MaskDGNN: Self-Supervised Dynamic Graph Neural Networks with Activeness-aware Temporal Masking

Yiming He[1] , **Xiang Li**[1] , **Zhongying Zhao**[2] , **Haobing Liu**[1] , **Peilan He**[1] and **Yanwei Yu**[1*]
[1]Faculty of Information Science and Engineering, Ocean University of China
[2]Faculty of Computer Science and Engineering, Shandong University of Science and Technology
{heyiming7773,lixiang1202}@stu.ouc.edu.cn, zzysuin@163.com,
{haobingliu,hepeilan,yuyanwei}@ouc.edu.cn

Abstract

Integrating dynamics into graph neural networks (GNNs) provides deeper insights into the evolution of dynamic graphs, thereby enhancing the temporal representation in real-world dynamic network problems. Existing methods extracting critical information from dynamic graphs face two key challenges, either overlooking the negative impact of redundant information or struggling in addressing the distribution shifting issue in dynamic graphs. To address these challenges, we propose MaskDGNN, a novel dynamic GNN architecture that consists of two modules: First, self-supervised activeness-aware temporal masking mechanism selectively retains edges between highly active nodes while masking those with low activeness, effectively reducing redundancy. Second, adaptive frequency enhancing graph representation learner amplifies the frequency-domain features of nodes to capture intrinsic features under distribution shifting. Experiments on five real-world dynamic graph datasets demonstrate that MaskDGNN outperforms state-of-the-art methods, achieving an average improvement of 7.07% in accuracy and 13.87% in MRR for link prediction tasks.

1 Introduction

Dynamic graphs are prevalent in real-world systems, capturing the evolving relationships and interactions within complex networks [Kazemi *et al.*, 2020]. For instance, in social networks [Song *et al.*, 2019; Ma *et al.*, 2020], they model dynamic user interactions such as friendship formation and information diffusion. In transportation networks [Kumar *et al.*, 2019; Wu *et al.*, 2019b], they reflect real-time traffic patterns, with nodes and edges representing locations and their changing connectivity. Similarly, in e-commerce networks [Sharma *et al.*, 2023; Zhang *et al.*, 2022], they track temporal shifts in user behaviors, such as browsing, purchasing, and reviewing products. These networks are characterized by temporally evolving nodes and edges, exhibiting rich temporal features that are essential for understanding network dynamics. Analyzing these properties provides valuable insights into the behavior and evolution of complex systems, enabling more accurate predictions and informed decision-making.

Compared to static graph methods [You *et al.*, 2021; Hassani and Khasahmadi, 2020; Yu *et al.*, 2022; Fu *et al.*, 2023], dynamic graph approaches are more adept at capturing the temporal evolution of nodes and edges, offering enhanced expressive power for modeling complex real-world interactions. Existing dynamic graph neural networks (DGNNs) [Xue *et al.*, 2022] can be broadly categorized into continuous and discrete models. On the one hand, continuous DGNNs [Tian *et al.*, 2023; Rossi *et al.*, 2020; Cong *et al.*, 2023; Poursafaei *et al.*, 2022] capture fine-grained temporal information by transforming dynamic graph data into time series and employing temporal encoders, such as Transformers, to model dynamic patterns [Yu *et al.*, 2023; Xu *et al.*, 2020]. These approaches excel in capturing detailed temporal dynamics by leveraging powerful temporal encoders. Discrete DGNNs [Liu *et al.*, 2021], on the other hand, employ more lightweight designs, focusing on capturing dynamic dependencies without relying on continuous time series. Early models such as EvolveGCN [Pareja *et al.*, 2020], DGNN [Manessi *et al.*, 2020], and DySAT [Sankar *et al.*, 2020] utilize recurrent networks or temporal encoders to capture the temporal evolution of graphs. More recent models, including Roland [You *et al.*, 2022], WinGNN [Zhu *et al.*, 2023] and DyTed [Zhang *et al.*, 2023], replace these traditional temporal encoders with techniques such as meta-learning [Finn *et al.*, 2017], sliding window strategies, and disentangling methods. These approaches offer significant improvements in performance, enabling real-time processing while maintaining high predictive accuracy.

Despite significant advancements in dynamic graph learning, two key challenges persist: *(1) Existing methods tend to overlook the negative impact of redundant information on dynamic graphs.* Many methods rely on extensive historical information, overemphasizing low-activeness nodes and infrequently changing edges which may lead to redundant information. This reduces the model's ability to focus on key interactions and harms its predictive performance. Emphasizing the dynamic changes of highly active and critical nodes is crucial for improving learning effectiveness. *(2) Most existing methods struggle in addressing the distribution shifting*

[*]Yanwei Yu is the corresponding author.

issue on dynamic graphs. The temporal structure distribution in dynamic graphs evolves over time, caused by changes in node relationships. For instance, in a Bitcoin trading network, traders typically follow certain patterns based on market trends, but a sudden shift in focus, such as a major news event or a market crash, can dramatically alter trading behaviors. Existing models struggle to adapt to these shifts, hindering their ability to capture evolving interactions and compromising predictive accuracy.

To address these challenges, we propose MaskDGNN, a novel dynamic graph neural network that reduces the negative impact of redundant information and captures intrinsic features under distribution shifting. MaskDGNN consists of two main components: (1) Self-supervised activeness-aware temporal masking module: It identifies and retains edges between highly active nodes while masking edges between nodes with low activeness. By comparing successive graph snapshots, this strategy filters out redundant information, ensuring the model focuses on the most relevant interactions, thereby improving predictive accuracy. (2) Adaptive frequency enhancing graph representation learner: This module addresses the feature capture issue under the temporal structure distribution shifting challenge in dynamic graphs by leveraging Fourier transforms to enhance the graph's temporal features. It identifies and emphasizes the key frequency components in active node representations, allowing the model to better adapt to distribution shifts, such as sudden changes in trading patterns or behaviors. Additionally, a sliding window strategy aggregates long-term dependencies, improving robustness and capturing macro-level structural changes across graph snapshots.

Our contributions are summarized as follows:

- We propose MaskDGNN, a novel dynamic graph neural network architecture that integrates a self-supervised activeness-aware temporal masking module with an adaptive frequency enhancing graph representation learner. This design reduces redundant information while capturing intrinsic features under distribution shifting, significantly improving link prediction accuracy.
- We introduce an activeness-aware masking mechanism that compares each node's activeness between consecutive time steps, retaining edges of highly active nodes and masking low-activity ones, effectively filtering out redundancy and focusing on key temporal interactions.
- Extensive experiments on five real-world dynamic graph datasets show that MaskDGNN outperforms state-of-the-art methods, with average improvements of 7.07%, 2.62%, 13.87%, and 7.31% in Accuracy, AUC, MRR, and Recall@10, respectively.

