FreqTAD: Multi-scale Frequency Encoding and Time-Frequency Attention for Anomaly Detection in Dynamic Graphs

Chao Li[1, 2], **Runshuo Liu**[1], **Zhongying Zhao**[1*], **Hui Zhou**[1], **Qingtian Zeng**[1*]
[1]College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China
[2]College of Electronic and Information Engineering, Shandong University of Science and Technology, Qingdao 266590, China
lichao@sdust.edu.cn, rsliu@sdust.edu.cn, zzysuin@163.com, zhouhui1026@foxmail.com, qtzeng@163.com

Abstract

Anomaly detection in dynamic graphs aims to capture the dynamic evolution characteristics of graphs, and then identify abnormal behaviors that deviate from normal patterns. However, previous studies fail to decouple periodic and bursty information during the time encoding process, which hinders their performances. In addition, most existing methods use attention mechanisms to capture the importance of time points. They fail to leverage the normal and abnormal characteristics in the frequency domain. To address the above issues, we propose a model that integrates multi-scale **Freq**uency encoding with **T**ime-frequency Attention for **A**nomaly **D**etection in dynamic graphs, named **FreqTAD**. We design a multi-scale frequency encoder that decomposes time series into distinct periodic and bursty components. Moreover, we present an effective time-frequency attention mechanism that focuses on frequency components to differentiate frequency-domain features of normal and abnormal behaviors. Experiments on four datasets demonstrate the superior performance of FreqTAD in both detection accuracy and computational efficiency.

Code —
https://github.com/ZZY-GraphMiningLab/FreqTAD

Introduction

Various real-world graphs exhibit dynamic evolutionary characteristics (Chen et al. 2024) and contain abnormal behavior patterns (Chen et al. 2024). For instance, misinformation propagation in social networks (Li et al. 2024) and fraudulent transactions in financial networks (Wang et al. 2025) have attracted widespread attention (Su, Zou, and Wu 2024). However, traditional anomaly detection methods designed for static graphs ignore temporal evolutionary information (Tian, Qi, and Guo 2024). They cannot effectively capture anomalous behavior patterns in dynamic environments. Therefore, anomaly detection in dynamic graphs has emerged as a natural solution (Zheng et al. 2025).

In the early exploration, researchers mainly adopted statistical methods (Chang et al. 2021) to identify anomalous patterns in graphs. For example, MIDA (Bhatia et al. 2020) and SedanSpot (Eswaran and Faloutsos 2018) are two representative statistics-based methods that detect anomalies through micro-clustering techniques and edge stream statistical analysis. However, statistical methods have limitations in capturing complex nonlinear relationships (Bhatia et al. 2023) and high-dimensional feature interactions. This makes it difficult to effectively model complex anomalous patterns in modern graph data. Subsequently, deep learning techniques were gradually introduced into the field of anomaly detection in dynamic graphs. Methods based on deep neural networks such as SLADE (Lee, Kim, and Shin 2024) and GeneralDyG (Yang, Zhao, and Shen 2025) learn deep representations of nodes through multi-layer nonlinear transformations, enabling them to capture more complex anomalous patterns. Although these methods have shown significant performance, existing methods still face the following two challenges:

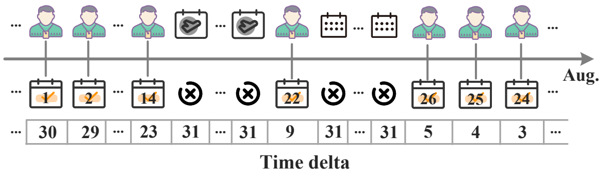


Figure 1: An employee's attendance records in August. The time delta in the figure represents the number of days from the check-in date to the end of the month. The data demonstrates that employee daily check-in behavior exhibits obvious periodicity.

(1) Existing time encoding strategies are unable to adaptively handle multi-scale frequency characteristics in time series, which constrains modeling performance. Taking the employee's attendance scenario as an example, as shown in Figure 1, employees' daily check-in behaviors exhibit obvious periodic patterns (low and medium frequency features). However, when employees experience consecutive absences and then return to work, this anomalous behavior pattern (high frequency features) disrupts the original periodic regularity. Existing time encoding methods mainly focus on basic sequence modeling. For example, TGN (Rossi et al. 2020) uses trigonometric functions with predefined frequencies to encode time delta. SLADE (Lee, Kim, and Shin 2024) adopts the improved time encoding method based on Bochner's theorem from GraphMixer (Cong et al. 2023), us-

---

[*]Corresponding Authors.

ing only a single linear layer to process all frequency domain features. However, these methods focus on time delta computation and cannot clearly distinguish and adaptively process different frequency domain features. Consequently, they overlook the frequency domain differences between periodic and bursty characteristics in time series, lacking multi-scale frequency modeling capabilities, which limits the representational ability of time encoding.

(2) Existing attention mechanisms cannot distinguish the importance of different frequency components in time series, limiting their ability to capture complex temporal dynamics. In real-world dynamic graphs, temporal patterns across different frequency scales are often intertwined. However, existing methods such as TGAT (Xu et al. 2020) feed temporal features into attention mechanisms and can only evaluate the importance of individual time points in the temporal dimension. Similarly, GeneralDyG (Yang, Zhao, and Shen 2025) adopts Transformer modules to capture temporal information. This approach focuses on temporal patterns without specifically targeting different frequency components within the time series. As a result, these methods treat normal and abnormal frequency components equally, causing the model to lack frequency domain perception capability when fusing node features, neighbor features, and temporal features. This limitation constrains precise modeling of complex temporal dynamics.

