Graph-Based Diffusion Model for Service Recommendation

Xuan Zhang, Xiang Deng, Hongxing Yuan, Chunyu Wei, Yushun Fan, and Jia Zhang, *Senior Member, IEEE*

***Abstract*—With the widespread adoption of cloud-based services and Service-Oriented Computing, efficient service recommendation has become pivotal for optimizing service discovery and composition in large-scale ecosystems. While recent diffusion-based recommendation methods have achieved impressive results in service-oriented scenarios, existing approaches predominantly treat user-service interactions as isolated events, overlooking the potential of higher-order collaborative signals between users and services. Such signals, which encapsulate richer and more nuanced relationships, can be naturally captured using graph-based data structures. To address this limitation, we extend diffusion-based service recommendation methods to the graph domain by directly modeling user-service bipartite graphs with diffusion models. This enables better modeling of the higher-order connectivity inherent in complex interaction dynamics. However, this extension introduces two primary challenges: (1) Noise Heterogeneity, where interactions are influenced by various forms of continuous and discrete noise, and (2) Relation Explosion, referring to the high computational costs of processing large-scale graphs. To tackle these challenges, we propose a Graph-based Diffusion Model for Service Recommendation (GDMSR). To address noise heterogeneity, we introduce a multi-level noise corruption mechanism that integrates both continuous and discrete noise, effectively simulating real-world interaction complexities. To mitigate relation explosion, we design a user-active guided diffusion process that selectively focuses on the high-value edges and active users, reducing inference costs while preserving critical service-level dependencies. Extensive experiments on six real-world service datasets demonstrate that GDMSR consistently outperforms state-of-the-art methods, highlighting its effectiveness in capturing higher-order collaborative signals and improving service recommendation performance.**

***Index Terms*—Generative service recommendation, diffusion model, graph neural networks.**

Received 23 April 2025; revised 15 December 2025; accepted 16 December 2025. Date of publication 22 December 2025; date of current version 5 February 2026. This work was supported by the National Natural Science Foundation of China under Grant 62173199 and Grant 6250072448. *(Corresponding authors: Chunyu Wei; Yushun Fan.)*

Xuan Zhang, Hongxing Yuan, and Yushun Fan are with the Department of Automation, Beijing National Research Center for Information Science, Technology, Tsinghua University, Beijing 100190, China (e-mail: xuan-zha23@mails.tsinghua.edu.cn; yuanhx24@mails.tsinghua.edu.cn; fanyus@tsinghua.edu.cn).

Xiang Deng is with the Department of Automation, Tsinghua University, Beijing 100190, China (e-mail: dx23@mails.tsinghua.edu).

Chunyu Wei is with the School of Information, Renmin University of China, Beijing 100872, China (e-mail: weichunyu@ruc.edu.cn).

Jia Zhang is with the Department of Computer Science, Southern Methodist University, Dallas, TX 75205 USA (e-mail: jiazhang@smu.edu).

The source code will be made publicly available at: https://github.com/zx19971219/GDMSR.

Digital Object Identifier 10.1109/TSC.2025.3646723

I. Introduction

With the rapid development and proliferationof the internet, more and more service platforms have put their services online [1], such as e-commerce platforms (e.g., Amazon, Taobao) [4] and social networks (e.g., Facebook). Consequently, the variety of online services has expanded significantly, encompassing both traditional software services, such as web services or APIs, and any web component accessible via HTTP calls [2]. These services, which embody a broad concept of knowledge services, provide users with a wide range of options. Service recommendation has enhanced user experiences by suggesting services and content that align with individual preferences, effectively mitigating information overload across various online domains. In contrast to traditional discriminative model-based recommender systems [5], [6], [7], generative recommender models based on Generative Adversarial Networks (GANs) [8] or Variational Auto-Encoders (VAEs) [9] hypothesis that user-service interactions are correlated with various latent factors, such as user preferences and service popularity. Due to their superior capacity to model the joint distribution of these complex latent factors, generative recommender systems have demonstrated significant advancements [10], [11], [12].

Recently, the notable successes of diffusion models (DMs) [13] in high-quality image generation tasks have inspired researchers to explore their application in service recommender systems [14], [15], [17]. Compared to traditional generative methods, DMs provide a highly stable training process [20] and can be interpreted through score matching [32], [33], [34] and Langevin dynamics [35], [36]. Additionally, they can be understood from the perspective of diffusion probabilistic models [13], [37], which define a forward diffusion process to add noise to data and a reverse process to recover it.

However, most existing diffusion-based recommendation methods treat a period of interaction sequence [22], [23], [24], [89] or historical interaction sequence [77] as a single training sample. These methods overlook the higher-order collaborative signals between users and services in complex service systems. For example, if two users have interacted with many of the same services, it suggests they share similar preferences. Additionally, longer interaction chains, such as $s_1 \leftarrow u_2 \leftarrow s_3 \leftarrow u_4$ implies that u_4 is likely to show interest in s_1, as a similar user u_2 has already interacted with s_1 [25]. Therefore, we argue that it is crucial to explicitly consider these higher-order connections in diffusion-based models focused on interaction generation. Graphs, as a data structure, are inherently suited to modeling relationships between entities in service scenarios. Numerous studies have incorporated graphs into recommender systems, employing techniques such as random walks [26], [27], [28],

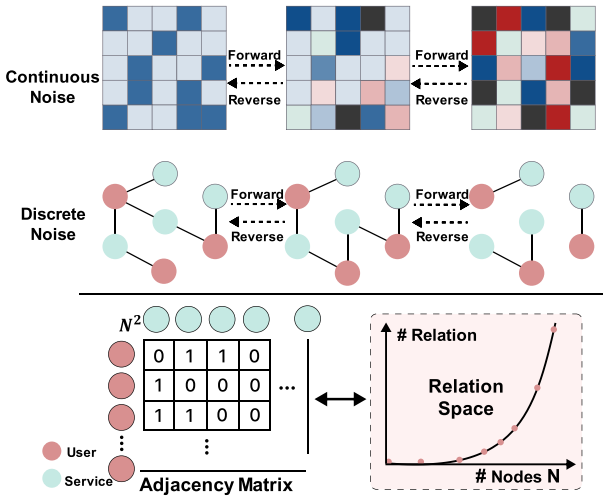

Fig. 1. Service systems involve various forms of heterogeneous noise (continuous noise and discrete noise). As the number of users and services increases, the complexity of their interactions can grow explosively.

[29] or multi-layer graph convolutions [30], [31], [75] to effectively capture higher-order collaborative signals among services. Consequently, we believe that introducing graph-based topological constraints in diffusion-based recommendation methods can better model the higher-order connectivity in complex interaction generation. To achieve this goal, we need to address two notable challenges, as illustrated in Fig. 1.

- *Noise Heterogeneity:* In recommendation contexts, user-service interactions are inherently complex, shaped by both discrete and continuous factors. Discrete feedback, such as clicks or purchases, reflects binary decisions driven by historical interactions, while continuous variables, like interaction duration, capture nuanced preferences. These combined factors form a joint distribution that requires heterogeneous noise modeling. Consequently, most existing diffusion models [15], [77] relying on a single noise type may be inadequate for the complex dynamics of recommendation scenarios.
- *Relation Explosion:* Traditional graph-based diffusion models require the computation of a latent vector or probability for each node pair in the graph, resulting in a computational complexity of $O(N^2)$. In recommender systems, where the number of users and services is typically very large, this $O(N^2)$ complexity poses a significant challenge to the application of traditional graph-based diffusion models.

To this end, we propose a Graph-based Diffusion Model for Service Recommendation (GDMSR) to address the aforementioned challenges synergistically.

To address the issue of **Noise Heterogeneity**, we propose a multi-level corruption to capture various forms of noise present in complex recommender systems during the *forward process*. For user feature-level perturbations, we introduce continuous noise to corrupt their feature vectors, simulating varying intensities of perturbations in real-world scenarios. For structure-level perturbations, we introduce discrete noise to corrupt interactions between users and services. To ensure consistency in denoising, we first employ an alignment module within the denoising network to integrate the corrupted features and topological structure into a unified corrupted graph. Subsequently, through multiple layers of graph convolution, we iteratively aggregate the higher-order collaborative signals, thereby enhancing graph denoising capabilities. This approach effectively models heterogeneous noise in complex recommender systems by merging both levels into a unified space, thus facilitating joint denoising.

To address the issue of **Relation Explosion**, we propose a user-active guided generation strategy to enhance the *reverse process* of the diffusion model. On one hand, we preserve the original graph's structural information, specifically the user node degree distribution, allowing us to edit edges during the forward process without compromising the integrity of the original interaction structure. On the other hand, during the reverse process, we identify active users based on their degree distribution and retain only the corresponding edges, discarding those associated with inactive users. This ensures that computational resources are prioritized for the most important edges and users, making the model well-suited for large-scale service recommendation scenarios. The contributions of this paper are summarized as follows:

- We propose a novel Graph-based Diffusion Model for Service Recommendation (GDMSR) that applies multi-level corruption to capture heterogeneous noise in real-world service scenarios while accounting for the higher-order connectivity in complex interaction generation.
- We incorporate a user-active guided generation strategy to more effectively alleviate the heavy computational burden associated with iterative refinement in the reverse process of the diffusion model.
- Extensive experiments demonstrate that our method outperforms state-of-the-art baselines across six datasets. The comprehensive analysis and experimental results confirm the computational efficiency of our approach.

The remainder of this article is organized as follows. Section II discusses related work. Section III formally defines the problem. Section IV details the proposed GDMSR framework. Section V presents extensive experiments with analyses.Finally, Section VI draws conclusions.

II. Related Work

A. Generative Service Recommendation

Discriminative service recommendation models predict the probability of interactions between users and services [38], [39]. While existing discriminative models are relatively easy to train, generative service recommendation models better capture the underlying data distribution and complex, non-linear relationships [40], [41]. Most generative models can be broadly categorized into two main types: VAE-based approaches [43], [44], proposed by [9], which have demonstrated effectiveness in capturing the latent structure of user interactions by learning an encoder for posterior estimation [42] and a decoder for predicting interaction probabilities across all services [45]. GAN-based models, exemplified by the work of [46], further improve service recommendation quality by addressing data sparsity and cold-start issues through adversarial training that refines the service recommendation distribution [47], [48], [49]. Additionally, generative retrieval models have been explored in

sequential service recommendation tasks, demonstrating their potential to handle sequential dependencies and improve service recommendation accuracy [93], [94], [95].

Recently, diffusion model-based service recommender systems have emerged as a superior approach for their stability and high-quality generation. By iteratively reducing noise, they offer improved robustness and flexibility over traditional generative models, resulting in more accurate and diverse service recommendations [15], [23], [51].

B. Diffusion Models

A central challenge in generative modeling is balancing flexibility and computational feasibility [88]. The core concept of diffusion models addresses this by systematically perturbing the data distribution through a forward diffusion process, followed by learning the reverse process to restore the data distribution [13]. This approach results in a generative model that is both highly flexible and computationally efficient [52], [53]. Existing diffusion models can be categorized into two types: conditional generation [54], [55], [56], [87] and unconditional generation [57], [58].