2 Related Work

2.1 Dynamic Graph Neural Networks

Research on DGNNs [Yang *et al.*, 2024; Kazemi *et al.*, 2020] mainly focuses on capturing the dynamic changes in graph structures and their temporal dependencies to enhance the ability of dynamic graph representation learning. Existing DGNNs can be classified into two types based on how time is recorded: discrete DGNNs and continuous DGNNs [Wang *et al.*, 2021; Jin *et al.*, 2022; Kachole *et al.*, 2023; Trivedi *et al.*, 2019]. This paper primarily discusses the related work in discrete DGNNs. Early works such as EvolveGCN [Pareja *et al.*, 2020] and DGNN [Manessi *et al.*, 2020] leverage the combination of recurrent networks (GRU [Cho *et al.*, 2014] or LSTM [Hochreiter and Schmidhuber, 1997]) with GCN parameters and the integration of long short-term memory networks with graph convolutional networks. DGT [Cong *et al.*, 2021] and DySAT [Sankar *et al.*, 2020] are based on Transformers [Vaswani, 2017] and are designed to capture dynamic graph information. MTSN [Liu *et al.*, 2021] models the local higher-order structures and temporal evolution of dynamic networks by learning the motif structure of the graph. Dyngraph2vec [Goyal *et al.*, 2020] adopts a deep architecture combining dense and recurrent layers to learn temporal dynamics in dynamic graphs. Recent studies further optimized dynamic graph modeling frameworks. For instance, DyTed [Zhang *et al.*, 2023] proposes a disentangled representation learning approach that leverages contrastive learning to distinguish between temporal and invariant features. DRLAN [Liu *et al.*, 2020] introduces a hybrid offline-online architecture for updating node embeddings in large-scale attributed dynamic networks. ROLAND [You *et al.*, 2022] is a general framework from the perspective of meta-learning, enabling the transformation of static GNNs into dynamic GNNs. WinGNN [Zhu *et al.*, 2023] utilizes a sliding window mechanism to reduce parameter size, alleviating the computational and storage burdens of long sequences. SILD [Zhang *et al.*, 2024] employs Fourier transforms to achieve different frequency separation. STDGL [Liu *et al.*, 2024b] designs time-aware auxiliary tasks that incorporate local and global connectivity information.

2.2 Self-Supervised Masked Graph Models

With the growing use of graph neural networks (GNNs) in areas like social networks, recommendation systems, and bioinformatics, graph masking strategies become essential for highlighting key structures during training. These strategies mask certain nodes or edges to guide the model in focusing on critical graph information and reconstructing the masked parts, improving its understanding of graph structures. Initially introduced in computer vision by MAE [He *et al.*, 2022], the concept is extended in GraphMAE [Hou *et al.*, 2022], which masks node features to generate compact graph representations. S2GAE [Tan *et al.*, 2023] integrates masking with graph autoencoders, setting the stage for future developments. MaskGAE [Li *et al.*, 2023] enhances graph structure capture by using random walks to determine masking positions, while Bandana [Zhao *et al.*, 2024] introduces edge masking with a continuous range $[0, 1]$, balancing structural preservation and flexibility. Finally, StructMAE [Liu *et al.*, 2024a] employs progressive masking based on structural scoring to improve the model's ability to reconstruct key graph structures.

3 Preliminaries

Definition 1 (Discrete-Time Dynamic Graphs)**.** *Given a dynamic graph dataset, we define a discrete-time dynamic*

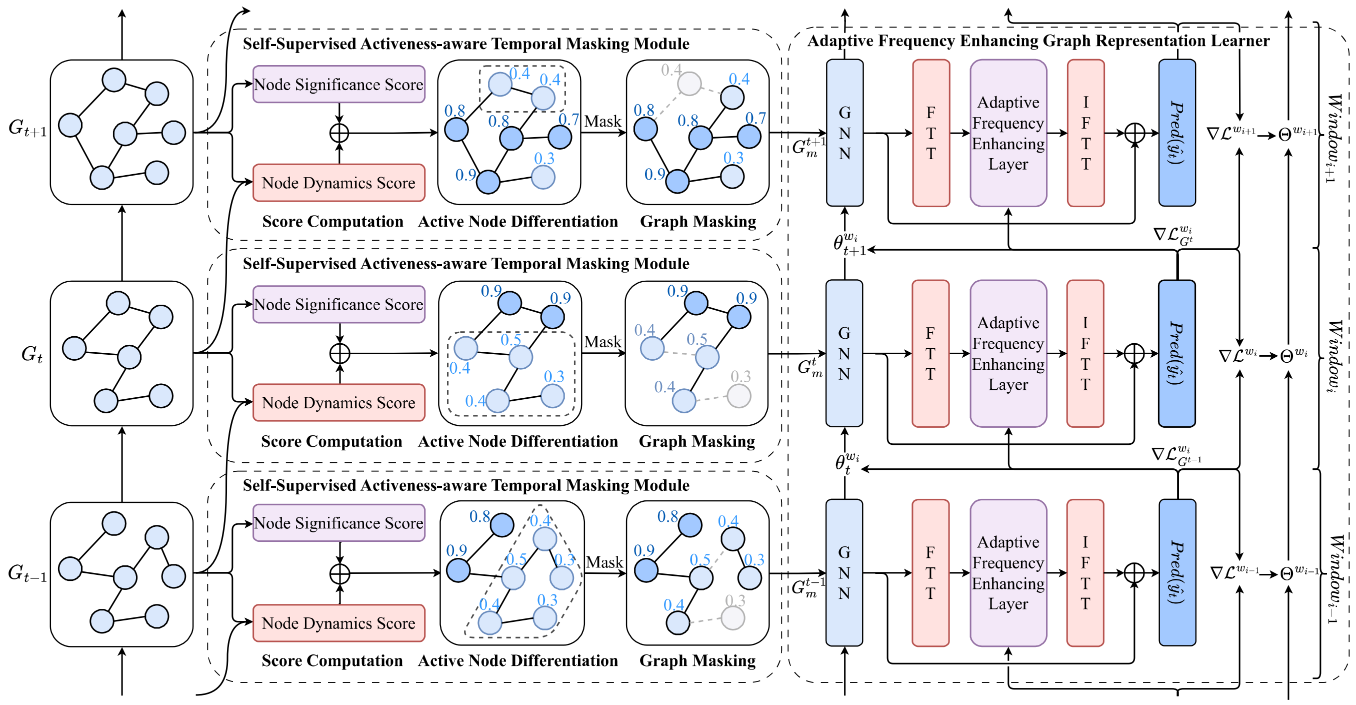

Self-Supervised Activeness-aware Temporal Masking Module
Node Significance Score
Node Dynamics Score
Score Computation
Active Node Differentiation
Graph Masking
Mask
Adaptive Frequency Enhancing Graph Representation Learner
GNN
FTT
Adaptive Frequency Enhancing Layer
IFTT
$Pred(\hat{y}_t)$
G_{t+1}
G_t
G_{t-1}
G_m^{t+1}
G_m^t
G_m^{t-1}
$\theta_{t+1}^{w_i}$
$\theta_t^{w_i}$
$\nabla\mathcal{L}^{w_{i+1}}$
$\nabla\mathcal{L}^{w_i}$
$\nabla\mathcal{L}^{w_{i-1}}$
$\nabla\mathcal{L}_{G^t}^{w_i}$
$\nabla\mathcal{L}_{G^{t-1}}^{w_i}$
$\Theta^{w_{i+1}}$
Θ^{w_i}
$\Theta^{w_{i-1}}$
$Window_{i+1}$
$Window_i$
$Window_{i-1}$

