MoDiff - Graph Generation with Motif-Aware Diffusion Model

Yuwei Xu
yuweixu@link.cuhk.edu.cn
School of Data Science
The Chinese University of Hong Kong, Shenzhen
Shenzhen, China

Chenhao Ma*
machenhao@cuhk.edu.cn
School of Data Science
The Chinese University of Hong Kong, Shenzhen
Shenzhen, China

Abstract

Temporal graphs, widely used in social network modeling, are valuable for research but pose challenges due to data accessibility and privacy concerns. High-quality graph generation models can produce surrogate data for sharing and training, benefiting tasks such as behavior analysis, anomaly detection, and data augmentation. However, existing deep learning and probabilistic approaches often struggle to balance global statistical properties with local structural details. To overcome this limitation, we leverage motifs, small subgraphs that serve as the building blocks of complex networks, to encode local information. Based on a spectral analysis of motifs, we propose MoDiff, a novel motif-aware diffusion model for temporal graph generation. MoDiff integrates motifs into a diffusion framework by employing motif-enhanced Hermitian matrices that capture local structures and edge orientations, while the spectral diffusion model efficiently generates graphs. Moreover, MoDiff supports controllable graph generation by adjusting density parameters to simulate the evolution of temporal graphs. Experimental results demonstrate that MoDiff outperforms existing approaches, reducing degree discrepancies by 10–50% and clustering discrepancies by 50–90%, while better preserving higher-order structural features. Our code is available at: https://github.com/Yuwe1XU/MoDiff.

CCS Concepts

• **Networks** → *Network simulations*; • **Computing methodologies** → *Network science*.

Keywords

Graph generative model; temporal motifs; graph diffusion

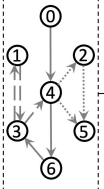
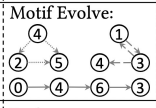
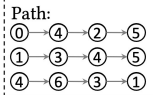
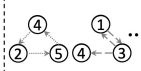
ACM Reference Format:
Yuwei Xu and Chenhao Ma. 2025. MoDiff - Graph Generation with Motif-Aware Diffusion Model. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (KDD '25), August 3–7, 2025, Toronto, ON, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3711896.3737053

*Chenhao Ma is the corresponding author.

KDD '25, Toronto, ON, Canada

ACM ISBN 979-8-4007-1454-2/2025/08
https://doi.org/10.1145/3711896.3737053

Table 1: Demo of generation processes for four methods, with quantitative metrics results on StackOverflow graph (1000-Node Scale). Metrics are MMD, where smaller is better.

Graph	Model	Local Structure	Global Structure	MMD
	TASBM		State : $C^{in}, C^{out}, a_u^{in}, a_u^{out}$ Edge Arrival : $\theta_{ij}, \theta_{a_u a_v}$ Time window : T, δ Motif Count : $N_M, E[N_M]$ …	Deg.: 0.260 Clu.: 0.749 Orb.:1.3e-2
	MTM	Motif Evolve: 		Deg.: 0.188 Clu.: 0.228 Orb.:5.1e-4
	TagGen	Path: 		Deg.: 0.089 Clu.: 0.445 Orb.:3.4e-3
	MoDiff	Motif Label: 	Motif labelled Adj matrix ⇩ Spectral Decompose 	Deg.: 0.045 Clu.: 0.028 Orb.:3.1e-4

1 Introduction

Temporal graphs, characterized by directed edges with specific timestamps, are pivotal in analyzing complex real-world systems such as communication networks [11, 39, 40], financial transactions [5, 10], and physical models. They are widely applied in social network analysis for tasks such as behavior prediction, anomaly detection, and user profiling [3, 13, 28, 34]. Training models on temporal graph data requires large-scale datasets. However, real-world data is sensitive, raising significant privacy concerns. To overcome these challenges, graph generation techniques [12, 23, 26, 27] have become essential for simulating realistic data, enabling surrogate data sharing, data augmentation, and privacy protection.

Existing graph generators and their limitations. Several types of graph generators have been developed to meet the demands. *Parametric models* like EpicNet [26], **TASBM** [22] and STM [23] generate graphs using distribution-based parameters like community ratios and node activity state. *Probabilistic models*, such as **MTM** [12], further model the occurrence and transitions of special patterns, like motifs, instead of individual edges. *Likelihood-based generative models*, including GraphGAN [30], GraphVAE [27], GraphRNN [36] and **TagGen** [42], generate complete graphs by simulating the edge distribution of nodes through random walk. Table 1 presents demos from one representative method per category, with metrics evaluated on a 1000-node scale StackOverflow dataset. Details are provided in Section 4.

However, as shown in Table 1, these methods face key limitations in their perception scope. *Parametric models* rely on parameters based on the entire graph, making capturing correlations between

individual edges challenging. *Probabilistic models* improve this by incorporating motif evolution but overlook how finer-grained structures influence the global graph. Expanding these models to larger regions is theoretically possible but impractical. The exponential growth in combinations and increased noise severely escalate complexity and instability. Likewise, *likelihood-based methods* face similar challenges. They often rely on sampling multiple random walks per node to simulate edge distributions, which creates a significant computational bottleneck. TagGen, for instance, requires extensive random walk operations to process 1 million nodes (1,000 graphs × 1,000 nodes), with one training epoch requiring half an hour.

Our Work. Balancing the perception of fine-grained details with global structural information remains challenging—expanding local features to the global scale is often impractical. Instead, integrating global generation mechanisms with fine-grained structures offers a viable solution. Recent advancements—such as diffusion-based graph generators that directly produce adjacency matrices—have shown promise in capturing global edge correlations. Meanwhile, motifs have long served as a fundamental tool for micro-level graph analysis [2, 4, 8, 11, 20]. Using rigorous statistical methods and hypothesis testing, we analyze the role of motifs to identify those that most effectively label fine-grained features, and we further examine how these motifs impact the graph's spectral decomposition to enhance its structural representation.

Hence, we propose MoDiff, a Motif-aware Diffusion Model that utilizes motif analysis to encode fine-grained structural details, effectively bridging local patterns with global edge correlations. MoDiff builds upon the spectral diffusion method with three new designs: 1) To capture fine-grained structures, we use motifs to label recurring local patterns in temporal graphs and use statistical analysis to identify and quantify the most effective ones. 2) To overcome the limitation of existing diffusion-based generators—which are typically designed for undirected graphs—we represent the adjacency matrix as a Hermitian matrix, thereby accommodating edge directions, bidirectional influences, and asymmetry, with real-valued eigenvalues facilitating effective spectral diffusion. 3) To mitigate the computational cost and potential information loss associated with training on multiple data labeled via different motifs (as seen in ensemble approaches), we introduce a Magnetic Hermitian Matrix that encodes multiple motifs within a single matrix, enabling integrated training of the diffusion model. Section 4 compares MoDiff against several state-of-the-art temporal graph generation models on real-world datasets across diverse domains. Our method reduces degree discrepancy by 10-50% and clustering discrepancy by 50-90% compared to other approaches, while preserving a near-1 ratio of higher-order structures between the generated and target graphs without specialized training.

Our key contributions can be summarized as follows:

- *New Findings.* We demonstrate how statistical and hypothesis testing methods can effectively select motifs as structural labels for deep learning models, such as diffusion models, and analyze their impact on graph properties and performance.
- *New Model.* We introduce MoDiff, a novel temporal graph generator based on the spectral diffusion model. It features an efficient encoding strategy that captures graph structural information w.r.t. multiple motifs within a single Magnetic Hermitian Matrix, enriching input data while avoiding the inefficiencies and information loss of separate training.
- *Controllable Generation.* During decoding, we introduce a single parameter to control graph density. This ensures adjustable graph density without compromising structural stability, and we provide theoretical validation of this method.
- *SOTA Performance.* Extensive experiments on real-world datasets demonstrate MoDiff's superiority in preserving structural information, such as degree distribution and clustering, while achieving strong performance on other high-level structural features.

2 Background

In this section, we first present the definition of the temporal graph and its generation problem in Section 2.1, and then briefly review the temporal motifs and analyze them in Section 2.2.

2.1 Problem Definition

Temporal graphs are commonly represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X})$. Here, $\mathcal{V}$ is the set of nodes, and $\mathcal{E}$ is the set of events, where each event $e_i \in \mathcal{E}$ is a 3-tuple (u_i, v_i, t_i) representing a directed event from source node u_i to target node v_i at time t_i. $\mathcal{X}$ is a matrix for encoding node attribute information. To capture the dynamic evolution of the temporal graph, we divide the timeline into K consecutive, non-overlapping time windows of equal duration $\delta \in \mathbb{R}^+$, with the k-th interval denoted as $[k\delta, (k+1)\delta)$, where $k \in [0, K)$. To facilitate the model processing, the temporal graph in the k-th time window is projected as a snapshot $G^k = (\mathcal{V}^k, E^k, \mathcal{X}^k)$, where $E^k = \{(u_i, v_i) \mid (u_i, v_i, t_i) \in \mathcal{E} \wedge t_i \in [k\delta, (k+1)\delta)\}$. While the $\mathcal{X}^k$ denotes the attribute set of vertex $v \in \mathcal{V}^k$ at timestamp k.

Problem 1. Given the observed network $\mathcal{G}^k$ up to time k, the task is to generate $\tilde{G}^{k+1} = (\tilde{\mathcal{V}}^{k+1}, \tilde{E}^{k+1}, \tilde{\mathcal{X}}^{k+1})$ satisfying $S(\tilde{G}^{k+1}) \approx S(G^{k+1})$, where $S(\cdot)$ denotes a set of statistical measures of the graph, including but not limited to degree and clustering coefficient.

Unlike prediction, generation focuses on preserving overall structural and statistical characteristics rather than forecasting specific future events or edges with high accuracy.

2.2 Motif on Graph Generation

Temporal Motif. *Motifs* are small and recurring subgraph patterns defined by a specific combination of edges. In a temporal graph, motifs can be labeled with the number of events they have. An l-event motif $\mathcal{M}^l = (\mathcal{V}', \mathcal{E}')$ is defined as a connected temporal subgraph of $\mathcal{G}$, where $\mathcal{V}' \subseteq \mathcal{V}$, $\mathcal{E}' \subseteq \mathcal{E}$, and $|\mathcal{E}'| = l$. After projection, $\mathcal{M}^l = (\mathcal{V}', \mathcal{E}')$ will be reduced to the projected motif $M^l = (\mathcal{V}', E')$ by ignoring the timestamps on edges.

