DyGMAE: A Novel Dynamic Graph Masked Autoencoder for Link Prediction

Weixiong Liu[1] **Junwei Cheng**[1] **Zhongyu Pan**[1] **Chaobo He**[*1] **Quanlong Guan**[2]

[1]School of Computer Science, South China Normal University, Guangzhou, China
[2]College of Information Science and Technology, Jinan University, Guangzhou, China

Abstract

Dynamic link prediction (DLP) is a crucial task in graph learning, aiming to predict future links between nodes at subsequent time in dynamic graphs. Recently, graph masked autoencoders (GMAEs) have shown promising performance in self-supervised learning. However, their application to DLP is under-explored. Existing GMAEs struggle to capture temporal dependencies, and their random masking causes crucial information loss for DLP. Moreover, most existing DLP methods rely on local information, ignoring global information and failing to capture complex features in real-world dynamic graphs. To address these issues, we propose DyGMAE, a novel dynamic GMAE method specifically designed for DLP. DyGMAE introduces a Multi-Scale Masking Strategy (MSMS), which generates multiple graph views by masking parts of the edges and tries to reconstruct them. Additionally, a multi-scale masking representation alignment module with a contrastive learning objective is employed to align representations which are encoded by unmasked edges across these views. Through this design, different masked views can provide diverse information to alleviate the drawbacks of random masking, and contrastive learning can align different views to mitigate the problem of exploiting local and global information simultaneously. Experiments on benchmark datasets show DyGMAE achieves superior performance in the DLP task.

1 INTRODUCTION

Dynamic graphs [Gravina and Bacciu, 2024] exhibit a remarkable capacity to model real-world interaction changes. This distinct feature enables their extensive application in diverse dynamic systems, such as social networks [Min et al., 2021, Cheng et al., 2025], disease transmission networks [Zhu et al., 2022], and transportation systems [Yu et al., 2020]. In dynamic graph learning, dynamic link prediction (DLP) is one fundamental task, which aims to forecast the appearance or disappearance of links over time [Qin and Yeung, 2023, Firouzkouhi et al., 2024], and can be regarded as a temporal extension of static link prediction [He et al., 2023]. DLP plays a crucial role in diverse applications including traffic forecasting [Li et al., 2023a] and disease control [Wang et al., 2022]. Given the complexity of dynamic graphs, which involves intricate structural patterns and temporal dependencies, finding effective methods for DLP remains a significant challenge.

During the pursuit of addressing this challenge, self-supervised learning (SSL) has emerged as a promising approach due to its ability to leverage large amounts of unlabeled data [Gao et al., 2023, Zhang et al., 2023]. In particular, graph masked autoencoders (GMAEs) [Hou et al., 2022, Liu et al., 2024a,b], a generative SSL framework, have recently excelled in graph-related tasks. However, their potential in DLP has not been fully explored. Specifically, GMAE extends the masked autoencoder, an SSL framework in computer vision [He et al., 2022] and natural language processing [Devlin, 2018] without labeled data, to graphs. Compared with other SSL frameworks, GMAEs have achieved success in the static graph link prediction task by masking and reconstructing edges [Li et al., 2023b], which has been found to remarkably boost the prediction accuracy. Based on the above considerations, we reasonably speculate that GMAE might be adaptable to dynamic graphs to enhance the performance of DLP. However, despite their success in static graphs, current GMAE methods face several limitations when applied to dynamic graphs. First, simple random masking easily causes critical information loss, resulting in suboptimal link prediction performance. Second, they struggle to model time-evolving structures vital for DLP because they are designed for static graphs.

[*]Corresponding author: hechaobo@foxmail.com

In addition to the limitations of current GMAE in the dynamic graph scenario, existing DLP methods also suffer from significant drawbacks. Specifically, as [Gao et al., 2023] pointed out, most of these methods are unable to capture the complex features in real-world dynamic graphs comprehensively. They tend to focus only on local information while overlooking global information. In fact, both global and local information in dynamic graphs are crucial for DLP. Local information offers node-specific short-term dynamics, while global information reveals the overall structures and long-term dependencies of the graph. Together, they enable a comprehensive understanding of the graph. For instance, some methods, such as [Hajiramezanali et al., 2019, Yang et al., 2022, 2021, Jiao et al., 2024], focus solely on local structural information using graph neural networks (GNNs), neglecting the broader context. These methods often struggle to capture local and global patterns simultaneously, limiting their ability to represent the full complexity of dynamic graphs. Although DGCN [Gao et al., 2023] tries to capture global information by maximizing mutual information between local and global representations, its heavy reliance on local information for global representation aggregation restricts its performance.

Intuitively, GMAE uses a masking strategy to generate different structural views and reconstruct the masked parts, facilitating the exploration of diverse structures and evolving patterns. Considering GMAE's limitations in DLP and the problem of exploiting both local and global information, we propose DyGMAE, a novel dynamic graph masked autoencoder specifically designed for DLP. In DyGMAE, we introduce a Multi-Scale Masking Strategy (MSMS) to tackle both of the above problems. MSMS generates diverse masked and unmasked views by applying different edge masking techniques and edge masking ratios, and attempts to reconstruct the masked edges from the unmasked portions. Furthermore, in MSMS, we incorporate an alignment module with a contrastive learning objective [He et al., 2025]. This module aims to integrate the information obtained from different masking views, including both local and global information, by aligning different views. By applying the MSMS, DyGMAE can more effectively explore different aspects of the structural patterns and temporal dependencies in multi-reconstruction phases, significantly enhancing its ability to capture comprehensive information and thus improving DLP performance. Finally, the refined embeddings are processed by a gated recurrent unit (GRU) based temporal modeling module, which captures temporal dependencies in dynamic graphs, and are subsequently used for DLP. As a result, DyGMAE effectively captures both the structural evolution and temporal dependencies of dynamic graphs, addressing key challenges in DLP. We summarize our main contributions as follows:

- We propose DyGMAE, a novel dynamic graph masked autoencoder tailored for DLP. It extends the GMAE framework to effectively capture both structural patterns and temporal dependencies in dynamic graphs.
- DyGMAE integrates a MSMS to mitigate the information loss induced by random masking and address the challenge of acquiring both local and global information. To our knowledge, this is the first attempt to combine GMAEs with contrastive learning for DLP.
- Our experimental results show that DyGMAE achieves superior performance in the DLP task across several real-world dynamic graph datasets, outperforming state-of-the-art methods.

2 PRELIMINARIES

In this section, we first introduce notations and formulate the problems, then illustrate the graph masked autoencoder framework.

2.1 NOTATIONS AND PROBLEM FORMULATION

Without loss of generality, we model dynamic graphs as consisting of a series of snapshots, denoted as $\mathcal{G} = \{G_1, G_2, \ldots, G_T\}$, where T represents the total number of snapshots. Each snapshot $G_t = (V_t, E_t)$, for $1 \leq t \leq T$, consists of a node set V_t and an edge set E_t. The adjacency matrix at time step t is denoted by $\mathbf{A}_t \in \{0,1\}^{|V_t|\times|V_t|}$.

DLP aims to forecast the future link states in a dynamic graph by utilizing the historical snapshots $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_T$. Its objective is to predict the link states in the next snapshot: $\hat{\mathbf{A}}_{T+1}$. Dynamic New Link Prediction (DNLP) is a specialized type of DLP task. It focuses on identifying the links that exist in the current snapshot $\mathcal{G}_T$ of dynamic graphs but were absent in the previous snapshot $\mathcal{G}_{T-1}$. Compared with DLP, DNLP is more challenging as it demands stronger generalization capabilities to effectively tackle this task.

2.2 GRAPH MASKED AUTOENCODER

We take a static graph as an example to illustrate the GMAE framework. It has three core components: a masking module $f_{\mathrm{M}}(\cdot)$, an encoder $f_{\mathrm{E}}(\cdot)$, and a decoder $f_{\mathrm{D}}(\cdot)$. Given graph G, the masking module creates a masked graph G^m and an unmasked graph G^u by masking nodes, features, or edges. The unmasked graph G^u goes through the encoder $f_{\mathrm{E}}(\cdot)$ to generate latent representations $\mathbf{Z}$ containing the graph's key information. The decoder $f_{\mathrm{D}}(\cdot)$ then reconstructs an approximation $\hat{G}$ of the original graph from $\mathbf{Z}$.

