ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240842

Learning a Mini-Batch Graph Transformer via Two-Stage Interaction Augmentation

Wenda Li [a,b,c,†], **Kaixuan Chen** [a,c,†], **Shunyu Liu** [a,c,†], **Tongya Zheng**[d,e], **Wenjie Huang**[a,c] **and Mingli Song**[a,c,*]

[a]State Key Laboratory of Blockchain and Security, Zhejiang University
[b]School of Software Technology, Zhejiang University
[c]Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security
[d]Big Graph Center, School of Computer and Computing Science, Hangzhou City University
[e]College of Computer Science and Technology, Zhejiang University Hangzhou, China

Abstract. Mini-batch Graph Transformer (MGT), as an emerging graph learning model, has demonstrated significant advantages in semi-supervised node prediction tasks with improved computational efficiency and enhanced model robustness. However, existing methods for processing local information either rely on sampling or simple aggregation, which respectively result in the loss and squashing of critical neighbor information. Moreover, the limited number of nodes in each mini-batch restricts the model's capacity to capture the global characteristic of the graph. In this paper, we propose **LGMformer**, a novel MGT model that employs a two-stage augmented interaction strategy, transitioning from local to global perspectives, to address the aforementioned bottlenecks. The *local interaction augmentation* (LIA) presents a neighbor-target interaction Transformer (NTIformer) to acquire an insightful understanding of the co-interaction patterns between neighbors and the target node, resulting in a locally effective token list that serves as input for the MGT. In contrast, *global interaction augmentation* (GIA) adopts a cross-attention mechanism to incorporate entire graph prototypes into the target node representation, thereby compensating for the global graph information to ensure a more comprehensive perception. To this end, LGMformer achieves the enhancement of node representations under the MGT paradigm. Experimental results related to node classification on the ten benchmark datasets demonstrate the effectiveness of the proposed method. Our code is available at https://github.com/l-wd/LGMformer.

1 Introduction

Graph Transformer (GT) models [29, 30, 36] apply the Transformer [31] architecture into graph-structured data, leveraging the well-established self-attention mechanism to capture the relationship among all graph nodes. This strategic adaptation not only preserves the advantages of the Transformer model but also mine the inherent graph structure information [6, 5, 13], allowing it to achieve the robust representation learning capabilities and satisfactory prediction performance in various domains, such as academic network analysis [11], bioinformatics [25], and traffic flow forecasting [16]. Therefore, GT models represent a significant advancement in the field of graph representation learning, providing a powerful and adaptable framework for processing and understanding complex graph-structured data.

By taking the scale of the graph data and limitations of computing resources into consideration, GT models can be divided into two categories based on training strategies: Full-batch GT (FGT) [8, 20, 33, 36] and Mini-batch GT (MGT) [3, 10, 19, 39, 33]. FGT models employ a straightforward application strategy by treating each node as a token and calculating the global attention between these tokens, allowing for the capture of underlying dependencies among distant nodes. However, the design of the global attention inherently leads to a quadratic rise in computational complexity as the number of graph nodes increases, accompanied with a substantial demand for computing resources. Although recent researches [15, 33, 35] effort to reduce algorithmic complexity to linearity using techniques such as the kernelized softmax operator, these methods remain inadequate for effectively training models with extremely large-scale graph data, mainly due to limited computing resources. To address this challenge, MGT models are developed to optimize computational resource management by strategically dividing the data into smaller, more manageable subsets, and processing them sequentially at each training step. This approach significantly enhances resource utilization efficiency, especially when dealing with large-scale graph data, which avoids the limitations of traditional full-batch processing methods constrained by computational resources.

Based on the utilization of neighbor information, the existing MGT methods can be classified into three strategies: neighbor sampling [19, 39], subgraph partition [33, 34, 35], and node tokenization [3, 10]. Neighbor sampling and subgraph partition methods, which involve sampling the k neighbors of target nodes and dividing the graph into separate subgraphs, suffer from the drawback of information loss. Specifically, neighbor sampling leads to the loss of crucial node information, while subgraph partition negatively impacts the original graph's structure. Furthermore, node tokenization involves representing each node as a sequence composed of multiple tokens, with each token capturing the neighborhood information of the node at various hop distances. However, the neighborhood features are merged into a single vector through a simple aggregation like an average or a summation operation, leading to the squashing of the critical neighborhood. To this end, we can infer that such MGT models will lead to two following issues: (1) The loss and squash-

[†]Equal Contribution. Email: {lwdup, chenkx, liushunyu}@zju.edu.cn.
[*]Corresponding Author. Email: brooksong@zju.edu.cn.

