DNFormer: Differential Attention for Graph Transformers

Zizhen Wang[1] , **Dongxiao He**[1,*] , **Zhizhi Yu**[1,*] , **Kuntharrgyal Khysru**[2] , **Weixiong Zhang**[3]
[1]College of Intelligence and Computing, Tianjin University, Tianjin, China
[2]Key Laboratory of Artificial Intelligence Application Technology, Qinghai Minzu University, Xining, China
[3]Department of Health Technology and Informatics, and Department of Data Science and Artificial Intelligence, The Hong Kong Polytechnic University, Kowloon, Hong Kong
{wzz_0043, hedongxiao, yuzhizhi}@tju.edu.cn, gtj186@tju.edu.cn, weixiong.zhang@polyu.edu.hk

Abstract

Graph Transformers (GTs) have emerged as a powerful paradigm for graph representation learning, leveraging attention mechanisms to enable flexible information exchange. Recent GTs often adopt local attention to restrict computation to neighborhoods, thereby reducing costs and enhancing scalability. However, local attention computation poses a serious issue that significantly reduces the efficacy of the overall approaches – because computation is restricted to local neighborhoods, node pairs that share topological patterns receive similar attention scores, thereby reducing attention's discriminative power. Consequently, GT-based models typically overemphasize common structural motifs but fail to capture node-specific properties. To address this problem, we propose DNFormer, a novel GT grounded in differential modeling. We introduce a differential local attention mechanism in DNFormer that computes attention scores as differences across multiple topology-aware similarity measures, rather than relying on a single score. This mechanism suppresses shared topological patterns among neighbors and emphasizes relative node-to-node distinctions. As a result, DNFormer mitigates structural dominance in attention aggregation, better capturing node-specific relational patterns while preserving local structural properties. Extensive experiments confirm DNFormer's superior performance over representative GT baselines.

1 Introduction

Graph Neural Networks (GNNs), a promising paradigm for graph representation learning, have attract much attention lately, owing to their ability to effectively model both node attributes and graph topologies, and have demonstrated strong performance across a wide range of downstream tasks, such as node classification [Kipf and Welling, 2017; Velickovic *et al.*, 2018], link prediction [Zhang and Chen, 2018], and graph classification [Zhang *et al.*, 2018]. Classical GNNs, such as graph convolution networks (GCNs) [Kipf and Welling, 2017] and graph attention networks (GATs) [Velickovic *et al.*, 2018], adopt message passing mechanisms, where node representations are iteratively updated by aggregating the features of their neighbors. While effective, this paradigm imposes inherent constraints: the receptive field grows only linearly with network depth, restricting its ability to capture long-range dependencies [Xu *et al.*, 2019; Li *et al.*, 2018]. Moreover, as layers stack deeper, GNNs often suffer from over-smoothing [Oono and Suzuki, 2020] (i.e., node representations become indistinguishable) and over-squashing [Alon and Yahav, 2021] (i.e., exponentially increasing information from distant nodes is compressed into fixed-size embeddings). These limitations have sparked increased interest in graph Transformer architectures, offering a promising alternative to traditional message-passing GNNs.

Graph Transformers (GTs) adapt the Transformer [Vaswani *et al.*, 2017] architecture to graphs by using self-attention mechanism to compute pairwise node interactions and update node representations via weight aggregation [Ying *et al.*, 2021; Dwivedi and Bresson, 2020]. However, early GT variants rely primarily on feature-only attention and lack effective mechanisms to incorporate graph structures, often leading to the omission of critical topological properties. To address these drawbacks, recent studies have proposed structure-aware GTs to explicitly integrate topological information. These GT-based approaches can be grouped into two categories. The first combines GNN-type local propagation with Transformer blocks to jointly leverage both local and global representations, as exemplified by GraphGPS [Dwivedi and Bresson, 2020], CoBFormer [Xing *et al.*, 2024], and DualFormer [Zhuo *et al.*, 2025]. The second category incorporates graph topologies into attention computation via explicit structural priors, including shortest-path distance, Laplacian or positional encodings, and structure-biased attention or attention biases [You *et al.*, 2020; Kreuzer *et al.*, 2021]. Nevertheless, many of these approaches rely predominantly on fully connected self-attention, where the quadratic computation overhead of modeling global interactions among all node pairs becomes prohibitively expensive for large graphs, thus significantly limiting their scalability.