To address the above problems, we propose a model that integrates multi-scale **Freq**uency encoding with **T**ime-frequency Attention for **A**nomaly **D**etection in dynamic graphs, called **FreqTAD**. Specifically, we design a multi-scale frequency domain time encoder that transforms time delta information into high, medium, and low frequency components with adaptive weight adjustment. In addition, we construct a time-frequency attention module that considers both temporal and frequency dimensions to provide more comprehensive node features. Finally, we adopt a contrastive learning strategy to optimize node feature representations for anomaly detection tasks. The main contributions of this work are as follows:

- We propose a method for anomaly detection in dynamic graphs called FreqTAD that identifies anomalous behavior patterns through frequency domain analysis and time-frequency attention mechanisms.
- We design a multi-scale frequency encoder that can effectively capture periodic and bursty characteristics in time series.
- We construct a time-frequency attention mechanism that can consider importance allocation in both temporal and frequency dimensions.
- Experimental results on four datasets demonstrate that FreqTAD achieves superior performance in both anomaly detection accuracy and computational efficiency.

Related Work

Anomaly Detection in Dynamic Graphs

Anomaly detection in dynamic graphs has emerged as a rapidly growing research area. Researchers have explored various deep neural networks for applications (Fang et al. 2023; Fu et al. 2025). SAD (Tian et al. 2023) and EL2-DGAD (Chen et al. 2025) proposed semi-supervised anomaly detection methods, but semi-supervised approaches still require high-quality labeled data. SLADE (Lee, Kim, and Shin 2024) detects dynamic anomalies through self-supervised learning, avoiding dependence on labels. GeneralDyG (Yang, Zhao, and Shen 2025) designed a general framework for anomaly detection in dynamic graphs, improving the generalization capability of such method. AnoEdge (Bhatia et al. 2023) focuses on anomalous edge detection in streaming graphs, optimized for specific scenarios. Although these methods perform well in their respective application scenarios, they generally lack frequency domain analysis capabilities in temporal modeling, making it difficult to effectively distinguish periodic and bursty behavioral patterns.

Frequency Domain Analysis in Graph Learning

Frequency domain analysis has been widely applied in graph signal processing and graph neural networks (Liu and Du 2025; Bo et al. 2021). ChebNet (Defferrard, Bresson, and Vandergheynst 2016) defines convolution operations in the frequency domain through Chebyshev polynomials. SpectralGCN (Bruna et al. 2014) performs spectral analysis based on the graph Laplacian matrix. In the field of anomaly detection in dynamic graphs, F-FADE (Chang et al. 2021) first introduced the concept of frequency decomposition into this domain, capturing anomalous patterns in edge streams through frequency domain analysis. However, its frequency decomposition strategy is relatively simple and lacks adaptive modeling of the importance of different frequency components.

Methodology

Preliminaries

Notations. A dynamic graph can record the complete temporal information of a dynamic network, represented as $\mathcal{G} = \{(u_i, v_j, t, \mathbf{e}_{ij}(t)); i, j = 1, 2, \ldots, n\}$, where u_i, v_j denote nodes in the network, t represents the timestamp, and $\mathbf{e}_{ij}(t)$ represents the edge feature vector at time t.

Problem Definition. The goal of anomaly detection in dynamic graphs is to construct a function $f : \mathcal{G}(t) \rightarrow \mathbb{R}^{|\mathcal{V}(t)|}$ that assigns an anomaly score $s_v(t)$ to each node $v \in \mathcal{V}(t)$ at time t, enabling the identification of various types of anomalies in dynamic graphs.

Framework of FreqTAD

As shown in Figure 2, FreqTAD consists of four key components: 1) Dynamic Graph Processing, 2) Multi-scale Frequency Encoding, 3) Time-Frequency Attention, and 4) Contrastive Learning. Details are as follows.

Dynamic Graph Processing

The dynamic graph processing module is responsible for extracting historical interaction information and temporal features from the original dynamic graph, providing fundamen-