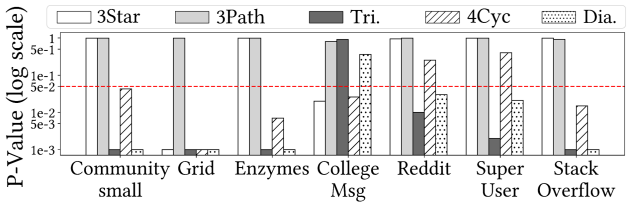


Figure 1: MOSER [18]: Verifying whether a motif is crucial.

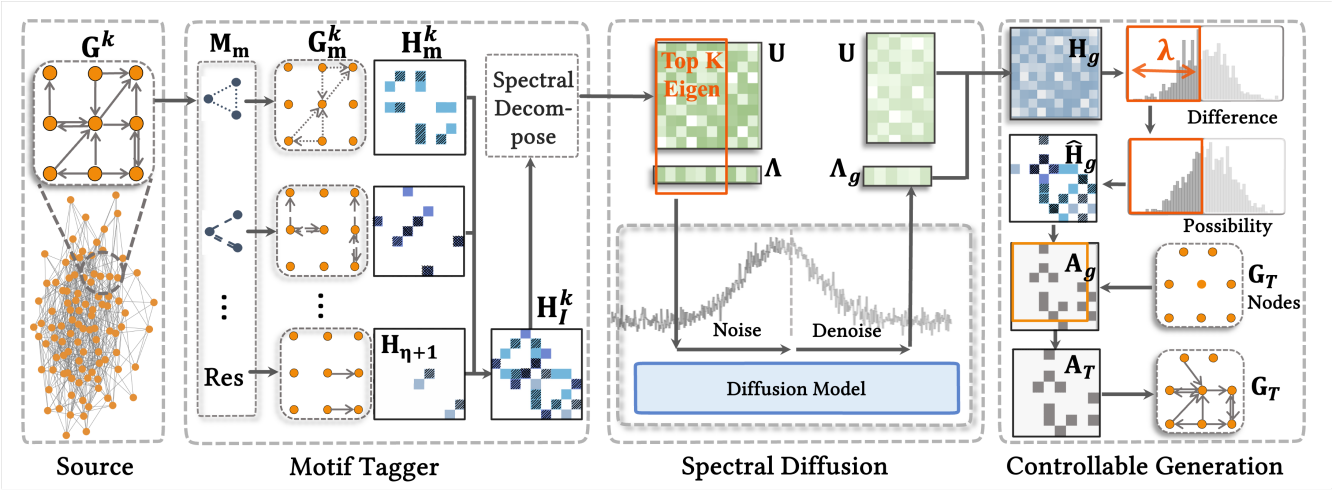


Figure 2: Framework of MoDiff.

Selecting appropriate motifs to capture local graph structures requires an efficient strategy due to the vast number of possible motifs. Deep learning methods often depend on additional labeling, which can be costly. To assess the structural relevance of motifs, we integrate existing hypothesis testing techniques and analyze motif occurrence probabilities across multiple graphs. We adapt the motif detection algorithm, MOSER [18], to analyze the structural roles of several motifs in various graphs: 3-Star, 3-Path, Triangle (Tri.), 4-Cycle (4Cyc), and Diamond (Diam.). Figure 1 illustrates the p-values obtained from hypothesis testing on motif occurrence probability, displayed on a log scale. A dashed line at 0.05 marks the significance threshold, with values below this threshold indicating motifs with structural importance. The results suggest that motifs connecting more nodes (e.g., 3-Star, 3-Path) emphasize network connectivity but lack structural specificity. In contrast, loop-dominated motifs, such as Triangles and 4-Cycles, are more distinctive in both undirected and social networks, making them effective for annotating finer-grained structures.

Table 2: Proportion of High-Frequency Differences.

Scale	CollegeMsg	Reddit	SuperUser	StackOverflow
100	11.7	10.3	18.0	10.9
500	23.1	16.7	12.8	17.8
1000	19.3	11.2	12.8	19.0

Motifs also exhibit structural specificity at the spectral level, particularly in high-frequency eigenvalues. We applied motif-based labeling to the edges of triangles in adjacency matrices across subgraphs of temporal graphs. To eliminate bias, we also randomly labeled an equal number of edges for comparison. The results show minimal differences in low-frequency eigenvalues but significant deviations in high-frequency ones. Table 2 presents the proportion of these differences, with high-frequency eigenvalue deviations ranging from at least 10% to nearly 20% across different graph scales. This difference demonstrates that using motifs as fine-grained structural labels is more prominently reflected in high-frequency eigenvalues. Since high-frequency eigenvalues are more sensitive to local structures, motif labeling effectively captures structural variations. Therefore, motif labeling is applicable to both adjacency matrix and spectral space. A detailed diagram can be found in Appendix A.

3 Method

In this section, we present our motif-based temporal network generator (MoDiff). We provide an overall framework of the generator in Section 3.1, and detail the key modules in Section 3.2 and 3.3, with a rationality analysis in Section 3.4.

3.1 MoDiff Framework

Figure 2 illustrates the overall architecture of MoDiff. Given a temporal graph $G^k = (\mathcal{V}^k, E^k, \mathcal{X}^k)$ at time window k, the goal of MoDiff is to generate a new graph $\tilde{G}^{k+1} = (\mathcal{V}^k, \tilde{E}^{k+1})$ for time window $k+1$. The key challenge lies in capturing both the global and local structure information while guaranteeing robustness and efficiency. To address the challenges above, MoDiff incorporates two key components: **Motif Tagger**, which encodes different views of the graph w.r.t. different motifs into a Magnetic Hermitian Matrix to ensure capturing the graph structural information from multiple perspectives efficiently; **Controllable Diffusion-based Generator**, which utilizes spectral decomposition to avoid performing diffusion on the whole matrix and decodes the generated Hermitian matrix into the graph while supporting controllable parameters.

As shown in Figure 2, the overall process of MoDiff contains four steps. First, the whole graph is divided into subgraphs to fit the input shape of the model. Secondly, **Motif Tagger** incorporates temporal motifs as fine-grained structures to enrich the adjacency matrix with higher-order structural information. Using the analyzed motifs, motif-related edges are detected and encoded. These encoded subgraphs are merged into a single Hermitian matrix, improving efficiency while preserving information integrity. Details are provided in Section 3.2. Thirdly, the eigenvectors and eigenvalues of the Hermitian matrix are extracted via spectral decomposition, allowing the model to avoid operating on the full matrix. The diffusion model is trained with subgraphs from 0 to $k-1$, to

generate eigenvalues (Section 3.3), which are combined with eigenvectors uniformly sampled from observed subgraphs to construct a new Hermitian matrix. The final step, **Controllable Generation**, converts the Hermitian matrix into a graph with adjustable density. MoDiff maps continuous values to discrete edges or motifs in the adjacency matrix, ensuring it reflects the graph structure (Section 3.4). The decoded adjacency matrix is then aligned with the target graph via node correspondence to produce the final structure.

3.2 Motif Tagger

Due to the asymmetry of directed edges and the binary 0-1 labeling, traditional adjacency matrices present two key challenges: ① it limits the applicability of spectral decomposition, which the downstream diffusion model relies on, and ② it contains limited information beyond basic edge connectivity.

To address Challenge ①, we first make adjacency matrix spectrally decomposable by converting it into a Hermitian matrix. Hermitian matrix refers to a complex matrix equaling its own conjugate transpose, and has all real eigenvalues. The Hermitian matrix allows us to achieve three key objectives: 1)distinguishing edge directions, 2) efficient spectral decomposition, and 3) recovering asymmetric adjacency in the graph. Thus, the eigenvalues of Hermitian can be directly used for the training of the spectral diffusion model, and the generated adjacency matrix can be mapped to the graph in a direction-preserving manner.

To address Challenge ②, we propose an efficient and low-cost labeling approach using motifs. Motifs, which represent various types of network behavior, effectively capture higher-order structural information in graphs. While valid motifs can be identified based on prior analysis, the challenge lies in combining them to generate the target graph. A simple solution is ensemble learning, where multiple models are trained on different motif views and their outputs combined. However, this approach significantly increases computational costs and compromises global structural information. To overcome these limitations, we further utilize the Hermitian matrix, whose components can encode additional information. By embedding different motif views into the Hermitian matrix, we preserve both local structures and directional edge information, enabling a more efficient and unified representation.

Encoding Hermitian matrix involves extracting and representing motif views from G^k based on different motifs. The motif view on motif M_m denotes as $G_m^k = (\mathcal{V}, E_m^k)$, where $E_m^k = \{e \in E^k | \exists S \in \mathcal{V}, e \in G^k[S] \text{ and } G^k[S] \sim M_m\}$. $G^k[S]$ denotes the subgraph induced by S and $G^k[S] \sim M_m$ denotes isomorphic. In other words, G_m^k includes all edges belonging to subgraphs isomorphic to M_m. Thus, each view G_m^k contains the information specific to M_m.

Since representing various graphs with Hermitian matrices is still an open question, we adopt a method similar to the Magnetic Laplacian in this work. For motif view G_m^k with $|\mathcal{V}| = n$, we simplify the derivation by ignoring the time label k and denote its adjacency matrix as $\mathbf{A}_m \in \mathbb{R}^{n \times n}$. The Magnetic Hermitian matrix is constructed by combining the connection strength matrix A_m^s and the phase matrix $\mathbf{\Theta}_m^{(\theta)}$.

$$\begin{aligned} A_m^s(r) &= r(\mathbf{A}_m + \mathbf{A}_m^T), \quad r \in \mathbb{R} \\ \mathbf{\Theta}_m^{(\theta)} &= \exp(2\pi\theta \cdot i(\mathbf{A}_m - \mathbf{A}_m^T)), \quad \theta \in \mathbb{R}^+ \end{aligned} \tag{1}$$

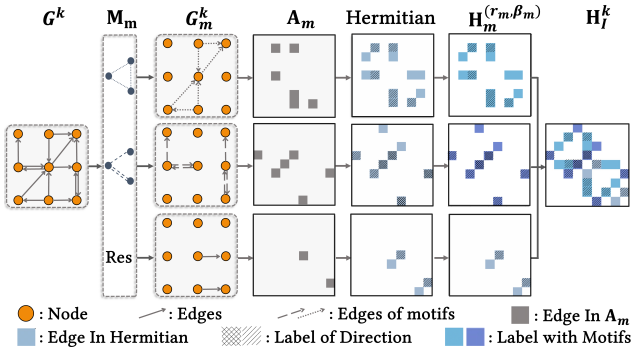


Figure 3: The process to derive the integrated Hermitian.

where r determines the strength of A_m^s, and the phase matrix $\mathbf{\Theta}_m^{(\theta)}$ rotates the imaginary part by angle θ. To clearly distinguish bidirectional from unidirectional edges while preserving edge directionality, θ is fixed as 0.25. In this case, for a unidirectional edge (u, v), $\mathbf{\Theta}_m^{(.25)}(u, v) = -\mathbf{\Theta}_m^{(.25)}(v, u) = i$. And for bidirectional edges, both (u, v) and (v, u) yield 1. For simplicity, we will denote $\mathbf{\Theta}_m = \mathbf{\Theta}_m^{(.25)}$ throughout the rest of this article.

