Self-supervised Masked Graph Autoencoder via Structure-aware Curriculum

Haoyang Li [1] **Xin Wang** [1] **Zeyang Zhang** [1] **Zongyuan Wu** [1] **Linxin Xiao** [1] **Wenwu Zhu** [1]

Abstract

Self-supervised learning (SSL) on graph-structured data has attracted considerable attention recently. Masked graph autoencoder, as one promising generative graph SSL approach that aims to recover masked parts of the input graph data, has shown great success in various downstream graph tasks. However, existing masked graph autoencoders fail to consider the degree of difficulty of recovering the masked edges that often have different impacts on the model performance, resulting in suboptimal node representations. To tackle this challenge, in this paper, we propose a novel curriculum based self-supervised masked graph autoencoder that is able to capture and leverage the underlying degree of difficulty of data dependencies hidden in edges, and design better mask-reconstruction pretext tasks for learning informative node representations. Specifically, we first design a difficulty measurer to identify the underlying structural degree of difficulty of edges during the masking step. Then, we adopt a self-paced scheduler to determine the order of masking edges, which encourages the graph encoder to learn from easy to difficult parts. Finally, the masked edges are gradually incorporated into the reconstruction pretext task, leading to high-quality node representations. Experiments on several real-world node classification and link prediction datasets demonstrate the superiority of our proposed method over state-of-the-art graph self-supervised learning baselines. This work is the first study of curriculum strategy for masked graph autoencoders, to the best of our knowledge.

[1]Department of Computer Science and Technology, BNRist, Tsinghua University, Beijing, China. Correspondence to: Xin Wang <xin_wang@tsinghua.edu.cn>, Wenwu Zhu <wwzhu@tsinghua.edu.cn>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

1. Introduction

Graph-structured data is ubiquitous across various domains, including social networks, citation networks, and e-commerce systems. Graph neural networks (GNNs) have demonstrated significant success in learning meaningful representations from such data, particularly in supervised (Xu et al., 2019) and semi-supervised (Kipf & Welling, 2017; Hamilton et al., 2017) learning settings, where task-specific labels are available to guide the learning process. However, acquiring a large amount of high-quality annotations is often expensive and impractical in real-world applications.

Self-supervised learning (SSL), an unsupervised paradigm widely adopted in computer vision and natural language processing (Chen et al., 2020; He et al., 2020), has recently gained significant attention in the field of graph learning. SSL enables models to learn informative representations by solving carefully designed pretext tasks without requiring labeled data. Existing graph SSL methods can be broadly categorized into contrastive and generative approaches. Contrastive methods, such as DGI (Veličković et al., 2019), MVGRL (Hassani & Khasahmadi, 2020), and BGRL (Thakoor et al., 2022), predominate the field by leveraging instance discrimination as the primary pretext task. Although augmentation-free variants like AFGRL (Wang et al., 2022a) and IGCL (Li et al., 2023a) present promising alternatives, many classical contrastive methods still depend heavily on heuristic graph augmentations. Their performance can degrade when the selected augmentations are misaligned with the objectives of downstream tasks (Zhang et al., 2021a). In contrast, generative SSL methods address this limitation more naturally by reconstructing missing components of the input graph. Representative models such as GPT-GNN (Hu et al., 2020b), GraphMAE (Hou et al., 2022), and S2GAE (Tan et al., 2023) demonstrate strong performance while avoiding the need for manually crafted augmentations.

Despite the notable progress of generative graph SSL methods, existing approaches typically ignore the varying difficulty levels of pretext tasks during training and treat all training samples uniformly, resulting in suboptimal performance. Intuitively, pretext tasks should be designed to start with easier data samples and gradually progress to more difficult ones. Introducing overly challenging tasks at the early

stages of training can overwhelm the GNN encoder, which is typically initialized with random parameters and lacks the capacity to handle complex reconstructions. Conversely, prolonged training on overly simplistic tasks yields diminishing benefits and fails to further improve representation quality. The design of tailored easy-to-hard pretext tasks for enhancing graph representation learning remains an underexplored direction, which poses the following challenges.

- It is technically difficult to design tailored reconstruction tasks to encourage the GNNs to capture informative patterns of the input graph into representations.
- It is challenging to derive a proper principle to quantify the difficulty of reconstruction samples for training the GNNs.
- It is also non-trivial to design a feasible scheduling strategy to gradually exploit data samples for reconstruction by explicitly considering the current training status of GNNs.

To tackle these challenges, we propose Curriculum Masked Graph AutoEncoder (**Cur-MGAE**), a novel framework designed to capture and leverage the inherent difficulty of structural dependencies in graph edges. **Cur-MGAE** aims to construct more effective mask-reconstruction pretext tasks by integrating a curriculum learning paradigm into generative graph self-supervised learning. Specifically, our method enables GNNs to learn informative representations by gradually incorporating training samples in a tailored easy-to-hard order. We first design a structure-aware edge reconstruction task, where the goal is to recover intentionally masked edges based on the remaining unmasked graph structure. This pretext task encourages the GNN encoder to extract meaningful patterns from the graph. To quantify task difficulty, we introduce a self-supervised mechanism that identifies the easiest $\mathcal{K}$ edges the encoder is most confident in reconstructing. This allows for a principled estimation of reconstruction difficulty across edge samples. Furthermore, we develop a self-paced learning strategy that dynamically selects edges to be used in training, progressively increasing the task difficulty in alignment with the encoder's evolving learning capacity. The processes of edge selection and structural reconstruction are integrated into a unified training framework. Ultimately, the GNN encoder is trained using a meaningful curriculum that aligns edge difficulty with model capacity, yielding more powerful node representations and improved performance on downstream tasks.

We theoretically analyze the convergence guarantee of this tailored training paradigm by demonstrating its ability to avoid saddle points and achieve second-order convergence. Extensive experiments on various real-world node classification and link prediction benchmarks show that our proposed model can achieve significant performance gains against state-of-the-art methods.

The contributions of this paper are summarized as follows:

- We introduce a novel method that trains the GNN encoder by presenting data in a tailored, easy-to-hard order, enabling more effective design of pretext tasks. To the best of our knowledge, this is the first work to explore curriculum learning in graph self-supervised learning.
- We propose a unified framework that jointly reconstructs missing edges based on the unmasked graph structure and schedules training edges using a self-paced learning strategy, thereby improving the effectiveness of the GNN encoder.
- We provide theoretical analysis of the convergence properties of the proposed **Cur-MGAE** method and demonstrate through extensive experiments that it consistently outperforms state-of-the-art graph SSL methods, including both contrastive and generative approaches.

The rest of the paper is organized as follows. We first introduce the details of our proposed **Cur-MGAE** method in Section 2. In Section 3, we present the experimental results to show the effectiveness of the method, including quantitative comparisons, ablation studies, etc. We review the related works in Section 4. Finally, we conclude this work in Section 5.

2. Method

In this section, we present **Cur-MGAE** , a self-supervised framework designed to learn informative representations through the structure-aware curriculum. Specifically, **Cur-MGAE** consists of three key components: a structure-aware masked autoencoder, a complexity-guided curriculum masking module, and a self-paced mask scheduler. The overall framework is illustrated in Figure 1. The key notations in the method are summarized in Appendix A.

2.1. Structure-aware Masked Autoencoder

We propose a structure-aware masked autoencoder based on the edge reconstruction task to learn informative node representations without requiring extra supervision.

GNN Encoder. Let the input graph be denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ are the sets of nodes and edges, respectively. It can be represented as $\mathcal{G} = (\mathbf{X}, \mathbf{A})$, where $\mathbf{X}$ is the node feature matrix and $\mathbf{A}$ is the adjacency matrix. We apply a GNN to encode node representations:

$$h_v^{(k)} = \text{COM}(h_v^{(k-1)}, \text{AGG}(h_u^{(k-1)} : u \in N_v)), \quad (1)$$

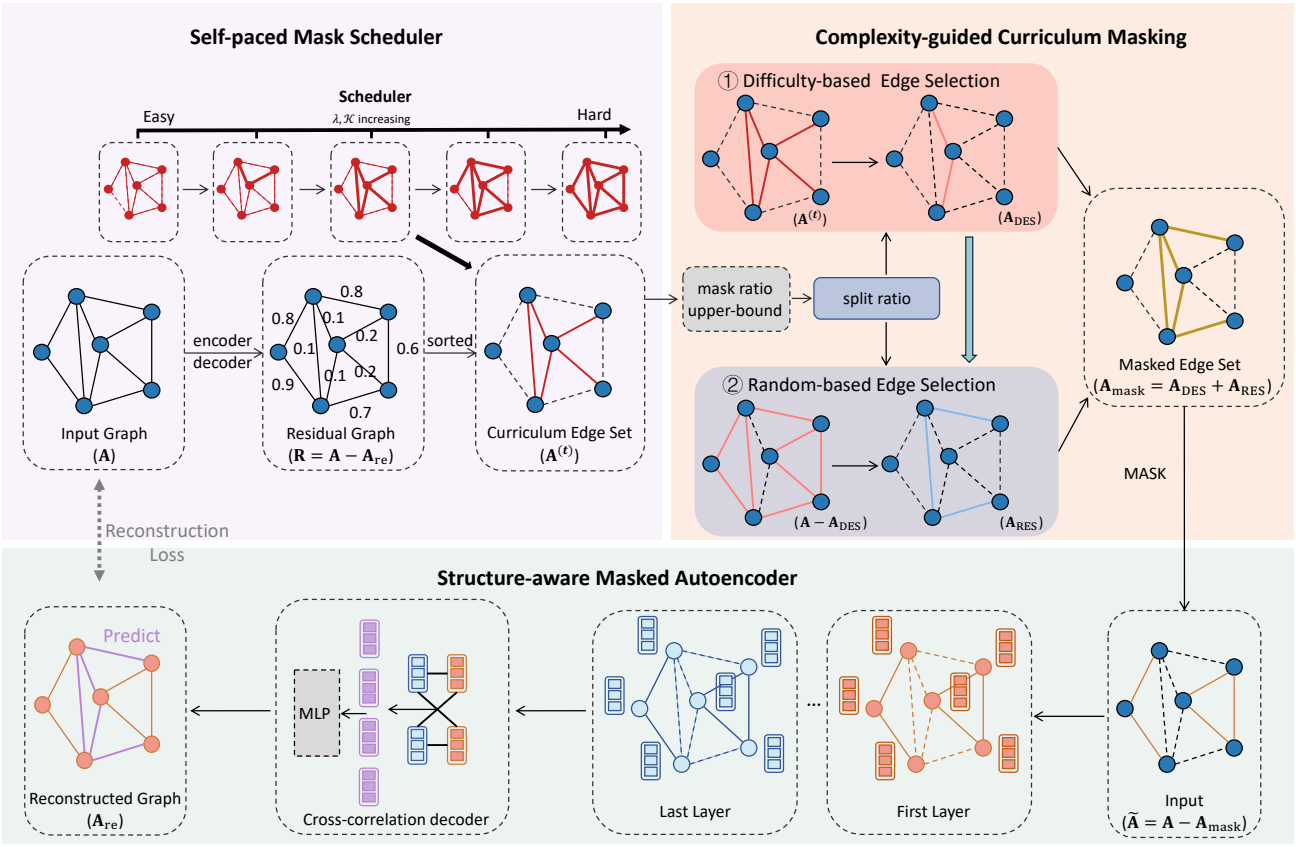

Figure 1. The framework of our proposed **Cur-MGAE** method. Given an input graph, we first introduce a complexity-guided curriculum masking module to identify edges to be masked, where each edge is assigned a difficulty score based on its reconstruction residual error. Next, we design a self-paced mask scheduler to dynamically schedule the masking curriculum according to the training stage. Finally, we employ a structure-aware masked autoencoder to perform self-supervised reconstruction of the graph structure.

where $h_v^{(k)}$ is the embedding of node v at the k-th layer, and $N_v = \{u : (v,u) \in \mathcal{E}\}$ denotes its neighborhood. AGG$(\cdot)$ aggregates messages from neighbors, and COM$(\cdot)$ combines them to update the embedding. We stack K GNN layers to derive multi-hop embeddings $\{h_v^{(1)}, h_v^{(2)}, ..., h_v^{(K)}\}$. The final node representations are denoted as ENC$(\mathbf{X}, \mathbf{A}) \in \mathbb{R}^{N\times d}$, where N is the number of nodes, d is the embedding dimension, and ENC$(\cdot)$ is the encoder.