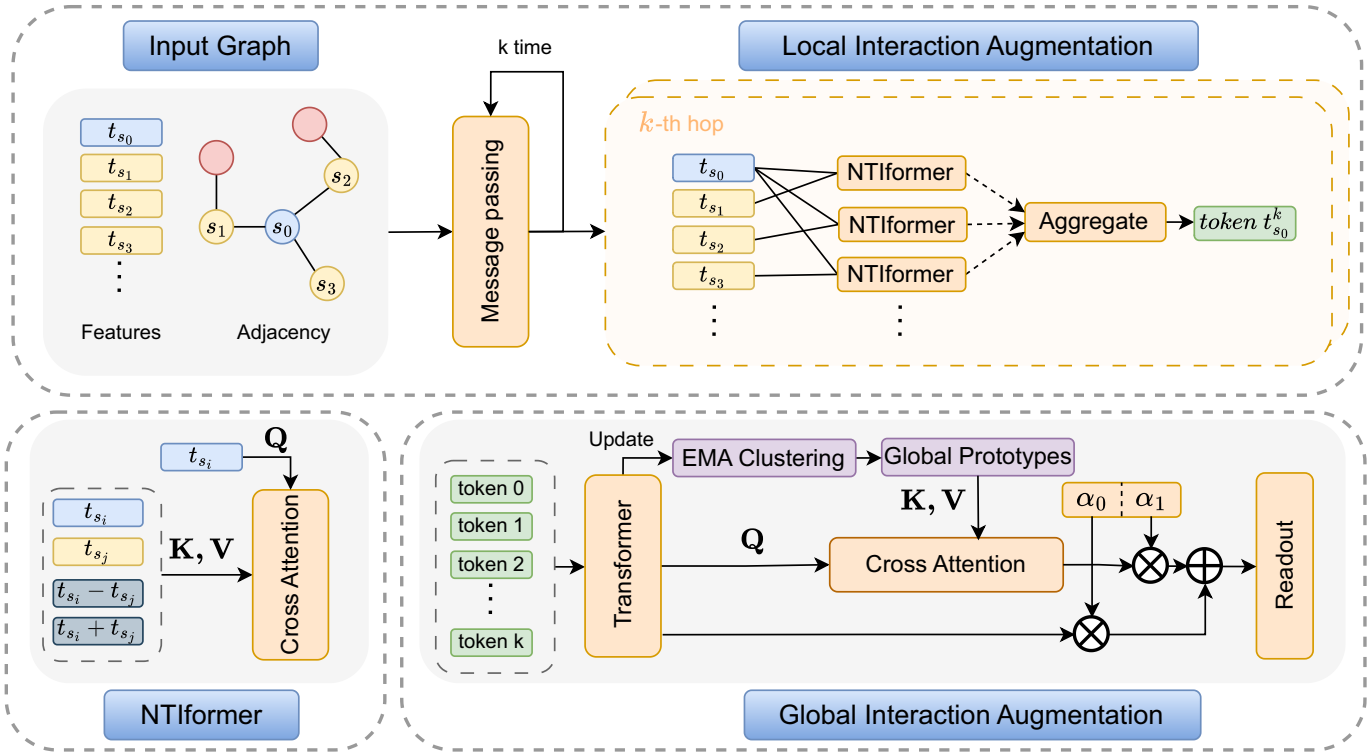

Figure 1. Illustration of the proposed LGMformer. First, LGMformer computes the multi-hop aggregation features for each node in a given graph. Afterwards, during the local interaction augmentation stage, the target node interacts with each neighbor using the NTIformer to obtain high-level semantic information. Finally, in the global interaction augmentation stage, the augmented global information is obtained by interacting with the global prototypes.

ing of critical neighborhood information has a negative impact on graph representation learning. (2) The limited number of nodes in each mini-batch restricts the model's capacity to capture the global characteristic of the graph.

To mitigate the negative impact of the MGT models, we suggest to enhance the integration of critical neighbors' information while compensating for the distant node information from a global perspective. The central concept is to alleviate issue of critical node information being squashed without losing the neighborhood information, while expanding the receptive field to a global scale, allowing the model to capture information related to the current node from more distant perspectives. Thus, the challenge is:

Challenge

How to design an effective MGT framework that enhances the integration of critical neighbors' information and expands the receptive field to a global scale?

In this paper, we adopt the node tokenization MGT as the basic framework to avoid the issue of losing neighbors' information, and then develop a **L**ocal to **G**lobal interaction augmented **M**ini-batch graph transformer (**LGMformer**) to address another two main issues in the MGT paradigm, i.e., the squashing of critical neighbors' information and the absence of a global perspective. Specifically, as shown in Figure 1, we first develop a NTIformer to directly learns the direct interaction patterns between neighboring nodes and the target node. This process is designed to capture the crucial neighboring information associated with the target node into the node token list, thereby enhancing the augmentation of local interaction. Then, we employ EMA Clustering to obtain global prototypes capable of representing the entire graph, and utilize cross-attention mechanism achieve effective interaction with the global context. This method aims to compensate for the critical information from distant nodes, thereby achieving the augmentation of global interaction. To this end, we have successfully addressed two inherent drawbacks of the MGT via both local and global interaction augmentation.

The contributions of this paper are summarized below:

- We introduce a two-stage interaction augmentation strategy that progressively enhances the capture of critical nodes from a local to global scale, effectively addressing two predominant issues that exist in the MGT framework.
- We design a tailored NTIformer to augment local interaction within the MGT framework, and further achieve global interaction augmentation through cross-attention mechanism between the target nodes and the entire graph prototypes.
- The proposed LGMformer framework demonstrates its superior performance through extensive experiments conducted on ten node classification datasets, including both homogeneous and heterogeneous graph of varying sizes.

2 Preliminaries

Notation. Let $G = (V, E)$ represent an undirected and attributed graph, where V is a set of n nodes and E is a set of m edges. Usually, each node in graph G has a d-dimensional node vector x_i representing the node's features. Thus graph G has a node feature matrix $\mathbf{X} = [x_1, x_2, \ldots, x_n]^\top \in \mathbb{R}^{n \times d}$. Edges E describe the relationships

between nodes, and all edges form an adjacency matrix $\mathbf{A}$, where $\mathbf{A}_{ij}$ represents the relationship between nodes v_i and v_j. Let $\mathbf{D}$ be the diagonal degree matrix where $\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}$. We recall that the normalized adjacency matrix $\hat{\mathbf{A}}$ is defined as $\hat{\mathbf{A}} = \tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}$ where $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{D}}$ denote the adjacency matrix and degree matrix with self-loops, respectively.

Message Passing Scheme. The majority of Graph Neural Networks (GNNs) [41, 4] architecture act by propagating information between adjacent nodes of the graph. Each message passing GNN [38, 42] aggregates the features of all the nodes around the node and passes them into the next GNN layer or downstream machine learning task [21, 23, 24] as the new features of the node. Thus, the Message Passing Scheme at the ℓ layer in GNNs can be represented as:

$$b_i^{(\ell)} = \mathrm{Agg}\Big(\Big\{h_j^{(\ell-1)} \mid j \in \mathcal{N}(i)\Big\}\Big), \tag{1}$$

$$h_i^{(\ell)} = \mathrm{Combine}(h_i^{(\ell-1)}, b_i^{(\ell)}), \tag{2}$$

where $\mathcal{N}(i)$ denotes the adjacent node set of i. $\mathrm{Agg}(\cdot)$ denotes the aggregation operation. $\mathrm{Combine}(\cdot,\cdot)$ denotes the combination operation. $h_i^{(\ell)}$ is the node features of i at the ℓ layer with the initialization of $h_i^{(0)} = x_i$ in GNNs.

Transformer. The critical component of Transformer is the self-attention module which computes the correlation between input tokens and extracts important information based on the correlation. Given a matrix $\mathbf{H} \in \mathbb{R}^{n\times d}$, where n and d represent the number of input tokens and the token feature's dimension, respectively. A standard self-attention mechanism is formalized as follows:

$$\begin{aligned}\mathrm{SelfAttn}(\mathbf{H}) &= \mathrm{softmax}\left(\mathrm{M_s}\right)\mathbf{H}\mathbf{W}_V,\\ \mathrm{M_s} &= \frac{\mathbf{H}\mathbf{W}_Q\left(\mathbf{H}\mathbf{W}\right)^\top}{\sqrt{d_K}},\end{aligned} \tag{3}$$

where $\mathbf{W}_Q \in \mathbb{R}^{d\times d_K}$, $\mathbf{W}_K \in \mathbb{R}^{d\times d_K}$, and $\mathbf{W}_V \in \mathbb{R}^{d\times d_V}$ are the projection matrices.

