HC-GAE: The Hierarchical Cluster-based Graph Auto-Encoder for Graph Representation Learning

Lu Bai[1,2], **Zhuo Xu**[1], **Lixin Cui**[3,*], **Ming Li**[4,5], **Yue Wang**[3], **Edwin R. Hancock**[6]
[1]School of Artificial Intelligence, Beijing Normal University, Beijing 100875, China
[2]Engineering Research Center of Intelligent Technology and Educational Application, Ministry of Education, Beijing Normal University, Beijing 100875, China
[3]School of Information, Central University of Finance and Economics, Beijing 100081, China
[4]Zhejiang Institute of Optoelectronicss, Jinhua 321004, China
[5]Zhejiang Key Laboratory of Intelligent Education Technology and Application, Zhejiang Normal University, Jinhua 321004, China
[6]Department of Computer Science, University of York, York YO10 5GH, United Kingdom

Abstract

Graph Auto-Encoders (GAEs) are powerful tools for graph representation learning. In this paper, we develop a novel Hierarchical Cluster-based GAE (HC-GAE), that can learn effective structural characteristics for graph data analysis. To this end, during the encoding process, we commence by utilizing the hard node assignment to decompose a sample graph into a family of separated subgraphs. We compress each subgraph into a coarsened node, transforming the original graph into a coarsened graph. On the other hand, during the decoding process, we adopt the soft node assignment to reconstruct the original graph structure by expanding the coarsened nodes. By hierarchically performing the above compressing procedure during the decoding process as well as the expanding procedure during the decoding process, the proposed HC-GAE can effectively extract bidirectionally hierarchical structural features of the original sample graph. Furthermore, we re-design the loss function that can integrate the information from either the encoder or the decoder. Since the associated graph convolution operation of the proposed HC-GAE is restricted in each individual separated subgraph and cannot propagate the node information between different subgraphs, the proposed HC-GAE can significantly reduce the over-smoothing problem arising in the classical convolution-based GAEs. The proposed HC-GAE can generate effective representations for either node classification or graph classification, and the experiments demonstrate the effectiveness on real-world datasets.

1 Introduction

In real-world applications, graph structure data has been widely used for characterizing pairwise relationships among the components of complex systems. With the recent rapid development of deep learning, the graph representation learning approaches relying on neural networks are introduced for the analysis of various graph data, e.g., social networks [12], transportation networks [20], protein compounds [34], 3D shape [2, 5], etc. One challenging arising in these studies is that the graph data has a nonlinear structure defined in an irregular non-Euclidean space, and it is hard to directly employ traditional neural networks to learn graph representations.

To overcome the above problem, there have been increasing interests to further generalize traditional neural networks, especially the Convolutional Neural Network (CNN) [18], for the irregular graph

[*]Corresponding Author: cuilixin@cufe.edu.cn

data. These are the so-called convolution-based Graph Neural Networks (GNNs) [4] and their related approaches [13] proposed for graph-based tasks, utilizing both graph features and topologies. For instance, the Higher-order Graph Convolutional Network (HiGCN) [11] has been developed based on the higher-order interactions to recognize intrinsic features across varying topological scales. Its effective expressiveness makes it capable for various graph-based tasks. The DeepRank-GNN [26] has been proposed by combining the rotation-invariant graphs and the GNN to represent protein-protein complexes. Because the GNN models can extract graph representations with more semantic learning under supervised conditions, researchers have focused more on seeking a self-supervised framework associated with the GNNs to accomplish representation learning.

As a typical framework of representation learning, the classical Auto-Encoder [22] has been proposed to extract impressive results by reconstructing the input information. Especially, the Graph Auto-Encoder (GAE) [30] associated with the GNN model has further generalized the reconstruction ability for graph structures [15, 19]. Due to the extensibility, the GAEs have been developed as a family of classical models for self-supervised representation learning, and there are adequate derivation models belonging to GAEs. For instance, the Self-Supervised Masked Graph Autoencoders (GraphMAE) [9] focusing on the feature reconstruction adopts a masking strategy and the scaled cosine error in the training model. Compared to the traditional GAE approach like VGAE [29], its decoder is retrofitted with the GNN and the re-masking operation. Based on the GraphMAE, the S2GAE [28] continues to adopt the masking strategy to improve the auto-encoder framework. To generate the cross-representation, the decoder is designed to capture the cross-correlation of nodes.

Challenges. Although the classical GAE-based methods achieve the effective performance for graph representation learning, they still have some significant challenging problems summarized as follows.

(a) The limitation for multiple downstream tasks: Generally, the representations extracted from the GAEs can be divided into several categories, including the node-level representations for node classification, the graph-level representations for graph classification, etc. Specifically, it is difficult for the GAEs to generate universal representations for multiple downstream tasks simultaneously. This is because the GAEs tend to over-emphasize the node features. For instance, the GraphMAE [9] focuses more on the node feature reconstruction, resulting in **topological missing** and weakening the the structure information reconstruction. This is harmful for the graph-level representation learning.

(b) The over-smoothing problem: The GAEs are usually proposed based on the GNNS, thus both the decoder and encoder modules of the GAEs are defined associated with a number of stacked graph convolution operations, that rely on the node information propagation between adjacent nodes. When the GAE becomes deeper, the node features tend to be similar or indistinguishable after multiple rounds of information passing [21], resulting in the notorious over-smoothing problem [6] and influence the performance of the GAEs.

Contributions. The aim of this paper is to overcome the above challenging problems by proposing a novel HC-GAE model. Overall, the main contributions are threefold.

