1. APPENDIX

1.1. Appendix A: Bottleneck Analysis of Spatial Graph Neural Networks

In this section, we conduct a rigorous theoretical exploration of the fundamental limitations of GNNs in the context of information propagation.

1.1.1. Notation

Let $G = (V, E, X)$ be an undirected graph with node set $V = \{v_1, \ldots, v_N\}$, edge set $E \subseteq \binom{V}{2}$, and node-feature matrix $X = [x_1, \ldots, x_N]^\top \in \mathbb{R}^{N \times d_0}$, where $x_i \in \mathbb{R}^{d_0}$ is the feature of node v_i. Let $A \in \{0,1\}^{N \times N}$ be the (symmetric) adjacency matrix and $D_A = \mathrm{diag}(A\mathbf{1})$ its degree matrix. The (unnormalized) graph Laplacian is $L_A = D_A - A$. We also use $\tilde{A} = A + I_N$, $\tilde{D} = \mathrm{diag}(\tilde{A}\mathbf{1})$, and the normalized adjacency $\hat{A} = \tilde{D}^{-1/2}\tilde{A}\,\tilde{D}^{-1/2}$. Write $\mathcal{N}(i) = \{\, j \mid A_{ij} = 1 \,\}$ for the 1-hop neighbors of v_i.

Let $H^{(\ell)} \in \mathbb{R}^{N \times d_\ell}$ denote the node-embedding matrix at layer ℓ, with rows $h_i^{(\ell)}$, weight matrix $W^{(\ell)} \in \mathbb{R}^{d_{\ell-1} \times d_\ell}$, and pointwise nonlinearity $\sigma(\cdot)$.

To model structure adaptively, we introduce a learnable modulation matrix $M \in \mathbb{R}^{N \times N}$, where each entry m_{ij} modulates the strength of messages from v_j to v_i. The associated modulated Laplacian is $L_M = D_M - M$, where $D_M = \mathrm{diag}(M\mathbf{1})$.

Thermodynamic quantities include node internal energy u_i, node pressure P_i, and node influence volume V_i. We define the graph-level aggregates $U = \sum_i u_i$, $P = \frac{1}{N}\sum_i P_i$ (or a scalar parameter), and $V = \sum_i V_i$, and use the graph enthalpy $H_G = U + P\,V$ as a global structural constraint.

We adopt an InfoNCE-based contrastive objective $\mathcal{L}_{\text{InfoNCE}}$ to optimize the representations.

1.1.2. Theoretical Analysis of the Fundamental Limitations of Information Propagation

The preceding discussion highlights the need for a quantifiable metric to assess "the amount of aggregation," "the effectiveness of aggregation," and "redundancy" at a given depth. To this end, we introduce a node-level metric.

Definition 1 (Information Aggregation Quality Index) For a given graph $G = (V, E)$ and the L-layered neural network, the information aggregation quality index $C_v^{(L)}$ for each node $v \in V$ is defined as:

$$C_v^{(L)} = \frac{\left|\mathcal{N}_v^{(L)}\right|}{|V|} \cdot \frac{\sum\limits_{u \in \mathcal{N}_v^{(L)}} w_{vu}^{(L)} \cdot \mathbb{1}[\cdot] \cdot s(h_v^{(L)}, h_u^{(L)})}{\sum\limits_{u \in V} s(x_v, x_u)} \cdot e^{-\lambda R_v^{(L)}} \tag{1}$$

where:

- $\mathcal{N}_v^{(L)} = \{u \in V \mid d(u, v) \le L\}$ denotes the set of nodes that can be aggregated to node v within L hops in the L-th layer of message passing;
- $w_{vu}^{(L)} \in [0, 1]$ is the aggregation weight, satisfying $\sum\limits_{u \in \mathcal{N}_v^{(L)}} w_{vu}^{(L)} = 1$;
- $s(a, b) \in [0, 1]$ is a similarity function between two node representations (such as cosine similarity or Gaussian kernel);
- x_u denotes the original feature of node u, and $h_u^{(L)}$ denotes its representation at the L-th layer;
- $\mathbb{1}[\cdot] = 1(y_u = y_v)$ is an indicator function for label consistency;
- $R_v^{(L)} = \frac{1}{|\mathcal{N}_v^{(L)}|} \sum\limits_{u \in \mathcal{N}_v^{(L)}} \left\| h_u^{(L)} - \bar{h}_v^{(L)} \right\|_2^2$ is the structural redundancy of the aggregated representations, where $\bar{h}_v^{(L)}$ is the mean of neighborhood representations;
- $\lambda > 0$ is a structural redundancy decay factor.

This definition introduces a comprehensive Information Aggregation Quality Index $C_v^{(L)}$, which quantifies how effectively a graph neural network aggregates information for a given node v at layer L. It jointly captures three critical aspects: Aggregation breadth, measuring how many nodes contribute to the representation; Aggregation quality, evaluating how relevant and label-consistent the aggregated features are; and Structural redundancy, penalizing over-smoothed or redundant representations in the local neighborhood. This index provides a principled way to assess the balance between too little and too much aggregation, helping identify when a GNN is under-aggregating or suffering from over-smoothing. Building on this foundation, we explore how neighborhood diversity evolves with increasing network depth.