Then we focus on distinguishing motif views. The symbol $\odot$ denotes element-wise multiplication, while $\mathrm{Re}(\mathbf{\Theta}_m)$ and $\mathrm{Im}(\mathbf{\Theta}_m)$ represent the real and imaginary components of $\mathbf{\Theta}_m$, respectively. Using factors r and β to independently control the real and imaginary parts, the matrix $\mathbf{H}_m^{(r_m,\beta_m)}$ for motif M_m is expressed as:

$$\mathbf{H}_m^{(r_m,\beta_m)} = \mathbf{A}_m^s(r_m) \odot (\mathrm{Re}(\mathbf{\Theta}_m) + \beta_m \cdot \mathrm{Im}(\mathbf{\Theta}_m)) \,. \tag{2}$$

The parameters r_m and β_m in Equation 2 allow $\mathbf{H}_m^{(r_m,\beta_m)}$ to distinguish edges within motif view G_m^k. Then MoDiff merges the Hermitian matrices $\mathbf{H}_m^{(\alpha,\beta)}$ from η selected motif views into a integrated matrix $\mathbf{H}_I$. To retain the full graph, edges do not belong to any motif view makes up residual graph $G_{Res}^k = (\mathcal{V}, E_{Res}^k)$, which is encoded into $\mathbf{H}_{Res}^{(r_{\eta+1},\beta_{\eta+1})}$. The operation $\oplus$ is defined as a selective merge, preserving only the non-zero elements from the encoded Hermitian matrices. So with $\eta + 1$ Hermitian matrices, the final integrated matrix $\mathbf{H}_I^k$ of G^k can be expressed as:

$$\mathbf{H}_I^k = \mathbf{H}_1^{(r_1,\beta_1)} \oplus \mathbf{H}_2^{(r_2,\beta_2)} \oplus \cdots \oplus \mathbf{H}_{\mu+1}^{(r_\mu,\beta_\mu)} \oplus \mathbf{H}_{Res}^{(r_{\mu+1},\beta_{\mu+1})} \tag{3}$$

Since an edge can belong to multiple motifs, the merging order of the Hermitian matrices should follow the structural importance of each motif, as outlined in previous work [7, 18, 21], which we omit here for brevity. Figure 3 provides a simple example to illustrate the process of encoding motif views and constructing the integrated Hermitian matrix from the original graph. In the matrix diagram, squares indicate existing values, with varying shades representing different encoding methods.

3.3 Diffusion with Spectral Decomposition

In prior work, GDSS [6] first employed diffusion models for graph generation, but its performance was limited by the direct use of the adjacency matrix. GSDM [15], proposed by Luo et al., improved performance by conducting diffusion only on eigenvalues obtained by spectral methods. However, GSDM is restricted to undirected

Algorithm 1: MoDiff Training with Motif Tagger

Input : Graphs G^k with various k, selected motifs M_m, noise steps T
Output : Learned score models $\rho_{\theta(\mathcal{X})}$, $\rho_{\theta(\Lambda)}$

/* Preprocessing: */
foreach G^k **do**
  Extract Subgraph G_m^k with motif M_m
  Compute Hermitian embedding H_I^k via Eqs.1–3
  Extract spectral pair $(\mathcal{X}^k, \Lambda^k)$
/* Training Loop: */
for *training epochs* **do**
  Sample mini-batch $\{(\mathcal{X}, \Lambda)\}$; // omit k
  foreach $(\mathcal{X}, \Lambda)$ *in mini-batch* **do**
    Perturb: $(\mathcal{X}, \Lambda) \xrightarrow{\text{noise, Eq. 4}} (\mathcal{X}_t, \Lambda_t)$
    Compute scores $\rho_{\theta(\mathcal{X})}(\mathcal{X}_t, \Lambda_t)$ and $\rho_{\theta(\Lambda)}(\mathcal{X}_t, \Lambda_t)$
    Compute losses $\mathcal{L}_{\mathcal{X}}$, $\mathcal{L}_{\Lambda}$ via Eq. 6
    Update $\theta_{\mathcal{X}}, \theta_{\Lambda}$ by gradient descent

Algorithm 2: MoDiff Sampling with Controllable Density

Input : Trained models $\rho_{\theta(\mathcal{X})}, \rho_{\theta(\Lambda)}$, target graph nodes N_{target}, Training data $\mathcal{D}_k$, denoise steps T_{den}
Output : Generated graph G_{gen}

/* Initialization: */
Extract $\{U_i\}$ via Hermitian spectral decomposition (Eqs. 1–3)
Determine λ via data-driven or heuristic methods
Sample noise $\mathcal{X}_T \sim \mathcal{N}(0, I)$, $\Lambda_T \sim \mathcal{N}(0, I)$
/* Reverse Diffusion: */
for $t = T_{den}$ *to* 1 **do**
  Predict score $\rho_{\theta(\mathcal{X})}(\mathcal{X}_t, \Lambda_t)$, $\rho_{\theta(\Lambda)}(\mathcal{X}_t, \Lambda_t)$
  Update $\mathcal{X}_{t-1}, \Lambda_{t-1}$ by solver of reversed SDE (Eq. 5)
/* Reconstruction: */
Form Hermitian matrix $\mathbf{H}_g = U_i \, \Lambda_g \, U_i^{\dagger}$
Threshold: $\hat{\mathbf{H}}_g = \text{Clip}(\mathbf{H}_g, \lambda)$ (Eq. 15)
Binarize: $A_g = \mathbb{I}(\hat{\mathbf{H}}_g > 0)$
Mask: $\hat{A} = \text{Mask}(\mathcal{X}_g, A_g, N_{target})$
Return $G_{\text{gen}} = (N_{target}, \hat{A})$

graphs and struggles to capture fine-grained structural details. The integrated Hermitian matrix by MoDiff enables spectral diffusion models to be applied on directed graphs.

The spectral diffusion model applies the diffusion process to the eigenvalues. The full training workflow using the motif-tagger enhanced Hermitian matrix is outlined in Algorithm 1. In general, diffusion models progressively add noise to data and then learn to reverse the process to reconstruct the original data, leveraging stochastic differential equations (SDEs) to model both the forward (noise addition) and reverse (denoising) dynamics. For the spectral method, consider the spectral decomposition of the integrated Hermitian matrix $\mathbf{H}_I^k$ as $\mathbf{U}^k \, \mathbf{\Lambda}^k \, (\mathbf{U}^k)^{\dagger}$. Here, $\mathbf{U}^k$ contains the eigenvectors as its columns, $(\mathbf{U}^k)^{\dagger}$ is the conjugate transpose of $\mathbf{U}^k$, and $\mathbf{\Lambda}^k$ is a diagonal matrix of eigenvalues. $\mathcal{X}^k \in \mathbb{R}^{n \times d}$ can encode node features like degrees (no need for other expensive labels), where d is the feature-length. Although joint diffusion over $\mathcal{X}^k$ and $\mathbf{\Lambda}^k$ can enhance training [6, 15], *this chapter focuses only on the derivation of* $\mathbf{\Lambda}^k$, since the primary difference is whether noise is added to diagonal elements, thus requiring separate models. And *the notation of time window k is also omitted in this chapter* to simplify.

Since spectral theory does not rely on full matrices, the dimensions of $\mathbf{\Lambda}$ can be reduced from n to l to lower computational costs. With diagonal matrix $\mathbf{\Lambda} \in \mathbb{R}^{l \times l}$ of l selected eigenvalues and $\mathcal{X} \in \mathbb{R}^{n \times d}$, the SDE of the whole spectral diffusion generator can be expressed as following equation:

$$\mathrm{d}\mathbf{\Lambda}_t = \mathbf{f}^{\Lambda}(\mathbf{\Lambda}_t, t)\mathrm{d}t + \sigma_t^{\Lambda}\mathrm{d}\mathbf{B}_t^{\Lambda}, \quad t \in [0, T] \tag{4}$$

where $\mathbf{f}^{\Lambda}(\cdot, t) : \mathbb{R}^{l \times l} \mapsto \mathbb{R}^{l \times l}$ is drift function acting only on the diagonal spectrum. σ_t^{Λ} refers to the scalar function for fluctuation of diffusion terms. $\mathbf{B}_t^{\Lambda}$ are Brownian motions on $\mathbb{R}^{l \times l}$, which acts only on the diagonal elements. Note that $\mathbf{f}^{\mathcal{X}}(\cdot, t)$ and $\mathbf{B}_t^{\mathcal{X}}$ for $\mathcal{X}$ acts on all the elements.

Suppose $p_t(\cdot)$ as the probability density function of $\mathcal{X}_t$ at time t, the Reversed time SDE, which aims to train the model for denoising data back towards D, can be written as:

$$\mathrm{d}\bar{\mathbf{\Lambda}}_t = [\mathbf{f}^{\Lambda}(\mathbf{\Lambda}_t, t) - \sigma_t^2 \nabla_{\mathbf{\Lambda}_t} \log p_t(\mathbf{\Lambda}_t | \mathbf{\Lambda}_0)]\mathrm{d}\bar{t} + \sigma_t^{\Lambda}\mathrm{d}\overline{\mathbf{B}}_t^{\Lambda}, \tag{5}$$

where $\overline{\mathbf{B}}_t^{\Lambda}$ is reversed time Brownian motions on diagonal elements, $\mathrm{d}\bar{t} = -\mathrm{d}t$ is the backward timestamp for denoise from T to 0. $\nabla_{\mathcal{X}_t} \log p_t(\cdot)$is a score function representing the original feature probability at time t, it can be estimated with a score-based GNN $\rho_{\theta(\Lambda)}(\cdot, \cdot)$. The training objective of GNN is to approximately estimate the score by minimizing:

$$\mathcal{L}_{\Lambda}(\theta) = \mathbb{E}_{\mathbf{G}} \mathbb{E}_{\mathbf{\Lambda}_t | \mathbf{G}} || \rho_{\theta(\Lambda)}(\mathcal{X}_t, \mathbf{\Lambda}_t) - \nabla_{\mathbf{\Lambda}_t} \log p_t(\mathbf{\Lambda}_t | \mathbf{\Lambda}_0) ||^2. \tag{6}$$

For generated $\mathbf{\Lambda}_g$, since it is close to $\mathbf{\Lambda}_0$, it can be regarded as the result of adding a perturbation $\Delta\mathbf{\Lambda}$ to $\mathbf{\Lambda}_0$:

$$\mathbf{\Lambda}_g = \mathbf{\Lambda}_0 + \Delta\mathbf{\Lambda}. \tag{7}$$

3.4 Controllable Graph Generation

The diffusion model generates eigenvalues within the prior feature space. However, since these eigenvalues may deviate from the target eigenvalues, the resulting Hermitian matrix $\mathbf{H}_g = \mathbf{U}\mathbf{\Lambda}_g\mathbf{U}^{\dagger}$ consists of continuous values rather than values corresponding to different discrete motif tags in $\mathbf{H}_I$. To round the values to motif tags, we show that for edge (u, v), the probability of its occurrence can be approximated with the minimum distance between generated matrix $\mathbf{H}_g(u, v)$ and motif tags $\mathbf{H}_m(u, v)$ to any motif M_m.