Cross-correlation Decoder. After obtaining node embeddings from the GNN encoder, we design a cross-correlation decoder to reconstruct the graph structure by capturing inherent similarities between nodes following (Tan et al., 2023). Specifically, we compute edge embeddings as:

$$h_{e_{v,u}} = ||_{k,j=1}^{K} h_v^{(k)} \odot h_u^{(j)}, \tag{2}$$

where $\odot$ denotes element-wise multiplication, $||$ represents concatenation, and $h_{e_{v,u}} \in \mathbb{R}^{dK^2}$ is the resulting edge embedding. This formulation emphasizes shared features between node pairs while suppressing dissimilar components, enabling the model to retain only highly correlated structural patterns. The resulting edge embeddings are fed into a multilayer perceptron (MLP) with a sigmoid activation to estimate the existence probability of each edge: $g(v,u) = \text{MLP}(h_{e_{v,u}})$. By selectively preserving informative and correlated features, this design filters out noise and facilitates more accurate and efficient edge prediction.

Reconstruction Task. We adopt a mask-reconstruction paradigm for self-supervised learning to enhance the quality of the learned node representations. Specifically, we select a part of the edges to be masked in the original graph to obtain the perturbed graph: $\tilde{\mathbf{A}} = \mathbf{A} - \mathbf{A}_{\text{mask}}$. We denote $\mathbf{A}_{\text{mask}}$ as the adjacency matrix of the masked edges $\mathcal{E}_{\text{mask}}$. Then we adopt the reconstruction loss as the supervision signal: $\mathcal{L}_{SSL} = \ell(\mathbf{A}, \text{DEC}(\text{ENC}(\mathbf{X}, \tilde{\mathbf{A}}))$, whose implementation is as follows:

$$\mathcal{L}_{SSL} = -\frac{1}{|\mathcal{E}_{\text{mask}}|} \sum_{(v,u)\in\mathcal{E}_{\text{mask}}} \log \frac{\exp(g(v,u))}{\sum_{v'\in\mathcal{V}} \exp(g(v,v'))}. \tag{3}$$

$g(\cdot)$ is the predicted probability of the presence of an existing edge, namely $g(v,u) = \text{MLP}(||_{k,j=1}^{K} h_v^{(k)} \odot h_u^{(j)})$,

where $h_u^{(j)}$ and $h_v^{(k)}$ denote the j-th and k-th hidden representation of node u and v, respectively. The learned node representations encode rich structural and attribute information, sufficient to reconstruct the original graph from perturbed inputs and further enhance performance in downstream tasks.

2.2. Complexity-guided Curriculum Masking

Since different edges in a graph contribute unequally to its structure, randomly masking edges can pose optimization challenges during the reconstruction process. In particular, masking too many critical structural edges early in training may lead to excessively difficult reconstruction tasks, especially when the GNN encoder is still undertrained. To mitigate this issue, we introduce a complexity-guided curriculum masking module that progressively increases task difficulty by selecting edges in an easy-to-hard manner. The key idea is to identify structurally important edges and postpone their masking, thereby enabling a smoother and more effective learning trajectory.

Specifically, we identify edges that are structurally important to the graph and progressively mask them to increase the learning difficulty during the training process. We formally define the difficulty of an edge as a score reflecting how challenging it is for the current model to predict the edge correctly. To quantify this difficulty, we first use the current model to reconstruct the original graph as: $\mathbf{A}_{\text{re}} = \text{DEC}(\text{ENC}(\mathbf{X}, \mathbf{A}))$.

The reconstructed adjacency matrix $\mathbf{A}_{\text{re}}$ captures the model's internal estimation of edge probabilities, which can be interpreted as its confidence in the existence of each edge. Intuitively, lower confidence suggests that an edge is harder to reconstruct for the current model, indicating higher structural complexity. We therefore propose to use the structural residual (Zhang et al., 2023a) between the original and predicted graphs as a proxy for edge difficulty: $\mathbf{R} = \mathbf{A} - \mathbf{A}_{\text{re}}$. By applying a masking strategy that targets the $\mathcal{K}$ easiest edges, those with the smallest residuals, we simplify the reconstruction task, particularly during the early training stages. This allows the model to focus on learning fundamental structural patterns before encountering more complex ones (Zhang et al., 2023a; Li et al., 2023b). Consequently, this curriculum-guided masking strategy enhances both the efficiency and effectiveness of the training process.

2.3. Self-paced Mask Scheduler

Here, we propose a self-paced mask scheduler to progressively and autonomously incorporate an increasing number of edges throughout the training process (Zhang et al., 2023a; Li et al., 2023b). One straightforward solution is to gradually increase the value of $\mathcal{K}$ during the training process. However, dynamically identifying and updating a suitable $\mathcal{K}$ during training is non-trivial. Edge selection inherently poses a discrete optimization problem over a large topological space, which significantly complicates the learning process. To address this, we relax the edge selection matrix $\mathbf{S}^{(t)}$ from binary values to continuous values within $[0, 1]$, transforming the problem into a continuous constrained optimization task. Specifically, we treat the masking constraint as a Lagrangian multiplier and introduce a regularization component to the loss function: $f(\mathbf{S}; \lambda, \mathbf{A}) = \lambda||\mathbf{S}^{(t)} \odot \mathbf{A} - \mathbf{A}||$. Here, $\mathbf{S}^{(t)}$ denotes the soft edge selection matrix at training iteration t, sharing the same dimensions as the adjacency matrix $\mathbf{A}$. After optimization, the entries in $\mathbf{S}^{(t)}$ are thresholded at 0.5 to yield a binary mask $0, 1$ for edge selection, resulting in the masked adjacency matrix $\mathbf{A}^{(t)} = \mathbf{S}^{(t)} \odot \mathbf{A}$, where $\odot$ denotes element-wise multiplication.

Note that the regularization term promotes the masking of as many edges as possible, governed by the coefficient λ. As λ increases during training, more edges are gradually incorporated. This process effectively schedules edge masking in an easy-to-hard manner. The evolving strategy for updating λ is detailed in Appendix D.3. Combining both the residual-based selection and the regularization term (Zhang et al., 2023a), the loss function for our self-paced mask scheduler is given by:

$$\mathcal{L}_{SPCL} = \beta \sum_{i,j} S_{ij} R_{ij} + f(\mathbf{S}; \lambda, \mathbf{A}), \tag{4}$$

where β is a balancing hyperparameter, $\mathbf{S}$ extracts the selected edges, and $R_{ij} = ||A_{ij} - \tilde{A}_{ij}^{(t)}||$ denotes the edge residual, with $\tilde{A}_{ij}^{(t)}$ being the predicted edge value. The L_2 norm is used for measuring residuals.

However, continuously increasing the number of masked edges may eventually compel the model to make uninformed or arbitrary predictions about the graph structure. To avoid this issue, we introduce a *mask ratio* hyperparameter that constrains the maximum proportion of edges allowed to be masked. Another critical concern is that relying solely on difficulty-based edge selection tends to consistently mask only the easiest edges during training. This could limit the model's generalizability by overfitting to these simpler structures. To address this, we introduce a *split ratio* hyperparameter that enables a portion of the masked edges to be selected randomly. Specifically, a fraction of the masked edges, denoted as $\mathbf{A}_{\text{DES}}$, is sampled uniformly at random, while the remaining edges, denoted as $\mathbf{A}_{\text{RES}}$, are selected based on their difficulty scores. The union of these two subsets forms the complete masked edge set $\mathbf{A}_{\text{mask}}$. A lower *split ratio* results in a higher degree of randomness in the masking process, thereby encouraging exploration. Conversely, a higher *split ratio* prioritizes difficulty-based masking, enhancing exploitation. By appropriately tuning this hyperparameter, we strike a balance between exploration and exploitation, thereby mitigating overfitting and improving the quality of

Algorithm 1 The optimization process of **Cur-MGAE**

Input: Node features $\mathbf{X}$, adjacency matrix $\mathbf{A}$, step size μ, regularization coefficient γ
Output: Trained GNN parameters $\mathbf{w}$
Initialize $\mathbf{w}^{(0)}$, $\mathbf{S}^{(0)}$, and $\lambda^{(0)}$
Compute $\mathbf{A}^{(0)} = \mathbf{S}^{(0)} \odot \mathbf{A}$
while not converged **do**
  $\mathbf{w}^{(t)} = \text{argmin}_{\mathbf{w}} \mathcal{L}_{SSL}(\mathbf{X}, \mathbf{A}^{(t-1)}; \mathbf{w}) + \frac{\gamma}{2}||\mathbf{w} - \mathbf{w}^{(t-1)}||$
  Generate embedding $\mathbf{Z}^{(t)}$ from $\mathbf{A}$ using the updated GNN model f
  For all node pairs (i, j), predict edge existence $\tilde{A}_{ij}^{(t)} = g(z_i^{(t)}, z_j^{(t)})$
  Relax $\mathbf{S}^{(t)}$ to the continuous domain and optimize it as $\mathbf{S}^{(t)} = \text{argmin}_{\mathbf{S}} \mathcal{L}_{SPCL} + \frac{\gamma}{2}||\mathbf{S} - \mathbf{S}^{(t-1)}||$
  $\mathcal{K} = |\{(i, j) : S_{ij}^{(t)} \geq 0.5\}|$
  if $\mathcal{K} \geq$ *mask ratio* $\times |\mathcal{E}|$ **then**
    $\mathcal{K} =$ *mask ratio* $\times |\mathcal{E}|$
  else
    Update λ based on the designed curriculum pace
  end if
  Select the top-*split ratio* $\times \mathcal{K}$ edges with the highest S_{ij} values and assign to $\mathbf{S}_{\text{DES}}^{(t)}$
  Randomly select the remaining $(1 -$ *split ratio*$) \times \mathcal{K}$ edges and assign to $\mathbf{S}_{\text{RES}}^{(t)}$
  $\mathbf{S}^{(t)} = \mathbf{S}_{\text{DES}}^{(t)} + \mathbf{S}_{\text{RES}}^{(t)}$
  Update the perturbed adjacency matrix $\mathbf{A}^{(t)} = \mathbf{S}^{(t)} \odot \mathbf{A}$
end while

the learned node representations.

2.4. Optimization Procedure

Our proposed model aims to minimize the objective function $\mathcal{L}_{all}$, which involves optimizing two distinct sets of parameters. This naturally leads to a challenging bi-level optimization problem. To address this, we design an optimization algorithm that jointly trains two separate self-supervised modules, each with its corresponding objective. The overall loss function is formulated as:

$$\mathcal{L}_{all} = \mathcal{L}_{SSL} + \mathcal{L}_{SPCL}. \tag{5}$$

To ensure a smooth transition across training iterations, we incorporate regularization terms into the optimization process: $\frac{\gamma}{2}||\mathbf{w} - \mathbf{w}^{(t-1)}||$ and $\frac{\gamma}{2}||\mathbf{S} - \mathbf{S}^{(t-1)}||$. These terms penalize abrupt changes in the model parameters $\mathbf{w}$ and the edge selection matrix $\mathbf{S}$, thereby stabilizing training dynamics. The complete training procedure is outlined in Algorithm 1.