First, we propose a novel Hierarchical Cluster-based GAE (**HC-GAE**) for graph representation learning. Specifically, for the encoding process, we adopt the hard node assignment to decompose a sample graph into a family of separated subgraphs. We perform the graph convolution operation for each subgraph to further extract node features and compress the nodes belonging to each subgraph into a coarsened node, transforming the original graph into a coarsened graph. Since the separated subgraphs are isolated from each other, the convolution operation cannot propagate the node information between different subgraphs. The proposed HC-GAE can in turn reduce the over-smoothing problem arising in the classical GAEs. Moreover, since the effect of the graph structure perturbation is limited within each subgraph, the required convolution operation performed on each subgraph can strengthen the robustness of the encoder for the proposed HC-GAE. As a result, the outputs of the encoder can be employed as the graph-level representations. On the other hand, for the decoding process, we adopt the soft node assignment to reconstruct the original graph structure by expanding each coarsened node into all retrieved nodes probabilistically. Thus, the outputs of the decoder can be employed as the node-level representations. Since the HC-GAE is defined by hierarchically performing the above compressing procedure during the decoding process as well as the expanding procedure during the decoding process, the proposed HC-GAE can effectively extract bidirectionally hierarchical structural features of the original sample graph, resulting in effective hierarchical graph-level and node-level representations for either graph classification or node classification.

Second, we propose a new loss function for training the proposed HC-GAE model. For calculating the complete loss value, we integrate the local loss from the subgraphs in the encoding operation and the global loss from the reconstructed graphs in the decoding operation. The global loss can capture the information from both the structure and the feature reconstruction processes. The combination of these two pretext tasks broadens the strict requirement causing the topological closeness. In addition, to avoid the over-fitting problem, we add the local loss as the regularization in our loss function.

Third, we empirically evaluate the performance of the proposed HC-GAE model on both node and graph classification tasks, demonstrating the effectiveness of the proposed model.

This paper organizes as follows. Section 2 reviews some related works. Section 3 gives the definition of the proposed model. Section 4 provides experimental evaluations. Section 5 gives conclusions.

2 Related Works

2.1 The Graph Neural Network

GNNs are widely employed for many real-world applications [16, 31, 35], and have achieved a prominent success. The input data of GNNs is a graph, that is a kind of non-Euclidean data and contains nodes and edges. With the complex structures of the graphs, GNNs aim to leverage the information passing mechanism among nodes for graph embedding learning. The procedure of the information passing can be divided into aggregating, combining and readout. Specifically, given an input sample graph $G(V, E)$ with the node set V ($|V| = n$) and the edge set E, the node information is represented as the feature matrix $X \in \mathbb{R}^{n\times d}$ with d features, and the structure information is represented as the adjacent matrix $A \in \{0, 1\}^{n\times n}$. The GNN for each layer is defined as

$$Z_G = \mathrm{GNN}(X, A; \Theta), \tag{1}$$

where Θ is the parameter set, and Z_G is the graph embedding result for downstream tasks.

The Graph Convolutional Network (GCN) [13] is a typical derivative model of GNNs, and generalizes the classical Convolutional Neural Network (CNN) [18] from the regular data to the irregular graph-structured data. One typical GCN model can be defined as the following layer-wise scheme, i.e.,

$$H^{(l+1)} = \mathrm{ReLU}(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}), \tag{2}$$

where $H^{(l)} \in \mathbb{R}^{n\times d}$ is the hidden embedding matrix for the l-th layer, $W^l \in \mathbb{R}^{d\times d}$ is the trainable parameter matrix for the l-th layer, $\tilde{A} = A + I$ is the adjacency matrix associated with the self loop, $\tilde{D} = \sum_j \tilde{A}_{ij}$ is the degree matrix of $\tilde{A}$, and $H^{(l+1)}$ is the embedding matrix extracted for the next $l + 1$-th layer. The GCN model can effectively capture the global structural representations through the multiple stacked convolution layers defined by Eq.(2). Unfortunately, since the convolution operation depends on the node information propagation between adjacent nodes. The GNN suffers from
similar when the GCN becomes deeper, e.g., the GNN with more than 5 convolution layers.

2.2 The Graph Auto-Encoder

The GAE is a powerful self-supervised framework for graph representation learning. Typical instances of GAEs include the DeepWalk [25] and the Node2Vec [7], where the encoders usually play an important role for learning the latent representations of vertices. By associating with the convolution operations of GNNs, the encoders of GAEs are able to cope with the non-Euclidean data [24]. As a self-supervised learning model, the aim of GAEs during the training process is the graph reconstruction [17], that can be categorized into the fine-grained and the coarse-grained targets.

Specifically, the fine-grained target contains either nodes or edges. For instance, the Variational Graph Auto-Encoder (VGAE) model [29] adopts two stages, i.e. the encoder and the decoder, to accomplish the representation learning. For an input graph $G(V, E)$, the goal of VGAE is to embed the graph through the encoder function $f : V \times E \to Z \in \mathbb{R}^{n\times d}$, that maps the original node feature matrix of the node set V to the embedding matrix Z. Then, the decoder reconstructs the graph through the function $g : Z \to E'$, where E' is the reconstructed edge set for the original edge set E. During the encoding and decoding processes, the VGAE obtains the conditional probability distributions

$q(Z \mid V, E)$ and $p(E' \mid Z)$ from the encoder and the decoder respectively, and the loss function is

$$\mathcal{L} = \mathrm{KL}[q(Z \mid V, E) \| p(Z)] - \mathbb{E}_{q(Z|V,E)}[\log p(E' \mid Z)], \tag{3}$$

where $\mathrm{KL}[\cdot]$ is the Kullback-Leibler divergence, $\mathbb{E}$ is the expection, and $p(Z)$ is the Gaussian prior.

On the other hand, the coarse-grained target contains the subgraph or the path structure. For example, the Heterogeneous Graph Masked Autoencoder (HGMA) [30] adopts the dynamic masking strategy to mask the nodes and the edges on the paths, and then reconstruct the path. Similar approaches also include the Masked Graph Autoencoder (MaskGAE) [14], that aims to reconstruct the masked edges and the node degrees jointly. Recently, some researchers have noted that the combination of the Graph Contrastive Learning (GCL) [38] and the GAE framework can further capture the complex interdependency residing on graphs. Under this scenario, the Self-supervised Learning for Graph Anomaly Detection (SL-GAD) [41] has been developed to obtain double subgraphs through the graph view sampling, and then respectively reconstructs them in two decoders for constrastive learning.

