Less is More: on the Over-Globalizing Problem in Graph Transformers

Yujie Xing [1] **Xiao Wang** [2] **Yibo Li** [1] **Hai Huang** [1] **Chuan Shi** [1]

Abstract

Graph Transformer, due to its global attention mechanism, has emerged as a new tool in dealing with graph-structured data. It is well recognized that the global attention mechanism considers a wider receptive field in a fully connected graph, leading many to believe that useful information can be extracted from all the nodes. In this paper, we challenge this belief: does the globalizing property always benefit Graph Transformers? We reveal the over-globalizing problem in Graph Transformer by presenting both empirical evidence and theoretical analysis, i.e., the current attention mechanism overly focuses on those distant nodes, while the near nodes, which actually contain most of the useful information, are relatively weakened. Then we propose a novel Bi-Level Global Graph Transformer with Collaborative Training (CoBFormer), including the inter-cluster and intra-cluster Transformers, to prevent the over-globalizing problem while keeping the ability to extract valuable information from distant nodes. Moreover, the collaborative training is proposed to improve the model's generalization ability with a theoretical guarantee. Extensive experiments on various graphs well validate the effectiveness of our proposed CoBFormer. The source code is available for reproducibility at: https://github.com/null-xyj/CoBFormer.

1. Introduction

Graph-structured data, an essential and prevalent form in the real world, plays a vital role in modeling object interactions, such as social networks (Yang et al., 2017), transportation networks (Chen et al., 2023), and protein-protein interaction networks. Graph Neural Networks (GNNs) (Kipf & Welling, 2017; Veličković et al., 2018; Hamilton et al., 2017; Yu et al., 2023), as representative graph machine learning methods, effectively utilize their message-passing mechanism to extract useful information and learn high-quality representations from graph data. However, GNNs face challenges with layer stacking due to over-smoothing (Li et al., 2018; Nt & Maehara, 2019; Oono & Suzuki, 2020) and over-squashing (Topping et al., 2021; Deac et al., 2022) problems, which limit their receptive fields to near neighbors. In contrast, Transformers (Vaswani et al., 2017), with their global attention mechanism, have shown exceptional expressive capability, which makes significant strides in various fields, including natural language processing (Devlin et al., 2018) and computer vision (Dosovitskiy et al., 2021). Incorporating Transformers into graph data presents an excellent solution to these challenges since they naturally construct a fully connected graph and adaptively learn interaction relationships with the powerful global attention mechanism.

In graph-level tasks like molecular property prediction, numerous Graph Transformers leveraging global attention have achieved remarkable success (Ying et al., 2021; Kreuzer et al., 2021; Wu et al., 2021; Rampášek et al., 2022). This success is largely attributed to their global perception capability. Inspired by the successful applications in graph-level tasks, researchers have attempted to solve the scalability challenge posed by the $O(N^2)$ complexity of the global attention mechanism and make efforts to adapt this mechanism for node-level task, aiming at expanding the receptive field and enhancing the model's expressive ability (Kuang et al., 2022; Zhang et al., 2022; Zhu et al., 2023; Liu et al., 2023a; Wu et al., 2022; Kong et al., 2023; Wu et al., 2023).

Although the global attention module has been recognized as the fundamental unit of Graph Transformer, the following question remains largely unknown:

Does the globalizing property always benefit Graph Transformers?

Understanding the attention mechanism in Graph Transformers, particularly its globalizing property, can provide valuable guidelines and insights for the development of advanced Graph Transformers. In this study, we reveal the over-globalizing problem in Graph Transformers by presenting both empirical evidence and theoretical analysis.

[1]School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, China [2]School of Software, Beihang University, Beijing, China. Correspondence to: Xiao Wang <xiao_wang@buaa.edu.cn>, Chuan Shi <shichuan@bupt.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

In particular, we empirically find that there is an inconsistency between the distribution of learned attention scores across all node pairs and the distribution of nodes that are actually informative, i.e., the global attention mechanism tends to focus on higher-order nodes, while the useful information often appears in lower-order nodes. Despite that higher-order nodes may provide additional information, the current attention mechanism overly focuses on those nodes. Theoretically, we demonstrate that an excessively expanded receptive field can diminish the effectiveness of the global attention mechanism, further implying the existence of the over-globalizing problem.

Once the weakness of the global attention mechanism in Graph Transformers is identified, another question naturally emerges: *how to improve the current global attention mechanism to prevent the over-globalizing problem in Graph Transformers, while still keeping the ability to extract valuable information from high-order nodes?* Usually, one can alleviate this problem by implicitly or explicitly integrating a local module (e.g., GNNs) to complement Graph Transformers (Zhao et al., 2021; Zhang et al., 2022; Kuang et al., 2022; Kong et al., 2023; Liu et al., 2023a; Wu et al., 2023). However, the different properties of local smoothing in GNNs and over-globalizing in Graph Transformers raise a fundamental question about which information will predominantly influence the node representation. Moreover, the prevalent approach of fusing local and global information through linear combination is inadequate and potentially leads to incorrect predictions, even in situations where using either local or global information alone could have achieved accurate predictions.

In this paper, we propose a novel Bi-Level Global Graph Transformer with Collaborative Training (CoBFormer). Specifically, we first partition the graph into distinct clusters with the METIS algorithm. Subsequently, we propose the bi-level global attention (BGA) module, which consists of an intra-cluster Transformer and an inter-cluster Transformer. This module effectively mitigates the over-globalizing problem while keeping a global receptive ability by decoupling the information within intra-clusters and between inter-clusters. To capture the graph structure information neglected by the BGA module, a Graph Convolution Network (GCN) is adopted as the local module. Finally, we propose collaborative training to integrate the information learned by the GCN and BGA modules and boost their performance. We summarize our contributions as follows:

- We demonstrate a crucial phenomenon: Graph Transformers typically yield the over-globalizing problem of attention mechanism for node classification. Both the theoretical analysis and empirical evidence are provided to show that this problem will fundamentally affect Graph Transformers. Our discoveries provide a perspective that offers valuable insights into the improvement of Graph Transformers.

- We propose CoBFormer, a Bi-Level Global Graph Transformer with Collaborative Training, which effectively addresses the over-globalizing problem. Theoretical analysis implies that our proposed collaborative training will improve the model's generalization ability.

- Extensive experiments demonstrate that CoBFormer outperforms the state-of-the-art Graph Transformers and effectively solves the over-globalizing problem.

2. Preliminaries

We denote a graph as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the node set $\mathcal{V}$ contains N nodes and the edge set $\mathcal{E}$ contains E edges. All edges formulate an adjacency matrix $\mathbf{A} = [a_{uv}] \in \{0,1\}^{N\times N}$, where $a_{uv} = 1$ if there exists an edge from node u to v, and 0 otherwise. Graph $\mathcal{G}$ is often associated with a node feature matrix $\mathbf{X} = [\mathbf{x}_u] \in \mathbb{R}^{N\times d}$, where $\mathbf{x}_u$ is a d dimensional feature vector of node u. The label set is denoted as $\mathcal{Y}$. Labels of nodes are represented with a label matrix $\mathbf{Y} = [\mathbf{y}_u] \in \mathbb{R}^{N\times|\mathcal{Y}|}$, where $\mathbf{y}_u$ is the one-hot label of node u. We use bold uppercase letters to represent matrices and bold lowercase letters to represent vectors.

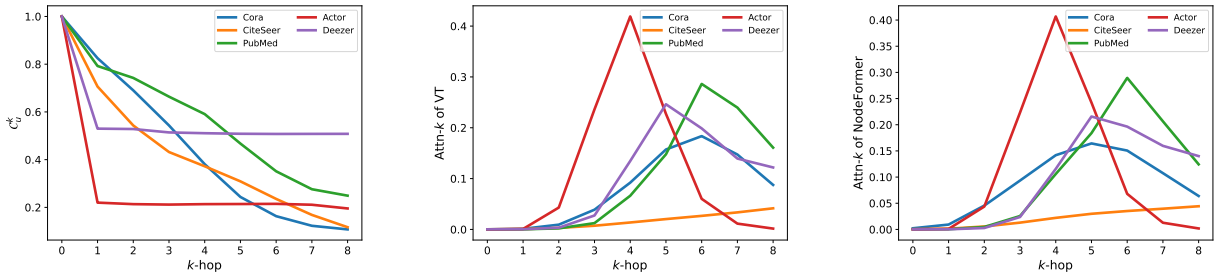
Graph Transformers. Graph Transformers allow each node in a graph to attend to any other nodes by its powerful global attention mechanism as follows:

$$\begin{aligned} \text{Attn}(\mathbf{H}) &= \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{h}}\right)\mathbf{V}, \\ \mathbf{Q} = \mathbf{H}\mathbf{W}_Q, \mathbf{K} &= \mathbf{H}\mathbf{W}_K, \mathbf{V} = \mathbf{H}\mathbf{W}_V, \end{aligned} \tag{1}$$

where $\mathbf{H} \in \mathbb{R}^{N\times h}$ denotes the hidden representation matix and h is the hidden representation dimension. $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{h\times h}$ are trainable weights of linear projection layers. The attention score matrix is $\hat{\mathbf{A}} = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{h}}\right) \in \mathbb{R}^{N\times N}$, containing the attention scores of any node pairs. α_{uv} is the element of $\hat{\mathbf{A}}$, representing the attention score between node u and v. It can be seen that Graph Transformers globally update the node representations by multiplying the attention score matrix $\hat{\mathbf{A}}$ with the node representation matrix $\mathbf{V}$.