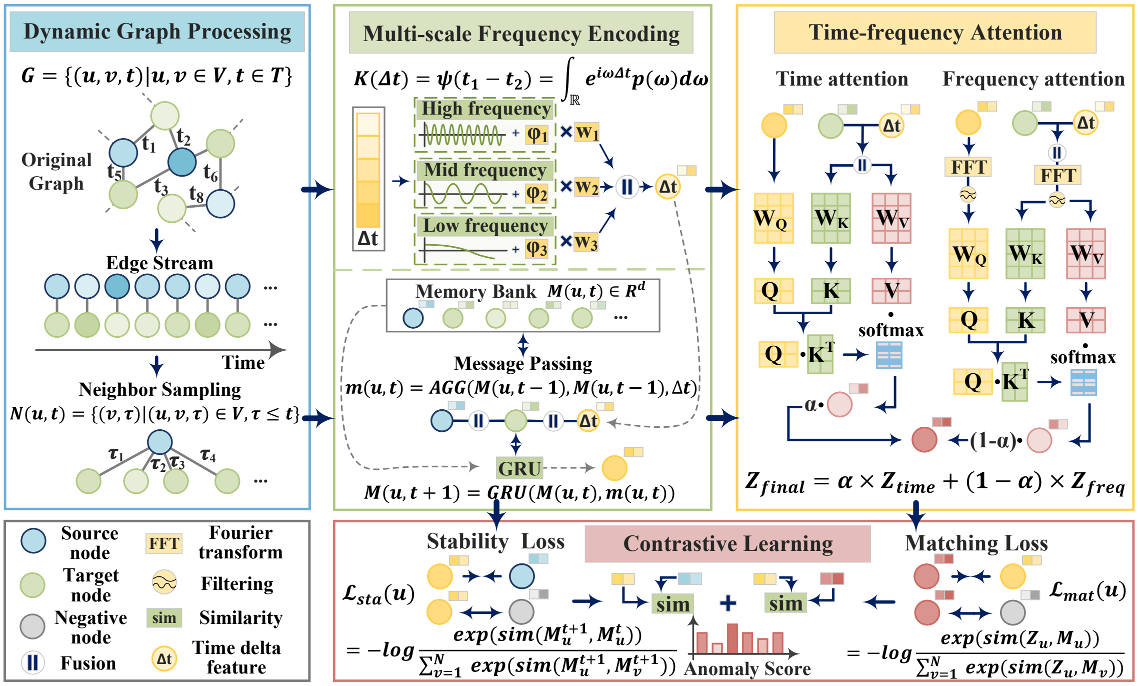


Figure 2: The framework of FreqTAD. The model first performs neighbor sampling and time delta extraction using the dynamic graph processing module, then inputs temporal information into the multi-scale frequency encoding module for frequency domain decomposition and memory update. Subsequently, it computes attention weights in both temporal and frequency dimensions via the time-frequency attention module, and finally adopts a contrastive learning strategy for anomaly detection.

tal data for subsequent frequency encoding and anomaly detection.

Given a dynamic graph $\mathcal{G}$, all edges are arranged in chronological order to form an edge stream:

$$\mathcal{S} = \{(u_1, v_1, t_1), (u_2, v_2, t_2), \ldots, (u_m, v_m, t_m)\}, \quad (1)$$

where $t_1 \leq t_2 \leq \ldots \leq t_m$.

For a target node u at time t, its historical interaction neighbors are extracted through the edge stream:

$$\mathcal{N}(u, t) = \{(v, \tau) | (u, v, \tau) \in \mathcal{S}, \tau \leq t\}. \quad (2)$$

To control computational complexity, a fixed window size sampling strategy is adopted, selecting the most recent K neighbors:

$$\mathcal{N}_K(u, t) = \text{TopK}(\mathcal{N}(u, t), K). \quad (3)$$

For each neighbor interaction $(v, \tau) \in \mathcal{N}_K(u, t)$, the time delta is computed as the basic temporal feature:

$$\Delta t = t - \tau. \quad (4)$$

These time delta information reflects the temporal intervals of node interactions, containing rich temporal dynamic information that will serve as input to the multi-scale frequency encoding module.

Multi-scale Frequency Encoding

Existing time encoding methods mainly focus on basic sequence modeling of time deltas, ignoring the differences between different frequency components in time series. Our designed multi-scale frequency encoding module can adaptively process different frequency domain features in temporal information and maintain historical states of nodes through a memory update mechanism.

To map time delta information to the frequency domain, we adopt the kernel function (Xu et al. 2020):

$$K(\Delta t) = \int e^{i\omega t} p(\omega) d\omega, \quad (5)$$

where $p(\omega)$ is the frequency density function. In practical implementation, we use the real part of cosine functions to approximate this mapping:

$$\mathbb{E}[\cos(\omega \Delta t)] = \frac{1}{d} \sum_{i=1}^{d} \cos(\omega_i \Delta t). \quad (6)$$

We divide the frequency domain information into three different frequency scales, each specifically capturing specific types of temporal patterns:

Low frequency component ($\omega_{\text{low}} = 1/10^0 \sim 1/10^4$) captures long-term periodic patterns, with encoding function:

$$\mathbf{h}_{\text{low}} = \cos(\mathbf{W}_{\text{low}} \cdot \Delta t + \phi_1) \odot \mathbf{w}_1. \quad (7)$$

Mid frequency component ($\omega_{\text{mid}} = 1/10^3 \sim 1/10^7$) captures medium-term transitional feature patterns, with encoding function:

$$\mathbf{h}_{\text{mid}} = \sin(\mathbf{W}_{\text{mid}} \cdot \Delta t + \phi_2) \odot \mathbf{w}_2. \quad (8)$$

High frequency component ($\omega_{\text{high}} = 1/10^6 \sim 1/10^{10}$) captures bursty events and short-term anomalous behavior features, with encoding function:

$$\mathbf{h}_{\text{high}} = \cos(\mathbf{W}_{\text{high}} \cdot \Delta t + \phi_3) \odot \mathbf{w}_3. \tag{9}$$

where $\odot$ denotes element-wise multiplication, $\mathbf{w}_i$ is the weight vector for the i-th frequency component, and ϕ_i is the phase shift parameter.

To adaptively adjust the importance of different frequency components, we introduce learnable frequency weight parameters:

$$\mathbf{w} = [\mathbf{w}_1; \mathbf{w}_2; \mathbf{w}_3] \in \mathbb{R}^{3d}, \tag{10}$$

$$\phi = [\phi_1, \phi_2, \phi_3]^\top \in \mathbb{R}^{3d}. \tag{11}$$

The final temporal feature is obtained by concatenating the three frequency components:

$$\mathbf{h}_{\text{time}} = [\mathbf{h}_{\text{low}}; \mathbf{h}_{\text{mid}}; \mathbf{h}_{\text{high}}] \in \mathbb{R}^d. \tag{12}$$

After obtaining the temporal features $\mathbf{h}_{\text{time}}$ from multi-scale frequency encoding, we introduce a memory update mechanism to maintain historical state information of nodes. Each node u maintains a state vector $\mathbf{M}(u,t) \in \mathbb{R}^d$ in the memory bank.

