Flow Matching for Denoised Social Recommendation

Yinxuan Huang [1*] Ke Liang [1*] Zhuofan Dong [2] Xiaodong Qu [3] Tianxiang Wang [1] Yue Han [1] Jingao Xu [1] Bin Zhou [1] Ye Wang [1]

Abstract

Graph-based social recommendation (SR) models suffer from various noises of the social graphs, hindering their recommendation performances. Either graph-level redundancy or graph-level missing will indeed influence the social graph structures, further influencing the message propagation procedure of graph neural networks (GNNs). Generative models, especially diffusion-based models, are usually used to reconstruct and recover the data in better quality from original data with noises. Motivated by it, a few works take attempts on it for social recommendation. However, they can only handle isotropic Gaussian noises but fail to leverage the anisotropic ones. Meanwhile the anisotropic relational structures in social graphs are commonly seen, so that existing models cannot sufficiently utilize the graph structures, which constraints the capacity of noise removal and recommendation performances. Compared to the diffusion strategy, the flow matching strategy shows better ability to handle the data with anisotropic noises since they can better preserve the data structures during the learning procedure. Inspired by it, we propose RecFlow which is the first flow-based SR model. Concretely, RecFlow performs flow-based method on the structure representations of social graphs. Then, a conditional learning procedure is designed for optimization. Extensive performances prove the promising performances of our RecFlow from six aspects, including superiority, effectiveness, robustnesses, sensitivity, convergence and visualization. Code are available at https://github.com/AfraThroneUp/RecFlow.

[*]Equal contribution [1]School of Computer Science, National University of Defense Technology, Changsha, Hunan, China [2]University of Chicago, Chicago, Illinois, United States [3]Harbin Institute of Technology(Shenzhen), Shenzhen, Guangdong, China. Correspondence to: Ye Wang <Yewang@nudt.edu.cn>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

1. Introduction

As online content continues to grow exponentially, the challenges of managing information sensitivity and urgency have become increasingly pressing (Lin et al., 2025; Wang et al., 2024), driving the rise of recommendation systems (Liu et al., 2023b). Despite advancements, recommendation systems still face challenges such as collaborative information sparsity. The rise of social media has shifted their focus from user-item interactions to integrating social networks to enhance recommendations. By leveraging social relationships as auxiliary information, social recommendation (SR) systems can mitigate these issues, making SR a key research area (Liang et al., 2023). Early models of SR relied on matrix factorization (Salakhutdinov & Mnih, 2007; Yang et al., 2013), which drew upon social theories to exploit the influence of nearby or connected users on individual preferences. Additionally, social relationships naturally form graph-structured data, making graph neural networks (GNNs) an effective tool for graph representation learning (Wang et al., 2023a; Dai et al., 2023). GNNs have demonstrated exceptional performance in aggregating neighborhood information of nodes and have been widely applied in the social recommendation domain, enabling the deep exploration and effective utilization of more valuable information (Huang et al., 2021a; Liang et al., 2023).

Despite advancements in social recommendation systems, current approaches often struggle to effectively mitigate two critical graph structural issues: redundancy and incompleteness (Lin et al., 2023). These limitations primarily stem from noisy social connections in real-world data, where low-quality relationships inject interference that significantly degrades recommendation performance (Lin et al., 2024c;a). Graph neural networks (GNNs) exhibit particular vulnerability to such noise due to their inherent reliance on message propagation mechanisms across social edges, a characteristic that amplifies error transmission through the network (Wang et al., 2023a; Lin et al., 2024b). Recent studies have extended diffusion models to graph-based recommendation systems. DiffRec (Wang et al., 2023b) applies continuous diffusion by adding Gaussian noise to user/item embeddings and optimizes them through a denoising process. RecDiff (Li et al., 2024b) introduces a multi-step diffusion and denoising framework for modeling complex social con-

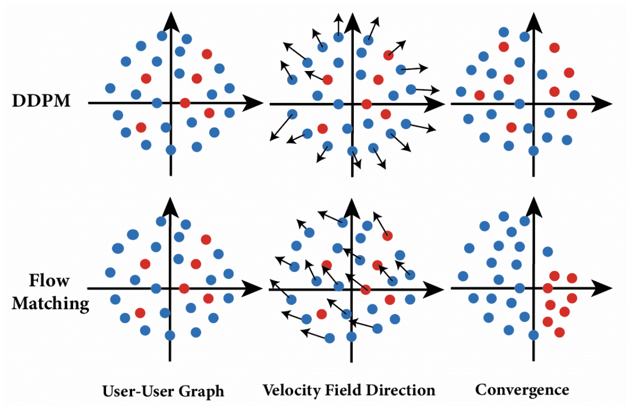

Figure 1. Illustration of denoising diffusion probabilistic models (DDPM) and flow matching based models where blue and red nodes represent normal and noisy data. Compared to DDPM, flow-based models can get better discriminative capacity, further leading to better denoise performance.

nections. These efforts build upon the growing success of diffusion models in various domains (Liu et al., 2024; Lee et al., 2024; Jiang et al., 2024; Deng et al., 2024).

Social data often exhibit strong anisotropy, as shown by the directional distribution of vector fields in our preliminary analysis (Figure 4). This conflicts with the **isotropic Gaussian noise** (where noise is modeled as $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$) assumption in conventional denoising diffusion probabilistic models (DDPM), leading to two key issues: representation degradation, as isotropic noise blurs user embedding distinctiveness, and unstable training, as DDPM's iterative denoising undermines convergence consistency (Kingma et al., 2023). To address these limitations, we adopt Flow Matching, a generative approach that learns continuous velocity fields to construct direct sampling paths and model an ODE flow, enabling it to handle **non-isotropic noise**(modeled by $\epsilon \sim \mathcal{N}(\mu, \Sigma)$ with $\mu \neq 0$ or non-diagonal Σ) (Zhao et al., 2024; Lipman et al., 2022). Figure 1 illustrates the differences between these generative approaches: DDPM injects isotropic noise without directional awareness, while Flow Matching guides data towards clean distributions in a more discriminative manner, improving denoising performance on graph-structured data (Lipman et al., 2022). Building on this, we propose RecFlow, a flow-based social recommendation model that leverages Flow Matching to capture the directional dynamics of user interactions, enabling more stable training, better denoising, and improved recommendation accuracy. Building on this, we propose **RecFlow**, a flow-based social recommendation model that leverages Flow Matching to capture the directional dynamics of user interactions, enabling more stable training, better denoising, and improved recommendation accuracy.

In summary, we makes the following contributions:

- We propose RecFlow, a novel social recommendation model that explicitly captures anisotropy in social networks through velocity fields, addressing the limitations of isotropic noise assumptions in conventional diffusion models.
- By integrating flow matching with social recommendation, RecFlow models directional data dynamics more effectively, leading to improved user preference representations.
- Extensive experiments on benchmark datasets demonstrate the effectiveness of RecFlow, with significant performance gains. A visualization experiment further visualizes the evolution of velocity fields over time, highlighting the impact of our approach.