Although the above GAE methods significantly improve the performance of the graph representation learning, they still suffer from some common drawbacks. First, these GAE methods usually have superior performance for node classification, but have poor performance for graph classification. This is due to the fact that these methods only focus on the node feature reconstruction and ignore the global graph structure reconstruction, resulting in topological missing. Thus, these methods tends to be elusive for multiple downstream tasks [17]. Second, since the decoder and the encoder of these GAE methods usually employ the GNN models (i.e., the associated graph convolution operation) to extract the node feature information. The over-smoothing problem of GNNs mentioned in Section 2.1 also affects the feature learning for these GAE-based methods. Moreover, when the perturbation of the graph structure is conducted, the noise can also be propagated to the neighbor nodes through the edges. After several rounds of information passing, the generated graph representations may be noisy.

3 The Methodology

To overcome the aforementioned challenges, we propose a novel (**HC-GAE**) model to learn effective graph representations. The framework of the proposed model is shown in Figure 1. Similar to the classical GAEs, the HC-GAE model has two stages, including the encoder as well as the decoder. During the encoding process, the encoder can layer-wisely compress each sample graph into a series of coarsened graphs, and the results of the encoder can be employed as the graph-level representations for graph classification. During the decoding process, the decoder can reconstruct the structure of the original sample graph, and outputs the node-level representations for node classification. In the following subsections, we commence by introducing the encoder and decoder of the proposed HC-GAE model. Moreover, we define a family of loss function for the proposed HC-GAE model. Finally, we theoretically analyze the effectiveness of the proposed model.

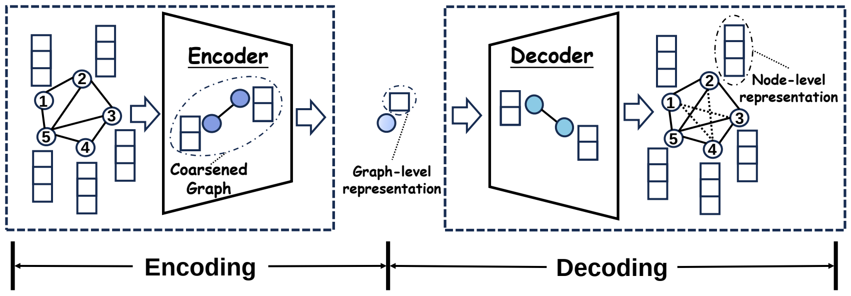

Figure 1: The architecture of our proposed HC-GAE model.

3.1 The GNN Encoder associated with the Hard Node Assignment

The first module of the proposed HC-GAE model is the encoder, that consists of multiple GNN layers and adopts the hierarchical computational architecture to compress the original input graph into a series coarsened graphs with shrinking sizes. Details of this GNN-based encoder are shown in Figure 2, where each layer includes the node assignment and coarsening processes. The first process is to generate separated subgraphs from the original graph, and the second process compresses the subgraphs into coarsened nodes of a coarsened graph.

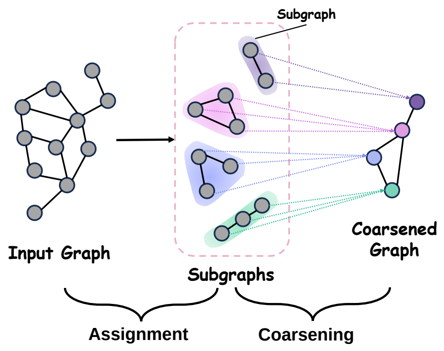

Figure 2: The computational architecture for our proposed layer in the GNN encoder.

Hard Assignment. For each l-th layer of the encoder, the input graph is denoted as $G^{(l)} = (X^{(l)}, A^{(l)})$, where $X^{(l)} \in \mathbb{R}^{n_{(l)} \times d_{(l)}}$ is the node feature matrix and $A^{(l)} \in \mathbb{R}^{n_{(l)} \times n_{(l)}}$ is the adjacent matrix. The number of nodes for $G^{(l)}$ is $n_{(l)}$, and each node has $d_{(l)}$ features. Note that, $G^{(l)}$ can be either the original input sample graph when $l = 1$ or the coarsened graph when $l > 1$. For the assignment process, the graph $G^{(l)}$ is decomposed into a number of separated subgraphs, by assigning the $n_{(l)}$ nodes into $n_{(l+1)}$ clusters through the hard node assignment. To achieve this, we commence by computing the soft node assignment matrix S_{soft} as

$$S_{\text{soft}} = \begin{cases} \text{softmax}(\text{GNN}(X^{(l)}, A^{(l)})) & \text{if } l = 1 \\ \text{softmax}(X^{(l)}) & \text{if } l > 1 \end{cases}, \tag{4}$$

where $S_{\text{soft}} \in \mathbb{R}^{n_{(l)} \times n_{(l+1)}}$ and $n_{(l+1)} < n_{(l)}$. To guarantee the separability of the resulting subgraphs, we need to assign each node into an unique cluster through a hard node assignment matrix $S^{(l)} \in \{0, 1\}^{n_{(l)} \times n_{(l+1)}}$, where each (i, j)-th entry of $S^{(l)}$ can be computed through S_{soft}, i.e.,

$$S^{(l)}(i, j) = \begin{cases} 1 & \text{if } S_{\text{soft}}(i, j) = \max\limits_{\forall j \in n_{l+1}} [S_{\text{soft}}(i, :)] \\ 0 & \text{otherwise} \end{cases}. \tag{5}$$

Clearly, each i-th row of the hard assignment matrix $S^{(l)}$ selects the maximum element as 1 and the remaining elements as 0. As a result, the i-th node is only assigned to the j-th separated subgraph $G_j^{(l)}(V_j^{(l)}, E_j^{(l)})$, where $V_j^{(l)}$ consists of the nodes belonging to the j-th cluster and $E_j^{(l)}$ remains the original edge connections between the nodes in $V_j^{(l)}$ from the original input graph $G^{(l)}$.