Although diffusion models are closely related to other research areas, such as computer vision [50], NLP [59], and signal processing [60], their progress in the field of personalized service recommendation has been relatively slow. InDiRec [85] replaces heuristic data augmentation with intent-aware diffusion, enabling more accurate and robust modeling of user intent for sequential recommendation tasks. DiffRec [77] employs a forward noise addition and reverse denoising process on user interaction histories to generate service recommendations. It treats the user interaction sequence as a single sample, neglecting the higher-order collaborative signals between users and services. DiffRec adds Gaussian noise to the user-service sequence step by step and utilizes Multilayer Perceptrons (MLPs) for denoising, but the simple MLP structure often fails to capture crucial higher-order signals for accurate service recommendations. Our GDMSR addresses this by implementing a graph-based diffusion model that captures higher-order signals in interactions. Recent studies have increasingly focused on the integration of higher-order information within diffusion processes to enhance service recommendation performance. GiffCF [90] leverages heat equations and filtering mechanisms on an service-service graph, effectively capturing service relationships through a diffusion framework. In contrast, GDMSR explicitly models the diffusion process on the user-service bipartite graph, enabling it to directly capture user-service interactions and their propagation. Meanwhile, CF-Diff [91] proposes a collaborative filtering approach based on diffusion models, where noise is modeled using a single Gaussian distribution. However, it does not explicitly incorporate graph-based diffusion. Instead, CF-Diff adopts rule-based encoding strategies—such as counting incoming links from $(h-1)$-hop neighbors—to extract higher-order collaborative signals. This approach is inherently static and non-learnable, limiting its flexibility and adaptability to complex service recommendation scenarios. Conversely, GDMSR adopts a user-guided denoising process, iteratively refining the graph structure. This step-by-step denoising not only reduces computational overhead but also yields a more accurate and efficient representation of higher-order collaborative signals. CDiff4Rec [84], on the other hand, extracts collaborative signals from service features (e.g., review words) by generating pseudo-users. It primarily focuses on user behavior similarity and enhances personalized recommendation by aggregating neighborhood information from both real and pseudo-users. However, CDiff4Rec does not explicitly model high-order user-service interactions within the graph structure. It simply adds Gaussian noise to user interaction vectors without distinguishing noise types, potentially overlooking heterogeneous noise patterns present in real-world scenarios. DiffGT [92] introduces anisotropic and directional noise into the diffusion process. However, by modeling discrete structural perturbations using continuous Gaussian noise, it fails to preserve the inherent sparsity of the service graph. This results in fully noisy graphs, which significantly hinder the denoising network's ability to capture the underlying structural properties of the service data. Additionally, some studies have examined diffusion processes within social networks [61], [62], primarily investigating how social connections influence user preferences under single-type noise [63]. These methods differ from GDMSR by focusing on social influence instead of direct user-service diffusion.

C. Graph-Based Service Recommendation

Graph Neural Networks (GNNs) have been essential in leveraging graph-structured data for service recommendation tasks [64], [65], [66]. The field has evolved from simple random walks [29] to Graph Convolutional Networks (GCNs) [31] and attention mechanisms-based methods [67]. Random walks [68] captured basic relationships but struggled with complex dependencies. GCNs [38], [69], [75] improved this by aggregating neighborhood information across layers, effectively capturing higher-order collaborative signals. Attention mechanism-based methods further enhance this by dynamically weighting nodes and edges [70], refining the capture of critical higher-order signals. However, most diffusion-based service recommendation methods [15], [23], [77] often overlook these higher-order collaborative signals in complex service recommender systems. GDMSR introduces graph topology constraints in diffusion-based service recommendation methods to better model the higher-order connectivity.

While existing diffusion-based recommendation models have made promising progress, they often suffer from limitations such as oversimplified denoising architectures, insufficient modeling of high-order graph structure, or reliance on heuristic designs. In contrast, GDMSR integrates learnable graph-based diffusion with a user-guided denoising strategy, offering a unified, scalable framework. This design enables more expressive modeling of collaborative signals and better adaptability to complex recommendation scenarios.

III. Preliminary

A. Diffusion Model

In this section, we introduce the core concepts of diffusion models (DMs), which comprise both forward and reverse processes.

a) Forward Process: Given an input data sample $\mathbf{y}^0$, the forward process is defined by $q(\mathbf{y}^t|\mathbf{y}^{t-1})$, which incrementally corrupts $\mathbf{y}^0$ over T steps by adding noise points $(\mathbf{z}^1, \ldots, \mathbf{z}^T)$. This process exhibits a Markov structure, where $q(\mathbf{y}^1, \ldots, \mathbf{y}^T|\mathbf{y}^0) = q(\mathbf{y}^1|\mathbf{y}^0) \prod_{t=2}^{T} q(\mathbf{y}^t|\mathbf{y}^{t-1})$. As $T \to \infty$, $q(\mathbf{y}^T)$ approaches a convergent distribution.

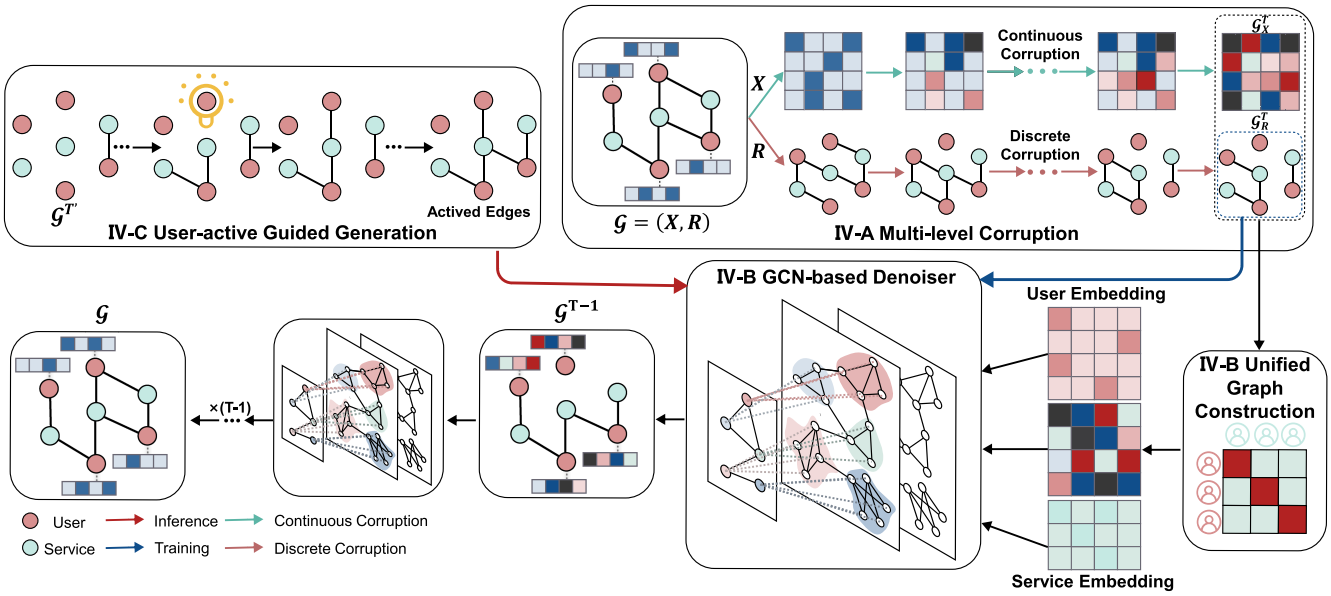

Fig. 2. The Framework of GDMSR. The bipartite graph is introduced with independent corruptions at both the structural and feature levels through Multi-level Corruption. A graph-based denoising network then aligns the user and service representations from corruptions, utilizing a GCN-based architecture to reconstruct the corrupted information.

b) Reverse Process: The denoising model ϕ_θ is trained to learn the reverse distribution $p_\theta(\mathbf{y}^{t-1}|\mathbf{y}^t)$ from $\mathbf{y}^t$ to $\mathbf{y}^{t-1}$. The model follows the joint distribution $p_\theta(\mathbf{y}^{0:T}) = p(\mathbf{y}^T)\prod_{t=1}^{T} p_\theta(\mathbf{y}^{t-1} \mid \mathbf{y}^t)$. $p(\mathbf{y}^T)$ is the convergent distribution in q. To generate new samples, noise is sampled from a prior distribution and then progressively inverted using a denoising model. Formally, both Gaussian noise and Bernoulli noise conform to this distribution.

Generally, an effective denoising model must satisfy three key conditions: 1) The distribution $q(\mathbf{y}^t|\mathbf{y}^0)$ should have a closed-form formula, allowing for parallel training across different time steps. 2) The posterior $p_\theta(\mathbf{y}^{t-1}|\mathbf{y}^t) = \int q(\mathbf{y^{t-1}}|\mathbf{y}^t, \mathbf{y}^0)dp_\theta(\mathbf{y}^0)$ should be expressed in closed form, enabling $\mathbf{y}^0$ to be the target for the denoising model. 3) The limit distribution $q_\infty = \lim_{T\to\infty} q(\mathbf{y}^T|\mathbf{y}^0)$ should be independent of $\mathbf{y}^0$ to be the prior distribution for inference.

B. Recap GCN

Let $|\mathcal{U}| = M$ and $|\mathcal{S}| = N$ represent the sizes of the user set and the service set in a service ecosystem, respectively. The interaction matrix $\mathbf{R} \in \mathbb{R}^{M\times N}$ denotes the interactions between users and services, where the entries of the matrix are defined as follows:

$$r_{us} = \begin{cases} 1 & \text{if user u interacted with service s,} \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

we define the set of all users and services as $\mathcal{V} = \mathcal{U} \cup \mathcal{S}$, and the edge set as $\mathcal{E} = \{(u, s) \mid r_{us} = 1, u \in \mathcal{U}, s \in \mathcal{S}\}$, where each edge (u, s) represents an interaction between user u and service s. We construct a bipartite graph $\mathcal{G} = (\mathbf{X}, \mathbf{R})$, where $\mathbf{X} = [x_{u1}, \ldots, x_{uM}] \in \mathbb{R}^{d_x \times M}$ represents the feature matrix of the users in $\mathcal{U}$, with d_x denoting the feature dimension. The adjacency matrix $\mathbf{A}_\mathcal{G}$ for the graph $\mathcal{G}$ is defined as:

$$\mathbf{A}_\mathcal{G} = \begin{pmatrix} 0 & \mathbf{R} \\ \mathbf{R}^T & 0 \end{pmatrix}. \tag{2}$$

The degree matrix $\mathbf{D}_\mathcal{G} \in \mathbb{N}^{(M+N)\times(M+N)}$ of $\mathcal{G}$ is a diagonal matrix, the diagonal element d_{ii} represents the number of non-zero entries in the i-th row vector of $\mathbf{A}_\mathcal{G}$. In Graph Convolutional Networks (GCNs) with L layers, the representation of an ego node is updated by aggregating information from its neighboring nodes:

$$\mathbf{Z}^{(l)} = F(\mathbf{Z}^{(l-1)}, \mathcal{G}), \tag{3}$$

here, $\mathbf{Z}^{(l)}$ represents the user or service representations at the l-th layer, and $\mathbf{Z}^{(l-1)}$ is the representations of the previous layer. The function F denotes the aggregation function used to aggregate information from neighboring nodes.