Time Complexity. The time complexity of our proposed model is $O(Ed+Nd^2)$, where N and E denote the numbers of nodes and edges in the graph, respectively, and d is the dimensionality of the node representations. Specifically, our model adopts a message-passing GNN as the encoder, which incurs a complexity of $O(Ed + Nd^2)$. The decoder and the self-paced mask scheduler each contribute a time complexity of $O(Ed)$, as they involve computing residual errors for each edge. The complexity-guided curriculum masking module operates with a time complexity of $O(E)$, since it selects from existing edges rather than the full $N \times N$ set of potential edges. Overall, the time complexity of our method is comparable to the baselines, which also typically scale as $O(Ed + Nd^2)$.

2.5. Theoretical Analyses

We present theoretical analyses on the convergence properties of our method in Theorems 1 and 2 following (Zhang et al., 2023a). Detailed proofs are provided in Appendix B.

Theorem 1 (Convergence Away from Saddle Points)**.** *For a sufficiently large γ, if the second derivatives of $\mathcal{L}_{SSL}(\mathbf{X}, \mathbf{A}^{(t-1)}; \mathbf{w})$ and $f(\mathbf{S}; \lambda, \mathbf{A})$ are continuous, any bounded sequence $(\mathbf{w}^{(t)}, \mathbf{S}^{(t)})$ generated by Algorithm 1 with random initialization will almost surely avoid convergence to any strict saddle point of $\mathcal{L}_{all}$.*

Theorem 2 (Convergence to Second-order Stationary Points)**.** *For a sufficiently large γ, if the second derivatives of $\mathcal{L}_{SSL}(\mathbf{X}, \mathbf{A}^{(t-1)}; \mathbf{w})$, and $f(\mathbf{S}; \lambda, \mathbf{A})$ are continuous, and both functions satisfy the Kuradyka-Lojasiewicz (KL) property (Wang et al., 2022b), then any bounded sequence $(\mathbf{w}^{(t)}, \mathbf{S}^{(t)})$ generated by Algorithm 1 with random initialization will almost surely converge to a second-order stationary point of $\mathcal{L}_{all}$.*

3. Experiment

In this section, we conduct comprehensive experiments to evaluate the effectiveness of the proposed **Cur-MGAE** method. This includes the experimental setup, quantitative evaluations on node classification and link prediction benchmarks, and in-depth analyses. Additional experimental results are provided in Appendix G.

3.1. Experimental Setup

Datasets. We evaluate node classification on three Planetoid datasets (Cora, Citeseer, and Pubmed (Sen et al., 2008)) and three commonly used citation/co-authorship datasets: Coauthor-CS (Shchur et al., 2019), Coauthor-Physics (Shchur et al., 2019), and OGBN-arxiv (Hu et al., 2020a). Accuracy (%) is used as the evaluation metric for these tasks. For link prediction, we use the same three Planetoid datasets and additional large-scale benchmarks from the Open Graph Benchmark (OGB) (Hu et al., 2021), including OGBN-ddi, OGBL-collab, and OGBL-ppa. We report the area under the ROC curve (AUC, %) (Bradley, 1997) for the three Planetoid datasets, and the Hit rate (Hits@N) for OGB datasets, following (Tan et al., 2023) for fair comparison.

Table 1. Node classification accuracy (%) of our proposed method and baselines. In each column, the boldfaced score denotes the best result among all methods. The rightmost column shows the average rank. Our method achieves the best average rank.

Dataset	Cora	Citeseer	Pubmed	Coauthor-CS	Coauthor-Physics	OGBN-arxiv	Rank
DGI	85.41 ± 0.34	74.51 ± 0.51	76.80 ± 0.60	92.77 ± 0.38	94.55 ± 0.13	67.08 ± 0.43	9.50
GIC	**87.70 ± 0.01**	**76.39 ± 0.02**	77.40 ± 1.90	91.33 ± 0.30	93.49 ± 0.42	64.00 ± 0.22	9.17
MVGRL	85.86 ± 0.15	73.18 ± 0.22	80.10 ± 0.70	92.87 ± 0.13	95.35 ± 0.08	68.33 ± 0.32	8.42
BGRL	86.16 ± 0.20	73.96 ± 0.14	82.05 ± 0.85	93.35 ± 0.06	**96.16 ± 0.09**	71.77 ± 0.19	4.00
GAE	83.60 ± 0.52	63.37 ± 1.21	78.23 ± 1.63	89.79 ± 0.09	93.26 ± 0.05	66.01 ± 0.37	13.67
GraphSage	74.30 ± 1.84	60.20 ± 2.15	81.96 ± 0.74	89.74 ± 0.19	93.35 ± 0.06	64.79 ± 2.91	13.00
ARGVA	85.86 ± 0.72	73.10 ± 0.86	81.51 ± 1.00	84.68 ± 0.26	92.89 ± 0.11	50.06 ± 1.21	12.08
GPT-GNN	84.69 ± 0.09	71.82 ± 0.13	81.45 ± 0.18	91.07 ± 0.21	95.02 ± 0.15	70.16 ± 0.10	10.33
RRL	57.29 ± 0.13	59.57 ± 1.77	75.06 ± 0.37	84.71 ± 0.95	94.90 ± 0.02	66.36 ± 0.13	14.33
GraphMAE	85.45 ± 0.40	72.48 ± 0.77	81.10 ± 0.40	**93.47 ± 0.04**	96.13 ± 0.03	71.86 ± 0.00	6.50
GraphMAE2	84.50 ± 0.60	73.40 ± 0.30	81.40 ± 0.50	92.13 ± 0.12	95.44 ± 0.08	71.89 ± 0.03	8.25
MaskGAE	87.31 ± 0.05	75.20 ± 0.07	83.58 ± 0.45	92.31 ± 0.05	95.79 ± 0.02	70.99 ± 0.12	4.50
Bandana	84.62 ± 0.37	73.60 ± 0.16	83.53 ± 0.51	93.10 ± 0.05	95.57 ± 0.04	71.09 ± 0.24	6.33
AUG-MAE	84.30 ± 0.40	73.20 ± 0.40	81.40 ± 0.40	92.15 ± 0.22	95.34 ± 0.60	71.90 ± 0.20	8.58
S2GAE	86.15 ± 0.25	74.60 ± 0.06	84.19 ± 0.21	91.70 ± 0.08	95.82 ± 0.03	72.02 ± 0.05	4.50
Cur-MGAE	87.25 ± 0.55	74.68 ± 0.37	**85.86 ± 0.14**	92.69 ± 0.17	95.91 ± 0.05	**73.00 ± 0.06**	**2.83**

Baselines. We compare **Cur-MGAE** against two groups of state-of-the-art baselines. The first group includes contrastive graph self-supervised learning methods: DGI (Veličković et al., 2019), GIC (Mavromatis & Karypis, 2021), MVGRL (Hassani & Khasahmadi, 2020), and BGRL (Thakoor et al., 2022). The second group includes generative graph SSL methods such as GAE (Kipf & Welling, 2016), GraphSAGE (Hamilton et al., 2017), ARGVA (Pan et al., 2019), GPT-GNN (Hu et al., 2020b), RRL (Zhu et al., 2020), GraphMAE (Hou et al., 2022), GraphMAE2 (Hou et al., 2023), MaskGAE (Li et al., 2023d), Bandana (Zhao et al., 2024), AUG-MAE (Wang et al., 2024), and S2GAE (Tan et al., 2023).

3.2. Experimental Results

Node Classification. Table 1 summarizes the node classification accuracy of **Cur-MGAE** and all baselines. Our method outperforms both contrastive and generative self-supervised baselines, achieving the highest average rank. This result demonstrates the benefit of scheduling training data using a difficulty-aware curriculum derived from reconstruction residuals, which enables more effective representation learning. For instance, **Cur-MGAE** improves classification accuracy by 1.67% on Pubmed and nearly 1% on OGBN-arxiv compared to the strongest baseline.

Link Prediction. Table 2 presents the link prediction performance of **Cur-MGAE** and baseline methods[1]. The results indicate that generative graph SSL methods (e.g., GraphMAE, MaskGAE, S2GAE) generally outperform contrastive methods, highlighting the effectiveness of the reconstruction-based pretext tasks that recover masked structures from the remaining graph context. Our curriculum-based method **Cur-MGAE** achieves the best performance on 2 out of 6 datasets and reports competitive results on the remaining datasets. For example, it improves performance by approximately 2% over the strongest baselines on OGBL-ddi and OGBL-ppa. This improvement is attributed to **Cur-MGAE** 's ability to adaptively select training samples based on task difficulty, unlike most baselines that treat all training data equally, leading to suboptimal results. MaskGAE (Li et al., 2023d), a strong baseline that jointly reconstructs masked edges and node degrees, performs well on smaller datasets but underperforms on large-scale benchmarks. A plausible explanation is that it overlooks the varying difficulty of reconstructing different edges, which becomes more impactful in large, complex graphs where informative representations are harder to extract. In contrast, our method introduces a curriculum-driven strategy that prioritizes easier samples in early training stages and progressively incorporates harder ones. Notably, none of the baselines achieves consistently strong performance across all datasets, whereas **Cur-MGAE** demonstrates stable and superior effectiveness.

[1]Note that GraphMAE2 and AUG-MAE are omitted here since they are node or graph classification methods and are not designed for link prediction tasks.

3.3. Visualization of Learned Edge Selection Curriculum

To qualitatively evaluate the learned edge selection strategy, we construct synthetic datasets with ground-truth edge difficulty labels, following previous works (Karimi et al., 2018; Abu-El-Haija et al., 2019; Zhang et al., 2023a). Each synthetic graph consists of 5,000 nodes partitioned into 10 equally sized groups, with node labels ranging from 1 to 10. The corresponding visualization is provided in Appendix F. The node features are generated from overlapping multi-Gaussian distributions that define each node's position in

Table 2. Link prediction results (%) of our proposed method and baselines. **Cur-MGAE** achieves consistently strong performance across both small-scale and large-scale benchmark datasets. "–" indicates out-of-memory errors on a 24GB GPU, while "/" denotes that the method is not applicable to the corresponding dataset.

Dataset Metric	Cora AUC	Citeseer AUC	Pubmed AUC	OGBL-ddi Hits@20	OGBL-collab Hits@50	OGBL-ppa Hits@10	Rank
DGI	90.02 ± 0.80	95.53 ± 0.40	91.24 ± 0.60	–	–	–	11.17
GIC	93.54 ± 0.60	97.04 ± 0.50	93.71 ± 0.30	–	–	–	9.67
MVGRL	87.46 ± 0.38	88.95 ± 0.66	88.36 ± 0.59	–	–	–	13.33
BGRL	87.08 ± 0.24	85.82 ± 0.36	96.75 ± 0.12	–	21.58 ± 1.92	–	12.17
GAE	91.09 ± 0.01	90.52 ± 0.04	96.40 ± 0.01	37.07 ± 5.07	44.75 ± 1.07	2.52 ± 0.47	7.33
GraphSage	86.33 ± 1.06	85.65 ± 2.56	89.22 ± 0.87	53.90 ± 4.74	54.63 ± 1.12	1.87 ± 0.67	9.00
ARGVA	92.40 ± 0.00	91.94 ± 0.00	96.81 ± 0.00	20.43 ± 4.66	28.39 ± 2.51	0.41 ± 0.26	7.83
GPT-GNN	92.28 ± 0.31	91.36 ± 0.66	97.83 ± 0.03	37.05 ± 5.96	42.41 ± 1.80	1.57 ± 0.94	6.67
RRL	88.46 ± 1.85	85.47 ± 1.01	93.10 ± 0.49	16.84 ± 2.23	29.88 ± 2.94	0.24 ± 0.19	10.83
GraphMAE	89.19 ± 0.00	91.20 ± 0.11	93.72 ± 0.00	–	22.79 ± 1.62	0.18 ± 0.28	10.92
MaskGAE	**96.66 ± 0.17**	**98.00 ± 0.23**	**98.84 ± 0.04**	16.25 ± 1.60	32.47 ± 0.59	0.23 ± 0.04	5.00
Bandana	95.71 ± 0.12	96.89 ± 0.21	97.26 ± 0.16	/	48.67 ± 3.82	1.32 ± 1.26	4.92
S2GAE-SAGE	95.05 ± 0.76	94.85 ± 0.49	97.38 ± 0.17	66.00 ± 9.49	49.27 ± 0.96	1.37 ± 0.38	4.67
S2GAE-GCN	93.52 ± 0.23	93.29 ± 0.49	98.30 ± 0.12	65.91 ± 3.50	**54.74 ± 1.06**	3.98 ± 1.33	3.83
Cur-MGAE	95.22 ± 0.54	95.20 ± 0.31	98.43 ± 0.06	**68.50 ± 5.06**	52.28 ± 1.35	**5.96 ± 0.96**	**2.67**

the feature space. The labels are then assigned according to the feature distributions, resulting in 10 distinct classes. Edge difficulty is defined based on the label similarity between node pairs: edges between nodes with identical labels are considered *easy*, those between adjacent labels are of *medium* difficulty, and those connecting nodes with distant labels are deemed *hard* to reconstruct. To control the prevalence of easy edges, we introduce a *homophily coefficient* (*homo*) that specifies the ratio of easy edges in the graph. For all other potential edges, the connection probability decreases exponentially with label distance. Formally, the probability of an edge between nodes u and v is defined as: $p_{uc} \propto e^{-|c_u - c_v|}$, where $|c_u - c_v|$ denotes the shortest label distance in a circular label arrangement. We generate three synthetic datasets by setting the *homophily coefficient* to 0.1, 0.5, 0.9, and split each graph into training, validation, and test sets with equal numbers of nodes.

