Recommendation via Collaborative Diffusion Generative Model

Joojo Walker[1], Ting Zhong[1], Fengli Zhang[1], Qiang Gao[3], and Fan Zhou[1,2](✉)

[1] University of Electronic Science and Technology of China, 610054 Chengdu, Sichuan, China
fan.zhou@uestc.edu.cn
[2] Kash Institute of Electronics and Information Industry, 844000 Kashgar, Xinjiang, China
[3] Southwestern University of Finance and Economics, 611130 Chengdu, Sichuan, China

Abstract. Despite the success of classical collaborative filtering (CF) methods in the recommendation systems domain, we point out two issues that essentially limit this class of models. **Firstly**, most classical CF models predominantly yield weak collaborative signals, which makes them deliver suboptimal recommendation performance. **Secondly**, most classical CF models produce unsatisfactory latent representations resulting in poor model generalization and performance. To address these limitations, this paper presents the Collaborative Diffusion Generative Model (CODIGEM), the *first-ever* denoising diffusion probabilistic model (DDPM)-based CF model. CODIGEM effectively models user-item interactions data by obtaining the intricate and non-linear patterns to generate strong collaborative signals and robust latent representations for improving the model's generalizability and recommendation performance. Empirically, we demonstrate that CODIGEM is a very efficient generative CF model, and it outperforms several classical CF models on several real-world datasets. Moreover, we illustrate through experimental validation the settings that make CODIGEM provide the most significant recommendation performance, highlighting the importance of using the DDPM in recommendation systems.

Keywords: Recommendation systems · Collaborative filtering · Denoising diffusion probabilistic model · Generative model

1 Introduction

The enormous magnitude of user-item interactions data on the internet today has necessitated the design of various personalized recommendation models to deliver to users a set of unseen items that may be of interest to them [2,15,22,23]. Among the several recommendation techniques available, classical collaborative filtering (CF)-based techniques have been widely adopted. Classical CF methods predict the preferences of users for items by learning from user-item historical interactions, employing either explicit feedback (e.g., ratings and reviews) or implicit feedback (e.g., clicks and views) [3,9,15]. In general, there are two kinds of classical CF-based approaches: neighborhood-based techniques and model-based methods [15]. The neighborhood-based approaches such as Item-KNN [5]) use original user-item interaction data (e.g., rating matrices) to infer unseen ratings by combining similar users' preferences or similar items. On the other hand, model-based methods such as matrix factorization (MF) [1,12,19]) obtain user tastes for items by employing the idea that a low-dimensional latent vector might represent a user's taste or an item's attribute. These matrix factorization techniques (latent factor models) deconstruct the high-dimensional user-item rating matrix into low-dimensional user and item latent vectors. Subsequently, recommendation prediction is made by computing the dot product of the latent vectors of the user and item [16,20]. It is important to note that Classical CF models are still dominant approaches both in the industry and academia due to their simplicity and intuitive justification for the computed predictions [4,15,23].

Challenges: Despite the success of classical collaborative filtering (CF) models, there are still pertinent issues with these models [1,12,19]. **Firstly**, most classical CF models predominantly yield weak collaborative signals, which makes them deliver suboptimal recommendation performance. This issue is because these classical CF models are designed with the notion that users and items have a linear relationship, so they do not capture the intricate information and non-linearities embedded in the real-world user-item interaction data. **Secondly**, most classical CF models produce unsatisfactory latent representations resulting in poor model generalization and performance. This problem is prevalent because the models cannot extensively capture high-quality collaborative information from the underlying data distribution due to their non-Bayesian properties.

Contributions: Recently, a new class of deep generative model (DGM) called denoising diffusion probabilistic models (DDPM) has achieved exceptional performance on several image synthesis benchmarks, even outperforming generative adversarial networks (GANs) [6,10,11,21]. Inspired by the outstanding results of DDPM, we extend DDPM to implicit feedback data-based collaborative filtering (CF) to address the pertinent limitations of classical CF models mentioned above. Notably, we systematically explore and design a novel DDPM-based CF model called Collaborative Diffusion Generative Model (CODIGEM). CODIGEM adopts a forward Gaussian diffusion process and a reversed diffusion procedure that utilizes flexible parameterized neural networks to capture high-quality collaborative information from the underlying implicit feedback data. This novel technique yields quality latent representations to improve the model's generalization and produce excellent recommendations. Overall, our main contributions can be summarized as follows:

- We present the *first-ever* DDPM-based model that effectively models the user-items interactions data to capture the intricate and non-linear patterns in order to generate strong collaborative signals for the implicit feedback-based recommendations.
- We alleviate the issue of unsatisfactory generation of latent representations by effectively capturing the underlying distribution of the implicit feedback data.