Considering the case of using a sequence $\mathbf{H_1} \in \mathbb{R}^{n_1\times d_1}$ to query another sequences $\mathbf{H_2} \in \mathbb{R}^{n_2\times d_2}$, where n_1 and n_2 are sequence length of $\mathbf{H_1}$ and $\mathbf{H_2}$ respectively, d_1 and d_2 are hidden dimension of $\mathbf{H_1}$ and $\mathbf{H_2}$ respectively. The standard cross attention is presented as follows:

$$\begin{aligned}\mathrm{CrossAttn}(\mathbf{H_1},\mathbf{H_2}) &= \mathrm{softmax}\left(\mathrm{M_c}\right)\mathbf{H_2}\mathbf{W}'_V,\\ \mathrm{M_c} &= \frac{\mathbf{H_1}\mathbf{W}'_Q\left(\mathbf{H_2}\mathbf{W}'_K\right)^\top}{\sqrt{d'_K}},\end{aligned} \tag{4}$$

where $\mathbf{W}'_Q \in \mathbb{R}^{d_1\times d'_K}$, $\mathbf{W}'_K \in \mathbb{R}^{d_2\times d'_K}$, and $\mathbf{W}'_V \in \mathbb{R}^{d_2\times d'_V}$ are the projection matrices.

In the graph transformer, the feature matrix is generally used as an input to the attention mechanism, i.e., $\mathbf{H} = \mathbf{X}$, and then the information interaction between nodes is achieved by following the standard self-attention mechanism.

3 LGMformer

In this section, we present the proposed LGMformer, which mainly consists of Local Interaction Augmentation (LIA) and Global Interaction Augmentation (GIA). We first describe the definition of the NTIformer and use it incorporating with the node tokens to enhance the local interaction augmentation. Then, we introduce the global prototypes learning implementation based on clustering to achieve global interaction augmentation. Finally, we give a concrete implementation of the LGMformer and present some details.

3.1 Local Interaction Augmentation

In general, detailed local structure and local features are essential, thus any absence or addition of edges or nodes result in a dramatic change of effect. Currently localized operations such as subgraph partition and neighbor sampling, suffer performance limitations due to information loss caused by inevitable node omission. To avoid information loss, node tokenization operation aggregate information from different hop neighbors as tokens, but leading to the over-squashing of local information. To adequately exploit the local information, we take a local interaction perspective and analyzing the information between connected nodes.

Observation 1. *Common information exists between the connected nodes of an arbitrary edge. Node labels are completely determined by the information contained in node attributes for node classification tasks, the common information that can significantly enhance the distinctive attributes of target nodes and help in their classification is useful.*

Obviously, the useful common information for node classification should be picked and aggregated as much as possible from the perspective of information interaction, whether the connected nodes of an edge exhibit the same class.

Motivated by the above analysis, we design the neighbor-target interaction transformer (NTIformer) to extract the useful common information between connected nodes and augment the target representation. Assuming that the input embedding of local interaction augmented module is $\mathbf{L} = [\mathbf{T}^0, \mathbf{T}^1, \cdots, \mathbf{T}^K]^\top$, where $\mathbf{T}^k \in \mathbb{R}^{n\times d_m}$, $\mathbf{L} \in \mathbb{R}^{(K+1)\times n\times d_m}$, K is the number of hops, n is the number of nodes, and d_m is the hidden dimension. Given an arbitrary edge, the connected nodes are s_i, s_j. Consider that the useful parts of the common information for nodes s_i and s_j may exhibit greater disparities due to the class. We implement synthesis and differentiation effects among neighboring nodes using two distinct operations with regarding s_i as the target node. The synthesis effect, denoted by $t_{s_i} - t_{s_j}$, contains the information that exclude common information and retain information unique to the target node s_i which may be necessary information for node classification of the target node s_i. The differentiation effect, denoted by $t_{s_i} + t_{s_j}$, contains the information that have enhanced the common information and retain necessary information for node classification of the target node s_i. Next, we regard them as the neighbor-target interaction tokens which are the inputs to the attention mechanism in NTIformer. In the input embedding $\mathbf{L}$, different hop node embeddings are associated with higher-order information due to multi-hop aggregation. The neighbor-target interaction tokens of the k hop for target node s_i is calculated as:

$$E^k_{s_i,s_j} = [t^k_{s_i}, t^k_{s_j}, t^k_{s_i} - t^k_{s_j}, t^k_{s_i} + t^k_{s_j}]^\top, \tag{5}$$

where $t^k_{s_i}$ and $t^k_{s_j}$ are embeddings of nodes s_i and s_j of k hop respectively.

In contrast to the standard transformer, NTIformer avoids the use of a quadratic complexity self-attention mechanism for computing pairwise similarities between tokens. Our motivation lies in enhancing the information pertaining to the target nodes. To achieve this, we consider the target node as a query and employ cross attention to determine the attention between the target node and other tokens as:

$$\mathrm{NTIformer}(t^k_{s_i}, t^k_{s_j}) = \mathrm{CrossAttn}(t^k_{s_i}, E^k_{s_i,s_j}). \tag{6}$$

For the target node s_i, the common information interaction of a single edge is not enough. Thus, to avoid missing any neighboring

Algorithm 1 EMA Clustering

Input: Embedding of batch nodes: $\mathbf{G}$.The batch norm module: $\mathrm{bn}()$. Function to find the nearest center for each feature: $\mathrm{FindNearest}(,)$. γ is a hyperparameter.

function UPDATE($\mathbf{G}$)
  $\mathbf{G} = \mathrm{bn}(\mathbf{G})$
  $D = \mathrm{FindNearest}(\mathbf{G}, \mathbf{P})$
  $c = c \cdot \gamma + D^T \mathbf{1} \cdot (1 - \gamma)$
  $v = v \cdot \gamma + D^T \mathbf{G} \cdot (1 - \gamma)$
  $v = v/c$
  $\mathbf{P} = v \cdot bn.running\_std + bn.running\_mean$
end function

information and acquire more abundant and complete common information, it is necessary to interact with each of its neighboring nodes and then aggregate the interacted information. Therefore, the node representation of k hop after common information interaction can be calculated as:

$$\mathrm{AN}(t_{s_i}^k) = \mathrm{Agg}\Big(\Big\{\mathrm{NTIformer}(t_{s_i}^k, t_{s_j}^k) \mid s_j \in \mathcal{N}(s_i)\Big\}\Big). \quad (7)$$