Several attempts have been made to adopt local atten-

*Corresponding author

tion mechanisms for GTs, in which each node attends only to a limited set of neighbors, typically the 1-hop neighborhood or a sampled local subgraph. For instance, Exphormer builds sparse attention patterns using expander graphs together with original graph edges to perform scalable message passing [Shirzad *et al.*, 2023]. NAGphormer constructs neighborhood-aware token sequences (e.g., ego-graphs) and applies attention within these local contexts [Chen *et al.*, 2023]. LargeGT leverages subgraph sampling to restrict attention computation to compact local substructures for large graphs [Dwivedi *et al.*, 2023]. In short, the primary objective of these methods is to reduce computational cost to nearly linear in the number of edges, while maintaining efficiency and preserving structural bias by restricting message exchange to graph's connectivities.

Despite their promising results, local attention in GTs is typically less discriminative than expected, often overemphasizing shared structural patterns while underweighting node-specific properties. Our close analysis of this phenomenon revealed that attention scores computed within a neighborhood contain a shared, non-discriminative component, which we term common-mode noise. This arises because the neighbors of a given center node frequently share a similar local topology, such as one-hop proximity and recurring motifs like hubs and triangles. When attention scores incorporate topology-aware terms, these terms can contribute similarly across many neighbors, forming a large, shared component in their scores. For example, consider a high-degree node whose neighbors play similar structural roles. If the attention score includes a shared structural bias, many neighbors receive nearly identical topology-related contributions. In such a case, feature-based differences are small relative to the shared component, leading the softmax to produce nearly uniform weights across the neighborhood and causing many neighbors to be weighted similarly. Seriously, this shared component provides little discriminative power, resulting in a low-contrast attention computation. Consequently, the model may overemphasize common structural patterns while overlooking node-specific properties.

To address these serious issues, we propose a **D**iffere**N**tial local attention graph trans**Former**, called DNFormer, which suppresses common-mode topology bias in neighborhood-restricted attention. In local attention, the logit between a node and its neighbors typically mixes (i) a topology-related bias that is similar across many neighbors (e.g., frequent structural patterns) and (ii) smaller neighbor-specific signals. When topology-related terms have large magnitudes and are highly correlated across neighbors, they make the logits within a neighborhood overly similar, leading to low-contrast softmax weights and reduced discriminability.

Motivated by common-mode rejection, DNFormer computes two topology-aware similarity measures over the same neighborhood and forms a differential logit by subtraction. This subtraction removes the correlated, topology-induced part that many neighbors share, while preserving relative differences across neighbors. Consequently, the attention weights exhibit higher contrast within each neighborhood and are less dominated by frequent structural patterns, enabling more discriminative local aggregation. Then, we further enhance this mechanism with edge-adaptive modulation, where a learnable coefficient λ_{ij} is predicted from the features of the two endpoint nodes via a lightweight feed-forward network. This coefficient controls how strongly the second score is subtracted on each edge, allowing the model to adapt the degree of common-mode suppression to different local contexts. As a result, DNFormer achieves fine-grained, edge-specific control over attention computation while retaining the scalability of local attention.

We make the following main contributions:

- We identify common-mode noise as an overlooked issue in local attention for GTs, and explain why shared local topology can dominate attention scores.
- We propose a differential local attention mechanism and implement it in DNFormer to suppress shared structural signals and improve neighbor discrimination, without sacrificing efficiency.
- Extensive experiments show that DNFormer consistently improves over representative GT baselines on standard benchmarks.

2 Preliminaries

We first introduce the notations and briefly review Graph Transformers.

2.1 Notations

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ be an attributed graph, where $\mathcal{V}$ is the node set with N nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set, and $\mathbf{X} \in \mathbb{R}^{N \times F}$ is the node feature matrix. The feature of node i is denoted as $\mathbf{x}_i \in \mathbb{R}^F$. The adjacency matrix is $\mathbf{A} \in \{0,1\}^{N \times N}$, where $\mathbf{A}_{ij} = 1$ if $(i,j) \in \mathcal{E}$ and 0 otherwise.