This technique enhances the model's generalization and yields outstanding recommendation results.

– We design an efficient deep generative model (DGM)-based CF model. We highlight that, unlike DDPM employed in image synthesis that uses a costly iterative sampling process, CODIGEM is very efficient as it achieves good performance using very few iterative sampling processes. Besides, when compared to a different class of DGM-based CF model [17,18], CODIGEM exhibit superior computational efficiency.
– Empirically, we demonstrate that CODIGEM outperforms several classical CF models on several real-world datasets. This performance highlights the importance of using the DDPM in recommendation systems. To ensure reproducibility, we release our codes via this link[1].

2 Collaborative Diffusion Generative Model

Overview: Our novel Collaborative Diffusion Generative Model (CODIGEM) is essentially a DDPM [10,21]. DDPM is a kind of DGM that consists of two main processes: a forward diffusion (noising) process (FDP) and the reverse diffusion (noising) process (RDP). The overall architecture of CODIGEM is illustrated in Fig. 1. Next, we describe the mathematical underpinnings and the model learning method of CODIGEM.

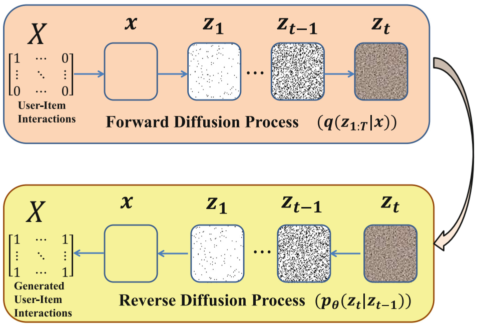
X
x
z_1
z_{t-1}
z_t
User-Item Interactions
Forward Diffusion Process $(q(z_{1:T}|x))$
Generated User-Item Interactions
Reverse Diffusion Process $(p_\theta(z_t|z_{t-1}))$

Fig. 1. An illustration of CODIGEM model architecture

Inspired by intuitions from non-equilibrium statistical physics Sohl-Dickstein et al. [21] proposed the first Deep diffusion probabilistic model (DDPM). The core intuition is to iteratively introduce noise into the underlying structure of data through a forward diffusion process (FDP). Next the structure of the data is regenerated through a reverse diffusion process (RDP). Recently, Ho et al.[10], designed a powerful and flexible DDPM framework that resulted in SOTA performance in the task of image synthesis. Generally, the goal is to determine a distribution over data, $p_\theta(\mathbf{x})$, but, we also consider other set of latent variables

[1] https://github.com/WorldChanger01/CODIGEM.

$\mathbf{z}_{1:T} = [\mathbf{z}_1, \ldots, \mathbf{z}_T]$. Now, we define the marginal likelihood by integrating out all latent variables:

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, \mathbf{z}_{1:T})\ \mathrm{d}\mathbf{z}_{1:T} \tag{1}$$

Using the first-order Markov chain with Gaussian transitions we can model the joint distribution as follows:

$$p_\theta(\mathbf{x}, \mathbf{z}_{1:T}) = p_\theta(\mathbf{x}|\mathbf{z}_1) \left(\prod_{i=1}^{T-1} p_\theta(\mathbf{z}_i|\mathbf{z}_{i+1}) \right) p_\theta(\mathbf{z}_T) \tag{2}$$

where $\mathbf{x} \in \mathbb{R}^D$ and $\mathbf{z}_i \in \mathbb{R}^D$ for $i = 1, \ldots, T$. Here, the latent and the observable variables have the same dimensions. Note that all the distributions are parameterized using deep neural networks (DNNs). Now we proceed to describe the forward diffusion process (FDP) and the reverse diffusion process (RDP).