Moreover, the high-level semantic information existing in the graph is also essential for target nodes, which originates from local complex structural interactions and nodes features indirectly connected to the target node under the guidance of the graph structure, and important information for target node classification can be captured in the interaction with them. A straightforward approach is to stack multiple layers of the general-purpose information interactions of neighboring nodes. However, stacking directly cannot interact well with multi-hop node information which is restricted from propagation by neighboring nodes [2], on the one hand, and lacks scalability due to the full-batch processing, on the other hand. Therefore, we can interact and aggregate with the neighbors by using different hop tokens in $\mathbf{L}$, which in turn can avoid the above problem. For the target node s_i, we make the node features obtained at each hop interact with its direct neighboring nodes respectively. Thus, the local interaction augmented token list of the target node s_i can be obtained following:

$$\mathrm{LIA}(\mathbf{L}_{s_i}) = \Big[\mathrm{AN}(t_{s_i}^0), \mathrm{AN}(t_{s_i}^1), \cdots, \mathrm{AN}(t_{s_i}^k)\Big]^\top. \quad (8)$$

The node representations obtained in this way not only take into account feature information from nodes of different hop, but also interact with the structure of neighboring nodes and allow us to learn richer node representations in a mini-batch manner.

3.2 Global Interaction Augmentation

Global information converge the model's global comprehension over the whole graph, which can provide rich interaction information for specific nodes. Especially in mini-batch training, since the consideration of information is mostly confined around the target node, local bias can easily be introduced due to local monotonous information, and global features can compensate for the lack of global information on the target node. Based on the above, we design global interaction with the clustering so as to learn richer and more comprehensive global features. The clustering method is an operation based on all nodes so that it provides global feature information

LGMformer uses the common K-Means method as a global clustering method and regards the cluster centers as queryable global prototypes [12]. Due to the limitation of mini-batch training, providing all nodes for K-Means to learn at once is not possible in training, so we dynamically update the cluster centers of K-Means via the exponential moving average (EMA) algorithm [19], which is summarized in Algorithm 1. Notably, we remove the additional structural encoding portion as the information provided to the global entity undergoes a more intricate structural interaction based on the local to global paradigm.

Assuming that the input embedding of global interaction augmented module is $\mathbf{G}$, and $\mathbf{P} \in \mathbb{R}^{n_c \times d_c}$ is the learned graph prototypes via K-Means, where n_c is the number of prototypes and d_c is the hidden dimension. We update $\mathbf{P}$ each epoch as:

$$\mathbf{P} = \mathrm{UPDATE}(\mathbf{G}), \quad (9)$$

Global interaction allows each specific node to directly access these prototypes to complement global insights, implemented as follows:

$$\mathrm{GIA}(g_{s_i}) = \mathrm{CrossAttn}(g_{s_i}, \mathbf{P}), \quad (10)$$

where g_{s_i} is the embedding of the given node u.

3.3 Implementation Details

Given graph features $\mathbf{X}$ and normalized adjacency matrix $\hat{\mathbf{A}}$.We use the token list formed by the node features after multi-hop aggregation before training: $\mathbf{U} = [\hat{\mathbf{A}}^0\mathbf{X}, \hat{\mathbf{A}}^1\mathbf{X}, \cdots, \hat{\mathbf{A}}^K\mathbf{X}]^\top$, where K is the number of hops. Thus, for the target node s_i, the input token list is denoted as $\mathbf{U}_{\mathbf{s_i}} = [u_{s_i}^0, u_{s_i}^1, \cdots, u_{s_i}^k]^\top$, where $u_{s_i}^0$ is the original node feature without aggregation. Then we map $\mathbf{U}_{\mathbf{s_i}}$ to node embeddings token list in the latent space with a neural layer, i.e., $\mathbf{L}_{\mathbf{s_i}} = \boldsymbol{f}_\theta(\mathbf{U}_{\mathbf{s_i}})$, where $\boldsymbol{f}_\theta(\cdot)$ can be a shallow (e.g., one-layer) MLP.

Next, the local interaction augmentation module is applied on $\mathbf{L}_{\mathbf{s_i}}$ to get the local augmented embedding token list: $\mathbf{Z}_{\mathbf{s_i}}^{(\mathbf{0})} = \mathrm{LIA}(\mathbf{L}_{\mathbf{s_i}}) + \mathbf{L}_{\mathbf{s_i}}$. Note that in the local interaction augmentation module, we use independent attention training parameters for aggregated features with different hops. The token list is then put into the transformer encoder :

$$\begin{aligned} \mathbf{Z}_{local}^{\prime(\ell)} &= \mathrm{MHA}\left(\mathrm{LN}\left(\mathbf{Z}_{s_i}^{(\ell-1)}\right)\right) + \mathbf{Z}_{s_i}^{(\ell-1)}, \\ \mathbf{Z}_{local}^{(\ell)} &= \mathrm{FFN}\left(\mathrm{LN}\left(\mathbf{Z}_{local}^{\prime(\ell)}\right)\right) + \mathbf{Z}_{local}^{\prime(\ell)}, \end{aligned} \quad (11)$$

where $\ell = 1, \ldots, L$ implies the ℓ-th layer of the transformer encoder. $\mathrm{LN}(\cdot)$ denotes layer normalization. $\mathrm{FFN}(\cdot)$ refers to the feed-forward neural Network.

Meanwhile, global interaction augmentation allow the token list to give attention to the global prototypes obtained through clustering as:

$$\mathbf{Z}_{global}^{(\ell)} = \mathrm{GIA}(\mathbf{Z}_{s_i}^{(\ell-1)}) + \mathbf{Z}_{s_i}^{(\ell-1)}. \quad (12)$$

$$\mathbf{Z}_{s_i}^{(\ell)} = \mathbf{Z}_{local}^{(\ell)} + \mathbf{Z}_{global}^{(\ell)}. \quad (13)$$

Finally, to obtain the final representation vector of node u, readout functions (e.g. mean, sum, attention) will be applied on $\mathbf{Z}_{s_i}^{\prime(\ell)}$. Figure 1 depicts the architecture of LGMformer.

Complexity Analysis. The overall time complexity of LGMformer is $O(n(K+1)^2 d + (K+1)Ed + n(K+1)n_c d)$, where K is the number of hops, n is the number of nodes, E is the number of edges in graph, d is the hidden dimension, and n_c is the number of cluster centers.

4 Experiments

4.1 Experimental Setup

Datasets. We have conducted experiments on twelve benchmark datasets, including nine small-scale datasets and three relatively large-scale datasets, which contain homophilic and heterophilic datasets respectively. In the small-scale datasets, computer, photo, and wikics are homophilic graphs, and roman-empire, minesweeper, tolokers, and questions are heterophilic [22] graphs. For computer and photo, we split the dataset into train, valid, and test sets as 60%:20%:20%. As for the other small-scale datasets, we use the official splits provided in their respective papers [27, 28]. In large-scale datasets, ogbn-arxiv is homophilic dataset from OGB [14]. Pokec and twitch-gamer [22] are heterophilic datasets. The split of the large dataset follows the settings of [14, 22]. We have adopted the homophily score calculation method from [22].