Node u generates messages by aggregating memory states of its neighbors and frequency-encoded features:

$$\mathbf{m}(u,t) = \text{AGG}(\mathbf{M}(u,t) \| \mathbf{M}(v,t-1) \| \mathbf{h}_{\text{time}}), \tag{13}$$

where AGG operation can be implemented using an MLP.

The node's memory state is updated using a Gated Recurrent Unit (GRU):

$$\mathbf{M}(u,t+1) = \text{GRU}(\mathbf{M}(u,t), \mathbf{m}(u,t)). \tag{14}$$

Time-Frequency Attention

Existing attention mechanisms only focus on importance allocation in the temporal dimension, ignoring differences in frequency components. We propose a time-frequency attention mechanism that simultaneously allocates attention weights in both temporal and frequency dimensions, capable of highlighting the importance of anomalous frequency components.

Time Attention. Time attention focuses on the importance of different time points in historical interaction sequences. Given updated memory state sequences and neighbor features, the temporal attention is computed as follows Eqn.(15):

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}, \tag{15}$$

where the query, key, and value matrices are obtained through linear transformations:

$$\mathbf{Q}_{\text{time}} = \mathbf{M}(u,t+1)\mathbf{W}_Q, \tag{16}$$

$$\mathbf{K}_{\text{time}} = [\mathbf{F}_n \| \mathbf{h}_{\text{time}}]\mathbf{W}_K, \tag{17}$$

$$\mathbf{V}_{\text{time}} = [\mathbf{F}_n \| \mathbf{h}_{\text{time}}]\mathbf{W}_V, \tag{18}$$

where $\mathbf{F}_n$ represents neighbor node features, and $\|$ denotes feature concatenation.

The output of temporal attention is:

$$\mathbf{Z}_{\text{time}} = \text{Attention}(\mathbf{Q}_{\text{time}}, \mathbf{K}_{\text{time}}, \mathbf{V}_{\text{time}}). \tag{19}$$

Frequency Attention. Frequency attention is transformed to the frequency domain through FFT, focusing on the importance of different frequency components. For a sequence of length N, its discrete Fourier transform is defined as:

$$\mathbf{H}_k = \sum_{n=0}^{N-1} \mathbf{h}_n e^{-i2\pi kn/N}, \quad k = (0, 1, \ldots, N-1), \tag{20}$$

where $\mathbf{H}_k$ is the k-th frequency component and i is the imaginary unit, $\mathbf{h}_n$ represents the feature vector at time step n.

We transform node memory features to the frequency domain:

$$\mathbf{h}_f = \text{FFT}(\mathbf{M}(u,t+1)). \tag{21}$$

After FFT transformation, we extract real and imaginary parts:

$$\mathbf{h}_{f,\text{real}} = \text{Re}(\mathbf{h}_f), \tag{22}$$

$$\mathbf{h}_{f,\text{imag}} = \text{Im}(\mathbf{h}_f). \tag{23}$$

where Re$(\cdot)$ and Im$(\cdot)$ represent the real and imaginary part extraction operators, respectively.

The real and imaginary parts are concatenated and processed through a filtering layer:

$$\mathbf{h}_{f1} = \text{Dropout}(\text{ReLU}(\text{LayerNorm}(\mathbf{W}_{\text{freq}}[\mathbf{h}_{f,\text{real}} \| \mathbf{h}_{f,\text{imag}}]))). \tag{24}$$

Neighbor features and temporal features are first concatenated, then transformed to the frequency domain following the same procedure, resulting in $\mathbf{h}_{f2}$.

Then attention is computed in the frequency domain:

$$\mathbf{Q}_{\text{freq}} = \mathbf{h}_{f1}\mathbf{W}_Q, \tag{25}$$

$$\mathbf{K}_{\text{freq}} = \mathbf{h}_{f2}\mathbf{W}_K, \tag{26}$$

$$\mathbf{V}_{\text{freq}} = \mathbf{h}_{f2}\mathbf{W}_V. \tag{27}$$

The output of frequency attention is:

$$\mathbf{Z}_{\text{freq}} = \text{Attention}(\mathbf{Q}_{\text{freq}}, \mathbf{K}_{\text{freq}}, \mathbf{V}_{\text{freq}}). \tag{28}$$

The final output of the time-frequency attention module is:

$$\mathbf{Z}_{\text{final}} = \alpha \cdot \mathbf{Z}_{\text{time}} + (1-\alpha) \cdot \mathbf{Z}_{\text{freq}}. \tag{29}$$

where α is a hyperparameter that balances between temporal and frequency domain information.

Contrastive Learning

We adopt a contrastive learning strategy to learn node representations for anomaly detection. By maximizing the similarity of positive sample pairs and minimizing the similarity of negative sample pairs, we optimize the model to make normal nodes cluster in the representation space while anomalous nodes stay away from normal patterns.

The stability loss ensures consistency in node representations across adjacent time steps, preventing drastic changes in normal node representations:

$$\mathcal{L}_{\text{sta}}(u) = -\log \frac{\exp(\text{sim}(\mathbf{M}_u^{(t+1)}, \mathbf{M}_u^{(t)}))}{\sum_{i=1}^{N} \exp(\text{sim}(\mathbf{M}_u^{(t+1)}, \mathbf{M}_i^{(t)}))}, \tag{30}$$

where sim$(\cdot,\cdot)$ is the cosine similarity function.