Figure 1: The overall architecture of the proposed MaskDGNN.

graph as a sequence of snapshots $\mathcal{G} = \{G^1, G^2, \ldots, G^T\}$, where T denotes the total number of snapshots. Each snapshot at time t, i.e., $G^t = (\mathcal{V}^t, \mathcal{E}^t)$, consists of a node set $\mathcal{V}^t$ and an edge set $\mathcal{E}^t \subseteq \mathcal{V}^t \times \mathcal{V}^t$. The adjacency matrix $\mathbf{A}^t \in \mathbb{R}^{|\mathcal{V}^t|\times|\mathcal{V}^t|}$ represents the edges in $\mathcal{E}^t$, and $\mathbf{X}^t \in \mathbb{R}^{|\mathcal{V}^t|\times f}$ denotes the node feature matrix at time t, where f is the dimensionality of the node features.

Problem. *The task of link prediction involves learning a function f that predicts the presence of an unobserved edge at time $t+1$, based on the node features and graph structure from the previous snapshots:*

$$f : \left(\{G^1, G^2, \ldots, G^t\}, \mathbf{X}^t\right) \longrightarrow \mathcal{E}^{t+1}. \tag{1}$$

4 Methodology

In this section, we provide a detailed description of our proposed MaskDGNN. The overall architecture of MaskDGNN is shown in Fig. 1, which consists of two key components: (1) *Self-Supervised Activeness-aware Temporal Masking Module*, and (2) *Adaptive Frequency Enhancing Graph Representation Learner*.

4.1 Self-Supervised Activeness-aware Temporal Masking Module

We propose a temporal masking mechanism that dynamically evaluates node activeness at each time step and selectively masks less active edges. This approach enhances dynamic graph representation learning by preserving critical structural updates while filtering out redundant information.

Node Activeness Score Calculation

To better focus on key and active nodes while minimizing interference from redundant nodes, we introduce a node activeness score $\mathcal{A}_{v_i}^t$ to evaluate each node. This score determines edge retention or masking: edges between high-activeness nodes are retained, while those between low-activeness nodes are masked with a certain probability. Specifically, $\mathcal{A}_{v_i}^t$ is defined as the sum of two components: the node dynamics score $\mathcal{D}_{v_i}^t$ and the node significance score $\mathcal{S}_{v_i}^t$.

The node dynamics score $\mathcal{D}_{v_i}^t$ quantifies the extent of changes in a node's edges over time. It is computed by comparing the node's edge changes between the current graph G^t and the previous time step graph G^{t-1}, along with the ratio of the node degree at the previous time step. The specific definition is as follows:

$$\mathcal{D}_{v_i}^t = \begin{cases} \lambda \cdot \dfrac{\Delta|\mathcal{N}_{v_i}^t|}{\sqrt{deg(v_i)^{t-1} + \eta}}, & \text{if } v_i \in G^{t-1}, \\ \lambda \cdot \dfrac{\Delta|\mathcal{N}_{v_i}^t|}{\sqrt{\Delta|\mathcal{N}_{v_i}^t| + \gamma}}, & \text{if } v_i \notin G^{t-1}, \end{cases} \tag{2}$$

where $v_i \in G^t$, and the calculation of the node dynamics score $\mathcal{D}_{v_i}^t$ depends on whether v_i exists in G^t or not. $deg(v_i)^{t-1}$ denotes the degree of node v_i in G^{t-1}, η and γ are introduced to control the weight of the dynamics score for existing and newly added nodes, λ is a parameter that adjusts the influence of $\Delta|\mathcal{N}_{v_i}^t|$ on the dynamics score, and $\Delta|\mathcal{N}_{v_i}^t|$ is calculated as:

$$\begin{aligned} \Delta|\mathcal{N}_{v_i}^t| &= |\mathcal{N}_{v_i}^t \oplus \mathcal{N}_{v_i}^{t-1}| \\ &= |(\mathcal{N}_{v_i}^t - \mathcal{N}_{v_i}^{t-1}) \cup (\mathcal{N}_{v_i}^{t-1} - \mathcal{N}_{v_i}^t)|, \end{aligned} \tag{3}$$

where $\oplus$ denotes the symmetric difference operator, $\mathcal{N}_{v_i}^{t-1}$ and $\mathcal{N}_{v_i}^{t}$ are the edge sets of node v_i at time step $t-1$ and t, respectively.

The node significance score $\mathcal{S}_{v_i}^t$ measures the influence of node v_i in the current graph G^t, which is computed using the PageRank algorithm $Rank(\cdot)$ to evaluate the global importance of each node within the graph:

$$\mathcal{S}_{v_i}^t = Rank(v_i) = \frac{1-d}{N} + d \sum_{v_j \in \mathcal{N}_{v_i}^t} \frac{Rank(v_j)}{|\mathcal{N}_{v_j}^t|}, \quad (4)$$

where d is the damping factor (typically set to 0.85), and N is the total number of nodes. The first term models random jumps to any node in the graph, while the second term aggregates contributions from the node's neighbors.

Finally, the node dynamics score $\mathcal{D}_{v_i}^t$ and the node significance score $\mathcal{S}_{v_i}^t$ are weighted and fused by control parameter α, resulting in a comprehensive node activeness score $\mathcal{A}_{v_i}^t$. This score integrates both node dynamics and global importance, providing a more holistic representation of the node's significance in the dynamic graph as follows:

$$\mathcal{A}_{v_i}^t = \alpha \mathcal{D}_{v_i}^t + (1-\alpha)\mathcal{S}_{v_i}^t. \quad (5)$$

Self-Supervised Temporal Masking

Self-supervised graph masking has been widely explored for static graphs, where masked structures help capture essential features and patterns. Building on this, we propose a self-supervised temporal masking module for dynamic graphs, using node activeness scores to guide the masking of redundant edges. Specifically, the mask probability $p_{e_{ij}}^t$ for the edge between nodes v_i and v_j at time step t is defined as follows:

$$p_{e_{ij}}^t = 1 - Mean(\mathcal{A}_{v_i}^t, \mathcal{A}_{v_j}^t), \quad (6)$$

where $\mathcal{A}_{v_i}^t$ and $\mathcal{A}_{v_j}^t$ represent the activeness scores of nodes v_i and v_j, and $Mean(\cdot)$ represents the mean pooling operation. Edges with low $p_{e_{ij}}^t$ can be considered redundant and are more likely to be masked.