Learned Edge Selection Curriculum. Using the synthetic datasets described above, we visualize and compare the learned edge selection curriculum with the ground-truth edge difficulty distribution. Figure 2 shows the proportion of selected edges categorized as easy, medium, and hard throughout training. In this figure, each row corresponds to a different *homophily coefficient*, while each column represents a different *split ratio*, a hyperparameter that controls the trade-off between exploration and exploitation in edge selection. Across all settings, we observe a consistent trend: the model initially favors selecting easier edges and gradually incorporates harder ones as training progresses. This behavior is consistent with the intended design of our curriculum-driven training strategy. Specifically, in early epochs, the model focuses on easier edges that align with its current learning capacity. As training advances and the model becomes more expressive, it begins to select increasingly difficult edges, effectively expanding its learning scope in a controlled, progressive manner. Interestingly, the *homophily coefficient* influences the dynamics of the easy-to-hard transition. As illustrated in the stacked plots, where the blue, orange, and red regions represent the relative proportions of selected easy, medium, and hard edges at each epoch, low homophily leads to a more aggressive curriculum. In this setting, the model selects a substantial fraction of medium and hard edges even in the early stages of training. In contrast, under high homophily, early edge selection is dominated by easy edges, with medium and hard edges incorporated more gradually. This pattern suggests that in highly homophilous graphs, the model adopts a more conservative learning trajectory, initially relying on structurally similar (and easier) edges before progressively transitioning to more challenging ones. These visualizations confirm that our structure-aware masking curriculum behaves as expected, progressively selecting edges from easier to harder throughout the training process.

3.4. Ablation Studies

To evaluate the effectiveness of our key modules and designs, we conduct ablation studies by modifying the components of **Cur-MGAE** . For simplicity, we present results on Cora, Citeseer, OGBL-ddi, and Coauthor-CS in Table 3, with similar trends observed on the remaining datasets. We use AUC (%) for link prediction and accuracy (%) for node classification as evaluation metrics.

Variant 'w/o Curri.'. It disables the complexity-guided curriculum masking module (Section 2.2) and instead applies random masking. The performance drop demonstrates

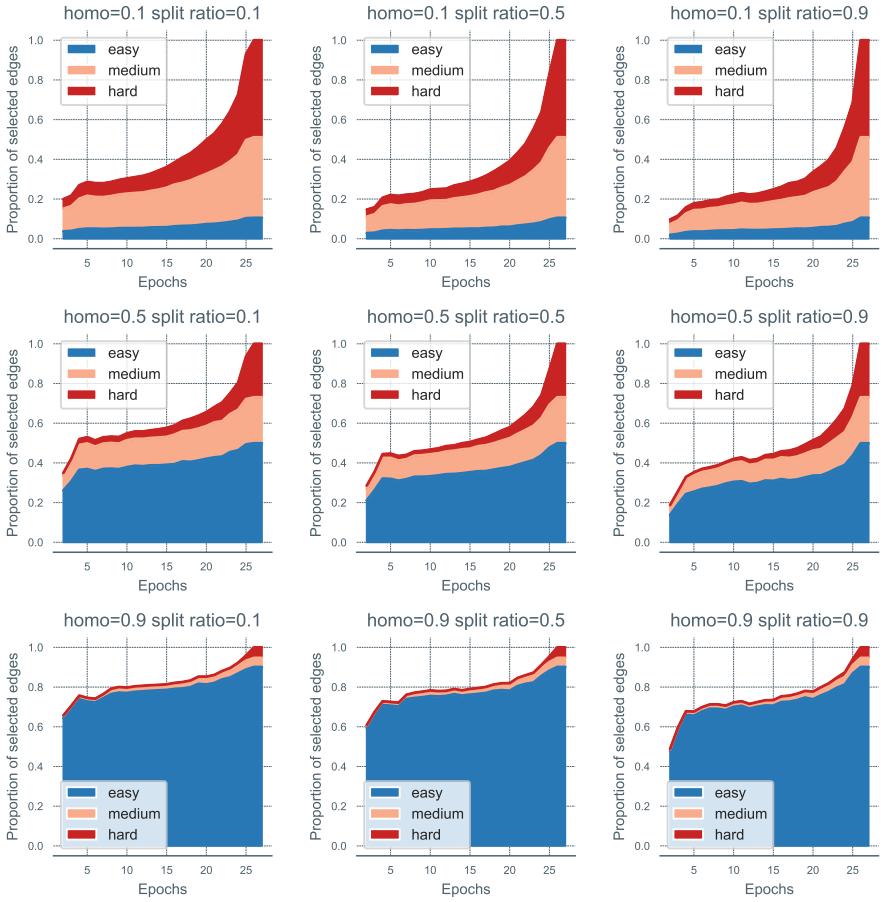

Figure 2. Edge selection dynamics across synthetic datasets under varying homophily coefficients (rows) and split ratios (columns). At each epoch, the stacked colored areas represent the relative proportions of selected easy (blue), medium (orange), and hard (red) edges.

that treating all samples equally during training can be sub-optimal. This supports the benefit of our structure-aware curriculum design in promoting better performance.

Variant 'split ratio is 0'. Setting the *split ratio* to 0 means that the self-paced mask scheduler (Section 2.3) randomly selects a fixed number of edges without considering their difficulty scores. Although this introduces stochasticity and helps prevent overfitting, the lack of difficulty-aware edge selection leads to noticeable performance degradation, highlighting the importance of meaningful edge scheduling.

Variant 'split ratio is 1'. In this variant, we set the *split ratio* to 1, which removes the randomness in the edge selection process of the self-paced mask scheduler (Section 2.3). Together with the previous variant, these two variants aim to assess the effectiveness of the self-paced scheduling mechanism. Without randomness, the scheduler consistently selects edges with the smallest difficulty scores during early training stages. This deterministic behavior may lead to overfitting and limit the model's ability to generalize, as reflected in the performance drop compared to the full model.

Variant 'w/o CC Dec.'. This variant evaluates the impact of our specially designed cross-correlation decoder in the structure-aware masked autoencoder (Section 2.1). We replace it with a simpler inner product decoder while keeping all other components unchanged. The significant drop in performance demonstrates that the cross-correlation decoder, by capturing multi-granular shared features between connected nodes, facilitates more informative representation learning and improves reconstruction accuracy.

Variant 'w/o CC Dec. & Curri.'. In this variant, both the decoder and the curriculum masking are removed: we use an inner product decoder and randomly mask the training samples. This combination results in the worst performance across all settings, confirming that the two modules, i.e., decoder and curriculum, play synergistic and crucial roles in achieving strong representation learning.

Overall, these ablation studies confirm the importance of the key components in our proposed method. The tailored structure-aware curriculum enables the identification of more informative edges by leveraging difficulty scores,

Table 3. Ablation studies on key components. "Curri." refers to the complexity-guided curriculum masking module, and "CC Dec." denotes the cross-correlation decoder.

	Link Prediction			Node Classification		
Datasets	Cora	Citeseer	OGBL-ddi	Cora	Citeseer	Coauthor-CS
Cur-MGAE	**95.22 ± 0.54**	**95.20 ± 0.31**	**68.50 ± 5.06**	**87.25 ± 0.55**	**74.68 ± 0.37**	**92.69 ± 0.17**
w/o Curri.	92.89 ± 0.40	93.66 ± 0.23	61.70 ± 9.64	86.08 ± 0.15	73.92 ± 0.44	91.67 ± 0.03
split ratio is 1	93.80 ± 1.24	92.17 ± 1.13	62.90 ± 11.31	86.13 ± 0.42	74.39 ± 0.17	91.58 ± 0.12
split ratio is 0	94.12 ± 0.47	92.21 ± 0.52	62.49 ± 8.97	86.93 ± 0.15	74.71 ± 0.24	91.69 ± 0.02
w/o CC Dec.	87.43 ± 0.53	85.49 ± 0.35	22.69 ± 3.65	83.05 ± 0.90	70.15 ± 0.32	90.55 ± 0.24
w/o CC Dec. & Curri.	87.21 ± 0.69	85.18 ± 0.99	20.73 ± 1.72	82.89 ± 0.11	69.09 ± 0.93	89.93 ± 0.03

while the *split ratio* introduces a controlled level of stochasticity to avoid overfitting. Meanwhile, the cross-correlation decoder mitigates the representational limitations introduced by the masking process and significantly enhances the reconstruction quality.

4. Related Work

Graph Self-Supervised Learning. Graph self-supervised learning (SSL) techniques (Liu et al., 2022; You et al., 2020; Peng et al., 2020; Xu et al., 2021; Sun et al., 2023b; Li et al., 2022c; 2021b; 2024a) are typically categorized into contrastive and generative paradigms. Recently, contrastive methods have gained significant attention. These approaches primarily focus on negative sampling strategies, such as corruption-based negative pair construction in DGI (Veličković et al., 2019), and in-batch negatives as used in GCA (Zhu et al., 2021). In contrastive learning, graph augmentation plays a critical role in creating effective training signals. However, the theoretical understanding of graph augmentation remains limited, raising concerns about its label invariance and optimality. In contrast, generative SSL methods aim to reconstruct missing parts of input graphs and are generally divided into autoregressive and autoencoding models. Although generative approaches have historically underperformed compared to contrastive methods, several notable autoregressive models have emerged, such as GPT-GNN (Hu et al., 2020b). In the autoencoding category, early models like GAE and VGAE (Kipf & Welling, 2016) set foundational benchmarks. More recent advances include GraphMAE (Hou et al., 2022), GraphMAE2 (Hou et al., 2023), GigaMAE (Shi et al., 2023), SeeGera (Li et al., 2023e), RARE (Tu et al., 2023), S2GAE (Tan et al., 2023), and Bandana (Zhao et al., 2024). Nonetheless, most existing methods neglect the varying difficulty levels of self-supervised tasks, treating all training samples equally and resulting in suboptimal performance on downstream tasks.