Coarsening. The coarsening process compresses the separated subgraphs into coarsened nodes of the resulting coarsened graph. Given the feature and adjacency matrices $X_j^{(l)}$ and $A_j^{(l)}$ of the subgraph $G_j^{(l)}$, we first adopt a local graph convolution operation to extract the local structural information as

$$Z_j^{(l)} = A_j^{(l)} X_j^{(l)} W_j^{(l)}, \tag{6}$$

where $W_j^{(l)} \in \mathbb{R}^{d_{(l)} \times d_{(l+1)}}$ ($d_{(l)} > d_{(l+1)}$) is the trainable weight matrix of the l-th layer, and $Z_j^{(l)} \in \mathbb{R}^{|V_j^{(l)}| \times d_{(l+1)}}$ is the resulting local structural representation of $G_j^{(l)}$. To compress each subgraph $G_j^{(l)}$ into a a coarsened node, we define a mapping vector $\mathbf{s}_j^l$ as

$$\mathbf{s}_j^{(l)} = \text{softmax}(A_j^{(l)} X_j^{(l)} D_j^{(l)}), \tag{7}$$

where $D_j^{(l)} \in \mathbb{R}^{d_{(l)} \times 1}$ is a trainable vector, and will play an important role in compressing each subgraph $G_j^{(l)}$ into the node of the coarsened graph. After several local graph convolution operations on the separated subgraphs in the l-th layer, we aggregate these local information to further generate the coarsened graph, as the input graph $G^{(l+1)}$ for the next $l+1$-th layer. To this end, for the original input graph $G^{(l)}$, we collect the feature matrices $Z_j^{(l)}$ of all j-th subgraphs $G_j^{(l)}$ as the extracted node feature matrix $Z^{(l)} \in \mathbb{R}^{n_{(l)} \times d_{(l+1)}}$, whose node sequences follow the original graph $G^{(l)}$, i.e., each vertex feature vector of $Z^{(l)}$ is equal to that of the corresponding node in $G_j^{(l)}$. This is because each node of $G_j^{(l)}$ essentially corresponds to an unique original node of $G^{(l)}$. Given the hard assignment matrix $S^{(l)}$ defined by Eq.(5) and the mapping vector $\mathbf{s}_j^{(l)}$ defined by Eq.(7), we compute the feature matrix $X^{(l+1)}$ and the adjacent matrix $A^{(l+1)}$ of the resulting coarsened graph $G^{(l+1)}$ as

$$X^{(l+1)} = \mathrm{Reorder}[\mathop{\Big\|}_{j=1}^{n_{l+1}} \mathbf{s}_j^{(l)^\top}]Z^{(l)}, \tag{8}$$

and

$$A^{(l+1)} = S^{(l)^\top} A^{(l)} S^{(l)}, \tag{9}$$

where $\|$ is a concatenation operation, and the function $\mathrm{Reorder}$ reorders the sequences of $[\cdot]$ to follow the same node sequence of $G_j^{(l)}$. For the encoder, we set the greatest layer number as L, and the resulting coarsened graph $G^{(L+1)} = (X^{(L+1)}, A^{(L+1)})$ is the input graph for the decoder. Furthermore, we adopt a non-parameterized readout function (e.g., the MaxPooling and MeanPooling) to further embed $X^{(L+1)}$ as ***the graph-level representation for graph classification***.

3.2 The GNN Decoder associated with the Soft Node Assignment

The second module of the proposed HC-GAE model is the decoder, that aims to reconstruct the structure of the original input sample graph, by expanding the each coarsened node into retrieved nodes probabilistically through the soft node assignment. Similar to the encoder, the decoder also consists of multiple GNN-based layers. Details of each decoder layer are shown in Figure 3, where the input is the retrieved graph and the output is the reconstructed graph.

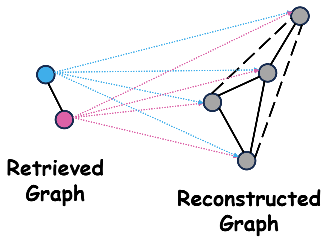

Figure 3: The illustration of our proposed layer in the GNN decoder.

Reconstruction. For each l'-th layer of the decoder, the input retrieved graph is denoted as $G'^{(l')} = (X'^{(l')}, A'^{(l')})$, where $X'^{(l')} \in \mathbb{R}^{n_{(l')} \times d_{(l')}}$ is the node feature matrix and $A'^{(l')} \in \mathbb{R}^{n_{(l')} \times n_{(l')}}$ is the adjacent matrix. Note that, when $l' = 1$, $G'^{(l')}$ is essentially the resulting coarsened graph $G^{(L)}$ of the encoder. For the reconstruction process with $G'^{(l')}$, we commence by computing the soft node assignment matrix $\bar{S}^{(l')} \in \mathbb{R}^{n_{(l')} \times n_{(l'+1)}}$ (i.e., the learnable node re-assignment matrix) as

$$\bar{S}^{(l')} = \mathrm{softmax}(\mathrm{GNN}_{l',\mathrm{re}}(X'^{(l')}, A'^{(l')})). \tag{10}$$

Moreover, we further compute the node feature embedding of $G'^{(l')}$ as

$$\bar{Z}^{(l')} = \mathrm{GNN}_{l',\mathrm{emb}}(X'^{(l')}, A'^{(l')}). \tag{11}$$