Baselines. We compared our method with 12 advanced baselines, These include four full-batch GNN methods: GCN [18], GAT [32], GPRGNN [7], and four scalable GNN methods: GraphSAINT [37], PPRGo [1], and GRAND+ [9]. For graph transformer, we compare GraphGPS [29], DIFFormer [34], NodeFormer [33], GOAT [19], and NAGphormer [3] Among them, GOAT is the method using neighbor sampling to extend large graphs, NodeFormer and DIFFormer are the method using subgraph partition to extend large graphs, and NAGphormer is the tokenized graph transformers.

4.2 Performance on Small-Scale Datasets

The results presented in Table 1 demonstrate the outstanding performance of our proposed method, LGMformer, across multiple datasets. One of the key advantages of LGMformer over GNN-based models is its ability to capture higher-order semantic information more effectively. This is achieved by allowing different-hop information to flow freely among neighboring nodes, as opposed to constrained message flow based on the graph structure, as observed in models such as GCN and GAT. Compared to sampling-based GNN methods like GraphSAINT and GRAND+ and graph transformer methods like GOAT, the comprehensive inclusion of neighboring nodes enables LGMformer to achieve superior performance by incorporating more local information. Moreover, in comparison to NAGphormer, LGMformer demonstrates significant performance gains, confirming the efficacy of local and global interaction augmentation. Additionally, unlike other approaches that struggle to balance performance on heterophilic graphs, LGMformer facilitates information aggregation among heterophilic nodes through local information interaction augmentation. As a result, it attains competitive performance on heterophilic datasets, particularly the roman-empire dataset, demonstrating an accuracy improvement of 1.71%. Furthermore, while maintaining balanced performance on heterophilic graphs, LGMformer also achieves competitive performance on homophilic datasets.

4.3 Performance on Large-Scale Datasets

To demonstrate the scalability of our proposed method, LGMformer, we conducted additional experiments on three large-scale graph datasets. In terms of the baseline comparison, we selected three scalable GNN models and four graph transformer methods that offer acceptable computational costs. The results presented in Table 3 indicate that LGMformer performs exceptionally well on large-scale graphs, particularly on Pokec, with an accuracy gain of up to 4.73%. Comparing LGMformer to NAGphormer, we observe significant performance improvements, further validating the effectiveness of our approach. These results highlight the ability of LGMformer to effectively preserve both local and global information through interaction augmentation, thereby addressing the challenges of node classification in large-scale graphs.

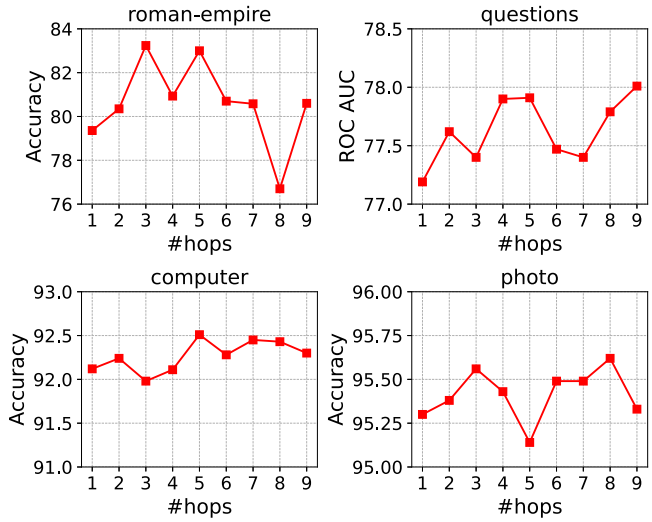

Figure 2. The performance with different length of token list.

4.4 Ablation Study

The length of input token list. The length of the input token list determines the number of neighbor hops that the target node can directly access, constituting the local receptive field of the target node. By varying the length of the input token list, we explore the impact of the size of the local receptive field on the node classification performance for the target node. The findings are presented in Figure 2. We observe that the length of the token list affects performance metrics differently across various datasets, depending on the neighborhood structure of each dataset. While a longer token list can offer more information, it does not necessarily result in improved performance. On the one hand, aggregating more distant information may introduce additional noise. On the other hand, due to the iterative aggregation of tokens from the neighborhood based on the structure, distant tokens from neighboring nodes exhibit higher similarity and thus learn similar representations. Consequently, instead of blindly increasing the length of the input token list, it is valuable to direct attention towards localization and augment the target node information based on local feature and structure interactions.

NTIformer tokens. We investigate the impact of specific tokens in NTIformer on local interaction augmentation by activating only the designated tokens. The results, as depicted in Figure 3, demonstrate that including these manually constructed tokens leads to performance improvements for graphs such as roman-empire and questions. This suggests that the attention mechanism, which is constructed solely from the nodes' own features, fails to adequately extract the significant common information shared by the nodes on both sides of an edge. On the other hand, the locally constructed tokens foster complex local interactions that efficiently extract crucial information. Thus, our proposed NTIformer module effectively enhances the local interaction module's ability to extract important informa-

Table 1. Average results of small-scale homophilic and heterophilic datasets. ± corresponds to one standard deviation of the average evaluation over 10 trials. Accuracy is reported for roman-empire, computer, photo, and wikics. ROC AUC is reported for minesweeper, tolokers, and questions. Bold and underline indicate the best and second best results, respectively. "AR" means average rank of results on small-scale datasets.