We denote the neighborhood of node i as $\mathcal{N}(i) = \{j \mid (i,j) \in \mathcal{E}\}$. More generally, we use $\mathcal{S}(i)$ to denote the attention candidate set of node i. For global attention, $\mathcal{S}(i) = \mathcal{V}$; for local attention, $\mathcal{S}(i)$ is a small subset such as $\mathcal{N}(i)$ or a sampled local subgraph.

For node classification, each node i has a label $y_i \in \{1, 2, \ldots, c\}$, where c is the number of classes. The goal is to learn a predictor f that maps node features and graph structure to class scores for all nodes.

2.2 Graph Transformers

Graph Transformers extend Transformer to graph data. Given node representations $\mathbf{H} \in \mathbb{R}^{N \times d}$, a single attention head projects them into query, key, and value spaces:

$$\mathbf{Q} = \mathbf{H}\mathbf{W}_Q, \quad \mathbf{K} = \mathbf{H}\mathbf{W}_K, \quad \mathbf{V} = \mathbf{H}\mathbf{W}_V, \tag{1}$$

where $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{d \times d'}$ and d' is the per-head dimension.

For node i, attention scores are computed over a set of nodes $\mathcal{S}(i)$, which may correspond to its neighbors or a predefined receptive field. The attention weight from node i to node j is given by

$$\alpha_{ij} = \mathrm{softmax}_{j \in \mathcal{S}(i)} \left(\frac{\mathbf{q}_i^\top \mathbf{k}_j}{\sqrt{d'}} + b_{ij} \right), \tag{2}$$

where $\mathbf{q}_i$ and $\mathbf{k}_j$ denote the i-th and j-th rows of $\mathbf{Q}$ and $\mathbf{K}$, respectively. The term b_{ij} is an additive bias that incorporates graph structure into attention computation.

Different Graph Transformers instantiate b_{ij} in different ways, such as using edge features, relative positional encodings, shortest-path distance biases, or other structural encodings [Ying *et al.*, 2021; Dwivedi and Bresson, 2020].

The output representation of node i is obtained as a weighted sum of value vectors:

$$\mathbf{z}_i = \sum_{j \in \mathcal{S}(i)} \alpha_{ij} \mathbf{v}_j, \tag{3}$$

where $\mathbf{v}_j$ is the j-th row of $\mathbf{V}$. Multi-head attention applies this computation in parallel across multiple heads, followed by standard feed-forward layers and residual connections.

Full attention set $\mathcal{S}(i) = \mathcal{V}$, which requires computing all node pairs and is expensive on large graphs. Local attention restricts $\mathcal{S}(i)$ to a small set, commonly $\mathcal{N}(i)$ or a sampled local subgraph, which reduces computation to be roughly linear in the number of edges [Shirzad *et al.*, 2023]. In many GT variants, local attention is used together with topology-aware terms b_{ij} so that attention remains structure-aware while being scalable. It is worth noting that our method builds on this local, topology-aware attention setting.

3 Methodology

In this section, we introduce DNFormer, a graph transformer architecture that computes local attention using differential attention and an edge-adaptive mechanism. The model computes attention by constructing two attention distributions over each neighborhood and combining them through a differential operation. A theoretical analysis of the differential attention mechanism is provided in Sec. 3.3.

3.1 Overview

As shown in Figure 1, given the input node representations $\mathbf{H}^{(l)}$ and the graph structure $\mathbf{A}$, DNFormer first applies linear projections to obtain the query, key, and value features. To enable differential local attention, the query and key channels are expanded and split into two parallel subspaces, which produce two attention distributions over the same local neighborhood, denoted as Attn_1 and Attn_2. DNFormer then combines them through a subtractive operation, where Attn_2 is scaled by an edge-adaptive coefficient λ_{ij}. The coefficient λ_{ij} is predicted for each edge from the concatenation of its endpoint node features using a lightweight feed-forward network with a positive activation, enabling per-edge (and per-head) modulation of the subtraction strength. The resulting attention weights are used to aggregate value features within the neighborhood, thereby increasing attention contrast and improving neighbor discrimination during local message aggregation. Finally, the aggregated representation is fed into LayerNorm and a feed-forward network (FFN) with residual connections to produce the output $\mathbf{H}^{(l+1)}$.