2. Related Work

In this section, we provide a comprehensive review of related studies in the areas of social recommendation and generative models, and clarify how our work aligns with and builds upon the existing research.

2.1. Graph-based Social Recommendation

Graph-based Social Recommendation has gained significant attention for incorporating social relationships. Early works like DiffNet(Wu et al., 2019b) used Graph Convolutional Networks (GCNs) to model social influence, while later models like GraphRec(Fan et al., 2019) and DANSER(Wu et al., 2019c) added attention mechanisms to account for varying influence levels. More recent approaches, such as MHCN(Yu et al., 2021b) and HOSR(Liu et al., 2020), capture higher-order relationships and distant influences. Models like RecoGCN(Xu et al., 2019), DGRec(Song et al., 2019b), TGRec(Bai et al., 2020), and KCGN(Huang et al., 2021b) integrate diverse data sources, including agent, temporal, and knowledge graph information. To address noisy social relations, recent methods like DSL(Wang et al., 2023a) and GDMSR(Quan et al., 2023) focus on denoising by identifying and removing irrelevant or redundant social connections, thus improving recommendation quality and efficiency. These methods struggle with noisy or irrelevant social connections. Flow matching models, by modeling influence spread and denoising, provide a promising solution, refining user representations to enhance the robustness and accuracy of recommendations.

2.2. Diffusion Model-based Recommenders

Generative recommenders have attracted interest in recent studies. Some studies focused on leveraging diffusion models to enhance data representation and mitigate noise inherent in social connections. For instance, RecDiff(Li et al., 2024a) employed a latent diffusion paradigm to denoise user representations derived from social networks, demonstrating improved robustness in handling the diverse noisy effects of user social contexts. Similarly, DiffuASR(Liu et al., 2023a) proposed a diffusion-based pseudo sequence generation framework, and fills in the gap between the generations of continuous images and discrete sequences. CGSoRec(He et al., 2024) proposed a condition-guided social recommendation model, leveraging a conditional constraint in the diffusion process to incorporate social connections. This allows the model to refine user preferences based on their social connections. Similarly, DIEXRS(Guo et al., 2023) uses a diffusion framework to model user preferences, and then trains a textual decoder to generate explanations based on the denoised user representation, enhancing the interpretability of diffusion recommenders. In contrast to these established approaches, our proposed Recflow introduces a novel methodology by leveraging flow matching models (Liu et al., 2022), making it more effective in capturing intricate patterns.

3. Preliminary

In this section, we briefly introduce the preliminaries of flow-matching models. A flow-matching model is a type of generative model that bridges the gap between a source distribution p_x and a target distribution p_z by learning a neural network to parameterize the velocity field of an Ordinary Differential Equation (ODE). The ODE is defined as:

$$dx_t = v_\theta(x_t, t)dt, \tag{1}$$

where v_θ denotes the learnable velocity field parameterized by a neural network. The ODE ensures that the intermediate distributions x_t remain consistent with the learned probability path for all $t \in [0, 1]$, and the velocity field v_θ directs the flow from the initial state x to the target state z, effectively transforming the source distribution into the target distribution over time.

Forward Process: This process converts samples from $x \sim p_x$ to align with p_z. The interpolation between x and z is defined through the linear blend $x_t = tz + (1 - t)x$, satisfying the ODE:

$$dx_t = (z - x)dt \tag{2}$$

Reverse Process: Conversely, this process generates samples starting from $z \sim p_z$ and reverses the flow dynamics. The reverse ODE, mirroring the forward process, is defined as:

$$dx_t = (x - z)dt \tag{3}$$

The effectiveness of the rectified flow depends on the precise estimation of velocity v. To align v with the direction $(z - x)$, the model solves a least squares regression problem, optimizing the velocity field v_θ to closely match the ideal flow between x and z.

The training of the neural network involves minimizing the loss function L, defined as:

$$\mathcal{L}_2 = \int_0^1 \mathbb{E}_{x,z}\left[\|(z - x) - v_\theta(x_t, t)\|^2\right] dt \tag{4}$$

This loss quantifies the discrepancy between the ideal and predicted velocities over the time interval $[0, 1]$, enabling the flow to follow the desired trajectory by accurately predicting the velocity at any point t. The parameterized neural network v_θ is thus trained to minimize L, facilitating an efficient and accurate modeling of transitions from x to z.

4. Proposed Model

As illustrated in Figure 2, we integrated collaborative and social signals within a unified flow-based generative framework, the overall architecture of RecFlow consists of three key components: Graph-based Collaborative Pattern Encoding, the RecFlow Module, and a Joint Optimization Module.

4.1. Problem Setting

Users and items are defined as the sets $U = \{u_1, u_2, \ldots, u_n\}$ and $V = \{v_1, v_2, \ldots, v_n\}$, respectively. Interactions between users and items are represented by the matrix $R \in \mathbb{R}^{|U|\times|V|}$, where the element $r_{u,v} = 1$ indicates that user u interacts with item v, and $r_{u,v} = 0$ otherwise. Social relationships between users are described by the matrix $S \in \mathbb{R}^{|U|\times|U|}$, where $s_{u,u'} = 1$ signifies a social interaction between user u and user u', and $s_{u,u'} = 0$ otherwise. Based on these interaction matrices, the following graph structures are constructed:

- **Collaborative Graph** $G_r = (U, V, E_r)$, where the edge set $E_r = \{\, (u, v) \mid r_{u,v} = 1 \,\}$ represents interactions between users and items.
- **Social Graph** $G_s = (U, E_s)$, where the edge set $E_s = \{\, (u, u') \mid s_{u,u'} = 1 \,\}$ represents social relationships between users.

Our RecFlow leverages both collaborative and social graphs specifically, the collaborative graph G_r generates node embeddings denoted as E_r and the social graph G_s generates node embeddings denoted as E_s. The predicted user-item interaction value $\hat{r}_{u,v}$ is computed as:

$$\hat{r}_{u,v} = \text{Pred}(e_u, e_v), \tag{5}$$

where e_u and e_v are the embeddings of the user u and item v, respectively.