IV. Methodology

The overall framework of our proposed method is illustrated in Fig. 2. We introduce independent corruptions at both the structural and feature levels of the bipartite graph (Section IV-A), aiming to simulate the inherent noise in real-world recommender systems. A graph-based denoising network then aligns the user and service representations from corruptions, utilizing a GCN-based architecture to reconstruct the corrupted information (Section IV-B). To address relation explosion in large-scale systems, we propose a user-active guided generation strategy (Section IV-C) that selectively retains edges for activated users, reducing computational complexity and improving inference efficiency. Training and inference details are in Sections IV-D and IV-E.

A. Multi-Level Corruption

In real-world recommendation scenarios, noise is inherently heterogeneous. Existing diffusion-based methods typically rely on single-noise modeling, struggling to address the inherent complexity of recommendation tasks. To overcome this, we propose a novel approach integrating two distinct noise types into the forward diffusion process: *Discrete Corruption* and

Continuous Corruption. Independently applied at the structural and feature levels of the bipartite graph, these corruptions generate two complementary views for topological and numerical corruptions, respectively. This design enables our method to capture diverse forms of heterogeneous noise, enhancing model robustness and overall performance.

1) Discrete Corruption: To model topological corruption within the bipartite graph, we utilize a probabilistic transition matrix based on the marginal distributions of edge types. Specifically, we define a transition matrix $\boldsymbol{Q}^t$, where $[\boldsymbol{Q}^t]_{\mathbf{e},\mathbf{e}'} = q(e^t = \mathbf{e}' \mid e^{t-1} = \mathbf{e})$ represents the probability of an edge transitioning from state $\mathbf{e}$ at time step $t-1$ to state $\mathbf{e}'$ at time step t. Here, $\mathbf{e}$ and $\mathbf{e}'$ denote the two possible edge states (e.g., $[1,0]$ and $[0,1]$), corresponding to the absence and presence of an edge, respectively. This transition matrix is parameterized to reflect the marginal distributions of edge types, thereby ensuring that the corrupted graph maintains statistical consistency with the original bipartite graph structure. Let $\boldsymbol{m} \in \mathbb{R}^{c\times c}$ represents the marginal distribution matrix of edge types in the original bipartite graph $\mathcal{G}_R^0$. The transition matrix is defined as:

$$\boldsymbol{Q}^t = \alpha_R^t \boldsymbol{I} + \beta_R^t \mathbf{1}_c \boldsymbol{m}, \tag{4}$$

where α_R^t and β_R^t are step-dependent weights, $\boldsymbol{I}$ is the identity matrix, $\mathbf{1}_c$ is an all-ones matrix of size $c \times c$, and c represents the number of edge types. We transform the original binary interaction matrix $\mathbf{R}$ of a bipartite graph $\mathcal{G} = (\mathbf{X}, \mathbf{R})$ into one-hot encoded vectors. This results in a new matrix $\overrightarrow{\mathbf{R}} \in \mathbb{R}^{M\times N\times 2}$, where $\overrightarrow{\mathbf{R}}_{u,i} = [1,0]$ indicates the absence of an edge, i.e., $r_{us} = 0$, and $\overrightarrow{\mathbf{R}}_{u,i} = [0,1]$ corresponds to the presence of an edge, i.e., $r_{us} = 1$. Given a bipartite graph $\mathcal{G}_R^0 = \mathcal{G}$, the transition to $\mathcal{G}_R^t$ at time t is defined as:

$$q\left(\overrightarrow{\mathbf{R}}^t \mid \overrightarrow{\mathbf{R}}^{t-1}\right) = \overrightarrow{\mathbf{R}}^{t-1}\boldsymbol{Q}^t, \tag{5}$$

$$\mathcal{G}_R^t = (\boldsymbol{X}^0, \mathbf{R}^t). \tag{6}$$

Using the reparameterization trick and the multiplicativity of $\boldsymbol{Q}^t$, $\overrightarrow{\mathbf{R}}^t$ is derived from $\overrightarrow{\mathbf{R}}^0$ as:

$$q\left(\overrightarrow{\mathbf{R}}^t \mid \overrightarrow{\mathbf{R}}^0\right) = \overrightarrow{\mathbf{R}}^0 \overline{\boldsymbol{Q}}^t, \tag{7}$$

since $(\mathbf{1}_c \boldsymbol{m})^2 = \mathbf{1}_c \boldsymbol{m}$, we have $\overline{\boldsymbol{Q}}^t = \boldsymbol{Q}^1 \cdots \boldsymbol{Q}^t = \overline{\alpha}_R^t \boldsymbol{I} + \overline{\beta}_R^t \mathbf{1}_c \boldsymbol{m}$, where $\overline{\alpha}_R^t = \prod_{t'=1}^t \alpha_R^{t'} \in (0,1)$ and $\overline{\beta}_R^t = 1 - \overline{\alpha}_R^t$, when $T \to \infty$, $\overline{\boldsymbol{Q}}^T$ approaches $\mathbf{1}_c \mathbf{m}$. By defining transition matrices that align with the real data probabilities, we can effectively introduce discrete corruption to the structure of the bipartite graph. The corrupted bipartite graph after T steps is denoted as $\mathcal{G}_R^T$.

2) Continuous Corruption: In addition to modeling topological graph noise, we also model continuous noise at the user feature level, which accounts for numerical corruption in service recommendations. Specifically, given the initial feature $\mathbf{X}^0 \in \mathcal{G}$, the transition to $\mathbf{X}^t$ and $\mathcal{G}_X^t$ at time $t \in (1, \ldots, T)$ are represented as follows:

$$q(\boldsymbol{X}^t \mid \boldsymbol{X}^{t-1}) = \mathcal{N}\left(\boldsymbol{X}^t; \sqrt{1-\beta_X^t}\boldsymbol{X}^{t-1}, \beta_X^t \boldsymbol{I}\right), \tag{8}$$

$$\mathcal{G}_X^t = (\boldsymbol{X}^t, \boldsymbol{R}^0), \tag{9}$$

here, $\beta_X^t \in (0,1)$ controls the scale of Gaussian noise added at step t. Using additivity of independent Gaussian noise and reparameterization trick, $\boldsymbol{X}^t$ is directly derived from $\boldsymbol{X}^0$:

$$q\left(\boldsymbol{X}^t \mid \boldsymbol{X}^0\right) = \mathcal{N}\left(\boldsymbol{X}^t; \sqrt{\overline{\alpha}_X^t}\boldsymbol{X}^0, \left(1-\overline{\alpha}_X^t\right)\boldsymbol{I}\right), \tag{10}$$

where $\alpha_X^t = 1 - \beta_X^t, \overline{\alpha}_X^t = \prod_{t'=1}^t \alpha_X^{t'}$, through reparameterization, we obtain $\boldsymbol{X}^t = \sqrt{\overline{\alpha}_X^t}\boldsymbol{X}^0 + \sqrt{1-\overline{\alpha}_X^t}\boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Here, to regulate the added noise, we adopt a linear noise schedule for $1 - \overline{\alpha}_X^t$:

$$1 - \overline{\alpha}_X^t = S_{CC} \cdot \left[\alpha_{\min} + \frac{t-1}{T-1}\left(\alpha_{\max} - \alpha_{\min}\right)\right], \tag{11}$$

where S_{CC} controls the noise scales, while $\alpha_{\min}$ and $\alpha_{\max}$ represent the lower and upper bounds of the added noise, respectively. If $T \to \infty$, the $\boldsymbol{X}^t$ approaches a standard Gaussian distribution. The corrupted bipartite graph at T step is denoted as $\mathcal{G}_X^T$. By simultaneously modeling topological and numerical corruption as complementary perspectives, our method is able to effectively capture the complex dynamics of user-service interactions in recommender systems.

B. Graph-Based Denoising Network

Previous studies [77], [89] primarily utilized small multi-layer perceptron (MLP) networks for denoising tasks, often overlooking higher-order structural information. This limited their ability to model complex dependencies. To overcome this, we introduce a novel graph-based denoising framework that fully exploits higher-order information within bipartite graphs. Our approach begins with a unified graph construction module that integrates corrupted node features with topological structure. An iterative Graph Convolutional Network (GCN)-based denoiser then progressively removes noise and refines both structural and feature representations. By combining higher-order information with iterative denoising, our framework captures intricate node relationships and delivers superior performance.

1) Unified Graph Construction: In recommender systems, the topological structure of a bipartite graph is influenced by user and service features. Conversely, these features also reveal properties of the graph topology. This bidirectional relationship highlights the necessity of ensuring consistency between feature-level and topological-level attributes within the bipartite graph. To enable robust learning under real-world noise, introducing diversity in the corruption process is critical. Notably, we do not enforce semantic consistency during corruption itself. Instead, an alignment module projects both corrupted views into a shared semantic space. This design allows the forward diffusion process to capture both noise types while preserving semantic coherence for subsequent denoising. Leveraging contrastive learning [71], [72], we design a unified graph construction module that acts as a regularizer, guiding the denoising process to retain meaningful semantics while modeling interaction heterogeneity.

Specifically, we transform $\overrightarrow{\mathbf{R}}^T \in \mathbb{R}^{M\times N\times 2}$ of $\mathcal{G}_R^T$ into $\mathbf{R}^T \in \mathbb{R}^{M\times N}$ by sampling to ensure a diverse graph structure. Subsequently, we process two distinct corrupted views of the graph: $\mathcal{G}_R^T$, which incorporates discrete corruption, and $\mathcal{G}_X^T$, which applies continuous corruption through linear projection layers. This results in two separate sets of user features, $\boldsymbol{X}'$ and $\boldsymbol{X}''$, each capturing unique characteristics associated with their respective types of corruption. We perform

representation learning on $\boldsymbol{X}'$ and $\boldsymbol{X}''$ to aligning user features. The same user in both matrices forms positive pairs $((x'_u \in \boldsymbol{X}', x''_u \in \boldsymbol{X}'') \mid u \in \mathcal{U})$, while different users form negative pairs $((x'_u \in \boldsymbol{X}', x''_v \in \boldsymbol{X}'') \mid u, v \in \mathcal{U})$. The formula in (12) forces positive user pairs' representations to be close.

$$\mathcal{L}_{ugc} = \sum_{u \in \mathcal{U}} -\log \frac{\exp\left(s\left(x'_u, x''_u\right)/\tau\right)}{\sum_{v \in \mathcal{U}} \exp\left(s\left(x'_u, x''_v\right)/\tau\right)}, \tag{12}$$

here, $s(\cdot)$ denotes the similarity between two representations, and τ is the temperature hyperparameter of the Softmax function. The unified user feature matrix $\overline{\mathbf{X}}^T$ is constructed by concatenating the aligned features $\boldsymbol{X}'$ and $\boldsymbol{X}''$ along with a corresponding learnable user embedding. The unified corrupted bipartite graph $\mathcal{G}^T$ is constructed by $\overline{\mathbf{X}}^T$ and the interaction matrix $\mathbf{R}^T$ from $\mathcal{G}_R^T$. This graph and a learnable service embedding $\mathbf{I}$ are subsequently input into a GCN-based denoiser, which applies a reverse process to iteratively generate a clean bipartite graph.