The matching loss optimizes the final node representation through contrastive learning, ensuring that the node's final representation remains consistent with its memory state:

$$\mathcal{L}_{\text{mat}}(u) = -\log \frac{\exp(\text{sim}(\mathbf{Z}_{\text{final},u}, \mathbf{M}_u))}{\sum_{i=1}^{N} \exp(\text{sim}(\mathbf{Z}_{\text{final},u}, \mathbf{M}_i))}. \quad (31)$$

The overall loss function of the model is a weighted sum of the two loss terms:

$$\mathcal{L} = \mathcal{L}_{\text{sta}} + \lambda \mathcal{L}_{\text{mat}}, \quad (32)$$

where λ is a hyperparameter that balances the two loss terms, determined through validation set tuning.

Anomaly Scoring

After training completion, the anomaly score of a node is obtained by computing the degree of deviation of its final representation from normal patterns.

The stability score captures the stability degree of node memory during temporal evolution, identifying anomalous nodes whose behavior patterns undergo drastic changes:

$$S_{\text{sta}}(u_i, t_i) = \text{sim}(\mathbf{Z}_i^{(t)}, \mathbf{M}_i^{(t)}). \quad (33)$$

The matching score measures the consistency between node embedding and its memory representation, reflecting the degree of matching between the node's current behavior and historical behavior patterns:

$$S_{\text{mat}}(u_i, t_i) = \text{sim}(\mathbf{M}_i^{(t)}, \mathbf{M}_i^{(t-1)}). \quad (34)$$

where $\mathbf{M}_i^{(t-1)}$ represents the memory state of node u_i at the previous time step.

Combining stability score and matching score provides a more comprehensive anomaly assessment:

$$S_{\text{final}}(u_i, t_i) = \frac{2 - S_{\text{sta}}(u_i, t_i) - S_{\text{mat}}(u_i, t_i)}{4}. \quad (35)$$

Since $S_{\text{sta}}(u_i, t_i)$ and $S_{\text{mat}}(u_i, t_i)$ are in $(-1, 1)$, their sum ranges from $(-2, 2)$. We normalize to $(0, 1)$ by adding 2 and dividing by 4.

The Algorithm and Complexity Analysis

The overall time complexity of FreqTAD is $\mathcal{O}(|E|d + T^2 d + d^2 + N^2 d)$, where $|E|$ represents the number of edges, d represents the feature dimension, T represents the temporal feature length, and N represents the number of nodes. This complexity consists of four main components: dynamic graph processing ($\mathcal{O}(|E|)$), multi-scale frequency encoding ($\mathcal{O}(|E|d)$), temporal-frequency attention ($\mathcal{O}(T^2 d + d^2)$), and contrastive learning ($\mathcal{O}(N^2 d)$). Compared to existing methods, FreqTAD maintains reasonable computational complexity through parallel processing of time and frequency dimensions.

Experiment

In this section, we first introduce the experimental settings, and then analyze the performance of FreqTAD by answering the following Research Questions.

- **RQ1:** How does FreqTAD perform compared to other state-of-the-art methods?
- **RQ2:** Is FreqTAD efficient in terms of training time and computational resources?
- **RQ3:** What impact do the key components of FreqTAD have on anomaly detection performance?
- **RQ4:** What influence do hyperparameters have on FreqTAD's performance?
- **RQ5:** How does multi-scale frequency domain time encoding compare with traditional Bochner encoding?

Experiment Settings

Datasets: We conduct experiments on seven dynamic graph datasets: four with anomaly labels (Wikipedia, Reddit, Bitcoinalpha, Bitcoinotc) for detection tasks, and three additional datasets (Enron, UCI, Social Evolution) to validate our multi-scale frequency encoding. These datasets span diverse scenarios including page editing, social interactions, financial transactions, email communications, and forum activities.

Baselines: We compare against ten anomaly detection baseline methods under the inductive setting, including **four statistical-based methods** (SedanSpot (Eswaran and Faloutsos 2018), MIDAS-R (Bhatia et al. 2020), F-FADE (Chang et al. 2021), AnoEdge-1 (Bhatia et al. 2023)) and **six neural network-based methods** (JODIE (Kumar, Zhang, and Leskovec 2019), DyRep (Trivedi et al. 2019), TGAT (Xu et al. 2020), TGN (Rossi et al. 2020), SAD (Tian et al. 2023), SLADE (Lee, Kim, and Shin 2024)). Statistical-based methods are evaluated directly on the test set without training, while neural network methods are trained on the training set and hyperparameters are tuned through the validation set.

Evaluation Metrics: We use Area Under the ROC Curve (AUC) and Average Precision (AP) as evaluation metrics.

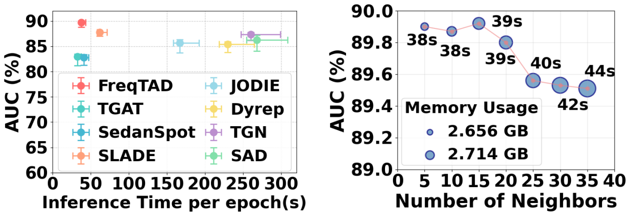
Implementation Details: We implement FreqTAD using PyTorch and accelerate model training with an Intel Core™ i5-12400F CPU (2.50GHz) and NVIDIA® GeForce RTX 3060 GPU (16GB). We use PyCharm as the coding platform. The loss function is optimized using the Adam optimizer with an exponential decay learning rate scheduler. For training settings, we set the batch size to 100, learning rate to 3e-6, training epochs to 30, neighbor sampling number to $K = 20$, and attention heads to 2.