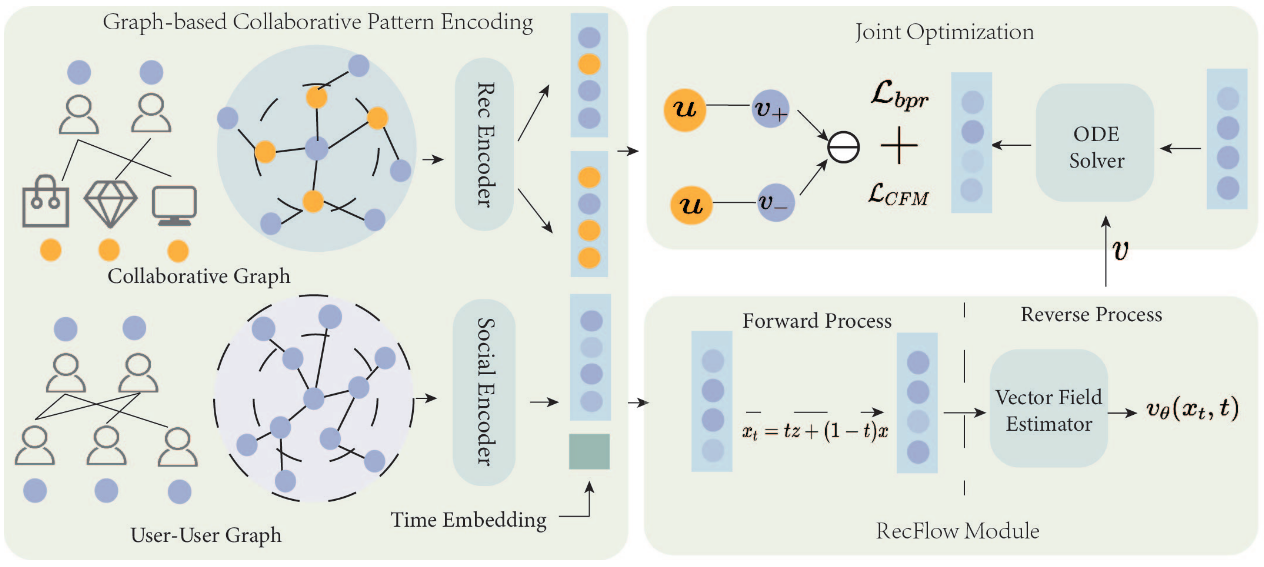

Figure 2. The overall framework of our RecFlow.

Algorithm 1 RecFlow Training

Input: Users' social interaction embedding E^s
Output: The reconstructed embedding which is denoted as $\hat{e}_0$

```
1: while not converged do
2:     $t \sim \mathcal{U}(0,1)$        # Sample time
3:     $x \sim p_x$        # Sample data
4:     $z \sim p_z$        # Sample noise
5:     $x_t = \Psi_t(z|x)$        # Conditional flow
6:     Gradient step: $\nabla_\theta \| v_t^\theta(x_t) - \hat{x}_t \|^2$
7: end while
```

Algorithm 2 RecFlow Inference

Input: Users' interaction vectors x_u, $u = 1, 2, \ldots, |U|$; optimized parameter θ
Output: Predicted user embeddings or interaction outcomes

```
1: for $u \in U$ do
2:     Let $x_0 \leftarrow x_u$        # Initialize with user data
3:     Sample $t \sim p_t$        # Time step sampling
4:     Calculate $x_t \leftarrow f_\theta(x_0, t)$        # Run learned model
5:     if needsPostProcessing then
6:         $\hat{y} \leftarrow \text{Process}(x_t)$        # Final prediction
7:     end if
8: end for
```

4.2. Graph-based Collaborative Pattern Encoding

Drawing inspiration from the effectiveness of simplified Graph Neural Networks (GNNs), we incorporated a lightweight Graph Convolutional Network (lightGCN) as the graph encoder in our RecFlow architecture (Jiang et al., 2023). LightGCN is widely recognized as a robust graph recommender for modeling implicit interactions in top-k recommendations. we construct a collaborative graph to encode user-item interactions using the Rec Encoder, and a user-user graph to capture social relationships among users using the Social Encoder. On the user-item graph G_r, the embeddings are propagated across layers using the following equation:

$$E_r^{(l)} = (L_r + I) \cdot E_r^{(l-1)}, \tag{6}$$

The computation of L_r is given by:

$$L_r = D_r^{-\frac{1}{2}} A_r D_r^{-\frac{1}{2}}, \tag{7}$$

where $A_r \in \mathbb{R}^{(|\mathcal{U}|+|\mathcal{V}|)\times(|\mathcal{U}|+|\mathcal{V}|)}$ is the adjacency matrix of the bipartite graph G_r, the embedding matrix $E_r^{(l)} \in \mathbb{R}^{(|\mathcal{U}|+|\mathcal{V}|)\times d}$ captured the embeddings in the l-th iteration of the GCNs. The initial embeddings E_0^r are randomly generated learnable parameters. And D_r is the corresponding diagonal degree matrix, defined as:

$$A_r = \begin{bmatrix} 0 & R \\ R^\top & 0 \end{bmatrix}. \tag{8}$$

where L_r is the normalized Laplacian matrix of G_r, and I is the identity matrix. For the user's social graph G_s, the embedding propagation follows a similar process:

$$E_s^{(l)} = (L_s + I) \cdot E_s^{(l-1)}, \tag{9}$$

where L_s is the normalized Laplacian matrix, computed as:

$$L_s = D_s^{-\frac{1}{2}} S D_s^{-\frac{1}{2}}, \tag{10}$$

S being the adjacency matrix of the social graph G_s. After propagating through L layers, the final embedding for each user-item pair is obtained by aggregating embeddings across all layers as follows:

$$\hat{e}_{u,u'} = \frac{1}{L+1} \sum_{l=0}^{L} e_{u,u'}^{(l)}, \tag{11}$$

where $e_{u,u'}^{(l)}$ represents the l-th layer embedding between user u and u'.

Similarly, for a user-item pair (u, v), the predicted embedding is:

$$\hat{e}_{u,v} = \frac{1}{L+1} \sum_{l=0}^{L} e_{u,v}^{(l)}. \tag{12}$$

4.3. RecFlow Module

In the forward process, the RecFlow module begins by sampling a time step t from a uniform distribution $\mathcal{U}(0, 1)$. This time step is then encoded into a time embedding vector E_t, which is concatenated with the user representation E_s obtained from the social encoder. The perturbed input x_t is defined as a linear interpolation between Gaussian noise z and the socially-informed embedding E_s:

$$x_t = tz + (1-t)E_s \tag{13}$$

Unlike conventional flow-matching methods that typically treat z as the origin and interpolate toward data samples, our formulation conditions the diffusion trajectory on the social embedding E_s, enabling the model to incorporate social context into the forward process.

In the reverse process, we train a vector field estimator $v_\theta(x_t, t)$ to approximate the velocity field. Here, θ denotes the set of learnable parameters. The estimator is defined as:

$$v_\theta(x_t, t) = \mathrm{FC}^2(E_s \,\|\, E_t), \quad \mathrm{FC}(x) = \sigma(Wx + b) \tag{14}$$

where E_t is the time embedding at step t, $\|$ denotes vector concatenation, and FC^2 represents two consecutive fully connected layers. The function $\sigma(\cdot)$ denotes a non-linear activation (e.g., ReLU or GELU), and W, b are the weight matrix and bias vector of each linear transformation.