2) GCN-based Denoiser: Modeling graph dependencies requires higher-order information. We propose a GCN-based denoiser to model p_θ and perform denoising, recovering complex user-service relationships. Specifically, the GCN denoiser takes $\mathbf{Z}_T^{(0)} = \text{concat}(\overline{\mathbf{X}}^T, \mathbf{I})$ as the input node representations and $\mathbf{A}^T$ as the graph structure, enabling the model to aggregate information across nodes. Through iterative message passing, the GCN refines the representations of each node layer by layer. The representation of each node at layer l, denoted $\mathbf{Z}_t^{(l)}$, is computed using the propagation rule in (13).

$$\begin{aligned} \mathbf{Z}_t^{(l)} &= F(\mathbf{Z}^{(l-1)}, \mathcal{G}^t) \\ &= (\mathbf{D}^t)^{-\frac{1}{2}} \mathbf{A}^t (\mathbf{D}^t)^{-\frac{1}{2}} \mathbf{Z}_t^{(l-1)}, \end{aligned} \tag{13}$$

here, $\mathbf{A}^T$ and $\mathbf{D}^T$ denote the adjacency matrix and the degree matrix derived from $\mathbf{R}^T$ of $\mathcal{G}_R^T$. After applying multiple layers of graph convolution, we obtain the refined node representations $\mathbf{Z}_t$ for the bipartite graph. We divide $\mathbf{Z}_t$ into P_t and Q_t, the similarity between P_t and Q_t is measured using the cosine similarity metric for loss calculation, which is defined as:

$$\text{CosineSim}(P_t, Q_t) = \frac{P_t \cdot Q_t}{\|P_t\| \|Q_t\|}. \tag{14}$$

3) Reverse Process: In summary, the denoiser ϕ_θ models the reverse process, which progressively predicts the target graph $\widehat{\mathcal{G}}$ from the input graph $\mathcal{G}^T$ at each step. The joint probability $p_\theta(\mathcal{G}^{t-1} \mid \mathcal{G}^t)$ can be decomposed into a product over users and edges as follows:

$$\begin{aligned} p_\theta\left(\mathcal{G}^{t-1} \mid \mathcal{G}^t\right) &= p_\theta\left(\overline{\mathbf{X}}^{t-1} \mid \overline{\mathbf{X}}^t\right) p_\theta\left(\mathbf{R}^{t-1} \mid \mathbf{R}^t\right) \\ &= \prod_{\overline{x} \in \mathcal{U}} p_\theta\left(\overline{x}^{t-1} \mid \overline{x}^t\right) \prod_{r \in \mathcal{E}} p_\theta\left(r^{t-1} \mid r^t\right). \end{aligned} \tag{15}$$

The underlying assumption is that each user's features and corresponding edges evolve independently based on the corrupted graph from the previous time step. This distribution is used to sample $\mathcal{G}^{t-1}$ at each step until $\mathcal{G}$ is generated.

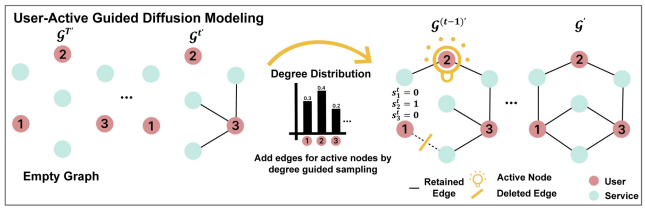

Fig. 3. User-Active Guided Diffusion Modeling. In the inference process, GDMSR iteratively adds edges from $\mathcal{G}^{T'}$ based on the original graph's degree distribution. The probabilities (0.3, 0.4, 0.2) determine if a user node is activated at step t, with only activated nodes' edges retained (e.g., node 2 has a 0.4 probability of retaining its edges).

C. User-Active Guided Generation

Although the training process effectively captures higher-order information, the inference process often demands multiple iterative steps. The method described earlier involves calculating nearly all edges in the interaction matrix, which becomes impractical for large-scale recommendation systems. Inspired by progressive denoising—starting from a noisy state and refining progressively toward the target distribution—we introduce a user-active generation strategy to enhance inference efficiency. Specifically, instead of using the complete bipartite graph throughout the reverse process, we start with an initially empty bipartite graph $\mathcal{G}^{T'}$ and continuously add edges under the guidance of user activity. At each step t, edges are added using the transition matrix $\boldsymbol{Q}^t$, as defined in Section IV-A1, while only the edges connected to activated users are retained. To ensure that the inferred graph gradually approximates the structure of the original graph as the process converges, the active users at step t are determined based on the degree distribution of the original graph $\mathcal{G}$. As illustrated in Fig. 3, for each user $u \in \mathcal{U}$, a binary vector $s_u^t \in 0, 1$ is sampled based on the user's degree d_u^t in the original interaction graph $\mathcal{G}$. This vector serves as an indicator of user activity, specifying which users are active at time step t:

$$s_u^t = \begin{cases} 1 & \text{with probability } \frac{d_u^t}{D} \\ 0 & \text{with probability } 1 - \frac{d_u^t}{D} \end{cases} \tag{16}$$

where $D = \max_{u \in \mathcal{U}} d_u^t$ is the maximum user degree in $\mathcal{G}$. This formulation encourages active users with rich interactions to be selected more often during generation. The interaction matrix $\mathbf{R}^{t'}$, after edge addition, replaces $\mathbf{R}^t$ (as described in Section IV-B) as the input to the GCN-based denoiser to predict the structure at the previous step $t-1$. The process iterates until reaching $t = 1$. With such a strategy, edges are gradually added, preserving the original graph's statistical properties while significantly reducing computational complexity. Consequently, the reverse process in (15) is redefined as follows:

$$\begin{aligned} p_\theta\left(\mathcal{G}^{t-1} \mid \mathcal{G}^t\right) &= p_\theta(\overline{\mathbf{X}}^{t-1} \mid \mathbf{X}^t) \cdot p_\theta(\mathbf{R}^{t-1} \mid \mathbf{R}^t, \mathbf{s}^t) \\ &= \prod_{\overline{x} \in \mathcal{U}} p_\theta\left(\overline{x}^{t-1} \mid \overline{x}^t\right) \cdot \prod_{r \in \mathcal{E}} p_\theta\left(r^{t-1} \mid r^t, \mathbf{s}^t\right), \end{aligned} \tag{17}$$

This progressive, activity-aware generation strategy not only improves computational efficiency by sampling active users but also enhances the structural consistency of the generated graph. As the process unfolds, the strategy ensures that the

Algorithm 1: GDMSR Training.

Input: User-service interaction bipartate graph $\mathcal{G} = (\mathbf{X}, \mathbf{R})$ and randomly initialized θ, diffusion steps T

repeat
  Sample $t \sim \mathcal{U}(1, \ldots, T)$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and $\mathbf{Q}^t$
  Sample $\mathcal{G}_R^t$ and $\mathcal{G}_X^t$ given $\mathcal{G}$, t, $\boldsymbol{\epsilon}$, and $\mathbf{Q}^t$ via (6) and (9)
  Compute $\widehat{\mathcal{G}}$ through ϕ_θ via (13) and (14)
  Compute $\mathcal{L}_{\text{ugc}}$ using (12) and $\mathcal{L}_{\text{diff}}$ using (19)
  Take gradient descent on $\nabla_\theta(\lambda_1 \mathcal{L}_{\text{ugc}} + \mathcal{L}_{\text{diff}})$ to optimize θ
until convergence

Output: Optimized θ

TABLE I
DESCRIPTIVE STATISTICS OF THE DATASETS

Dataset	#Users	#Services	#Interaction	Density
Amazon-game	2,343	1,700	39,263	0.99%
ML-1M	5,949	2,810	571,531	3.42%
Citeulike-t	7,947	25,975	132,275	0.06%
Yelp	54,574	34,395	1,402,736	0.07%
Amazon-book	108,822	94,949	3,146,256	0.03%

evolving graph remains statistically aligned with the original graph, facilitating more accurate and efficient denoising.

D. Optimization

We optimize the denoiser model ϕ_θ by minimizing diffusion loss $\mathcal{L}_{\text{diff}} = \sum_{t=1}^{T} \mathcal{L}_t$, and the calculation of $\mathcal{L}_t$ is as follows:

$$\mathcal{L}_t = \mathbb{E}_{\boldsymbol{q}(\mathcal{G}^t|\mathcal{G})}(\|\widehat{\mathcal{G}} - \mathcal{G}\|_2^2), \tag{18}$$

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{t\sim\mathcal{U}(1,T)}\mathcal{L}_t, \tag{19}$$

which regulates the predicted $\widehat{\mathcal{G}}$ to approximate $\mathcal{G}$. In practical, we uniformly sample step t at each training iteration to optimize $\mathcal{L}_{\text{diff}}$ over $t \sim \mathcal{U}(1, \ldots, T)$ ((19)). The procedure is detailed in Algorithm 1. The loss function consists of the unified graph construction loss $\mathcal{L}_{\text{ugc}}$ and the diffusion loss $\mathcal{L}_{\text{diff}}$:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\text{ugc}} + \mathcal{L}_{\text{diff}}, \tag{20}$$

where the hyperparameter λ_1 serves as a weighting factor to balance the contributions of these two objectives.

E. Inference

During the inference phase, the GDMSR takes the original bipartite graph $\mathcal{G}$ as input, which is corrupted by the addition of discrete and continuous noise, and outputs a clean bipartite graph. Specifically, GDMSR introduces noise into the original graph $\mathcal{G}$, resulting in the corrupted graphs $\mathcal{G}_R^T$ and $\mathcal{G}_X^T$. The corrupted graph $\mathcal{G}^T$, with aligned user features $\overline{\mathbf{X}}^T$ and $\mathbf{R}^{T'}$ generated through user-active guided generation, is then fed into a GCN-based denoiser for further generation. This network progressively denoises the graph, step by step, to predict the clean graph $\widehat{\mathcal{G}}$. Finally, the recovered graph $\widehat{\mathcal{G}}$ is utilized for service ranking, as detailed in Algorithm 2.

TABLE II
DESCRIPTIVE STATISTICS OF THE PROGRAMMABLE WEB DATASET

Description	Value
Number of mashups	7,974
Mashups invoking only two services	3,194
Number of services	24,034
Average number of services in one mashup	2.09
Number of services used in at least one mashup	1,154
Sparsity of mashup-service matrix	0.274%

Algorithm 2: GDMSR Inference.