3 METHODOLOGY

The overall framework of the proposed DyGMAE is shown in Figure 1, detailed as follows: **(A) Overall architecture of**

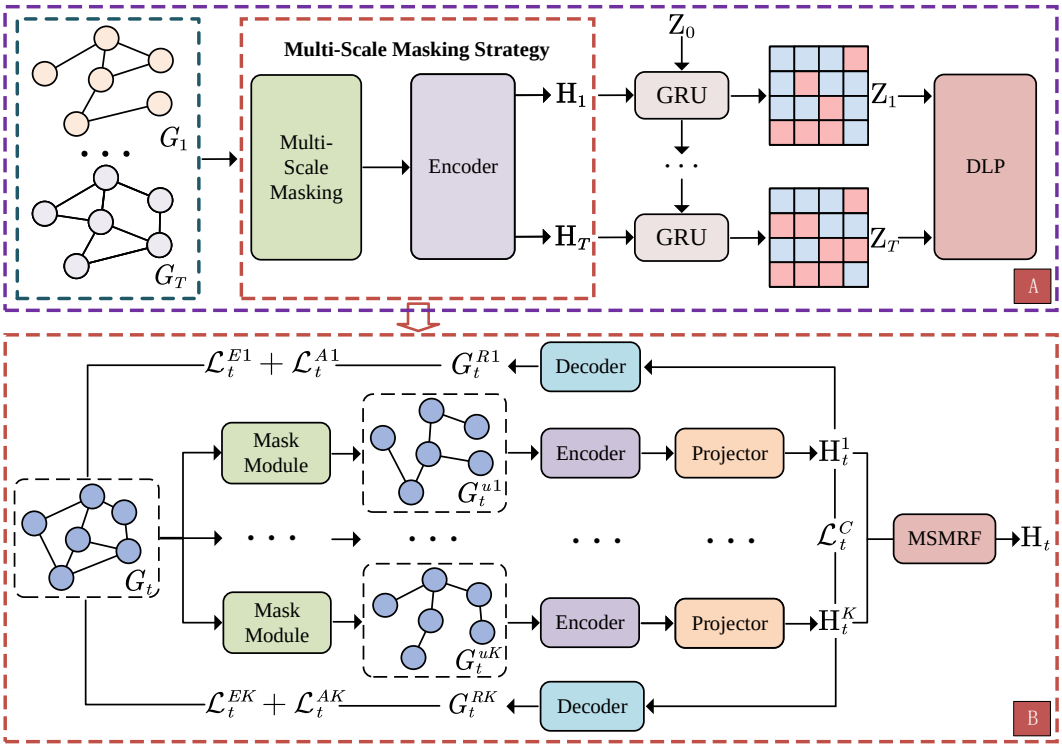


Figure 1: The overall architecture of DyGMAE.

DyGMAE: Dynamic graph snapshots $\{G_1, G_2, \ldots, G_T\}$ are processed by MSMS to generate multiple unmasked views. Aggregated representations from these views are fed into the GRU for temporal modeling, yielding final node representations $\mathbf{Z}_T$ for DLP. **(B) Multi-Scale Masking Strategy (MSMS)**: This module generates unmasked G_t^u graphs through the application of diverse masking strategies to each snapshot, which are passed through an encoder and projector for node representations. The contrastive loss $\mathcal{L}_t^C$ aligns representations of different unmasked views, while edge and adjacency reconstruction losses $\mathcal{L}_t^E$ and $\mathcal{L}_t^A$ refine embeddings. Graph representations are used for Multi-Scale Masking Representation Fusion (MSMRF) across views, then sent to the GRU for further temporal modeling. In the subsequent sections, we explain each DyGMAE module in detail, emphasizing how the GMAE framework adapts to dynamic graphs and how MSMS boosts DLP performance.

3.1 EDGE MASKING STRATEGY

Many related GMAE methods [Hou et al., 2022] mask node features and attempt to reconstruct them. However, our task is DLP, and thus we need to bridge the gap between feature reconstruction and link prediction. To this end, we employ two edge-masking strategies as part of our MSMS: random edge masking and path-wise random masking.

Random Edge Masking. Random edge masking is performed by randomly masking a subset of edges in the graph. The edges to be masked, denoted as E^m, are sampled using a Bernoulli distribution as follows:

$$E^m \sim \text{Bernoulli}(p), \tag{1}$$

$$E^u = E - E^m, \tag{2}$$

where p denotes the probability of masking an edge, and E^u represents the set of remaining edges.

Path-Wise Random Masking. Following the inspiration from [Li et al., 2023b], we adopt the path-wise random masking strategy. Different from random edge masking, it masks edges via random walks [Berahmand et al., 2022] from root nodes. This disrupts local connections, forcing path reconstruction to capture complex structural dependencies and higher-order node relationships. Formally, the masked edges are sampled as:

$$E^m \sim \text{RandomWalk}(R, n_{\text{walk}}, l_{\text{walk}}), \tag{3}$$

where R is a set of randomly selected root nodes, n_{walk} denotes the number of random walks per root node, and l_{walk} is the length of each random walk.

For dynamic graphs, we define the unmasked snapshot at time step t as $G_t^u = (V_t, E_t^u)$, with $E_t^u \subseteq E_t$ being the remaining unmasked edges. The masked snapshot is $G_t^m = (V_t, E_t^m)$, where $E_t^m = E_t - E_t^u$. This approach

adds stochasticity and compels the method to leverage the partially unmasked graph G_t^u and historical information, facilitating the capture of more crucial information.

3.2 MULTI-SCALE MASKING STRATEGY (MSMS)

Multi-Scale Masking. Facing the challenges in DLP with GMAE, which involve the critical information loss induced by random masking strategies and the limitations in acquiring local and global information for complex data modeling, we put forward the MSMS as a solution. MSMS generates multiple masked and unmasked views for each snapshot by applying different edge masking strategies and varying the masking probabilities p as we introduced in Section 3.1. Specifically, given a dynamic graph snapshot $G_t = (V_t, E_t)$, the multi-scale masking creates K unmasked versions of the snapshot through the set of masking strategies $\{f_{\text{M}}^1, f_{\text{M}}^2, \ldots, f_{\text{M}}^K\}$, denoted as $\{G_t^{u1}, G_t^{u2}, \ldots, G_t^{uK}\}$. Each unmasked view captures distinct structural perspectives and different evolving dynamics. Edges are masked according to predefined probabilities $\{p_1, p_2, \ldots, p_K\}$, with each p_i controlling the extent of masking in the corresponding view. These different edge masking lead to various characteristics of the unmasked views. For example, one view may focus on a higher masking ratio to encourage the method to infer global structures, while another view may apply a lower masking ratio to preserve local and fine-grained information.

By introducing multiple unmasked views, the multi-masking strategy enriches the diversity of training data and allows the method to extract richer structural patterns and temporal dependencies from various unmasked views in the edge reconstruction phase. This capability is crucial for DLP, as it enables the model to capture both local and global information in the graph over time, thereby improving its ability to predict future links accurately.

Unmasked Graph Encoder. The dynamic graph snapshot is processed through the multi-scale masking, which generates multiple unmasked views of the graph. Then one such view, $G_t^u = (V_t, E_t^u)$, is fed into the encoder to compute the latent node representations. The encoder consists of a stack of L GNN layers, which operate on the unmasked part of the graph. The latent representation at the $(l+1)$-th layer is computed as:

$$\mathbf{H}_{t,l+1} = \sigma_1\left(\tilde{\mathbf{D}}_t^{-\frac{1}{2}}\tilde{\mathbf{A}}_t^u\tilde{\mathbf{D}}_t^{-\frac{1}{2}}\mathbf{H}_{t,l}\mathbf{W}_l\right), \tag{4}$$

where $\mathbf{H}_{t,l+1} \in \mathbb{R}^{N\times d_{\text{out}}}$ represents the node embeddings at layer $l+1$ for the t-th snapshot. In this equation, N denotes the number of nodes in the graph, and d_{out} is the dimensionality of the output embeddings. $\tilde{\mathbf{A}}_t^u = \mathbf{A}_t^u + \mathbf{I} \in \mathbb{R}^{N\times N}$ represents the unmasked adjacency matrix of the t-th snapshot with self-loops added, where $\mathbf{A}_t^u$ is the adjacency matrix of the unmasked graph G_t^u, generated by one of the multi-scale masking views. The diagonal degree matrix corresponding to $\tilde{\mathbf{A}}_t^u$ is denoted as $\tilde{\mathbf{D}}_t \in \mathbb{R}^{N\times N}$. $\mathbf{H}_{t,l} \in \mathbb{R}^{N\times d_{\text{in}}}$ represents the input node embeddings at layer l, where d_{in} is the dimensionality of the input features. Finally, $\mathbf{W}_l \in \mathbb{R}^{d_{\text{in}}\times d_{\text{out}}}$ is the learnable weight matrix of the l-th GNN layer, and $\sigma_1(\cdot)$ is an activation function such as ReLU.

Projector. To refine and enhance the latent representations produced by the encoder for link prediction, we introduce a multi-layer perceptron (MLP) as a projector following the encoder. The transformation is defined as:

$$\mathbf{H}_t = \mathbf{MLP}(\mathbf{H}_{t,L}), \tag{5}$$

where $\mathbf{H}_t \in \mathbb{R}^{N\times d_{\text{proj}}}$ represents the refined node embeddings after projection, and $\mathbf{H}_{t,L} \in \mathbb{R}^{N\times d_{\text{out}}}$ denotes the latent representations output from the last GNN layer of the encoder. The MLP is used to project the embeddings into a new space, where d_{proj} is the dimensionality of the projected embeddings.

Representation Refinement via Reconstruction. In order to ensure that the representations generated by each unmasked view contain more informative and critical structural features for reconstructing the masked edges and get better DLP performance, we feed each representation into the decoder to enhance its reconstruction ability and mitigate the issues of noise and redundancy. We employ two types of loss functions to guide the training process. Next, for simplicity, we will explain the loss functions using one view as an example.