Specifically, starting from x_t, the model integrates the reverse-time flow using an ODE solver to obtain the clean embedding x_0, which approximates the original user preference vector in the latent space. The reverse integration is performed as:

$$x_0 = x_t + \int_t^0 v_\theta(x_\tau, \tau)\, d\tau \tag{15}$$

During inference, the model fixes the optimized parameters θ, and no further training is performed, then we sample a latent representation $z \sim \mathcal{N}(0, I)$ and timestep t and then applies the learned v to reconstruct the user embedding e_u.

4.4. Joint Optimization

To integrate social relationships with encoded user-item interaction patterns, RecFlow employs a hidden-space reflow mechanism to generate the final user embeddings for prediction. This process is defined as:

$$\hat{r}_{u,v} = \tilde{e}_u^\top e'_v, \quad \tilde{e}_u = e'_u + \hat{e}_\theta(e'_u, t), \tag{16}$$

where t denotes a sampled diffusion time step for user u, e'_u and e'_v represent the initial embeddings of the user and item obtained from the respective graph encoders, and $\hat{e}_\theta(\cdot, t)$ denotes the learned reflow adjustment from the vector field estimator.

The model is optimized by minimizing a joint loss function that combines recommendation and diffusion objectives:

$$\mathcal{L} = \sum_{(u,v^+,v^-)} -\log \sigma(\hat{r}_{u,v^+} - \hat{r}_{u,v^-}) + \lambda_1 \sum_t \mathcal{L}_{\text{fm}} \tag{17}$$

Here, (u, v^+, v^-) denotes a user with a positive and a negative item in a pairwise training setup following the Bayesian Personalized Ranking (BPR) paradigm (Rendle et al., 2012). The flow-matching loss $\mathcal{L}_{\text{fm}}$, computed over sampled diffusion steps t, guides the learning of the reverse-time vector field. Additionally, L2 regularization (weight decay) with coefficient λ_1 is applied to all trainable parameters Θ to prevent overfitting. The form of $\mathcal{L}_{\text{CFM}}$ follows the definition in Eq. (3) and the flow estimation process illustrated in Figure 2.

4.5. Disucssion and Analysis

In this section, we present a detailed analysis of the time and space complexity of our RecFlow model. Besides, we further provide the theoretical analysis between our RecFlow and original DDPM methods to better illustrate the efficiency of the flow matching strategy.

Time Complexity. Initially, RecFlow performs graph-level information propagation on both the holistic collaborative graph $\mathcal{G}_r$ and the social graph $\mathcal{G}_s$, this process requires $\mathcal{O}((|\mathcal{E}_r| + |\mathcal{E}_s|) \times d)$ calculations for message passing. The overall time complexity of RecFlow during training is dominated by the graph-level propagation and the gradient updates, resulting in:

$$\mathcal{O}((|\mathcal{E}_r| + |\mathcal{E}_s|) \times d) \tag{18}$$

for each iteration of training. And complexity of flow matching is O(N), eliminating multi-step iterations and requiring only one global optimization.

Space Complexity. The space complexity is primarily determined by the storage required for the graphs and embeddings. The collaborative graph $\mathcal{G}_r$ requires storing the adjacency matrix of the user-item interactions, which has a space complexity of $\mathcal{O}(|U| \times |V|)$, the social graph $\mathcal{G}_s$ requires storing the adjacency matrix of the user-user interactions, which has a space complexity of $\mathcal{O}(|U|^2)$, assuming a dense representation. Thus, the overall space complexity of RecFlow is:

$$\mathcal{O}(|U| \times |V| + |U|^2 + (|U| + |V|) \times d) \tag{19}$$

This accounts for the storage of the graph structures and the user and item embeddings. And space complexity is O(D), dependent solely on data dimensionality and independent of timesteps.

Theoretical Analysis. In the context of generative modeling, **Flow Matching** and **DDPM** both aim to generate data through controlled transformations of noise. As illustrated in Figure 3, FM constructs a linear interpolation between X_0 and X_1, leading to a continuous and deterministic path. Mathematically, this is described as:

$$X_t = (1 - t)X_0 + tX_1 \tag{20}$$

where X_t follows a simple ODE-driven trajectory. In contrast, DDPM follows a stochastic diffusion process:

$$X_t = \sqrt{\alpha_t}X_0 + \sqrt{\beta_t}\epsilon \tag{21}$$

where ϵ is sampled from a Gaussian prior. The nonlinear and stochastic nature of DDPM results in higher variance in sampling paths, leading to inefficiencies. And FM is governed by an Ordinary Differential Equation (ODE), which enables continuous time evaluation and faster sampling via adaptive solvers. In contrast, DDPM relies on discrete Stochastic Differential Equations (SDEs), requiring a large number of steps for accurate generation. Empirically, FM achieves comparable quality with fewer function evaluations, reducing computational overhead.

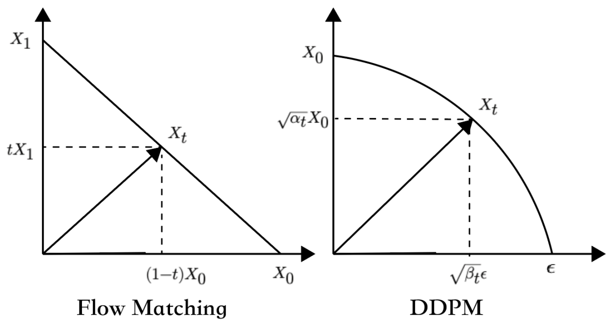

Figure 3. Theoretical comparison between DDPM and Flow Matching.

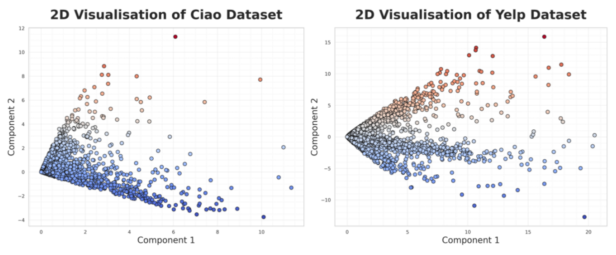

Figure 4. A illustration of anisotropic attributes of two typical datasets, i.e., Ciao and Epinions.