Theorem 2 (Structural Representation Degradation in Graph Neural Networks within Local Neighborhoods) Given a graph $G = (V, E)$, where V is the set of nodes and E is the set of edges. Each node $v \in V$ has a representation $h_v^{(l)} \in \mathbb{R}^d$ at layer l. Consider the following information propagation mechanism of a graph neural network:

$$h_v^{(l+1)} = \sigma\left(\sum_{u \in \mathcal{N}(v) \cup \{v\}} W^{(l)} h_u^{(l)}\right) \tag{2}$$

Where:

- $\sigma(\cdot)$: non-linear activation function;
- $W^{(l)} \in \mathbb{R}^{d \times d}$: learnable parameter matrix at layer l;
- $\mathcal{N}(v)$: the set of neighbors of node v;

If there exists a constant $0 < \gamma < 1$ such that for all layers L and at least one node $v \in V$, the following holds:

$$R_v^{(L+1)} \leq \gamma R_v^{(L)} \tag{3}$$

Then, as the number of graph network layers L increases, the structural discrepancy of the neighborhood representation of node v will decay exponentially:

$$R_v^{(L)} \leq \gamma^L R_v^{(0)} \tag{4}$$

Therefore, we obtain:

$$\lim_{L \to \infty} R_v^{(L)} = 0 \tag{5}$$

This theorem characterizes the structural representation degradation problem in GNNs. It shows that, as the number of layers increases, the feature diversity within a node's local neighborhood—measured by the structural redundancy $R_v^{(L)}$—decays exponentially, eventually tending to zero. This means that all neighboring nodes become increasingly indistinguishable in their representations, leading to over-smoothing. The theorem formally explains why deep GNNs may lose local structural information and struggle to preserve discriminative features, highlighting an inherent limitation of stacking too many GNN layers.

Theorem 2 Proof.

Given a graph $G = (V, E)$ with node representations $h_v^{(l)} \in \mathbb{R}^d$ at layer l, and a graph neural network with propagation defined as:

$$h_v^{(l+1)} = \sigma\left(\sum_{u \in \mathcal{N}(v) \cup \{v\}} W^{(l)} h_u^{(l)}\right). \tag{6}$$

If there exists a constant $0 < \gamma < 1$ such that for all layers L and at least one node $v \in V$, the following inequality holds:

$$R_v^{(L+1)} \leq \gamma R_v^{(L)}, \tag{7}$$

then the structural redundancy $R_v^{(L)}$ decays exponentially as the number of layers L increases:

$$R_v^{(L)} \leq \gamma^L R_v^{(0)}, \quad \text{and thus,} \quad \lim_{L \to \infty} R_v^{(L)} = 0. \tag{8}$$

We start by recalling the definition of structural redundancy $R_v^{(L)}$:

$$R_v^{(L)} = \frac{1}{|\mathcal{N}_v^{(L)}|} \sum_{u \in \mathcal{N}_v^{(L)}} \|h_u^{(L)} - \bar{h}_v^{(L)}\|_2^2, \tag{9}$$

where

$$\bar{h}_v^{(L)} = \frac{1}{|\mathcal{N}_v^{(L)}|} \sum_{u \in \mathcal{N}_v^{(L)}} h_u^{(L)}. \tag{10}$$

By the hypothesis of the theorem, there exists a constant $0 < \gamma < 1$ such that for all $L \geq 0$:

$$R_v^{(L+1)} \leq \gamma R_v^{(L)}. \tag{11}$$

Expanding this recursive inequality iteratively, we obtain:

$$\begin{aligned} R_v^{(L)} &\leq \gamma R_v^{(L-1)} && (12)\\ &\leq \gamma^2 R_v^{(L-2)} && (13)\\ &\vdots && (14)\\ &\leq \gamma^L R_v^{(0)}. && (15) \end{aligned}$$

Thus, we have explicitly shown the exponential decay of the structural redundancy:

$$R_v^{(L)} \leq \gamma^L R_v^{(0)}. \tag{16}$$

Next, we consider the limit behavior as $L \to \infty$:

$$\lim_{L \to \infty} R_v^{(L)} \leq \lim_{L \to \infty} \gamma^L R_v^{(0)}. \tag{17}$$

Given that $0 < \gamma < 1$, it is well-known that:

$$\lim_{L \to \infty} \gamma^L = 0. \tag{18}$$

Therefore, we directly conclude that:

$$\lim_{L \to \infty} R_v^{(L)} = 0. \tag{19}$$

This completes the proof. **Theorem 3 (Threshold Principle of Information Aggregation Quality in GNNs).** Let $C_v^{(L)}$ be the Information Aggregation Quality Index of node $v \in V$ at layer L as defined in Definition 1.

Then, given a threshold value $\tau \in (0, 1)$ and two positive integers $L_1 < L_2$, the following statements hold:

- When $L < L_1$, the aggregation scope is too narrow. Thus, there exists at least one node $v \in V$ such that:

$$C_v^{(L)} \leq \tau \quad \text{(Aggregation is incomplete)} \tag{20}$$

- When $L > L_2$, despite broader aggregation scope, excessive smoothing and structural redundancy lead to degradation. Thus, there exists at least one node $v \in V$ such that:

$$C_v^{(L)} \leq \tau \quad \text{(Aggregation quality is degraded)} \tag{21}$$

Proof of Theorem 3

We proceed in two steps, corresponding to the two regimes defined by the theorem.

Step 1 (Incomplete Aggregation, $L < L_1$): By Definition 1, we have:

$$C_v^{(L)} = \frac{|\mathcal{N}_v^{(L)}|}{|V|} \cdot \frac{\sum_{u \in \mathcal{N}_v^{(L)}} w_{vu}^{(L)} \cdot \mathbb{1}[\cdot](y_u = y_v) \cdot s(h_v^{(L)}, h_u^{(L)})}{\sum_{u \in V} s(x_v, x_u)} \cdot e^{-\lambda R_v^{(L)}}. \tag{22}$$

When the network depth L is small ($L < L_1$), the set $\mathcal{N}_v^{(L)} = \{u \in V \mid d(u, v) \leq L\}$ is limited. Therefore, we have:

$$\frac{|\mathcal{N}_v^{(L)}|}{|V|} \ll 1. \tag{23}$$

Thus, the first term (aggregation breadth) of $C_v^{(L)}$ is significantly small. Even if similarity $s(\cdot, \cdot)$ is relatively high for nearby nodes, the small cardinality of the aggregated nodes leads to:

$$C_v^{(L)} \approx \frac{|\mathcal{N}_v^{(L)}|}{|V|} \cdot (\text{bounded terms}) \ll 1. \tag{24}$$

Therefore, given the threshold $\tau \in (0, 1)$, there exists at least one node v satisfying:

$$C_v^{(L)} \leq \tau, \quad \text{for } L < L_1, \tag{25}$$

which demonstrates the incomplete aggregation at shallow depths.