The first loss function, edge reconstruction loss $\mathcal{L}_t^E$, is designed to leverage the unmasked edges to reconstruct the masked edges, the loss function maximizes the probability of the masked link nodes in the original graph while minimizing the probability of unlinked nodes, thereby promoting the reconstruction of the graph structure and enhancing the method's reconstruction ability. The loss is formulated using a binary cross-entropy function, as follows:

$$\mathcal{L}_t^+ = \frac{1}{|E_t^+|}\sum_{(i,j)\in E_t^+}\log p_f(\mathbf{h}_{t,i}, \mathbf{h}_{t,j}), \tag{6}$$

$$\mathcal{L}_t^- = \frac{1}{|E_t^-|}\sum_{(i',j')\in E_t^-}\log(1 - p_f(\mathbf{h}_{t,i'}, \mathbf{h}_{t,j'})), \tag{7}$$

$$\mathcal{L}_t^E = -\left(\mathcal{L}_t^+ + \mathcal{L}_t^-\right), \tag{8}$$

where the set of positive edges E_t^+ corresponds to the edges present in the masked graph at time t, while E_t^- is the set of negative edges, with the number of negative edges equal to the number of positive edges. $p_f(\mathbf{h}_{t,i}, \mathbf{h}_{t,j})$ computes the probability that there is an edge between nodes i and j, based on their embeddings $\mathbf{h}_{t,i}$ and $\mathbf{h}_{t,j}$ at time t. By minimizing $\mathcal{L}_t^E$, the method learns to reconstruct the masked edges, resulting in better DLP performance.

The second loss function, adjacency matrix reconstruction loss $\mathcal{L}_t^A$, is designed to recover the original graph connections from the latent representations derived from the unmasked parts of the graph. The reconstruction error for the graph structure at time t is defined as:

$$\mathcal{L}_t^A = \|\mathbf{A}_t - \tilde{\mathbf{A}}_{t,\text{pre}}\|_F^2, \tag{9}$$

where $\tilde{\mathbf{A}}_{t,\text{pre}}$ represents the predicted adjacency matrix at time t and is decoded by $\mathbf{H}_t$ through two MLP layers. And $\mathbf{A}_t$ is the ground-truth adjacency matrix. Since the elements of $\mathbf{A}_t$ are predominantly zeros due to the graph's sparsity, we adopt the following loss to improve the method's reconstruction ability [Richard et al., 2012]:

$$\mathcal{L}_t^A = \|\mathbf{A}_{t+1} - \tilde{\mathbf{A}}_{t+1}\|_F^2 + \gamma\|\tilde{\mathbf{A}}_{t+1}\|_1 + \delta\|\tilde{\mathbf{A}}_{t+1}\|_*, \tag{10}$$

where the term $\|\tilde{\mathbf{A}}_{t+1}\|_1$ imposes sparsity on the predicted matrix, while $\|\tilde{\mathbf{A}}_{t+1}\|_*$ is the nuclear norm that encourages a low-rank structure. The weights γ and δ control the importance of the sparsity and low-rank constraints, respectively.

By optimizing both loss functions, DyGMAE improves its ability to reconstruct the masked edges and capture the all nodes connectivity across several unmasked views. However, as different view representations are discrete in nature, there exists a challenge in effectively integrating them.

Alignment Module With Contrastive Learning. After obtaining representations from multiple unmasked graph views, $\{\mathbf{H}_t^1, \mathbf{H}_t^2, \ldots, \mathbf{H}_t^K\}$, we apply contrastive learning to align these latent representations, preserving complex features. This contrastive alignment enhances the consistency of the learned features and boosts the integration of local and global information across different views.

We adopt the following contrastive loss on two views to enforce alignment between representations of the same node across different unmasked views at time t:

$$\mathcal{L}_t^C = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp\left(\text{sim}(\mathbf{h}_i^1, \mathbf{h}_i^2)/\tau\right)}{\sum_{j=1}^{N}\exp\left(\text{sim}(\mathbf{h}_i^1, \mathbf{h}_j^2)/\tau\right)}, \tag{11}$$

where $\mathbf{h}_i^1$ and $\mathbf{h}_i^2$ stand for the latent representations corresponding to the same node i within two separate unmasked views, $\text{sim}(\cdot,\cdot)$ denotes a similarity function (e.g., cosine similarity), and τ is a temperature parameter that controls the sharpness of distribution.

Multi-Scale Masking Representation Fusion. Although individual representations $\{\mathbf{H}_t^1, \mathbf{H}_t^2, \ldots, \mathbf{H}_t^K\}$ generated from different unmasked views capture distinct structural and evolving features, they fail to fully represent the comprehensive characteristics of the graph. To address this, we introduce a Multi-Scale Masking Representation Fusion (MSMRF) to aggregate the latent representations derived from multiple unmasked views, combining information across different perspectives. This fusion process enhances the method's ability to learn multi-level features, including both global and fine-grained structural patterns and temporal dependencies. The fused representation of the t-th snapshot is computed as:

$$\mathbf{H}_t = \text{Aggregate}(\mathbf{H}_t^1, \mathbf{H}_t^2, \ldots, \mathbf{H}_t^K), \tag{12}$$

where $\text{Aggregate}(\cdot)$ denotes the aggregation function, which can be implemented through a variety of approaches. These include calculating the mean, performing summation, using the max operation, and applying attention mechanisms.

3.3 TEMPORAL DEPENDENCY MODELING WITH GRU

Capturing temporal dependencies is crucial for DLP, as the relationships between nodes evolve over time. In dynamic graphs, the probability of a node forming a link with another node at the current snapshot depends not only on the current graph structure but also on the temporal evolution of node interactions, which is influenced by previous snapshots. To capture these evolving dependencies, we employ a GRU [Cho et al., 2014], a model well-suited for sequential data processing, enabling the effective capture of temporal dependencies in the graph. Specifically, the current latent representation $\mathbf{H}_t$, generated by the encoder, and the final representation $\mathbf{Z}_{t-1}$ from the previous time step are combined as inputs to compute the final representation $\mathbf{Z}_t$ at time t. For the initial step ($t = 0$), $\mathbf{Z}_0$ is initialized randomly. The GRU updates are defined as follows:

$$\mathbf{Q}_t = \sigma_2(\mathbf{W}_q\mathbf{H}_t + \mathbf{U}_q\mathbf{Z}_{t-1}), \tag{13}$$

$$\mathbf{R}_t = \sigma_2(\mathbf{W}_r\mathbf{H}_t + \mathbf{U}_r\mathbf{Z}_{t-1}), \tag{14}$$

$$\hat{\mathbf{Z}}_t = \phi(\mathbf{W}_h\mathbf{H}_t + \mathbf{U}_h(\mathbf{R}_t \odot \mathbf{Z}_{t-1})), \tag{15}$$

$$\mathbf{Z}_t = (1 - \mathbf{Q}_t) \odot \mathbf{Z}_{t-1} + \mathbf{Q}_t \odot \hat{\mathbf{Z}}_t, \tag{16}$$

where $\mathbf{Q}_t$ and $\mathbf{R}_t$ denote the update gate and reset gate, respectively, and $\hat{\mathbf{Z}}_t$ is the candidate state. Here, $\sigma_2(\cdot)$ represents the sigmoid activation function, $\phi(\cdot)$ denotes the hyperbolic tangent function ($\tanh$), and $\odot$ is the element-wise product. The parameters $\mathbf{W}_q, \mathbf{W}_r, \mathbf{W}_h$ and $\mathbf{U}_q, \mathbf{U}_r, \mathbf{U}_h$ are learnable weight matrices.

3.4 DECODER FOR DLP

After obtaining the representations $\mathbf{Z}_t$ for each time step, these representations are used to predict the existence of edges in the dynamic graph at next time step $t + 1$ for the DLP task. We utilize dot-product as the decoder. The decoder operation is defined as follows:

$$\hat{\mathbf{A}}_{t+1} = f_{\text{Dec}}(\mathbf{Z}_t), \tag{17}$$

where $\hat{\mathbf{A}}_{t+1}$ is the predicted adjacency matrix for time step $t + 1$, and a higher value in $\hat{\mathbf{A}}_{t+1}$ indicates a higher probability of an edge between nodes.

3.5 FINAL OBJECTIVE

DyGMAE integrates three key components for its overall objective: edge reconstruction loss, adjacency matrix reconstruction loss and contrastive loss. The overall loss function for a single snapshot t is defined as:

$$\mathcal{L}_t = \mathcal{L}_t^E + \mathcal{L}_t^A + \mathcal{L}_t^C. \tag{18}$$

Extending this to the entire dynamic graph, the total loss is aggregated across all snapshots $t \in \{1, 2, \ldots, T\}$ and all masked views $k \in \{1, 2, \ldots, K\}$, and is expressed as:

$$\mathcal{L} = \sum_{t=1}^{T} \left(\sum_{k=1}^{K} \left(\mathcal{L}_t^{Ek} + \mathcal{L}_t^{Ak} \right) + \lambda \mathcal{L}_t^C \right), \tag{19}$$

where $\mathcal{L}_t^{Ek}$ represents the edge reconstruction loss and $\mathcal{L}_t^{Ak}$ is the adjacency matrix reconstruction loss at the k-th view of snapshot t. The weighting coefficient λ controls the relative importance of the contrastive loss.

By integrating these components across every snapshot and view, our method successfully captures the structural patterns and temporal dependencies in both local and global aspects. The full algorithm and complexity analysis of DyGMAE is presented in the Appendix A.