3. Over-Globalizing Problem

Empirical observations. In this section, we closely examine the distribution of the attention scores α_{uv} to investigate what information the attention mechanism captures. Specifically, we define the proportion of the k-th hop neighbors



(a) (b) (c)

Figure 1. (a) The average $\mathcal{C}_u^k$ with different k-hop on five real-world datasets. (b) The Attn-k of Vanilla Transformer. (c) The Attn-k of NodeFormer.

sharing the same label with node u as follows:

$$\mathcal{C}_u^k = \frac{|v \in \mathcal{N}^k(u) : \mathbf{y}_u = \mathbf{y}_v|}{|\mathcal{N}^k(u)|}, \tag{2}$$

where $\mathcal{N}^k(u)$ denotes the k-th hop neighbors of node u. Larger $\mathcal{C}_u^k$ indicates a higher proportion of useful nodes in the k-th hop neighbors. Then we denote the average attention scores allocated to the k-th hop neighbors as Attn-k, which is formulated as:

$$\text{Attn-}k = \mathbb{E}_{u \in \mathcal{V}} \sum_{v \in \mathcal{N}^k(u)} \alpha_{uv}. \tag{3}$$

Larger Attn-k implies that the model pays more attention to the k-th hop information.

We present the changes of the average $\mathcal{C}_u^k$ across three homophilic graphs (Cora, CiteSeer and PubMed) and two heterophilic graphs (Actor and Deezer) in Figure 1(a). We can observe that: (1) For homophilic graphs, $\mathcal{C}_u^k$ will gradually decrease as the k increases. (2) For heterophilic graphs, $\mathcal{C}_u^k$ will rapidly decrease when $k = 1$ and then remains nearly unchanged. This demonstrates that homophilic graphs benefit more from the local structure for node classification, while heterophilic graphs gain more information from the global receptive field. Then we visualize the Attn-k of Vanilla Transformer (VT) and NodeFormer (Wu et al., 2022) to check whether the trend of Attn-k is consistent with Figure 1(a). To achieve this visualization, we take the average attention score across each head and plot the results for the first layer. It is worth noting that the subsequent layers will exhibit a similar trend. As can be seen in Figures 1(b) and 1(c), surprisingly, we find that the majority of the attention scores are actually allocated to distant higher-order neighbors, regardless of whether the graphs are homophilic or heterophilic. We identify this phenomenon as the over-globalizing problem in Graph Transformers, underscoring the limitations of the global attention mechanism.

Theoretical analysis. Here we further theoretically explore the impact of the over-globalizing problem in Graph Transformers. Ideally, Graph Transformers would allocate higher attention scores to nodes with similar embeddings, thereby implicitly learning a graph structure that ensures the smoothness of embeddings among adjacent nodes. Consequently, $\|\mathbf{Z} - \hat{\mathbf{A}}\mathbf{Z}\|_F$ would be relatively small (Shuman et al., 2013; Kalofolias, 2016). Here $\mathbf{Z}$ symbolizes the node embeddings. So we employ $\|\mathbf{Z} - \hat{\mathbf{A}}\mathbf{Z}\|_F$ to evaluate the smoothness of the embeddings among adjacent nodes learned by Graph Transformers. A smaller $\|\mathbf{Z} - \hat{\mathbf{A}}\mathbf{Z}\|_F$ indicates a better smoothness, suggesting that Graph Transformers can effectively recognize useful nodes and aggregate information from them, achieving better node classification performance.

Then we investigate the factors influencing $\|\mathbf{Z} - \hat{\mathbf{A}}\mathbf{Z}\|_F$. Before that, we denote $\mathcal{C}_u$ as the proportion of nodes belonging to the same class in the reachable set of node u. If the reachable set of node u is the K-hop neighbors, then $\mathcal{C}_u$ can be formulated as follows:

$$\mathcal{C}_u = \frac{\sum_{k=0}^{K} \mathcal{C}_u^k |\mathcal{N}^k(u)|}{\sum_{k=0}^{K} |\mathcal{N}^k(u)|}. \tag{4}$$

Now we can establish the connection between $\|\mathbf{Z} - \hat{\mathbf{A}}\mathbf{Z}\|_F$, α_{uv} and $\mathcal{C}_u$ as follows:

Theorem 3.1. *For a given node u and a well-trained Graph Transformer, let $\eta_u = \mathbb{E}_{v \in \mathcal{V}, \mathbf{y}_u = \mathbf{y}_v} \exp(\frac{\mathbf{q}_u \mathbf{k}_v^T}{\sqrt{d}})$, $\gamma_u = \mathbb{E}_{v \in \mathcal{V}, \mathbf{y}_u \neq \mathbf{y}_v} \exp(\frac{\mathbf{q}_u \mathbf{k}_v^T}{\sqrt{d}})$. Then, we have:*

$$\begin{aligned} \|\mathbf{Z} - \hat{\mathbf{A}}\mathbf{Z}\|_F &\leq \sqrt{2}L \sum_{u \in \mathcal{V}} \sum_{v \in \mathcal{V}, \mathbf{y}_u \neq \mathbf{y}_v} \alpha_{uv} \\ &= \sqrt{2}L \sum_{u \in \mathcal{V}} \frac{1}{1 + \frac{\mathcal{C}_u}{1 - \mathcal{C}_u} \frac{\eta_u}{\gamma_u}}. \end{aligned} \tag{5}$$

where L is a Lipschitz constant.

The proofs are given in Appendix A.1. Theorem 3.1 indicates that $\|\mathbf{Z} - \hat{\mathbf{A}}\mathbf{Z}\|_F$ is bounded by the sum of the attention scores of node pairs with different labels and negatively correlated with $\mathcal{C}_u$, since η_u and γ_u are constants for a given Graph Transformer. Then we further study the variations of $\mathcal{C}_u$ in Theorem 3.2.

Theorem 3.2. *To analyze the impact of k on $\mathcal{C}_u^k$, we assume that each node has an equal probability $\frac{1}{|\mathcal{Y}|}$ of belonging to any given class. Given the edge homophily $\rho = \frac{|(u,v)\in\mathcal{E}:\mathbf{y}_u=\mathbf{y}_v|}{|\mathcal{E}|}$, $\mathcal{C}_u^k$ can be recursively defined as:*

$$\mathcal{C}_u^k = \begin{cases} 1, & if\ k=0 \\ \rho, & if\ k=1 \\ \frac{1+|\mathcal{Y}|\rho\mathcal{C}_u^{k-1}-\rho-\mathcal{C}_u^{k-1}}{|\mathcal{Y}|-1}. & if\ k=2,3,\cdots \end{cases} \tag{6}$$

And $\mathcal{C}_u^k$ possesses the following properties:

$$\begin{cases} \mathcal{C}_u^{\infty} = \frac{1}{|\mathcal{Y}|} & \\ \mathcal{C}_u^k \geq \mathcal{C}_u^{k+1}, & if\ \rho \geq \frac{1}{|\mathcal{Y}|},\ k=0,1\cdots \\ \mathcal{C}_u^{2k} > \mathcal{C}_u^{2(k+1)}, & if\ \rho < \frac{1}{|\mathcal{Y}|},\ k=0,1\cdots \\ \mathcal{C}_u^{2k+1} < \mathcal{C}_u^{2(k+1)+1}, & if\ \rho < \frac{1}{|\mathcal{Y}|}.\ k=0,1\cdots \end{cases} \tag{7}$$

We provide the proof in Appendix A.2. Theorem 3.2 indicates that in homophilic graphs, where ρ is relatively large, $\mathcal{C}_u^k$ will gradually decrease and converge to $\frac{1}{|\mathcal{Y}|}$, as the k increases. However, in heterophilic graphs, where ρ is relatively small, it will fluctuate around $\frac{1}{|\mathcal{Y}|}$ and eventually converge to $\frac{1}{|\mathcal{Y}|}$. Combining with Theorem 3.1, we find that in homophilic graphs, as the receptive field expands, the gradually decreased $\mathcal{C}_u^k$ will lead to a reduced $\mathcal{C}_u$ and a larger $\|\mathbf{Z}-\hat{\mathbf{A}}\mathbf{Z}\|_F$, implying that an over-expanded receptive field adversely affects the global attention. Conversely, in heterophilic graphs, global attention brings in additional information that cannot be captured within the local neighborhood. Based on Theorem 3.2, we visualize the theoretical variations of $\mathcal{C}_u^k$ in Figure 2. Compared with Figure 1(a), we can find that our theories align well with real-world scenarios. More visualization results of theoretical scenarios are provided in Appendix D.1.

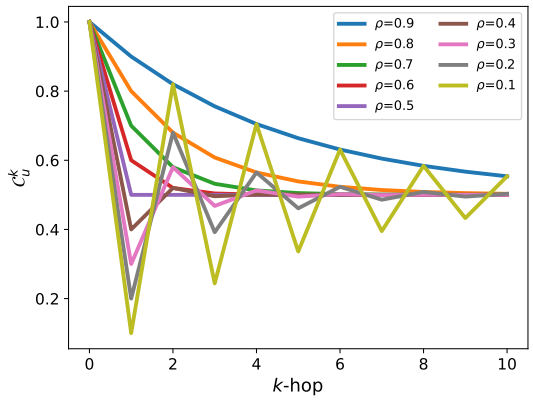


Figure 2. The variations of $\mathcal{C}_u^k$ under various ρ settings for a scenario involving only two classes with uniformly distributed labels.

Experimental analysis. Inspired by Theorem 3.1, we define the Attention Signal/Noise Ratio (Attn-SNR) as the metric to quantify the ability of Graph Transformers to distinguish useful nodes as follows:

Definition 3.3. The Attention Signal/Noise Ratio (Attn-SNR) is:

$$\text{Attn-SNR} = 10\lg\left(\frac{\sum_{\mathbf{y}_u=\mathbf{y}_v}\alpha_{uv}}{\sum_{\mathbf{y}_u\neq\mathbf{y}_v}\alpha_{uv}}\right). \tag{8}$$

For a given Graph Transformer, a smaller Attn-SNR usually implies that the attention mechanism pays more attention to nodes with different labels, which may be caused by the over-globalizing problem. We evaluate Vanilla Transformer and NodeFormer utilizing Attn-SNR and accuracy on Cora and Citeseer. Furthermore, we deliberately improve the Attn-SNR of Vanilla Transformer by doubling the attention scores between nodes sharing the same label, and report its performance. The results are presented in Table 1, indicating that: (1) Vanilla Transformer (VT) typically shows the least Attn-SNR, resulting in the poorest performance. NodeFormer (NF) exhibits a higher Attn-SNR and achieves superior performance. (2) Remarkably, the Denoised Vanilla Transformer (VT-D), artificially directed to achieve higher Attn-SNR, demonstrates better performance than Vanilla Transformer. This is because the over-globalizing problem can be alleviated by doubling the attention scores between nodes with the same label, which are more likely to appear in the local neighborhood, thereby enhancing the model's classification capability.