Input: User-service interaction bipartite graph $\mathcal{G} = (\mathbf{X}, \mathbf{R})$, parameters θ, diffusion steps T

Initialize an empty graph $\mathcal{G}^{T'}$ with $\mathbf{R}^{T'}$
for $t = T$ to 1 **do**
  Sample $\mathcal{G}_R^t$ and $\mathcal{G}_X^t$ given $\mathcal{G}$, t, $\boldsymbol{\epsilon}$, and $\mathbf{Q}^t$ via (6) and (9)
  $\mathbf{R}^{(t-1)'} = \text{User-active}(\mathcal{G}^{t'})$ according to Section IV-C
  Compute $\widehat{\mathcal{G}}^{t-1}$ through ϕ_θ with $\mathbf{R}^{(t-1)'}$, $\mathcal{G}_R^t$ and $\mathcal{G}_X^t$
end for

Output: Generated $\widehat{\mathcal{G}}$

V. EXPERIMENTS

A. Dataset Description

We conduct experiments on six widely-used real-world datasets: 1) ML-1 M[1] (movie ratings), 2) Yelp[2] (a service rating dataset where users share reviews and ratings), 3) Amazon-Book[3] and 4) Amazon-Game[4] (book and game reviews from Amazon) and 5) Citeulike-t[5] (a dataset collected from CiteULike and Google Scholar) and 6) Programmable Web[6] (the largest online API registry containing a wide range of Web APIs and mashups). Following previous methods [77], [84], for the first five datasets, we sort historical interactions by timestamp, remove users with fewer than four interactions, and split the datasets into training, validation, and test sets with ratios of 7:1:2 (ML-1 M, Yelp, Amazon-Book) and 8:1:1 (Amazon-Game, Citeulike-t). Descriptive statistics for these five datasets are shown in Table I. For the ProgrammableWeb dataset, we collected Web API and mashup records from ProgrammableWeb.com covering the period from September 2005 to November 2020 [16]. We removed unused APIs that were never included in any mashup, and retained only mashups that consist of at least two APIs. For evaluation, records from July 2010 to September 2020 were used as the test set. Descriptive statistics for this dataset are shown in Table II.

B. Experimental Setup

1) Evaluation Metrics: We use three evaluation metrics: 1) $Recall@K$ ($R@K$), measuring the proportion of relevant services in the top K recommendations. In (21), $R(u)$ represents

[1] [Online]. Available: https://grouplens.org/datasets/movielens/1~m/
[2] [Online]. Available: https://www.yelp.com/dataset/
[3] [Online]. Available: https://jmcauley.ucsd.edu/data/amazon/
[4] [Online]. Available: https://cseweb.ucsd.edu/~jmcauley/datasets/amazon_v2/
[5] [Online]. Available: https://github.com/js05212/citeulike-t
[6] [Online]. Available: https://www.programmableweb.com/

the set of top K recommendations for user u, while $T(u)$ is services that user u is interested in. 2)$HR@K$ ($H@K$),which evaluates whether the target service appears within the top-K positions of the recommendation list, thereby reflecting whether the user's actual requirement is successfully captured by the system. The metric is calculated as shown in (22), where $|y_u^{test}|$ denotes the number of services used by user u in the test set. 3) Normalized Discounted Cumulative Gain $NDCG@K$ ($N@K$), which considers the presence and position of relevant services. Eq. (23) shows the calculation, where rel_i is the relevance score, DCG is the ranking, and $iDCG$ is for normalization.

$$Recall@K = \frac{\sum_{u\in\mathcal{U}} |R(u) \cap T(u)|}{\sum_{u\in\mathcal{U}} |T(u)|} \tag{21}$$

$$HR@K = \frac{\sum_{i=1}^{K} rel_i}{|y_u^{test}|} \tag{22}$$

$$NDCG@K = \frac{DCG@K}{iDCG}, \quad \text{with } \begin{cases} DCG@K = \sum_{i=1}^{K} \frac{rel_i - 1}{\log_2(i+1)} \\ iDCG = \max_{\text{ranking}} DCG \end{cases} \tag{23}$$

2) Compared Methods: To demonstrate the effectiveness of GDMSR, we conducted a comparative analysis on six datasets against eighteen methods, grouped as follows: 1) classical matrix factorization methods like MF [74], NCF [7]; 2) GCN-based methods LightGCN [75], CoACN [16]; 3) text-based methods DeepCoNN [18], GSR [19]; 4) generative autoencoder methods, including MultiDAE [80], CDAE [81], Ease [82], ConVAE [83] and MultiVAE [80]; 5) contrastive learning methods like LightGCL [79] and NFGCL [78]; 6) diffusion generative methods like CODIGEM [76], MultiDAE++ [80], DiffRec [77], CDiff4Rec [84] and CF-Diff [91].

- **MF** [74] is a classical matrix factorization-based collaborative filtering method.
- **NCF** [7] is a neural network-based collaborative filtering model that enhances the non-linear modeling capacity of traditional matrix factorization by introducing a multilayer perceptron (MLP) architecture.
- **LightGCN** [75] is a lightweight graph convolutional network that generates node representations by aggregating information from neighboring nodes.
- **CoACN** [16] incorporates service functional domain information by designing a functional domain-level attention unit, which enhances the representation learning of both users and services.
- **DeepCoNN** [18] is a deep learning model that jointly learns user and service representations from textual information to perform service recommendation tasks.
- **GSR** [19] is a service recommendation approach that automatically acquires services based on user requirements. It employs reinforcement learning to guide the recommendation process and utilizes BERT to retrieve semantically relevant user requirements.
- **MultiDAE** [80] employs dropout to investigate a denoising autoencoder with a multinomial likelihood function.
- **CDAE** [81] trains an autoencoder to recover interactions that have been randomly corrupted.
- **Ease** [82] defines a linear model that is geared toward sparse data using neighborhood information.
- **ConVAE** [83] is a conditioned variational autoencoder for constrained top-N service recommendation where the recommended services must satisfy a given condition.
- **MultiVAE** [80] employs variational autoencoders (VAEs) to recover interaction and uses Bayesian inference for parameter estimation.
- **LightGCL** [79] utilizes singular value decomposition for contrastive augmentation, enabling unconstrained structural refinement and capturing global collaborative relations more effectively.
- **NFGCL** [78] achieves high-quality representations without relying on negative sampling or graph augmentation.
- **CODIGEM** [76] models the diffusion process using multiple autoencoders (AEs), but only utilizes the first AE to predict interactions.
- **MultiDAE++** [80] incrementally adds Gaussian noise to interaction data and trains a MultiDAE to recover the interactions in a single step.
- **DiffRec** [77] applies Gaussian noise to each of the user's interactions and then reverses the process step by step.
- **CDiff4Rec** [84] generates pseudo-users from service features and leverages collaborative signals from both real and pseudo personalized neighbors identified through behavioral similarity.
- **CF-Diff** [91] is capable of making full use of collaborative signals along with multi-hop neighbors by a cross-attention-guided multi-hop autoencoder.

3) Hyper-Parameter Settings: We select hyperparameters based on the highest $NDCG@20$ on the test set. Learning rates are tuned from [0.0002, 0.001, 0.005, 0.01]. The batch size is fixed at 400, with latent embedding dimensionality and λ_1 set to 1000 and 0.1, respectively. The model uses a step embedding size of 10 and a 2-layer GCN. The diffusion step t is chosen from [2, 5, 10, 20, 50, 100, 500], with $t = 5$ performing best. The continuous noise scale is set within [0.00001, 0.25] (optimal: 0.1), while the discrete noise scale is selected from [0.0010, 0.0008, 0.0007, 0.0006, 0.0005, 0.0003] (optimal: 0.0008). The range of β_X^t in (11) is set to $\alpha_{\min} = 0.001$ and $\alpha_{\max} = 0.01$ via grid search on validation NDCG. The learning rate is 0.00001 with a time step of 5. All baseline models use official source codes. For fair comparison, we maintain consistent learning rates and batch sizes with GDMSR, while baseline-specific hyperparameters follow the original papers. All experiments are conducted on an NVIDIA GeForce GTX 3090 GPU with PyTorch.

For the Programmable Web dataset, we maintain consistency with baselines by splitting the Mashup training set into a training and a validation set based on interaction history for GDMSR training. During testing, for each Mashup in the test set, we identify the Mashup with the most similar textual description embedding from the training set. The interactions of this corresponding Mashup are then used in the diffusion process. Additionally, the textual description embedding of the Mashup is introduced as the initialization for the Mashup embedding in the GCN denoising process, which is then used to predict the APIs that the Mashup is likely to call.

C. Performance Comparisons

We summarized the performance of various methods on six datasets in terms of $HR@K$, $Recall@K$ and $NDCG@K$ ($K = 10, 20$). As shown in Tables III, IV, and V, GDMSR outperforms

TABLE III
PERFORMANCE COMPARISON OF THE GDMSR FRAMEWORK AND OTHER BASELINES ON THREE DATASETS. % IMPROVE. INDICATES THE RELATIVE IMPROVEMENT OF GDMSR OVER THE BEST BASELINE RESULTS.

Model	ML-1M				Yelp				Amazon-book			
	$R@10\uparrow$	$R@20\uparrow$	$N@10\uparrow$	$N@20\uparrow$	$R@10\uparrow$	$R@20\uparrow$	$N@10\uparrow$	$N@20\uparrow$	$R@10\uparrow$	$R@20\uparrow$	$N@10\uparrow$	$N@20\uparrow$
MF	0.0876	0.1503	0.0749	0.0966	0.0341	0.0560	0.0210	0.0276	0.0437	0.0689	0.0264	0.0339
LightGCN	0.0987	0.1707	0.0833	0.1083	0.0540	0.0904	0.0325	0.0436	0.0534	0.0822	0.0325	0.0411
MultiDAE	0.0995	0.1753	0.0803	0.1067	0.0522	0.0864	0.0316	0.0419	0.0571	0.0855	0.0357	0.0442
CDAE	0.0991	0.1705	0.0829	0.1078	0.0444	0.0703	0.0280	0.0360	0.0538	0.0737	0.0361	0.0422
MultiVAE	0.1007	0.1726	0.0825	0.1076	0.0567	0.0945	0.0344	0.0458	0.0628	0.0935	0.0393	0.0485
CODIGEM	0.0972	0.1699	0.0837	0.1087	0.0470	0.0775	0.0292	0.0385	0.0300	0.0478	0.0192	0.0245
MultiDAE++	0.1009	0.1771	0.0815	0.1079	0.0544	0.0909	0.0328	0.0438	0.0580	0.0864	0.0363	0.0448
DiffRec	0.1058	0.1787	0.0901	0.1148	0.0581	0.0960	0.0363	0.0478	0.0695	0.1010	0.0451	0.0547
CF-Diff	0.1077	0.1843	0.0912	0.1176	0.0585	0.0962	0.0368	0.0480	0.0499	0.0717	0.0337	0.0404
GDMSR	0.1078	0.1861	0.0916	0.1178	0.0634	0.1044	0.0392	0.0515	0.0916	0.1315	0.0587	0.0707
% Improve.	0.09%	0.98%	0.44%	0.17%	8.38%	8.52%	6.52%	7.29%	31.80%	30.20%	30.16%	29.25%

TABLE IV
PERFORMANCE COMPARISON OF THE GDMSR FRAMEWORK AND OTHER BASELINES ON TWO DATASETS. % IMPROVE. INDICATES THE RELATIVE IMPROVEMENT OF GDMSR OVER THE BEST BASELINE RESULTS.

Model	Amazon-game		Citeulike-t	
	R@20	N@20	R@20	N@20
MF	0.1839	0.0844	0.1535	0.0792
LightGCN	0.1899	0.0854	0.1574	0.0800
Ease	0.2108	0.1012	0.1591	0.0852
LightGCL	0.2126	0.0890	0.1563	0.0829
NFGCL	0.2173	0.0951	0.1587	0.0844
MultiVAE	0.2181	0.0995	0.1447	0.0770
ConVAE	0.2192	0.0997	0.1477	0.0775
DiffRec	0.2193	0.1034	0.1491	0.0791
CDiff4Rec	0.2255	0.1052	0.1616	0.0885
GDMSR	0.2547	0.1204	0.1788	0.0940
% Improve.	12.95%	14.45%	10.64%	6.21%

TABLE V
PERFORMANCE COMPARISON OF THE GDMSR FRAMEWORK AND OTHER BASELINES ON PROGRAMMABLE WEB DATASET. % IMPROVE. INDICATES THE RELATIVE IMPROVEMENT OF GDMSR OVER THE BEST BASELINE RESULTS.