4 EXPERIMENTS

In this section, we conduct the experiments on five real-world dynamic graph datasets to verify the effectiveness of our proposed DyGMAE method.

4.1 EXPERIMENTAL SETTINGS

Datasets. We conduct experiments on five dynamic graph datasets, including Enron [Benson et al., 2018], DBLP, Facebook [Hajiramezanali et al., 2019], Email [Gao et al., 2023], and AS733 [Yang et al., 2021]. Detailed information about these datasets is provided in Appendix B.

Baselines. The baselines include static autoencoder methods such as GAE and VGAE [Kipf and Welling, 2016], as well as dynamic autoencoder methods like DynAE, DynRNN, DynAERNN [Goyal et al., 2020], and VGRNN [Hajiramezanali et al., 2019]. Additionally, we compare with advanced methods like EvolveGCN [Pareja et al., 2020], DGCN [Gao et al., 2023], DySAT [Sankar et al., 2020], HTGN [Yang et al., 2021], and HGWaveNet [Bai et al., 2023]. More details about baselines are provided in Appendix C.

Evaluation Tasks and Metrics. To evaluate the effectiveness of our method, we conduct experiments on two link prediction tasks: DLP and DNLP here. The evaluation measures the ability of the methods to distinguish true from false edges by calculating Average Precision (AP) and Area Under the Receiver Operating Characteristic Curve (AUC) scores. In this setup, all known edges in the test snapshots are treated as positive samples (true links), while an equal number of non-existent edges are sampled as negative samples (false links).

4.2 IMPLEMENTATION DETAILS

We conduct all experiments following the same settings as in [Hajiramezanali et al., 2019]. The training process uses the snapshots from the training set, while the performance of all methods is evaluated on the test snapshots for DLP and DNLP. During training, train snapshots are masked, while the test snapshots use the original unmasked graph data. For a fair comparison, each experiment is run five times with different random seeds to minimize the impact of randomness, and the results are reported as averages with standard deviations. The other implementation details can be seen in Appendix D.

4.3 EXPERIMENTAL RESULTS

The results of DLP and DNLP are presented in Table 1 and Table 2, respectively. For the test sets, both the average values and standard deviations are reported. Certain results are based on findings from previously published papers. We conduct experiments on multi-scale masking representation fusion type analysis as well as multi-step dynamic link prediction, which are presented in Appendix E and Appendix F, respectively.

Dynamic Link Prediction. Results of DLP are shown in Table 1. As we can see, DyGMAE consistently outperforms state-of-the-art methods in DLP, achieving higher AUC and AP scores across various real-world dynamic graph datasets. Specifically, in Facebook dataset, DyGMAE gets 3.88% and 4.41% improvement on AUC and AP compared with the second-best method VGRNN. Although HTGN and HTWaveNet achieve good performance in the hyperbolic space, they can only capture the local information, which limits their performance. Compared to DGCN, which can capture global information, we achieve significant improvements across all five datasets. The strong performance of DyGMAE indicates its superiority in capturing both local and global structural patterns and temporal dependencies. Compared with other autoencoders, including the static autoencoders like GAE and VGAE, as well as the dynamic autoencoders such as DynAE, DynRNN, DynAERNN, and VGRNN, our approach yields remarkable improvements. This outcome not only validates the feasibility of the GMAE framework in DLP task but also confirms the rationality of our initial inspiration.

Dynamic New Link Prediction. As shown in Table 2, in the task of DNLP, which demands that all methods

Table 1: AUC and AP scores of DLP in real-world dynamic graphs. Best results are in bold, and the suboptimal results are underlined.

Dataset	Enron		DBLP		Facebook		Email		AS733	
Metric	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP
GAE	92.55±0.76	93.64±0.51	84.71±0.73	87.78±0.42	89.47±0.65	88.93±0.80	81.34±0.18	89.37±0.12	93.81±1.12	93.56±1.04
VGAE	92.46±0.49	93.64±0.30	85.51±0.44	88.45±0.17	88.91±0.22	88.16±0.29	82.58±0.36	89.95±0.19	95.98±0.82	96.11±1.24
DynAE	74.22±0.74	76.00±0.77	63.14±1.30	64.02±1.08	56.06±0.29	56.04±0.37	83.04±1.55	83.61±1.37	75.57±1.38	74.31±1.22
DynRNN	86.41±1.36	85.61±1.46	75.70±1.09	78.95±1.55	73.18±0.60	75.88±0.42	86.80±1.62	86.13±1.35	87.43±1.02	88.98±0.86
DynAERNN	87.43±1.19	89.37±1.17	76.06±1.08	81.84±0.89	76.02±0.88	78.55±0.73	88.37±1.19	89.64±1.27	88.82±0.63	89.11±0.64
VGRNN	93.18±0.48	93.95±0.33	85.32±0.66	88.35±0.63	90.29±0.24	90.14±0.36	93.89±1.58	95.31±1.05	96.65±0.48	96.35±0.53
EvolveGCN	91.37±0.54	92.84±0.41	83.65±0.29	87.23±0.29	85.71±0.51	86.47±0.78	80.86±2.23	86.89±1.90	93.64±0.47	94.23±0.51
DGCN	83.36±0.62	80.31±0.83	75.06±1.14	73.56±1.08	71.77±1.15	71.92±2.21	93.77±0.69	92.47±0.76	92.55±0.38	92.04±0.29
DySAT	94.23±0.44	95.05±0.36	86.28±0.49	89.01±0.32	89.76±0.19	89.29±0.21	86.69±0.28	92.22±0.16	96.77±0.37	97.14±0.41
HTGN	95.25±0.17	95.63±0.28	89.35±0.34	91.86±0.28	87.26±0.56	87.49±0.29	94.86±0.27	95.84±0.45	98.96±0.05	98.75±0.06
HGWaveNet	95.36±0.23	94.85±0.36	89.88±0.17	92.58±0.25	88.72±0.33	87.55±0.52	92.66±0.37	93.96±0.31	99.12±0.06	99.15±0.11
DyGMAE	**96.89±0.15**	**96.91±0.15**	**90.77±0.34**	**92.65±0.22**	**94.17±0.10**	**94.55±0.09**	**95.82±0.33**	**96.75±0.15**	**99.53±0.06**	**99.64±0.04**

Table 2: AUC and AP scores of DNLP in real-world dynamic graphs. Best results are in bold, and the suboptimal results are underlined.

Dataset	Enron		DBLP		Facebook		Email		AS733	
Metric	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP
GAE	87.57±1.07	87.99±0.86	78.11±1.26	82.15±0.97	88.55±0.53	87.58±0.77	75.73±0.32	85.68±0.26	88.74±1.72	89.16±1.25
VGAE	87.30±0.82	87.66±0.73	78.81±1.05	82.98±0.50	88.64±0.18	87.59±0.24	77.35±0.57	86.50±0.32	89.36±1.64	89.93±1.67
DynAE	66.10±0.71	66.50±1.12	58.14±1.16	58.82±1.06	54.62±0.22	54.62±0.22	80.54±1.41	79.86±1.25	69.46±1.62	69.83±1.64
DynRNN	83.20±1.01	80.96±1.37	71.71±0.73	75.34±0.67	73.32±0.60	75.52±0.50	81.13±1.17	81.41±1.49	76.87±1.52	78.35±0.26
DynAERNN	83.77±1.65	85.16±1.04	71.99±1.04	77.68±0.66	76.35±0.50	78.70±0.44	84.33±1.55	86.54±1.60	77.64±1.27	77.73±1.32
VGRNN	87.77±1.02	87.98±1.59	75.25±0.92	79.28±0.68	87.61±0.35	86.77±0.49	92.77±1.23	94.04±0.80	82.14±2.35	89.37±2.93
EvolveGCN	84.79±0.68	85.82±0.53	73.68±0.62	78.04±0.54	82.21±0.83	82.12±0.63	74.50±3.10	81.99±2.15	77.45±0.91	83.66±0.86
DGCN	81.25±1.68	77.92±1.12	74.15±1.55	75.31±1.41	72.36±0.68	70.30±1.23	90.33±0.84	90.23±0.71	88.71±1.07	88.48±0.92
DySAT	89.70±0.58	89.52±0.78	79.23±0.84	82.83±0.67	88.84±0.17	87.67±0.14	82.30±0.54	89.26±0.29	85.51±0.77	88.72±0.81
HTGN	91.61±0.49	90.78±0.77	82.84±0.75	84.84±0.89	83.22±0.58	82.61±0.88	92.89±0.42	94.02±0.49	96.62±0.26	95.69±0.24
HGWaveNet	92.57±0.25	91.85±0.32	83.22±0.23	85.84±0.32	85.83±0.59	83.79±0.75	90.67±0.32	92.57±0.28	**96.85±0.19**	95.77±0.22
DyGMAE	**93.35±0.28**	**92.53±0.13**	**83.85±0.82**	**86.21±0.78**	**92.09±0.15**	**92.18±0.10**	**93.98±0.51**	**95.16±0.17**	94.43±0.15	**96.55±0.05**