5. Experiment

In this section, we present a series of experiments conducted to evaluate the performance of our RecFlow method, focusing on the following six questions:

- **Q1:** How does RecFlow perform in comparison to other state-of-the-art social recommendation methods?
- **Q2:** What are the key contributions of RecFlow's main modules?
- **Q3:** Is RecFlow robust enough to effectively handle noisy and sparse data in social recommendation (SR)?
- **Q4:** How do different settings impact the performance of RecFlow?
- **Q5:** How does RecFlow efficiently capture and represent the anisotropy of social data in different timesteps?
- **Q6:** How does the complexity of our method compare to that of alternative approaches?

Before showing and analyzing the experimental results, we first present the experimental settings below.

5.1. Experiment Settings

Datasets and Evaluation Metrics. We conducted experiments on three publicly available social recommendation datasets: Ciao, Yelp and Epinions. Detailed statistics for these datasets are provided in Table 2.

We conducted a preliminary analysis of the Ciao and Epinions dataset. Figure.4 shows the distribution of social data in the Ciao and Epinions dataset. After reducing the data from high-dimensional space to two dimensions using Principal Component Analysis (PCA), it can be observed that the data points are more spread out along Component 1, while the distribution is more concentrated and less variable along Component 2. This aligns with the characteristics of social network data, where there are often dominant relational patterns, while others are secondary or sparse. In the social

Table 1. Statistics of experimental datasets

Data	Ciao	Yelp	Epinions
# **Users**	1,925	99,262	14,680
# **Items**	15,053	105,142	233,261
# **Interactions**	23,223	672,513	447,312
# **Social Ties**	65,084	1,298,522	632,144

graph, certain user groups with strong internal connections form tightly-knit social clusters, reflecting prominent relational patterns in the data. Additionally, the presence of a dominant direction in the data causes the vector field to display a clear anisotropic distribution, indicating that user interactions are more concentrated along specific directions.

Evaluation Protocols. To evaluate, we ultilized 2 commonly used metrics: Hit Ratio HR@N and Normalized Discounted Cumulative Gain (NDCG)@N as metrics, where N represents the number of items recommended to the user, widely ultilized in Top-N recommendations.

Compared Baselines. We compared RecFlow with 12 baseline models representing the latest social recommendation research approaches, including: PMF (Salakhutdinov & Mnih, 2007), TrustMF (Yang et al., 2013), GraphRec (Fan et al., 2019), DiffNet (Wu et al., 2019a), DGRec (Song et al., 2019a), NGCF (Wang et al., 2019), MHCN (Yu et al., 2021a), KCGN (Huang et al., 2021a), SMIN (Long et al., 2021), GDMSR(Quan et al., 2023), DSL (Wang et al., 2023a), RecDiff (Li et al., 2024a).

Implementation Details. All experiments are conducted on a machine with an RTX A800 for a fair comparison. The experimental settings and hyper-parameters details of our RecFlow framework are elaborated in. The learning rate was tuned within $[5e^{-4}, 1e^{-3}, 5e^{-3}]$ with a 0.96 decay factor per epoch. Batch sizes were selected from [1024, 2048, 4096, 8192], and hidden dimensions from [64, 128, 256, 512]. The parameter γ was set according to the γ_{pct}-percentile of node embedding distances for each dataset. The optimal number of GNN layers was chosen from [1, 2, 3, 4]. The Timestep embedding size is selected from 4,8,16,32. And the batch size for Ciao is 2048, while for Yelp and Epinions is 4096. Regularization weights λ_1 were selected from $[1e^{-3}, 1e^{-2}, 1e^{-1}, 1e^{0}, 1e^{1}]$.

5.2. Performance Comparison (RQ1)

Table 2 summarizes the experimental results across three datasets, with RecFlow's metrics bolded and top baselines underlined. RecFlow demonstrates marked improvements over existing approaches, more specifically, on Ciao, RecFlow achieves 0.725 Recall (+2.0%) and 0.438 NDCG (+4.5%) over RecDiff. For Yelp and Epinions, it maintains robust gains: 0.618 vs. 0.597 Recall(+3.5%) and 0.341 vs. 0.308 NDCG (+10.7%) on Yelp; 0.486 vs. 0.460 Recall (+ 5.6%) on Epinions, demonstrating adaptability to varying data densities. Notably, methods with self-supervised learning (SSL)—MHCN (local-global contrast), KCGN, SMIN (hierarchical relations), and DSL (predictive consistency)—consistently outperform traditional approaches. SSL mitigates noise propagation and interaction sparsity by extracting latent relational patterns, enabling stable representation learning. RecFlow's denoising process, guided by a velocity field, directly optimizes trajectories toward clean data distributions, suppressing noise during refinement. This explains its 5.6–10.7% improvements over diffusion-based RecDiff on sparse datasets.

Table 2. Overall performance analysis.

	Ciao		Yelp		Epinions	
Method	**Recall**	**NDCG**	**Recall**	**NDCG**	**Recall**	**NDCG**
TrustMF	0.539	0.343	0.371	0.193	0.265	0.195
SAMN	0.604	0.384	0.403	0.208	0.329	0.226
DiffNet	0.528	0.328	0.557	0.292	0.384	0.273
GraphRec	0.540	0.335	0.419	0.201	0.334	0.246
DGRec	0.517	0.319	0.410	0.209	0.326	0.236
NGCF	0.559	0.363	0.450	0.230	0.353	0.243
MHCN	0.621	0.378	0.567	0.292	0.438	0.321
KCGN	0.602	0.350	0.460	0.234	0.2201	0.1456
SMIN	0.588	0.354	0.485	0.251	0.333	0.228
GDMSR	0.560	0.355	0.513	0.246	0.368	0.241
DSL	0.606	0.389	0.504	0.259	0.365	0.267
RecDiff	0.712	0.419	0.597	0.308	0.460	0.336
RecFlow	**0.725**	**0.438**	**0.618**	**0.341**	**0.486**	**0.341**

5.3. Ablation Study (RQ2)

To evaluate the impact of different components in the RecFlow framework, we performed an ablation study using three benchmark datasets: Ciao, Yelp, and Epinions. The results are presented in Table 4.

- **w/o Flow-Matching**: This configuration excludes the holistic flow-matching module, leaving only the GNN for learning user-item and social relations. As shown in Table 4, the absence of the flow module results in a significant decrease in both Recall and NDCG across all datasets. Specifically, Recall drops by approximately 13% (Ciao), 7% (Yelp), and 12% (Epinions), while NDCG decreases by about 13% (Ciao), 16% (Yelp), and 31% (Epinions). This emphasizes the importance of the denoising mechanism in our model.

- **w/o CL**: In this configuration, we remove the conditional learning (CL) guidance for flow matching. The results show a notable performance decline, particularly in NDCG across all datasets, with a decrease of around 8% (Ciao), 11% (Yelp), and 12% (Epinions).