To ensure that the total number of masked edges aligns with the desired masking ratio, the adjusted mask probability $\hat{p}_{e_{ij}}^t$ for each edge is derived by scaling the original mask probability:

$$\hat{p}_{e_{ij}}^t = \min\left(p_{e_{ij}}^t \times \frac{\lfloor N_{\text{edges}} \times p \rfloor}{\sum_{i=1}^{N_{\text{edges}}} p_{e_{ij}}^t}, 1\right), \quad (7)$$

where N_{edges} is the total number of edges, and p is the overall proportion of masked edges in the graph G^t. The final decision to mask each edge at time t is determined by the adjusted mask probability $\hat{p}_{e_{ij}}^t$:

$$\text{Mask}(e_{ij}^t) = 1 - U_{ij}, \quad \text{where } U_{ij} \sim \mathbb{B}(\hat{p}_{e_{ij}}^t), \quad (8)$$

where U_{ij} is a random variable following a Bernoulli distribution $\mathbb{B}(\hat{p}_{e_{ij}}^t)$. If $U_{ij} = 1$, the edge e_{ij}^t is not masked ($\text{Mask}(e_{ij}^t) = 0$); if $U_{ij} = 0$, the edge e_{ij}^t is masked ($\text{Mask}(e_{ij}^t) = 1$).

After masking operation on each snapshot, we obtain the masked graph set $\mathcal{G}_m = \{G_m^1, G_m^2, \ldots, G_m^T\}$ and the retained graphs set $\mathcal{G}_r = \{G_r^1, G_r^2 \ldots, G_r^T\}$, noted that $G_m^t \cup G_r^t = G^t$.

Unlike previous models directly discard masked edges, The self-supervised loss for the masked edges treats them as positive samples, preserving the link information in the original graph and ensuring the model comprehends the overall graph structure.

$$\mathcal{L}_{ssl}^t = -\sum_{e_{ij} \in \mathcal{E}_m^t} y_{u,v}^t \log \hat{y}_{u,v}^t, \quad (9)$$

where $\mathcal{E}_m^t$ represents the set of masked edges at time t, and treated as positive samples in the training process. Here, $y_{u,v}^t$ is the ground-truth label for the edge e_{ij} at time t, and $\hat{y}_{u,v}^t$ is the predicted result.

4.2 Adaptive Frequency Enhancing Graph Representation Learner

We propose this module to enhance dynamic graph representations through spectral-temporal processing. It transforms node embeddings via FFT, applies learnable frequency filters to capture intrinsic features under distribution shifting, and employs a sliding window to capture long-term dependencies through gradient aggregation.

Graph Convolutional Encoder

We feed the retained graph $\mathcal{G}_r$ into a l-layer decoupled GCN [Wu *et al.*, 2019a] to ensure the model focuses on the more critical information:

$$\mathbf{H}_t^{(l)} = \hat{\mathbf{A}}\mathbf{H}_t^{(l-1)}\mathbf{W}_h^{(l)}, \quad (10)$$

where $\mathbf{H}_t^{(l)} \in \mathbb{R}^{n \times d}$ is the representation of the nodes derived at l-th layer at time t, $\hat{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ denotes the normalized adjacent matrix given that $\mathbf{A}$ is the adjacent matrix with self-loops and $\mathbf{D}$ is the corresponding diagonal degree matrix, and $\mathbf{W}^{(l)}$ is a learnable parameter matrix at l-th layer.

Adaptive Frequency Enhancing Layer

Inspired by FreeDyG [Tian *et al.*, 2023], we enhance key frequency components in the graph representations using Fourier-based modulation. Given the l-th layer GNN output $\mathbf{H}_t^{(l)} \in \mathbb{R}^{L \times d}$ at time t, we first apply the 1D Fast Fourier Transform (FFT) along the temporal dimension:

$$\mathcal{H}_t^{(l)} = \mathcal{F}\left(\mathbf{H}_t^{(l)}\right), \quad (11)$$

where $\mathcal{F}$ denotes the 1D FFT, and $\mathbf{H}_t^{(l)} \in \mathbb{C}^{\lceil \frac{L}{2}+1 \rceil \times d}$ represents the frequency components of $\mathbf{H}_t^{(l)}$, with $\mathbb{C}$ denoting the complex number domain.

Next, we adaptively enhance the frequency components by multiplying them element-wise with a learnable complex number tensor $\mathcal{W} \in \mathbb{C}^{\lceil \frac{L}{2}+1 \rceil \times d}$:

$$\widehat{\mathcal{H}}_t^{(l)} = \mathcal{W} \cdot \mathcal{H}_t^{(l)}, \quad (12)$$

where $(\cdot)$ represents element-wise multiplication, and $\widehat{\mathcal{H}}_t^{(l)} \in \mathbb{C}^{\lceil \frac{L}{2}+1 \rceil \times d}$ denotes the enhanced frequency components.

Finally, we apply the inverse FFT to transform the enhanced frequency components $\widehat{\mathcal{H}}_t^{(l)}$ back into the time domain:

$$\hat{\mathbf{H}}_t^{(l)} \leftarrow \mathcal{F}^{-1}\left(\widehat{\mathcal{H}}_t^{(l)}\right), \tag{13}$$

where $\mathcal{F}^{-1}$ denotes the inverse 1D FFT, converting the complex number tensor back into a real number tensor. Then, a residual connection and dropout are applied to the resulting tensor:

$$\mathbf{H}_t^{(l)} \leftarrow \mathbf{H}_t^{(l)} + \text{Dropout}\left(\hat{\mathbf{H}}_t^{(l)}\right). \tag{14}$$

Then, we predict the probability of edge from node u to node v through a multi-layer perceptron (MLP):

$$\hat{y}_{u,v}^t = \text{MLP}(\text{Concat}(\mathbf{h}_{u,t}^{(l)}, \mathbf{h}_{v,t}^{(l)})), \tag{15}$$

where $\hat{y}_{u,v}^t$ is the predicted label at time t, $\mathbf{h}_{u,t}^{(l)}$ and $\mathbf{h}_{v,t}^{(l)}$ are the feature embeddings of nodes u and v, obtained from $\mathbf{H}_t^l$.