With $\mathbf{\Lambda}_g$ generated by the diffusion process and $\mathbf{H}_g = \mathbf{U}\mathbf{\Lambda}_g\mathbf{U}^{\dagger}$, the probability of the target matrix $\mathbf{H}_T$ is expressed as $\mathbb{P}(\mathbf{H}_T | \mathbf{H}_g)$. With Bayes' Theorem, the probability $\mathbb{P}(\mathbf{H}_T | \mathbf{H}_g)$ can be converted into the likelihood function $\mathbb{P}(\mathbf{H}_g | \mathbf{H}_T)$. Decompose the Hermitian matrix $\mathbf{H}_T$ of the target graph as $\mathbf{H}_T = \mathbf{U}_T\mathbf{\Lambda}_T\mathbf{U}_T^{\dagger}$. The $\mathbf{\Lambda}_T$ can be projected to the eigenspace of $\mathbf{U}$ with $\mathbf{U}\mathbf{U}^{\dagger} = \mathbf{I}$:

$$\begin{aligned} \mathbf{H}_T = \mathbf{U}\mathbf{U}^{\dagger}\mathbf{H}_T\mathbf{U}\mathbf{U}^{\dagger} = \mathbf{U}\mathbf{U}^{\dagger}\mathbf{U}_T\mathbf{\Lambda}_T\mathbf{U}_T^{\dagger}\mathbf{U}\mathbf{U}^{\dagger} = \mathbf{U}\hat{\mathbf{\Lambda}}_T\mathbf{U}^{\dagger}, \\ \hat{\mathbf{\Lambda}}_T = \mathbf{U}^{\dagger}\mathbf{U}_T\mathbf{\Lambda}_T\mathbf{U}_T^{\dagger}\mathbf{U}, \end{aligned} \tag{8}$$

and the likelihood function $\mathbb{P}(\mathbf{H}_g|\mathbf{H}_T)$ can then be expressed as:

$$\mathbb{P}(\mathbf{H}_g|\mathbf{H}_T) = \mathbb{P}(\mathbf{U}\mathbf{\Lambda}_g\mathbf{U}^\dagger|\mathbf{U}\hat{\mathbf{\Lambda}}_T\mathbf{U}^\dagger) = \mathbb{P}(\mathbf{\Lambda}_g|\hat{\mathbf{\Lambda}}_T). \tag{9}$$

As described in Equation 7, the generated eigenvalues $\mathbf{\Lambda}_g$ can be expressed by adding perturbation $\Delta\mathbf{\Lambda}$ to $\hat{\mathbf{\Lambda}}_T$: $\mathbf{\Lambda}_g = \hat{\mathbf{\Lambda}}_T + \Delta\mathbf{\Lambda}$. As $\mathbf{U}^\dagger\mathbf{U} = \mathbf{I}$, we have

$$\|\mathbf{H}_T - \mathbf{H}_g\|_F = \|\mathbf{U}(\hat{\mathbf{\Lambda}}_T - \mathbf{\Lambda}_g)\mathbf{U}^\dagger\|_F = \|\Delta\mathbf{\Lambda}\|, \tag{10}$$

where $\|\cdot\|_F$ is Frobenius norm, which can be expressed as: $\|\mathbf{A}\|_F = \sqrt{\Sigma_i\Sigma_j|\mathbf{A}(i,j)|^2}$. It is similar to Euclidean distance in the real domain, which works on $\Delta\mathbf{\Lambda}$. So Equation 10 can also be expressed as:

$$\|\Delta\mathbf{\Lambda}\| = \sqrt{\Sigma_{u=1}^{n}\Sigma_{v=1}^{n}|\mathbf{H}_T(u,v) - \mathbf{H}_g(u,v)|^2}. \tag{11}$$

Since there is no prior knowledge of the target Hermitian matrix $\mathbf{H}_T$, it can only be approximated with the Hermitian matrix tagged by motifs. Let $\mathbf{H}_m(u,v)$ denote possible motif tag values of the edge between u and v w.r.t. motif M_m. As the edge has four possible cases, $u \to v$, $u \leftarrow v$, $u \leftrightarrow v$, and no connection, $\mathbf{H}_m(u,v)$ also have four possible values, and the actual values can be computed based on α_m and β_m. Then, for each node pair (u,v), we approximate $|\mathbf{H}_T(u,v) - \mathbf{H}_g(u,v)|$ by

$$\min_{m,\mathbf{H}_m(u,v)}\{|\mathbf{H}_m(u,v) - \mathbf{H}_g(u,v)|^2\} \le |\mathbf{H}_T(u,v) - \mathbf{H}_g(u,v)|^2. \tag{12}$$

The motif M_m and $\mathbf{H}_m(u,v)$ that minimize the left-hand side is always no larger than $\mathbf{H}_T(u,v)$, since $\mathbf{H}_T(u,v)$ is one possible value of $\mathbf{H}_m(u,v)$.

When the eigenvalue difference $\Delta\mathbf{\Lambda}$ becomes smaller, $\mathbf{H}_T(u,v)$ aligns more closely with $\mathbf{H}_g(u,v)$, leading to a higher likelihood of similarity. Then, the minimum distance $\min_{m,\mathbf{H}_m(u,v)}\{|\mathbf{H}_m(u,v) - \mathbf{H}_g(u,v)|^2\}$ also decreases, indicating closer proximity between the compared elements. as shown in Equation 12. By applying a sigmoid function, we convert the difference between $\mathbf{H}_g(u,v)$ and $\mathbf{H}_m(u,v)$ into the probability $\mathbb{P}$ of the edge (u,v) appearing:

$$\mathbb{P}((u,v),\sigma,\tau) = \frac{1}{1+\exp\left(\frac{\min\{|\mathbf{H}_m(u,v)-\mathbf{H}_g(u,v)|\}-\sigma}{\tau}\right)}, \tag{13}$$

where σ serves as a scale factor controlling the probability, and τ is a smoothing factor. With this conversion, the possibility of edge existence in the generated graph can be transformed into a more straightforward function with threshold λ and ϵ:

$$\min\{|\mathbf{H}_m(u,v) - \mathbf{H}_g(u,v)|\} \le \lambda \Rightarrow \mathbb{P}((u,v) \text{ in } G) \ge 1-\epsilon. \tag{14}$$

The threshold λ, typically analyzed based on training data or trends but also supporting manual adjustment, is used to generate the graph view $\hat{\mathbf{H}}_g$ from $\mathbf{H}_g$:

$$\hat{\mathbf{H}}_g = \text{Clip}(\mathbf{H}_g,\lambda) = \begin{cases}\mathbf{H}_m(u,v), \text{when } |\mathbf{H}_g(u,v) - \mathbf{H}_m(u,v)| \le \lambda;\\ 0, \qquad\qquad \text{else.}\end{cases} \tag{15}$$

The $\hat{\mathbf{H}}_g$ would then be decoded into adjacency matrix $A_g \in \mathrm{R}^{n\times n}$. Then the full matrix A_g will be mapped into the target nodes to generate the adjacency matrix A_T and graph G_T. Details on the MoDiff Sampling with Controllable Density procedure can be found in Algorithm 2. Despite the dense variety by λ, the structural properties of the generated graph always remain similar to those of the original graphs.

4 Experiments

In this section, we present a comprehensive evaluation. First, we introduce experimental settings, datasets used, and graph generation baselines. After that, we aim to answer the following questions: **Q1.** Can MoDiff outperform existing baselines in multilevel graph discrepancy? **Q2.** How effectively does MoDiff preserve high-order structural features? **Q3.** Which design components most contribute to MoDiff's performance? **Q4.** Can MoDiff generate graphs with controllable properties? **Q5.** How scalable is MoDiff?

4.1 Setup

All experiments were conducted on a Linux system running on a machine equipped with an Intel Gold 6226R CPU, 512 GB of memory, an NVIDIA RTX 3090 with 24 GB VRAM, and 1 TB SSD. The implementation was done using Python.

Datasets. We evaluate the approaches on four real-world temporal networks. The networks include CollegeMsg(CM), Reddit(RD), SuperUser(SU), and StackOverflow (SO). Their statistics are shown in Table 3, where **MD** is the mean degree.

Table 3: Dataset statistics.

Name	Nodes	Edges	Events	MD	Span(days)
CollegeMsg(CM)	1.90K	20.3K	59.8K	10.7	193
Reddit(Re)	54K	234K	572K	4.3	1,217
SuperUser(SU)	194K	925K	1.44M	4.8	2,773
StackOverflow(SO)	260K	4.15M	6.35M	16.0	886

We aim to generate subgraphs by analyzing the known edge distribution within communities. In temporal networks, especially social networks, interactions between users often occur within a limited time frame and then fade. Community subgraphs can capture localized and meaningful structural patterns. The community subgraphs are sampled under time windows at varying scales: 100, 500, and 1000 nodes, with a fluctuation of ±20%. These scales encompass approximately 2-3 hop neighbors, i.e., the typical GNN training ranges. For CollegeMsg, due to its small size (1899 nodes), the experiment at 1000 scale was conducted on the entire graph. Most methods degrade significantly due to limited data.