Anomaly Detection Performance (RQ1)

Table 1 presents a comparison of anomaly detection performance between FreqTAD and baseline methods across four datasets. From Table 1, the observations are as follows: 1) FreqTAD achieves the best or near-best AUC performance on multiple datasets, showing significant advantages compared to other methods. 2) FreqTAD demonstrates good stability and robustness across all tested datasets. 3) Deep learning-based methods generally outperform traditional statistical methods, which can be attributed to the stronger nonlinear modeling capabilities of deep neural networks. These results indicate that FreqTAD has superior performance in anomaly detection tasks, benefiting from multi-

Method	Wikipedia	Reddit	Bitcoinalpha	Bitcoinotc
SedanSpot (ICDM'18)	82.79 ± 1.24	58.89 ± 1.67	69.01 ± 0.85	71.80 ± 0.92
MIDAS (AAAI'20)	62.83 ± 2.15	60.02 ± 1.38	64.52 ± 0.76	62.24 ± 1.45
F-FADE (WSDM'21)	44.80 ± 0.12	49.84 ± 0.18	53.67 ± 0.09	51.20 ± 0.15
AnoEdge (KDD'23)	47.44 ± 0.58	48.39 ± 0.63	62.61 ± 0.42	65.89 ± 0.37
JODIE (KDD'19)	85.68 ± 0.45	61.52 ± 0.89	*73.48 ± 1.12*	69.20 ± 0.78
DyRep (ICLR'19)	85.60 ± 0.52	63.49 ± 0.94	73.38 ± 1.05	70.85 ± 0.83
TGAT (ICLR'20)	83.32 ± 0.87	65.17 ± 1.23	71.59 ± 0.96	68.42 ± 1.08
TGN (ICML'20)	87.37 ± 0.34	67.22 ± 0.71	69.97 ± 0.88	75.78 ± 0.41
SAD (IJCAI'23)	*86.25 ± 0.67*	*68.35 ± 1.15*	68.50 ± 2.34	64.59 ± 3.12
SLADE (KDD'24)	87.81 ± 0.49	**72.14 ± 0.56**	73.59 ± 0.38	*72.46 ± 0.44*
FreqTAD (Ours)	**89.75 ± 0.31**	72.08 ± 0.42	**78.32 ± 0.25**	**77.39 ± 0.38**

Table 1: Performance comparison on anomaly detection (AUC %). The **best** results are highlighted in **bold**, the second-best results are underlined, and the *third-best* results are shown in *italics*.



(a) (b)

Figure 3: (a) Performance-efficiency comparison of different methods on Wikipedia dataset. (b) Impact of neighbor count on AUC performance, circle size represents memory usage, annotations show runtime.

scale frequency encoding that can effectively distinguish periodic and bursty features in time series. Additionally, the time-frequency attention mechanism can simultaneously focus on importance allocation in both temporal and frequency dimensions.

Training Efficiency (RQ2)

Figure 3 (a) illustrates the performance-efficiency trade-off of different methods on the Wikipedia dataset. FreqTAD achieves optimal anomaly detection performance with inference time of only approximately 40 seconds, significantly outperforming other deep learning methods that require 200-300 seconds. It is positioned in the upper-left region of the figure, achieving the best performance-efficiency balance.

Meanwhile, Figure 3 (b) shows that the impact of neighbor count K on model performance exhibits a trend of first increasing then decreasing on the Wikipedia dataset. When the number of neighbors is 15, optimal AUC performance is achieved, and performance declines beyond this point while computation time and memory usage increase. This indicates the existence of an optimal neighbor count balance point, where excessive neighbors increase computational costs and introduce noise leading to performance degradation.

Ablation Study (RQ3)

We conduct ablation studies on the FreqTAD model to evaluate the contribution of each key component. Figure 4 presents the performance of different variants: w/o-time (removing the multi-scale frequency encoding module), w/o-atti (removing the time-frequency attention mechanism), w/o-match (removing the matching loss), and w/o-stable (removing the stability loss).

The experimental results in Figure 4 are summarized as follows: 1) FreqTAD outperforms the w/o-time and w/o-stable variants, highlighting the capability of the time encoding module in capturing temporal dependencies and the important role of the stability component in enhancing anomaly detection performance. 2) The superior performance compared to w/o-atti emphasizes the effectiveness of the attention mechanism in identifying key anomalous patterns in sequences. 3) The better results than w/o-match demonstrate the effectiveness of the matching mechanism in integrating multi-dimensional feature information, which helps distinguish between normal and abnormal behavioral patterns. These ablation study results verify the necessity and effectiveness of each component in FreqTAD.

Hyper-parameter Analysis (RQ4)

To analyze the performance characteristics of FreqTAD, we conducted sensitivity analysis on the key hyperparameter α, which controls the weight allocation between temporal and frequency domain attention in our time-frequency attention mechanism.

As shown in Figure 5, different datasets exhibit distinct preferences for attention mechanisms. Wikipedia demonstrates stable performance across all α values with optimal results at $\alpha = 0.6$, showing excellent robustness. Bitcoinotc achieves peak performance at $\alpha = 0.4$, indicat-

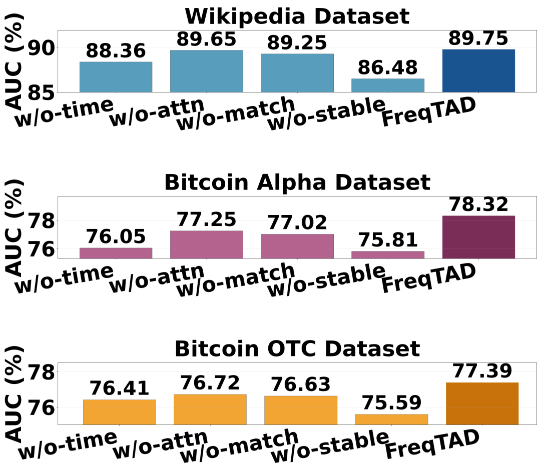


Figure 4: Ablation study results and analysis.