Model	H@10	N@10	H@20	N@20
MF	0.3721	0.4797	0.4686	0.5238
NCF	0.4129	0.5141	0.4878	0.5511
DeepCoNN	0.4023	0.5120	0.5096	0.5689
LightGCN	0.4396	0.5396	0.5006	0.5792
CoACN	0.4828	0.6050	0.5447	0.6368
GSR	0.5079	0.6284	0.5836	0.6693
GDMSR	0.5320	0.6439	0.6190	0.6977
% Improve.	4.74%	2.47%	6.07%	4.24%

all the baselines across all metrics. Additionally, we have the following observations:

- The matrix factorization method (MF) demonstrates the weakest performance among baselines except on Citeulike-t. This is primarily because MF uses explicit user interactions, overlooks implicit user preference information, and fails to capture higher-order relationships. The superior performance of NCF over MF demonstrates that increasing the model's non-linearity and complexity can enhance the effectiveness of service recommendations based on Mashup creation.
- Enhancing the understanding of user requirement descriptions and service descriptions can effectively improve service recommendation performance in Mashup creation scenarios. Compared with MF, which only employs simple keyword matching, models such as DeepCoNN, GSR, and CoACN can more effectively capture contextual information by leveraging rich textual features.
- Most generative models outperform discriminative models (MF and LightGCN) due to their superior capacity to model the joint distribution of complex latent factors.
- Among generative baselines, CF-Diff, CDiff4Rec and DiffRec outperform others, highlighting the effectiveness of the diffusion process and the step-by-step denoising approach for recommendation tasks. CDiff4Rec achieves the best performance by effectively leveraging service-side information to capture user preferences, though it lacks explicit modeling of higher-order collaborative signals among users. CODIGEM performs worse, likely due to its sole reliance on the first AE for inference.
- Compared to MF and LightGCN, contrastive learning approaches such as LightGCL and NFGCL achieve stronger performance, likely due to learning more discriminative, robust user-service representations through objectives that explicitly enforce alignment and uniformity.
- GDMSR achieves state-of-the-art performance on all datasets by effectively capturing higher-order collaborative signals between users and services during denoising, enabling better generative service recommendations.
- User interaction data in real-world service recommender systems often provides only the final click results, while the underlying decision-making process remains highly complex and influenced by a diverse set of factors. GDMSR models the heterogeneous noise present in real-world service recommendation scenarios by explicitly capturing the inherent noise embedded within user interaction data and improves the overall service recommendation performance.
- On large-scale datasets, GDMSR demonstrates a more pronounced advantage. This is because traditional generative methods primarily focus on element-wise user interaction generation, making it difficult to capture the complex interdependencies among nodes in large bipartite graphs. In

TABLE VI
INFLUENCE OF MULTI-LEVEL CORRUPTION AND CONTRASTIVE LEARNING ON YELP

Model	$R@10$	$R@20$	$N@10$	$N@20$
GDMSR$_{CC}$	0.0571	0.0926	0.0355	0.0462
GDMSR$_{DC}$	0.0578	0.0952	0.0351	0.0464
GDMSR$_{NC}$	0.0548	0.0893	0.0333	0.0438
GDMSR$_{NUC}$	0.0626	0.1032	0.0386	0.0513
GDMSR	0.0634	0.1044	0.0392	0.0515

contrast, GDMSR works at the full-graph level, making it better suited for large-scale scenarios.

D. Ablation Studies

1) Efficiency of Multi-Level Corruption and Contrastive Learning: To investigate the influence of multi-level corruption, we considered several variants of GDMSR:

- *GDMSR$_{CC}$:* In the Multi-level Corruption stage, we retained only continuous corruption.
- *GDMSR$_{DC}$:* In the Multi-level Corruption stage, we retained only discrete corruption.
- *GDMSR$_{NC}$:* We removed the diffusion process, directly inputting the original topology graph into the GCNs.
- *GDMSR$_{NUC}$:* We removed the contrastive learning during the unified graph construction, directly concatenated the two views after corruption.

Table VI shows the impact of different corruptions on Yelp. Removing either discrete corruption (GDMSR$_{CC}$) or continuous corruption (GDMSR$_{DC}$) leads to a decline in performance and training instability. Because diverse noise in service recommendation scenarios requires multiple diffusion levels to fully capture it, and using just one makes the model less accurate. Without the diffusion process, GDMSR reduces to a model that inputs the original graph directly into the GCNs, omitting the critical step-by-step denoising process (GDMSR$_{NC}$). GDMSR operates as a Markov process with graph edits, is permutation equivariance, and provides an evidence lower bound for likelihood estimation. These properties likely explain its superior performance compared to GDMSR$_{NC}$. Moreover, removing contrastive learning in unified graph construction (GDMSR$_{NUC}$) weakens semantic alignment between two corrupted views, which degrades performance. This component ensures the denoising process effectively captures and models interaction heterogeneity.

2) Efficiency of User-Active Guided Generation: We sampled 3,000 users and their associated services, creating a subdataset from the Yelp dataset, referred to as Yelp-s. This subgraph was utilized to assess the impact of GDMSR and its variant, GDMSR$_{NUA}$, on computational memory efficiency. In GDMSR$_{NUA}$, the user-active guided strategy is canceled, meaning that during inference, the absence of user-active nodes diminishes the differentiation in the significance of predicted edges. These redundant edges not only result in a loss of topological information but also introduce additional computational overhead. This could explain why GDMSR outperforms GDMSR$_{NUA}$, as demonstrated in Table VII. We further compare GDMSR with DiffRec, a strong diffusion-based baseline. Although GDMSR consumes more computational resources due to its enhanced modeling components (e.g., noise-aware diffusion and graph propagation), its cost remains comparable to DiffRec, which relies solely on an MLP for denoising. Crucially, this trade-off is justified by GDMSR's superior recommendation performance.

TABLE VII
INFLUENCE OF USER-ACTIVE GUIDED GENERATION STRATEGY

Model	$R@10$	$R@20$	$N@10$	$N@20$
GDMSR$_{NUA}$	0.0259	0.0420	0.0205	0.0261
GDMSR	**0.0267**	**0.0427**	**0.0208**	**0.0264**

Model	Inference	
	Speed (s)	Memory (MiB)
DiffRec	**89**	**3758**
GDMSR$_{NUA}$	204	12256
GDMSR	**159**	**7486**

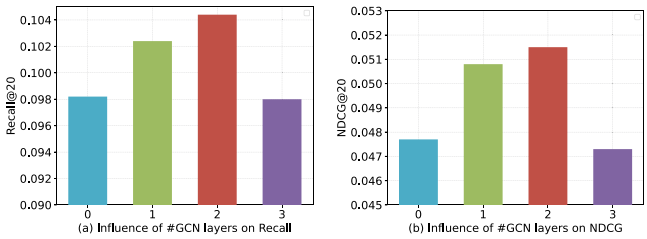

Fig. 4. Influence of the number of GCN layers l.

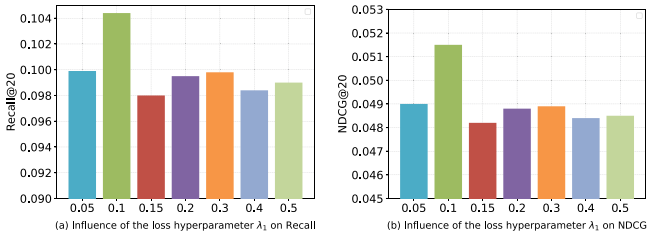

Fig. 5. Influence of the loss hyperparameter λ_1.

E. Parameter Sensitivity Analysis

1) GCN Layers: We investigated the influence of the number of GCN layers, denoted as l, on performance. When $l = 0$, the GCN reduces to a multilayer perceptron (MLP), which is inadequate for processing complex graph information, resulting in suboptimal performance. As the number of GCN layers increases, performance initially improves but subsequently declines, indicating that excessive higher-order information may introduce unnecessary noise. Detailed results are presented in Fig. 4.

2) Sensitivity Analysis of λ_1: To explore the sensitivity of GDMSR to the hyperparameter λ_1, we conducted a series of experiments varying λ_1. The results indicate that the service recommendation performance is optimal when $\lambda_1 = 0.1$. As λ_1 increases, neither Recall nor NDCG exhibits any significant improvement. Detailed results are presented in Fig. 5.

3) Noise Scale: We evaluate the impact of hyperparameters S_{CC} in (11) and S_{DC} parameterizing $\boldsymbol{m}$ in (4) on service recommendation performance. As shown in Fig. 6, GDMSR achieves optimal performance at $S_{CC} = 0.1$ and $S_{DC} = 0.0008$. However, performance declines with increasing S_{CC}. An appropriate S_{DC} must align with the original graph's topology, as excessively large or small values can disrupt the contained information.

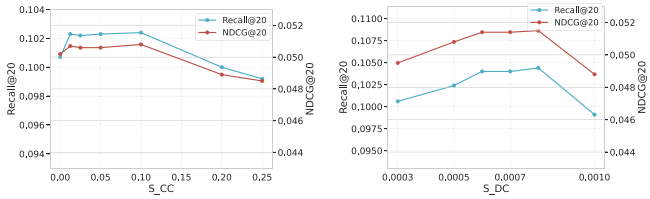

Fig. 6. Influence of the noise scale S_{CC} and S_{DC}.

TABLE VIII
EVALUATION ON LONG-TAIL DISTRIBUTION

Methods	R@10	R@20	N@10	N@20
Average	0.1050	0.1795	0.0867	0.1127
GDMSR	**0.1120**	**0.1797**	**0.0920**	**0.1164**

F. Additional Analysis

1) Evaluation on Long-Tail Distribution: In user-active guided generation, GDMSR incrementally adds edges to an initially empty graph based on probability, ensuring a diverse graph structure. Unlike fixed graph architectures where highly popular users tend to dominate, this approach provides less popular (long-tail) users with a probability of being activated, thereby increasing their exposure. This probabilistic sampling enhances the model's robustness and helps alleviate potential popularity bias. We conducted experiments on the ML-1 M dataset, focusing on the long-tail distribution. We identified the bottom 20% of users based on their interaction frequency and compared the performance of our user-active guided strategy against that of a baseline employing an average probability-based strategy for this subgroup. The results in Table VIII illustrate that GDMSR effectively reduces popularity bias and offers better recommendations for long-tail users.

2) Qualitative Analysis: To further illustrate the superiority of GDMSR, we conducted a case study on the ML-1 M dataset. Specifically, we examined two users, user 1 and user 2093, who interacted with the same four movies: 211, 189, 261, and 823. GDMCF successfully predicted user 1's interactions, while the baseline method, DiffRec, failed to predict the interaction with movie 261. This highlights the advantage of incorporating higher-order information in GDMSR.