Step 2 (Aggregation Quality Degradation, $L > L_2$): As the depth L increases ($L > L_2$), we have:

$$|\mathcal{N}_v^{(L)}| \to |V| \quad \text{(aggregation breadth is large)}. \tag{26}$$

However, due to Theorem 2 (structural representation degradation), we know the structural redundancy $R_v^{(L)}$ satisfies exponential decay:

$$\lim_{L \to \infty} R_v^{(L)} = 0. \tag{27}$$

This implies representations become increasingly indistinguishable, leading to excessive smoothing. Particularly, excessive smoothing means that the node similarity $s(h_v^{(L)}, h_u^{(L)})$ no longer discriminates label-consistent nodes effectively, hence decreasing the second term (aggregation quality):

$$\sum_{u \in \mathcal{N}_v^{(L)}} w_{vu}^{(L)} \mathbb{1}[\cdot](y_u = y_v) s(h_v^{(L)}, h_u^{(L)}) \to \text{small constant}. \tag{28}$$

Additionally, as nodes become similar, even nodes with different labels begin contributing equally, thus further degrading aggregation quality.

Moreover, the redundancy penalty term $e^{-\lambda R_v^{(L)}}$ approaches 1 since $R_v^{(L)} \to 0$, but the aggregation quality degradation dominates. Thus, for large enough $L > L_2$, we again have:

$$C_v^{(L)} \leq \tau. \tag{29}$$

Combining Step 1 and Step 2: From the two steps above, we have rigorously demonstrated two scenarios:

- At small depth ($L < L_1$), aggregation is incomplete ($C_v^{(L)} \leq \tau$);
- At large depth ($L > L_2$), aggregation is excessively smoothed and degraded ($C_v^{(L)} \leq \tau$).

Therefore, there exists an optimal aggregation depth region between L_1 and L_2 that achieves better balance, and outside this range, the aggregation quality index is below the threshold τ.

This completes the proof.

This theorem reveals the **threshold principle** for information aggregation quality in GNNs. It shows that when the network depth is too shallow, the aggregation scope of a node is too narrow, resulting in incomplete information aggregation. On the other hand, when the network is too deep, although the aggregation scope expands, the quality of information still declines due to over-smoothing and structural redundancy. In both cases, the information aggregation quality index $C_v^{(L)}$ falls below a critical threshold τ, indicating suboptimal aggregation performance.

Choosing GNN depth per dataset is costly. Stability is limited by (i) locality—an L-layer GNN accesses at most L-hop information, and heterogeneous graph statistics (diameter, clustering, homophily) preclude a universal L; and (ii) the locality trade-off: larger L broadens context but causes over-smoothing, while smaller L misses long-range signals. Because spectral/geometric properties (e.g., mixing time, expansion/cut characteristics, local curvature) vary across graphs and substructures, the optimal depth and propagation radius shift, reducing robustness and transferability. We propose a *path-independent state function* as a global regularizer to enlarge the receptive field without over-smoothing, stabilizing GNN representations and improving practical generalization and efficiency.

1.2. Appendix B: Related Work

1.2.1. Spectral Decomposition

The spectral decomposition of the normalized Laplacian matrix $\mathrm{L}_{\mathrm{norm}}$ is defined as $\mathrm{L}_{\mathrm{norm}} = \mathrm{Lap}(A) = U\Lambda U^{\top}$, where the diagonal matrix $\Lambda = \mathrm{eig}(\mathrm{Lap}(A)) = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ consists of real eigenvalues known as the graph spectrum, and $U = [u_1\, u_2\, \ldots\, u_n] \in \mathbb{R}^{n\times n}$ are the corresponding orthonormal eigenvectors known as the spectral bases.

1.2.2. Graph Neural Networks

Graph Neural Networks (GNNs) are a class of neural networks designed to work with graph-structured data. One popular variant of GNNs is the Graph Convolutional Network (GCN). In a GCN, each node aggregates features from its neighbors, incorporating both its own features and those of its neighbors to update its representation. This process is formulated as follows:

$$\mathbf{H}^{(l+1)} = \sigma\left(\mathbf{D}^{-\frac{1}{2}}\hat{\mathbf{A}}\mathbf{D}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right), \tag{30}$$

1.2.3. Graph Contrastive Learning

Graph contrastive learning approaches usually design different views and aim to pre-train a graph encoder by maximizing agreement between representations of views. Generally, given an origin graph $\mathcal{G} = (X, A)$, two augmented views are denoted as $\mathcal{G}_1 = (X_1, A_1)$ and $\mathcal{G}_2 = (X_2, A_2)$, by applying the data augmentation function $t(\cdot)$. The representations of all nodes in augmented views are denoted as $H^1 = f_\theta(X^1, A^1)$ and $H^2 = f_\theta(X^2, A^2)$, where $f_\theta(\cdot)$ is an GNN encoder. The agreement between the node representations is commonly measured through Mutual Information (MI). Thus, the contrastive objective can be generally formulated as:

$$\max_{\theta} \mathcal{MI}(\mathbf{H}^1, \mathbf{H}^2) \tag{31}$$

1.3. Appendix c:Theoretical Analysis

1.3.1. Properties of the Internal Energy Definition of a Node

Monotonicity and Locality

Firstly, consider the internal energy definition for node i:$u_i = \frac{1}{2}\sum w_{ij}\|h_i - h_j\|_2^2$

This energy measure is local, as it depends only on node i and its neighboring nodes j, with the edge weight w_{ij} reflecting the similarity and relationship between the features of nodes i and j. The energy is calculated by evaluating the squared difference between the features of the nodes.