Table 3. Comparison of different sampling methods

	Ciao		Yelp		Epinions	
Method	**Recall**	**NDCG**	**Recall**	**NDCG**	**Recall**	**NDCG**
ODE-Solver	0.725	0.438	0.618	0.341	0.486	0.341
Multistep Heun	0.710	0.411	0.594	0.322	0.460	0.325
RK4	0.699	0.403	0.590	0.317	0.453	0.320
Heun	0.683	0.399	0.581	0.319	0.449	0.318
Euler	0.670	0.383	0.570	0.311	0.438	0.305

Table 4. Different solver of sampling

	Ciao		Yelp		Epinions	
Method	**Recall**	**NDCG**	**Recall**	**NDCG**	**Recall**	**NDCG**
RecFlow	0.725	0.438	0.618	0.341	0.486	0.341
RecFlow w/o Flow	0.633	0.380	0.573	0.301	0.429	0.297
RecFlow w/o CL	0.692	0.401	0.597	0.320	0.443	0.312
RecFlow w/o CL of both	0.621	0.407	0.589	0.312	0.417	0.302

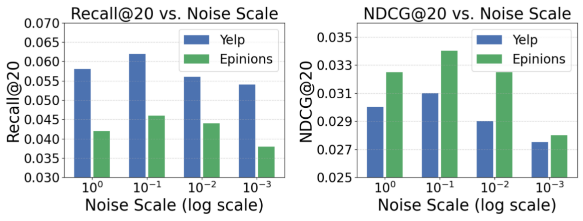

Figure 5. Robustness Analysis.

This demonstrates the critical role of **C**onditional **L**earning in improving model accuracy.

- **w/o both**: In this case, both the flow module and the CL label guidance are removed. The performance experiences a significant drop, with Recall decreasing by about 14% (Ciao), 5% (Yelp), and 7% (Epinions), and NDCG dropping by approximately 7% (Ciao), 9% (Yelp), and 15% (Epinions). This highlights the essential contributions of both the flow-matching process and label guidance in enhancing the model's ability to learn effective user-item and social relationships.

The choice of sampling method creates a clear trade-off between computational cost and model accuracy (see Table 3): the basic Euler method, with its single first-order step, yields the lowest Recall and NDCG, while Heun's two-stage predictor–corrector boosts both metrics modestly at only twice the cost. Leveraging history, the multistep Heun scheme further raises performance, and the four-stage RK4 delivers similar gains by reducing local error through fourth-order updates. Finally, an adaptive ODE solver such as Dormand–Prince, which dynamically adjusts its step size to satisfy error tolerances, consistently achieves the highest Recall and NDCG on Ciao (and likewise leads on Yelp and Epinions), demonstrating that, when resources allow, more precise integration yields the strongest recommendation quality.

5.4. Robustness Analysis (RQ3)

This section examines the effect of the noise scale factor (τ) on the noising process. By scaling the minimum and maximum noise in the scheduler to $\tau \cdot \bar{s}_{\text{min}}$ and $\tau \cdot \bar{s}_{\text{max}}$, respectively, we test the model's performance at different noise scales (1, 0.1, 0.01, 0.001). The results, shown in Figure 5, reveal the following:

- **Increasing the noise scale** improves model performance, with higher Recall@20 and NDCG@20 values for both Yelp and Epinions as the noise scale decreases from 1 to 0.1. This demonstrates the effectiveness of the RecDiff framework's denoising mechanism.

- **Excessive noise beyond a threshold** leads to performance degradation, especially for Yelp and Epinions. As noise scales reach 10^{-2} and 10^{-3}, a noticeable decline in NDCG@20 suggests that too much noise interferes with the model's ability to retain important user-item data.

5.5. Sensitivity Analysis (RQ4)

This section explores the influence of key hyperparameters on model performance, including the dimensionality of hidden representations (d), timestep embedding (d'), and maximum diffusion steps (T) on all datasets. On the Ciao dataset, Recall@20 and NDCG@20 show fluctuations as embedding dimensions, timestep embedding sizes, and diffusion steps change. More specifically, when the embedding dimension is d=64, Recall@20 is 0.725 and NDCG@20 is 0.438, which are the best values compared to other settings. On the Yelp dataset, all settings yield relatively low and similar values for Recall@20 and NDCG@20. The best performance is achieved with an embedding dimension of d=120 and a timestep embedding size of T=50. On the Epinions dataset, the best performance occurs with embedding dimension d=64, and performance improves as the timestep embedding size increases. These results reveal the following:

- **Embedding dimensionality (d)**: For the Ciao and Epinions datasets, embedding dimensions of d=64 give the best results, whereas for the Yelp dataset, increasing the dimension to d=120 provides optimal performance. This shows the importance of adapting the dimensionality based on the dataset's characteristics and size. Increasing d generally improves performance, except for Ciao and Epinions, where larger values cause slight degradation due to overfitting.

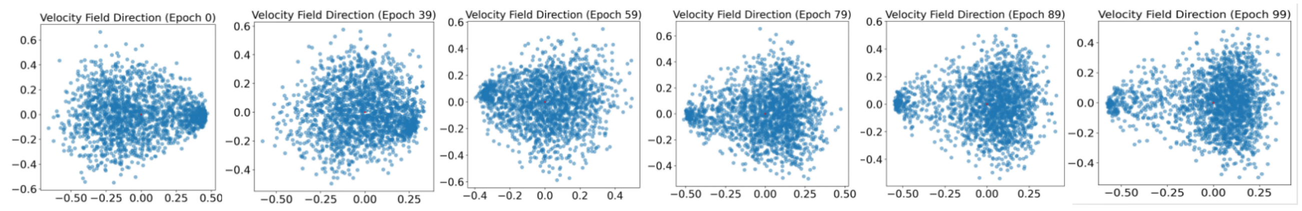

Figure 6. Convergence of the velocity field direction over different epochs.

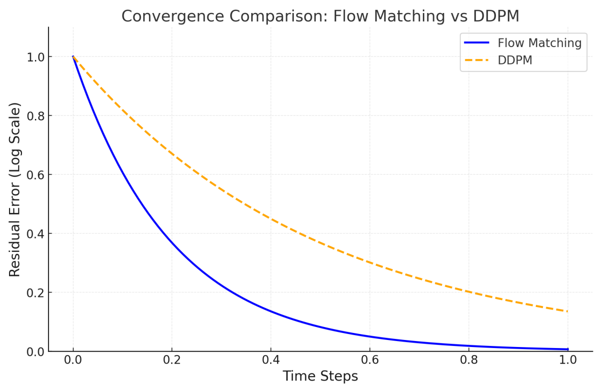

Figure 7. Convergence comparison of loss curves

- **Time step embedding size (d')**: On the Ciao and Yelp datasets, the timestep embedding size plays a crucial role in optimizing performance, with values of 50 giving the best results for Yelp. For Epinions, larger timestep embeddings improve performance progressively, indicating that the model benefits from higher temporal granularity. Larger dimensions boost diffusion's positive impact on denoising. However, excessively large sizes reduce diffusion effectiveness, lowering performance.