Table 1. The Attn-SNR and testing accuracy of different models.

Dataset	Metric	VT	NF	VT-D
Cora	Attn-SNR	-6.97	0.43	12.05
	Accuracy	55.18	80.20	82.12
CiteSeer	Attn-SNR	-7.19	-5.09	8.72
	Accuracy	50.72	71.50	61.80

4. The Proposed Method

In this section, we introduce the Bi-Level Global Graph Transformer with Collaborative Training (CoBFormer). An overview of CoBFormer is shown in Figure 3. Specifically, we first use the METIS algorithm (Karypis & Kumar, 1998) to partition the graph into different clusters. Then we propose a novel bi-level global attention (BGA) module, which can decouple the information within intra-clusters and between inter-clusters by an intra-cluster Transformer and an inter-cluster Transformer. Additionally, we incorporate a GCN as the local module to learn the graph structural information. Finally, we propose collaborative training to integrate the information obtained by the GCN and BGA

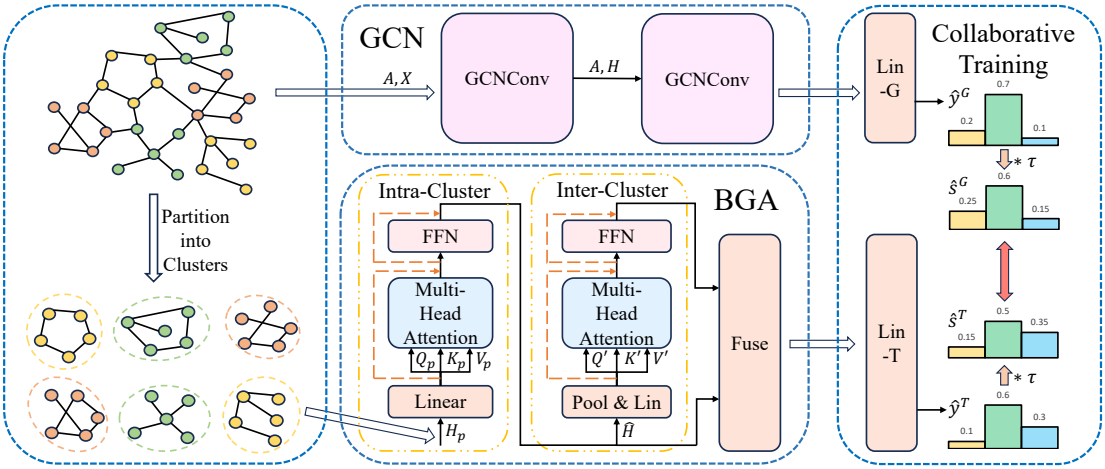


Figure 3. The overall framework of our proposed CoBFormer.

modules and theoretically prove that the generalization ability can be improved with our proposed collaborative training.

4.1. Bi-Level Global Attention Module

Traditional Graph Transformers utilize the global attention mechanism to capture information between any node pairs, causing the over-globalizing problem. Therefore, we need to guarantee that local information can be captured, so as to alleviate the problem. To achieve this goal, we first partition the graph into P non-overlapping clusters using METIS (Karypis & Kumar, 1998). We denote the set of clusters as $\mathcal{P} = \{\mathcal{G}_i\}$, where $\mathcal{G}_i = \{\mathcal{V}_i, \mathcal{E}_i\}$ represents a subgraph of $\mathcal{G}$, satisfying $\bigcup \mathcal{G}_i = \mathcal{G}, \bigcap \mathcal{G}_i = \varnothing$.

The local information is usually within each cluster, so we employ an intra-cluster Transformer. The node features in cluster p are represented as $\mathbf{X}_p \in \mathbb{R}^{\frac{N}{P} \times d}$. We apply an MLP to project the original node features into a latent space as $\mathbf{H}_p^1 = \text{MLP}(\mathbf{X}_p)$. Subsequently, the hidden representations $\mathbf{H}_p^1 \in \mathbb{R}^{\frac{N}{P} \times h}$ are fed into the intra-cluster Transformer to learn the updated hidden representations $\hat{\mathbf{H}}_p^k$:

$$\hat{\mathbf{H}}_p^k = \text{FFN}\left(\text{Softmax}\left(\frac{\mathbf{Q}_p\mathbf{K}_p^T}{\sqrt{h}}\right)\mathbf{V}_p\right), \quad \mathbf{Q}_p = \mathbf{H}_p^k\mathbf{W}_Q^k, \mathbf{K}_p = \mathbf{H}_p^k\mathbf{W}_K^k, \mathbf{V}_p = \mathbf{H}_p^k\mathbf{W}_V^k, \tag{9}$$

where $\mathbf{W}_Q^k, \mathbf{W}_K^k$ and $\mathbf{W}_V^k \in \mathbb{R}^{h \times h}$ are trainable weights of the linear projection layers in the k-th intra-cluster Transformer, and FFN represents a Feed-Forward Neural Network. In each attention block and FFN block, a residual connection (He et al., 2016) and a layer normalization (Ba et al., 2016) are applied.

Subsequently, we apply mean pooling to $\hat{\mathbf{H}}_p^k$ to obtain the cluster representations $\mathbf{P}^k \in \mathbb{R}^{P \times h}$. The p-th row of $\mathbf{P}^k$, represented as $\mathbf{p}_p^k$, is calculated by $\mathbf{p}_p^k = \text{MEAN}(\hat{\mathbf{H}}_p^k) \in \mathbb{R}^{1 \times h}$. Following this, the cluster representations $\mathbf{P}^k$ are fed into the inter-cluster Transformer:

$$\hat{\mathbf{P}}^k = \text{FFN}\left(\text{Softmax}\left(\frac{\mathbf{Q}'\mathbf{K}'^T}{\sqrt{h}}\right)\mathbf{V}'\right), \quad \mathbf{Q}' = \mathbf{P}^k\mathbf{W}_{Q'}^k, \mathbf{K}' = \mathbf{P}^k\mathbf{W}_{K'}^k, \mathbf{V}' = \mathbf{P}^k\mathbf{W}_{V'}^k, \tag{10}$$

where $\mathbf{W}_{Q'}^k, \mathbf{W}_{K'}^k$ and $\mathbf{W}_{V'}^k \in \mathbb{R}^{h \times h}$ are trainable weights of the linear projection layers in the k-th inter-cluster Transformer. Although the inter-cluster Transformer learns the attentions between different clusters, it can approximate the global attention mechanism in Equation (1) and effectively learn the global information as in Proposition 4.1.

Proposition 4.1. *Given $u \in \mathcal{V}_p, v \in \mathcal{V}_q$, along with a well-trained inter-cluster attention score matrix $\dot{\mathbf{A}} \in \mathbb{R}^{P \times P}$. Let $\dot{\alpha}_{pq}$ represent the attention score between clusters p and q. Then the approximate attention score between node u and v can be expressed as $\hat{\alpha}_{uv} = \frac{\dot{\alpha}_{pq}}{|\mathcal{V}_q|}$.*

The proof can be found in Appendix A.4. It can be seen that the attention score α_{uv} of the global attention mechanism can be approximately represented as $\frac{\dot{\alpha}_{pq}}{|\mathcal{V}_q|}$, which can be calculated by our inter-cluster Transformer.

Now with both $\hat{\mathbf{H}}_p^k$ and $\hat{\mathbf{P}}^k$, we concatenate the node representation with its corresponding cluster representation and calculate the output node representations $\mathbf{H}_p^{k+1}$ with a fusion linear layer parameterized by $\mathbf{W}_f$:

$$\mathbf{H}_p^{k+1} = \left(\hat{\mathbf{H}}_p^k \| \mathbf{1}^{\frac{N}{P}} \hat{\mathbf{p}}_p^T\right)\mathbf{W}_f, \tag{11}$$

here $\|$ indicates the concatenation operation, $\mathbf{1}^{\frac{N}{P}}$ is an all-one column vector with a dimensional of $\frac{N}{P}$ and $\hat{\mathbf{p}}_p$ is the hidden representation of cluster p. By decoupling the information within intra-clusters and between inter-clusters,

our BGA module can alleviate the over-globalizing problem while preserving the expressive ability of global attention. Concurrently, our method achieves significant computational and memory efficiency by focusing on intra-cluster and inter-cluster attention. The time and space complexity of our BGA module are $O(\frac{N^2}{P} + P^2)$, reaching $O(N^{\frac{4}{3}})$ under the optimal condition. Note that the efficiency of our BGA module can be further improved by linear attention techniques (Choromanski et al., 2020; Wang et al., 2020; Wu et al., 2023).

4.2. Collaborative Training

With the proposed BGA module capturing intra-cluster and inter-cluser information, we further introduce a GCN as the local module to supplement the graph structure information ignored by the BGA module. Instead of directly employing a linear combination of information from the local module and the global attention module for node classification, we propose a collaborative training approach for the GCN and BGA modules. We denote the labeled node set as $\mathcal{V}_{\mathcal{L}}$ and the unlabeled node set as $\mathcal{V}_{\mathcal{U}}$, with $\mathbf{L}$ and $\mathbf{U}$ representing their respective label matrix. In previous works, a model is trained by predicting the label distribution of the labeled nodes with a cross-entropy loss. However, this method does not guarantee satisfactory performance on unlabeled nodes.