Method		Heterophilic				Homophilic			AR
		roman-empire	minesweeper	tolokers	questions	computer	photo	wikics	
GNNs	GCN	73.69 ± 0.74	89.75 ± 0.52	83.64 ± 0.67	76.09 ± 1.27	89.65 ± 0.52	92.70 ± 0.20	77.47 ± 0.85	5.7
	GAT	53.45 ± 0.27	72.23 ± 0.56	77.22 ± 0.73	76.28 ± 0.64	90.78 ± 0.13	93.87 ± 0.11	76.91 ± 0.82	7.5
	GPRGNN	64.85 ± 0.27	86.24 ± 0.61	72.94 ± 0.97	55.48 ± 0.91	89.32 ± 0.29	94.49 ± 0.14	78.12 ± 0.23	8.8
Scalable GNNs	GraphSAINT	68.57 ± 0.42	88.47 ± 1.34	82.08 ± 0.93	68.40 ± 4.62	90.22 ± 0.15	91.72 ± 0.13	65.17 ± 2.13	7.8
	PPRGo	70.78 ± 0.51	64.88 ± 1.50	70.14 ± 0.88	57.75 ± 0.81	88.69 ± 0.21	93.61 ± 0.12	76.92 ± 0.72	10.2
	GRAND+	16.50 ± 0.34	68.11 ± 1.10	71.56 ± 0.65	71.29 ± 1.30	88.74 ± 0.11	94.75 ± 0.12	78.10 ± 0.60	9.5
Scalable Graph Transformers	GraphGPS	82.00 ± 0.61	90.63 ± 0.67	83.71 ± 0.48	71.73 ± 1.47	91.19 ± 0.54	95.06 ± 0.13	**78.66 ± 0.49**	3.7
	DIFFormer	79.10 ± 0.32	**90.89 ± 0.58**	83.57 ± 0.68	72.15 ± 1.31	91.99 ± 0.76	95.10 ± 0.47	73.46 ± 0.56	3.2
	NodeFormer	64.49 ± 0.73	86.71 ± 0.88	78.10 ± 1.03	74.27 ± 1.46	86.98 ± 0.62	93.46 ± 0.35	74.73 ± 0.94	8.3
	GOAT	71.59 ± 1.25	81.09 ± 1.02	83.11 ± 1.04	75.76 ± 1.66	90.96 ± 0.90	92.96 ± 1.48	77.00 ± 0.77	6.5
	NAGphormer	74.34 ± 0.77	84.19 ± 0.66	78.32 ± 0.95	68.17 ± 1.53	91.22 ± 0.14	95.49 ± 0.11	77.16 ± 0.72	5.7
	LGMformer	**83.71 ± 0.64**	90.87 ± 0.44	**84.07 ± 1.03**	**77.75 ± 1.26**	**92.08 ± 0.25**	**95.59 ± 0.30**	78.28 ± 0.69	**1.2**

Table 2. Graph Datasets and Statistics.

Dataset	Homophily Score	#Nodes	#Edges
Computer	0.700	13,752	245,861
Photo	0.772	7,650	119,081
WikiCS	0.568	11,701	216,123
roman-empire	0.023	22,662	32,927
minesweeper	0.009	10,000	39,402
tolokers	0.187	11,758	519,000
questions	0.072	48,921	153,540
ogbn-arxiv	0.416	169,343	1,166,243
twitch-gamers	0.090	168,114	6,797,557
pokec	0.000	1,632,803	30,622,564

Table 3. Average accuracy of large-scale homophilic and heterophilic datasets. The missing results means the training cannot be finished within an acceptable time budget.

Method	ogbn-arxiv	pokec	twitch-gamer	AR
GraphSAINT	66.95 ± 0.18	68.99 ± 0.27	61.77 ± 0.27	5.3
PPRGo	59.54 ± 0.02	60.84 ± 0.02	59.83 ± 0.02	7.0
GRAND+	36.43 ± 0.00	50.76 ± 0.00	-	8.0
DIFFormer	69.86 ± 0.25	73.89 ± 0.35	61.22 ± 0.12	4.3
NodeFormer	67.19 ± 0.83	71.00 ± 1.30	62.14 ± 0.17	4.3
GOAT	**72.41 ± 0.40**	66.37 ± 0.94	62.27 ± 0.58	3.3
NAGphormer	70.13 ± 0.55	76.59 ± 0.25	64.38 ± 0.04	2.3
LGMformer	71.30 ± 0.16	**81.32 ± 0.45**	**64.70 ± 0.11**	**1.3**

tion more efficiently. Notably, the interaction of tokens s_i and s_j in the roman-empire graph results in a remarkable accuracy gain of up to 1.75%.

With and without global interaction augmentation. We analyze the impact of global interaction augmentation on the performance of LGMformer by comparing the model's performance with and without this augmentation. The results are presented in Figure 4. It can be observed that the global interaction augmentation module yields a substantial performance improvement for the datasets examined, implying that global information can offer valuable insights to specific nodes, thereby enhancing their performance.

Table 4. The performance of different number of global prototypes. "range" means the difference between the maximum and minimum results.

Sizes	roman-empire	questions	computer	photo
512	83.09 ± 0.41	77.74 ± 1.36	92.38 ± 0.25	95.33 ± 0.33
1024	83.36 ± 0.36	77.71 ± 1.09	92.44 ± 0.43	95.36 ± 0.36
2048	83.06 ± 0.62	77.95 ± 1.25	92.30 ± 0.44	95.36 ± 0.43
4096	83.71 ± 0.64	77.75 ± 1.26	92.08 ± 0.25	95.59 ± 0.30
range	0.65	0.24	0.36	0.26

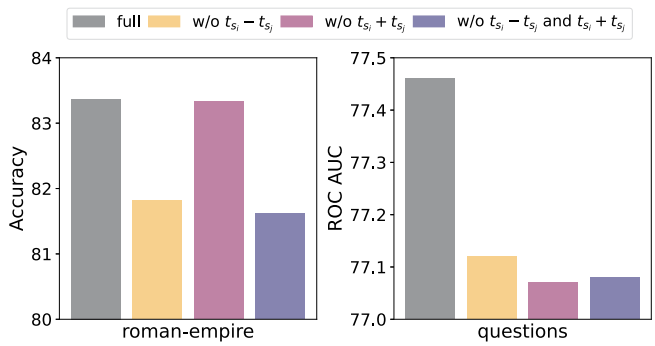

Figure 3. The performance with different token activations in local.

Global number of prototypes. We examine the impact of the number of prototypes on global performance within the global interaction augmentation module. The results are presented in Table 4. Although different datasets exhibit varying sensitivities to the number of global prototypes, overall, these variations are minimal. Intuitively, as the number of prototypes increases, the abundance of global information also increases. However, the number of global prototypes has limited influence on the enhancement of the global interaction module in our proposed model, LGMformer. This demonstrates that the purpose of the global module is to compensate for a potential dearth of global information resulting from excessive localization. As a result, a surplus of fine-grained information is not

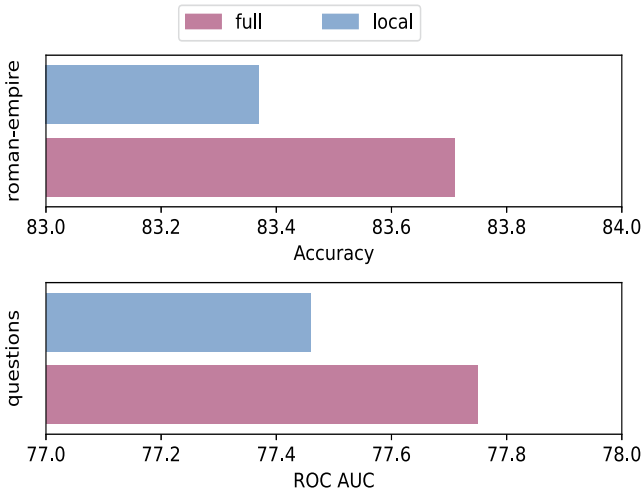

Figure 4. The performance of LGMformer with and without global interaction augmentation for graph datasets.