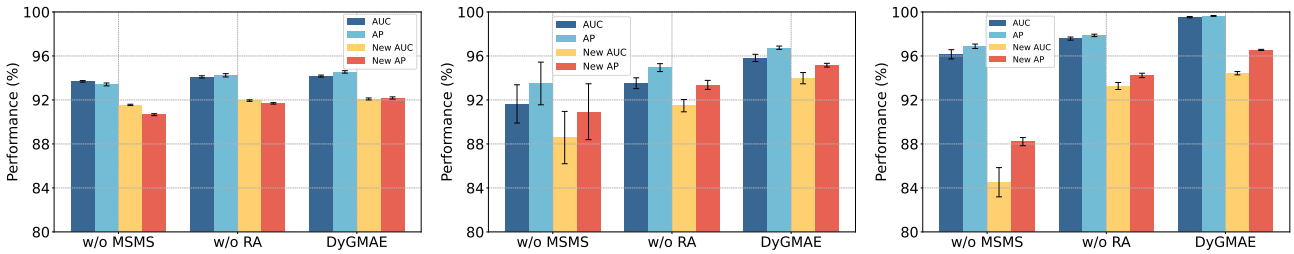
should have enhanced inductive capabilities and acquire more global information, DyGMAE consistently outperforms other methods. Especially, on the Facebook dataset, DyGMAE achieves a significant improvement of 3.25% in AUC and 4.51% in AP compared to the second-best method DySAT. This notable gap indicates that DyGMAE has a stronger ability to capture the potential patterns and changes in the social graph structure within this dataset. Moreover, when looking at other datasets, DyGMAE also demonstrates superior generalization. This outstanding performance is attributed to the simultaneous capture of local and global information, which enhances the ability to model changes and enables excellent generalization across diverse scenarios.

4.4 ABLATION STUDY

To better understand the contributions of different modules to the performance improvements of DyGMAE, we conduct an ablation study by removing key components. Specifically, we analyze two variants: **w/o MSMS:** This variant removes the MSMS and replaces it with a single masking strategy. **w/o RA:** In this variant, we eliminate the multi-scale masking representation alignment module, leaving only multi-scale masking without representation alignment in MSMS. We conduct experiments on three datasets: Facebook, Email, and AS733, with the results shown in Figure 2. Overall, DyGMAE achieves the best performance when all components are included, while removing either the MSMS module or the contrastive alignment leads to significant performance drops across all metrics.

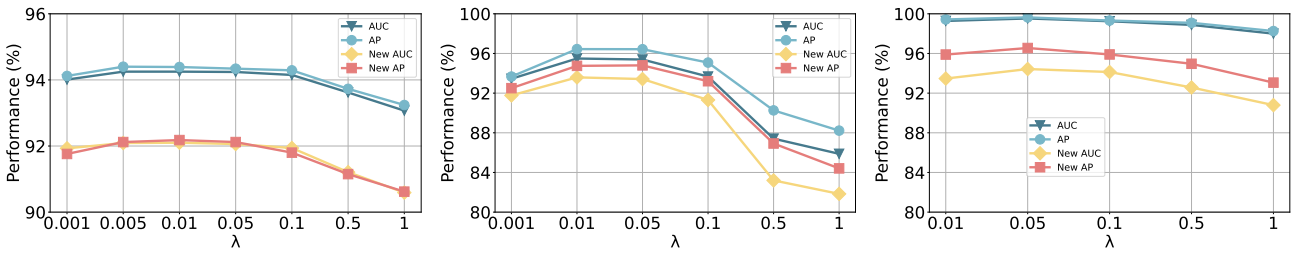
Regarding the impact of removing MSMS, in Facebook, removing MSMS results in a decrease of 0.48% in AUC, 1.41% in AP, 0.54% in New_AUC, and 1.52% in New_AP. Specifically, for Email with long snapshots where DyGMAE struggles to capture dynamic dependencies, MSMS helps alleviate this issue. However, removing MSMS leads to drops of 4.18% in AUC, 3.25% in AP, 5.4% in New_AUC, and 4.23% in New_AP. In AS733 for DNLP, significant improvement is due to MSMS, which brings increases of 3.38% in AUC, 2.76% in AP, 9.91% in New_AUC, and 8.34% in New_AP. Given the long datasets and test snapshots, MSMS is key for capturing long-term dependencies and ensuring good generalization. The ablation results further confirm the validity of our motivation that GMAE benefits DLP and our design related to MSMS.

Furthermore, removing the multi-scale masking representation alignment module (w/o RA) also degrades performance. In the Facebook, Email, and AS733 datasets, the drops in AUC are 0.07%, 2.30%, and 1.95% respectively; in AP, 0.31%, 1.81%, and 1.77%; in New_AUC, 0.16%, 2.50%,



(a) Facebook (b) Email (c) AS733

Figure 2: Ablation study results in Facebook, Email, and AS733 datasets. AUC and AP refer to DLP, while New_AUC and New_AP correspond to DNLP.



(a) Facebook (b) Email (c) AS733

Figure 3: Parameter study results in Facebook, Email, and AS733 datasets. AUC and AP refer to DLP, while New_AUC and New_AP correspond to DNLP.

and 1.16%; and in New_AP, 0.49%, 1.79%, and 2.32%. Interestingly, the performance of the w/o RA variant is better than the w/o MSMS variant in all three datasets, but still shows a clear gap when compared to the full DyGMAE. This demonstrates the effectiveness of the representation alignment in extracting more meaningful and comprehensive representations, especially in larger graph datasets where it is more challenging to acquire local and global information. Additionally, in all three datasets, DyGMAE exhibits smaller variance compared to the other two variants, validating its robustness.

Overall, these findings underline the crucial role of both multi-scale masking and contrastive learning in enhancing the discriminative power and generalization ability of the learned representations.

4.5 HYPER-PARAMETER SENSITIVITY ANALYSIS

We perform parameter analysis experiments on multiple datasets to examine the sensitivity of the hyperparameter λ, which controls the contribution of the contrastive loss objective for aligning the representations across different masked views in MSMS. The results, shown in Figure 3, indicate that the sensitivity of λ varies across different real-world dynamic graph datasets for both DLP and DNLP tasks.

When λ is too small, DyGMAE fails to align the representations effectively, causing the loss of complex features and comprehensive representations, which results in a performance decline in both tasks. As λ increases, the alignment between representations becomes more pronounced, leading to improved consistency in representations across different views. However, when λ exceeds a certain threshold, the method starts to over-align the representations, leading to the loss of important details and consequently decreasing performance. We observe that the best performance for both tasks is achieved under different values of λ. Regarding the changes in λ, there are different performance variation curves. Performance degrades when λ is either too small or too large, highlighting the sensitivity of the method to this hyperparameter.

4.6 EFFECTIVENESS OF RECONSTRUCTION OBJECTIVES

To evaluate the impact of the reconstruction-based training objectives in DyGMAE, we perform ablation studies by selectively removing the edge-level reconstruction loss ($\mathcal{L}_t^E$) and the adjacency-level reconstruction loss ($\mathcal{L}_t^A$). We assess model performance on both standard and new link prediction settings across three datasets: Enron, Facebook, and Email.

The results are summarized in the following: We observe that removing either reconstruction objective leads to a con-

Table 3: Ablation results showing the contribution of reconstruction losses ($\mathcal{L}_t^E$ and $\mathcal{L}_t^A$) to link prediction performance. "New AUC" and "New AP" denote metrics for new link prediction.

Dataset	Setting	AUC (%)	AP (%)	New AUC (%)	New AP (%)
Enron	Full Model	96.89 ± 0.15	96.91 ± 0.15	93.35 ± 0.28	92.53 ± 0.13
	w/o $\mathcal{L}_t^E$	96.20 ± 0.22	96.25 ± 0.13	92.49 ± 0.49	92.09 ± 0.35
	w/o $\mathcal{L}_t^A$	94.47 ± 0.65	94.49 ± 0.46	89.40 ± 1.18	89.17 ± 1.04
Facebook	Full Model	94.17 ± 0.10	94.55 ± 0.09	92.09 ± 0.15	92.18 ± 0.10
	w/o $\mathcal{L}_t^E$	93.57 ± 0.06	93.61 ± 0.11	91.48 ± 0.15	91.32 ± 0.14
	w/o $\mathcal{L}_t^A$	89.55 ± 0.05	88.66 ± 0.05	86.95 ± 0.25	85.65 ± 0.27
Email	Full Model	95.82 ± 0.33	96.75 ± 0.15	93.98 ± 0.51	95.16 ± 0.17
	w/o $\mathcal{L}_t^E$	94.82 ± 0.34	95.76 ± 0.15	92.78 ± 0.65	93.83 ± 0.26
	w/o $\mathcal{L}_t^A$	90.32 ± 1.05	92.97 ± 0.80	86.11 ± 1.41	89.58 ± 1.28

sistent performance drop. Specifically: Removing the edge-level loss ($\mathcal{L}_t^E$) causes moderate degradation, indicating that reconstructing individual edges is essential for capturing fine-grained local patterns. Removing the adjacency-level loss ($\mathcal{L}_t^A$) leads to a more pronounced performance decline, especially on New Link Prediction metrics. This suggests adjacency-level modeling provides richer context and structural awareness, which is particularly important for discovering new interactions. The performance gap is most evident on the Facebook and Email datasets, where missing $\mathcal{L}_t^A$ leads to drops of up to 5%–7% AUC/AP in the new link prediction task. These findings reinforce that both $\mathcal{L}_t^E$ and $\mathcal{L}_t^A$ are crucial and complementary. Together with contrastive alignment, they enable DyGMAE to effectively model both local connectivity and global topology evolution, thereby significantly enhancing its generalization in dynamic link prediction.