Here, we employ two linear layers, Lin-G and Lin-T, to map the outputs of the GCN and BGA modules onto the label space:

$$\begin{aligned} \hat{\mathbf{Z}}^G &= \text{Lin-G}(\text{GCN}(\mathbf{A}, \mathbf{X})), \\ \hat{\mathbf{Z}}^T &= \text{Lin-T}(\text{BGA}(\mathbf{X}, \mathcal{P})). \end{aligned} \tag{12}$$

Then we use the SoftMax function to calculate the predicted labels and soft labels (Hinton et al., 2015):

$$\begin{aligned} \hat{\mathbf{Y}}^G = \text{SoftMax}(\hat{\mathbf{Z}}^G),\ \hat{\mathbf{Y}}^T = \text{SoftMax}(\hat{\mathbf{Z}}^T), \\ \hat{\mathbf{S}}^G = \text{SoftMax}(\hat{\mathbf{Z}}^G * \tau),\ \hat{\mathbf{S}}^T = \text{SoftMax}(\hat{\mathbf{Z}}^T * \tau), \end{aligned} \tag{13}$$

where τ is a temperature coefficient used to control the smoothness of the soft labels. The objective function can be formulated as:

$$\begin{aligned} \mathcal{L}_{ce} &= -\left(\mathbb{E}_{\mathbf{y}_u, u\in\mathcal{V}_{\mathcal{L}}} \log(\hat{\mathbf{y}}_u^G) + \mathbb{E}_{\mathbf{y}_u, u\in\mathcal{V}_{\mathcal{L}}} \log(\hat{\mathbf{y}}_u^T)\right), \\ \mathcal{L}_{co} &= -\left(\mathbb{E}_{\hat{\mathbf{s}}_u^G, u\in\mathcal{V}_{\mathcal{U}}} \log(\hat{\mathbf{s}}_u^T) + \mathbb{E}_{\hat{\mathbf{s}}_u^T, u\in\mathcal{V}_{\mathcal{U}}} \log(\hat{\mathbf{s}}_u^G)\right), \\ \mathcal{L} &= \alpha * \mathcal{L}_{ce} + (1-\alpha) * \mathcal{L}_{co}. \end{aligned} \tag{14}$$

where $\mathbf{y}_u$ represents the true label of node u. $\hat{\mathbf{y}}_u^G$ and $\hat{\mathbf{y}}_u^T$ are the predicted labels of node u by the GCN and BGA modules, respectively. $\hat{\mathbf{s}}_u^G$ and $\hat{\mathbf{s}}_u^T$ denote the soft labels generated by each. $\mathcal{L}_{ce}$ is the cross-entropy loss, a standard choice for classification tasks. $\mathcal{L}_{co}$ is designed to encourage mutual supervision between the GCN and BGA modules. The parameter α is used to balance the contributions of $\mathcal{L}_{ce}$ and $\mathcal{L}_{co}$.

Now we prove that our proposed collaborative training can improve the generalization ability of our GCN module and BGA module, thereby achieving better classification performance.

Theorem 4.2. *Consider $P(\mathbf{L}, \mathbf{U})$ as the true label distribution, $P_G(\mathbf{L}, \mathbf{U})$ as the predicted label distribution by the GCN, and $P_T(\mathbf{L}, \mathbf{U})$ as the predicted label distribution by the BGA module. The following relations hold:*

$$\begin{aligned} \mathbb{E}_{P(\mathbf{L},\mathbf{U})} \log P_G(\mathbf{L}, \mathbf{U}) =& \mathbb{E}_{P(\mathbf{L})} \log P_G(\mathbf{L}) + \\ & \mathbb{E}_{P_T(\mathbf{U}|\mathbf{L})} \log P_G(\mathbf{U}|\mathbf{L}) - \\ & \text{KL}(P_T(\mathbf{U}|\mathbf{L}) \| P(\mathbf{U}|\mathbf{L})), \\ \mathbb{E}_{P(\mathbf{L},\mathbf{U})} \log P_T(\mathbf{L}, \mathbf{U}) =& \mathbb{E}_{P(\mathbf{L})} \log P_T(\mathbf{L}) + \\ & \mathbb{E}_{P_G(\mathbf{U}|\mathbf{L})} \log P_T(\mathbf{U}|\mathbf{L}) - \\ & \text{KL}(P_G(\mathbf{U}|\mathbf{L}) \| P(\mathbf{U}|\mathbf{L})), \end{aligned} \tag{15}$$

where KL$(\cdot\|\cdot)$ is the Kullback-Leibler divergence.

The proof is given in Appendix A.3. Taking the training process of the GCN module as an example, $\mathbb{E}_{P(\mathbf{L},\mathbf{U})} \log P_G(\mathbf{L}, \mathbf{U})$ is the cross entropy between $P(\mathbf{L}, \mathbf{U})$ and $P_G(\mathbf{L}, \mathbf{U})$. We aim to maximize it so that our model can achieve the best performance on labeled nodes and unlabeled nodes. However, it cannot be maximized directly since the label distribution of unlabeled nodes is unknown. Theorem 4.2 suggests that $\mathbb{E}_{P(\mathbf{L},\mathbf{U})} \log P_G(\mathbf{L}, \mathbf{U})$ can be decomposed into three terms. The first term is the cross entropy between $P(\mathbf{L})$ and $P_G(\mathbf{L})$, which can be maximized by optimizing the $\mathcal{L}_{ce}$. It will ensure a good performance on labeled nodes. The second term is the cross entropy between $P_T(\mathbf{U}|\mathbf{L})$ and $P_G(\mathbf{U}|\mathbf{L})$, which can be maximized by optimizing the $\mathcal{L}_{co}$. This term indicates that we can further improve the performance of the GCN module on unlabeled nodes by collaboratively training with the BGA module. Note that the third term is the Kullback-Leibler divergence between $P_T(\mathbf{U}|\mathbf{L})$ and $P(\mathbf{U}|\mathbf{L})$, which is a constant when we optimize the GCN module and this value will gradually decrease during the optimization process of the BGA module. Therefore, the performance of the GCN module can be improved by the loss in Equation (14). Similarly, the performance of the BGA module can be improved.

5. Experiments

Datasets. We select seven datasets to evaluate, including homophilic graphs, i.e., Cora, CiteSeer, Pubmed (Yang et al., 2016), Ogbn-Arxiv, Ogbn-Products (Hu et al., 2020) and heterophilic graphs, i.e., Actor, Deezer (Lim et al., 2021b). For Cora, CiteSeer, PubMed, we adopt the public split offered by PyG (Fey & Lenssen, 2019). For Ogbn-Arxiv and Ogbn-Products, we use the public splits in OGB (Hu et al., 2020). For Actor and Deezer, we perform five random splits of the nodes into train/valid/test sets, with the ratio of

Table 2. Quantitative results ($\% \pm \sigma$) on node classification.

Dataset	Metric	GCN	GAT	NodeFormer	NAGphormer	SGFormer	CoB-G	CoB-T
Cora	Mi-F1	81.44 ± 0.78	81.88 ± 0.99	80.30 ± 0.66	79.62 ± 0.25	81.48 ± 0.94	84.96 ± 0.34	**85.28 ± 0.16**
	Ma-F1	80.65 ± 0.91	80.56 ± 0.55	79.12 ± 0.66	78.78 ± 0.57	79.28 ± 0.49	83.52 ± 0.15	**84.10 ± 0.28**
CiteSeer	Mi-F1	71.84 ± 0.22	72.26 ± 0.97	71.58 ± 1.74	67.46 ± 1.33	71.96 ± 0.13	**74.68 ± 0.33**	74.52 ± 0.48
	Ma-F1	68.69 ± 0.38	65.67 ± 2.28	67.28 ± 1.87	64.47 ± 1.58	68.49 ± 0.65	69.73 ± 0.45	**69.82 ± 0.55**
PubMed	Mi-F1	79.26 ± 0.23	78.46 ± 0.22	78.96 ± 2.71	77.36 ± 0.96	78.04 ± 0.41	80.52 ± 0.25	**81.42 ± 0.53**
	Ma-F1	79.02 ± 0.19	77.82 ± 0.22	78.14 ± 2.51	76.76 ± 0.91	77.86 ± 0.32	80.02 ± 0.28	**81.04 ± 0.49**
Actor	Mi-F1	30.97 ± 1.21	30.63 ± 0.68	35.42 ± 1.37	34.83 ± 0.95	**37.72 ± 1.00**	31.05 ± 1.02	37.41 ± 0.36
	Ma-F1	26.66 ± 0.82	20.73 ± 1.58	32.37 ± 1.38	32.20 ± 1.11	34.11 ± 2.78	27.01 ± 1.77	**34.96 ± 0.68**
Deezer	Mi-F1	63.10 ± 0.40	62.20 ± 0.41	63.59 ± 2.24	63.71 ± 0.58	66.68 ± 0.47	63.76 ± 0.62	**66.96 ± 0.37**
	Ma-F1	62.07 ± 0.31	60.99 ± 0.56	62.70 ± 2.20	62.06 ± 1.28	65.22 ± 0.68	62.32 ± 0.94	**65.63 ± 0.36**
Arxiv	Mi-F1	71.99 ± 0.14	71.30 ± 0.11	67.98 ± 0.60	71.38 ± 0.20	72.50 ± 0.28	**73.17 ± 0.18**	72.76 ± 0.11
	Ma-F1	51.89 ± 0.19	48.84 ± 0.31	46.24 ± 0.20	51.38 ± 0.47	**52.83 ± 0.31**	52.31 ± 0.40	51.64 ± 0.09
Products	Mi-F1	75.49 ± 0.24	76.19 ± 0.40	70.71 ± 0.27	76.41 ± 0.53	72.54 ± 0.80	78.09 ± 0.16	**78.15 ± 0.07**
	Ma-F1	37.02 ± 0.92	35.15 ± 0.20	30.09 ± 0.02	37.48 ± 0.38	33.72 ± 0.42	**38.21 ± 0.22**	37.91 ± 0.44

50%:25%:25% (Lim et al., 2021b). The detailed statistics of the datasets can be found in Appendix B.