3.2 Differential Attention Mechanism

Considering that attention weights over a neighborhood may become similar across neighbors after normalization, we introduce a differential attention mechanism for local neighborhood aggregation. Specifically, it computes two distinct attention distributions over the same neighborhood and subtracts one from the other, thereby enhancing the contrast among neighbors and improving discriminative aggregation.

Let $\mathbf{X} \in \mathbb{R}^{N \times d}$ denote the input node feature matrix, we project $\mathbf{X}$ into query, key, and value projections via learnable linear transformations:

$$\mathbf{Q} = \mathbf{X}\mathbf{W}_Q, \quad \mathbf{K} = \mathbf{X}\mathbf{W}_K, \quad \mathbf{V} = \mathbf{X}\mathbf{W}_V, \tag{4}$$

where $\mathbf{W}_Q, \mathbf{W}_K \in \mathbb{R}^{d \times 2md'}$ and $\mathbf{W}_V \in \mathbb{R}^{d \times md'}$. Here m denotes the number of attention heads and $d' = d/m$ is the dimension of each head. The query and key projections are intentionally expanded to twice the standard width in order to support two parallel attention pathways.

The projected queries and keys are reshaped into $\mathbb{R}^{N \times m \times 2d'}$ and split along the feature dimension:

$$\mathbf{Q} = [\mathbf{Q}^{(1)} \,\|\, \mathbf{Q}^{(2)}], \quad \mathbf{K} = [\mathbf{K}^{(1)} \,\|\, \mathbf{K}^{(2)}], \tag{5}$$

where $\mathbf{Q}^{(1)}, \mathbf{Q}^{(2)}, \mathbf{K}^{(1)}, \mathbf{K}^{(2)} \in \mathbb{R}^{N \times m \times d'}$. The query and key projections are split into two subspaces $\mathbf{Q}^{(1)}, \mathbf{K}^{(1)}$ and $\mathbf{Q}^{(2)}, \mathbf{K}^{(2)}$, producing two separate sets of attention scores over the same neighborhood.

For each edge $(i, j) \in \mathcal{E}$ and attention head m, we compute two sets of unnormalized attention scores using scaled dot-products:

$$\begin{aligned} \alpha_{ij}^{(1,m)} &= \frac{1}{\sqrt{d'}} \left(\mathbf{q}_i^{(1,m)}\right)^\top \mathbf{k}_j^{(1,m)}, \\ \alpha_{ij}^{(2,m)} &= \frac{1}{\sqrt{d'}} \left(\mathbf{q}_i^{(2,m)}\right)^\top \mathbf{k}_j^{(2,m)}, \end{aligned} \tag{6}$$

where $\mathbf{q}_i^{(r,m)}$ and $\mathbf{k}_j^{(r,m)}$ denote the m-th head embeddings of node i and j in the r-th subspace. The scaling factor $\frac{1}{\sqrt{d'}}$ follows standard practice to stabilize optimization. Each set of scores is independently normalized over the neighborhood $\mathcal{N}(i)$:

$$\begin{aligned} a_{ij}^{(1,m)} &= \frac{\exp(\alpha_{ij}^{(1,m)})}{\sum_{k \in \mathcal{N}(i)} \exp(\alpha_{ik}^{(1,m)})}, \\ a_{ij}^{(2,m)} &= \frac{\exp(\alpha_{ij}^{(2,m)})}{\sum_{k \in \mathcal{N}(i)} \exp(\alpha_{ik}^{(2,m)})}. \end{aligned} \tag{7}$$

This normalization ensures that both attention branches operate on comparable scales prior to combination. We then define the final attention weight as the difference between two normalized attention distributions:

$$a_{ij}^{(m)} = a_{ij}^{(1,m)} - \lambda_{ij} \cdot a_{ij}^{(2,m)}, \tag{8}$$

where $\lambda_{ij} > 0$ is an edge-dependent coefficient that scales the second attention distribution. The coefficient λ_{ij} scales the second attention distribution on each edge and the computation of λ_{ij} is described in Sec. 3.4.