5.6. Convergence Analysis (RQ5)

In Figure.7, Flow Matching exhibits a faster convergence speed compared to DDPM, as the horizontal axis represents the diffusion model's time steps, ranging from 0 to 1. The vertical axis indicates the residual error at each time step. It is evident that the residual error of Flow Matching decreases rapidly in the early time steps, demonstrating a significantly faster convergence trend. In contrast, DDPM's error decreases at a slower pace, highlights the advantage of Flow Matching in modeling vector fields and achieving efficient convergence, making it more suitable for handling complex distributions and data scenarios.

5.7. Visualization of Flow matching velocity field direction over different epochs(RQ6)

To better understand how the velocity field evolves during training, we design an experiment that visualizes the direction of the flow-matching vector field across different epochs. In Figure.6, the data points are unevenly distributed, showing clear directionality and concentration, which highlights the anisotropy of social data. The model captures this anisotropy within the velocity field, represented by arrows that dynamically adjust direction as the epoch evolves, demonstrates the convergence dynamics of velocity field directions. In early stages (Epoch 0–38), directions exhibit high dispersion (-0.50 to +0.50), reflecting unstable parameter adjustments. As training progresses (Epoch 59–99), directions progressively cluster near the origin (range: -0.25 to +0.25), indicating stabilized optimization trajectories. Notably, post-Epoch 78, data density intensifies with directional similarity, signifying unified parameter updates.

6. Conclusion

In this paper, we propose RecFlow, a novel flow-matching generative model tailored for social recommendation. Our approach leverages flow matching on user-user graphs to enhance recommendation accuracy. To evaluate the effectiveness of RecFlow, we conduct extensive experiments against several strong baselines, and the results clearly demonstrate the superior performance and robustness of our method. Looking forward, future work will explore more efficient and scalable extensions of to explore its practicality and effectiveness in real-world recommendation scenarios.

Impact Statement

This paper introduces RecFlow, a flow-based social recommendation model that captures anisotropic in user interactions. By leveraging flow matching, RecFlow enhances representation learning and denoising efficiency, emphasizes the practical benefits for personalized recommendation. We also acknowledge potential societal risks, such as bias amplification, and highlight the need for fairness and robustness in future recommendation systems.

Acknowledgments

This work is supported by National Natural Science Foundation of China (No .62302507) and Hunan Provincial Natural Science Foundation (No. 2023JJ40684).

References

Bai, T., Zhang, Y., Wu, B., and Nie, J.-Y. Temporal graph neural networks for social recommendation. In *2020 IEEE International Conference on Big Data (Big Data)*, pp. 898–903. IEEE, 2020.

Dai, E., Lin, M., Zhang, X., and Wang, S. Unnoticeable backdoor attacks on graph neural networks. In *Proceedings of the ACM Web Conference 2023*, pp. 2263–2273, 2023.

Deng, Y., He, X., Mei, C., Wang, P., and Tang, F. Fireflow: Fast inversion of rectified flow for image semantic editing, 2024. URL https://arxiv.org/abs/2412.07517.

Fan, W., Ma, Y., Li, Q., He, Y., Zhao, E., Tang, J., and Yin, D. Graph neural networks for social recommendation. In *The world wide web conference*, pp. 417–426, 2019.

Guo, Y., Cai, F., Chen, H., Chen, C., Zhang, X., and Zhang, M. An explainable recommendation method based on diffusion model. In *2023 9th International Conference on Big Data and Information Analytics (BigDIA)*, pp. 802–806. IEEE, 2023.

He, X., Fan, W., Wang, R., Wang, Y., Wang, Y., Pan, S., and Wang, X. Balancing user preferences by social networks: A condition-guided social recommendation model for mitigating popularity bias. *arXiv preprint arXiv:2405.16772*, 2024.

Huang, C., Xu, H., Xu, Y., Dai, P., Xia, L., Lu, M., Bo, L., Xing, H., Lai, X., and Ye, Y. Knowledge-aware coupled graph neural network for social recommendation. In *AAAI*, pp. 4115–4122. AAAI Press, 2021a. URL https://ojs.aaai.org/index.php/AAAI/article/view/16533.

Huang, C., Xu, H., Xu, Y., Dai, P., Xia, L., Lu, M., Bo, L., Xing, H., Lai, X., and Ye, Y. Knowledge-aware coupled graph neural network for social recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 4115–4122, 2021b.

Jiang, Y., Yang, Y., Xia, L., and Huang, C. Diffkg: Knowledge graph diffusion model for recommendation, 2023. URL https://arxiv.org/abs/2312.16890.

Jiang, Y., Xia, L., Wei, W., Luo, D., Lin, K., and Huang, C. Diffmm: Multi-modal diffusion model for recommendation, 2024. URL https://arxiv.org/abs/2406.11781.

Kingma, D. P., Salimans, T., Poole, B., and Ho, J. Variational diffusion models, 2023. URL https://arxiv.org/abs/2107.00630.

Lee, S., Lin, Z., and Fanti, G. Improving the training of rectified flows, 2024. URL https://arxiv.org/abs/2405.20320.

Li, Z., Xia, L., and Huang, C. Recdiff: Diffusion model for social recommendation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pp. 1346–1355, 2024a.

Li, Z., Xia, L., and Huang, C. Recdiff: Diffusion model for social recommendation, 2024b. URL https://arxiv.org/abs/2406.01629.

Liang, K., Meng, L., Liu, M., Liu, Y., Tu, W., Wang, S., Zhou, S., Liu, X., and Sun, F. A survey of knowledge graph reasoning on graph types: Static, dynamic, and multimodal, 2023. URL https://arxiv.org/abs/2212.05767.

Lin, M., Xiao, T., Dai, E., Zhang, X., and Wang, S. Certifiably robust graph contrastive learning. *Advances in Neural Information Processing Systems*, 36:17008–17037, 2023.

Lin, M., Chen, Z., Liu, Y., Zhao, X., Wu, Z., Wang, J., Zhang, X., Wang, S., and Chen, H. Decoding time series with llms: A multi-agent framework for cross-domain annotation. *arXiv preprint arXiv:2410.17462*, 2024a.

Lin, M., Dai, E., Xu, J., Jia, J., Zhang, X., and Wang, S. Stealing training graphs from graph neural networks. *arXiv preprint arXiv:2411.11197*, 2024b.

Lin, M., Zhang, Z., Dai, E., Wu, Z., Wang, Y., Zhang, X., and Wang, S. Trojan prompt attacks on graph neural networks. *arXiv preprint arXiv:2410.13974*, 2024c.