Baselines. We compare our method with five baselines, including two classic GNNs: GCN (Kipf & Welling, 2017) and GAT (Veličković et al., 2018), and three state-of-the-art Graph Transformers: NodeFormer (Wu et al., 2022), NAGphormer (Chen et al., 2022), and SGFormer (Wu et al., 2023). Note that in our proposed CoBFormer, the GCN module (CoB-G) and BGA module (CoB-T) each predict the node label independently, so we report their performance simultaneously. Experimental implementation details are given in Appendix C.

Node Classification Results. Table 2 reports the experimental results on node classification. We select Micro-F1 and Macro-F1 as metrics to conduct a comprehensive performance comparison. We can observe that: (1) Both GCN and BGA modules of CoBFormer outperform all baselines in homophilic graphs by a substantial margin, demonstrating the effectiveness of CoBFormer. (2) In heterophilic graphs, the performance of our BGA module is comparable to, even surpasses, the best baseline, SGFormer. This indicates that our BGA module can successfully capture global information. (3) Traditional Graph Transformers exhibit superior performance on heterophilic graphs when compared with GCN and GAT. However, their advantage in homophilic graphs is relatively limited. This suggests that local information plays a more crucial role in homophilic graphs, whereas global information significantly enhances model performance in heterophilic graphs. These results are consistent with our analysis in Section 3. We further conduct a significance test between our method and SGFormer, the best Transformer-based baseline. The results are provided in Appendix D.2 and they indicate that our method not only significantly outperforms SGFormer in homophilic graphs but also achieves substantial improvements in heterophilic graphs.

Table 3. Test accuracy and GPU memory of various CoBFormer variants. 'V-A' denotes the vanilla global attention. 'B-A' represents the BGA module. 'C-T' indicates whether collaborative training is applied.

Dataset	V-A	B-A	C-T	CoB-G	CoB-T	MEM
Cora	√	×	×	81.44	54.86	0.85G
	√	×	√	83.78	83.82	0.85G
	×	√	×	81.44	68.72	0.38G
	×	√	√	84.96	85.28	0.38G
PubMed	√	×	×	79.26	71.22	8.42G
	√	×	√	80.38	80.36	8.42G
	×	√	×	79.26	74.52	0.50G
	×	√	√	80.52	81.42	0.50G
Deezer	√	×	×	62.07	66.49	20.23G
	√	×	√	63.67	66.86	20.23G
	×	√	×	62.07	66.56	3.97G
	×	√	√	63.76	66.96	3.97G

6. Ablation Studies & Analysis

Ablation Study. We carry out ablation studies on Cora, PubMed, and Deezer to evaluate the two fundamental components of our CoBFormer: the BGA module and the collaborative training approach, where the results are shown in Table 3. Key observations include: (1) The accuracy of our BGA module consistently outperforms vanilla global attention on all datasets, irrespective of the use of collaborative training, demonstrating the effectiveness of our BGA module. (2) Collaborative training leads to significant accuracy improvement in both the GCN and BGA modules, indicating that it enhances the model's generalization ability by encouraging mutual learning. (3) The BGA module significantly reduces GPU memory, addressing scalability concerns. Specifically, GPU memory usage is largely reduced by 94% for PubMed and 80% for Deezer.

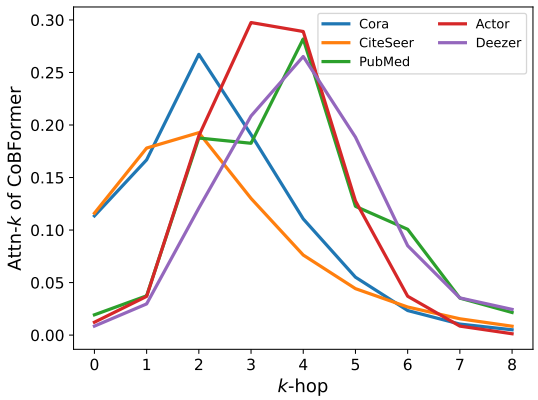


Figure 4. The Attn-k of CoBFormer.

Over-Globalizing Problem. To demonstrate our CoBFormer's ability to alleviate the over-globalizing problem, we visualize the Attn-k of our CoBFormer in Figure 4. Compared with Figures 1(b) and 1(c), CoBFormer allocates more attention scores in the local region than Vanilla Transformer and NodeFormer, indicating that our BGA module can effectively alleviate the over-globalizing problem by decoupling the intra-cluster information and inter-cluster information.

We further calculate the Attn-SNR and test accuracy to show our model's capabilities to distinguish useful nodes and extract valuable information, where the results are shown in Figure 5. Obviously, the CoB-T significantly improves the Attn-SNR and substantially boosts performance on the Cora, CiteSeer, and PubMed. It underscores that our CoBFormer can effectively mitigate the over-globalizing problem. For Actor and Deezer, our CoB-T achieves performance comparable to that of VT, implying that our CoBFormer can effectively capture global information.

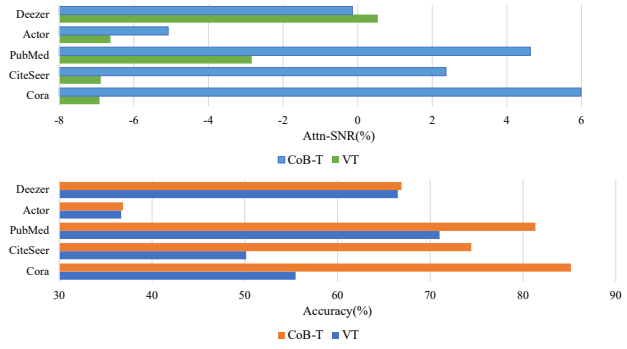


Figure 5. The Attn-SNR and Accuracy of VT and our CoBFormer.

Parameter Study. We analyze the key parameters: the collaborative learning strength coefficient α, the temperature coefficient τ, and the number of clusters P. First, we vary α in $\{1.0, 0.9, 0.8, 0.7\}$ and report the best performance in

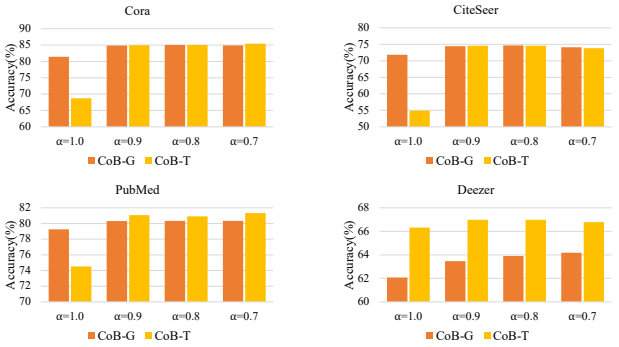


Figure 6. The average test accuracy of CoBFormer for different α.

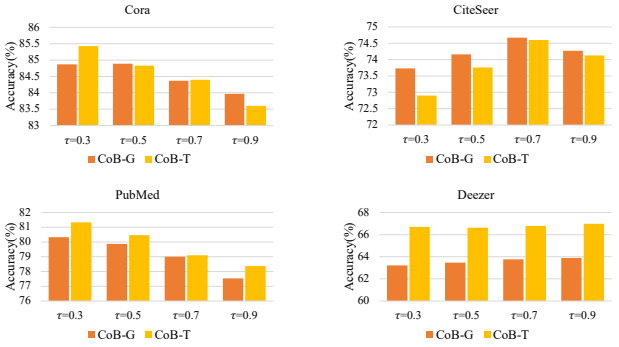


Figure 7. The average test accuracy of CoBFormer for different τ.

Figure 6. Our results show that our model achieves notable performance improvements for all values except when $\alpha = 1$, in which case the collaborative training method is not employed. Furthermore, it exhibits consistent performance across the other α values, underscoring the effectiveness and robustness of our collaborative training approach. Next, we fix the optimal α and report the best performance for various τ values in $\{0.9, 0.7, 0.5, 0.3\}$. Figure 7 suggests that the choice of τ significantly impacts performance, emphasizing the importance of selecting an appropriate τ for optimal results. Finally, we fix the best α and τ values and select different numbers of clusters to report the classification accuracy in Figure 8. The results indicate that

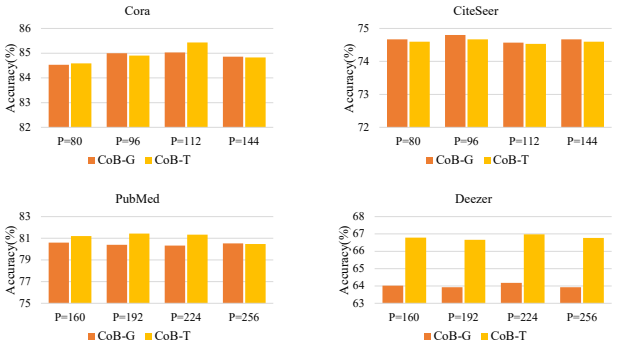


Figure 8. The average test accuracy of CoBFormer for different P.

on Deezer, performance changes minimally with different cluster counts, whereas for other datasets, optimal cluster selection is vital for peak performance. Nevertheless, the model typically surpasses others even without ideal cluster configurations, showcasing its robustness across various cluster counts.

7. Related Work

Graph Neural Networks. Representative GNNs, such as GCN (Kipf & Welling, 2017) and GAT (Veličković et al., 2018), leverage a message-passing mechanism to recursively aggregate neighbor information. However, due to over-smoothing (Li et al., 2018; Nt & Maehara, 2019; Oono & Suzuki, 2020) and over-squashing (Topping et al., 2021; Deac et al., 2022) problems, GNNs typically cannot stack multiple layers to capture information from distant nodes. Moreover, early designs of GNNs largely rely on the homophily assumption (McPherson et al., 2001) that nodes of the same type are more likely to be connected. Although some GNNs are designed for heterophilic graphs (Pei et al., 2020; Lim et al., 2021a; Chien et al., 2021; Bo et al., 2021), they still suffer from the same issues of over-smoothing and over-squashing, resulting in a limited receptive field.