2.1 Forward Diffusion Process

In this forward diffusion procedure $Q_\phi(\mathbf{z}_{1:T}|\mathbf{x})$, the structure of user-item interaction data (x) is degraded over time by applying specified noise schedule. Similar to hierarchical VAEs, we define a family of variational posteriors for the FDP in the following way:

$$Q_\phi(\mathbf{z}_{1:T}|\mathbf{x}) = q_\phi(\mathbf{z}_1|\mathbf{x}) \left(\prod_{i=2}^{T} q_\phi(\mathbf{z}_i|\mathbf{z}_{i-1}) \right) \tag{3}$$

The significant point is the definition of these distributions. Here, we formulate them as Gaussian diffusion process following the example of Sohl-Dickstein et al. [21] in this way:

$$q_\phi(\mathbf{z}_i|\mathbf{z}_{i-1}) = \mathcal{N}(\mathbf{z}_i|\sqrt{1-\beta_i}\mathbf{z}_{i-1}, \beta_i \mathbf{I}) \tag{4}$$

where $\mathbf{z}_0 = \mathbf{x}$. A single step of the diffusion, $q_\phi(\mathbf{z}_i|\mathbf{z}_{i-1})$, works in a relatively easy manner. Mainly, it utilizes the previously generated variable $\mathbf{z}_{i-1}$, scales it by $\sqrt{1-\beta_i}$ and then adds noise with variance β_i. Notably, we can use the reparameterization trick to define it as this:

$$\mathbf{z}_i = \sqrt{1-\beta_i}\mathbf{z}_{i-1} + \sqrt{\beta_i} \odot \epsilon, \tag{5}$$

where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$. In principle, β_i could be learned by backpropagation, however, as noted in previous research [10,21], it could be fixed. In our recommendation model, we find that this is a very sensitive hyperparameter that significantly affects model performance.

2.2 Reverse Diffusion Process

For the reverse denoising process ($P_\theta(\mathbf{x}_{0:T})$), the goal is to retrieve the original user-item interaction data ($\mathbf{x}$) from the noisy input. Note that the reverse procedure is also parameterized using first-order Markov chain with a learned Gaussian transition distribution as follows:

$$P_\theta(\mathbf{x}_{0:T}) = p_\theta(\mathbf{x}|\mathbf{z}_1)\left(\prod_{i=1}^{T-1} p_\theta(\mathbf{z}_i|\mathbf{z}_{i+1})\right) \tag{6}$$

$$p_\theta(\mathbf{z}_i|\mathbf{z}_{i+1}) = \mathcal{N}(\mu_\theta(\mathbf{z}_i,t), \Sigma_\theta(\mathbf{z}_i,t)) \tag{7}$$

2.3 Learning Procedure of CODIGEM

The learning objective is essentially the evidence lower bound (ELBO) [14]. We highlight that, the distribution $q_\phi(\mathbf{z}_1|\mathbf{x})$ will resemble an isotropic Gaussian given T and a well-behaved variance schedule of β_t. By choosing a latent variable from the isotropic Gaussian distribution and executing the reverse procedure (RDP), we can generate novel user-item interaction instances from the underlying data distribution. The RDP is trained to minimize the following upper bound over the negative log-likelihood. Basically, the learning objective (ELBO) of CODIGEM is derived as follows:

$$\ln p_\theta(\mathbf{x}) = \ln \int Q_\phi(\mathbf{z}_{1:T}|\mathbf{x})\frac{p_\theta(\mathbf{x},\mathbf{z}_{1:T})}{Q_\phi(\mathbf{z}_{1:T}|\mathbf{x})}\,\mathrm{d}\mathbf{z}_{1:T} \tag{8}$$

$$\geq \mathbb{E}_{Q_\phi(\mathbf{z}_{1:T}|\mathbf{x})}[\ln p_\theta(\mathbf{x}|\mathbf{z}_1) + \sum_{i=1}^{T-1}\ln p_\theta(\mathbf{z}_i|\mathbf{z}_{i+1}) + \ln p_\theta(\mathbf{z}_T) - \sum_{i=2}^{T}\ln q_\phi(\mathbf{z}_i|\mathbf{z}_{i-1}) - \ln q_\phi(\mathbf{z}_1|\mathbf{x})] \tag{9}$$

$$= \mathbb{E}_{Q_\phi(\mathbf{z}_{1:T}|\mathbf{x})}[\ln p_\theta(\mathbf{x}|\mathbf{z}_1) + \ln p_\theta(\mathbf{z}_1|\mathbf{z}_2) + \sum_{i=2}^{T-1}\ln p_\theta(\mathbf{z}_i|\mathbf{z}_{i+1}) + \ln p_\theta(\mathbf{z}_T) - \sum_{i=2}^{T-1}\ln q_\phi(\mathbf{z}_i|\mathbf{z}_{i-1}) - \ln q_\phi(\mathbf{z}_T|\mathbf{z}_{T-1}) - \ln q_\phi(\mathbf{z}_1|\mathbf{x})] \tag{10}$$