Note that, the associated graph convolution operations $\mathrm{GNN}_{l',\mathrm{re}}$ and $\mathrm{GNN}_{l',\mathrm{emb}}$ for Eq.(10) and Eq.(11) do not share the parameters. Specifically, the operation $\mathrm{GNN}_{l',\mathrm{re}}$ generates a probabilistic distribution to expand the nodes of $G'^{(l')}$ to the corresponding nodes of the resulting reconstructed graph $G'^{(l'+1)}$, and the operation $\mathrm{GNN}_{l',\mathrm{emb}}$ extracts the new embeddings for $G'^{(l')}$. With $\bar{S}^{(l')}$ and $\bar{Z}^{(l')}$ to hand, we compute the node feature matrix $X'^{(l'+1)} \in \mathbb{R}^{n_{(l'+1)} \times d_{(l'+1)}}$ and the adjacency matrix $A'^{(l'+1)} \in \mathbb{R}^{n_{(l'+1)} \times n_{(l'+1)}}$ for the reconstructed graph $G'^{(l'+1)}$ as

$$X'^{(l'+1)} = \bar{S}^{(l')^\top} \bar{Z}^{(l')}, \tag{12}$$

and

$$A'^{(l'+1)} = \bar{S}^{(l')^\top} A'^{(l')} \bar{S}^{(l')}. \tag{13}$$

We employ $G'^{(l'+1)}$ as the input retrieved graph for the next $l'+1$-th layer, and set the greatest layer number as L' (L' equals to L of the encoder). The resulting graph $G'^{(L'+1)} = (X'^{(L'+1)}, A'^{(L'+1)})$ is the output of our HC-GAE, and $X'^{(L'+1)}$ is ***the node-level representations for node classification***.

3.3 The Loss Function

The proposed HC-GAE model consists of the encoder focusing on compressing the graph structural information and the decoder aiming at reconstructing the original graph structure. Therefore, the loss function $\mathcal{L}_{\mathrm{HC-GAE}}$ of HC-GAE includes two individual parts, and is defined as

$$\begin{aligned}
\mathcal{L}_{\mathrm{local}} &= \sum_{l=1}^{L} \sum_{j=1}^{n_{(l+1)}} \mathrm{KL}[q(Z_j^{(l)} \mid X_j^{(l)}, A_j^{(l)}) \| p(Z^{(l)})], \\
\mathcal{L}_{\mathrm{global}} &= -\sum_{l=1}^{L} \mathbb{E}_{q(X^{(L)},A^{(L)})|X^{(l)},A^{(l)})}[\log p(X'^{(L-l+2)}, A'^{(L-l+2)} \mid X^{(L)}, A^{(L)})], \\
\mathcal{L}_{\mathrm{HC-GAE}} &= \mathcal{L}_{\mathrm{local}} + \mathcal{L}_{\mathrm{global}},
\end{aligned} \tag{14}$$

where $\mathcal{L}_{\mathrm{local}}$ is the local loss, $\mathcal{L}_{\mathrm{global}}$ is the global loss, and $p(Z^{(l)})$ is the Gaussian prior for the l-th layer. Specifically, $\mathcal{L}_{\mathrm{local}}$ aims at training the subgraph generation, that reserves the local information and avoids the over-smoothing problem for the GNN-based encoder. $\mathcal{L}_{\mathrm{global}}$ focuses on reconstructing the graph features and structures. The combination of $\mathcal{L}_{\mathrm{local}}$ and $\mathcal{L}_{\mathrm{global}}$ broadens the reconstruction requirement for multiple downstream tasks (i.e., the node and graph classification), since $\mathcal{L}_{\mathrm{local}}$ is a regularization for the loss and can strengthen the graph representations associated with the additional local information for the proposed HC-GAE model. Note that, for the node classification, we only need to reconstruct the structure information (i.e., the adjacency matrix $A^{(\cdot)}$) rather than the node feature information (i.e., the feature matrix $X^{(\cdot)}$) for the loss function $\mathcal{L}_{\mathrm{HC-GAE}}$.

3.4 Discussions

(a) Why is the proposed HC-GAE model effective for multiple downstream tasks?

As we mentioned in Section 1 and 2.2, most existing GAE methods tend to over-emphasize the graph feature reconstruction, not only ignoring the topological structure information but also aggravating the over-fitting problem for the graph features [32]. Thus, these GAE methods are usually employed for node classification, and have poor performance for graph classification. To overcome this shortcoming, the proposed HC-GAE model adopt the following schemes. **First**, the HC-GAE adopts the node assignment strategy to improve the encoding and decoding performance. Specifically, for the encoder, we utilize the hard node assignment to decompose the input graph into separated subgraphs to extract local structural information, and this can reduce the redundancy for the graph-level representation learning. On the other hand, for the decoder, we use the soft node assignment to reconstruct the original graph structure, and employ the associated node features of the reconstructed graph as the node-level representations. The combination of these assignment strategies guarantees that the HC-GAE can effectively learn multi-level representations for various downstream tasks. **Second**, the loss function of the HC-GAE consists of both the local and global loss. The global loss $\mathcal{L}_{\mathrm{global}}$ can either reconstruct the graph features or the topological structures, reducing the over-emphasizing on graph features. The local loss $\mathcal{L}_{\mathrm{local}}$ not only plays a role of regularization in $\mathcal{L}_{\mathrm{HC-GAE}}$, but also captures the local information from the separated subgraphs, improving the generalization of the graph representations through the local information [39].

(b) Why does the proposed HC-GAE model reduce the over-smoothing problem?

Unlike the hierarchical-based GNN methods [29, 37], during the encoding process, we perform the local graph convolution operation within each individual separated subgraph. Thus, the information can not be propagated to other subgraphs, significantly reducing the over-smoothing problem.