Curriculum Learning. Curriculum Learning (CL) is a training strategy that starts with simpler tasks and gradually moves to more complex ones, inspired by the way humans learn in educational settings (Bengio et al., 2009; Wang et al., 2021a; Zhou et al., 2023; 2024; Huang et al., 2024; Zhang et al., 2024; Yao et al., 2024; Ge et al., 2025). A foundational approach in this domain is the "Baby Step" algorithm (Spitkovsky et al., 2010), which controls both the complexity and order of training samples. This idea was later extended to the self-paced learning paradigm (Kumar et al., 2010), which selects training samples based on their loss values, allowing models to learn at their own pace. In addition, several automatic CL frameworks have been proposed, including transfer teacher (Hacohen & Weinshall, 2019), reinforcement learning-based curriculum teacher (Zhao et al., 2020), and others customized for different datasets, models, and objectives (Sinha et al., 2020). CL has also been integrated into various domains such as disentangled recommendation systems (Chen et al., 2021; Wang et al., 2023), combinatorial optimization (Zhang et al., 2022c), neural architecture search (Zhou et al., 2022; Yao et al., 2024; Qin et al., 2023), and video grounding (Lan et al., 2023). Several studies have extended CL to graph domains (Li et al., 2023b), such as GNN-CL (Li et al., 2024b) and CurGraph (Wang et al., 2021b). A central component of CL methods is the mechanism to assess sample complexity and guide the training schedule by determining the order or relative importance of samples. However, most existing methods heavily rely on label supervision to train encoders and overlook self-supervised settings where labels are not available. More importantly, these methods typically treat training samples as independent instances, whereas our approach addresses a more challenging structure-aware curriculum setting in which training samples are inherently interconnected and thus cannot be treated as independent.

5. Conclusion

In this paper, we propose a graph self-supervised learning strategy based on curriculum learning, named **Cur-MGAE**, which builds upon a masked graph autoencoder framework. Our method assesses the difficulty of reconstructing masked edges and thus guides the model to learn in an easy-to-hard manner. This approach leads to the acquisition of more informative graph representations. We provide a theoretical analysis of the convergence properties of the proposed method. Extensive experiments on real-world benchmarks demonstrate that our proposed method consistently outperforms state-of-the-art graph self-supervised learning baselines in both node classification and link prediction tasks.

Acknowledgements

This work is supported by National Natural Science Foundation of China No.62222209, Beijing National Research Center for Information Science and Technology under Grant No.BNR2023TD03006.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Abu-El-Haija, S., Perozzi, B., Kapoor, A., Alipourfard, N., Lerman, K., Harutyunyan, H., Steeg, G. V., and Galstyan, A. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing, 2019.

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pp. 41–48, 2009.

Bradley, A. P. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159, 1997.

Cai, J., Wang, X., Li, H., Zhang, Z., and Zhu, W. Multimodal graph neural architecture search under distribution shifts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 8227–8235, 2024.

Chen, H., Chen, Y., Wang, X., Xie, R., Wang, R., Xia, F., and Zhu, W. Curriculum disentangled recommendation with noisy multi-feedback. *Advances in Neural Information Processing Systems*, 34:26924–26936, 2021.

Chen, H., Wang, X., Zhang, Z., Li, H., Feng, L., and Zhu, W. Autogfm: Automated graph foundation model with adaptive architecture customization. In *International Conference on Machine Learning*, 2025.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Ding, K., Wang, Y., Yang, Y., and Liu, H. Eliciting structural and semantic global knowledge in unsupervised graph contrastive learning. AAAI'23/IAAI'23/EAAI'23. AAAI Press, 2023. ISBN 978-1-57735-880-0. doi: 10.1609/aaai.v37i6.25898.

Fan, X., Gong, M., Wu, Y., Qin, A., and Xie, Y. Propagation enhanced neural message passing for graph representation learning. *TKDE*, 2021.

Feng, W., Li, H., Wang, X., Duan, X., Qian, Z., Liu, W., and Zhu, W. Multimedia cognition and evaluation in open environments. In *Proceedings of the 1st International Workshop on Multimedia Content Generation and Evaluation: New Methods and Practice*, pp. 9–18, 2023.

Gao, J., Gao, J., Ying, X., Lu, M., and Wang, J. Higher-order interaction goes neural: A substructure assembling graph attention network for graph classification. *TKDE*, 2021.

Ge, C., Wang, X., Zhang, Z., Chen, H., Fan, J., Huang, L., Xue, H., and Zhu, W. Dynamic mixture of curriculum lora experts for continual multimodal instruction tuning. In *International Conference on Machine Learning*, 2025.

Gong, X., Yang, C., and Shi, C. Ma-gcl: Model augmentation tricks for graph contrastive learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37: 4284–4292, 06 2023. doi: 10.1609/aaai.v37i4.25547.

Hacohen, G. and Weinshall, D. On the power of curriculum learning in training deep networks. In *International conference on machine learning*, pp. 2535–2544. PMLR, 2019.

Hamilton, W. L., Ying, R., and Leskovec, J. Inductive representation learning on large graphs. NIPS'17, pp. 1025–1035, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Hassani, K. and Khasahmadi, A. H. Contrastive multi-view representation learning on graphs. In *International conference on machine learning*, pp. 4116–4126. PMLR, 2020.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738, 2020.

Hou, Z., Liu, X., Cen, Y., Dong, Y., Yang, H., Wang, C., and Tang, J. Graphmae: Self-supervised masked graph autoencoders. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 594–604, 2022.

Hou, Z., He, Y., Cen, Y., Liu, X., Dong, Y., Kharlamov, E., and Tang, J. Graphmae2: A decoding-enhanced masked self-supervised graph learner. In *Proceedings of the ACM Web Conference 2023*, WWW '23, pp. 737–746, 2023.

Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., and Leskovec, J. Open graph benchmark: Datasets for machine learning on graphs. In *NeurIPS*, 2020a.

Hu, W., Fey, M., Ren, H., Nakata, M., Dong, Y., and Leskovec, J. Ogb-lsc: A large-scale challenge for machine learning on graphs, 2021.

Hu, Z., Dong, Y., Wang, K., Chang, K.-W., and Sun, Y. Gpt-gnn: Generative pre-training of graph neural networks. In *KDD*, 2020b.

Huang, B., He, F., Wang, Q., Chen, H., Li, G., Feng, Z., Wang, X., and Zhu, W. Neighbor does matter: Curriculum global positive-negative sampling for vision-language pre-training. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 8005–8014, 2024.

Ji, H., Wang, X., Shi, C., Wang, B., and Yu, P. Heterogeneous graph propagation network. *TKDE*, 2021.

Jiang, B., Kloster, K., Gleich, D. F., and Gribskov, M. Aptrank: an adaptive pagerank model for protein function prediction on bi-relational graphs. *Bioinformatics*, 33 (12):1829–1836, 2017.

Jiang, W. and Luo, J. Graph neural network for traffic forecasting: A survey. *arXiv preprint arXiv:2101.11174*, 2021.

Karimi, F., Génois, M., Wagner, C., Singer, P., and Strohmaier, M. Homophily influences ranking of minorities in social networks. *Scientific Reports*, 8, 2018.

Kipf, T. N. and Welling, M. Variational graph auto-encoders. In *NeurIPS Workshop on Bayesian Deep Learning*, 2016.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

Kumar, M., Packer, B., and Koller, D. Self-paced learning for latent variable models. *Advances in neural information processing systems*, 23, 2010.

Lan, X., Yuan, Y., Chen, H., Wang, X., Jie, Z., Ma, L., Wang, Z., and Zhu, W. Curriculum multi-negative augmentation for debiased video grounding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

Lee, N., Lee, J., and Park, C. Augmentation-free self-supervised learning on graphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36:7372–7380, 06 2022. doi: 10.1609/aaai.v36i7.20700.

Li, H., Cui, P., Zang, C., Zhang, T., Zhu, W., and Lin, Y. Fates of microscopic social ecosystems: Keep alive or dead? In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 668–676, 2019a.

Li, H., Wang, X., Zhang, Z., Ma, J., Cui, P., and Zhu, W. Intention-aware sequential recommendation with structured intent transition. *IEEE Transactions on Knowledge and Data Engineering*, 34(11):5403–5414, 2021a.

Li, H., Wang, X., Zhang, Z., Yuan, Z., Li, H., and Zhu, W. Disentangled contrastive learning on graphs. *Advances in Neural Information Processing Systems*, 34, 2021b.

Li, H., Wang, X., Zhang, Z., and Zhu, W. Ood-gnn: Out-of-distribution generalized graph neural network. *IEEE Transactions on Knowledge and Data Engineering*, 2022a.

Li, H., Wang, X., Zhang, Z., and Zhu, W. Out-of-distribution generalization on graphs: A survey. *arXiv preprint arXiv:2202.07987*, 2022b.

Li, H., Zhang, Z., Wang, X., and Zhu, W. Disentangled graph contrastive learning with independence promotion. *IEEE Transactions on Knowledge and Data Engineering*, 2022c.

Li, H., Zhang, Z., Wang, X., and Zhu, W. Learning invariant graph representations for out-of-distribution generalization. In *Advances in Neural Information Processing Systems*, 2022d.

Li, H., Cao, J., Zhu, J., Luo, Q., He, S., and Wang, X. Augmentation-free graph contrastive learning of invariant-discriminative representations. *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–11, 2023a. doi: 10.1109/TNNLS.2023.3248871.

Li, H., Wang, X., and Zhu, W. Curriculum graph machine learning: A survey. *International Joint Conference on Artificial Intelligence*, 2023b.

Li, H., Zhang, Z., Wang, X., and Zhu, W. Invariant node representation learning under distribution shifts with multiple latent environments. *ACM Transactions on Information Systems*, 42(1):1–30, 2023c.

Li, H., Wang, X., Zhang, Z., Chen, H., Zhang, Z., and Zhu, W. Disentangled graph self-supervised learning for out-of-distribution generalization. In *International Conference on Machine Learning*, 2024a.

Li, H., Wang, X., Zhu, X., Wen, W., and Zhu, W. Disentangling invariant subgraph via variance contrastive estimation under distribution shifts. In *International Conference on Machine Learning*, 2025.

Li, J., Wu, R., Sun, W., Chen, L., Tian, S., Zhu, L., Meng, C., Zheng, Z., and Wang, W. What's behind the mask: Understanding masked graph modeling for graph autoencoders, 2023d.

Li, Q., Zhu, Z., and Tang, G. Alternating minimizations converge to second-order optimal solutions. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3935–3943. PMLR, 09–15 Jun 2019b.

Li, X., Ye, T., Shan, C., Li, D., and Gao, M. Seegera: Self-supervised semi-implicit graph variational auto-encoders with masking. *Proceedings of the ACM Web Conference 2023*, 2023e.

Li, X., Fan, Z., Huang, F., Hu, X., Deng, Y., Wang, L., and Zhao, X. Graph neural network with curriculum learning for imbalanced node classification. *Neurocomputing*, pp. 127229, 2024b.

Liu, Y., Jin, M., Pan, S., Zhou, C., Zheng, Y., Xia, F., and Philip, S. Y. Graph self-supervised learning: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 35(6):5879–5900, 2022.

Ma, J., Cui, P., Kuang, K., Wang, X., and Zhu, W. Disentangled graph convolutional networks. In *International conference on machine learning*, pp. 4212–4221. PMLR, 2019.

Mavromatis, C. and Karypis, G. Graph infoclust: Maximizing coarse-grain mutual information in graphs. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 541–553. Springer, 2021.

Miao, X., Zhang, W., Shao, Y., Cui, B., Chen, L., Zhang, C., and Jiang, J. Lasagne: A multi-layer graph convolutional network framework via node-aware deep architecture. *TKDE*, 2021.

Mo, Y., Peng, L., Xu, J., Shi, X., and Zhu, X. Simple unsupervised graph representation learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(7): 7797–7805, Jun. 2022.

Pan, S., Hu, R., Long, G., Jiang, J., Yao, L., and Zhang, C. Adversarially regularized graph autoencoder for graph embedding, 2019.

Pan, S., Hu, R., Fung, S.-F., Long, G., Jiang, J., and Zhang, C. Learning graph embedding with adversarial training methods. In *Proceedings of the International Conference on Learning Representations (ICLR 2020)*, 2020.