Baselines. We compare our model with the following generative methods: **TASBM** [22] and **STM** [23] are parametric models. **MTM** [12] is a probabilistic model. We provide two variants: **MTM-All** is trained on the whole graph, generates the whole graph, and extracts the community subgraph, while **MTM-Sub** is directly trained on the community subgraph. Deep models include **TagGen** [42], which uses GAN and random walk, and **GDSS** [6] and **HGDM** [32], using the Diffusion model. MoDiff-NoM means MoDiff without Motif Tagger, as detailed in section 4.4. We employed the C++ implementations of TASBM and MTM, which were provided by their authors. In TASBM, the number of time windows is set to 10 as suggested. For MTM, we follow the recommended parameters $l_{max} = 4$ and $\delta = 1$. TagGen's window size matches ours. STM's 'duration' is set based on the graph's span. GDSS and HGDM use parameters of corresponding scales and asymmetric samples. In general, we follow the recommended settings. More details on the model introduction and implementation can be found in the appendix C.1.

Table 4: Generation quality by maximum mean discrepancy. The best result is bold. The second result is underlined.

	Dataset	CollegeMsg			Reddit			SuperUser			StackOverflow		
Scale	Model	Deg.	Clus.	Orbit	Deg.	Clus.	Orbit	Deg.	Clus.	Orbit	Deg.	Clus.	Orbit
100	TASBM	0.649	0.179	8.1e-2	0.151	1.159	4.6e-2	0.258	0.790	1.1e-2	0.560	0.476	4.1e-3
	TagGen	0.755	0.844	7.3e-2	0.089	0.399	6.2e-3	0.128	0.412	7.6e-3	0.173	0.238	7.8e-4
	STM	0.623	0.321	**1.3e-4**	0.121	0.742	6.4e-2	0.428	1.018	8.0e-5	0.423	0.523	**1.4e-5**
	MTM-All	1.042	0.291	2.4e-3	1.109	1.797	5.2e-2	1.196	1.255	5.7e-3	1.112	0.786	1.4e-3
	MTM-Sub	0.813	0.208	1.3e-3	0.158	0.905	1.7e-4	0.237	0.446	**1.9e-6**	0.148	0.123	1.4e-4
	GDSS	0.876	0.203	5.4e-3	0.123	0.332	1.9e-2	0.590	0.981	3.4e-3	0.353	0.432	2.9e-2
	HGDM	0.703	0.246	3.9e-5	0.141	0.859	2.6e-3	0.258	0.776	1.2e-3	0.603	0.536	1.7e-4
	MoDiff-NoM	0.393	0.201	9.2e-4	0.149	0.276	1.5e-3	0.157	0.351	2.1e-4	0.149	0.131	1.1e-4
	MoDiff(ours)	**0.213**	**0.143**	4.5e-4	**0.086**	**0.057**	**5.7e-5**	**0.058**	**0.033**	6.7e-5	**0.113**	**0.027**	8.0e-5
500	TASBM	0.623	**0.166**	2.0e-1	1.053	1.617	1.0e-2	0.372	0.986	1.4e-3	0.374	0.672	8.1e-3
	TagGen	0.517	0.220	3.3e-2	0.705	0.523	1.7e-2	0.117	0.569	2.5e-3	0.165	0.352	3.2e-3
	STM	0.658	0.256	**4.4e-7**	0.781	1.409	1.2e-2	0.301	1.077	1.2e-4	0.364	0.756	2.7e-4
	MTM-All	0.801	0.173	3.7e-3	1.564	1.506	2.4e-2	1.098	1.099	1.1e-2	1.146	0.718	3.8e-3
	MTM-Sub	0.876	0.255	4.1e-3	0.865	0.999	3.5e-4	0.153	0.461	5.6e-4	0.203	0.162	3.1e-4
	GDSS	0.716	0.257	1.0e-1	1.095	1.695	7.3e-2	0.342	1.010	8.5e-2	0.274	0.613	1.1e-3
	HGDM	0.920	0.261	5.2e-3	0.939	1.349	1.3e-2	0.540	0.954	2.7e-4	0.171	0.786	**1.0e-6**
	MoDiff-NoM	0.384	0.268	1.8e-3	**0.302**	0.292	1.3e-4	0.430	0.441	7.5e-4	0.251	0.122	3.6e-4
	MoDiff(ours)	**0.280**	0.255	5.6e-4	0.479	**0.066**	**1.1e-5**	**0.095**	**0.084**	**5.5e-5**	**0.095**	**0.043**	8.0e-6
1000	TASBM	1.980	0.079	5.9e-3	0.275	1.776	2.7e-2	0.349	1.136	1.2e-3	0.260	0.749	1.3e-2
	TagGen	1.397	0.040	**3.6e-6**	0.162	0.412	2.0e-5	**0.057**	0.571	6.7e-3	0.089	0.445	3.4e-3
	STM	1.990	0.149	2.9e-4	0.183	1.716	7.8e-2	0.295	1.165	3.8e-4	0.452	0.775	9.9e-4
	MTM-All	**1.211**	**0.027**	8.2e-3	1.050	1.726	7.2e-2	1.081	1.194	1.7e-2	1.090	0.773	7.2e-3
	MTM-Sub	**1.211**	**0.027**	8.2e-3	0.280	1.152	3.9e-4	0.154	0.581	1.1e-3	0.188	0.228	5.1e-4
	GDSS	1.957	0.381	1.4e-2	0.545	1.819	9.7e-2	0.332	1.181	2.6e-3	0.172	0.786	2.3e-2
	HGDM	1.989	0.167	1.0e-3	0.437	1.399	5.3e-3	0.735	0.919	4.9e-4	0.568	0.761	3.3e-4
	MoDiff-NoM	1.694	0.256	3.8e-3	0.164	0.402	3.7e-3	0.454	0.671	1.6e-3	0.186	0.230	3.8e-4
	MoDiff(ours)	1.777	0.113	5.9e-4	**0.147**	**0.041**	**7.6e-8**	0.083	**0.092**	**3.1e-4**	**0.045**	**0.028**	**3.1e-4**

4.2 Preserving Global Graph Property

To answer **Q1**, we report the experimental results in Table 4. As a generation task, Maximum Mean Discrepancy (MMD) [15, 38] evaluates the discrepancy between generated and real graphs across specific distributions. Smaller MMD values indicate better generation quality. The detailed formulas of MMD are in the appendix C.2. The metrics include MMD of degree distribution (*Deg.*), clustering coefficient (*Clus.*), and specific orbit distribution (*Orbit*) on nodes. They characterize the structural patterns of nodes and edges.

As shown in Table 4, **1)** our model, MoDiff, performs better than other approaches in terms of degree distribution. Spectral decomposition ensures consideration of the full graph's degree distribution during generation, while controllable density allows the average degree to simulate the target graph. **2)** MoDiff achieves much better results in clustering and orbit distribution, reducing the MMD of *Deg.* by 10–50% and *Clus.* by 50–90%. This improvement stems from the integrated Hermitian matrix, which effectively encodes edge directions and diverse motif views. Performance on CollegeMsg at large scales is limited due to the dataset's small size, leading to insufficient training data.

In previous methods, TASBM, while efficient, performs poorly on clustering and orbit distributions (e.g., 1.159 *Clus.* on Reddit at scale 100) due to limited structural capture. TagGen excels at degree distribution but also struggles with higher-order structures. On the contrary, STM excels in *Orbit* due to its focus on motifs, but underperforms on degree distribution. Similarly, MTM-Sub struggles to preserve global properties, as shown by high *Deg.* scores like 0.865 for Reddit at scale 500. MTM-All, which extracts subgraphs from the large graph, performs worse than MTM-Sub across all metrics. GDSS struggles with noise due to its diffusion on the adjacency matrix, affecting its ability to maintain global structure. HGDM, adding an auto-encoder, still performs poorly on *Clus.* due to the inability to capture finer structures.

4.3 Preserving Higher-order Structural Statistics

To answer **Q2**, we evaluate different methods w.r.t. preserving higher-order structures. We generate graphs at the scale of 500 nodes in each dataset, same in Section 4.2. Inspired by the previous work [12, 23], we evaluate four structural metrics: the number of connected components (NCC), the size of the largest connected component (LCC), the mean degree (MD), and the mean of the triangle motif count (MTriangles). Figure 4 reports the relative values (e.g., $\frac{\text{NCC of generated}}{\text{NCC of target}}$) for these metrics, where the target

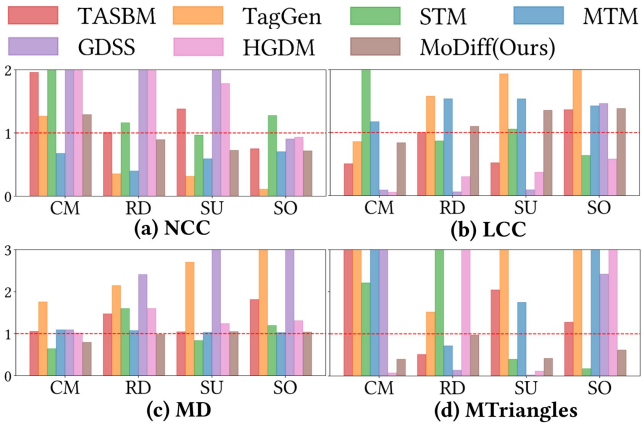


Figure 4: Relative values ($\frac{\text{gen value}}{\text{tgt value}}$) w.r.t. different metrics.

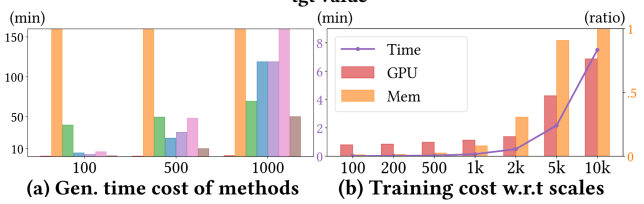


Figure 5: Cost of different methods over different scales.

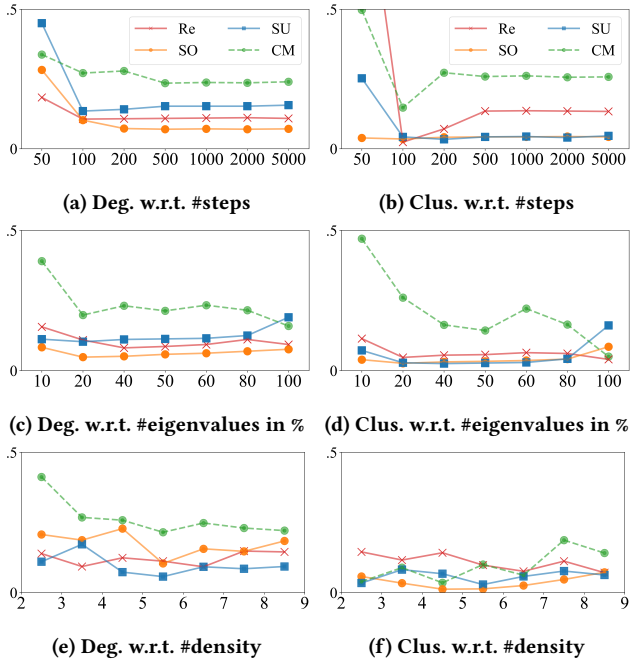


Figure 6: MMD w.r.t. different hyperparameters.

value of 1 is marked with a dashed red line. The closer the result is to the red line, the better the preservation.