The output representation of node i under head m is obtained by aggregating value embeddings using the differential attention weights:

$$\mathbf{h}_i^{(m)} = \sum_{j \in \mathcal{N}(i)} a_{ij}^{(m)} \cdot \mathbf{v}_j^{(m)}, \tag{9}$$

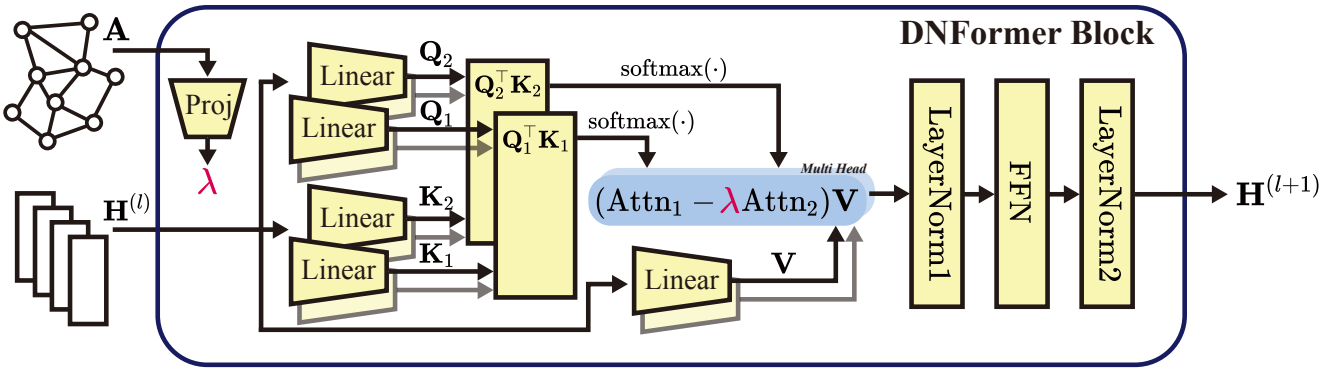


Figure 1: Overview of the differential local attention Graph Transformer (DNFormer) block.

where $\mathbf{v}_j^{(m)}$ denotes the m-th head of the value projection $\mathbf{V}$.

In summary, the proposed differential attention mechanism computes attention weights by subtracting two normalized attention distributions to reduce uniformity in attention. By subtracting two attention distributions computed over the same neighborhood, the model mitigates uniform attention patterns in local aggregation. Notably, this design preserves the standard attention aggregation while modifying how attention weights are computed.

3.3 Theoretical Analysis: Common-Mode Noise in Local Attention

In this section, we provide a theoretical justification for the proposed differential attention mechanism. We first formalize the notion of common-mode noise in local attention aggregation. We then show that standard graph attention mechanisms inherently produce such noise. Finally, we prove that the proposed differential attention formulation effectively suppresses this common-mode component.

Definition of Common-Mode Noise

We consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and focus on a fixed node $i \in \mathcal{V}$ with neighborhood $\mathcal{N}(i)$. For a given attention head m, let $a_{ij}^{(m)}$ denote the normalized attention weight assigned to neighbor $j \in \mathcal{N}(i)$.

Definition 3.1 (Common-Mode Noise)**.** For node i and attention head m, a common-mode noise component is a function:

$$c_i^{(m)} : \mathcal{N}(i) \to \mathbb{R},$$

such that $c_i^{(m)}(j) = \kappa_i^{(m)}$ for all $j \in \mathcal{N}(i)$, where $\kappa_i^{(m)}$ is a constant independent of j. An attention weight $a_{ij}^{(m)}$ is said to contain common-mode noise if it can be decomposed as:

$$a_{ij}^{(m)} = \tilde{a}_{ij}^{(m)} + c_i^{(m)}(j), \tag{10}$$

where $\tilde{a}_{ij}^{(m)}$ captures neighbor-specific relational variation.

This definition characterizes common-mode noise as a constant term that is added uniformly to the attention weights of all neighbors of a node and is independent of the specific neighbor being considered.

Existence of Common-Mode Noise in Local Attention

Here we analyze how standard attention-based graph aggregation can give rise to common-mode components.