Sliding Window Gradient Aggregation

Inspired by WinGNN [Zhu *et al.*, 2023], we also adopt a sliding window gradient aggregation strategy to model long-term dependencies in dynamic graphs. This strategy utilizes two parameter layers: inter-window parameters Θ^{w_i} and intra-window parameters $\theta_t^{w_i}$. At each time step t, the inner parameters $\theta_t^{w_i}$ are updated using snapshot-specific gradients, while the outer parameters Θ^{w_i} are updated by aggregating the inner gradients at the end of each window w_i. Based on the label data $y_{u,v}^t$ on G^t within window w_i, we can obtain the train loss via cross-entropy loss:

$$\begin{aligned}\mathcal{L}_{G^t}^{w_i} = \mathcal{L}_{ssl}^t &- \sum_{(u,v)\in\mathcal{E}_r^t} y_{u,v}^t \log \hat{y}_{u,v}^t \\ &- \sum_{(u,v)\in\mathcal{E}_{neg}^t} (1-y_{u,v}^t)\log(1-\hat{y}_{u,v}^t),\end{aligned} \tag{16}$$

where $\mathcal{L}_{ssl}^t$ is the self-supervised loss term, which includes contributions from the masked edges $\mathcal{E}_m^t$, treated as positive samples. The second term computes the loss for the retained edges $\mathcal{E}_r^t$, using true labels $y_{u,v}^t$ and predicted probabilities $\hat{y}_{u,v}^t$. The third term calculates the loss for negative samples $\mathcal{E}_{\text{neg}}^t$. The gradient aggregation process is defined as:

$$\theta_{t+1}^{w_i} \leftarrow \theta_t^{w_i} + \tau \cdot \frac{\partial \mathcal{L}_{G^t}^{w_i}}{\partial \theta_t^{w_i}}, \tag{17}$$

where $\theta_t^{w_i}$ represents the model parameters at time step t within sliding window w_i, $\mathcal{L}_{G^t}^{w_i}$ is the loss function for the snapshot at time t within w_i, and τ is the learning rate for θ.

The gradient aggregation across multiple snapshots in the sliding window w_i is computed as follows:

$$\nabla\mathcal{L}^{w_i} = \sum_{t=t_0}^{t_0+l_w} \frac{\partial \mathcal{L}_{G^t}^{w_i}}{\partial \theta_t^{w_i}} \cdot \zeta_t, \tag{18}$$

where $\nabla\mathcal{L}^{w_i}$ denotes the aggregated gradient over the window, and ζ_t represents the weight assigned to each snapshot.

For the next window, the inter-window parameters $\Theta^{w_{i+1}}$ are updated as follows:

$$\Theta^{w_{i+1}} = \Theta^{w_i} + \beta \cdot \nabla\mathcal{L}^{w_i} + \beta \cdot \frac{\partial \mathcal{L}_{G^t}^{w_i}}{\partial \theta_t^{w_i}}, \tag{19}$$

where β is the learning rate for Θ, and Θ^{w_i} represents the outer parameters in window i.

5 Experiments

In this section, we perform model evaluation to investigate the effectiveness of our MaskDGNN[1] and baseline methods on five real-world datasets. Our experiments aim to answer the research questions as follows:

- **RQ1:** What is the performance of our MaskDGNN as compared to various state-of-the-art social relationship inference methods?
- **RQ2:** How do the key components contribute to the performance?
- **RQ3:** How do the key hyperparameters influence the performance of the proposed MaskDGNN?

Dataset	#Nodes	#Edges	#Snapshots
Bitcoin-Alpha	3,783	24,186	226
Bitcoin-OTC	5,881	35,592	262
UCI-Message	1,899	59,835	28
MOOC	7,144	411,749	30
Wiki-Talk	1,140,149	7,833,140	73

Table 1: Summary of dataset statistics.

5.1 Datasets

In our experiments, we evaluate the performance of all methods using five publicly available real-world datasets: Bitcoin-Alpha, Bitcoin-OTC [Kumar *et al.*, 2016], UCI-Message [Panzarasa *et al.*, 2009], MOOC [Kumar *et al.*, 2019], and Wiki-Talk [De Rijke, 2017]. Notably, Wiki-Talk is a large-scale dynamic graph dataset comprising 1 million nodes and 7 million edges, which serves as a robust benchmark to assess the scalability and performance of our model in handling large-scale dynamic graphs. Detailed statistical information for all datasets is provided in Table 1.

5.2 Baselines

We compare our MaskDGNN with five baselines:

- **EvolveGCN** [Pareja *et al.*, 2020]: It captures the evolution of GCN parameters by using sequence models like GRU or LSTM to encode parameters at each time step.
- **DGNN** [Manessi *et al.*, 2020]: It employs a stacked encoder to model node dynamics by leveraging LSTM on the representations encoded by GNNs.
- **dyngraph2vec** [Goyal *et al.*, 2020]: It captures temporal transitions in dynamic graphs through an auto-encoder architecture that integrates dense and recurrent layers.

[1]https://github.com/heyimingheyiming/MaskDGNN

Dataset	Metric	EvolveGCN	DGNN	dyngraph2vec	ROLAND	WinGNN	MaskDGNN	%Improv.
Bitcoin-Alpha	Accuracy	51.99±0.25	OOM.	OOM.	66.21±2.76	85.44±0.78	***95.54±0.44**	11.82%
	AUC	63.71±1.03	OOM.	OOM.	90.21±1.18	92.13±0.62	***97.82±0.31**	6.18%
	MRR	3.28±0.28	OOM.	OOM.	14.52±0.65	23.20±6.61	***31.53±3.19**	35.86%
	Recall@10	7.06±1.19	OOM.	OOM.	31.25±2.28	66.75±4.12	***77.16±1.83**	15.60%
Bitcoin-OTC	Accuracy	50.48±0.03	54.08±0.68	58.29±4.55	86.60±0.52	87.25±1.57	***90.98±0.55**	4.29%
	AUC	55.38±1.66	59.13±6.49	62.12±10.75	90.07±1.30	91.86±0.74	***94.25±0.38**	2.62%
	MRR	11.27±0.58	15.16±0.58	35.39±2.50	16.54±1.22	38.31±1.81	***42.76±1.95**	11.62%
	Recall@10	20.58±1.65	31.09±2.16	58.29±6.74	41.77±3.39	73.93±1.30	***76.20±1.83**	3.07%
UCI-Message	Accuracy	59.85±2.54	50.91±0.05	50.88±3.11	81.83±0.64	84.73±1.36	***87.29±1.28**	3.02%
	AUC	71.99±1.83	52.19±0.56	54.30±1.14	91.81±0.31	93.63±0.89	***96.04±0.37**	2.57%
	MRR	8.17±0.23	1.52±0.01	17.84±0.49	11.84±0.26	22.10±1.21	***24.91±1.72**	12.71%
	Recall@10	14.37±0.49	4.56±0.73	36.22±1.67	25.14±0.92	40.83±1.19	***45.78±1.33**	12.12%
MOOC	Accuracy	75.80±0.86	OOM.	OOM.	89.12±0.32	87.39±0.14	***98.78±0.05**	10.84%
	AUC	87.05±0.51	OOM.	OOM.	94.91±0.16	98.72±0.03	**99.76±0.06**	1.05%
	MRR	7.81±0.63	OOM.	OOM.	52.97±3.74	60.16±1.42	***63.65±0.83**	5.80%
	Recall@10	15.92±0.93	OOM.	OOM.	84.52±1.47	95.23±0.59	***97.86±0.16**	2.76%
Wiki-Talk	Accuracy	OOM.	OOM.	OOM.	71.43±11.87	89.48±0.12	***94.31±0.14**	5.40%
	AUC	OOM.	OOM.	OOM.	83.12±13.15	98.43±0.03	**99.12±0.02**	0.7%
	MRR	OOM.	OOM.	OOM.	26.09±10.43	33.48±0.72	***34.60±1.18**	3.35%
	Recall@10	OOM.	OOM.	OOM.	45.12±14.35	61.52±1.03	***63.37±1.36**	3.01%

Table 2: Performance comparison of all models on five real-world datasets. "%Improv." represents the performance gain of the best result compared to the second-best result. The best results are highlighted in **bold**, and the best among the baselines is underlined. "OOM." indicates an out-of-memory error encountered during model execution in our environment, and * indicates statistical significance (p-value $<$ 0.05) between MaskDGNN and the second-best baseline.