As shown in Figure 4, our MoDiff effectively preserves the original graph structure. Despite being a deep model without extra labeling costs, it outperforms both parametric models based on graph statistics and probabilistic models based on motif transitions. **1)** For connected components, MoDiff matches top-performing models like STM and TASBM, accurately capturing their number and size, demonstrating its strong global perception. **2)** For average degree, most models handle it smoothly, except for TagGen due to overly long walk steps and GDSS due to noise. MoDiff also works well with controllable generation. **3)** While previous studies perform poorly on motif count ratios due to their susceptibility to noise, MoDiff demonstrates strong motif simulation capabilities, even when many models struggle in this area.

4.4 Ablation & Controllable Generation

To answer **Q3**, we study the effect of the Motif Tagger module via ablation study. Then we examine the effect of varying the number of diffusion steps and varying k for top-k spectral features. For **Q4**, we access the quality of generated graphs with different densities by controlling λ. The experiments are conducted on 100-node scale subgraphs of four datasets.

Ablation on Motif Tagger. We demonstrate the effectiveness of Motif Tagger in preserving local information through ablation experiments. MoDiff-NoM is trained only with edge directional information without motif views. The results of MoDiff-NoM are presented in Table 4. In most cases, MoDiff outperforms MoDiff-NoM, with more pronounced gaps in *Clus.*. This is because the clustering coefficient is more influenced by higher-order structures captured by motif views. Additionally, without the Motif Tagger, MoDiff-NoM performs similarly to MTM and TagGen, indicating that its advantage is not solely dependent on the diffusion model.

Varying the Number of Diffusion Steps. Figures 6a and 6b show the impact of diffusion steps. With too few steps, the model performs poorly due to insufficient denoising. Around 100 steps, results improve and stabilize, but excessive steps also cause over-denoising and significantly increase generation time. Usually, using around 100 steps ensures efficiency and quality in generation.

Varying the Number of Top Eigenvalues. Spectral decomposition reduces feature dimensions, allowing diffusion models to scale to larger graphs. Figure 6c and 6d illustrates generation results using different percentages of eigenvalues. Too few eigenvalues reduce accuracy, while 20–30% yield strong performance. Beyond this, adding eigenvalues could introduce noise. Usually, using around 20% could enhance both scalability and precision.

Generating Graphs with Different Density. MoDiff controls graph density by adjusting the threshold λ in equation 14. It can be computed based on the training data or temporal trend, or manually set. This allows users to generate graphs with the desired density without expertise. Figures 6e and 6f show the MMD results of generated subgraphs with different densities across 7 intervals (from 2 to 9, spanning 1 unit each). MoDiff is the same one used in Table 4 without being specifically trained for this task. By adjusting the threshold λ, we generate graphs with varying densities and compare them to subgraphs of corresponding density. The generated results maintain the structural properties and do not fluctuate rapidly with density changes.

4.5 Scalability

For **Q5**, we assess the generation and training costs. Figure 5a shows that MoDiff significantly reduces generation time compared

to random walk-based methods or diffusion models using adjacency matrix. This reduction in generation time makes MoDiff highly efficient for integration into downstream tasks, such as GNN training. Figure 5b illustrates bars for memory-usage ratios and a line for per-epoch training time as the graph scale increases. The batch size decreases from 64 at 100 nodes to 2 at 10k nodes. At 10k-node scale with batch size 2, the memory usage approaches the limits of a single 24GB GPU and 512GB of system RAM. Despite this, the spectral diffusion model keeps training costs low and enables fast generation on a single GPU, even for graphs over 10k nodes.

5 Related Work

Here we briefly summarize the previous work on the generative models for temporal networks, diffusion model on graphs, and temporal motifs.

Temporal Graph Generator. Recently, temporal graph generation has gained significant attention, with methods falling into three main categories. First, parameter-based approaches [9, 24, 37, 41] generate graphs directly from parameters. For example, EpiC-Net [26] creates time-evolving networks inspired by epidemiology. TASBM focuses on community analysis and motif simulation[22]. Purohit et al. proposed Structural Temporal Modeling (STM) [23], which generates graphs using the frequencies of basic atomic motifs. Second, many methods are based on transition probabilities. Liu et al. developed the Motif Transition Model (MTM) [12] to simulate the evolution of motifs in temporal graphs, making the generated graphs reflect realistic temporal dynamics. RTGEN++ [16] studies dynamic degree distribution evolution, and Scholtes [25] combines variable-order Markov chains with edges in k-th order pathways. Lastly, deep learning methods are also prominent. Likelihood-based models, such as GANs, VAEs, and score-based models, generate entire graphs holistically [27, 30, 36, 38]. TagGen [42] employs self-attention and discriminator on random walk sequences. Despite their advantages, existing generators often work within modeling scopes that are either too broad or too narrow, and deep learning methods reliant on random walks struggle with large graphs.

Diffusion Model on Graph. Compared to the methods mentioned earlier, diffusion models for graph generation are still in early stages. A classical approach involves applying stochastic differential equations (SDE) to the adjacency matrix, as seen in GDSS and HGDM [6, 31, 32]. However, adding direct noise to the adjacency matrix would significantly disrupt the graph structure. To address this, Luo et al. introduce GSDM [15], which performs low-rank diffusion SDE in the graph spectrum space. It enhances the stability of denoising in the Reversed-SDE process. Additionally, diffusion models are applied to other graph-related tasks. Luo et al. introduced GALA [14], which uses the diffusion model for domain adaptation on undirected graphs. Tian et al. proposed Conda [29] for data augmentation with Encoder. However, these models are not designed for temporal graph generation, facing challenges such as direction labeling, large scale, and structural preservation.

Motif on Temporal Network. Higher-order temporal subgraph structures, i.e., temporal motifs, are small subgraph patterns representing specific structural arrangements of nodes and edges. Motifs are treated as building blocks of large complex graphs [17]. For instance, Alon et al. [1] identified triangle structures, known as "feedforward loops" in genetics networks, as persistence detectors. Motif provides insights into the functional and organizational principles of the network [2, 4, 8, 11, 20], and assists in network trend analysis [10, 11, 13, 34]. Selecting effective motifs thus requires rigorous mining and statistical frameworks [18, 43] to assess significance and frequency, while recent motif-aware learning approaches [33, 35] further automate the discovery of task-relevant subgraphs. Some generators (e.g., TASBM, STM, MTM) use motifs to enhance structure, but they typically focus on single patterns and overlook broader motif dependencies. Persistent homology provides a rigorous approach to the entire graph topology by tracking the topological features across filtrations [19, 44], yet it often requires filtration design and intensive computation. In contrast, motif extracts directly interpretable, adjacency-driven patterns with small overhead, generalizes effectively across diverse graphs.

6 Conclusion

In this paper, we propose MoDiff, a motif-aware spectral diffusion that preserves both global and finer structures during generation. MoDiff encodes the adjacency matrix into Magnetic Hermitian Matrix enriched with motif views, and employs a spectral diffusion process to generate eigenvalues efficiently. We validate MoDiff on four real-world networks across diverse domains. Our method preserves both global properties and high-order structures while enabling generation at varying densities. Our scalability study further shows that MoDiff efficiently handles subgraphs with up to 10k nodes. While MoDiff excels in structural preservation and scalability, it relies on fixed-size graphs and coarse timestamps—trade-offs deemed reasonable given its overall effectiveness. Promising future directions include developing dynamic diffusion schemes to handle variable or larger subgraphs, as well as enhancing the Hermitian encoding with finer temporal resolution or heterogeneous edge semantics to further expand its generative capabilities.

Acknowledgments

Chenhao Ma was partially supported by NSFC under Grant 62302421, Basic and Applied Basic Research Fund in Guangdong Province under Grant 2023A1515011280, 2025A1515010439, Ant Group through CCF-Ant Research Fund, Shenzhen Research Institute of Big Data under grant SIF20240004, and the Guangdong Provincial Key Laboratory of Big Data Computing, The Chinese University of Hong Kong, Shenzhen.

References

[1] Uri Alon. 2007. Network motifs: theory and experimental approaches. *Nature Reviews Genetics* 8, 6 (01 Jun 2007), 450–461. doi:10.1038/nrg2102

[2] Austin R. Benson, David F. Gleich, and Jure Leskovec. 2016. Higher-order organization of complex networks. *Science* 353, 6295 (2016), 163–166. doi:10.1126/science.aad9029 arXiv:https://www.science.org/doi/pdf/10.1126/science.aad9029

[3] Gianluigi Folino, Carla Otranto Godano, and Francesco Sergio Pisani. 2023. An ensemble-based framework for user behaviour anomaly detection and classification for cybersecurity. *The Journal of Supercomputing* 79, 11 (01 Jul 2023), 11660–11683. doi:10.1007/s11227-023-05049-x

[4] Y. Hulovatyy, H. Chen, and T. Milenković. 2015. Exploring the structure and function of temporal networks with dynamic graphlets. *Bioinformatics* 31, 12 (06 2015), i171–i180. doi:10.1093/bioinformatics/btv227

[5] Ruoming Jin, Scott McCallen, and Eivind Almaas. 2007. Trend Motif: A Graph Mining Approach for Analysis of Dynamic Complex Networks. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*. 541–546. doi:10.1109/ICDM.2007.92

[6] Jaehyeong Jo, Seul Lee, and Sung Ju Hwang. 2022. Score-based generative modeling of graphs via the system of stochastic differential equations. In *International conference on machine learning*. PMLR, 10362–10383.

[7] Zahra Razaghi Moghadam Kashani, Hayedeh Ahrabian, Elahe Elahi, Abbas Nowzari-Dalini, Elnaz Saberi Ansari, Sahar Asadi, Shahin Mohammadi, Falk Schreiber, and Ali Masoudi-Nejad. 2009. Kavosh: a new algorithm for finding network motifs. *BMC bioinformatics* 10 (2009), 1–12.