Peng, Z., Huang, W., Luo, M., Zheng, Q., Rong, Y., Xu, T., and Huang, J. Graph representation learning via graphical mutual information maximization. In *Proceedings of The Web Conference 2020*, WWW '20, pp. 259–270, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450370233.

Qin, Y., Wang, X., Zhang, Z., Chen, H., and Zhu, W. Multi-task graph neural architecture search with task-aware collaboration and curriculum. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023.

Sen, P., Namata, G., Bilgic, M., Getoor, L., Gallagher, B., and Eliassi-Rad, T. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008.

Shchur, O., Mumme, M., Bojchevski, A., and Günnemann, S. Pitfalls of graph neural network evaluation, 2019.

Shi, Y., Dong, Y., Tan, Q., Li, J., and Liu, N. Gigamae: Generalizable graph masked autoencoder via collaborative latent space reconstruction. *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 2023.

Sinha, S., Garg, A., and Larochelle, H. Curriculum by smoothing. *Advances in Neural Information Processing Systems*, 33:21653–21664, 2020.

Spitkovsky, V. I., Alshawi, H., and Jurafsky, D. From baby steps to leapfrog: How "less is more" in unsupervised dependency parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 751–759, 2010.

Sun, D., Cao, M., Ding, Z., and Luo, B. Graph contrastive learning with intrinsic augmentations. In Pan, L., Zhao, D., Li, L., and Lin, J. (eds.), *Bio-Inspired Computing: Theories and Applications*, pp. 343–357, Singapore, 2023a. Springer Nature Singapore. ISBN 978-981-99-1549-1.

Sun, Q., Zhang, W., and Lin, X. Progressive hard negative masking: From global uniformity to local tolerance. *IEEE Transactions on Knowledge and Data Engineering*, 35 (12):12932–12943, 2023b.

Tan, Q., Liu, N., Huang, X., Choi, S.-H., Li, L., Chen, R., and Hu, X. S2gae: Self-supervised graph autoencoders are generalizable learners with graph masking. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pp. 787–795, 2023.

Thakoor, S., Tallec, C., Azar, M. G., Azabou, M., Dyer, E. L., Munos, R., Veličković, P., and Valko, M. Large-scale representation learning on graphs via bootstrapping. *International Conference on Learning Representations*, 2022.

Tu, W., Liao, Q., Zhou, S., Peng, X., Ma, C., Liu, Z., Liu, X., Cai, Z., and He, K. Rare: Robust masked graph autoencoder. *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–14, 2023.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. Graph attention networks. In *ICLR*, 2018.

Veličković, P., Fedus, W., Hamilton, W. L., Liò, P., Bengio, Y., and Hjelm, R. D. Deep Graph Infomax. In *International Conference on Learning Representations*, 2019.

Wang, H., Wang, J., Wang, J., Zhao, M., Zhang, W., Zhang, F., Xie, X., and Guo, M. Graphgan: Graph representation learning with generative adversarial nets. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI 2018)*, pp. 2508–2515, 2018.

Wang, H., Zhang, J., Zhu, Q., and Huang, W. Augmentation-free graph contrastive learning with performance guarantee, 2022a.

Wang, J., Li, H., and Zhao, L. Accelerated gradient-free neural network training by multi-convex alternating optimization, 2022b.

Wang, L., Tao, X., Liu, Q., Wu, S., and Wang, L. Rethinking graph masked autoencoders through alignment and uniformity, 2024.

Wang, X., Chen, Y., and Zhu, W. A survey on curriculum learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):4555–4576, 2021a.

Wang, X., Pan, Z., Zhou, Y., Chen, H., Ge, C., and Zhu, W. Curriculum co-disentangled representation learning across multiple environments for social recommendation. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 36174–36192. PMLR, 23–29 Jul 2023.

Wang, Y., Wang, W., Liang, Y., Cai, Y., and Hooi, B. Curgraph: Curriculum learning for graph classification. In *Proceedings of the Web Conference 2021*, WWW '21, pp. 1238–1248, New York, NY, USA, 2021b. Association for Computing Machinery.

Wheeden, R., Wheeden, R. L., and Zygmund, A. *Measure and Integral: An Introduction to Real Analysis*. 1977.

Wu, Z., Ramsundar, B., Feinberg, E. N., Gomes, J., Geniesse, C., Pappu, A. S., Leswing, K., and Pande, V. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

Xu, D., Cheng, W., Luo, D., Chen, H., and Zhang, X. Infogcl: Information-aware graph contrastive learning. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 30414–30425. Curran Associates, Inc., 2021.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *ICLR*, 2019.

Yao, Y., Wang, X., Qin, Y., Zhang, Z., Zhu, W., and Mei, H. Data-augmented curriculum graph neural architecture search under distribution shifts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 16433–16441. AAAI Press, 2024.

Ye, Y. and Ji, S. Sparse graph attention networks. *TKDE*, 2021.

You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., and Shen, Y. Graph contrastive learning with augmentations. In *NeurIPS*, 2020.

Yu, M., Ding, Z., Yu, J., Zhang, W., Yang, M., and Zhao, M. Graph contrastive learning with adaptive augmentation for knowledge concept recommendation. In *2023 26th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pp. 1281–1286, 2023. doi: 10.1109/CSCWD57460.2023.10152806.

Zhang, H., Wu, Q., Yan, J., Wipf, D., and Yu, P. S. From canonical correlation analysis to self-supervised graph neural networks. *Advances in Neural Information Processing Systems*, 34:76–89, 2021a.

Zhang, M. and Chen, Y. Link prediction based on graph neural networks. In *NeurIPS*, pp. 5165–5175, 2018.

Zhang, Y., Zhu, H., Song, Z., Koniusz, P., and King, I. Costa: Covariance-preserving feature augmentation for graph contrastive learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, pp. 2524–2534, New York, NY, USA, 2022a. Association for Computing Machinery. ISBN 9781450393850.

Zhang, Y., Wang, X., Chen, H., Fan, J., Wen, W., Xue, H., Mei, H., and Zhu, W. Large language model with curriculum reasoning for visual concept recognition. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 6269–6280, 2024.

Zhang, Z., Cui, P., and Zhu, W. Deep learning on graphs: A survey. *TKDE*, 2020.

Zhang, Z., Cui, P., Pei, J., Wang, X., and Zhu, W. Eigen-gnn: A graph structure preserving plug-in for gnns. *TKDE*, 2021b.

Zhang, Z., Wang, X., Zhang, Z., Li, H., Qin, Z., and Zhu, W. Dynamic graph neural networks under spatio-temporal distribution shift. In *Thirty-Sixth Conference on Neural Information Processing Systems*, 2022b.

Zhang, Z., Zhang, Z., Wang, X., and Zhu, W. Learning to solve travelling salesman problem with hardness-adaptive curriculum. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 9136–9144, 2022c.

Zhang, Z., Wang, J., and Zhao, L. Relational curriculum learning for graph neural networks, 2023a.

Zhang, Z., Wang, X., Zhang, Z., Qin, Z., Wen, W., Xue, H., Li, H., and Zhu, W. Spectral invariant learning for dynamic graphs under distribution shifts. *Advances in Neural Information Processing Systems*, 36:6619–6633, 2023b.

Zhao, M., Wu, H., Niu, D., and Wang, X. Reinforced curriculum learning on pre-trained neural machine translation models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 9652–9659, 2020.

Zhao, P., Pan, Y., Li, X., Chen, X., Tsang, I. W., and Liao, L. Coarse-to-fine contrastive learning on graphs. *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–13, 2023. doi: 10.1109/TNNLS.2022.3228556.

Zhao, Z., Li, Y., Zou, Y., Tang, J., and Li, R. Masked graph autoencoder with non-discrete bandwidths. In *Proceedings of the ACM Web Conference 2024*, WWW '24, pp. 377–388, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400701719. doi: 10.1145/3589334.3645370.

Zhou, Y., Wang, X., Chen, H., Duan, X., Guan, C., and Zhu, W. Curriculum-nas: Curriculum weight-sharing neural architecture search. In *Proceedings of the 30th ACM International Conference on Multimedia*, pp. 6792–6801, 2022.

Zhou, Y., Wang, X., Chen, H., Duan, X., and Zhu, W. Intra- and inter-modal curriculum for multimodal learning. In *Proceedings of the 31st ACM International Conference on Multimedia*, pp. 3724–3735, 2023.

Zhou, Y., Pan, Z., Wang, X., Chen, H., Li, H., Huang, Y., Xiong, Z., Xiong, F., Xu, P., Zhu, W., et al. Curbench: curriculum learning benchmark. In *Forty-first International Conference on Machine Learning*, 2024.

Zhu, J., Zeng, W., Zhang, J., Tang, J., and Zhao, X. Cross-view graph contrastive learning with hypergraph. *Inf. Fusion*, 99(C), nov 2023. ISSN 1566-2535.

Zhu, Q., Du, B., and Yan, P. Self-supervised training of graph convolutional networks, 2020.

Zhu, Y., Xu, Y., Yu, F., Liu, Q., Wu, S., and Wang, L. Graph contrastive learning with adaptive augmentation. In *Proceedings of the Web Conference 2021 (WWW '21)*, pp. 2069–2080. ACM, 2021.

In the Appendix, we begin by summarizing the key notations in Section A. Then, we prove the convergence of our proposed method in Section B. We further elaborate on related works in Section C. Additional experimental details are provided in Section D. In Section E, we further analyze the time and space complexity to demonstrate the efficiency of our method. Finally, the visualization of the synthetic dataset is shown in Section F, and we present additional experimental results in Section G.

A. Notations

The key notations are summarized in Table 4.

Table 4. Notations.

Notation	Description
$\mathcal{V}$	Node set
$\mathcal{E}$	Edge set
$\mathcal{E}_{\text{mask}}$	Masked edges
$\mathbf{X}$	Node feature matrix
$\mathbf{A}$	Adjacency matrix
$\mathbf{A}_{\text{mask}}$	Adjacency matrix of the masked edges
$\mathbf{A}^{(t)}$	Adjacency matrix of curriculum selected edges at step t
$\mathbf{A}_{\text{DES}}$	Adjacency matrix of the difficult-based selected edges
$\mathbf{A}_{\text{RES}}$	Adjacency matrix of the random-based selected edges
$\mathbf{A}_{\text{re}}$	Adjacency matrix of reconstructed edges
$\hat{\mathbf{A}}$	Predicted adjacency matrix
$\mathcal{L}_{SSL}$	Loss function for self-supervised learning
$\mathcal{L}_{SPCL}$	Loss function for self-paced curriculum learning
$h_v^{(k)}$	Embedding of node v at the k-th layer
N_v	Set of direct neighbors of node v
COM$(\cdot)$	Combination function for updating node embeddings
AGG$(\cdot)$	Aggregation function for neighborhood information
N	Number of nodes
E	Number of edges
d	Dimensionality of embeddings
ENC$(\cdot)$	Encoder
DEC$(\cdot)$	Decoder
$h_{e_{v,u}}^{(k)}$	Edge representation between nodes v and u at layer k
K	Number of layers
I	Number of neighbors per node for aggregation
$\mathcal{K}$	Number of edges that need to be masked in the training process
$g(\cdot)$	Predicted probability of the presence of an existing edge
$\mathbf{S}^{(t)}$	Edge selection matrix at step t
λ	Regularization coefficient taking control of the number of edges to be selected
β	Balancing hyper-parameter
$\|\|\cdot\|\|$	l_2 norm
$\mathbf{w}$	GNN model parameter
γ	Training transition smoothing regularizer coefficient

B. Theoretical Analyses

Following the work (Zhang et al., 2023a), we have the following convergence guarantees for Algorithm 1:

B.1. Proof of Theorem 1

Proof. Assuming the second-order derivatives of $\mathcal{L}_{SSL}$ and $f(\mathbf{S};\lambda,\mathbf{A})$ are continuous, and given that the sequence $(\mathbf{w}^{(t)},\mathbf{S}^{(t)})$ is bounded, the second-order derivatives of $\mathcal{L}_{SSL}$ and $f(\mathbf{S};\lambda,\mathbf{A})$ are also bounded (Zhang et al., 2023a). This implies that

$$\max\left\{\left\|\nabla^2_{\mathbf{w}}\mathcal{L}_{SSL}(\mathbf{X},\mathbf{A}^{(t-1)};\mathbf{w})\right\|,\left\|\nabla^2_{\mathbf{S}}f(\mathbf{S};\lambda,\mathbf{A})\right\|\right\}\leq p, \tag{6}$$

where $p>0$ is a constant.