4 Experiments

This section empirically evaluates the performance of the proposed HC-GAE[2] for both node and graph classification on benchmark datasets, whose detailed statistical information are shown in Table 1 and Table 2. Specifically, for the HC-GAE, we select the Adam optimizer to train the parameters, and set the epoch, hidden dimension and dropout as 50, 128 and 0.5, respectively. For the encoder and

[2]The source code is available on `https://github.com/JonathanGXu/HC-GAE`

Table 1: Datasets for node classification

Datasets	Cora	CiteSeer	PubMed	Computers	CS
Nodes	2708	3312	19717	13752	18333
Edges	5429	4660	44338	245861	81894
Features	1433	3703	500	767	6805
Classes	7	6	3	10	15

Table 2: Datasets for graph classification

Datasets	IMDB-B	IMDB-M	PROTEINS	COLLAB	MUTAG
Graphs	1000	1500	1113	5000	188
Nodes(mean)	19.77	13	39.06	74.49	17.93
Edges(mean)	96.53	65.94	72.82	2457.78	19.79
Classes	2	3	2	3	1

decoder of the HC-GAE, we set their greatest layer numbers L and L' as 3, and their node numbers for the 3 layers as $\{128, 64, 32\}$ and $\{32, 64, 128\}$. Moreover, we set the batch size and the learning rate as 1024 and $1e-2$ for node classification, and 64 and $5e-4$ for graph classification. For either the proposed HC-GAE or the alternative baselines, we adopt the 10-fold cross validation to compute classification accuracies, and follow the original setups for different baselines.

4.1 Node Classification

Datasets and Baselines. For node classification, we employ five real-world datasets, i.e., Cora, CiteSeer, PubMed, Amazon-Computers and Coauthor-CS. Moreover, we compare the proposed model with six classical self-supervised models, i.e., DGI [33], VGAE [29], SSL-GCN [42], GraphSage [8], GraphMAE [9], and S2GAE [28]. To fairly compare the proposed model and other baselines, we follow the previous study [10] to carry out the related experiments, and utilize the SVM classifier to predict the node labels, and the classification accuracies are shown in Table 3.

Results. Clearly, the proposed HC-GAE outperforms most of the baselines. Only the accuracy of the GraphMAE is a little higher than our model. The reasons for the effectiveness are twofold. First, the HC-GAE model is defined based on a bidirectionally hierarchical computational architecture, and can extract more effective node feature information to compress and reconstruct the original sample graph structures. Second, the HC-GAE can significantly reduce the over-smoothing problem.

Table 3: Node classification performance based on accuracy. A.R. is the average rank.

Datasets	Cora	CiteSeer	PubMed	Computers	CS	A.R.
DGI	85.41$\pm$0.34	74.51$\pm$0.51	85.95$\pm$0.66	84.68$\pm$0.39	91.33$\pm$0.30	4.0
VGAE	83.60$\pm$0.52	63.37$\pm$1.21	78.23$\pm$1.63	87.21$\pm$0.26	89.79$\pm$0.09	5.2
SSL-GCN	57.29$\pm$0.13	59.57$\pm$1.77	75.06$\pm$0.37	80.49$\pm$0.10	84.71$\pm$0.95	6.8
GraphSage	74.30$\pm$1.84	60.20$\pm$2.15	81.96$\pm$0.74	87.05$\pm$0.25	89.74$\pm$0.19	5.6
GraphMAE	85.45$\pm$0.40	72.48$\pm$0.77	85.74$\pm$0.14	88.04$\pm$0.61	**93.47$\pm$0.04**	3.0
S2GAE	86.15$\pm$0.25	74.60$\pm$0.06	86.91$\pm$0.28	90.94$\pm$0.08	91.70$\pm$0.08	2.2
HC-GAE	**87.97$\pm$0.10**	**75.29$\pm$0.09**	**87.56$\pm$0.35**	**91.07$\pm$0.14**	92.28$\pm$0.07	**1.2**

4.2 Graph Classification

Datasets and Baselines. For graph classification, we adopt five standard graph datasets, i.e., IMDB-B, IMDB-M, PROTEINS, COLLAB and MUTAG. Moreover, we compare the proposed model with a graph kernel (i.e., WLSK [27]), two supervised GNN models (i.e., DGCNN [40] and D-iffPool [37]), and four self-supervised GAE models (i.e., Graph2Vec [23], InfoGCL [36], Graph-MAE [9], S2GAE [28]). We employ the SVM associated with the resulting graph representations to compute the classification accuracies in Table 4, following the previous study in [9].

Results. Clearly, the proposed HC-GAE model outperforms most of the alternative baselines. Only accuracy of the S2GAE on the COLLAB dataset is a little higher than the proposed HC-GAE. The effectiveness of the HC-GAE are due to the following reasons. First, the alternative GAE-based models (e.g., the GraphMAE) focuses on the graph feature reconstruction rather than the topological structure information. Thus these models are only effective for node classification and have poor performance for graph classification. By contrast, the HC-GAE can simultaneously focus on either the node feature or the structure information. Second, most of the alternative deep learning baselines are essentially GNN-based methods, that usually suffers from the over-smoothing problem. By contrast, the proposed HC-GAE can significantly reduce the over-smoothing through the local graph convolution within separated subgraphs of limited sizes. Third, unlike the proposed HC-GAE that is a deep learning method, the alternative WLSK kernel cannot provide an end-to-end learning manner, i.e., the kernel computation is separated from the training of the SVM classifier.

Table 4: Graph classification performance based on accuracy. A.R. is the average rank.

Datasets	IMDB-B	IMDB-M	PROTEINS	COLLAB	MUTAG	A. R.
WLSK	64.48±0.90	43.38±0.75	71.70±0.67	N/A	80.72±3.00	7.75
DGCNN	67.45±0.83	46.33±0.73	73.21±0.34	N/A	85.83±1.66	6.25
DiffPool	72.6±3.9	47.2±1.8	75.1±3.5	78.9±2.3	85.0±10.3	5.20
Graph2Vec	71.10±0.54	50.44±0.87	73.30±2.05	N/A	83.15±9.25	5.75
InfoGCL	75.10±0.90	51.40±0.80	N/A	80.00±1.30	91.20±1.30	3.50
GraphMAE	75.52±0.66	51.63±0.52	75.30±0.39	80.32±0.46	88.19±1.26	3.20
S2GAE	75.76±0.62	51.79±0.36	76.37±0.43	**81.02±0.53**	88.26±0.76	2.00
HC-GAE	**76.72±0.60**	**51.90±1.47**	**78.13±1.37**	80.41±0.02	**92.38±1.17**	**1.20**