[8] Lauri Kovanen, Márton Karsai, Kimmo K. Kaski, János Kertész, and Jari Saramäki. 2011. Temporal motifs in time-dependent networks. *Journal of Statistical Mechanics: Theory and Experiment* 2011 (2011), P11005.

[9] Jure Leskovec, Deepayan Chakrabarti, Jon Kleinberg, Christos Faloutsos, and Zoubin Ghahramani. 2010. Kronecker Graphs: An Approach to Modeling Networks. *J. Mach. Learn. Res.* 11 (March 2010), 985–1042.

[10] Penghang Liu, Rupam Acharyya, Robert E. Tillman, Shunya Kimura, Naoki Masuda, and Ahmet Erdem Sarıyüce. 2023. Temporal Motifs for Financial Networks: A Study on Mercari, JPMC, and Venmo Platforms. *ArXiv* abs/2301.07791 (2023).

[11] Penghang Liu, Valerio Guarrasi, and Ahmet Erdem Sariyüce. 2022. Temporal Network Motifs: Models, Limitations, Evaluation (Extended abstract). In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. 1531–1532. doi:10.1109/ICDE53745.2022.00142

[12] Penghang Liu and Ahmet Erdem Sariyüce. 2023. Using Motif Transitions for Temporal Graph Generation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Long Beach, CA, USA) *(KDD '23)*. Association for Computing Machinery, New York, NY, USA, 1501–1511. doi:10.1145/3580305.3599540

[13] Yufei Liu, Jia Wu, and Jie Cao. 2024. SBP-GCA: Social Behavior Prediction via Graph Contrastive Learning With Attention. *IEEE Transactions on Artificial Intelligence* 5, 9 (2024), 4708–4722. doi:10.1109/TAI.2024.3395574

[14] Junyu Luo, Yiyang Gu, Xiao Luo, Wei Ju, Zhiping Xiao, Yusheng Zhao, Jingyang Yuan, and Ming Zhang. 2024. GALA: Graph Diffusion-based Alignment with Jigsaw for Source-free Domain Adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2024), 1–14. doi:10.1109/TPAMI.2024.3416372

[15] Tianze Luo, Zhanfeng Mo, and Sinno Jialin Pan. 2023. Fast graph generation via spectral diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46, 5 (2023), 3496–3508.

[16] Maria Massri, Zoltan Miklos, Philippe Raipin, Pierre Meye, Amaury Bouchra Pilet, and Thomas Hassan. 2023. RTGEN++: A Relative Temporal Graph GENerator. *Future Generation Computer Systems* 146 (2023), 139–155. doi:10.1016/j.future.2023.03.023

[17] Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. 2002. Network motifs: simple building blocks of complex networks. *Science* 298, 5594 (2002), 824–827.

[18] Mohammad Matin Najafi, Chenhao Ma, Xiaodong Li, Reynold Cheng, and Laks V. S. Lakshmanan. 2023. MOSER: Scalable Network Motif Discovery Using Serial Test. *Proc. VLDB Endow.* 17, 3 (nov 2023), 591–603. doi:10.14778/3632093.3632118

[19] Viet The Nguyen, Duy Anh Pham, An Thai Le, Jans Peter, and Gunther Gust. 2025. Persistent Homology-induced Graph Ensembles for Time Series Regressions. arXiv:2503.14240 [cs.LG] https://arxiv.org/abs/2503.14240

[20] Ashwin Paranjape, Austin R. Benson, and Jure Leskovec. 2017. Motifs in Temporal Networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining* (Cambridge, United Kingdom) *(WSDM '17)*. Association for Computing Machinery, New York, NY, USA, 601–610. doi:10.1145/3018661.3018731

[21] Ali Pinar, C. Seshadhri, and Vaidyanathan Vishal. 2017. ESCAPE: Efficiently Counting All 5-Vertex Subgraphs *(WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 1431–1440. doi:10.1145/3038912.3052597

[22] Alexandra Porter, Baharan Mirzasoleiman, and Jure Leskovec. 2022. Analytical Models for Motifs in Temporal Networks. In *Companion Proceedings of the Web Conference 2022* (Virtual Event, Lyon, France) *(WWW '22)*. Association for Computing Machinery, New York, NY, USA, 903–909. doi:10.1145/3487553.3524669

[23] Sumit Purohit. 2018. Temporal Graph Generation Based on a Distribution of Temporal Motifs. In *In Proceedings of the 14th International Workshop on Mining and Learning with Graphs (2018)*, Vol. 7.

[24] Garry Robins, Pip Pattison, Yuval Kalish, and Dean Lusher. 2007. An introduction to exponential random graph (p*) models for social networks. *Social Networks* 29, 2 (2007), 173–191. doi:10.1016/j.socnet.2006.08.002 Special Section: Advances in Exponential Random Graph (p*) Models.

[25] Ingo Scholtes. 2017. When is a Network a Network? Multi-Order Graphical Model Selection in Pathways and Temporal Networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Halifax, NS, Canada) *(KDD '17)*. Association for Computing Machinery, New York, NY, USA, 1037–1046. doi:10.1145/3097983.3098145

[26] Pouyan Shirzadian, Blessy Antony, Akshaykumar G Gattani, Nure Tasnina, and Lenwood S Heath. 2023. A time evolving online social network generation algorithm. *Scientific Reports* 13, 1 (2023), 2395.

[27] Martin Simonovsky and Nikos Komodakis. 2018. Graphvae: Towards generation of small graphs using variational autoencoders. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27*. Springer, 412–422.

[28] Jianhua Sun, Qinhong Jiang, and Cewu Lu. 2020. Recursive social behavior graph for trajectory prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 660–669.

[29] Yuxing Tian, Aiwen Jiang, Qi Huang, Jian Guo, and Yiyan Qi. 2024. Latent Diffusion-based Data Augmentation for Continuous-Time Dynamic Graph Model *(KDD '24)*. Association for Computing Machinery, New York, NY, USA, 2900–2911. doi:10.1145/3637528.3671863

[30] Hongwei Wang, Jialin Wang, Jia Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Wenjie Li, Xing Xie, and Minyi Guo. 2021. Learning Graph Representation With Generative Adversarial Nets. *IEEE Transactions on Knowledge and Data Engineering* 33, 8 (2021), 3090–3103. doi:10.1109/TKDE.2019.2961882

[31] Yujie Wang, Shuo Zhang, Junda Ye, Hao Peng, and Li Sun. 2024. A Mixed-Curvature Graph Diffusion Model. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management* (Boise, ID, USA) *(CIKM '24)*. Association for Computing Machinery, New York, NY, USA, 2482–2492. doi:10.1145/3627673.3679708

[32] Lingfeng Wen, Xuan Tang, Mingjie Ouyang, Xiangxiang Shen, Jian Yang, Daxin Zhu, Mingsong Chen, and Xian Wei. 2024. Hyperbolic Graph Diffusion Model. *Proceedings of the AAAI Conference on Artificial Intelligence* 38, 14 (Mar. 2024), 15823–15831. doi:10.1609/aaai.v38i14.29512

[33] Chunjing Xiao, Shikang Pang, Wenxin Tai, Yanlong Huang, Goce Trajcevski, and Fan Zhou. 2024. Motif-Consistent Counterfactuals with Adversarial Refinement for Graph-level Anomaly Detection. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24)*. ACM, 3518–3526. doi:10.1145/3637528.3672050

[34] Ling Xing, Shiyu Li, Qi Zhang, Honghai Wu, Huahong Ma, and Xiaohui Zhang. 2024. A survey on social network's anomalous behavior detection. *Complex & Intelligent Systems* 10, 4 (01 Aug 2024), 5917–5932. doi:10.1007/s40747-024-01446-8

[35] Pengwei Yan, Kaisong Song, Zhuoren Jiang, Yangyang Kang, Tianqianjin Lin, Changlong Sun, and Xiaozhong Liu. 2024. Empowering dual-level graph self-supervised pretraining with motif discovery. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence (AAAI'24/IAAI'24/EAAI'24)*. AAAI Press, Article 1026, 9 pages. doi:10.1609/aaai.v38i8.28774

[36] Jiaxuan You, Rex Ying, Xiang Ren, William Hamilton, and Jure Leskovec. 2018. GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 5708–5717.

[37] Stephen J. Young and Edward R. Scheinerman. 2007. Random Dot Product Graph Models for Social Networks. In *Algorithms and Models for the Web-Graph*, Anthony Bonato and Fan R. K. Chung (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 138–149.

[38] Liming Zhang, Liang Zhao, Shan Qin, Dieter Pfoser, and Chen Ling. 2021. TG-GAN: Continuous-time Temporal Graph Deep Generative Models with Time-Validity Constraints. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia) *(WWW '21)*. Association for Computing Machinery, New York, NY, USA, 2104–2116. doi:10.1145/3442381.3449818

[39] Yi-Qing Zhang, Xiang Li, Jian Xu, and Athanasios V. Vasilakos. 2015. Human Interactive Patterns in Temporal Networks. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45, 2 (2015), 214–222. doi:10.1109/TSMC.2014.2360505

[40] Qiankun Zhao, Yuan Tian, Qi He, Nuria Oliver, Ruoming Jin, and Wang-Chien Lee. 2010. Communication motifs: a tool to characterize social communications. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management* (Toronto, ON, Canada) *(CIKM '10)*. Association for Computing Machinery, New York, NY, USA, 1645–1648. doi:10.1145/1871437.1871694

[41] Shuwen Zheng, Chaokun Wang, Cheng Wu, Yunkai Lou, Hao Feng, and Xuran Yang. 2024. Temporal Graph Generation Featuring Time-Bound Communities. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*. 2365–2378. doi:10.1109/ICDE60146.2024.00187

[42] Dawei Zhou, Lecheng Zheng, Jiawei Han, and Jingrui He. 2020. A Data-Driven Graph Generative Model for Temporal Interaction Networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Virtual Event, CA, USA) *(KDD '20)*. Association for Computing Machinery, New York, NY, USA, 401–411. doi:10.1145/3394486.3403082

[43] Yingli Zhou, Yixiang Fang, Chenhao Ma, Tianci Hou, and Xin Huang. 2024. Efficient Maximal Motif-Clique Enumeration over Large Heterogeneous Information Networks. *Proceedings of the VLDB Endowment* 17, 11 (2024), 2946–2959.