Theorem 3.2. *Consider a single-head attention mechanism with attention weights computed as:*

$$a_{ij} = \frac{\exp(\alpha_{ij})}{\sum_{k\in\mathcal{N}(i)} \exp(\alpha_{ik})}, \quad \alpha_{ij} = \frac{1}{\sqrt{d'}} \mathbf{q}_i^\top \mathbf{k}_j.$$

Assume that the neighborhood features satisfy

$$\mathbf{k}_j = \boldsymbol{\mu}_i + \boldsymbol{\epsilon}_j, \quad \mathbb{E}_{j\in\mathcal{N}(i)}[\boldsymbol{\epsilon}_j] = \mathbf{0},$$

where $\boldsymbol{\mu}_i$ represents a shared neighborhood component. Then the resulting attention weights a_{ij} admit a decomposition that includes a non-zero common-mode noise component.

Proof. Substituting the decomposition of $\mathbf{k}_j$ into the attention logits yields:

$$\alpha_{ij} = \frac{1}{\sqrt{d'}} \mathbf{q}_i^\top \boldsymbol{\mu}_i + \frac{1}{\sqrt{d'}} \mathbf{q}_i^\top \boldsymbol{\epsilon}_j.$$

Let

$$\beta_i = \frac{1}{\sqrt{d'}} \mathbf{q}_i^\top \boldsymbol{\mu}_i, \quad \delta_{ij} = \frac{1}{\sqrt{d'}} \mathbf{q}_i^\top \boldsymbol{\epsilon}_j.$$

Then

$$a_{ij} = \frac{\exp(\beta_i + \delta_{ij})}{\sum_{k\in\mathcal{N}(i)} \exp(\beta_i + \delta_{ik})} = \frac{\exp(\delta_{ij})}{\sum_{k\in\mathcal{N}(i)} \exp(\delta_{ik})}.$$

Using a first-order Taylor expansion of $\exp(\delta_{ij})$ around zero and the assumption $\mathbb{E}[\delta_{ij}] = 0$, we obtain

$$a_{ij} = \frac{1}{|\mathcal{N}(i)|} + \mathcal{O}(\delta_{ij}).$$

The leading term $\frac{1}{|\mathcal{N}(i)|}$ is constant across all $j \in \mathcal{N}(i)$ and therefore constitutes a common-mode noise component according to Definition 1. □

This result shows that softmax normalization introduces a constant term into the attention weights that is identical across all neighbors.

Suppression of Common-Mode Noise via Differential Attention

Here we analyze how the proposed differential attention mechanism attenuates the common-mode component identified above.

Theorem 3.3. *Let $a_{ij}^{(1,m)}$ and $a_{ij}^{(2,m)}$ be the two normalized attention scores defined in Section 3.2. Assume that both scores admit decompositions of the form:*

$$a_{ij}^{(r,m)} = \tilde{a}_{ij}^{(r,m)} + c_i^{(m)}, \quad r \in \{1, 2\},$$

where $c_i^{(m)}$ is a shared neighborhood-level component. Then the differential attention:

$$a_{ij}^{(m)} = a_{ij}^{(1,m)} - \lambda_{ij} a_{ij}^{(2,m)}$$

suppresses the common-mode component whenever $\lambda_{ij} \approx 1$.

Proof. Substituting the decompositions into the differential attention yields:

$$a_{ij}^{(m)} = \tilde{a}_{ij}^{(1,m)} - \lambda_{ij} \tilde{a}_{ij}^{(2,m)} + (1 - \lambda_{ij}) c_i^{(m)}.$$

If $\lambda_{ij} = 1$, the constant term $c_i^{(m)}$ is eliminated. If λ_{ij} is close to 1, the contribution of $c_i^{(m)}$ is reduced by the factor $(1 - \lambda_{ij})$. In both cases, the attention weight depends less on the constant term shared across neighbors. □

This theorem shows that the constant term shared across neighbors is removed when the two attention distributions are subtracted.