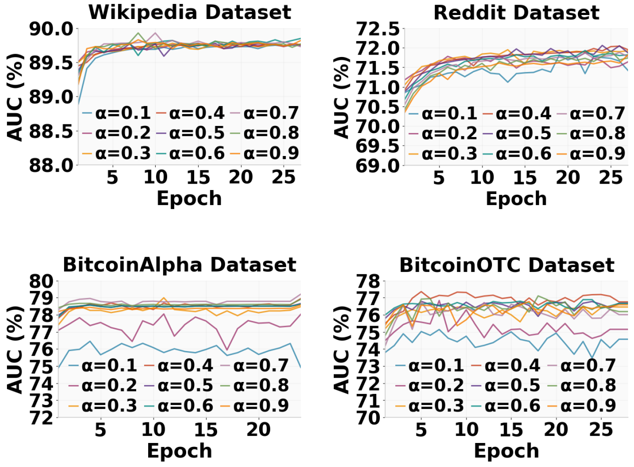


Figure 5: Performance comparison of hyperparameter α across different datasets.

ing that frequency domain attention plays a crucial role in detecting anomalies in this dataset. In contrast, BitcoinAlpha prefers temporal attention with optimal performance at $\alpha = 0.7$, suggesting that anomalous behaviors are better captured through temporal continuity changes rather than frequency domain variations.

The analysis reveals that balanced configurations $\alpha \in [0.4, 0.7]$ consistently perform well across all datasets, validating the effectiveness of time-frequency attention fusion. Over-dependence on a single attention mechanism leads to performance degradation, demonstrating the necessity of multi-dimensional attention integration in our approach.

Multi-scale Frequency Encoding Performance (RQ5)

Since TGAT is the first model to propose time encoding using Bochner's theorem, we conduct experiments by unifying our proposed multi-scale frequency encoding method with the Bochner method under the TGAT framework, with results shown in Table 2.

Dataset	Metric	Bochner	M-s Freq.	Imp.
Wikipedia	AUC	93.20	**94.14**	0.94
	AP	93.80	**94.40**	0.60
Reddit	AUC	96.07	**97.08**	1.01
	AP	96.30	**97.35**	1.05
Bitcoinalpha	AUC	73.10	**85.40**	12.30
	AP	71.82	**85.02**	13.20
Bitcoinotc	AUC	78.37	**88.17**	9.80
	AP	79.19	**89.21**	10.02
UCI	AUC	71.82	**73.92**	2.10
	AP	68.80	**73.54**	4.74
Enron	AUC	56.15	**70.66**	14.51
	AP	60.70	**73.63**	12.93
Social Evolve	AUC	67.57	**74.60**	7.03
	AP	65.37	**72.77**	7.40

Table 2: Performance comparison of time encoding methods (M-s Freq. = Multi-scale Frequency, Imp. = Improvement).

The experimental results demonstrate that the multi-scale frequency encoding method achieves significant performance improvements across all datasets. The method exhibits consistent superiority across different types of datasets, with particularly outstanding performance on financial transaction datasets and communication network datasets. The results indicate that multi-scale frequency-domain encoding has stronger modeling capabilities when processing data with complex multi-scale temporal features. In contrast, the improvements on social network datasets are relatively modest, which may be attributed to the relatively regular temporal patterns in such data, where traditional methods can already capture the basic features adequately. Overall, these results fully validate the effectiveness and generalizability of multi-scale frequency encoding compared to traditional Bochner encoding in capturing complex temporal dependencies in temporal graphs.

Conclusion

In this paper, we propose FreqTAD, an anomaly detection in dynamic graphs method that integrates multi-scale frequency encoding with time-frequency attention mechanisms. Our method addresses two key limitations in existing approaches: the inability to distinguish between periodic and bursty features in time encoding, and the lack of frequency-domain awareness in attention mechanisms. To tackle the above challenges, we design a multi-scale frequency encoder that decomposes time series into periodic and bursty components, and propose a time-frequency attention mechanism that considers both temporal and frequency dimensions. Experimental results on four dynamic graph datasets demonstrate the effectiveness and efficiency of FreqTAD, laying the foundation for its practical application in real-world anomaly detection systems.

Acknowledgments

This research is supported by the National Key R&D Program of China (Grant No. 2022ZD0119501), the National Natural Science Foundation of China (Grant No. 62472263, 52374221, 52574256, 62502288), the Taishan Scholar Program of Shandong Province (tstp20250506, tsqn202211154), Shandong Youth Innovation Team, the Natural Science Foundation of Shandong Province (Grant No. ZR2024MF034, ZR2025QC624), the Ministry of Education in China Foundation for Humanities and Social Sciences (Grant No. 24YJAZH058, 24YJCZH461), the Guangxi Key Laboratory of Trusted Software (KX202305).

References

Bhatia, S.; Hooi, B.; Yoon, M.; Shin, K.; and Faloutsos, C. 2020. Midas: Microcluster-Based Detector of Anomalies in Edge Streams. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI 2020)*, 3242–3249.

Bhatia, S.; Wadhwa, M.; Kawaguchi, K.; Shah, N.; Yu, P. S.; and Hooi, B. 2023. Sketch-Based Anomaly Detection in Streaming Graphs. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD 2023)*, 93–104.