Therefore, monotonicity holds: if the feature of a neighboring node j is enhanced, i.e., $\|h_j - h_i\|_2^2$ decreases, the internal energy of node i, u_i, will decrease; conversely, if the feature of node j is diminished, the internal energy u_i will increase.

This reduction in energy corresponds to the "similarity" between neighboring nodes: if the features of two nodes are more similar, their energy will be lower, thereby resulting in a smoother graph structure that better reflects the relationship between nodes.

Impact of Weighted Edges.

Weighted edges play a crucial role in this definition. The edge weight w_{ij} reflects the strength of the association between nodes i and j. When the weight between nodes i and j is large, the feature difference between these nodes contributes more significantly to the graph energy. Edges with larger weights make the connected nodes more important. During energy maximization, these nodes have a greater impact on the graph's representation power and learning performance.

Significance of Maximizing Graph Energy.

Maximizing graph energy enhances the diversity of the graph and the distinguishability of the nodes by increasing the differences between nodes. The issue of excessive smoothing in GNNs during training is a common problem. The edge weight w_{ij} is used to control the contribution of the feature differences between neighboring nodes to the total energy. By increasing the feature differences between nodes i and j, the energy increases, which encourages the model to preserve the diversity between nodes in the graph, thus preventing excessive smoothing of node features. Compared to minimizing energy, maximizing energy highlights the differences between nodes, which is particularly important for tasks such as node classification and graph classification. Moreover, maximizing energy ensures that the structural information of the graph is effectively retained, thereby producing expressive representations by allowing for a better expansion of relationships between neighboring nodes.

1.3.2. Mathematical Modeling of Node Pressure

We model the pressure experienced by a node as a measure induced by its neighbors in the graph structure. Specifically, this pressure is quantified via a gradient-based formulation, capturing how neighboring nodes influence the target node.

In conventional neural network training, gradients are computed with respect to the model parameters in order to facilitate backpropagation. However, in our context, we consider the gradient of the loss function with respect to the input x, which may initially seem counterintuitive. In reality, this analysis reflects how the output of the neural network function $f(x)$ evolves in response to variations in x, and thus necessitates tools from functional analysis.

A neural network can be regarded as a composite function:

$$Z = f_\theta(x) \tag{32}$$

Applying the chain rule, the gradient of the loss $\mathcal{L}$ with respect to the input x is given by:

$$\nabla_x \mathcal{L}(f_\theta(X)) = \frac{\partial \mathcal{L}}{\partial f} \cdot \frac{\partial f_\theta(X)}{\partial X} \tag{33}$$

To further formalize this notion, we invoke the **Riesz Representation Theorem**, which provides a rigorous foundation for interpreting gradients as inner products in Hilbert spaces.

Riesz Representation Theorem.

Let $\mathcal{H}$ be a real Hilbert space and let L: $\mathcal{H} \to \mathbb{R}$ be a continuous linear functional. Then there exists a unique $g \in \mathcal{H}$ such that for all $f \in \mathcal{H}$,

$$L(f) = \langle f, g \rangle_{\mathcal{H}}. \tag{34}$$

This theorem implies that any continuous linear functional on a Hilbert space can be represented as an inner product with a fixed element of the space. In our context, this provides a principled way to interpret the gradient $\nabla_x \mathcal{L}$ as a projection of the functional variation of $f(x)$ in the direction of greatest sensitivity.

We define the Fréchet derivative of the loss functional $\mathcal{L}(f)$ as a linear functional $D\mathcal{L}[f]$. Its value in a given direction h is defined as:

$$D\mathcal{L}[f](h) = \lim_{\epsilon \to 0} \frac{\mathcal{L}(f + \epsilon h) - \mathcal{L}(f)}{\epsilon} \tag{35}$$

Here, $D\mathcal{L}[f] \in \mathcal{H}^*$, representing a linear functional on the function space $\mathcal{H}$.

According to the Riesz Representation Theorem, there exists a unique element $g \in \mathcal{H}$ such that:

$$D\mathcal{L}[f](h) = \langle h, g \rangle_{\mathcal{H}} \quad \text{for some unique } g \in \mathcal{H}. \tag{36}$$

We refer to $g = \frac{\delta \mathcal{L}}{\delta f}$ as the **Riesz representative** of the gradient of the loss functional, which can be intuitively understood as the "steepest descent direction."

Now we aim to prove:

The direction of the Riesz representative g is aligned with the functional gradient $\frac{\delta \mathcal{L}}{\delta x}$.

We use the **Cauchy–Schwarz inequality** to prove this alignment:

$$|\langle g, J \rangle| \leq \|g\| \cdot \|J\|, \quad \text{where } J = \frac{\partial f_\theta}{\partial X} \tag{37}$$

When the equality holds: $g \parallel J$, it means that the input feature perturbation reaches its maximum effect.

In fact, during the process of gradient descent training, based on the principle of maximum descent efficiency, the system naturally aligns g and $\frac{\partial f_\theta}{\partial X}$ over time through multiple rounds of training.