- **ROLAND** [You *et al.*, 2022]: It offers a universal framework to adapt any static GNN to dynamic settings through meta-learning, by treating node embeddings as hierarchical states and recursively updating them over time.
- **WinGNN** [Zhu *et al.*, 2023]: It enhances ROLAND by incorporating a sliding window mechanism to reduce model size, alleviating computational and storage demands in long-sequence scenarios.

5.3 Experiment Settings and Evaluation Metrics

In our evaluation, we use Accuracy (Acc for short), Area Under the Curve (AUC for short), Mean Reciprocal Rank (MRR for short), and Recall@10 (R@10 for short) to quantify the performance of different methods. For the baselines, we adopt the parameter settings recommended in their respective papers and further fine-tune them to achieve optimal performance. In our model, the node embedding dimension d is set to 64, the mask ratio p ranges from 10% to 40% depending on the dataset, the node dynamics score ratio α is set to 0.7, and the window size w is determined based on the snapshot length, typically ranging from 4 to 8 for each dataset. The parameters λ, η, and γ are set to 1.0, 2.0, and -0.5, respectively. The learning rate for θ, denoted by τ, is 0.008, while the learning rate for Θ, denoted by β, is set to 0.01. The dropout rate is fixed at 0.1, and the Graph Convolutional Encoder consists of 2 layers. To prevent overfitting, we apply early stopping during validation with a patience of 10. Experiments for each method are conducted ten times, and the average results are reported. All experiments are performed on four RTX 3090 GPUs with 24GB of memory.

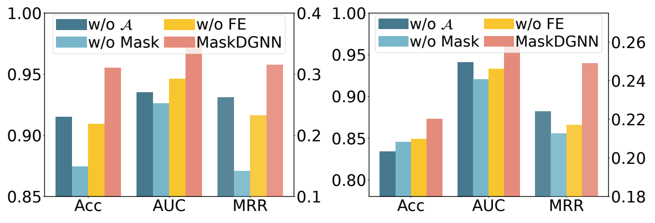
5.4 Experiment Results

Performance Comparison (RQ1)

We evaluate the performance of MaskDGNN and all baselines in link prediction tasks on five real-world datasets. Experimental results are shown in Table 2. The best results are highlighted in bold, and the second-best ones are underlined.

As shown in Table 2, MaskDGNN outperforms all models across all metrics. For instance, on the Bitcoin-Alpha dataset, it boosts MRR by 35.86% and improves R@10, AUC, and Acc by 15.60%, 11.82%, and 6.18%, respectively. On the UCI-Message dataset, MaskDGNN surpasses the second-best model by 12.71% in MRR and 12.12% in R@10. This strong performance is due to its ability to capture intrinsic features during distribution shifts, enabling adaptation to dynamic graph changes, particularly in datasets with high node activeness and rapid fluctuations like Bitcoin-Alpha and UCI-Message. On the MOOC dataset, with higher edge density, MaskDGNN achieves 98.78% in Acc and 99.76% in AUC, benefiting from the combined effect of the node activeness-aware masking and adaptive frequency enhancing strategies. Overall, MaskDGNN shows significant improvements across four key metrics, with average gains of 7.07% in Acc, 2.62% in AUC, 13.87% in MRR, and 7.31% in R@10, highlighting its robustness and effectiveness.

MaskDGNN also performs robustly on large-scale dynamic graphs. On the Wiki-Talk dataset, which contains over 1 million nodes and 7 million edges, most baseline models fail due to out-of-memory (OOM) issues. In contrast, MaskDGNN not only runs successfully but also achieves 93.30% in Acc and 99.12% in AUC. It leverages its node activeness-aware masking mechanism to remove a significant



(a) Bitcoin-Alpha **(b)** UCI-Message

Figure 2: The comparison of MaskDGNN and its variants on Bitcoin-Alpha and UCI-Message datasets, where Acc and AUC are represented using the left vertical axis, while MRR is shown using the right vertical axis.

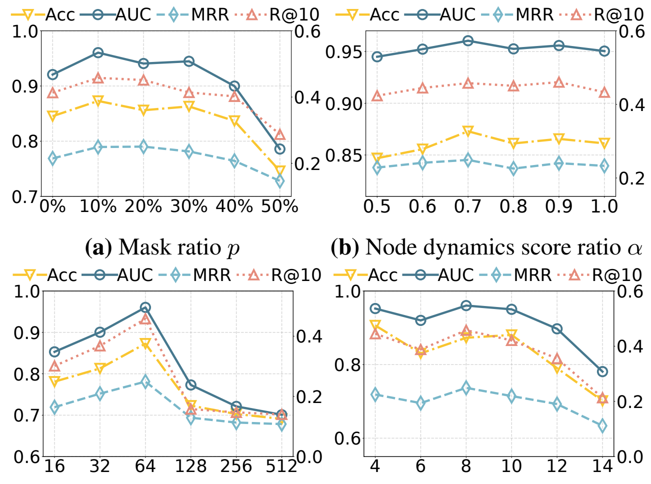
proportion of redundant edges (about 30%-40%), reducing the graph size and avoiding memory issues. Additionally, it adopts an adaptive frequency enhancing strategy, optimizing the graph data by capturing important frequency components, which enhances its ability to capture temporal features under distribution shifting.

Ablation Study (RQ2)

To analyze the effectiveness of the key components in our model, we conduct the following ablation studies:

- ***w/o*** $\mathcal{A}$ removes the mask rate node-activeness scores and conducts random mask mechanism.
- ***w/o* Mask** removes the self-supervised activeness-aware temporal masking module.
- ***w/o* FE** removes the frequency enhancing layer.

