples, leading to lack of accuracy in the generated samples. In this case, as $KL\to\infty$, the penalty on G is huge.

As a result, we see from the above two cases that the asymmetry of KL divergence makes the G prefer to generate more repeated but safe samples rather than diverse samples in the adversarial training. Because the occurrence of Case 2 easily generates unreal samples, resulting in larger loss value and slower gradient descent speed. However, we want a loss function that is unbiased and guides the GAN to learn all possible distributions fairly. The symmetry of Wasserstein distance (Eq. (6)) makes up for this.

From Eq. (6), we obviously find that it is a symmetric measure of distributions. Whether $\mathbf{P}_g(x)\to 0$ and $\mathbf{P}_r(x)\to 1$ or $\mathbf{P}_g(x)\to 1$ and $\mathbf{P}_r(x)\to 0$, its value is the same. This indicates that Wasserstein distance as a loss function to guide G learning does not tend to generate certain fixed patterns in pursuit of the faster gradient descent direction. Thus, Wasserstein distance is more suitable as a loss function for GAN than KL divergence. In addition, an ablation study was conducted to verify the effectiveness of our model using the two-distance metrics in Section 4.4.3, the result is shown in Table 3.

References

[1] H. Liu, Y. Wang, Z. Zhang, et al., Matrix factorization recommender based on adaptive Gaussian differential privacy for implicit feedback, Inf. Process. Manag. 61 (4) (2024) 103720.

[2] J. Gong, Y. Zhao, J. Zhao, et al., Personalized recommendation via inductive spatiotemporal graph neural network, Pattern Recognit. 145 (2024) 109884.

[3] J. Deng, Q. Wu, S. Wang, et al., A novel joint neural collaborative filtering incorporating rating reliability, Inf. Sci. (Ny) 663 (2024) 120406.

[4] S. Wang, H. Tao, J. Li, et al., Towards fair and personalized federated recommendation, Pattern Recognit. 149 (2024) 110234.

[5] Q. Zhang, W. Liao, B.Yuan G.Zhang, J. Lu, A deep dual adversarial network for cross-domain recommendation, IEEE Trans. Knowl. Data Eng. 35 (4) (2021) 3266–3278.

[6] Q. Guo, et al., A survey on knowledge graph-based recommender systems, IEEE Trans. Knowl. Data Eng. 34 (8) (2020) 3549–3568.

[7] I. Goodfellow, et al., Generative adversarial networks, Commun. ACM 63 (11) (2020) 139–144.

[8] R. Zhang, et al., RFI-GAN: a reference-guided fuzzy integral network for ultrasound image augmentation, Inf. Sci. (Ny) 623 (2023) 709–728.

[9] J. Tang, Z. Gong, Z.Yin B.Tao, Advancing generalizations of multi-scale GAN via adversarial perturbation augmentations, Knowl. Based Syst. 284 (2024) 111260.

[10] S. Dey, T. Chakraborti, P. Basuchowdhuri, et al., MoMSGAN: mode collapse based degradation agnostic multi-scale super-resolution of medical images, in: Proceedings of the 14th Indian Conference on Computer Vision, Graphics and Image Processing, 2023, pp. 1–9.

[11] Q. Liu, Q. Zhang, W. Liu, et al., WSDS-GAN: a weak-strong dual supervised learning method for underwater image enhancement, Pattern Recognit. 143 (2023) 109774.

[12] T. Kaneko, H. Kameoka, Cyclegan-vc: non-parallel voice conversion using cycle-consistent adversarial networks, in: Proceeding of 26th European Signal Processing Conference, 2018, pp. 2100–2104.

[13] S. Dhar, N. Jana, S. Das, GLGAN-VC: a guided loss-based generative adversarial network for many-to-many voice conversion, IEEE Trans. Neural Netw. Learn. Syst. (2023) 1–14.

[14] M. Gao M, et al., Recommender systems based on generative adversarial networks: a problem-driven perspective, Inf. Sci. (Ny) 546 (2021) 1166–1185.

[15] D. Chae, J. Kang, S. Kim, J. Lee, Cfgan: a generic collaborative filtering framework based on generative adversarial networks, in: Proceeding of the 27th ACM International Conference on Information and Knowledge Management, 2018, pp. 137–146.

[16] J. Wang, et al., IRGAN: a minimax game for unifying generative and discriminative information retrieval models, in: Proceeding of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2017, pp. 515–524.

[17] H. Wang, et al., GraphGAN: graph Representation Learning with Generative Adversarial Nets, in: Proceeding of the Thirty-Second AAAI Conference on Artificial Intelligence, 2018, pp. 2508–2515.

[18] Q. Wang, et al., Enhancing collaborative filtering with generative augmentation, in: Proceeding of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 548–556.