$$= \mathbb{E}_{Q_\phi(\mathbf{z}_{1:T}|\mathbf{x})}[\ln p_\theta(\mathbf{x}|\mathbf{z}_1) + \sum_{i=2}^{T-1}(\ln p_\theta(\mathbf{z}_i|\mathbf{z}_{i+1}) - \ln q_\phi(\mathbf{z}_i|\mathbf{z}_{i-1})) + \ln p_\theta(\mathbf{z}_T) - \ln q_\phi(\mathbf{z}_T|\mathbf{z}_{T-1}) + \ln p_\theta(\mathbf{z}_1|\mathbf{z}_2) - \ln q_\phi(\mathbf{z}_1|\mathbf{x})] \tag{11}$$

$$\overset{df}{=} \mathcal{L}(\mathbf{x};\theta,\phi) \tag{12}$$

Eventually, the ELBO is the following:

$$\mathcal{L}(\mathbf{x};\theta,\phi) = \mathbb{E}_{Q_\phi(\mathbf{z}_{1:T}|\mathbf{x})}[\ln p_\theta(\mathbf{x}|\mathbf{z}_1) + \sum_{i=2}^{T-1}\left(\ln p_\theta(\mathbf{z}_i|\mathbf{z}_{i+1}) - \ln q_\phi(\mathbf{z}_i|\mathbf{z}_{i-1})\right) + \ln p_\theta(\mathbf{z}_T) - \ln q_\phi(\mathbf{z}_T|\mathbf{z}_{T-1}) + \ln p_\theta(\mathbf{z}_1|\mathbf{z}_2) - \ln q_\phi(\mathbf{z}_1|\mathbf{x})] \tag{13}$$

Note that we use continuous distribution to model $p(\mathbf{x}|\mathbf{z}_1)$. Moreover, we normalize our inputs to values between -1 and 1, and apply the Gaussian distribution with the unit variance and the mean being constrained to $[-1, 1]$ using the Tanh non-linear activation function. Subsequently, we obtain the expression: $p(\mathbf{x}|\mathbf{z}_1) = \mathcal{N}(\mathbf{x}|\tanh(NN(\mathbf{z}_1)), \mathbf{I})$, where $NN(\mathbf{z}_1)$ is a deep neural network.

2.4 Recommendation Generation

Given a user's click history $\mathbf{x}$, we can sample a latent distribution from $p(\mathbf{z}_T)$- an isotropic Gaussian distribution and execute the reverse procedure (RDP) to generate scores of each item for this user. In a typical $top - N$ recommendation system, we take the $top - N$ values as the prediction items for this user.

3 Experiments

To validate our model and technical contributions we aim to answer these all-important questions:

RQ1: Can the denoising diffusion probabilistic model (DDPM) effectively model non-linear user-item interactions? If so, how can we extend it to CF for implicit feedback data to obtain competitive performance?
RQ2: Is DDPM helpful in generating valuable recommendations? If yes, how does it work, and what is the cost?
RQ3: How do key hyperparameters of DDPM affect model performance?
RQ4: How efficient is the proposed DDPM-based CF model?

3.1 Experimental Settings

Datasets: We conducted our empirical evaluations on three real-world and publicly available datasets: MovieLens-1m (ML-1m)[2], MovieLens-20m (ML-20m)[3], and Amazon Electronics (AE)[4]. We adopt standard practise of data-preprocessing for implicit feedback-based recommendations. Specifically, for all these datasets, we used a rating value three (3) and above. We also only kept users with at least ten (10) interactions, and we used items which had at least ten (10) interactions. We additionally converted all scores to 1 because we are focusing on the implicit feedback setting. The dataset statistics is depicted in Table 1.

Baseline Models: We compare CODIGEM to the following baseline models to validate its performance:

Pop: This model considers the most popular items in a dataset and recommends these items to users.

[2] https://grouplens.org/datasets/movielens/1m/.
[3] https://grouplens.org/datasets/movielens/20m/.
[4] http://jmcauley.ucsd.edu/data/amazon/.

Table 1. Statistics of all datasets after pre-processing

Dataset	ML-20m	ML-1m	AE
Number of users	136677	6034	13456
Number of items	20108	3124	8361
Number of item interactions	9990682	834449	234521
Sparcity percentage	99.64%	95.57%	99.79%

Item-KNN [5]**:** A type of model-based recommendation algorithm that identifies the collection of items to be recommended by first determining the similarities between the various items.