We evaluate the contribution of each key component in MaskDGNN by individually removing them and analyzing their impact on performance on the Bitcoin-Alpha and UCI-Message datasets. As shown in Figure 2, when the node activeness-aware temporal masking mechanism is replaced by random masking (***w/o*** $\mathcal{A}$), there is a significant performance drop, especially in Acc (from 95.54% to 91.52% in Bitcoin-Alpha, and from 87.29% to 83.40% in UCI-Message). This phenomenon highlights the crucial role of activeness-aware masking in prioritizing key temporal patterns. Similarly, the removal of the self-supervised activeness-aware temporal masking module (***w/o* Mask**) results in an even more pronounced performance degradation, with MRR dropping from 31.53% to 14.15% in Bitcoin-Alpha and from 24.91% to 21.27% in UCI-Message. This further emphasizes the module's effectiveness in filtering out redundant information and focusing on key temporal changes. Additionally, removing the frequency-enhancing graph representation learner (***w/o* FE**) reduces MRR to 23.24% in Bitcoin-Alpha and to 21.72% in UCI-Message, demonstrating the module's ability to capture intrinsic features under distribution shift. These results collectively confirm that the activeness-aware temporal masking module and the adaptive frequency-enhancing graph representation learner are key to MaskDGNN's superior performance, enabling it to effectively capture important temporal dynamics while suppressing redundant information.



(a) Mask ratio p **(b)** Node dynamics score ratio α

(c) Embedding dimension d **(d)** Sliding window size w

Figure 3: Hyper-parameter impact of mask ratio p, node dynamics score ratio α, embedding dimension d, sliding window size w on UCI-Message dataset, where Acc and AUC are represented using the left vertical axis, while MRR and R@10 are shown using the right vertical axis.

Parameter Sensitivity Analysis (RQ3)

We investigate the sensitivity of four main parameters in the UCI dataset: mask ratio p, node dynamic score ratio α, embedding dimension d, and sliding window size w.

Mask ratio p. Figure 3a shows that the model achieves optimal performance at 10% masking, with stable results between 10% and 30%. Beyond 40%, performance drops sharply, indicating that while 10%-30% of edges are redundant, excessive masking may omit critical information. An appropriate mask ratio helps focus on essential features.

Node dynamics score ratio α. As shown in Figure 3b, the best performance occurs at $\alpha = 0.7$, with a slight drop at $\alpha = 0.5$, highlighting the importance of balancing node dynamics with global importance.

Embedding dimension d. Figure 3c shows performance improves up to $d = 128$, then declines. Smaller d fails to capture enough information, while larger d adds computational cost without benefit.

Sliding window size w. Figure 3d shows that MaskDGNN performs best with $w = 8$, effectively capturing long-term dependencies. Larger w increases cost without notable gain, so selecting an appropriate w balances performance and efficiency.

6 Conclusion

In this work, we propose a self-supervised dynamic graph neural network named MaskDGNN that combines a self-supervised activeness-aware temporal masking mechanism with an adaptive frequency enhancing graph representation learner. MaskDGNN effectively reduces redundant information and captures intrinsic temporal features under distribution shifting. Extensive experiments on five real-world dynamic graph datasets demonstrate the superiority and effectiveness of our model.

Acknowledgments

This work is partially supported by the National Natural Science Foundation of China under Grant Nos. 62176243, 62472263, and 62402465, the Fundamental Research Funds for the Central Universities under Grant No. 202442005, and the National Key R&D Program of China under Grant No. 2022ZD0117201.

References

[Cho *et al.*, 2014] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[Cong *et al.*, 2021] Weilin Cong, Yanhong Wu, Yuandong Tian, Mengting Gu, Yinglong Xia, Mehrdad Mahdavi, and Chun-cheng Jason Chen. Dynamic graph representation learning via graph transformer networks. 2021.

[Cong *et al.*, 2023] Weilin Cong, Si Zhang, Jian Kang, Baichuan Yuan, Hao Wu, Xin Zhou, Hanghang Tong, and Mehrdad Mahdavi. Do we really need complicated model architectures for temporal networks? *arXiv preprint arXiv:2302.11636*, 2023.

[De Rijke, 2017] Maarten De Rijke. *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. Association for Computing Machinery, 2017.

[Finn *et al.*, 2017] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proc. ICML*, pages 1126–1135. PMLR, 2017.

[Fu *et al.*, 2023] Chaofan Fu, Guanjie Zheng, Chao Huang, Yanwei Yu, and Junyu Dong. Multiplex heterogeneous graph neural network with behavior pattern modeling. In *Proc. ACM SIGKDD*, pages 482–494, 2023.

[Goyal *et al.*, 2020] Palash Goyal, Sujit Rokka Chhetri, and Arquimedes Canedo. dyngraph2vec: Capturing network dynamics using dynamic graph representation learning. *Knowledge-Based Systems*, 187:104816, 2020.

[Hassani and Khasahmadi, 2020] Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive multi-view representation learning on graphs. In *Proc. ICML*, pages 4116–4126. PMLR, 2020.

[He *et al.*, 2022] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proc. CVPR*, pages 16000–16009, 2022.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Hou *et al.*, 2022] Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. Graphmae: Self-supervised masked graph autoencoders. In *Proc. ACM SIGKDD*, pages 594–604, 2022.

[Jin *et al.*, 2022] Ming Jin, Yuan-Fang Li, and Shirui Pan. Neural temporal walks: Motif-aware representation learning on continuous-time dynamic graphs. *Proc. NeurIPS*, 35:19874–19886, 2022.

[Kachole *et al.*, 2023] Sanket Kachole, Yusra Alkendi, Fariborz Baghaei Naeini, Dimitrios Makris, and Yahya Zweiri. Asynchronous events-based panoptic segmentation using graph mixer neural network. In *Proc. CVPR*, pages 4082–4091, 2023.

[Kazemi *et al.*, 2020] Seyed Mehran Kazemi, Rishab Goel, Kshitij Jain, Ivan Kobyzev, Akshay Sethi, Peter Forsyth, and Pascal Poupart. Representation learning for dynamic graphs: A survey. *J. Mach. Learn. Res.*, 21(70):1–73, 2020.

[Kumar *et al.*, 2016] Srijan Kumar, Francesca Spezzano, VS Subrahmanian, and Christos Faloutsos. Edge weight prediction in weighted signed networks. In *Proc. IEEE ICDM*, pages 221–230. IEEE, 2016.

[Kumar *et al.*, 2019] Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proc. ACM SIGKDD*, pages 1269–1278, 2019.

[Li *et al.*, 2023] Jintang Li, Ruofan Wu, Wangbin Sun, Liang Chen, Sheng Tian, Liang Zhu, Changhua Meng, Zibin Zheng, and Weiqiang Wang. What's behind the mask: Understanding masked graph modeling for graph autoencoders. In *Proc. ACM SIGKDD*, pages 1268–1279, 2023.

[Liu *et al.*, 2020] Zhijun Liu, Chao Huang, Yanwei Yu, Peng Song, Baode Fan, and Junyu Dong. Dynamic representation learning for large-scale attributed networks. In *Proc. ACM CIKM*, pages 1005–1014, 2020.

[Liu *et al.*, 2021] Zhijun Liu, Chao Huang, Yanwei Yu, and Junyu Dong. Motif-preserving dynamic attributed network embedding. In *Proc. Web Conf.*, pages 1629–1638, 2021.