3) Analysis of Model Complexity: To assess GDMSR's scalability, we analyzed its time and space complexity. The time complexity is $O(T(MN + M^2d + L|E|d))$, dominated by graph convolution over the interaction graph, where M and N are user and service counts, T is diffusion steps, L is GCN layers, d is feature dimension, and $|E|$ is edge count. Space complexity is $O(MN + (M + N)d + L(M + N)d)$, suitable for large-scale use. Per-epoch training times are 21 s (Yelp), 2 s (Citeulike-t), and 169 s (Amazon-game), demonstrating its efficiency in real-world scenarios.

4) Feature Alignment Visualizition: To verify the efficiency of the unified graph construction module that aligns the feature-level and topological-level attributes within the bipartite graph, we visualized the unaligned features $\boldsymbol{X}'$ and $\boldsymbol{X}''$, as well as the aligned features $\boldsymbol{X}'$ and $\boldsymbol{X}''$ on Yelp. Fig. 7 demonstrates that, after alignment, the features of the aligned users became significantly closer.

5) Performance by User Activity Groups: To evaluate GDMSR across user activity levels, we divided Yelp users into five equal groups by interaction frequency (Table IX). Recall metrics consistently improve as activity decreases from Group 1 (Most Active) to Group 5 (Long-tail), demonstrating GDMSR's particular effectiveness for users with sparse data.

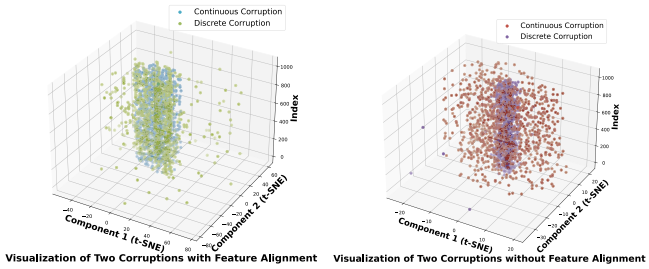

Fig. 7. Feature alignment visualizition of two corruptions.

TABLE IX
PERFORMANCE BY USER ACTIVITY GROUPS

User Activity Group	R@10	R@20	N@10	N@20
20%–40%	0.0584	0.0996	0.0378	0.0512
40%–60%	0.0666	0.1074	0.0382	0.0499
60%–80%	0.0698	0.1161	0.0378	0.0500
Bottom 20%	0.0717	0.1137	0.0366	0.0471

VI. CONCLUSION AND FUTURE WORK

Generative models outperform discriminative models by capturing the joint distribution of latent factors, and diffusion-based service recommendation methods have shown remarkable results. However, existing approaches operate primarily at the element-wise level, overlooking higher-order collaborative signals. To address this, we propose GDMSR, which captures higher-order collaborative signals at the graph level, thereby improving graph-based diffusion learning. To tackle noise heterogeneity, we employ multi-level corruption and align the corrupted features into a unified space for graph-based denoising. Additionally, we reduce inference costs by retaining only the edges associated with activated users in the bipartite graph. Extensive experiments on six datasets demonstrate that GDMSR consistently outperforms competing methods in both effectiveness and efficiency. In the future, we plan to develop more scalable training strategies for high-noise service environments and large-scale scenarios. Moreover, we aim to design user activity–aware mechanisms to dynamically identify active users, enhancing the model's ability to evolve via adaptive graph structures.

REFERENCES

[1] Y. Zhong, Y. Fan, and K. Huang and W. Tan, and J. Zhang, "Time-aware service recommendation for mashup creation," *IEEE Trans. Serv. Comput.*, vol. 8," no. 3, pp. 356–368, May/Jun. 2015.
[2] H. Lin, Y. Fan, J. Zhang, B. Bai, Z. Xu, and T. Lukasiewicz, "Toward knowledge as a service (KaaS): Predicting popularity of knowledge services leveraging graph neural networks," *IEEE Trans. Serv. Comput*, vol. 16, no. 1, pp. 642–655, Jan./Feb. 2023.
[3] C. Wu et al., "FeedRec: News feed recommendation with various user feedbacks," in *Proc. World Wide Web Conf.*, 2022, pp. 2088–2097.
[4] A. G.-Messica and L. Rokach, "Personal price aware multi-seller recommender system: Evidence from eBay," *Knowl. Based Syst.*, vol. 150, pp. 14–26, 2018.

[5] L. Chen, L. Wu, R. Hong, K. Zhang, and M. Wang, "Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 27–34.

[6] G. Zhou et al., "Deep interest network for click-through rate prediction," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 1059–1068.

[7] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. World Wide Web Conf.*, 2017, pp. 173–182.

[8] I. J. Goodfellow et al., "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[9] P. D. Kingma and M. Welling, "Auto-encoding variational bayes," in *Proc. Int. Conf. Learn. Representations*, 2014, pp. 1–14.

[10] H. Bharadhwaj, H. Park, and B. Y. Lim, "RecGAN: Recurrent generative adversarial networks for recommendation systems," in *Proc. ACM Conf. Recommender Syst.*, 2018, pp. 372–376.

[11] X. Chen, S. Li, H. Li, S. Jiang, Y. Qi, and L. Song, "Generative adversarial user model for reinforcement learning based recommendation system," in *Proc. Int. Conf. Mach. Learn.*, vol. 97, 2019, pp. 1052–1061.

[12] T. T. Nguyen et al., "Poisoning GNN-based recommender systems with generative surrogate-based attacks," *ACM Trans. Inf. Syst.*, vol. 41, no. 3, 2023, Art. no. 58.

[13] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 6840–6851.

[14] L. Wu, P. Sun, Y. Fu, R. Hong, X. Wang, and M. Wang, "A neural influence diffusion model for social recommendation," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 235–244.

[15] Z. Li, A. Sun, and C. Li, "DiffuRec: A diffusion model for sequential recommendation," *ACM Trans. Inf. Syst.*, vol. 42, no. 3, 2024, Art. no. 66.

[16] R. Yan, Y. Fan, J. Zhang, J. Zhang, and H. Lin, "Service recommendation for composition creation based on collaborative attention convolutional network," in *Proc. IEEE Int. Conf. Web Serv.*, 2021, pp. 397–405.

[17] Q. Liu et al., "Diffusion augmentation for sequential recommendation," in *Proc. Conf. Inf. Knowl. Manage.*, 2023, pp. 1576–1586.

[18] L. Zheng, V. Noroozi, and P. S. Yu, "Joint deep modeling of users and items using reviews for recommendation," in *Proc. ACM Int. Conf. Web Search Data Mining*, 2017, pp. 425–434.

[19] G. Fan et al., "Service recommendations for mashup based on generation model," *IEEE Trans. Serv. Comput.*, vol. 17, no. 4, pp. 1820–1834, Jul./Aug. 2024.

[20] J. Choi, J. Lee, C. Shin, S. Kim, H. Kim, and S. Yoon, "Perception prioritized training of diffusion models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 11462–11471.

[21] X. Han, H. Zheng, and M. Zhou, "CARD: Classification and regression diffusion models," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 18100–18115.

[22] Y. Qin, H. Wu, W. Ju, X. Luo, and M. Zhang, "A diffusion model for POI recommendation," *ACM Trans. Inf. Syst.*, vol. 42, no. 2, 2024, Art. no. 54.

[23] H. Ma et al., "Plug-in diffusion model for sequential recommendation," in *Proc. AAAI Conf. Artif. Intell.*, 2024, pp. 8886–8894.

[24] Y. Yang et al., "A survey on diffusion models for time series and spatio-temporal data," *ACM Comput. Surveys*, vol. 58, no. 3, 2025, Art. no. 3783986.

[25] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 165–174.

[26] A. Pirotte, J.-M. Renders, and M. Saerens, "Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 3, pp. 355–369, Mar. 2007.

[27] S. Lee, S.-I. Song, M. Kahng, D. Lee, and S.-G. Lee, "Random walk based entity ranking on graph for multidimensional recommendation," in *Proc. 5th ACM Conf. Recommender Syst.*, 2011, pp. 93–100.

[28] Z. Zhang and B. Wang, "Graph neighborhood routing and random walk for session-based recommendation," in *Proc. Int. Conf. Des. Mater.*, 2021, pp. 1517–1522.

[29] M. Choi, J. Kim, J. Lee, H. Shim, and J. Lee, "S-Walk: Accurate and scalable session-based recommendation with random walks," in *Proc. ACM Int. Conf. Web Search Data Mining*, 2022, pp. 150–160.

[30] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 974–983.

[31] H. Wang, M. Zhao, X. Xie, W. Li, and M. Guo, "Knowledge graph convolutional networks for recommender systems," in *Proc. World Wide Web Conf.*, 2019, pp. 3307–3313.

[32] A. Hyvärinen, "Estimation of non-normalized statistical models by score matching," *J. Mach. Learn. Res.*, vol. 6, pp. 695–709, 2005.

[33] P. Vincent, "A connection between score matching and denoising autoencoders," *Neural Comput.*, vol. 23, no. 7, pp. 1661–1674, 2011.

[34] Z. Kadkhodaie and E. P. Simoncelli, "Stochastic solutions for linear inverse problems using the prior implicit in a denoiser," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 13242–13254.

[35] R. M. Neal, "MCMC using hamiltonian dynamics," in *Handbook of Markov Chain Monte Carlo*. Boca Raton, FL, USA: Chapman Hall/CRC, 2011, pp. 113–162.

[36] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient Langevin dynamics," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 681–688.

[37] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 2256–2265.

[38] F. Liu, Z. Cheng, L. Zhu, Z. Gao, and L. Nie, "Interest-aware message-passing GCN for recommendation," in *Proc. World Wide Web Conf.*, 2021, pp. 1296–1305.

[39] Y. Wei, X. Wang, L. Nie, S. Li, D. Wang, and T.-S. Chua, "Causal inference for knowledge graph based recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 11, pp. 11153–11164, Nov. 2023.

[40] H. Liu, J. Wen, L. Jing, and J. Yu, "Deep generative ranking for personalized recommendation," in *Proc. 13th ACM Conf. Recommender Syst.*, 2019, pp. 34–42.

[41] F. Yuan, A. Karatzoglou, I. Arapakis, J. M. Jose, and X. He, "A simple convolutional generative network for next item recommendation," in *Proc. ACM Int. Conf. Web Search Data Mining*, 2019, pp. 582–590.

[42] P. Nema, A. Karatzoglou, and F. Radlinski, "Disentangling preference representations for recommendation critiquing with ß-VAE," in *Proc. Conf. Inf. Knowl. Manage.*, 2021, pp. 1356–1365.

[43] K. Luo, H. Yang, G. Wu, and S. Sanner, "Deep critiquing for VAE-based recommender systems," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2020, pp. 1269–1278.

[44] I. Shenbin, A. Alekseev, E. Tutubalina, V. Malykh, and S. I. Nikolenko, "RecVAE: A new variational autoencoder for top-N recommendations with implicit feedback," in *Proc. ACM Int. Conf. Web Search Data Mining*, 2020, pp. 528–536.

[45] W. Wang, X. Lin, F. Feng, X. He, M. Lin, and T.-S. Chua, "Causal representation learning for out-of-distribution recommendation," in *Proc. World Wide Web Conf.*, 2022, pp. 3562–3571.