BPR [19]**:** BPR is a Bayesian-based framework that presents a generic optimization approach for personalized ranking.

Weighted matrix factorization (WMF) [12]**:** WMF is a low-rank factorization algorithm with a linear structure.

ENMF [1] ENMF is a well-optimized neural recommendation model that employs whole training data without sampling.

NeuMF [8]**:** NeuMF generalizes MF for CF using a neural network to overcome the constraint of linear interaction in MF.

Multi-VAE [17]**:** This is a representative VAE-based CF model. Its objective function incorporates multinomial likelihood, which is a distinguishing feature.

MacridVAE [18]**:** MacridVAE is a model that is used for learning disentangled representations from user behavior.

Evaluation Metrics: We employ two standard recommender system metrics such as Recall@R (R@R) and the normalized discounted cumulative gain (NDCG@R). We contrast the predicted rank of the held-out items to the actual rank using the Recall@R and NDCG@R metrics. By sorting the unnormalized probability, we get the predicted rank. NDCG@R employs a monotonically increasing discount to signal the relevance of higher rankings over lower ones, whereas Recall@R considers all items ranked inside the first R to be equally relevant. Notably, we calculate the Recall and NDCG at rank positions 20 and 50.

Implementation Details: We implement CODIGEM with Pytorch. We divide each dataset into training, validation, and test sets using the ratio 8:1:1. We use Xavier initialization. We utilize a learning rate of 0.001 and train the model with the Adamax optimizer [13]. Hyper-parameters β_t and T are set to 0.0001 and 3, respectively. We use an architecture comprising six (6) layers of MLP with PReLU non-linear activation function in between the layers to parametrize the distributions in FDP and RDP components of CODIGEM. Note that we use the Tanh activation function in the last layer of the RDP. During training we use a batch size of 200 and set up our model to train for 100 epochs. Also, we adopt early stopping when model performance does not increase for ten (10) successive epochs.

3.2 Results and Analysis

RQ 1: Performance Evaluation: Table 2 depicts the performance of CODIGEM in comparison to representative classical and DNN-based recommendation models on the Movielens-1m (ML-1m), Movielens-20m (ML-20m), and Amazon Electronics (AE) datasets. For all the results, a paired t-test is conducted. Here, $p < 0.001$ indicates statistical significance. After careful analysis, we observe the following: Interestingly, BPR and Item-KNN models generally outperform DNN-based models such as ENMF and NeuMF, highlighting the intrinsic robustness of some traditional models. However, Linear models such as Pop and WMF are not competitive with respect to all the DNN-based methods (ENMF, NeuMF, MacridVAE, Multi-VAE, and CODIGEM) because of their inability to effectively capture complex non-linear patterns from user-item interactions data which can aid in the recommendation task. This trend demonstrates the importance of learning the non-linear information from user-item interactions data. Generally, deep generative models such as (MacridVAE, Multi-VAE, and CODIGEM) show strong performance over their non-generative counterparts. The possible reason for this trend is that generative models can effectively capture the intricate and non-linear patterns from the unobserved user-item interactions to obtain strong collaborative signals for the recommendation task. Moreover, these DGM-based models can obtain high-quality collaborative information from the underlying implicit feedback data, thereby generalizing better and producing excellent recommendations than the non-generative models. Multi-VAE, an earlier proposed DGM-based CF model, outperforms all classical and DNN-based models. Nevertheless, we point out that CODIGEM is yet another competitive DGM-based CF model worthy of research attention. Mainly, we see that CODIGEM outperforms robust classical models (BPR, Item-KNN, WMF, and Pop) and some DNN-based models on all the metrics for all the datasets. Moreover, CODIGEM is more computationally efficient than Multi-VAE.