In addition, the second-order derivatives of the reconstruction term $\sum_{i,j} S_{ij} \left\| A_{ij} - \tilde{A}_{ij}^{(t)} \right\|$ are also bounded, which implies

$$\max \left\{ \left\| \nabla_{\mathbf{w}}^2 \sum_{i,j} S_{ij} \left\| A_{ij} - \tilde{A}_{ij}^{(t)} \right\| \right\|, \left\| \nabla_{\mathbf{S}}^2 \sum_{i,j} S_{ij} \left\| A_{ij} - \tilde{A}_{ij}^{(t)} \right\| \right\| \right\} \leq q,$$

where $q > 0$ is a constant and $\tilde{\mathbf{A}}$ is a function of $\mathbf{w}$.

As a result, the objective function $\mathcal{L}_{all}$ is bi-smooth, i.e.,

$$\max \left\{ \left\| \nabla_{\mathbf{w}}^2 \mathcal{L}_{all} \right\|, \left\| \nabla_{\mathbf{S}}^2 \mathcal{L}_{all} \right\| \right\} \leq p + q,$$

and $\mathcal{L}_{all}$ satisfies Assumption 4 in (Li et al., 2019b). Therefore, according to Theorem 10 in (Li et al., 2019b), the second-order derivative of $\mathcal{L}_{all}$ is continuous, and for any $\gamma > p + q$, any bounded sequence $(\mathbf{w}^{(t)}, \mathbf{S}^{(t)})$ generated by Algorithm 1 almost surely avoids convergence to a strict saddle point of $\mathcal{L}_{all}$ (Zhang et al., 2023a). □

B.2. Proof of Theorem 2

Proof. As established in the previous proof, the objective function $\mathcal{L}_{all}$ satisfies Assumption 4 in (Li et al., 2019b). Furthermore, since $\mathcal{L}_{SSL}(\mathbf{X}, \mathbf{A}^{(t-1)}; \mathbf{w})$, $f(\mathbf{S}; \lambda, \mathbf{A})$, and $\sum_{i,j} S_{ij} \|A_{ij} - \tilde{A}_{ij}^{(t)}\|$ all satisfy the Kurdyka–Łojasiewicz (KL) property, the composite objective $\mathcal{L}_{all}$ also satisfies the KL property. As shown previously, $\mathcal{L}_{all}$ is continuous. In addition, according to the results in (Wheeden et al., 1977), the continuous differentiability of $\mathcal{L}_{all}$ implies that its gradient is Lipschitz continuous. Therefore, $\mathcal{L}_{all}$ satisfies Assumption 1 in (Li et al., 2019b). Since $\mathcal{L}_{all}$ satisfies both Assumptions 1 and 4, we can invoke Corollary 3 in (Li et al., 2019b) to conclude that for any $\gamma > p + q$, any bounded sequence $(\mathbf{w}^{(t)}, \mathbf{S}^{(t)})$ generated by Algorithm 1 will almost surely converge to a second-order stationary point of $\mathcal{L}_{all}$ (Zhang et al., 2023a). □

C. Additional Related Works

Here we provide additional discussions on related work in the areas of graph neural networks (GNNs) and graph adversarial training, complementing our earlier review on graph self-supervised learning and curriculum learning.

Graph Neural Networks. Graph-structured data are ubiquitous across a wide range of real-world applications (Hu et al., 2020a; Li et al., 2019a; 2021a). The emergence of graph neural networks (GNNs) (Kipf & Welling, 2017; Veličković et al., 2018; Xu et al., 2019; Li et al., 2022b; Cai et al., 2024; Li et al., 2023c; Zhang et al., 2023b; Li et al., 2025; Chen et al., 2025) has led to significant advances in graph representation learning (Zhang et al., 2020), demonstrating strong performance in tasks such as node classification (Kipf & Welling, 2017), link prediction (Zhang & Chen, 2018), and graph-level classification (Xu et al., 2019). GNNs have also shown promise in high-impact domains including drug discovery (Wu et al., 2018), protein function prediction (Jiang et al., 2017), and traffic forecasting (Jiang & Luo, 2021). These models typically rely on neighborhood aggregation or message-passing mechanisms, where node representations are iteratively refined by aggregating information from local neighbors (Veličković et al., 2018; Xu et al., 2019). Despite their success, many state-of-the-art GNNs (Ma et al., 2019; Ji et al., 2021; Fan et al., 2021; Gao et al., 2021; Miao et al., 2021; Ye & Ji, 2021; Zhang et al., 2021b; 2022b; Li et al., 2022d;a) require end-to-end supervised training with task-specific labels, which are often scarce or expensive to obtain (Feng et al., 2023). In contrast, our proposed method adopts a generative self-supervised framework that reduces dependence on labeled data, making it more suitable for large-scale or label-deficient graph datasets.

Graph Adversarial Training. Graph adversarial training is a robust learning paradigm that enhances model generalization and stability by introducing adversarial perturbations or generating adversarial examples. These methods typically involve a generator–discriminator setup, where the generator attempts to craft perturbations or fake samples to deceive the discriminator, which in turn is trained to resist such attacks. For example, GraphGAN (Wang et al., 2018) introduces adversarial training into graph representation learning by generating synthetic links and learning to discriminate between real and fake connections. ARGA and ARVGA (Pan et al., 2020) enforce the latent representations to match a given prior distribution using adversarial regularization, followed by graph reconstruction. GCA (Zhu et al., 2021) leverages centrality-based graph augmentations to emphasize critical structures and adaptively perturb unimportant components. AUG-MAE (Wang et al., 2024) introduces an adversarial masking strategy to generate hard-to-align node features, thereby

improving contrastive alignment. Unlike these adversarial learning approaches, our curriculum-based masking strategy explicitly defines a progression from easy to hard reconstruction tasks. This structured learning path is fundamentally distinct from adversarial objectives and focuses on improving representation learning by aligning the training dynamics with task difficulty.

D. Additional Experimental Details

D.1. Dataset Statistics

We summarize the statistics of the real-world datasets used in our experiments in Table 5. For datasets from the OGB benchmark, we follow the standardized experimental protocol, which provides predefined train/validation/test splits. According to the official guidelines, validation labels are intended solely for hyperparameter tuning and are not permitted to be used during model training. Notably, the OGB guidelines specify an exception for the OGBL-collab dataset, where an alternative protocol allows the use of validation labels during training. However, for consistency and fairness across experiments, we adopt the stricter protocol for all three OGB link prediction datasets used in this study—OGBL-collab, OGBL-ddi, and OGBL-ppa—where validation labels are excluded from the training process.

Table 5. Summary of the dataset statistics.

Dataset	# Nodes	# Edges	# Features	Train/Val/Test	# Classes
Cora	2,708	5,429	1,433	85/5/10	7
Citeseer	3,312	4,660	3,703	85/5/10	6
Pubmed	19,717	44,338	500	85/5/10	3
Coauthor-CS	18,333	81,894	6,805	–	15
Coauthor-Physics	34,493	247,962	8,415	–	5
ogbn-arxiv	169,343	1,166,243	128	–	40
OGBL-ddi	4,267	1,334,889	–	80/10/10	–
OGBL-collab	235,868	1,285,465	128	92/4/4	–
OGBL-ppa	576,289	30,326,273	58	70/20/10	–

D.2. Dataset License

The datasets included in this work are publicly available as follows:

1. **Plantoid Datasets**: https://github.com/kimiyoung/planetoid/raw/master/data/ with MIT License.
2. **Coauthor Datasets**: https://github.com/shchur/gnn-benchmark/raw/master/data/npz/ with MIT License.
3. **Open Graph Benchmark (OGB)**: https://ogb.stanford.edu.docs/graphprop/ with MIT License.

D.3. Implementation Details

We implement our models using PyTorch and employ Stochastic Gradient Descent (SGD) as the optimizer. The number of training epochs is set to 400 for node classification tasks and 200 for link prediction tasks, with an early stopping patience of 50 steps.

Our model supports various message-passing GNNs, and we adopt GCN (Kipf & Welling, 2017) and GraphSage (Hamilton et al., 2017) as the primary backbones in our experiments. For large-scale datasets from the OGB benchmark—OGBN-arxiv, OGBN-ddi, OGBL-collab, and OGBL-ppa—we set the representation dimensionality d to 256 and use a 3-layer GNN. For all other datasets, we set $d = 128$ and use a 2-layer GNN.

The cross-correlation decoder is implemented as a two-layer multilayer perceptron (MLP) with ReLU activation, and its hidden dimension is selected from $\{128, 256, 512, 1024\}$. The values for *split ratio* and *mask ratio* are tuned within the ranges [0, 1] and [0.4, 1] (with a step size of 0.1), respectively. The dropout rate is chosen from $\{0.3, 0.4, 0.5, 0.6\}$.

The hyperparameter λ controls the number of edges selected during training. A larger λ promotes the selection of more edges for masking and reconstruction. To facilitate an easy-to-hard curriculum learning scheme, i.e., progressively increasing the difficulty of the training samples by masking more edges, the number of masked edges $\mathcal{K}$ should increase over training

steps. Accordingly, λ is designed to increase with the iteration step t, following the schedule below (Zhang et al., 2023a):

$$\lambda = \begin{cases} \frac{\lambda_{\text{initial}}}{T \cdot \lfloor \frac{2}{3} \rfloor + 1 - t} & \text{if } t < T \cdot \lfloor \frac{2}{3} \rfloor, \\ \lambda_{\text{initial}} & \text{otherwise,} \end{cases} \tag{7}$$

where T denotes the total number of training epochs, and t is the current training step. This scheduling rule can be adapted to support alternative pacing strategies if needed.

For node classification, we evaluate the learned node representations using a downstream linear SVM classifier. We report the average 10-fold cross-validation accuracy with standard deviation over three repeated runs. For link prediction, we randomly sample an equal number of negative edges as positive ones to compute the AUC score. Results are reported as the mean and standard deviation over five repeated runs. Unless otherwise specified, all remaining hyperparameters are kept consistent with the settings in S2GAE (Tan et al., 2023) to ensure a fair comparison.

D.4. Hardware and Software Configuration

We conduct the experiments with the following hardware and software configurations:

- Operating System: Ubuntu 20.04.6 LTS
- CPU: Intel(R) Xeon(R) Gold 6348 CPU@2.60GHz
- GPU: NVIDIA GeForce RTX 4090 GPU
- Software: Python 3.8.13; PyTorch 2.0.1; PyTorch Geometric 2.3.1.

E. Additional Complexity Analysis

E.1. Time Complexity Analysis

To further assess the efficiency of our method in practice, we compare its training time per epoch against the strong baseline S2GAE (Tan et al., 2023) under identical hyperparameter settings. We report the average training time and standard deviation per epoch in Table 6. The results demonstrate that our proposed Cur-MGAE model achieves superior efficiency compared to S2GAE.

Table 6. Empirical Time Comparisons.