Graph Transformers. Transformers (Vaswani et al., 2017), benefiting from their global attention, naturally construct a fully connected graph with learnable edge weights, offering a novel approach to address the issues of over-smoothing and over-squashing in GNNs. Extensive works have achieved remarkable success in graph-level tasks (Ying et al., 2021; Kreuzer et al., 2021; Wu et al., 2021; Rampášek et al., 2022). This success is primarily attributed to their global perception capability, which is vital for graph-level tasks. Influenced by the success in graph-level tasks, researchers are now exploring the integration of the global attention mechanism into node-level tasks (Kuang et al., 2022; Zhang et al., 2022; Zhu et al., 2023; Liu et al., 2023a; Wu et al., 2022; Kong et al., 2023; Wu et al., 2023). These approaches aim to broaden the receptive field in large-scale graphs and amplify the expressive potential of the models.

8. Conclusion

In this paper, we discover the over-globalizing problem in Graph Transformers by presenting the theoretical insights and empirical results. We then propose CoBFormer, a bi-level global graph transformer with collaborative training, aiming at alleviating the over-globalizing problem and improving the generalization ability. Extensive experiments verify the effectiveness of CoBFormer.

Acknowledgements

This work is supported in part by the National Natural Science Foundation of China (No. U20B2045, 62322203, 62172052, 62192784, U22B2038).

Impact Statement

This paper presents work whose goal is to advance the field of Graph Machine Learning and will promote the application of graph machine learning in large-scale graph data mining, such as enhancing social network recommendations and reducing traffic network congestion. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Bo, D., Wang, X., Shi, C., and Shen, H. Beyond low-frequency information in graph convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 3950–3957, 2021.

Chen, F., Zhang, Y., Chen, L., Meng, X., Qi, Y., and Wang, J. Dynamic traveling time forecasting based on spatial-temporal graph convolutional networks. *Frontiers of Computer Science*, 17(6):176615, 2023.

Chen, J., Gao, K., Li, G., and He, K. Nagphormer: A tokenized graph transformer for node classification in large graphs. In *The Eleventh International Conference on Learning Representations*, 2022.

Chien, E., Peng, J., Li, P., and Milenkovic, O. Adaptive universal generalized pagerank graph neural network. In *International Conference on Learning Representations*, 2021.

Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

Deac, A., Lackenby, M., and Veličković, P. Expander graph propagation. In *Learning on Graphs Conference*, pp. 38–1. PMLR, 2022.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer,

M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Fey, M. and Lenssen, J. E. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019.

Hamilton, W., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., and Leskovec, J. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

Kalofolias, V. How to learn a graph from smooth signals. In *Artificial intelligence and statistics*, pp. 920–929. PMLR, 2016.

Karypis, G. and Kumar, V. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on scientific Computing*, 20(1):359–392, 1998.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

Kong, K., Chen, J., Kirchenbauer, J., Ni, R., Bruss, C. B., and Goldstein, T. Goat: A global transformer on large-scale graphs. In *International Conference on Machine Learning*, pp. 17375–17390. PMLR, 2023.

Kreuzer, D., Beaini, D., Hamilton, W., Létourneau, V., and Tossou, P. Rethinking graph transformers with spectral attention. *Advances in Neural Information Processing Systems*, 34:21618–21629, 2021.

Kuang, W., WANG, Z., Li, Y., Wei, Z., and Ding, B. Coarformer: Transformer for large graph via graph coarsening, 2022.

Li, Q., Han, Z., and Wu, X.-M. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Lim, D., Hohne, F. M., Li, X., Huang, S. L., Gupta, V., Bhalerao, O. P., and Lim, S.-N. Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021a.

Lim, D., Li, X., Hohne, F., and Lim, S.-N. New benchmarks for learning on non-homophilous graphs. *arXiv preprint arXiv:2104.01404*, 2021b.

Liu, C., Zhan, Y., Ma, X., Ding, L., Tao, D., Wu, J., and Hu, W. Gapformer: Graph transformer with graph pooling for node classification. In Elkind, E. (ed.), *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pp. 2196–2205. International Joint Conferences on Artificial Intelligence Organization, 8 2023a. Main Track.

Liu, Y., Yang, C., Zhao, T., Han, H., Zhang, S., Wu, J., Zhou, G., Huang, H., Wang, H., and Shi, C. Gammagl: A multi-backend library for graph neural networks. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2861–2870, 2023b.

McPherson, M., Smith-Lovin, L., and Cook, J. M. Birds of a feather: Homophily in social networks. *Annual review of sociology*, 27:415–444, 2001.

Nt, H. and Maehara, T. Revisiting graph neural networks: All we have is low-pass filters. *arXiv preprint arXiv:1905.09550*, 2019.

Oono, K. and Suzuki, T. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations*, 2020.

Pei, H., Wei, B., Chang, K. C.-C., Lei, Y., and Yang, B. Geom-gcn: Geometric graph convolutional networks. In *International Conference on Learning Representations*, 2020.

Rampášek, L., Galkin, M., Dwivedi, V. P., Luu, A. T., Wolf, G., and Beaini, D. Recipe for a general, powerful, scalable graph transformer. *Advances in Neural Information Processing Systems*, 35:14501–14515, 2022.

Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., and Vandergheynst, P. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE signal processing magazine*, 30(3):83–98, 2013.

Topping, J., Di Giovanni, F., Chamberlain, B. P., Dong, X., and Bronstein, M. M. Understanding over-squashing and bottlenecks on graphs via curvature. *arXiv preprint arXiv:2111.14522*, 2021.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks. In *International Conference on Learning Representations*, 2018.

Wang, S., Li, B. Z., Khabsa, M., Fang, H., and Ma, H. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

Wu, Q., Zhao, W., Li, Z., Wipf, D. P., and Yan, J. Nodeformer: A scalable graph structure learning transformer for node classification. *Advances in Neural Information Processing Systems*, 35:27387–27401, 2022.

Wu, Q., Zhao, W., Yang, C., Zhang, H., Nie, F., Jiang, H., Bian, Y., and Yan, J. Simplifying and empowering transformers for large-graph representations. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Wu, Z., Jain, P., Wright, M., Mirhoseini, A., Gonzalez, J. E., and Stoica, I. Representing long-range context for graph neural networks with global attention. *Advances in Neural Information Processing Systems*, 34:13266–13279, 2021.

Yang, C., Sun, M., Zhao, W. X., Liu, Z., and Chang, E. Y. A neural network approach to jointly modeling social networks and mobile trajectories. *ACM Transactions on Information Systems (TOIS)*, 35(4):1–28, 2017.

Yang, Z., Cohen, W., and Salakhudinov, R. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, pp. 40–48. PMLR, 2016.

Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y., and Liu, T.-Y. Do transformers really perform badly for graph representation? *Advances in Neural Information Processing Systems*, 34:28877–28888, 2021.

Yu, X., Liu, Z., Fang, Y., and Zhang, X. Learning to count isomorphisms with graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 4845–4853, 2023.

Zhang, Z., Liu, Q., Hu, Q., and Lee, C.-K. Hierarchical graph transformer with adaptive node sampling. *Advances in Neural Information Processing Systems*, 35: 21171–21183, 2022.

Zhao, J., Li, C., Wen, Q., Wang, Y., Liu, Y., Sun, H., Xie, X., and Ye, Y. Gophormer: Ego-graph transformer for node classification. *arXiv preprint arXiv:2110.13094*, 2021.

Zhu, W., Wen, T., Song, G., Ma, X., and Wang, L. Hierarchical transformer for scalable graph learning. In Elkind, E. (ed.), *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pp. 4702–4710. International Joint Conferences on Artificial Intelligence Organization, 8 2023. Main Track.

A. Proofs

A.1. Proof of Theorem 3.1

Proof: To prove Theorem 3.1, we assume the existence of a linear classifier, parameterized by $\mathbf{W}_C$, which satisfies the condition $\mathbf{Z}\mathbf{W}_C = \mathbf{Y}$. Consequently, this leads to $\mathbf{Z} = \mathbf{Y}\mathbf{W}_C^{-1}$.

$$\begin{aligned}
\|\mathbf{Z} - \hat{\mathbf{A}}\mathbf{Z}\|_F &\leq \sum_{u\in\mathcal{V}} \|\mathbf{z}_u - \sum_{v\in\mathcal{V}} \alpha_{uv}\mathbf{z}_v\|_2 && \cdots \sqrt{a+b+c} \leq \sqrt{a}+\sqrt{b}+\sqrt{c} \\
&= \sum_{u\in\mathcal{V}} \| \sum_{v\in\mathcal{V}} \alpha_{uv}(\mathbf{z}_u - \mathbf{z}_v)\|_2 && \cdots \sum_{v\in\mathcal{V}} \alpha_{uv} = 1 \\
&\leq \sum_{u,v\in\mathcal{V}} \alpha_{u,v}\|\mathbf{z}_u - \mathbf{z}_v\|_2 && \cdots \text{The Triangle Inequality} \\
&\leq L \sum_{u,v\in\mathcal{V}} \alpha_{u,v}\|\mathbf{y}_u - \mathbf{y}_v\|_2 && \cdots \text{Lipschitz Continuous} \\
&= L\sum_{u\in\mathcal{V}} \sum_{v\in\mathcal{V}, y_v = y_u} \alpha_{u,v}\|\mathbf{y}_u - \mathbf{y}_v\|_2 + L\sum_{u\in\mathcal{V}} \sum_{v\in\mathcal{V}, y_v \neq y_u} \alpha_{u,v}\|\mathbf{y}_u - \mathbf{y}_v\|_2 \\
&= \sqrt{2}L \sum_{u\in\mathcal{V}} \sum_{v\in\mathcal{V}, y_v \neq y_u} \alpha_{u,v}.
\end{aligned} \tag{16}$$