Bo, D.; Wang, X.; Shi, C.; and Shen, H. 2021. Beyond Low-frequency Information in Graph Convolutional Networks. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence*, 3950–3957.

Bruna, J.; Zaremba, W.; Szlam, A.; and LeCun, Y. 2014. Spectral Networks and Locally Connected Networks on Graphs. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR 2014)*.

Chang, Y.-Y.; Li, P.; Sosic, R.; Afifi, M. H.; Schweighauser, M.; and Leskovec, J. 2021. F-FADE: Frequency Factorization for Anomaly Detection in Edge Streams. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining (WSDM 2021)*, 589–597.

Chen, J.; Fu, S.; Ma, Z.; Feng, M.; Wirjanto, T. S.; and Peng, Q. 2025. Semi-supervised Anomaly Detection with Extremely Limited Labels in Dynamic Graphs. *arXiv preprint arXiv:2501.15035*.

Chen, X.; Xiong, Y.; Zhang, S.; Zhang, J.; Zhang, Y.; Zhou, S.; Wu, X.; Zhang, M.; Liu, T.; and Wang, W. 2024. DT-Former: A Transformer-Based Method for Discrete-Time Dynamic Graph Representation Learning. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM 2024)*, 301–311.

Cong, W.; Zhang, S.; Kang, J.; Yuan, B.; Wu, H.; Zhou, X.; Tong, H.; and Mahdavi, M. 2023. Do We Really Need Complicated Model Architectures For Temporal Networks? In *Proceedings of the Eleventh International Conference on Learning Representations (ICLR 2023)*.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *Proceedings of the 30th International Conference on Neural Information Processing Systems (NIPS 2016)*, 3844–3852.

Eswaran, D.; and Faloutsos, C. 2018. SedanSpot: Detecting Anomalies in Edge Streams. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17-20, 2018*, 953–958.

Fang, L.; Feng, K.; Gui, J.; Feng, S.; and Hu, A. 2023. Anonymous Edge Representation for Inductive Anomaly Detection in Dynamic Bipartite Graphs. *Proceedings of the VLDB Endowment*, 16(5): 1154–1167.

Fu, Y.; Zhou, C.; Li, J.; and Chen, L. 2025. Long-term Evolutionary Patterns Matter: Self-supervised Anomaly Detection on Dynamic Graphs. *Knowledge-Based Systems*, 311: 113049.

Kumar, S.; Zhang, X.; and Leskovec, J. 2019. Predicting Dynamic Embedding Trajectory in Temporal Interaction Networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1269–1278.

Lee, J.; Kim, S.; and Shin, K. 2024. SLADE: Detecting Dynamic Anomalies in Edge Streams without Labels via Self-Supervised Learning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD 2024)*, 1506–1517.

Li, J.; Wu, L.; Du, Y.; Hong, R.; and Li, W. 2024. Dual Graph Neural Networks for Dynamic Users' Behavior Prediction on Social Networking Services. *IEEE Transactions on Computational Social Systems*, 11(5): 7020–7031.

Liu, Y.; and Du, B. 2025. Frequency Domain-Oriented Complex Graph Neural Networks for Graph Classification. *IEEE Transactions on Neural Networks and Learning Systems*, 36(2): 2733–2746.

Rossi, E.; Chamberlain, B.; Frasca, F.; Eynard, D.; Monti, F.; and Bronstein, M. 2020. Temporal Graph Networks for Deep Learning on Dynamic Graphs. In *ICML 2020 Workshop on Graph Representation Learning*.

Su, J.; Zou, D.; and Wu, C. 2024. PRES: Toward Scalable Memory-Based Dynamic Graph Neural Networks. In *The Twelfth International Conference on Learning Representations (ICLR 2024)*.

Tian, S.; Dong, J.; Li, J.; Zhao, W.; Xu, X.; Wang, B.; Song, B.; Meng, C.; Zhang, T.; and Chen, L. 2023. SAD: Semi-Supervised Anomaly Detection on Dynamic Graphs. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence (IJCAI 2023)*, 2306–2314.

Tian, Y.; Qi, Y.; and Guo, F. 2024. FreeDyG: Frequency Enhanced Continuous-Time Dynamic Graph Model for Link Prediction. In *The Twelfth International Conference on Learning Representations (ICLR 2024)*.

Trivedi, R.; Farajtabar, M.; Biswal, P.; and Zha, H. 2019. DyRep: Learning Representations over Dynamic Graphs. In *7th International Conference on Learning Representations*.

Wang, X.; Wang, C.; Liu, H.; Wang, M.; Qin, T.; and Wang, P. 2025. FiGraph: A Dynamic Heterogeneous Graph Dataset for Financial Anomaly Detection. In *Companion Proceedings of the ACM Web Conference 2025 (WWW '25)*, 813–816.

Xu, D.; Ruan, C.; Körpeoglu, E.; Kumar, S.; and Achan, K. 2020. Inductive Representation Learning on Temporal Graphs. In *Proceedings of the Eighth International Conference on Learning Representations (ICLR 2020)*.

Yang, X.; Zhao, X.; and Shen, Z. 2025. A Generalizable Anomaly Detection Method in Dynamic Graphs. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence (AAAI 2025)*, 22001–22009.

Zheng, J.; Yang, C.; Zhang, T.; Cao, L.; Jiang, B.; Fan, X.; Wu, X.-M.; and Zhu, X. 2025. Dynamic Spectral Graph Anomaly Detection. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence (AAAI 2025)*, 13410–13418.