Table 5. The performance of with or without structural encoding graph datasets. LGMformer (R) and LGMformer (L) denote using Random values and Laplace structural encoding as additional embedding, respectively

Method	roman-empire	questions	computer	photo
LGMformer	83.71 ± 0.64	77.75 ± 1.26	92.08 ± 0.25	95.59 ± 0.30
LGMformer (R)	80.76 ± 0.72	77.53 ± 1.26	92.24 ± 0.34	95.59 ± 0.33
LGMformer (L)	79.84 ± 0.74	75.69 ± 1.21	92.30 ± 0.35	95.42 ± 0.50

essential at the global level; rather, the model simply necessitates the presence of appropriate global information.

Structural encoding. Considering that LGMformer does not currently include an encoding or modeling of the global structure, we aim to investigate the impact of implementing structural encoding in LGMformer. We utilize the eigenvectors of the Laplacian matrix of the graph adjacency matrix, a commonly used approach, as the structural encoding. These eigenvectors are then truncated following the standard practice. To embed the structural encoding, we expand the initial node features in this experiment. To mitigate potential biases introduced by dimension expansion, we employ a comparison method using random values for additional embeddings. The results are presented in Table 5. Notably, the influence of structural encoding on LGMformer varies significantly across different datasets, particularly the roman-empire and questions datasets, where incorporating global structural encoding leads to a considerable decrease in model performance. Conversely, the performance decline caused by normally distributed random values is comparatively minor. This discrepancy may arise from the additional bias carried by the structural encoding, which, when embedded into the input features, can impede or even misdirect the augmentation of information interaction in LGMformer.

5 Related Work and Discussion

Full-batch Graph transformer (FGT). FGT models treat each node as a token and calculating the global attention between these tokens, allowing for the capture of underlying dependencies among distant nodes. For example, GT [8] augments the original features of the input nodes by using Laplace feature vectors as position encoding and using them as embeddings and then calculates the global attention between nodes. NodeFormer [33] employs a kernerlized Gumbel-Softmax [17] operator to achieve efficient computation, which reduces the algorithmic complexity to linearity. SGFormer [35] designed simple global attention and demonstrates that using a one-layer attention can bring up surprisingly competitive performance. STAGNN [15] uses kernelized softmax to reduce the computational complexity of attention to linear complexity to capture global information and combines with message-passing mechanism to propose subtree attention to capture local information.

Mini-batch Graph Transformer (MGT). The quadratic complexity of the all-pair attention mechanism does not allow the transformer to scale directly to large graphs, and the attention matrix generated by a transformer containing millions of nodes consumes an unacceptably large amount of memory. Based on the utilization of neighbor information, the existing MGT methods can be classified into three strategies: neighbor sampling, subgraph partition, and node tokenization.

Neighbor sampling obtains the number of nodes within a specified range by sampling from the input perspective. GOAT [19] uses neighbor sampling to obtain neighbor nodes and calculates the attention between the central node and the neighbor nodes to learn local features, and learns global attention with the help of dimensionality reduction. ANS-GT [39] proposes adaptive sampling to integrate four different sampling methods, allowing the model to independently choose the appropriate sampling method and thus the appropriate node to learn local information during training, and introduces the corresponding coarsening graph to learn global information. Gophormer [40] samples a certain number of self-graphs for each node then uses transformers to learn the node representations on the self-graph, and uses consistent regularization and multi-sample inference strategies to attenuate the uncertainty introduced due to sampling. Subgraph partition divides the whole graph into different subgraphs and compute all-pair attention on the subgraphs. When some full-batch methods such as NodeFormer [33] and SGFormer [35] confront a larger graph, the memory they spent is still unacceptable, and they must split the whole graph into subgraphs. Node tokenization starts from the input and develops a token list based on the target node and train the transformer on the token list. NAGphformer [3] proposes to pre-extract features from different hop aggregations as token lists and train the transformer directly on it. PolyFormer [26] takes a spectral graph and polynomial perspective to build the token list with stronger representational capabilities, and use a customized polynomial attention mechanism, PolyAttn, to act as a node-based graph filter.

6 Conclusion

In this paper, we propose LGMformer, a novel scheme for Mini-batch Graph Transformer based on a two-stage interaction augmentation. In the local interaction augmentation, we first develop a NTIformer to directly learns the direct interaction patterns between neighboring nodes and the target node. Then, in the global interaction augmentation, we incorporate the information of global prototypes by interactively cross-attention between local and global representations. This design paradigm avoids the squashing of critical neighbors' information while compensating for the absence of global information. Experiments reveal LGMformer's strong performances on both homogeneous and heterogeneous graphs of varying sizes.

Acknowledgements

This work was supported in part by the Joint Funds of the Zhejiang Provincial Natural Science Foundation of China under Grant LHZSD24F020001, in part by the Ningbo Natural Science Foundation under Grant 2023J281, in part by the Zhejiang Province "LingYan" Research and Development Plan Project under Grant 2024C01114, and in part by the Zhejiang Province High-Level Talents Special Support Program "Leading Talent of Technological Innovation of Ten-Thousands Talents Program" under Grant 2022R52046.

References

[1] A. Bojchevski, J. Gasteiger, B. Perozzi, A. Kapoor, M. Blais, B. Rózemberczki, M. Lukasik, and S. Günnemann. Scaling graph neural networks with approximate pagerank. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2020.

[2] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, and X. Sun. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *Proceedings of the AAAI conference on artificial intelligence*, 2020.

[3] J. Chen, K. Gao, G. Li, and K. He. Nagphormer: A tokenized graph transformer for node classification in large graphs. In *International Conference on Learning Representations*, 2023.

[4] K. Chen, S. Liu, T. Zhu, J. Qiao, Y. Su, Y. Tian, et al. Improving expressivity of gnns with subgraph-specific factor embedded normalization. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023.

[5] K. Chen, J. Song, S. Liu, N. Yu, Z. Feng, G. Han, and M. Song. Distribution knowledge embedding for graph pooling. *IEEE Transactions on Knowledge and Data Engineering*, 35(8):7898–7908, 2023.