Table 2. Comparison of performance of baseline models and CODIGEM on ML-1M, ML-20M and AE datasets

Models	ML-1M				ML-20M				AE			
	R@20	R@50	N@20	N@50	R@20	R@50	N@20	N@50	R@20	R@50	N@20	N@50
Pop	0.1289	0.2202	0.1218	0.1460	0.1463	0.2473	0.1138	0.1422	0.0480	0.0899	0.0209	0.0296
BPR	0.2647	0.4252	0.2460	0.2919	0.2955	0.4596	0.2478	0.2964	0.0732	0.1254	0.0325	0.0436
WMF	0.2390	0.3994	0.2206	0.2675	0.2833	0.4254	0.2259	0.2904	0.0655	0.1118	0.0281	0.0378
ENMF	0.2582	0.4103	0.2467	0.2876	0.2234	0.3398	0.1941	0.2561	0.0564	0.0995	0.0263	0.0354
NeuMF	0.2403	0.4009	0.2219	0.2689	0.3021	0.4563	0.2469	0.2965	0.0702	0.1219	0.0307	0.0417
Item-KNN	0.2539	0.4039	0.2430	0.2829	0.2843	0.4264	0.2469	0.2909	0.0588	0.0917	0.0316	0.0389
Multi-VAE	0.2925	0.4588	0.2574	0.3081	0.3698	0.5348	0.3105	0.3508	0.0774	0.1340	0.0364	0.0484
MacridVAE	0.2389	0.3854	0.2371	0.2759	0.2996	0.4358	0.2704	0.3005	0.0769	0.1320	0.0353	0.0465
CODIGEM	**0.2796**	**0.4354**	**0.2447**	**0.3026**	**0.3512**	**0.4698**	**0.3031**	**0.3414**	**0.0771**	**0.1338**	**0.0360**	**0.0478**

RQ 2: Study of CODIGEM: CODIGEM is intrinsically a hierarchical latent variable model, and like VAE, they also utilize a family of variational posteriors.

In CODIGEM, we define these posterior distributions as a Gaussian diffusion process. Since CODIGEM and VAE-based CF models belong to the same family, we incorporate some well-known and proven VAE techniques into CODIGEM and study its impacts. Notably, we create these variants: CODIGEM-I (CODIGEM with annealing [17]), CODIGEM-II (CODIGEM with skip connections [7] in the FDP), CODIGEM-III (CODIGEM with skip connections in the RDP), CODIGEM-IV (CODIGEM with multinomial likelihood [17]). The empirical results are depicted in Fig. 3. From these results, we observe that annealing does not negatively affect model performance. We witness a slight decline in CODIGEM's performance when skip-connections are used. Additionally, employing the multinomial likelihood worsens CODIGEM's performance.

Table 3. Study of the impact of diverse techniques on CODIGEM

Model variants	Model performance	
	R@50	N@50
CODIGEM-I	0.4696	0.3411
CODIGEM-II	0.4627	0.3359
CODIGEM-III	0.4602	0.3311
CODIGEM-IV	0.2563	0.1353
CODIGEM	0.4698	0.3414

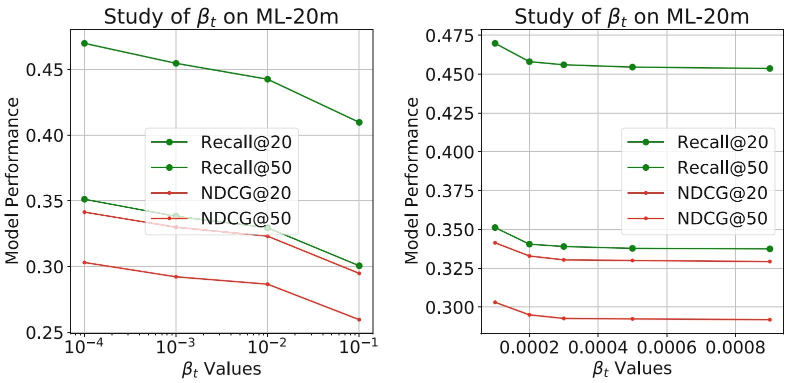

Fig. 2. The impact of β_t on CODIGEM on the ML-20m dataset. The left subfigure depicts a sharp decline in performance in the range 0.0001 to 0.1 and the right subfigure depicts a slight decline in performance in the range 0.0001 to 0.0009.