Lin, M., Liu, H., Tang, X., Zeng, J., Dai, Z., Luo, C., Li, Z., Zhang, X., He, Q., and Wang, S. How far are llms from real search? a comprehensive study on efficiency, completeness, and inherent capabilities. *arXiv preprint arXiv:2502.18387*, 2025.

Lipman, Y., Chen, R. T., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Liu, Q., Yan, F., Zhao, X., Du, Z., Guo, H., Tang, R., and Tian, F. Diffusion augmentation for sequential recommendation. In *Proceedings of the 32nd ACM International*

Conference on Information and Knowledge Management, pp. 1576–1586, 2023a.

Liu, S., Zhang, A., Hu, G., Qian, H., and seng Chua, T. Preference diffusion for recommendation, 2024. URL https://arxiv.org/abs/2410.13117.

Liu, X., Gong, C., and Liu, Q. Flow straight and fast: Learning to generate and transfer data with rectified flow, 2022. URL https://arxiv.org/abs/2209.03003.

Liu, Y., Chen, L., He, X., Peng, J., Zheng, Z., and Tang, J. Modelling high-order social relations for item recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 34(9):4385–4397, 2020.

Liu, Z., Mei, S., Xiong, C., Li, X., Yu, S., Liu, Z., Gu, Y., and Yu, G. Text matching improves sequential recommendation by reducing popularity biases, 2023b. URL https://arxiv.org/abs/2308.14029.

Long, X., Huang, C., Xu, Y., Xu, H., Dai, P., Xia, L., and Bo, L. Social recommendation with self-supervised metagraph informax network. In Demartini, G., Zuccon, G., Culpepper, J. S., Huang, Z., and Tong, H. (eds.), *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, pp. 1160–1169. ACM, 2021. doi: 10.1145/3459637.3482480. URL https://doi.org/10.1145/3459637.3482480.

Quan, Y., Ding, J., Gao, C., Yi, L., Jin, D., and Li, Y. Robust preference-guided denoising for graph based social recommendation. In *Proceedings of the ACM Web Conference 2023*, WWW '23, pp. 1097–1108. ACM, April 2023. doi: 10.1145/3543507.3583374. URL http://dx.doi.org/10.1145/3543507.3583374.

Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. Bpr: Bayesian personalized ranking from implicit feedback. abs/1205.2618:452–461, 2012.

Salakhutdinov, R. and Mnih, A. Probabilistic matrix factorization. In Platt, J. C., Koller, D., Singer, Y., and Roweis, S. T. (eds.), *NeurIPS*, pp. 1257–1264. Curran Associates, Inc., 2007. URL https://proceedings.neurips.cc/paper/2007/hash/d7322ed717dedf1eb4e6e52a37ea7bcd-Abstract.html.

Song, W., Xiao, Z., Wang, Y., Charlin, L., Zhang, M., and Tang, J. Session-based social recommendation via dynamic graph attention networks. In Culpepper, J. S., Moffat, A., Bennett, P. N., and Lerman, K. (eds.), *WSDM 2019*, pp. 555–563. ACM, 2019a. doi: 10.1145/3289600.3290989. URL https://doi.org/10.1145/3289600.3290989.

Song, W., Xiao, Z., Wang, Y., Charlin, L., Zhang, M., and Tang, J. Session-based social recommendation via dynamic graph attention networks. In *Proceedings of the Twelfth ACM international conference on web search and data mining*, pp. 555–563, 2019b.

Wang, B., Lin, M., Zhou, T., Zhou, P., Li, A., Pang, M., Li, H., and Chen, Y. Efficient, direct, and restricted black-box graph evasion attacks to any-layer graph neural networks via influence function. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pp. 693–701, 2024.

Wang, T., Xia, L., and Huang, C. Denoised self-augmented learning for social recommendation. In *IJCAI 2023*, pp. 2324–2331. ijcai.org, 2023a. doi: 10.24963/IJCAI.2023/258. URL https://doi.org/10.24963/ijcai.2023/258.

Wang, W., Xu, Y., Feng, F., Lin, X., He, X., and Chua, T.-S. Diffusion recommender model, 2023b. URL https://arxiv.org/abs/2304.04971.

Wang, X., He, X., Wang, M., Feng, F., and Chua, T. Neural graph collaborative filtering. In Piwowarski, B., Chevalier, M., Gaussier, É., Maarek, Y., Nie, J., and Scholer, F. (eds.), *Proc. of SIGIR*, pp. 165–174. ACM, 2019. doi: 10.1145/3331184.3331267. URL https://doi.org/10.1145/3331184.3331267.

Wu, L., Sun, P., Fu, Y., Hong, R., Wang, X., and Wang, M. A neural influence diffusion model for social recommendation. In Piwowarski, B., Chevalier, M., Gaussier, É., Maarek, Y., Nie, J., and Scholer, F. (eds.), *Proc. of SIGIR*, pp. 235–244. ACM, 2019a. doi: 10.1145/3331184.3331214. URL https://doi.org/10.1145/3331184.3331214.

Wu, L., Sun, P., Fu, Y., Hong, R., Wang, X., and Wang, M. A neural influence diffusion model for social recommendation. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pp. 235–244, 2019b.

Wu, Q., Zhang, H., Gao, X., He, P., Weng, P., Gao, H., and Chen, G. Dual graph attention networks for deep latent representation of multifaceted social effects in recommender systems. In *The world wide web conference*, pp. 2091–2102, 2019c.

Xu, F., Lian, J., Han, Z., Li, Y., Xu, Y., and Xie, X. Relation-aware graph convolutional networks for agent-initiated social e-commerce recommendation. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pp. 529–538, 2019.

Yang, B., Lei, Y., Liu, D., and Liu, J. Social collaborative filtering by trust. In Rossi, F. (ed.), *IJCAI 2013*, pp. 2747–2753. IJCAI/AAAI, 2013. URL http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6750.

Yu, J., Yin, H., Li, J., Wang, Q., Hung, N. Q. V., and Zhang, X. Self-supervised multi-channel hypergraph convolutional network for social recommendation. In Leskovec, J., Grobelnik, M., Najork, M., Tang, J., and Zia, L. (eds.), *Proc. of WWW*, pp. 413–424. ACM / IW3C2, 2021a. doi: 10.1145/3442381.3449844. URL https://doi.org/10.1145/3442381.3449844.

Yu, J., Yin, H., Li, J., Wang, Q., Hung, N. Q. V., and Zhang, X. Self-supervised multi-channel hypergraph convolutional network for social recommendation. In *Proceedings of the web conference 2021*, pp. 413–424, 2021b.

Zhao, W., Shi, M., Yu, X., Zhou, J., and Lu, J. Flow-turbo: Towards real-time flow-based image generation with velocity refiner, 2024. URL https://arxiv.org/abs/2409.18128.