[Liu *et al.*, 2024a] Chuang Liu, Yuyao Wang, Yibing Zhan, Xueqi Ma, Dapeng Tao, Jia Wu, and Wenbin Hu. Where to mask: Structure-guided masking for graph masked autoencoders. *arXiv preprint arXiv:2404.15806*, 2024.

[Liu *et al.*, 2024b] Lingwen Liu, Guangqi Wen, Peng Cao, Jinzhu Yang, Weiping Li, and Osmar R Zaiane. Capturing temporal node evolution via self-supervised learning: a new perspective on dynamic graph learning. In *Proc. WSDM*, pages 443–451, 2024.

[Ma *et al.*, 2020] Yao Ma, Ziyi Guo, Zhaocun Ren, Jiliang Tang, and Dawei Yin. Streaming graph neural networks. In *Proc. ACM SIGIR*, pages 719–728, 2020.

[Manessi *et al.*, 2020] Franco Manessi, Alessandro Rozza, and Mario Manzo. Dynamic graph convolutional networks. *Pattern Recognition*, 97:107000, 2020.

[Panzarasa *et al.*, 2009] Pietro Panzarasa, Tore Opsahl, and Kathleen M Carley. Patterns and dynamics of users' behavior and interaction: Network analysis of an online community. *J. Am. Soc. Inf. Sci. Technol.*, 60(5):911–932, 2009.

[Pareja *et al.*, 2020] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In *Proc. AAAI*, volume 34, pages 5363–5370, 2020.

[Poursafaei *et al.*, 2022] Farimah Poursafaei, Shenyang Huang, Kellin Pelrine, and Reihaneh Rabbany. Towards better evaluation for dynamic link prediction. *Proc. NeurIPS*, 35:32928–32941, 2022.

[Rossi *et al.*, 2020] Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*, 2020.

[Sankar *et al.*, 2020] Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao Yang. Dysat: Deep neural representation learning on dynamic graphs via self-attention networks. In *Proc. WSDM*, pages 519–527, 2020.

[Sharma *et al.*, 2023] Kartik Sharma, Mohit Raghavendra, Yeon-Chang Lee, and Srijan Kumar. Representation learning in continuous-time dynamic signed networks. In *Proc. ACM CIKM*, pages 2229–2238, 2023.

[Song *et al.*, 2019] Weiping Song, Zhiping Xiao, Yifan Wang, Laurent Charlin, Ming Zhang, and Jian Tang. Session-based social recommendation via dynamic graph attention networks. In *Proc. WSDM*, pages 555–563, 2019.

[Tan *et al.*, 2023] Qiaoyu Tan, Ninghao Liu, Xiao Huang, Soo-Hyun Choi, Li Li, Rui Chen, and Xia Hu. S2gae: Self-supervised graph autoencoders are generalizable learners with graph masking. In *Proc. WSDM*, pages 787–795, 2023.

[Tian *et al.*, 2023] Yuxing Tian, Yiyan Qi, and Fan Guo. Freedyg: Frequency enhanced continuous-time dynamic graph model for link prediction. In *Proc. ICLR*, 2023.

[Trivedi *et al.*, 2019] Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. Dyrep: Learning representations over dynamic graphs. In *Proc. ICLR*, 2019.

[Vaswani, 2017] A Vaswani. Attention is all you need. *Proc. NeurIPS*, 2017.

[Wang *et al.*, 2021] Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. Inductive representation learning in temporal networks via causal anonymous walks. *arXiv preprint arXiv:2101.05974*, 2021.

[Wu *et al.*, 2019a] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *Proc. ICML*, pages 6861–6871. PMLR, 2019.

[Wu *et al.*, 2019b] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Graph wavenet for deep spatial-temporal graph modeling. *arXiv preprint arXiv:1906.00121*, 2019.

[Xu *et al.*, 2020] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. *arXiv preprint arXiv:2002.07962*, 2020.

[Xue *et al.*, 2022] Guotong Xue, Ming Zhong, Jianxin Li, Jia Chen, Chengshuai Zhai, and Ruochen Kong. Dynamic network embedding survey. *Neurocomputing*, 472:212–223, 2022.

[Yang *et al.*, 2024] Leshanshui Yang, Clément Chatelain, and Sébastien Adam. Dynamic graph representation learning with neural networks: A survey. *IEEE Access*, 12:43460–43484, 2024.

[You *et al.*, 2021] Yuning You, Tianlong Chen, Yang Shen, and Zhangyang Wang. Graph contrastive learning automated. In *Proc. ICML*, pages 12121–12132. PMLR, 2021.

[You *et al.*, 2022] Jiaxuan You, Tianyu Du, and Jure Leskovec. Roland: graph learning framework for dynamic graphs. In *Proc. ACM SIGKDD*, pages 2358–2366, 2022.

[Yu *et al.*, 2022] Pengyang Yu, Chaofan Fu, Yanwei Yu, Chao Huang, Zhongying Zhao, and Junyu Dong. Multiplex heterogeneous graph convolutional network. In *Proc. ACM SIGKDD*, pages 2377–2387, 2022.

[Yu *et al.*, 2023] Le Yu, Leilei Sun, Bowen Du, and Weifeng Lv. Towards better dynamic graph learning: New architecture and unified library. *Proc. NeurIPS*, 36:67686–67700, 2023.

[Zhang *et al.*, 2022] Mengqi Zhang, Shu Wu, Xueli Yu, Qiang Liu, and Liang Wang. Dynamic graph neural networks for sequential recommendation. *IEEE Trans. Knowl. Data Eng.*, 35(5):4741–4753, 2022.

[Zhang *et al.*, 2023] Kaike Zhang, Qi Cao, Gaolin Fang, Bingbing Xu, Hongjian Zou, Huawei Shen, and Xueqi Cheng. Dyted: Disentangled representation learning for discrete-time dynamic graph. In *Proc. ACM SIGKDD*, pages 3309–3320, 2023.

[Zhang *et al.*, 2024] Zeyang Zhang, Xin Wang, Ziwei Zhang, Zhou Qin, Weigao Wen, Hui Xue, Haoyang Li, and Wenwu Zhu. Spectral invariant learning for dynamic graphs under distribution shifts. *Proc. NeurIPS*, 36, 2024.

[Zhao *et al.*, 2024] Ziwen Zhao, Yuhua Li, Yixiong Zou, Jiliang Tang, and Ruixuan Li. Masked graph autoencoder with non-discrete bandwidths. In *Proc. Web Conf.*, pages 377–388, 2024.

[Zhu *et al.*, 2023] Yifan Zhu, Fangpeng Cong, Dan Zhang, Wenwen Gong, Qika Lin, Wenzheng Feng, Yuxiao Dong, and Jie Tang. Wingnn: Dynamic graph neural networks with random gradient aggregation window. In *Proc. ACM SIGKDD*, pages 3650–3662, 2023.