RQ 3: Parameter Sensitivity: Here, we study the impact of the critical hyper-parameters such as β_t and T on the overall performance of CODIGEM. For β_t, we experiment with several values in the range of 0.0001 to 0.9. The best results

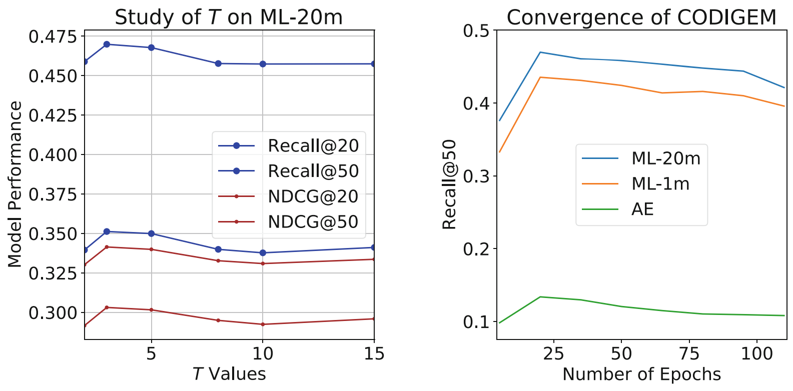

Fig. 3. The left subfigure shows the study of the impact of T on CODIGEM and the right subfigure depicts convergence of CODIGEM for all the datasets

across datasets are 0.0001. As depicted in Fig. 2, a change in the β_t value drastically affects model performance. Hence, a careful tuning of β_t on your dataset is always required. Regarding T, we observe that unlike DDPM models used in image synthesis, increasing the value of T does not improve model performance. From the left subfigure of Fig. 3, we consistently obtain the best model performance when T is 3.

RQ 4: Computational Efficiency: T is a hyperparameter that significantly impacts the computational efficiency of DDPM models-a large T value results in a computationally inefficient model (see Table 4). As indicated earlier (see the left subfigure of Fig. 3), we observed during the empirical studies of DDPM for CF that setting $T = 3$ yields optimal performance. To further ascertain the efficiency of CODIGEM, we study its training efficiency. From the convergence graphs of CODIGEM on all three datasets (see the right subfigure of Fig. 3), we notice that the CODIGEM model converges mainly around 15 to 25 epochs. On the other hand, MacridVAE and Multi-VAE converge to good performance after 75 to 200 epochs. Additionally, in Table 5 we report the training and evaluation convergence time for all the generative models. Overall, considering the rapid convergence and relatively less training time of CODIGEM, we can conclude that CODIGEM is more computationally efficient than the baseline VAE-models. We remark that our proposed CODIGEM model's computational efficiency is a significant advantage.

Table 4. Impact of T value on computational efficiency of CODIGEM

T value	Total model execution time	Number of epochs
3	824.682 s	18
5	1212.060 s	18
8	1861.488 s	18
10	2416.659 s	19
15	3384.233 s	18

Table 5. Comparison of computational efficiency of the generative models

Model	Training time (sec/epoch)	Evaluation time (sec/epoch)	Average number of epochs to convergence	Total time of convergence
MacridVAE	142.35	124.81	95	25380.20 s
Multi-VAE	53.17	48.45	150	15243.00 s
CODIGEM	48.22	46.67	18	1708.02 s

4 Concluding Remarks

Paper Conclusions: This paper presented the *first-ever* DDPM-based RS model that effectively models non-linear user-item interactions to generate strong collaborative signals for enhancing the generalizability of recommendations. We also demonstrated through systematic experimental validation the settings that enable CODIGEM to produce excellent recommendations performance. Moreover, the empirical studies indicate that CODIGEM is very efficient and outclasses several classical RS models on three real-world datasets. Overall, our findings highlight the significance of using the diffusion probabilistic model in recommendation systems. Our studies conclude that DDPM is a viable DGM alternative for RS tasks that needs more research attention.

Open Research Directions: Here, we highlight some unexplored potential of DDPM-based DGMs that can significantly enhance recommendation systems. **Firstly**, a crucial problem of VAE-based CF models is variational posterior collapsing to the prior, resulting in meaningless representation learning. DDPMs are robust against posterior collapse issues and can mitigate this limitation. Hence, integrating DDPMs and VAEs would be exciting research as both can take advantage of each other to generate excellent recommendations. **Secondly**, simplistic prior in VAE-based CF models results in sub-optimal recommendation performance. An interesting new direction would be to use DDPMs as flexible priors in VAEs. Moreover, we can incorporate important side information such as visual content-based information into these priors, ultimately addressing data sparsity and cold-start issues. **Lastly**, we can enhance the stability and performance of DDPMs by learning the covariance matrices in reverse diffusion and using different noise schedules.

Acknowledgements. This work was supported by National Natural Science Foundation of China (Grant No. 62176043, No. 62072077 and No. 62102326) and the Key Research and Development Project of Sichuan Province (Grant No. 2022YFG0314).