4.3 Ablation Study

To analyze the effectiveness of the proposed HC-GAE one step further, we replace the the hard assignment strategy of the encoder with the soft assignment strategy, and perform the same setups for graph classification. We denoted the proposed model with the soft assignment as **HC-GAE-SE**. Under this scenario, the encoder of the HC-GAE-SE cannot decompose the graph into separated subgraphs and cannot employ the local graph convolution operation to extract the feature information, since each node will be assigned to all clusters with different weights. The experimental comparisons are shown in Figure 4. Obviously, the performance of the HC-GAE-SE is lower than the proposed HC-GAE on any dataset. The reasons are twofold. First, unlike the proposed HC-GAE, the HC-GAE-SE cannot avoid the influence of the over-smoothing problem, without the local convolution within the separated subgraphs. Second, the loss function of the proposed HC-GAE relies on the local graph information through the separated subgraphs. However, this is not available for the HC-GAE-SE, thus it cannot extract the local information to enhance the effectiveness.

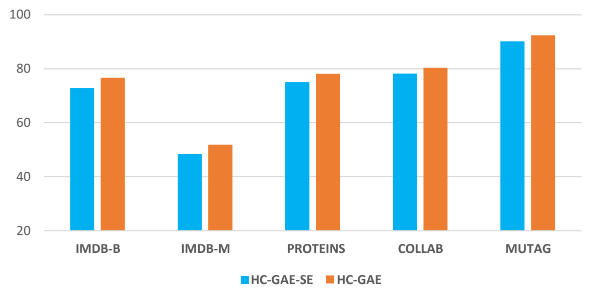

Figure 4: The ablation experiments on graph classification task.

5 Conclusions

In this paper, we have proposed a novel HC-GAE model to learn effective structural characteristics for either graph classification or node classification. During the encoding process, the HC-GAE has adopted the hard node assignment to decompose the input graph into a family of separated subgraphs, and compressed these subgraphs into coarsened nodes, resulting in a coarsened graph. During the decoding process, the HC-GAE has employed the soft node assignment to reconstruct the original graph structure, by expanding the coarsened graph of the encoder. The proposed HC-GAE can not only extract hierarchical structural features of the original graph, but also reduce the notorious over-smoothing problem arising in most classical GAEs, by restricting the convolution operation within each separated subgraphs during the encoding process. Experiments have demonstrated that the proposed HC-GAE has superior performance for both node and graph classification.

Our future work is to employ the structurally aligned (sub)structures, that have been successfully explored in our previous works [4, 5] for the proposed HC-GAE model. This will naturally result in new Transitive-Aligned HC-GAE models to guarantee the consistency between the structural correspondence information as well as the spatial position over all graphs. Finally, we will also use the Perron-Frobenius operator of hypergraphs [1, 3] associated with the proposed HC-AGE model, resulting in novel hypergraph-based HC-GAE models.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grants T2122020, 61602535 and 61976235. Ming Li acknowledged the supports from the National Natural Science Foundation of China (No. U21A20473, No. 62172370) and the Jinhua Science and Technology Plan (No. 2023-3-003a). This work is also supported in part by the Program for Innovation Research in the Central University of Finance and Economics, and the Emerging Interdisciplinary Project of CUFE.

References

[1] L. Bai, P. Ren, and E. R. Hancock. A hypergraph kernel from isomorphism tests. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, pages 3880–3885, 2014.

[2] L. Bai, Z. Zhang, C. Wang, X. Bai, and E. R. Hancock. A graph kernel based on the jensen-shannon representation alignment. In Q. Yang and M. J. Wooldridge, editors, *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3322–3328, 2015.

[3] L. Bai, F. Escolano, and E. R. Hancock. Depth-based hypergraph complexity traces from directed line graphs. *Pattern Recognition*, 54:229–240, 2016.

[4] L. Bai, Y. Jiao, L. Cui, L. Rossi, Y. Wang, P. S. Yu, and E. R. Hancock. Learning graph convolutional networks based on quantum vertex information propagation. *IEEE Transactions on Knowledge and Data Engineering*, 35(2):1747–1760, 2023.

[5] L. Cui, L. Bai, X. Bai, Y. Wang, and E. R. Hancock. Learning aligned vertex convolutional networks for graph classification. *IEEE Transactions on Neural Networks and Learning Systems*, 35(4):4423–4437, 2024.

[6] J. H. Giraldo, K. Skianis, T. Bouwmans, and F. D. Malliaros. On the trade-off between over-smoothing and over-squashing in deep graph neural networks. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 566–576, 2023.

[7] A. Grover and J. Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 855–864, 2016.

[8] W. L. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. pages 1024–1034, 2017.

[9] Z. Hou, X. Liu, Y. Cen, Y. Dong, H. Yang, C. Wang, and J. Tang. Graphmae: Self-supervised masked graph gutoencoders. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 594–604, 2022.

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

[11] Y. Huang, Y. Zeng, Q. Wu, and L. Lü. Higher-order graph convolutional network with flower-petals laplacians on simplicial complexes. *arXiv preprint arXiv:2309.12971*, 2023.

[12] W. Ju, Y. Gu, X. Luo, Y. Wang, H. Yuan, H. Zhong, and M. Zhang. Unsupervised graph-level representation learning with hierarchical contrasts. *Neural Networks*, 158:359–368, 2023.

[13] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[14] J. Li, R. Wu, W. Sun, L. Chen, S. Tian, L. Zhu, C. Meng, Z. Zheng, and W. Wang. Maskgae: Masked graph modeling meets graph autoencoders. *arXiv preprint arXiv:2205.10053*, 9:13, 2022.

[15] J. Li, X. Fu, S. Zhu, H. Peng, S. Wang, Q. Sun, P. S. Yu, and L. He. A robust and generalized framework for adversarial graph embedding. *IEEE Transactions on Knowledge and Data Engineering*, 35(11):11004–11018, 2023.