[44] Afra Zomorodian and Gunnar Carlsson. 2004. Computing persistent homology. In *Proceedings of the Twentieth Annual Symposium on Computational Geometry* (Brooklyn, New York, USA) *(SCG '04)*. Association for Computing Machinery, New York, NY, USA, 347–356. doi:10.1145/997817.997870

Appendix

A Graph Spectral Analysis

Figures 7 illustrate the spectral decomposition features of each dataset across multiple scales. From top to bottom, the four rows correspond to the datasets: CollegeMsg, Reddit, SuperUser, and StackOverflow. Within each row, the three groups of plots (from left to right) represent increasing subgraph sizes: 100-node, 500-node, and 1000-node scales. Each group of plots compares the eigenvalue spectra of three graph variants: the original graph, the graph with edges enhanced based on triangle motifs, and a randomly enhanced version where the same proportion of edges are randomly marked. Here, Original refers to the unmodified graph; Triangles refers to the graph where edges involved in triangle motifs are identified and enhanced; and Randomized refers to a control setting where an equal number of edges are randomly selected and marked, eliminating motif value influence. For high-frequency spectral components, triangle-based motif marking not only has a noticeable effect but also exhibits a significantly stronger growth trend compared to random assignment.

B Diffusion Model

B.1 MoDiff Compuation Complexity

Regarding complexity, we summarize it based on the process in Figure 2, where N denotes the number of nodes in G_k.

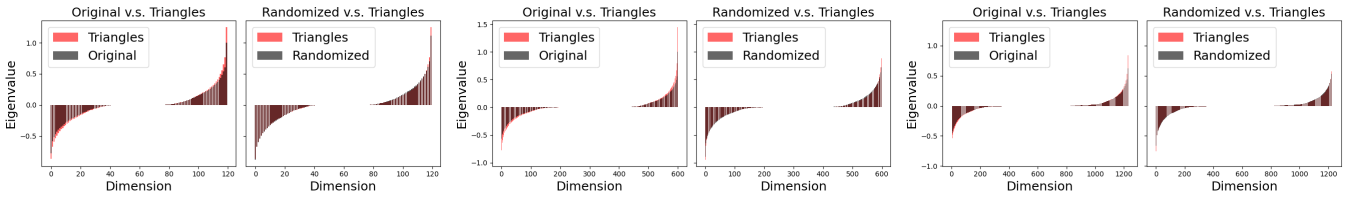
Motif Extraction in Motif Tagger employs MFinder with a complexity of $O(N * d^m)$, where m is the motif size and d is the maximum degree, which is always 3 or 4.

Encoding integrated Hermitian matrix with various motifs involves only matrix operations, with a complexity of $O(N^2)$.

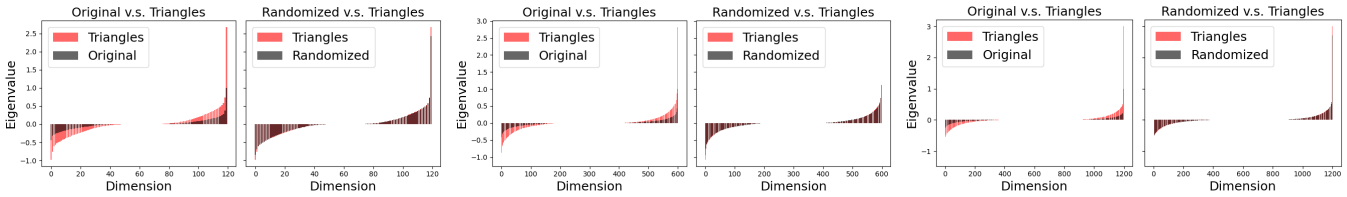
Spectral Decomposition has a complexity of approximately $O(KN^2)$, using methods like the Lanczos algorithm, where K represents the dimension of selected eigenvalues.

The **diffusion model** consists of noise addition and denoising. In MoDiff's denoising model, both GCN and MLP are utilized.

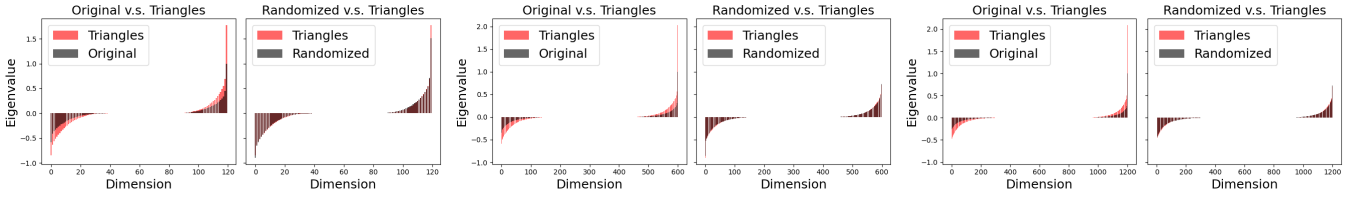
-Noise Process:



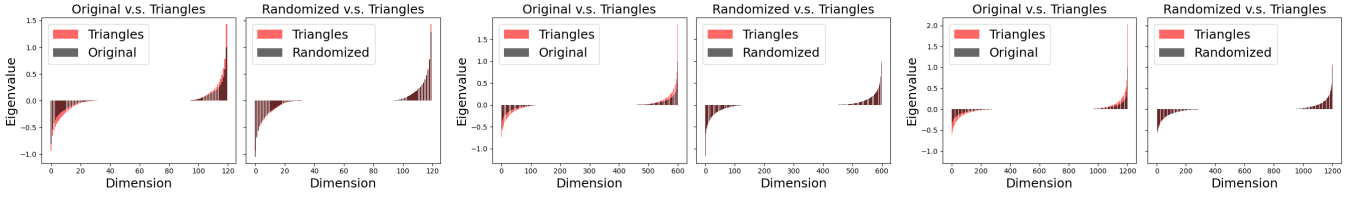
(a) CollegeMsg



(b) Reddit



(c) SuperUser



(d) StackOverflow

Figure 7: Spectral Analysis Comparison across four datasets and three scales.

The time complexity for the noise step is typically $O(T_{\text{n}} * N)$, where T_{n} is the number of noise steps.

-Denoising Process:

The denoising process involves performing T_{den} iterations. The GCN and MLP are used as follows: For GCN, the input is an $N \times a$-dimensional $\mathcal{X}$ and $N \times N$-dimensional Adjacency matrix. With L layers, the total complexity for GCN is: $O(L*N^2*a)$. The MLP takes the $N \times a$-dimensional output from the GCN and $N \times K$-dimensional eigenvalues as input. With h layers, the total complexity for the MLP is: $O(h * N * (a + K))$.

Therefore, the time complexity for the denoise step is: $O(T_{\text{den}} * (L*N^2*a+h*N*(a+K)))$ Due to the use of spectral decomposition, we only need to generate $(a + K)$ features in the MLP, rather than the complete $N \times N$ adjacency matrix.

Graph Conversion, similar to Motif Tagger but without decomposition, operates with a complexity of $O(N^2)$.

Overall, the Diffusion Model and Spectral Decomposition contribute the most to computational cost.

C Experiment

C.1 Baselines

Due to space limitations, we introduce each baseline method in detail here and attach their code links.

TASBM [22]: TASBM uses a block model approach to capture temporal motifs by clustering nodes based on temporal activity states and assigning probabilities for temporal interactions. It analytically models motif frequencies with closed-form expressions, avoiding the need for random sampling. This method efficiently simulates evolving temporal graphs. The code of TASBM is available at: https://github.com/aporter468/motifsanalyticalmodel

TagGen [42]: TagGen employs a deep generative framework that models temporal graphs by learning from system logs of timestamped interactions. It uses a bi-level self-attention mechanism to generate temporal random walks, integrating structural and temporal contexts. A discriminator refines these walks, ensuring plausible temporal networks. This approach effectively captures both structural and temporal characteristics. The code of TagGen is available at: https://github.com/davidchouzdw/TagGen

STM [23]: STM focuses on generating temporal graphs using a temporal-motif-based approach. It models real-world temporal graph dynamics and enables the generation of high-fidelity synthetic graphs. The method is versatile, supporting multi-type heterogeneous graphs and parameterized generation of linear, sub-linear, and super-linear preferential attachment graphs, ensuring adaptability for various graph structures. The code of STM is available at: https://github.com/temporal-graphs/STM

MTM-All [12] MTM models temporal graph generation through motif transition processes, leveraging temporal motifs to capture both global and local graph features. It computes transition probabilities and rates from the input graph and simulates events based on these properties. This approach effectively preserves temporal graph statistics.

MTM-Sub [12] learns the transition possibility of motifs to simulate the evolution of the temporal graph *within the subgraphs extracted*. The code of MTM is available at: https://github.com/erdemUB/KDD23-MTM

GDSS [6]: GDSS introduces a score-based generative model for graphs using a continuous-time framework. It employs a novel graph diffusion process, modeled via stochastic differential equations (SDEs), to jointly capture the distribution of nodes and edges. By tailoring score-matching objectives for this diffusion process and designing an efficient solver for the reverse SDEs, GDSS enables effective graph generation. The model demonstrates strong performance across undirectional graph datasets. The code of GDSS is available at: https://github.com/harryjo97/GDSS

HGDM [32]: HGDM leverages hyperbolic space for graph generation, addressing the limitations of Euclidean embeddings in capturing hierarchical graph structures. The model combines an auto-encoder to embed nodes in hyperbolic space with a diffusion model operating in this latent space. By incorporating edge information into a hyperbolic potential node space, HGDM effectively models hierarchical distributions. Experiments show it significantly improves graph generation quality, particularly for graphs with strong hierarchical characteristics. The code of HGDM is available at: https://github.com/LF-WEN/HGDM/

C.2 Maximum Mean Discrepancy

Maximum Mean Discrepancy (MMD) quantifies the distance between two probability distributions X and Y through kernel-based comparison. For graph distributions, MMD is defined as:

$$\text{MMD}^2(X, Y) = \frac{1}{n^2} \sum_{i,j} k(X_i, X_j) - \frac{2}{nm} \sum_{i,j} k(X_i, Y_j) + \frac{1}{m^2} \sum_{j,k} k(Y_j, Y_k)$$

Where X_i and Y_j represent specific graph subspace distributions (e.g., degree distributions), n and m are sample sizes of X and Y, and $k(x, y)$ is a kernel function measuring similarity. The three terms capture: 1. Within-distribution X similarities; 2. Cross-distribution similarities; 3. Within-distribution Y similarities. The kernels enable non-parametric distribution comparisons across various domains. Common kernel choices in MMD include Gaussian Kernel, Laplacian Kernel, and Polynomial Kernel. We use Gaussian Kernel in this work: $k(x, y)_{Gaussian} = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right)$, which is most popular and captures local similarities.