5 CONCLUSION

In this study, we propose DyGMAE, a novel dynamic graph masked autoencoder, specifically designed for dynamic link prediction. DyGMAE extends the GMAE framework to dynamic graphs and fully leverages the advantages of GMAE in link prediction. In DyGMAE, we propose a Multi-Scale Masking Strategy to learn structural and dynamic features. This strategy involves applying different masking strategies to the graph and attempting to reconstruct it. Within MSMS, a contrastive learning objective is incorporated to further align representations across masked views and enable the model to capture complex features. As a result, MSMS can effectively mitigate the information loss caused by random masking and capture the comprehensive representation of local and global information in dynamic graphs. Extensive experiments conducted on five real-world dynamic graph datasets demonstrate the superior performance of DyGMAE and validate the effectiveness of our proposed design. However, our method requires careful tuning of hyperparameters. Our future work plans to focus on adapting DyGMAE for other dynamic graph tasks and extending the GMAE framework to continuous-time dynamic graphs.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (62477016, 62377028, 62077045) and Guangdong Basic and Applied Basic Research Foundation (2024A1515011758, 2023B1515120064), Guangzhou Science and Technology Planning Project (202206030007, Nansha District: 2023ZD001, Development District: 2023GH01).

References

Qijie Bai, Changli Nie, Haiwei Zhang, Dongming Zhao, and Xiaojie Yuan. Hgwavenet: A hyperbolic graph neural network for temporal link prediction. In *Proceedings of the ACM Web Conference 2023*, pages 523–532, 2023.

Austin R Benson, Rediet Abebe, Michael T Schaub, Ali Jadbabaie, and Jon Kleinberg. Simplicial closure and higher-order link prediction. *Proceedings of the National Academy of Sciences*, 115(48):E11221–E11230, 2018.

Kamal Berahmand, Elahe Nasiri, Saman Forouzandeh, and Yuefeng Li. A preference random walk algorithm for link prediction through mutual influence nodes in complex networks. *Journal of King Saud University Computer and Information Sciences*, 34(8):5375–5387, 2022.

Ke-Jia Chen, Linsong Liu, Linpu Jiang, and Jingqiang Chen. Self-supervised dynamic graph representation learning via temporal subgraph contrast. *ACM Transactions on Knowledge Discovery from Data*, 18(1):1–20, 2023.

Junwei Cheng, Chaobo He, Pengxing Feng, Weixiong Liu, Kunlin Han, and Yong Tang. Community-aware variational autoencoder for continuous dynamic networks. In

Proceedings of the AAAI Conference on Artificial Intelligence, volume 39, pages 16001–16009, 2025.

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.

Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Narjes Firouzkouhi, Abbas Amini, Ahmed Bani-Mustafa, Arash Mehdizadeh, Sadeq Damrah, Ahmad Gholami, Chun Cheng, and Bijan Davvaz. Generalized fuzzy hypergraph for link prediction and identification of influencers in dynamic social media networks. *Expert Systems with Applications*, 238:121736, 2024.

Chao Gao, Junyou Zhu, Fan Zhang, Zhen Wang, and Xuelong Li. A novel representation learning for dynamic graphs based on graph convolutional networks. *IEEE Transactions on Cybernetics*, 53(6):3599–3612, 2023.

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

Alessio Gravina and Davide Bacciu. Deep learning for dynamic graphs: models and benchmarks. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

Ehsan Hajiramezanali, Arman Hasanzadeh, Krishna Narayanan, Nick Duffield, Mingyuan Zhou, and Xiaoning Qian. Variational graph recurrent neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

Chaobo He, Junwei Cheng, Xiang Fei, Yu Weng, Yulong Zheng, and Yong Tang. Community preserving adaptive graph convolutional networks for link prediction in attributed networks. *Knowledge-Based Systems*, 272: 110589, 2023.

Chaobo He, Hao Cheng, Jiaqi Yang, Yong Tang, and Quanlong Guan. Signed graph embedding via multi-order neighborhood feature fusion and contrastive learning. *Neural Networks*, 182:106897, 2025.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

Zhenyu Hou, Xiao Liu, Yukuo Cen, Yuxiao Dong, Hongxia Yang, Chunjie Wang, and Jie Tang. Graphmae: Self-supervised masked graph autoencoders. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 594–604, 2022.

Zhenyu Hou, Yufei He, Yukuo Cen, Xiao Liu, Yuxiao Dong, Evgeny Kharlamov, and Jie Tang. Graphmae2: A decoding-enhanced masked self-supervised graph learner. In *Proceedings of the ACM web conference 2023*, pages 737–746, 2023.

Pengfei Jiao, Xinxun Zhang, Zehao Liu, Long Zhang, Huaming Wu, Mengzhou Gao, Tianpeng Li, and Jian Wu. A deep contrastive framework for unsupervised temporal link prediction in dynamic networks. *Information Sciences*, 667:120499, 2024.

Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

Fuxian Li, Jie Feng, Huan Yan, Guangyin Jin, Fan Yang, Funing Sun, Depeng Jin, and Yong Li. Dynamic graph convolutional recurrent network for traffic prediction: Benchmark and solution. *ACM Transactions on Knowledge Discovery from Data*, 17(1):1–21, 2023a.

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 *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1268–1279, 2023b.

Rafael Lima. Hawkes processes modeling, inference, and control: An overview. *SIAM Review*, 65(2):331–374, 2023.

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*, 2024a.

Chuang Liu, Zelin Yao, Yibing Zhan, Xueqi Ma, Dapeng Tao, Jia Wu, Wenbin Hu, Shirui Pan, and Bo Du. Higmae: Hierarchical graph masked autoencoders. *arXiv preprint arXiv:2405.10642*, 2024b.

Linhao Luo, Gholamreza Haffari, and Shirui Pan. Graph sequential neural ode process for link prediction on dynamic and sparse graphs. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pages 778–786, 2023.

Yanchen Luo, Sihang Li, Yongduo Sui, Junkang Wu, Jiancan Wu, and Xiang Wang. Masked graph modeling with multi-view contrast. ICDE, 2024.

Shengjie Min, Zhan Gao, Jing Peng, Liang Wang, Ke Qin, and Bo Fang. Stgsn—a spatial–temporal graph neural network framework for time-evolving social networks. *Knowledge-Based Systems*, 214:106746, 2021.

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 *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5363–5370, 2020.

Phu Pham, Loan TT Nguyen, Ngoc Thanh Nguyen, Witold Pedrycz, Unil Yun, and Bay Vo. Comgcn: Community-driven graph convolutional network for link prediction in dynamic networks. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 52(9):5481–5493, 2021.

Meng Qin and Dit-Yan Yeung. Temporal link prediction: A unified framework, taxonomy, and review. *ACM Computing Surveys*, 56(4):1–40, 2023.

Liang Qu, Huaisheng Zhu, Qiqi Duan, and Yuhui Shi. Continuous-time link prediction via temporal dependent graph neural network. In *Proceedings of the Web Conference 2020*, pages 3026–3032, 2020.

Emile Richard, Pierre-André Savalle, and Nicolas Vayatis. Estimation of simultaneously sparse and low rank matrices. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, page 51–58, Madison, WI, USA, 2012. Omnipress. ISBN 9781450312851.

Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao Yang. Dysat: Deep neural representation learning on dynamic graphs via self-attention networks. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, WSDM '20, page 519–527, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450368223.

Sheng Tian, Ruofan Wu, Leilei Shi, Liang Zhu, and Tao Xiong. Self-supervised representation learning on dynamic graphs. In *Proceedings of the 30th ACM International Conference on nformation and Knowledge Management*, pages 1814–1823, 2021.

Yijun Tian, Kaiwen Dong, Chunhui Zhang, Chuxu Zhang, and Nitesh V Chawla. Heterogeneous graph masked autoencoders. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 9997–10005, 2023.

Liang Wang, Xiang Tao, Qiang Liu, and Shu Wu. Rethinking graph masked autoencoders through alignment and uniformity. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 15528–15536, 2024.

Xiangyu Wang, Yuan Li, Taiyu Ban, Jiarun Zhu, Lyuzhou Chen, Muhammad Usman, Xin Wang, Huanhuan Chen, Xiaomei Chen, Cyril Leung, et al. Dynamic link prediction for discovery of new impactful covid-19 research approaches. *IEEE Journal of Biomedical and Health Informatics*, 26(12):5883–5894, 2022.

Liping Yang, Xin Jiang, Yiming Ji, Hua Wang, Ajith Abraham, and Hongbo Liu. Gated graph convolutional network based on spatio-temporal semi-variogram for link prediction in dynamic complex network. *Neurocomputing*, 505:289–303, 2022. ISSN 0925-2312.

Menglin Yang, Min Zhou, Marcus Kalander, Zengfeng Huang, and Irwin King. Discrete-time temporal network embedding via implicit hierarchical learning in hyperbolic space. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1975–1985, 2021.

Yaowen Ye, Lianghao Xia, and Chao Huang. Graph masked autoencoder for sequential recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 321–330, 2023.