[19] J. Wen, X. Zhu, C. Wang, Z. Tian, A framework for personalized recommendation with conditional generative adversarial networks, Knowl. Inf. Syst. 64 (10) (2022) 2637–2660.

[20] Z. Yang, et al., GANRec: a negative sampling model with generative adversarial network for recommendation, Expert Syst. Appl. 214 (2023) 119155.

[21] C. Chen, P. Lai, C. Chen, ColdGAN: an effective cold-start recommendation system for new users based on generative adversarial networks, Appl. Intell. 53 (7) (2023) 8302–8317.

[22] T. Che, Y. Li, A. Jacob, et al., Mode regularized generative adversarial networks, in: Proceedings of International Conference on Learning Representations, 2017, pp. 1–13.

[23] M. Arjovsky, L. Bottou, Towards principled methods for training generative adversarial networks, in: Proceedings of International Conference on Learning Representations, 2017, pp. 1–17.

[24] N. Kodali, J. Abernethy, James Hays, et al., On convergence and stability of gans, 2017, arXiv preprint arXiv:1705.07215.

[25] T. Salimans, et al., Improved techniques for training gans, in: Advances in Neural Information Processing Systems, 2016, pp. 2234–2242.

[26] M. Arjovsky, S. Chintala, L. Bottou, Wasserstein gan, 2017, arXiv preprint arXiv:1701.07875.

[27] I. Gulrajani, et al., Improved training of wasserstein gans, in: Advances in Neural Information Processing Systems, 2017, pp: 5767–5777.

[28] Z. Xiao, K. Kreis, A. Vahdat, Tackling the generative learning trilemma with denoising diffusion gans, in: Proceedings of the Tenth International Conference on Learning Representations, 2022, pp. 1–28.

[29] Z. Wang Z, et al., Diffusion-gan: training gans with diffusion, in: Proceedings of the Eleventh International Conference on Learning Representations, 2023, pp. 1–26.

[30] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, et al., Deep unsupervised learning using nonequilibrium thermodynamics, in: Proceedings of the 32nd International Conference on Machine Learning, 2015, pp. 2256–2265.

[31] J. Ho, A. Jain, P. Abbeel, Denoising diffusion probabilistic models, in: Advances in neural information processing systems, 2020, pp. 6840–6851.

[32] Y. Song, J. Sohl-Dickstein, D. Kingma, et al., Score-based generative modeling through stochastic differential equations, in: Proceedings of International Conference on Learning Representations, 2021, pp. 1–36.

[33] W. Wang, Y. Xu, F. Feng, X. Lin, X. He, T. Chua, Diffusion recommender model, in: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2023, pp. 832–841.

[34] J. Zhao, W. Wang, Y. Xu, T. Sun, F. Feng, Denoising diffusion recommender model, 2024, arXiv preprint arXiv:2401.06982.

[35] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (8) (2009) 30–37.
[36] X. He, et al., Neural collaborative filtering, in: Proceeding of the 26th International Conference on World Wide Web, 2017, pp. 173–182.
[37] D. Liu, Y. Wang, C. Luo, J. Ma, An improved autoencoder for recommendation to alleviate the vanishing gradient problem, Knowl. Based Syst. 263 (2023) 110254.
[38] J. Deng, J. Guo, Y. Wang, A Novel K-medoids clustering recommendation algorithm based on probability distribution for collaborative filtering, Knowl. Based Syst. 175 (2019) 96–106.

Jiangzhou Deng received his Ph.D. degree in Management Science from Tianjin University. He is now lecturer of Dept. of management engineering in Chongqing University of Posts and Telecommunications. His main research interests include decision optimization and recommender systems. He has authored over 15 papers in international journals, China journals.

Songli Wang is currently pursuing the master's degree in management science and engineering from the School of Economics and Management, Chongqing University of Posts and Telecommunications. His main research interests involve recommender systems and deep learning.

Jianmei Ye received her Ph.D. degree in management science from Sichuan University, Chengdu, China, in 2020. She is a Lecturer in the School of Economics and Management, Chongqing University of Posts and Telecommunications, Chongqing, China. Her current research interests include fuzzy set theory, group decision making, Bayesian network, evidential reasoning, etc.

Lianghao Ji received his Ph.D. degree in computer science and technology from Chongqing University, Chongqing, in 2014. He is now professor of Dept. of computer science and technology in Chongqing University of Posts and Telecommunications. His current research interests include multiagent systems, complex and intelligent systems, and intelligent information processing.

Yong Wang received his Ph.D. degree in computer science from Chongqing University, China, 2007. He is now professor of Dept. of management engineering in Chongqing University of Posts and Telecommunications. His research interests involve information management, data analysis and recommender systems. He has published more than 50 refereed journal.