References

1. Chen, C., Zhang, M.I.N.: Efficient neural matrix factorization without sampling. ACM Trans. Inf. Syst. **38**(2), 1–28 (2020)
2. Chen, J., Zhao, C., Uliji, Chen, L.: Collaborative filtering recommendation algorithm based on user correlation and evolutionary clustering. Complex Intell. Syst. **6**(1), 147–156 (2020). https://doi.org/10.1007/s40747-019-00123-5
3. Chen, S., Peng, Y.: Matrix factorization for recommendation with explicit and implicit feedback. Knowl. Based Syst. (2018). https://doi.org/10.1016/j.knosys.2018.05.040
4. Dacrema, M.F., Cremonesi, P., Jannach, D.: Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In: Proceedings of the 13th ACM Conference on Recommender Systems, pp. 101–109. RecSys 2019, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3298689.3347058
5. Deshpande, M., Karypis, G.: Item-based top- N recommendation algorithms. ACM Trans. Inf. Syst. (2004). https://doi.org/10.1145/963770.963776
6. Dhariwal, P., Nichol, A.: Diffusion models beat GANs on image synthesis. CoRR abs/2105.05233 (2021)
7. Dieng, A.B., Kim, Y., Rush, A.M., Blei, D.M.: Avoiding latent variable collapse with generative skip models. In: AISTATS Proceedings of Machine Learning Research, vol. 89, pp. 2397–2405, PMLR (2019)
8. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: Proceedings of the 26th International Conference on World Wide Web - WWW 2017, pp. 173–182 (2017). DOI: https://doi.org/10.1145/3038912.3052569
9. He, X., Zhang, H., Kan, M.Y., Chua, T.S.: Fast matrix factorization for online recommendation with implicit feedback. In: SIGIR 2016 - Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 549–558 (2016). https://doi.org/10.1145/2911451.2911489
10. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: NeurIPS (2020)
11. Ho, J., Saharia, C., Chan, W., Fleet, D.J., Norouzi, M., Salimans, T.: Cascaded diffusion models for high fidelity image generation. J. Mach. Learn. Res. **23**, 47:1–47:33 (2022)
12. Hu, Y., Volinsky, C., Koren, Y.: Collaborative filtering for implicit feedback datasets. In: Proceedings - IEEE International Conference on Data Mining, ICDM, pp. 263–272. IEEE (2008). https://doi.org/10.1109/ICDM.2008.22
13. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: ICLR (Poster) (2015)

14. Kingma, D.P., Welling, M.: Auto-encoding variational Bayes. In: 2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings, pp. 1–14 (2014)
15. Koren, Y., Bell, R.: Advances in collaborative filtering. In: Recommender Systems Handbook, Second Edn., pp. 77–118. Springer, Boston, MA (2015). https://doi.org/10.1007/978-1-4899-7637-6-3
16. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009). https://doi.org/10.1109/MC.2009.263
17. Liang, D., Krishnan, R.G., Hoffman, M.D., Jebara, T.: Variational autoencoders for collaborative filtering Dawen. WWW (2018). https://doi.org/10.1145/3178876.3186150
18. Ma, J., Zhou, C., Cui, P., Yang, H., Zhu, W.: Learning disentangled representations for recommendation. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 32. Curran Associates, Inc. (2019). https://proceedings.neurips.cc/paper/2019/file/a2186aa7c086b46ad4e8bf81e2a3a19b-Paper.pdf
19. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: Bayesian personalized ranking from implicit feedback. In: UAI 2012, pp. 452–461 (2012). http://arxiv.org/abs/1205.2618
20. Salakhutdinov, R., Mnih, A.: Probabilistic matrix factorization. In: Advances in Neural Information Processing Systems, vol. 20, pp. 1257–1264 (2007). https://doi.org/10.1145/1390156.1390267
21. Sohl-Dickstein, J., Weiss, E.A., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. In: ICML. JMLR Workshop and Conference Proceedings, vol. 37, pp. 2256–2265. JMLR.org (2015)
22. Zhang, Q., Lu, J., Jin, Y.: Artificial intelligence in recommender systems. Complex Intell. Syst. **7**(1), 439–457 (2021)
23. Zhang, S., Yao, L., Sun, A., Tay, Y.: Deep learning based recommender system: a survey and new perspectives. ACM Comput. Surv. (CSUR) **52**(1), 1–35 (2019). http://arxiv.org/abs/1707.07435