[6] Z. Chen, T. Xu, X.-J. Wu, R. Wang, and J. Kittler. Hybrid riemannian graph-embedding metric learning for image set classification. *IEEE transactions on big data*, 9(1):75–92, 2021.

[7] E. Chien, J. Peng, P. Li, and O. Milenkovic. Adaptive universal generalized pagerank graph neural network. *arXiv preprint arXiv:2006.07988*, 2020.

[8] V. P. Dwivedi and X. Bresson. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*, 2020.

[9] W. Feng, Y. Dong, T. Huang, Z. Yin, X. Cheng, E. Kharlamov, and J. Tang. Grand+: Scalable graph random neural networks. In *Proceedings of the ACM Web Conference 2022*, 2022.

[10] D. Fu, Z. Hua, Y. Xie, J. Fang, S. Zhang, K. Sancak, H. Wu, A. Malevich, J. He, and B. Long. Vcr-graphormer: A mini-batch graph transformer via virtual connections. In *The Twelfth International Conference on Learning Representations*, 2023.

[11] Y. Hao, X. Wang, X. Wang, X. Wang, C. Chen, and M. Song. Walking with attention: Self-guided walking for heterogeneous graph embedding. *IEEE Transactions on Knowledge and Data Engineering*, 34 (12):6047–6060, 2021.

[12] Y. Hao, M. Wang, X. Wang, T. Zheng, X. Wang, W. Huang, and C. Chen. Heterogeneous graph prototypical networks for few-shot node classification. In *International Conference on Neural Information Processing*, pages 540–555. Springer, 2023.

[13] C. Hu, J.-T. Song, J.-S. Chen, R. Wang, and X.-J. Wu. Manifold-based multi-graph embedding for semi-supervised classification. *Pattern Recognition Letters*, 182:53–59, 2024.

[14] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In *Advances in Neural Information Processing Systems*, 2020.

[15] S. Huang, Y. Song, J. Zhou, and Z. Lin. Tailoring self-attention for graph via rooted subtrees. In *Advances in Neural Information Processing Systems*, 2024.

[16] G. Huo, Y. Zhang, B. Wang, J. Gao, Y. Hu, and B. Yin. Hierarchical spatio–temporal graph convolutional networks and transformer network for traffic flow forecasting. *IEEE Transactions on Intelligent Transportation Systems*, 24(4):3855–3867, 2023.

[17] E. Jang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

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

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

[20] D. Kreuzer, D. Beaini, W. Hamilton, V. Létourneau, and P. Tossou. Rethinking graph transformers with spectral attention. In *Advances in Neural Information Processing Systems*, 2021.

[21] J. B. Lee, R. Rossi, and X. Kong. Graph classification using structural attention. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2018.

[22] D. Lim, F. Hohne, X. Li, S. L. Huang, V. Gupta, O. Bhalerao, and S. N. Lim. Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. In *Advances in Neural Information Processing Systems*, 2021.

[23] S. Liu, W. Luo, Y. Zhou, K. Chen, Q. Zhang, H. Xu, Q. Guo, and M. Song. Transmission interface power flow adjustment: A deep reinforcement learning approach based on multi-task attribution map. *IEEE Transactions on Power Systems*, 39(2):3324–3335, 2024.

[24] S. Liu, Y. Zhou, M. Song, G. Bu, J. Guo, and C. Chen. Progressive dcision-making framework for power system topology control. *Expert Systems with Applications*, 235:121070, 2024.

[25] A. Ma, X. Wang, J. Li, C. Wang, T. Xiao, Y. Liu, H. Cheng, J. Wang, Y. Li, Y. Chang, et al. Single-cell biological network inference using a heterogeneous graph transformer. *Nature Communications*, 14(1):964, 2023.

[26] J. Ma, M. He, and Z. Wei. Polyformer: Scalable graph transformer via polynomial attention. 2023.

[27] P. Mernyei and C. Cangea. Wiki-cs: A wikipedia-based benchmark for graph neural networks. *arXiv preprint arXiv:2007.02901*, 2020.

[28] O. Platonov, D. Kuznedelev, M. Diskin, A. Babenko, and L. Prokhorenkova. A critical look at the evaluation of gnns under heterophily: Are we really making progress? *arXiv preprint arXiv:2302.11640*, 2023.

[29] L. Rampášek, M. Galkin, V. P. Dwivedi, A. T. Luu, G. Wolf, and D. Beaini. Recipe for a general, powerful, scalable graph transformer. In *Advances in Neural Information Processing Systems*, 2022.

[30] H. Shirzad, A. Velingker, B. Venkatachalam, D. J. Sutherland, and A. K. Sinop. Exphormer: Sparse transformers for graphs. In *International Conference on Machine Learning*, 2023.

[31] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

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

[33] Q. Wu, W. Zhao, Z. Li, D. Wipf, and J. Yan. Nodeformer: A scalable graph structure learning transformer for node classification. In *Advances in Neural Information Processing Systems*, 2022.

[34] Q. Wu, C. Yang, W. Zhao, Y. He, D. Wipf, and J. Yan. DIFFormer: Scalable (graph) transformers induced by energy constrained diffusion. In *The Eleventh International Conference on Learning Representations*, 2023.

[35] Q. Wu, W. Zhao, C. Yang, H. Zhang, F. Nie, H. Jiang, Y. Bian, and J. Yan. Sgformer: Simplifying and empowering transformers for large-graph representations. In *Advances in Neural Information Processing Systems*, 2023.

[36] C. Ying, T. Cai, S. Luo, S. Zheng, G. Ke, D. He, Y. Shen, and T.-Y. Liu. Do transformers really perform badly for graph representation? In *Advances in Neural Information Processing Systems*, 2021.

[37] H. Zeng, H. Zhou, A. Srivastava, R. Kannan, and V. Prasanna. Graphsaint: Graph sampling based inductive learning method. *arXiv preprint arXiv:1907.04931*, 2019.

[38] L. Zhang, D. Xu, A. Arnab, and P. H. Torr. Dynamic graph message passing networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[39] Z. Zhang, Q. Liu, Q. Hu, and C.-K. Lee. Hierarchical graph transformer with adaptive node sampling. In *Advances in Neural Information Processing Systems*, 2022.

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

[41] T. Zheng, Z. Feng, T. Zhang, Y. Hao, M. Song, X. Wang, X. Wang, J. Zhao, and C. Chen. Transition propagation graph neural networks for temporal networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[42] T. Zheng, X. Wang, Z. Feng, J. Song, Y. Hao, M. Song, X. Wang, X. Wang, and C. Chen. Temporal aggregation and propagation graph neural networks for dynamic representation. *IEEE Transactions on Knowledge and Data Engineering*, 2023.