Now, we can conclude that for all nodes, the gradient of the loss with respect to the features inevitably increases during training as the neural network evolves. Inspired by the **Weber–Fechner law**, which describes the relationship between stimulus intensity and perceived intensity, we note that since the gradient pressure is guaranteed to increase, what we need to capture is the **rate** of this increase. Therefore, we employ a **logarithmic function** to encapsulate it.

1.3.3. Graph Enthalpy-Based Information Conservation Cycle

We define "information conservation" in Graph Neural Networks (GNNs) as the requirement that node representations during encoding must satisfy the following conditions:

- The original information should not be completely lost (otherwise, over-smoothing occurs);
- The representations should not grow indefinitely (otherwise, stability is lost);
- The model should not "invent" information from thin air (otherwise, overfitting occurs).

In short, the changes in each node's information on the graph come from the "contribution of neighbors' information," rather than being generated arbitrarily by the model.

In the case of constant pressure, the constancy of pressure implies that the weights of the graph neural network remain unchanged, and thus the variation of graph enthalpy is solely determined by the change in internal energy. In this case, if $\Delta H = 0$, the system tends to reach an "information steady state," where the information flow between nodes stabilizes and the feature differences no longer change dramatically, indicating system stability. However, if $\Delta H < 0$, it implies that the feature differences decrease, the node representations become more similar, and information is overly diffused, leading to **over-smoothing**. Conversely, if $\Delta H > 0$ and is significantly large, the model may "invent" information, resulting in **overfitting**. Therefore, to prevent excessive diffusion of information that could lead to over-smoothing, we choose to **maximize the graph enthalpy** H, thus avoiding the enthalpy approaching 0. Theorem 12 proves that during the training process of graph neural networks, perturbations of the graph's features do not lead to significant changes in ΔH, meaning ΔH will not increase drastically.

1.4. Appendix D: Proofs of Theorems

1.4.1. Proof of Theorem 2

Theorem 6 (Graph-Level Energy Representation via Laplacian). The graph-level energy $U(\mathcal{G})$, defined as the sum of node internal energies, can be equivalently expressed through the combinatorial Laplacian $L = D - M$:

$$U(\mathcal{G}) = \frac{1}{2}\sum_{i,j} m_{ij}\|h_i - h_j\|_2^2 = \mathrm{tr}(H^\top LH). \tag{38}$$

We start from the definition of graph-level energy:

$$U(\mathcal{G}) = \frac{1}{2}\sum_{i,j} m_{ij}\|h_i - h_j\|_2^2. \tag{39}$$

Expanding the squared norm explicitly, we have:

$$U(\mathcal{G}) = \frac{1}{2}\sum_{i,j} m_{ij}(h_i - h_j)^\top(h_i - h_j). \tag{40}$$

This expression expands to:

$$U(\mathcal{G}) = \frac{1}{2}\sum_{i,j} m_{ij}(h_i^\top h_i + h_j^\top h_j - 2h_i^\top h_j). \tag{41}$$

Because the graph is undirected, each edge (i, j) is counted twice (once as (i, j), once as (j, i)). The factor $\frac{1}{2}$ corrects this double-counting.

Grouping the terms involving individual nodes separately, we obtain:

$$U(\mathcal{G}) = \frac{1}{2}\left(\sum_{i,j} m_{ij}h_i^\top h_i + \sum_{i,j} m_{ij}h_j^\top h_j - 2\sum_{i,j} m_{ij}h_i^\top h_j\right). \tag{42}$$

Since the degree matrix D is defined as $D = \mathrm{diag}(d_1, \ldots, d_N)$, with diagonal entries $d_i = \sum_j m_{ij}$, we rewrite the above as:

$$U(\mathcal{G}) = \frac{1}{2}\left(\sum_i d_i h_i^\top h_i + \sum_j d_j h_j^\top h_j - 2\sum_{i,j} h_i^\top m_{ij} h_j\right). \tag{43}$$

This simplifies to the trace form as:

$$U(\mathcal{G}) = \mathrm{tr}(H^\top D H) - \mathrm{tr}(H^\top M H). \tag{44}$$

Since the combinatorial Laplacian L is defined as:

$$L = D - M, \tag{45}$$

we finally have:

$$U(\mathcal{G}) = \mathrm{tr}(H^\top L H). \tag{46}$$

This completes the proof.

1.4.2. Proof of Theorem 5

Refined proof with explicit constant.

In addition to Theorem 11, we make the following mild assumptions: (A1) M is symmetric, $L = D - M$ is symmetric; (A2) node features are bounded, $\|h_i\| \le B_h$; (A3) the task loss satisfies $\left\|\frac{\partial \mathcal{L}}{\partial h_j}\right\| \le G_{\max}$ and $\left\|\frac{\partial^2 \mathcal{L}}{\partial h_j^2}\right\| \le H_{\max}$; (A4) P_G and V_G are differentiable with $|P_G| \le P_{\max}$, $|V_G| \le V_{\max}$ and $\left\|\frac{\partial V_G}{\partial h_j}\right\| \le L_V$; (A5) $\alpha > 0$, $\epsilon > 0$, $m_{ij} \le m_{\max}$, and each node j is contained in at most C_{nb} two-hop neighborhoods, with $Z_{\min} = \min_v |\mathcal{N}^{(1,2)}(v)| \ge 1$.