L is a Lipschitz constant. For a given node u and a well trained Graph Transformer, let $\eta_u = \mathbb{E}_{v\in\mathcal{V}, \mathbf{y}_u = \mathbf{y}_v} \exp(\frac{\mathbf{q}_u \mathbf{k}_v^T}{\sqrt{d}})$, $\gamma_u = \mathbb{E}_{v\in\mathcal{V}, \mathbf{y}_u \neq \mathbf{y}_v} \exp(\frac{\mathbf{q}_u \mathbf{k}_v^T}{\sqrt{d}})$. Let $\mathcal{C}_u$ represent the proportion of nodes that are reachable by node u and belong to the same class. Then the theory can be rewritten as:

$$\begin{aligned}
\|\mathbf{Z} - \hat{\mathbf{A}}\mathbf{Z}\|_F &\leq \sqrt{2}L \sum_{u\in\mathcal{V}} \sum_{v\in\mathcal{V}, y_v \neq y_u} \alpha_{uv} \\
&= \sqrt{2}L \sum_{u\in\mathcal{V}} \frac{\sum_{v\in\mathcal{V}, y_v \neq y_u} \gamma_u}{\sum_{v\in\mathcal{V}, y_v = y_u} \eta_u + \sum_{v\in\mathcal{V}, y_v \neq y_u} \gamma_u} \\
&= \sqrt{2}L \sum_{u\in\mathcal{V}} \frac{N(1-\mathcal{C}_u)\gamma_u}{N\mathcal{C}_u\eta_u + N(1-\mathcal{C}_u)\gamma_u} \\
&= \sqrt{2}L \sum_{u\in\mathcal{V}} \frac{1}{1 + \frac{\mathcal{C}_u}{1-\mathcal{C}_u}\frac{\eta_u}{\gamma_u}}.
\end{aligned} \tag{17}$$

A.2. Proof of Theorem 3.2

Proof: We outline the proof of Theorem 3.2 through the following several steps:

- When $k = 0$ or $k = 1$, the equation is obviously true.
- When $k > 1$, the derivation is as follows:

$$\begin{aligned}
\mathcal{C}_u^k &= \mathcal{C}_u^{k-1}\rho + \frac{(1-\mathcal{C}_u^{k-1})(1-\rho)}{|\mathcal{Y}| - 1} \\
&= \frac{\mathcal{C}_u^{k-1}\rho(|\mathcal{Y}|-1) + (1-\mathcal{C}_u^{k-1})(1-\rho)}{|\mathcal{Y}|-1} \\
&= \frac{1 + |\mathcal{Y}|\rho\mathcal{C}_u^{k-1} - \rho - \mathcal{C}_u^{k-1}}{|\mathcal{Y}|-1}.
\end{aligned} \tag{18}$$

- Then, we have:

$$\begin{aligned}\mathcal{C}_u^{k+1} - \mathcal{C}_u^k &= \frac{1 + |\mathcal{Y}|\rho\mathcal{C}_u^k - \rho - \mathcal{C}_u^k}{|\mathcal{Y}| - 1} - \mathcal{C}_u^k \\ &= \frac{1 + |\mathcal{Y}|\rho\mathcal{C}_u^k - \rho - \mathcal{C}_u^k|\mathcal{Y}|}{|\mathcal{Y}| - 1} \\ &= \frac{(1-\rho)(1 - \mathcal{C}_u^k|\mathcal{Y}|)}{|\mathcal{Y}| - 1}.\end{aligned} \tag{19}$$

  Notice that when $\mathcal{C}_u^k \geq \frac{1}{|\mathcal{Y}|}$, the right side of the equation becomes less than 0, leading to $\mathcal{C}_u^k \geq \mathcal{C}_u^{k+1}$. Conversely, when $\mathcal{C}_u^k < \frac{1}{|\mathcal{Y}|}$, the right side is greater than 0, resulting in $\mathcal{C}_u^k < \mathcal{C}_u^{k+1}$.

- Now, let us study the relation between $\mathcal{C}_u^k$ and $\frac{1}{|\mathcal{Y}|}$.

$$\begin{aligned}\mathcal{C}_u^k - \frac{1}{|\mathcal{Y}|} &= \frac{1 + |\mathcal{Y}|\rho\mathcal{C}_u^{k-1} - \rho - \mathcal{C}_u^{k-1}}{|\mathcal{Y}| - 1} - \frac{1}{|\mathcal{Y}|} \\ &= \frac{(|\mathcal{Y}|\rho - 1)(|\mathcal{Y}|\mathcal{C}_u^{k-1} - 1)}{|\mathcal{Y} - 1| \cdot |\mathcal{Y}|}.\end{aligned} \tag{20}$$

  When $\rho \geq \frac{1}{|\mathcal{Y}|}$ and $\mathcal{C}_u^{k-1} \geq \frac{1}{|\mathcal{Y}|}$, $\mathcal{C}_u^k$ will also be greater than $\frac{1}{|\mathcal{Y}|}$. This is always true when $\rho \geq \frac{1}{|\mathcal{Y}|}$. However, if $\rho < \frac{1}{|\mathcal{Y}|}$, then $\mathcal{C}_u^k$ will be greater than $\frac{1}{|\mathcal{Y}|}$ if $\mathcal{C}_u^{k-1} < \frac{1}{|\mathcal{Y}|}$, and less than $\frac{1}{|\mathcal{Y}|}$ if $\mathcal{C}_u^{k-1} > \frac{1}{|\mathcal{Y}|}$.

- Moreover, when we take into account both $\mathcal{C}_u^k$ and $\mathcal{C}_u^{k+2}$, we observe the following:

$$\begin{aligned}\mathcal{C}_u^{k+2} - \mathcal{C}_u^k &= \mathcal{C}_u^{k+2} - \mathcal{C}_u^{k+1} + \mathcal{C}_u^{k+1} - \mathcal{C}_u^k \\ &= \frac{(1-\rho)(1 - \mathcal{C}_u^{k+1}|\mathcal{Y}|)}{|\mathcal{Y}| - 1} + \frac{(1-\rho)(1 - \mathcal{C}_u^k|\mathcal{Y}|)}{|\mathcal{Y}| - 1} \\ &= \frac{(1-\rho)(2 - (\mathcal{C}_u^k + \mathcal{C}_u^{k+2})|\mathcal{Y}|)}{|\mathcal{Y}| - 1}.\end{aligned} \tag{21}$$

  Note that both $\mathcal{C}_u^k$ and $\mathcal{C}_u^{k+2}$ are either greater than $\frac{1}{|\mathcal{Y}|}$ or less than $\frac{1}{|\mathcal{Y}|}$. When they are greater than $\frac{1}{|\mathcal{Y}|}$, $\mathcal{C}_u^{k+2}$ will be less than $\mathcal{C}_u^k$. Conversely, when they are less than $\frac{1}{|\mathcal{Y}|}$, $\mathcal{C}_u^{k+2}$ will be greater than $\mathcal{C}_u^k$.

In summary, Theorem 3.2 is proven through the aforementioned derivations.

A.3. Proof of Theorem 4.2

Proof: We prove Theorem 4.2 using a technique of variational inference.

$$\begin{aligned}\mathbb{E}_{P(\mathbf{L},\mathbf{U})} \log P_G(\mathbf{L},\mathbf{U}) &= \int_{\mathbf{L},\mathbf{U}} P(\mathbf{L})P(\mathbf{U}|\mathbf{L}) \log P_G(\mathbf{L},\mathbf{U})\, d\mathbf{L}d\mathbf{U} \\ &= \int_{\mathbf{L},\mathbf{U}} P(\mathbf{L})P(\mathbf{U}|\mathbf{L}) \log \frac{P_G(\mathbf{L},\mathbf{U})P_T(\mathbf{U}|\mathbf{L})}{P_T(\mathbf{U}|\mathbf{L})}\, d\mathbf{L}d\mathbf{U} \\ &\approx \int_{\mathbf{L},\mathbf{U}} P(\mathbf{L})P_T(\mathbf{U}|\mathbf{L}) \log \frac{P_G(\mathbf{L},\mathbf{U})P(\mathbf{U}|\mathbf{L})}{P_T(\mathbf{U}|\mathbf{L}))}\, d\mathbf{L}d\mathbf{U} \\ &= \mathbb{E}_{P(\mathbf{L})P_T(\mathbf{U}|\mathbf{L})} \log \frac{P_G(\mathbf{L})P_G(\mathbf{U}|\mathbf{L})P(\mathbf{U}|\mathbf{L})}{P_T(\mathbf{U}|\mathbf{L})} \\ &= \mathbb{E}_{P(\mathbf{L})} \log P_G(\mathbf{L}) + \mathbb{E}_{P_T(\mathbf{U}|\mathbf{L})} \log P_G(\mathbf{U}|\mathbf{L}) + \mathbb{E}_{P_T(\mathbf{U}|\mathbf{L})} \log \frac{P(\mathbf{U}|\mathbf{L})}{P_T(\mathbf{U}|\mathbf{L})} \\ &= \mathbb{E}_{P(\mathbf{L})} \log P_G(\mathbf{L}) + \mathbb{E}_{P_T(\mathbf{U}|\mathbf{L})} \log P_G(\mathbf{U}|\mathbf{L}) - \mathrm{KL}(P_T(\mathbf{U}|\mathbf{L}) \| P(\mathbf{U}|\mathbf{L})).\end{aligned} \tag{22}$$

The proof of the latter follows by the same reasoning.

A.4. Proof of Proposition 4.1

Proof: Through the inter-cluster Transformer and the concatenation operation, a node u in cluster p is enabled to access information from cluster q by computing $\dot{\alpha}_{pq}\mathbf{v}'_q$. Here, $\mathbf{v}'_q$ is defined as $\mathbf{p}_p^k\mathbf{W}_{V'}^k$, where $\mathbf{p}_p^k$ is derived through a mean pooling operation, specifically $\mathbf{p}_p^k = \frac{\sum_{v\in\mathcal{V}_q}\mathbf{h}_v}{|\mathcal{V}_q|}$. Consequently, we can express the process as follows:

$$\begin{aligned}
&\dot{\alpha}_{pq}\mathbf{v}'_q \\
=&\dot{\alpha}_{pq}\mathbf{p}_p^k\mathbf{W}_{V'}^k \\
=&\dot{\alpha}_{pq}\frac{\sum_{v\in\mathcal{V}_q}\mathbf{h}_v}{|\mathcal{V}_q|}\mathbf{W}_{V'}^k \\
=&\sum_{v\in\mathcal{V}_q}\frac{\dot{\alpha}_{pq}}{|\mathcal{V}_q|}\mathbf{h}_v\mathbf{W}_{V'}^k
\end{aligned} \tag{23}$$

Equation (23) elucidates that node u can retrieve information from node v in cluster p by calculating $\frac{\dot{\alpha}_{pq}}{|\mathcal{V}_q|}\mathbf{h}_v\mathbf{W}_{V'}^k$, where the $\frac{\dot{\alpha}_{pq}}{|\mathcal{V}_q|}$ represents the approximate attention score between nodes u and v.