Byeonghyeop Yu, Yongjin Lee, and Keemin Sohn. Forecasting road traffic speeds by considering area-wide spatio-temporal dependencies based on a graph convolutional neural network (gcn). *Transportation Research Part C: Emerging Technologies*, 114:189–204, 2020.

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 *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3309–3320, 2023.

Yonghua Zhu, Junbo Ma, Changan Yuan, and Xiaofeng Zhu. Interpretable learning based dynamic graph convolutional networks for alzheimer's disease analysis. *Information Fusion*, 77:53–61, 2022.

Yuan Zuo, Guannan Liu, Hao Lin, Jia Guo, Xiaoqian Hu, and Junjie Wu. Embedding temporal network via neighborhood formation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2857–2866, 2018.

DyGMAE: A Novel Dynamic Graph Masked Autoencoder for Link Prediction (Supplementary Material)

Weixiong Liu[1] **Junwei Cheng**[1] **Zhongyu Pan**[1] **Chaobo He**[*1] **Quanlong Guan**[2]

[1]School of Computer Science, South China Normal University, Guangzhou, China
[2]College of Information Science and Technology, Jinan University, Guangzhou, China

A ALGORITHM AND COMPLEXITY

In this section, we will present a detailed account of the algorithm underlying our proposed DyGMAE method and conduct a comprehensive complexity analysis.

A.1 ALGORITHM

We summarize the details of our proposed method DyGMAE in the Algorithm 1. We input the dynamic graph snapshots $\mathcal{G}$, and the parameters. During the training of snapshots, for each individual snapshot, we generate K masks in MSMS. For each masked view, the encoder generates representations. Subsequently, we calculate the edge reconstruction loss and the adjacency matrix reconstruction loss using Equations 8 and 10 respectively. For each snapshot, we also need to compute the contrastive loss according to Equation 11. Then, we aggregate the representations from all views and pass them through the GRU. After that, we calculate the overall loss to update the parameters. Finally, we output the predicted adjacency matrix for the next time step.

A.2 COMPLEXITY ANALYSIS

We conduct a time-complexity analysis of DyGMAE. First, generating masked views for each snapshot t and each masking strategy K requires $O(T \cdot K \cdot M)$, where T is the number of snapshots, K is the number of masking strategies, and M is the number of edges. Next, computing the latent representations $\mathbf{H}_t^k$ using the encoder and projector involves a complexity of $O(T \cdot K \cdot (L \cdot (N + M) \cdot d^2 + N \cdot d^2))$, where L is the number of GNN layers, N is the number of nodes, and d is the feature dimension. The all reconstruction losses $\mathcal{L}_t^{Ek}$ and $\mathcal{L}_t^{Ak}$ are computed with a complexity of $O(T \cdot K \cdot N \cdot d)$. For the contrastive loss $\mathcal{L}_t^C$, aligning multi-mask representations incurs a complexity of $O(T \cdot K^2 \cdot N \cdot d)$. Aggregating representations from all masked views to obtain $\mathbf{H}_t$ requires $O(T \cdot K \cdot N \cdot d)$. Finally, updating temporal dependencies using GRU has a complexity of $O(T \cdot N \cdot d^2)$. Overall, the dominant factors in the time complexity are the number of snapshots T, the graph size N and M, and the feature dimension d, leading to an approximate complexity of $O(T \cdot (N + M) \cdot d^2)$ when K and L are small constants.

B DATASETS

We conduct experiments on five datasets, which vary in size and the length of their snapshots. Some datasets are large while others are small, and some have long-term snapshots whereas others have short-term ones. Table 4 summarizes the key statistics of the datasets used in our experiments, providing an overview of their size and structural properties. The notation

[*]Corresponding author: hechaobo@foxmail.com
[*]Corresponding author: hechaobo@foxmail.com

Algorithm 1: Dynamic Graph Masked Autoencoder (DyGMAE)

Input: Dynamic graph snapshots $\mathcal{G} = \{G_1, G_2, \ldots, G_T\}$; Number of masks K; Set of masking strategies $\{f_{\mathrm{M}}^1, f_{\mathrm{M}}^2, \ldots, f_{\mathrm{M}}^K\}$; Weight coefficients λ; Maximum iteration $epoch_max$

Output: Predicted adjacency matrix $\hat{\mathbf{A}}_{T+1}$

```
while epoch ≤ epoch_max do
    for each snapshot t ∈ {1, 2, ..., T} do
        for each view k ∈ {1, 2, ..., K} do
            Generate unmasked view G_t^{uk} using f_M^k;
            Compute latent representation H_t^k using the encoder and projector (Equations 4 and 5);
            Compute reconstruction losses L_t^{Ek} and L_t^{Ak} (Equations 8 and 10);
        end
        Compute contrastive loss L_t^C to align multi-scale masking representations (Equation 11);
        Aggregate representations from all masked views to obtain H_t (Equation 12);
        Update temporal dependencies using GRU to compute Z_t (Equation 13);
    end
    Compute total loss L (Equation 19);
    Update parameters using gradient descent;
end
return Predicted adjacency matrix Â_{T+1};
```

Table 4: Statistics of datasets.

Datasets	Enron	DBLP	Facebook	Email	AS733
#Nodes	184	315	663	2029	6,628
#Edges	115-266	165-308	844-1068	104-711	1734-12572
#Snapshots	11	10	9	29	30
Train : Test	8:3	7:3	6:3	26:3	20:10

"Train : Test" represents the number of time steps in the training snapshots and test snapshots, respectively. The datasets include: Enron[1], an email communication dataset where edges represent emails exchanged between employees; DBLP[2], an academic co-authorship network where nodes represent authors and edges represent collaborations; Facebook, a social communication network; Email[3], an email communication dataset. AS733[4], a graph representing autonomous systems (AS) of routers and their traffic exchanges via the Border Gateway Protocol;

C BASELINES

- GAE: GAE consists of an encoder and a decoder. The encoder learns to map the input graph data into a low-dimensional latent space, while the decoder reconstructs the graph data from the latent vectors.
- VGAE [Kipf and Welling, 2016]: The VGAE combines the concepts of the VAE and the GCN, and it consists of an encoder and a decoder. The encoder uses the GCN to map the node features of a graph to a latent space and outputs the mean and variance of the latent variables to represent the distribution of the node features.
- DynAE [Goyal et al., 2020]: It is a dynamic autoencoder method extended from the static graph autoencoder. It processes the adjacency matrix information of multiple time steps through fully connected layers. It is suitable for capturing short-term dynamic patterns, but has limited ability to model long-term dependencies.
- DynRNN [Goyal et al., 2020]: It inputs the sequence of node adjacency vectors into the LSTM (Long Short-Term Memory) units and learns the evolutionary patterns across time steps through the memory gating mechanism. The LSTM also serves as the decoder.

[1]https://www.cs.cornell.edu/ arb/data/email-Enron/
[2]https://github.com/VGraphRNN/VGRNN
[3]http://networkrepository.com/dynamic.php
[4]https://snap.stanford.edu/data/as-733.html

- DynAERNN [Goyal et al., 2020]: DynAERNN uses a fully connected encoder to reduce the dimension of the adjacency matrix into a low-dimensional representation. Then, it inputs the low-dimensional sequence into the LSTM for temporal modeling. The decoder used is a fully connected layer.
- VGRNN [Hajiramezanali et al., 2019]: It is a novel hierarchical variational model for representation learning on dynamic graphs. It extends the graph convolutional recurrent neural network (GCRN) to form a graph recurrent neural network (GRNN), then enhances it by introducing high-level latent random variables to create the variational graph recurrent neural network (VGRNN).
- EvolveGCN [Pareja et al., 2020]: EvolveGCN offers a solution for dynamic graphs by adapting GCN over time without relying on node embeddings. It uses an RNN to dynamically update GCN parameters, with two architectures explored for this process.
- DGCN [Gao et al., 2023]: DGCN is a GCN-based dynamic graph representation learning method. It maximizes the mutual information between local node and global graph representations to capture snapshot-level global structure and uses LSTM to update GCN weight parameters across time steps. A new Dice similarity is proposed to guide the aggregation and better distinguish the importance of neighboring nodes.
- DySAT [Sankar et al., 2020]: It employs two self-attention mechanisms, the structural attention block and the temporal attention block, to capture information from two dimensions: structural neighborhood and historical representations.
- HTGN [Yang et al., 2021]: HTGN migrates the space to the hyperbolic geometry space. It utilizes Hypergraph Neural Network (HGNN) and Hypergraph Gated Recurrent Unit (HGRU) to obtain topological and dynamic information. Moreover, it introduces the Hyperbolic Temporal Contextual Self-Attention (HTA) module to focus on historical states and the Hyperbolic Temporal Consistency (HTC) module to ensure stability and generalization ability.
- HGWaveNet [Bai et al., 2023]: HGWaveNet uses hyperbolic diffusion graph convolution (HDGC) to aggregate neighborhood information and hyperbolic dilated causal convolution (HDCC) to obtain historical state information.

D IMPLEMENTATION DETAILS

DyGMAE is implemented in Python 3.9, PyTorch 1.11 and executed on Intel Core i5-12490F CPUs and NVIDIA RTX 3070 GPUs. The detailed hyperparameters of our model for each dataset are presented in Table 5.