	Link Prediction			Node Classification		
	Cora	Citeseer	OGBL-ppa	Cora	Citeseer	OGBN-arxiv
Cur-MGAE	**0.045 ± 0.003s**	**0.043 ± 0.004s**	79.117 ± 1.753s	**0.093 ± 0.020s**	**0.089 ± 0.015s**	**1.718 ± 0.593s**
S2GAE	0.048 ± 0.008s	0.045 ± 0.009s	**77.469 ± 1.224s**	0.100 ± 0.020s	0.098 ± 0.009s	2.600 ± 0.357s

E.2. Space Complexity Analysis

As for space complexity, our model adopts GCN and GraphSAGE as backbone architectures, whose space complexities are given by $O(N \times d + E + \sum_{l=1}^{K} d_{l-1} \times d_l + N \times \sum_{l=1}^{K} d_l)$ and $O(N \times d + E + N \times I^K + \sum_{l=1}^{K} d_{l-1} \times d_l + N \times \sum_{l=1}^{K} d_l)$, respectively. Here, N denotes the number of nodes, E the number of edges, d the input feature dimension, K the number of GNN layers, d_{l-1} and d_l the input and output dimensions at layer l, and I the number of neighbors per node used in aggregation. Beyond the GNN backbones, our model introduces a complexity-guided curriculum masking module and a self-paced mask scheduler, both of which incur an additional space complexity of $O(E)$. These modules are lightweight and do not increase the overall space complexity beyond the standard GNN frameworks. Therefore, the space complexity of our model remains comparable to that of existing methods.

To complement the theoretical analysis, we empirically compute the number of parameters of our proposed model and various baseline methods under a unified configuration, where the embedding dimension is set to 128. The results are presented in Table 7, with all parameter counts calculated on the Cora dataset. For GraphMAE, we follow the original setup by using a GAT backbone with 4 attention heads.

Table 7. Comparison of total number of parameters across models.

	GraphMAE (Hou et al., 2022)	MaskGAE (Li et al., 2023d)	S2GAE (Tan et al., 2023)	**Cur-MGAE**
Number of Parameters	419,253	266,370	282,369	291,871

We observe that, despite the introduction of relatively complex components, our method maintains a comparable number of parameters to those of the baseline models.

F. Synthetic Dataset Visualization

Following previous work (Karimi et al., 2018; Abu-El-Haija et al., 2019; Zhang et al., 2023a), we build the synthetic dataset shown in Figure 3 to test the curriculum schedule and performance of our proposed model. Each point represents a node, with its x and y coordinates sampled from overlapping multi-Gaussian distributions. The nodes are categorized into 10 classes based on their features, and the corresponding classes are indicated by different colors. The difficulty of each edge is determined based on the relationship between the labels of its incident nodes: edges connecting nodes with the same label are considered easy, edges connecting nodes with adjacent labels are medium, and edges connecting nodes with non-adjacent labels are categorized as hard.

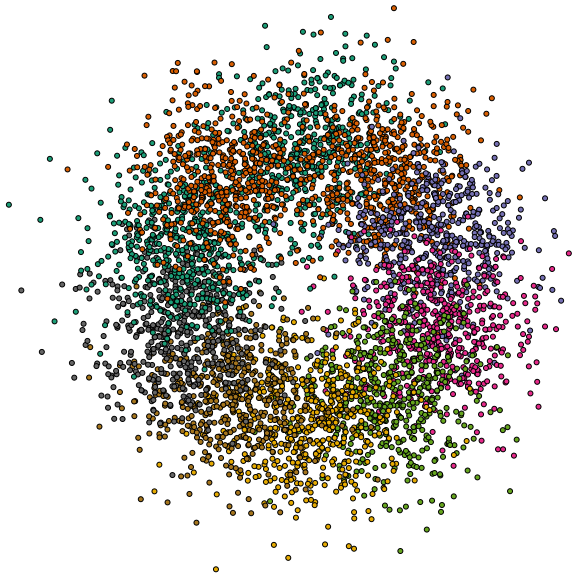

Figure 3. Visualization of the synthetic dataset. Each synthetic graph consists of 5,000 nodes, which are assigned to one of 10 classes based on their features. Edges are generated according to a probability that decreases with the cyclic distance between node labels. Specifically, the connection probability between nodes u and v is proportional to $e^{-|c_u - c_v|}$, where $|c_u - c_v|$ denotes the minimal cyclic distance between labels c_u and c_v in a circular label space.

G. Additional Experiments

G.1. Additional Comparisons with Contrastive Graph SSL Methods

Since the majority of the aforementioned baselines adopt generative paradigms, we additionally compare node classification performance of representative contrastive graph self-supervised methods to provide a more comprehensive evaluation, as shown in Table 8.

Table 8. Node classification accuracy (%) of our proposed method compared with contrastive baselines. Bold values indicate the best performance across all methods. '–' indicates that the method was not evaluated on the dataset due to unavailable code or omitted experiments in the original paper. Rank denotes the average performance rank across datasets. Our method achieves the highest accuracy across all datasets.

	Cora	Citeseer	Coauthor-Physics	Rank
GMI (Peng et al., 2020)	82.4 ± 0.6	71.7 ± 0.2	–	12.83
InfoGCL (Xu et al., 2021)	83.5 ± 0.3	73.5 ± 0.4	–	9.33
CCA-SSG (Zhang et al., 2021a)	84.2 ± 0.4	73.1 ± 0.3	95.38 ± 0.06	7.50
GCA-EV (Yu et al., 2023)	–	–	95.73 ± 0.03	10.67
AF-GCL (Wang et al., 2022a)	83.3 ± 0.1	72.1 ± 0.4	95.75 ± 0.15	7.17
AFGRL (Lee et al., 2022)	81.3 ± 0.2	68.7 ± 0.3	95.69 ± 0.10	7.67
SUGRL (Mo et al., 2022)	83.4 ± 0.5	73.0 ± 0.4	95.38 ± 0.11	6.33
C2F (Zhao et al., 2023)	–	–	94.09	8.33
COSTA-SV (Zhang et al., 2022a)	84.3 ± 0.3	72.8 ± 0.3	95.74 ± 0.02	5.50
COSTA-MV (Zhang et al., 2022a)	84.3 ± 0.2	72.9 ± 0.3	95.60 ± 0.02	5.00
IAG (Sun et al., 2023a)	86.1	73.6	–	4.00
S^3-CL (Ding et al., 2023)	84.5 ± 0.4	74.6 ± 0.4	–	3.33
H-GCL (Zhu et al., 2023)	84.8 ± 0.5	74.2 ± 0.3	–	2.83
IGCL (Li et al., 2023a)	79.3 ± 0.1	64.2 ± 0.1	95.85 ± 0.10	2.67
PHASES (Sun et al., 2023b)	–	–	95.82 ± 0.11	2.67
MA-GCL (Gong et al., 2023)	83.3 ± 0.4	73.6 ± 0.1	–	2.00
Cur-MGAE	**87.3 ± 0.6**	**74.7 ± 0.4**	**95.91 ± 0.05**	**1.00**

G.2. Experiment Results on Synthetic Datasets

We further conduct node classification and link prediction experiments on three synthetic datasets with varying levels of homophily, characterized by homophily coefficients of 0.1, 0.5, and 0.9 (denoted as Homo = 0.1, 0.5, and 0.9). Two representative baselines are adopted for comparison, and we report both node classification accuracy and link prediction AUC in the following table.

Table 9. Results on synthetic datasets.

	Homo=0.1		Homo=0.5		Homo=0.9	
	Link Pred.	Node Class.	Link Pred.	Node Class.	Link Pred.	Node Class.
Cur-MGAE	**52.73 ± 2.56**	**46.79 ± 0.11**	**57.20 ± 0.12**	**82.76 ± 0.67**	**80.97 ± 0.36**	**99.96 ± 0.02**
S2GAE(Tan et al., 2023)	50.04 ± 0.08	34.09 ± 0.13	51.28 ± 1.27	81.90 ± 0.17	80.48 ± 0.39	98.86 ± 0.00
BGRL(Thakoor et al., 2022)	51.84 ± 1.88	22.93 ± 1.38	53.46 ± 0.02	40.07 ± 4.40	80.68 ± 0.55	73.73 ± 0.68

The results show that our model consistently achieves the highest accuracy across all settings for both link prediction and node classification tasks. This demonstrates the effectiveness of the proposed structure-aware curriculum in learning powerful and informative node representations.

G.3. Hyperparameter Sensitivity

We investigated the sensitivity of some important hyperparameters of our method.

Effectiveness of the split ratio. The *split ratio* is a critical hyperparameter that introduces randomness into the edge selection process, thereby helping to mitigate overfitting. It determines the proportion of masked edges selected based on the difficulty-aware strategy. A lower *split ratio* implies that more edges are selected randomly, while a value of 1 indicates full reliance on difficulty-based selection. Specifically, when the *split ratio* is 0, edge masking is entirely random; when it is 1, edge selection is entirely difficulty-guided. As illustrated in Figure 4, an appropriately chosen *split ratio* strikes a balance between exploitation (leveraging informative edges) and exploration (incorporating diverse edge patterns), thereby enhancing overall model performance.

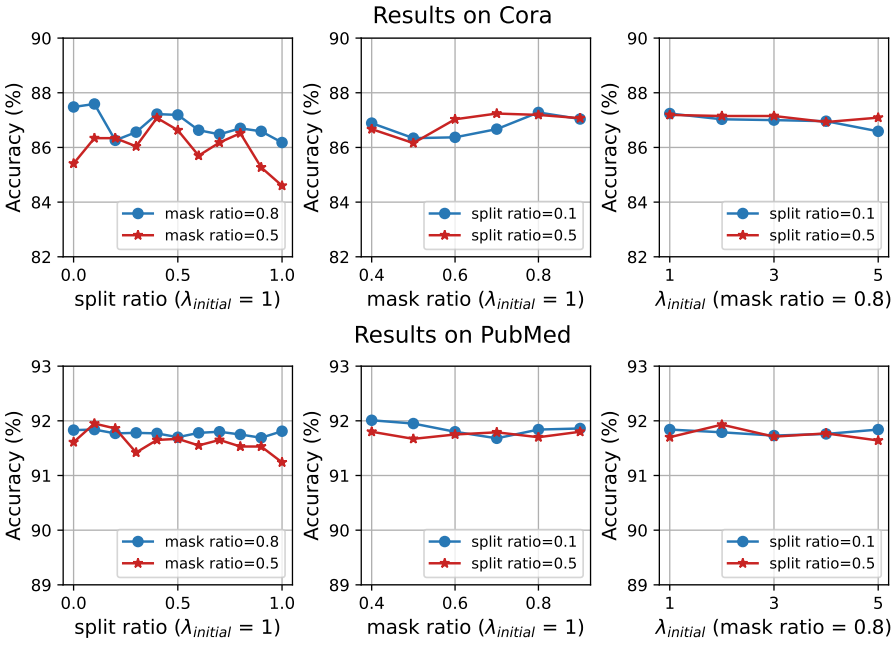

Figure 4. Impact of split ratio, mask ratio, and λ_{initial} on model performance, based on node classification results on Cora and Pubmed.

Effectiveness of the mask ratio. The *mask ratio* specifies the maximum proportion of edges that can be masked during training. A small *mask ratio* restricts the number of masked edges, limiting the model's exposure to sufficient learning signals and potentially trapping it in overly simple pretext tasks (e.g., predicting 10% of the edges using the remaining 90%). Conversely, an excessively high *mask ratio* (e.g., predicting 90% of the edges from only 10%) may lead to overly challenging tasks that degrade learning quality. Hence, selecting an appropriate *mask ratio* is essential for ensuring the training process remains both effective and stable.

Effectiveness of λ_{initial}. The hyperparameter λ_{initial} governs the pace of the structure-aware curriculum learning schedule. As shown in Figure 4, setting $\lambda_{\text{initial}} = 1$ yields optimal performance on the Cora dataset. A smaller λ_{initial} may lead to inadequate exposure to masked edges, limiting training diversity and reducing performance. On the other hand, a large λ_{initial} can result in the model being prematurely exposed to difficult tasks, which hampers early-stage learning. A well-chosen λ_{initial} enables a smooth progression from simple to complex tasks, avoiding overfitting to trivial edges while ensuring sufficient challenge during training. It reflects the significance of the structure-aware curriculum learning strategy of the proposed method.