B. Dataset Statistics

The detailed dataset statistics are listed in Table 4.

Table 4. The detailed dataset statistics.

Dataset	#Nodes	#Edges	#Feats	Edge hom	#Classes
Cora	2,708	5,429	1,433	0.83	7
CiteSeer	3,327	4,732	3,703	0.72	6
PubMed	19,717	44,338	500	0.79	3
Actor	7,600	26,752	931	0.22	5
Deezer	28,281	92,752	31,241	0.52	2
Ogbn-Arxiv	169,343	1,166,343	128	0.63	40
Ogbn-Products	2,449,029	61,859,140	100	0.81	47

C. Experiments Details

In this section, we provide a detailed description of the experimental setup for the empirical results in Section 5.

C.1. Training Manner

We train NAGphormer on all datasets using a mini-batch manner as in the official implementation. For the other baselines and our CoBFormer, we employ a full-batch training method on all datasets except Ogbn-Products, which is too large for all models. Instead, we partition Ogbn-Products into clusters and randomly select several clusters to form a subgraph for each training/evaluation step, ensuring that the size of the combined subgraph remains smaller than our predetermined batch size. We set the batch size to 100,000 for NodeFormer and 150,000 for the other models.

C.2. CoBFormer

We implement our CoBFormer using only one BGA module, as SGFormer has demonstrated that a single global attention layer suffices to achieve optimal expressive capability (Wu et al., 2023). The hyperparameters are selected through grid search within the following search space:

- learning rate within $\{5e-4, 1e-3, 5e-3, 1e-2, 5e-2\}$.
- GCN layers within $\{2, 3\}$.

- weight decay of GCN within $\{1e-4, 5e-4, 1e-3, 5e-3, 1e-2\}$.
- weight decay of BGA within $\{1e-5, 5e-5, 1e-4, 5e-4, 1e-3\}$.
- number of clusters:
  - For Cora, CiteSeer and Actor: $\{80, 96, 112, 144\}$.
  - For PubMed and Deezer: $\{160, 192, 224, 256\}$.
  - For Ogbn-Arxiv: $\{1536, 2048, 2560, 3072\}$.
  - For Ogbn-Products: $\{8192, 16384\}$.
- α within: $\{0.9, 0.8, 0.7\}$
- τ within: $\{0.9, 0.7, 0.5, 0.3\}$

We fix the dropout ratio of the GCN module at 0.5 and the dropout ratio of the BGA module at 0.1. Layer Normalization is applied to each attention block and FFN block.

We employ a full-batch training method on all datasets except Ogbn-Products. We partition Ogbn-Products into clusters and randomly select several clusters to form a subgraph for each training/evaluation step, ensuring that the size of the combined subgraph remains smaller than our predetermined batch size.

The Attn-SNR is computed using the intra-cluster attention scores.

C.3. Baselines

We implement GCN and GAT by PyG (Fey & Lenssen, 2019) and refer to GammaGL (Liu et al., 2023b). For the other baselines, we utilize the official repositories. The URLs of these repositories are listed below:

- NAGPhormer: https://github.com/JHL-HUST/NAGphormer
- NodeFormer: https://github.com/qitianwu/NodeFormer
- SGFormer: https://github.com/qitianwu/SGFormer

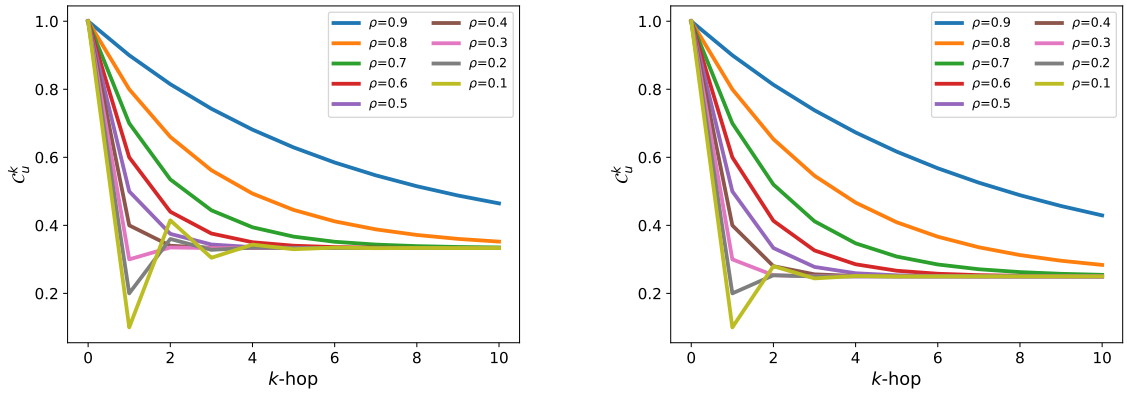
The key hyper-parameters of these baselines are selected from the searching space as follows:

- GAT: number of heads within $\{1, 2, 4, 8, 16\}$.
- NAGPhormer: number of heads within $\{1, 2, 4, 8\}$ and hops within $\{3, 5, 7, 10, 15\}$.
- NodeFormer: rb_order within $\{1, 2, 3\}$, edge regularization weight within $\{0, 1\}$, number of heads within $\{1, 2, 4\}$ and dropout ratio within $\{0, 0.2, 0.3, 0.5\}$.
- SGFormer: α within $\{0.5, 0.8\}$ and dropout ratio within $\{0, 0.2, 0.3, 0.5\}$.

D. More Empirical Results

D.1. More Visualization Results

In this section, we will provide additional visualization results for the theoretical variations of $\mathcal{C}_u^k$ when $|\mathcal{Y}| = 3$ and $|\mathcal{Y}| = 4$, as shown in Figures 9(a) and 9(b).



(a) (b)

Figure 9. Visualization results for Equation (6) when $|\mathcal{Y}| = 3$ and $|\mathcal{Y}| = 4$.

D.2. Results of Significance Test

We conduct a significance test between our method and SGFormer (the best Transformer-based baseline) on four datasets. Specifically, for each dataset, we run each method 50 times and record the accuracy of each run. We then calculate the p-value to assess the statistical significance of the improvements observed with our method. The results for three homophilic graphs are presented in the table below:

Table 5. Results of significance test on three homophilic graphs.

Dataset	p-value(CoB-G vs SGFormer)	p-value(CoB-T vs SGFormer)
Cora	1.39e-41	3.67e-38
CiteSeer	5.52e-43	1.23e-43
PubMed	2.95e-47	1.49e-52

The extremely low p-value indicates that our method significantly outperforms the baseline SGFormer in homophilic graphs. Additionally, we have calculated the p-value comparing the accuracies of CoB-T and SGFormer on the Deezer dataset, which is 2.7e-3. This result also suggests a significant improvement of our method in heterophilic graphs.

E. About Local Module

The over-globalizing problem demonstrated in Figures 1(b) and 1(c) suggests that a single global attention mechanism tends to allocate most attention scores to distant nodes, resulting in an excessive focus on distant information while neglecting local details. Consequently, integrating a local module to enhance the capture of local information is necessary. Specifically, Gophormer (Zhao et al., 2021), ANS-GT (Zhang et al., 2022), NodeFormer (Wu et al., 2022), and Gapformer (Liu et al., 2023a) modify the global attention matrix by incorporating the adjacency matrix, thus preserving message passing within the original topological structure. These methods implicitly employ a GNN to capture local information. Coarformer (Kuang et al., 2022) and SGFormer (Wu et al., 2023) explicitly use a GCN as their local module. GOAT (Kong et al., 2023) and HSGT (Zhu et al., 2023) learn local information through neighbor sampling, a technique also utilized by Gophormer and ANS-GT. NAGphormer (Chen et al., 2022), another neighbor sampling-based method, employs the aggregated features of

the k-hop neighborhood as tokens.

F. A case of Fusion Issue

In this section, we will provide a simple example to illustrate the issue we mentioned in Section 4.2. As is shown in Figure 10, in this illustrative example, the GCN accurately predicts the node as belonging to the true class 2, whereas the Transformer fails, assigning a high probability to class 3 instead. As a result of the linear combination, the node is incorrectly predicted to be in class 3.

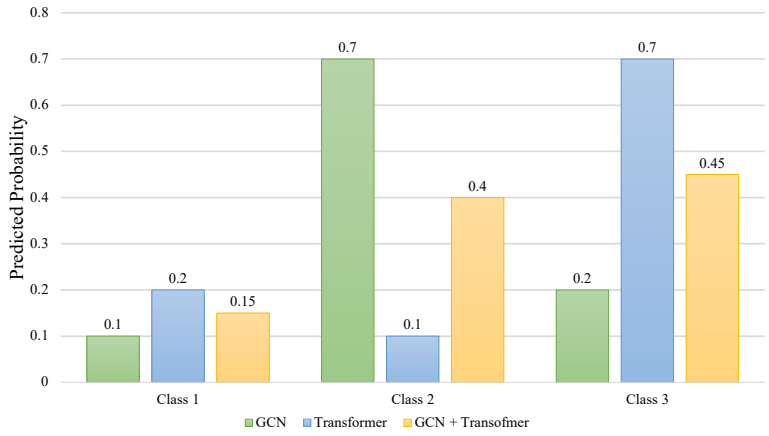


Figure 10. A graphic depiction highlighting the limitations of linear combination is presented.