Table 5: Hyperparameters settings.

Hyperparameters	**Enron**	**DBLP**	**Facebook**	**Email**	**AS733**
Learning Rate	1e-3	1e-3	1e-3	5e-3	1e-2
K	2	2	3	2	2
λ	0.05	0.5	0.01	0.01	0.05
Masking Strategies	Path, Edge	Edge, Edge	Path, Path, Path	Path, Edge	Path, Edge
Masking Rate p	0.5, 0.7	0.1, 0.2	0.4, 0.6, 0.8	0.1, 0.3	0.6, 0.7
Encoder	GCN	GCN	GCN	GCN	GCN
Encoder Layers L	2	2	2	2	2

E MULTI-SCALE MASKING REPRESENTATION FUSION TYPE ANALYSIS

In this study, we performed a comprehensive analysis of diverse fusion types within the Multi-Scale Masking Representation Fusion (MSMRF) introduced in Section 3.2. To gauge the efficacy of MSMRF, we conducted experiments employing four distinct fusion strategies: mean, sum, max, and an attention-based mechanism. The outcomes of the DLP and DNLP tasks are respectively presented in Table 6 and Table 7.

The experimental results indicate that different datasets require different strategies to achieve optimal performance, and there are significant differences in the performance of different strategies on various datasets. The max strategy demonstrates good performance across multiple datasets and tasks. However, the performance of each strategy varies significantly among different datasets. This suggests that in practical applications, we need to select appropriate fusion strategies according to the characteristics of the datasets to improve the accuracy and reliability of predictions.

Table 6: AUC and AP scores of DLP for various fusion methods. Best results are in bold.

Dataset	Enron		Facebook		Email	
Metric	AUC	AP	AUC	AP	AUC	AP
Mean	96.55±0.23	96.70±0.16	**94.25±0.09**	94.39±0.11	95.79±0.36	96.70±0.17
Sum	96.14±0.16	96.44±0.06	94.17±0.10	**94.55±0.09**	**95.82±0.33**	**96.75±0.15**
Max	**96.89±0.15**	**96.91±0.15**	93.25±0.21	93.56±0.18	95.67±0.36	96.60±0.15
Attention	96.25±0.08	96.57±0.06	93.43±0.10	93.41±0.12	95.01±1.18	96.17±0.79

Table 7: AUC and AP scores of DNLP for various fusion methods. Best results are in bold.

Dataset	Enron		Facebook		Email	
Metric	AUC	AP	AUC	AP	AUC	AP
Mean	92.38±0.66	92.00±0.57	**92.10±0.09**	91.93±0.10	93.88±0.56	95.04±0.19
Sum	91.84±0.41	91.92±0.42	92.09±0.15	**92.18±0.10**	**93.98±0.51**	**95.16±0.17**
Max	**93.35±0.28**	**92.53±0.13**	90.81±0.23	90.84±0.13	93.67±0.52	94.86±0.17
Attention	91.70±0.32	91.63±0.38	91.01±0.16	90.65±0.13	92.72±1.68	94.26±1.16

F MULTI-STEP DYNAMIC LINK PREDICTION RESULTS

Multi-step dynamic link prediction aims to train a projection function f. Given a sequence of dynamic graph adjacency matrices of length l, denoted as $\{\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_l\}$, this function is trained for prediction. This function maps the input snapshot sequence $\{\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_l\}$ to the future adjacency matrices $\mathbf{A}_{l+1}, \mathbf{A}_{l+2}, \cdots, \mathbf{A}_{l+k}$, i.e., $\mathbf{A}_{l+1}, \mathbf{A}_{l+2}, \cdots, \mathbf{A}_{l+k} = f(\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_l)$. It must be pointed out that the multi-step dynamic link prediction has no access to the entire set of test snapshots. In contrast, the dynamic link prediction can see the snapshot at the previous time step of the snapshot to be predicted, but the model parameters cannot be updated through training on the test set. Compared with DLP, multi-step dynamic link prediction requires methods to be more effective in capturing the dynamic evolution information of graph structures. We conducted multi-step dynamic link prediction and also multi-step dynamic new link prediction experiments on three datasets and compared with several state-of-the-art methods. The experimental settings were the same as those in the DLP experiments. The results are presented in Table 8 and Table 9 respectively. As can be seen, our method achieves the best performance on these two tasks, demonstrating that DyGMAE can better capture the dynamic dependencies.

Table 8: AUC and AP scores of multi-step dynamic link prediction in real-world dynamic graphs. Best results are in bold.

Dataset	Enron		Facebook		Email	
Metric	AUC	AP	AUC	AP	AUC	AP
VGRNN	91.91±0.32	92.15±0.58	89.32±0.51	89.59±0.49	92.14±0.85	92.73±1.32
HTGN	94.38±0.29	94.91±0.26	86.26±0.58	86.60±0.47	93.71±0.27	94.78±0.19
HGWaveNet	95.02±0.25	94.54±0.18	88.01±0.44	87.17±0.19	92.34±0.26	93.84±0.29
DyGMAE	**95.97±0.12**	**96.20±0.13**	**93.49±0.09**	**94.32±0.17**	**95.30±0.21**	**96.37±0.11**

Table 9: AUC and AP scores of multi-step dynamic new link prediction in real-world dynamic graphs. Best results are in bold.

Dataset	Enron		Facebook		Email	
Metric	AUC	AP	AUC	AP	AUC	AP
VGRNN	86.78±0.40	86.81±0.49	86.70±0.43	86.21±0.38	90.75±0.23	91.14±0.34
HTGN	90.01±0.34	89.53±0.33	82.96±0.58	82.66±0.48	91.89±0.33	93.26±0.12
HGWaveNet	92.16±0.46	91.74±0.35	85.08±0.56	83.11±0.32	90.23±0.31	92.15±0.42
DyGMAE	**92.34±0.24**	**92.15±0.32**	**91.60±0.13**	**92.09±0.11**	**93.50±0.33**	**94.88±0.13**

G RELATED WORK

In this section, we systematically review the relevant works regarding DLP and provide a brief introduction to the graph masked autoencoder.

G.1 DYNAMIC LINK PREDICTION

The field of DLP has seen significant progress and attention recently and many related methods have been proposed to address this challenge. For example, EvolveGCN [Pareja et al., 2020] uses RNNs to update graph convolution network (GCN) weight parameters dynamically at each time step, modeling temporal changes in graph sequences. Building on this, ComGCN [Pham et al., 2021] captures both node-level and community-level structural and evolutionary dynamics to improve the DLP performance. HTGN [Yang et al., 2021] and HGWaveNet [Bai et al., 2023] extend GCNs to hyperbolic space, better capturing the structural and temporal dependencies. HTNE [Zuo et al., 2018] models the DLP task as the neighborhood formation sequences with multivariate Hawkes process [Lima, 2023], and infers the current neighbor formation events. TDGNN [Qu et al., 2020] devised a novel temporal aggregator to incorporate dynamic information into the message propagation process of GNN, and developed an effective static GNN extension TDGNN for DLP. Following the idea of TDGNN, GSNOP [Luo et al., 2023] integrates the neural ordinary differential equations, and it not only obtains better DLP performance but also effectively alleviates the sparsity issue of dynamic graphs. SSL offers a powerful approach to leverage the abundant unlabeled data available in dynamic graphs. Among SSL-based methods for dynamic graph learning, contrastive approaches dominate. DDGCL [Tian et al., 2021] is the first self-supervised framework for dynamic graphs, extending contrastive learning by contrasting temporal views of the same node identity. Similarly, DySubC [Chen et al., 2023] uses temporal subgraph contrastive learning to capture both structural and dynamic features while maximizing mutual information. Other SSL generative methods, such as VGRNN [Hajiramezanali et al., 2019], combine variational autoencoders with RNNs to model time-evolving node representations, using probabilistic inference to capture temporal dependencies and model uncertainty. And [Goyal et al., 2020] proposed three autoencoder methods to deal with DLP.

G.2 GRAPH MASKED AUTOENCODERS

Graph masked autoencoders have attracted significant attention in graph learning for their ability to leverage self-supervised signals through masking and reconstruction, enabling models to learn meaningful representations without requiring labeled data. GraphMAE [Hou et al., 2022], the first GMAE-based method, focuses on masking and reconstructing node features, achieving notable performance improvements in node classification task. Building upon GraphMAE, to enhance the robustness, GraphMAE2 [Hou et al., 2023] introduces the multi-view random re-mask decoding and latent representation prediction strategies. MaskGAE [Li et al., 2023b] extends this by corrupting edges and paths, reconstructing edge and degree information to capture structural features. StructMAE [Liu et al., 2024a] refines the masking strategy by introducing a structure-guided approach, where nodes are scored based on structural significance and an easy-to-hard masking process gradually enhances structural awareness. Additionally, AUG-MAE [Wang et al., 2024] introduces adversarial masking and a uniformity regularizer to improve alignment and representation consistency. Other methods [Tian et al., 2023, Ye et al., 2023, Liu et al., 2024b, Luo et al., 2024] integrate GMAE with contrastive learning, heterogeneous graphs, and sequential recommendation tasks. However, these methods are designed for static graphs and cannot address both the structural patterns and temporal dependencies of dynamic graphs.