[16] J. Li, H. Shomer, H. Mao, S. Zeng, Y. Ma, N. Shah, J. Tang, and D. Yin. Evaluating graph neural networks for link prediction: Current pitfalls and new benchmarking. 2023.

[17] J. Li, R. Wu, W. Sun, L. Chen, S. Tian, L. Zhu, C. Meng, Z. Zheng, and W. Wang. What's behind the mask: Understanding masked graph modeling for graph autoencoders. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1268–1279, 2023.

[18] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou. A survey of convolutional neural networks: Analysis, applications, and prospects. *IEEE Transactions on Neural Networks and Learning Systems*, 33(12):6999–7019, 2022.

[19] Y. Liu, X. Yang, S. Zhou, X. Liu, Z. Wang, K. Liang, W. Tu, L. Li, J. Duan, and C. Chen. Hard sample aware network for contrastive deep graph clustering. In *Proceedings of the AAAI conference on artificial intelligence (AAAI)*, volume 37, pages 8914–8922, 2023.

[20] Z. Liu, Y. Chen, F. Xia, J. Bian, B. Zhu, G. Shen, and X. Kong. Tap: Traffic accident profiling via multi-task spatio-temporal graph representation learning. *ACM Transactions on Knowledge Discovery from Data*, 17(4):1–25, 2023.

[21] D. P. P. Mesquita, A. H. S. Jr., and S. Kaski. Rethinking pooling in graph neural networks. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[22] U. Michelucci. An introduction to autoencoders. *arXiv preprint arXiv:2201.03898*, 2022.

[23] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal. graph2vec: Learning distributed representations of graphs. *arXiv preprint arXiv:1707.05005*, 2017.

[24] E. Pan and Z. Kang. Beyond homophily: Reconstructing structure for graph-agnostic clustering. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pages 26868–26877, 2023.

[25] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: online learning of social representations. In *The ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 701–710, 2014.

[26] M. Réau, N. Renaud, L. C. Xue, and A. M. Bonvin. Deeprank-gnn: A graph neural network framework to learn patterns in protein-protein interfaces. *Bioinformatics*, 39(1):btac759, 2023.

[27] N. Shervashidze, S. V. N. Vishwanathan, T. Petri, K. Mehlhorn, and K. M. Borgwardt. Efficient graphlet kernels for large graph comparison. In D. A. V. Dyk and M. Welling, editors, *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 5 of *JMLR Proceedings*, pages 488–495, 2009.

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

[29] N. K. Thomas and M. Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2(10), 2016.

[30] Y. Tian, K. Dong, C. Zhang, C. Zhang, and N. V. Chawla. Heterogeneous graph masked autoencoders. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 37, pages 9997–10005, 2023.

[31] V. Vasudevan, M. Bassenne, M. T. Islam, and L. Xing. Image classification using graph neural network and multiscale wavelet superpixels. *Pattern Recognition Letters*, 166:89–96, 2023.

[32] P. Veličković, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm. Deep graph infomax. *arXiv preprint arXiv:1809.10341*, 2018.

[33] P. Velickovic, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm. Deep graph infomax. *ICLR (Poster)*, 2(3):4, 2019.

[34] L. Wang, H. Liu, Y. Liu, J. Kurtin, and S. Ji. Learning hierarchical protein representations via complete 3d graph networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.

[35] L. Wei, H. Zhao, Z. He, and Q. Yao. Neural architecture search for gnn-based graph classification. *ACM Transactions on Information Systems*, 42(1):1–29, 2023.

[36] D. Xu, W. Cheng, D. Luo, H. Chen, and X. Zhang. Infogcl: Information-aware graph contrastive learning. pages 30414–30425, 2021.

[37] Z. Ying, J. You, C. Morris, X. Ren, W. L. Hamilton, and J. Leskovec. Hierarchical graph representation learning with differentiable pooling. pages 4805–4815, 2018.

[38] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen. Graph contrastive learning with augmentations. 2020.

[39] Y. You, T. Chen, Z. Wang, and Y. Shen. When does self-supervision help graph convolutional networks? In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pages 10871–10880, 2020.

[40] M. Zhang, Z. Cui, M. Neumann, and Y. Chen. An end-to-end deep learning architecture for graph classification. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 32, 2018.

[41] Y. Zheng, M. Jin, Y. Liu, L. Chi, K. T. Phan, and Y.-P. P. Chen. Generative and contrastive self-supervised learning for graph anomaly detection. *IEEE Transactions on Knowledge and Data Engineering*, 35(12):12220–12233, 2021.

[42] Q. Zhu, B. Du, and P. Yan. Self-supervised training of graph convolutional networks. *arXiv preprint arXiv:2006.02380*, 2020.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract has directly pointed out the main contribution of this paper. Moreover, in Section 1 the contributions and scope are also clearly stated.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [No]

   Justification: This paper aims at overcoming the shortcomings existing in the Graph Auto-encoders (GAEs) and the Graph Neural Networks (GNNs). Therefore, this paper does not refer to any limitations of the proposed work.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: This paper provides the assumptions and proof for our proposed model. The details are shown in Section 1 and Section 3.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The details of our proposed model are shown in Section 3. And the experimental settings are shown in the Appendix. All these guarantee the that the experiment can be reproduced.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The dataset are from benchmarks. But we will have some further extension works based on this paper. Thus, the code is nod provided at this moment, but will be available in future.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide the required information in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provide the required information in Section 4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the required information in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This paper follows the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: This paper is about graph auto-encoder. We do not think that this paper refers to this problem.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper is about graph auto-encoder. We do not think that this paper refers to this problem.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The alternative methods for comparisons are based on the open resource.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [No]

    Justification: This paper is about graph auto-encoder. We do not think that this paper refers to this problem.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper is about graph auto-encoder. We do not think that this paper refers to this problem.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper is about graph auto-encoder. We do not think that this paper refers to this problem.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.