(i) Internal energy. Since $U_G = \mathrm{tr}(H^\top L H)$ and L is symmetric,

$$\frac{\partial U_G}{\partial h_j} = 2\sum_k L_{jk}\, h_k \quad \Rightarrow \quad \left\|\frac{\partial U_G}{\partial h_j}\right\| \le 2\Big(\sum_k |L_{jk}|\Big) \max_k \|h_k\| \le 2\,\|L\|_\infty\, B_h. \tag{47}$$

(ii) Pressure–volume product. Write $P_v = \alpha\, \frac{\log(S_v+\epsilon)}{Z_v}$ with $S_v = \sum_{t\in\mathcal{N}^{(1,2)}(v)} m_{vt} \left\|\frac{\partial \mathcal{L}}{\partial h_t}\right\|$ and $Z_v = |\mathcal{N}^{(1,2)}(v)|$. Only the term $t = j$ depends on h_j, and by smoothness of $\mathcal{L}$,

$$\left\|\frac{\partial}{\partial h_j}\left\|\frac{\partial \mathcal{L}}{\partial h_j}\right\|\right\| \le \left\|\frac{\partial^2 \mathcal{L}}{\partial h_j^2}\right\| \le H_{\max}. \tag{48}$$

Chain rule and $\frac{d}{dx}\log x = 1/x$ give

$$\left\|\frac{\partial P_v}{\partial h_j}\right\| \le \alpha \cdot \frac{1}{Z_v} \cdot \frac{1}{S_v + \epsilon} \cdot m_{vj}\, H_{\max} \le \alpha \cdot \frac{m_{\max}}{\epsilon} \cdot \frac{H_{\max}}{Z_v}. \tag{49}$$

Averaging over v and using that j can appear in at most C_{nb} two-hop neighborhoods yields

$$\left\|\frac{\partial P_G}{\partial h_j}\right\| \le \alpha \cdot \frac{m_{\max}}{\epsilon} \cdot \frac{H_{\max}}{Z_{\min}} \cdot C_{\mathrm{nb}} =: L_P. \tag{50}$$

Therefore,

$$\left\|\frac{\partial (P_G V_G)}{\partial h_j}\right\| \le \left\|\frac{\partial P_G}{\partial h_j}\right\| |V_G| + |P_G| \left\|\frac{\partial V_G}{\partial h_j}\right\| \le L_P\, V_{\max} + P_{\max}\, L_V. \tag{51}$$

(iii) Putting things together. We obtain the explicit bound

$$\left\|\frac{\partial H_G}{\partial h_j}\right\| \le 2\,\|L\|_\infty\, B_h \;+\; L_P\, V_{\max} \;+\; P_{\max}\, L_V =: K. \tag{52}$$

Combining with the first-order expansion used above gives

$$|\Delta H_G| \le K\, \|\Delta h_j\|. \tag{53}$$

This refines the existence of $K > 0$ by expressing it in terms of graph structure ($\|L\|_\infty$, $m_{\max}$, two-hop coverage C_{nb}), loss smoothness ($H_{\max}$), and boundedness/derivatives of P_G and V_G.

1.5. Theorem 6 and propf

Theorem 6 (Boundedness of Node Pressure). According to Definition 5, for any node v in the graph, its node pressure P_v is bounded. This means there exist finite lower and upper bounds $P_v^{\min}$ and $P_v^{\max}$ such that:

$$P_v^{\min} \leq P_v \leq P_v^{\max} \tag{54}$$

From Definition 5, node pressure P_v is defined as:

$$P_v = \alpha \cdot \frac{\log\left(\sum_{j \in \mathcal{N}(v)^{(1,2)}} m_{vj} \left\| \frac{\partial \mathcal{L}}{\partial h_j} \right\|_2 + \epsilon\right)}{|\mathcal{N}(v)^{(1,2)}|}. \tag{55}$$

We analyze each component separately:

- The gradient norms $\| \frac{\partial \mathcal{L}}{\partial h_j} \|_2$ are finite, given standard assumptions in neural network training.
- Edge weights m_{vj} are non-negative and finite.
- The neighbor set size $|\mathcal{N}(v)^{(1,2)}|$ is finite and positive, as the graph has a finite number of nodes.
- The constant $\epsilon > 0$ ensures the argument inside the logarithm is strictly positive.

Thus, the following holds:

$$0 < \frac{\epsilon}{|\mathcal{N}(v)^{(1,2)}|} \leq \frac{\sum_{j \in \mathcal{N}(v)^{(1,2)}} m_{vj} \left\| \frac{\partial \mathcal{L}}{\partial h_j} \right\|_2 + \epsilon}{|\mathcal{N}(v)^{(1,2)}|} \leq C < \infty \tag{56}$$

for some finite positive constant C, determined by maximum gradient norms and edge weights.
Applying the logarithm function (monotonically increasing and continuous on $(0, \infty)$), we have:

$$\log\left(\frac{\epsilon}{|\mathcal{N}(v)^{(1,2)}|}\right) \leq \log\left(\frac{\sum_{j \in \mathcal{N}(v)^{(1,2)}} m_{vj} \| \frac{\partial \mathcal{L}}{\partial h_j} \|_2 + \epsilon}{|\mathcal{N}(v)^{(1,2)}|}\right) \leq \log(C) \tag{57}$$

Multiplying by the finite scaling factor α, we get finite bounds for P_v:

$$\alpha \cdot \log\left(\frac{\epsilon}{|\mathcal{N}(v)^{(1,2)}|}\right) \leq P_v \leq \alpha \cdot \log(C) \tag{58}$$

Hence, we have established the existence of finite lower and upper bounds, defined as:

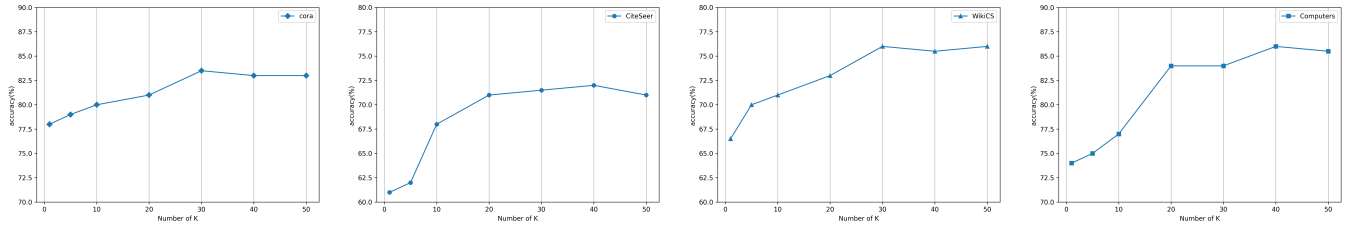
$$P_v^{\min} = \alpha \cdot \log\left(\frac{\epsilon}{|\mathcal{N}(v)^{(1,2)}|}\right), \quad P_v^{\max} = \alpha \cdot \log(C). \tag{59}$$

Therefore, the node pressure P_v is bounded.