[46] I. J. Goodfellow et al., "Generative adversarial nets," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[47] D. Yang, Z. Guo, Z. Wang, J. Jiang, Y. Xiao, and W. Wang, "A knowledge-enhanced deep recommendation framework incorporating GAN-based models," in *Proc. Int. Conf. Des. Mater.*, 2018, pp. 1368–1373.

[48] R. Gao, H. Xia, J. Li, D. Liu, S. Chen, and G. Chun, "DRCGR: Deep reinforcement learning framework incorporating CNN and GAN-based for interactive recommendation," in *Proc. Int. Conf. Des. Mater.*, 2019, pp. 1048–1053.

[49] X. He, Z. He, X. Du, and T.-S. Chua, "Adversarial personalized ranking for recommendation," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2018, pp. 355–364.

[50] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *Proc. Int. Conf. Learn. Representations*, 2021.

[51] Y. Jiang, Y. Yang, L. Xia, and C. Huang, "DiffKG: Knowledge graph diffusion model for recommendation," in *Proc. ACM Int. Conf. Web Search Data Mining*, 2024, pp. 313–321.

[52] F.-A. Croitoru, V. Hondru, R. T. Ionescu, and M. Shah, "Diffusion models in vision: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 9, pp. 10850–10869, Sep. 2023.

[53] L. Yang et al., "Diffusion models: A comprehensive survey of methods and applications," *ACM Comput. Surv.*, vol. 56, no. 4, pp. 105:1–105: 39, 2024.

[54] Y. Tashiro, J. Song, Y. Song, and S. Ermon, "CSDI: Conditional score-based diffusion models for probabilistic time series imputation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 24804–24816.

[55] Z. Zhang, Z. Zhao, J. Yu, and Q. Tian, "ShiftDDPMs: Exploring conditional diffusion models by shifting diffusion trajectories," in *Proc. AAAI Conf. Artif. Intell.*, 2023, pp. 3552–3560.

[56] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 10674–10685.

[57] J. Ho, T. Salimans, A. A. Gritsenko, W. Chan, M. Norouzi, and D. J. Fleet, "Video diffusion models," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 8633–8646.

[58] J. Austin, D. D. Johnson, J. Ho, D. Tarlow, and R. van den Berg, "Structured denoising diffusion models in discrete state-spaces," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 17981–17993.
[59] C. L. Li, J. Thickstun, I. Gulrajani, P. Liang, and T. B. Hashimoto, "Diffusion-LM improves controllable text generation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 4328–4343.
[60] E. Adib, A. S. Fernandez, F. Afghah, and J. J. Prevost, "Synthetic ECG signal generation using probabilistic diffusion models," *IEEE Access*, vol. 11, pp. 75818–75828, 2023.
[61] L. Wu, J. Li, P. Sun, R. Hong, Y. Ge, and M. Wang, "DiffNet++: A neural influence and interest diffusion network for social recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 10, pp. 4753–4766, Oct. 2022.
[62] X. Chen et al., "Group-based social diffusion in recommendation," *World Wide Web*, vol. 26, no. 4, pp. 1775–1792, 2023.
[63] D. Rafailidis and F. Crestani, "Recommendation with social relationships via deep learning," in *Proc. ACM SIGIR Int. Conf. Theory Inf. Retrieval*, 2017, pp. 151–158.
[64] W. Fan et al., "Graph neural networks for social recommendation," in *Proc. World Wide Web Conf.*, 2019, pp. 417–426.
[65] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui, "Graph neural networks in recommender systems: A survey," *ACM Comput. Surv.*, vol. 55, no. 5, 2023, Art. no. 97.
[66] J. Chang et al., "Sequential recommendation with graph neural networks," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2021, pp. 378–387.
[67] W. Chen et al., "Semi-supervised user profiling with heterogeneous graph attention networks," in *Proc. Int. Joint Conf. Artif. Intell.*, 2019, pp. 2116–2122.
[68] S. Baluja et al., "Video suggestion and discovery for YouTube: Taking random walks through the view graph," in *Proc. World Wide Web Conf.*, 2008, pp. 895–904.
[69] Y. Zheng, C. Gao, L. Chen, D. Jin, and Y. Li, "DGCN: Diversified recommendation with graph convolutional networks," in *Proc. World Wide Web Conf.*, 2021, pp. 401–412.
[70] G. Yue, R. Xiao, Z. Zhao, and C. Li, "AF-GCN: Attribute-fusing graph convolution network for recommendation," *IEEE Trans. Big Data*, vol. 9, no. 2, pp. 597–607, Apr 2023.
[71] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1597–1607.
[72] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 297–304.
[73] C. Luo, "Understanding diffusion models: A unified perspective," 2022, *arXiv:2208.11970*.
[74] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *Proc. Conf. Uncertainty Artif. Intell.*, 2009, pp. 452–461.
[75] X. He, K. Deng, X. Wang, Y. Li, Y.-D. Zhang, and M. Wang, "LightGCN: Simplifying and powering graph convolution network for recommendation," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2020, pp. 639–648.
[76] J. Walker, T. Zhong, F. Zhang, Q. Gao, and F. Zhou, "Recommendation via collaborative diffusion generative model," in *Proc. Int. Conf. Knowl. Sci. Eng. Manage.*, 2022, pp. 593–605.
[77] W. Wang, Y. Xu, F. Feng, X. Lin, X. He, and T.-S. Chua, "Diffusion recommender model," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2023, pp. 832–841.
[78] Y. Xiao, R. Ma, and J. Sang, "NFGCL: A negative-sampling-free graph contrastive learning framework for recommendation," *Inf. Sci.*, vol. 695, 2025, Art. no. 121732.
[79] X. Cai, C. Huang, L. Xia, and X. Ren, "LightGCL: Simple yet effective graph contrastive learning for recommendation," in *Proc. Int. Conf. Learn. Representations*, 2023.
[80] D. Liang, R. G. Krishnan, M. D. Hoffman, and T. Jebara, "Variational autoencoders for collaborative filtering," in *Proc. World Wide Web Conf.*, 2018, pp. 689–698.
[81] Y. Wu, C. DuBois, A. X. Zheng, and M. Ester, "Collaborative denoising auto-encoders for top-N recommender systems," in *Proc. ACM Int. Conf. Web Search Data Mining*, 2016, pp. 153–162.
[82] H. Steck, "Embarrassingly shallow autoencoders for sparse data," in *Proc. World Wide Web Conf.*, 2019, pp. 3251–3257.
[83] T. Carraro, M. Polato, L. Bergamin, and F. Aiolli, "Conditioned variational autoencoder for top-N item recommendation," in *Proc. Int. Conf. Artif. Neural Netw.*, 2022, pp. 785–796.
[84] G. Lee, Y. Zhu, H. Yu, Y. Zhou, and J. Li, "Collaborative diffusion model for recommender System," in *Proc. World Wide Web Conf.*, 2025, pp. 1091–1095.
[85] Y. Qu and H. Nobuhara, "Intent-aware diffusion with contrastive learning for sequential recommendation," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2025, pp. 1552–1561.
[86] C. Vignac, I. Krawczuk, A. Siraudin, B. Wang, V. Cevher, and P. Frossard, "DiGress: Discrete denoising diffusion for graph generation," in *Proc. Int. Conf. Learn. Representations*, 2023.
[87] X. Chen, J. He, X. Han, and L. Liu, "Efficient and degree-guided graph generation via discrete diffusion modeling," in *Proc. Int. Conf. Mach. Learn.*, 2023, pp. 4585–4610.
[88] L. Regenwetter, A. H. Nobari, and F. Ahmed, "Deep generative models in engineering design: A review," *J. Mech. Des.*, vol. 144, no. 7, 2022, Art. no. 071704.
[89] J. Lin et al., "A survey on diffusion models for recommender systems," 2024, *arXiv:2409.05033*.
[90] Y. Zhu, C. Wang, Q. Zhang, and H. Xiong, "Graph signal diffusion model for collaborative filtering," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2024, pp. 1380–1390.
[91] Y. Hou, J.-D. Park, and W.-Y. Shin, "Collaborative filtering based on diffusion models: Unveiling the potential of high-order connectivity," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2024, pp. 1360–1369.
[92] Z. Yi, X. Wang, and I. Ounis, "A directional diffusion graph transformer for recommendation," 2024, *arXiv:2404.03326*.
[93] S. Rajput et al., "Recommender systems with generative retrieval," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, vol. 36, 2023pp. 10299–10315.
[94] Y. Deldjoo et al., "A review of modern recommender systems using generative models (Gen-RecSys)," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2024, pp. 6448–6458.
[95] G. Penha, A. Vardasbi, E. Palumbo, M. D. Nadai, and H. Bouchard, "Bridging search and recommendation in generative retrieval: Does one task help the other," in *Proc. ACM Conf. Recommender Syst.*, 2024, pp. 340–349.

Xuan Zhang received the MS degree from Tsinghua University, Beijing, China, in 2023, where she is currently working toward the PhD degree with the Department of Automation. Her research interests include services computing, recommender systems, and large language model.

Xiang Deng received the BE degree from the Institute of Computer and Science, Beijing Jiaotong University, Beijing, China, in 2020. He is currently working toward the PhD (second year) degree with Automation Department, Tsinghua University, Beijing. His research interests include computer vision, computer graphics, and computational photography.

Hongxing Yuan received the ME degree from Nankai University, Tianjin, China, in 2024. She is currently working toward the PhD degree with the Department of Automation, Tsinghua University, Beijing, China. Her research interests include services computing, recommender systems, and large language model.

Chunyu Wei received the BS degree in control theory and application, and the PhD degree in control science and engineering from Tsinghua University, Bejing, China, in 2019 and 2024, respectively. He is currently a lecturer with the School of Information, Renmin University of China, Beijing. His research interests include services computing, recommender systems, and social computing.

Yushun Fan received the PhD degree in control theory and application from Tsinghua University, Beijing, China, in 1990. From 1993 to 1995, he was a visiting scientist, supported by Alexander von Humboldt Stiftung, with the Fraunhofer Institute for Production System and Design Technology (FHG/IPK), Germany. He is currently a tenured professor with the Department of Automation, vice director of National CIMS Engineering Research Center of China, director of the System Integration Institute, and director of the Networking Manufacturing Laboratory, Tsinghua University. He has authored ten books in enterprise modeling, workflow technology, intelligent agent, object-oriented complex system analysis, and computer integrated manufacturing. He has authored or coauthored more than 500 research papers in journals and conferences. His research interests include enterprise modeling methods, system integration, modern service science, and technology. He is a member of IFAC TC5.1 and TC 5.2, vice director of the China Standardization Committee for Automation System and Integration, and editorial member of *International Journal of Computer Integrated Manufacturing*.

Jia Zhang (Senior Member), IEEE received the BS and MS degrees in computer science from Nanjing University, Nanjing, China, and the PhD degree in computer science from the University of Illinois at Chicago, Chicago, IL, USA. She is currently Cruse C. and Marjorie F. Calahan Centennial chair in engineering, professor with the Department of Computer Science, Southern Methodist University, Dallas, TX, USA. She has coauthored one textbook titled "*Services Computing*" and has authored or coauthored more than 200 refereed journal papers, book chapters, and conference papers. Her research interests include center on data science infrastructure, with a focus on scientific workflows, software discovery, and knowledge graph. She is currently an associate editor of *IEEE Transactions on Services Computing*.