1.6. Appendix E:Algorithm and Complexity Analysis

1.6.1. Algorithm Description

We summarize the enthalpy-guided contrastive learning framework in Algorithm 1. The method contains two forward propagation paths—one standard GCN-based encoder and one augmented with a learnable structural modulation matrix M. Each path shares the same weight matrix W and is trained via two objectives: InfoNCE-based contrastive loss $\mathcal{L}_{\text{cl}}$ and negative log enthalpy loss $\mathcal{L}_{\text{el}}$.



(a) Cora (b) CiteSeer (c) WikiCS (d) Computers

Fig. 1: Comparison of performance on four benchmark datasets.

Algorithm 1 Beyond Locality: Enthalpy-Guided Unsupervised Graph Convolutional Networks

Require: Graph $G = (V, E, X)$, normalized adjacency $\hat{A}$, learnable modulation matrix M, encoder parameters W, temperature τ, hyperparameter α

Ensure: Node embeddings Z

The initialization of M is performed by constructing A_{KNN} via K-Nearest Neighbors, which is then merged with the original graph.

for each epoch **do**
  Generate two augmented graph views via GDA
  Compute $H_1 = \sigma((\hat{A} \odot M)HW)$
  Compute $H_2 = \sigma(\hat{A}HW)$
  Calculate contrastive loss $\mathcal{L}_{\text{cl}}$ from H_1, H_2
  Compute graph enthalpy H_G using Equation (15)
  Compute enthalpy loss $\mathcal{L}_{\text{el}} = -\log(H_G)$
  Update encoder W using $\mathcal{L}_{\text{cl}}$
  Update modulation matrix M using $\mathcal{L}_{\text{el}}$
end for
return Final node embeddings Z

1.6.2. Complexity Analysis

We analyze the time and space complexity of the proposed GE-GCL framework per training epoch.

Time Complexity.

Let n be the number of nodes, d the feature dimension, and m the number of edges. The primary computation per layer involves: Sparse matrix multiplications with $\hat{A}$ and $\hat{A} \odot M$: $\mathcal{O}(md)$; Computation of $\nabla_{h_j}\mathcal{L}$ in node pressure: $\mathcal{O}(nd)$; PageRank radius $R(v)$ for each node: approximately $\mathcal{O}(m \cdot t)$, where t is the iteration count (typically small); InfoNCE loss over positive and negative pairs: $\mathcal{O}(n^2 d)$ in worst-case, mitigated by mini-batching or negative sampling.

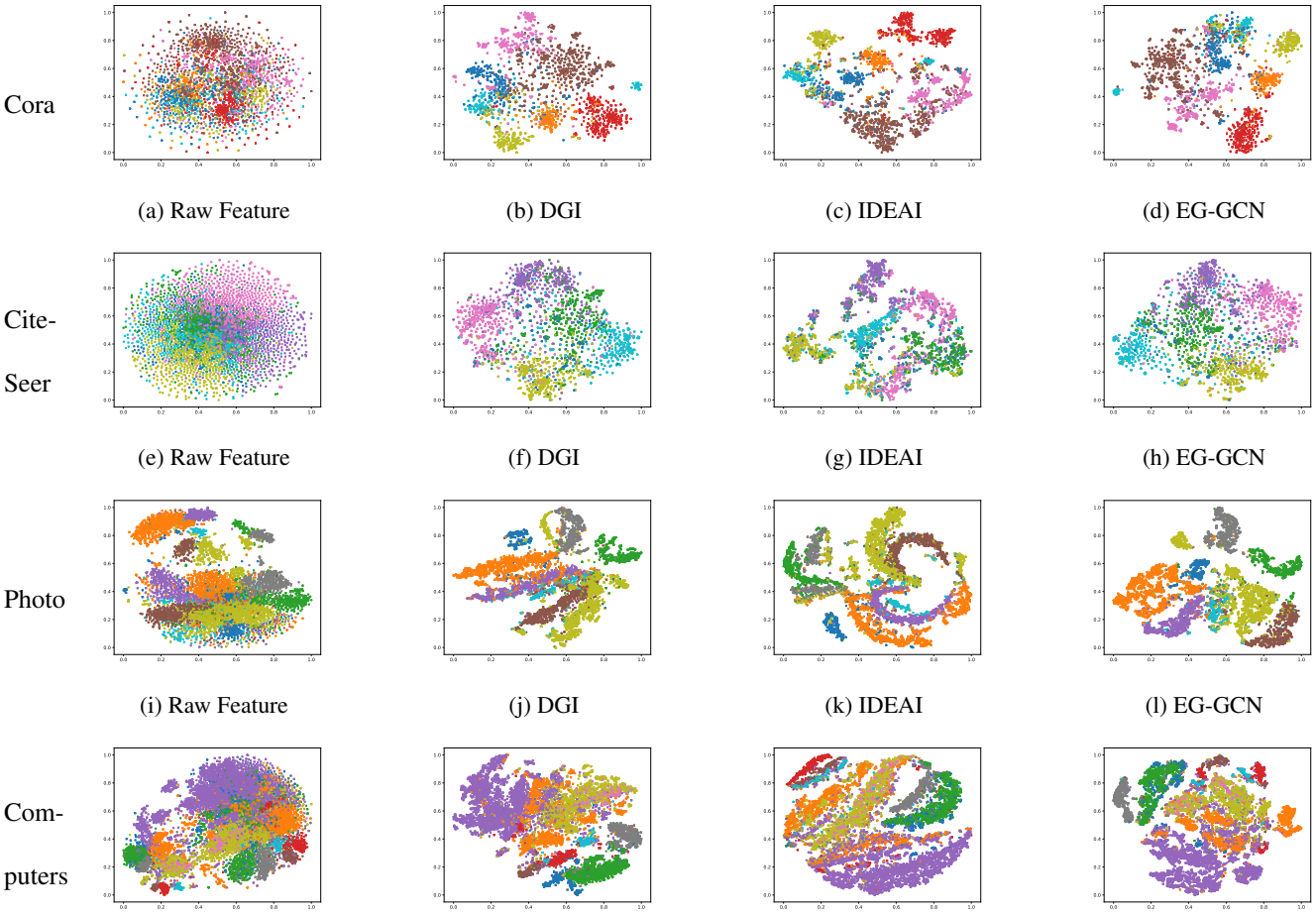
Overall, the forward and backward pass complexity per epoch is approximately:

$$\mathcal{O}(md + nd + n^2 d) = \mathcal{O}(n^2 d) \text{ in worst-case} \tag{60}$$

Space Complexity.

- Node features: $\mathcal{O}(nd)$
- Adjacency and modulation matrix M: $\mathcal{O}(m)$ for sparse storage
- Gradients and intermediate activations: $\mathcal{O}(nd)$

Hence, the overall space complexity is $\mathcal{O}(nd + m)$.



(a) Raw Feature (b) DGI (c) IDEAI (d) EG-GCN

(e) Raw Feature (f) DGI (g) IDEAI (h) EG-GCN

(i) Raw Feature (j) DGI (k) IDEAI (l) EG-GCN

(m) Raw Feature (n) DGI (o) IDEAI (p) EG-GCN

Fig. 2: t-SNE visualization on CORA, CiteSeer, Computers, and Photo datasets. Different colors indicate the different labels.

1.7. Appendix E: Supplemental Experiment

1.7.1. Qualitative Visualization

To qualitatively evaluate the embedding performance of our developed model, we employed t-SNE to visualize the learned node embedding representations from the Cora, CiteSeer, WiKiCS, Amazon Computers, and Amazon Photo datasets. These visualizations are presented in Figure 2. It's worth noting that the embedding representations generated by our EG-GCN method exhibit more discernible clustering in visual terms compared to the raw features, DGI, and IDEAI. This observation further emphasizes the effectiveness of our method in the field of self-supervised graph representation learning.

1.7.2. Experiments on Heterogeneous Graphs

Table 1: Performance comparison across ACM, Yelp, DBLP, and Aminer datasets using NMI and ARI metrics.

Method	**ACM**		**Yelp**		**DBLP**		**Aminer**	
	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI
DeepWalk	41.6±0.5	35.3±0.6	35.1±0.8	37.7±1.1	69.0±0.2	73.3±0.3	26.2±0.3	22.4±0.4
Mp2vec	21.4±0.7	21.1±0.5	38.9±0.6	39.5±0.5	73.5±0.4	77.7±0.6	30.4±0.4	25.5±0.6
DMGI	67.8±0.9	70.2±1.0	36.8±0.6	34.4±0.7	72.2±0.8	72.8±0.9	27.3±0.9	23.1±0.8
DMGIattn	**70.2±0.3**	**72.5±0.6**	38.1±0.8	40.2±0.6	69.6±0.6	73.9±0.4	28.3±0.3	25.5±0.5
HDMI	69.5±0.5	72.3±0.7	38.9±0.6	40.7±0.8	**73.1±0.3**	**74.4±0.4**	**33.5±0.4**	**28.9±0.5**
OUR	67.2±7.0	70.5±0.9	**39.1±0.7**	**41.2±0.3**	72.9±0.6	73.9±0.4	26.3±0.3	23.5±0.5

Our method is designed and evaluated primarily on homogeneous graphs. To verify the generality of our approach, we also conduct experiments on heterogeneous graphs. The relevant datasets are listed as follows.

Datasets: We evaluate on five heterogeneous and two homogeneous graph datasets. **ACM**[1], **DBLP**[1], and **Aminer**[2] are academic heterogeneous graphs with three types of nodes and four edge types; their labels correspond to paper or author categories. **Yelp**[3] is a business graph with four node types, six edge types, and business categories as labels. **Photo** and **Computers**[4] are homogeneous co-purchase graphs from Amazon, where nodes are products, edges indicate co-purchase behavior, and labels denote product categories.Detailed results are shown in Table 3.

We conduct a brief comparison with several state-of-the-art heterogeneous graph methods. Although our approach does not achieve a significant improvement, its performance is comparable to the baselines, with only marginal differences.

2. REFERENCES

[1] X. Wang, H. Ji, C. Shi, et al., "Heterogeneous graph attention network," *Proceedings of The World Wide Web Conference*, pp. 2022–2032, 2019.

[2] B. Hu, Y. Fang, and C. Shi, "Adversarial learning on heterogeneous information networks," *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 120–129, 2019.

[3] J. Zhao, X. Wang, C. Shi, et al., "Heterogeneous graph structure learning for graph neural networks," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 5, pp. 4697–4705, 2021.

[4] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann, "Pitfalls of graph neural network evaluation," *arXiv preprint arXiv:1811.05868*, 2018.