3.4 Edge-Adaptive Modulation

To enhance the expressiveness of the differential attention mechanism, we introduce an edge-adaptive module that dynamically modulates the influence of the subtractive attention component. Instead of using a fixed scalar λ across all edges, we learn a structure-aware coefficient $\lambda_{ij}^{(m)} > 0$ for each edge $\mathbf{A}_{ij}$ and attention head $m \in \{1, \ldots, h\}$. This allows the model to selectively suppress or retain information from neighbors based on their local structure. Formally, for each edge $\mathbf{A}_{ij}$, we construct an edge-specific input by concatenating the source and target node features:

$$\mathbf{z}_{ij} = [\mathbf{x}_i \,\|\, \mathbf{x}_j] \in \mathbb{R}^{2d}, \tag{11}$$

where $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$ are the raw node features of node i and j. This concatenated representation $\mathbf{z}_{ij}$ is passed through a small feed-forward network to predict the adaptive coefficients for all heads:

$$\lambda_{ij} = \mathrm{Softplus}\left(\mathbf{W}_2 \,\mathrm{ReLU}\left(\mathbf{W}_1 \mathbf{z}_{ij} + \mathbf{b}_1\right) + \mathbf{b}_2\right), \tag{12}$$

where $\mathbf{W}_1 \in \mathbb{R}^{d' \times 2d}$, $\mathbf{W}_2 \in \mathbb{R}^{h \times d'}$. The $\mathrm{Softplus}(\cdot)$ activation ensures $\lambda_{ij}^{(m)} > 0$ for all heads m, preserving the interpretability of the subtraction in the attention formulation. The resulting vector $\boldsymbol{\lambda}_{ij} = [\lambda_{ij}^{(1)}, \ldots, \lambda_{ij}^{(h)}]^\top$ is used to scale the second attention component per edge and per head, allowing the model to adaptively determine how much negative influence to apply based on local structural and semantic context.

3.5 Complexity and Stability

DNFormer preserves the edge-linear complexity of neighborhood-restricted attention. For each layer, attention and aggregation are computed only on edges, yielding a time complexity of $O(|\mathcal{E}|d)$ (up to a constant factor for multi-head attention), which is of the same order as existing sparse/local Graph Transformers. The edge-adaptive predictor MLP_λ operates on concatenated endpoint features and outputs a small vector per edge, introducing only lightweight per-edge computation and negligible overhead in practice.

We adopt several design choices to ensure stable optimization. The edge-adaptive coefficient λ_{ij} is constrained to be positive (e.g., via a positive activation), which keeps the subtractive fusion well-behaved and avoids unstable scaling of the second attention score. In addition, dropout on attention coefficients mitigates over-confident weighting, while LayerNorm and residual connections in the Transformer block improve gradient flow and reduce training instability. Empirically, DNFormer shows convergence speed comparable to that of standard Graph Transformers and remains robust under noisy or redundant neighborhood connections.

4 Experiments

In this section, we present extensive experiments to evaluate the effectiveness of the proposed DNFormer. We first describe the experimental settings, datasets, and baseline methods, followed by the implementation details and evaluation protocol.

4.1 Experimental Setup

Datasets. We evaluate DNFormer on seven widely used benchmarks, including three citation networks [Sen *et al.*, 2008] (Cora, CiteSeer, PubMed), two co-purchase [McAuley *et al.*, 2015] (Computers, Photo) and two co-authorship graphs [Sinha *et al.*, 2015] (CS, Physics). The detailed statistics of these datasets are summarized in Table 1. The citation networks contain papers as nodes and citation links as edges, with bag-of-words features and topic labels. The co-purchase and co-authorship graphs involve product or author nodes connected by purchase or collaboration relations, characterized by richer semantics and denser connectivity.

Dataset	Nodes	Edges	Features	Classes
Cora	2,708	5,429	1,433	7
CiteSeer	3,327	4,732	3,703	6
PubMed	19,717	44,338	500	3
Computers	13,752	245,861	767	10
Photo	7,650	119,081	745	8
CS	18,333	81,894	6,805	15
Physics	34,493	247,962	8,415	5

Table 1: Statistics of the datasets used in experiments.

Baselines. The baseline methods include five representative Graph Neural Networks (GNNs): GCN [Kipf and Welling, 2017], GAT [Velickovic *et al.*, 2018], GraphSAGE [Hamilton *et al.*, 2017], APPNP [Klicpera *et al.*, 2019], and SGC

Model	Cora	CiteSeer	PubMed	Computers	Photo	CS	Physics
GCN	83.21 ± 0.27	72.64 ± 0.33	83.91 ± 0.38	88.94 ± 0.60	92.88 ± 0.17	92.97 ± 0.05	96.07 ± 0.11
GAT	85.68 ± 0.55	72.50 ± 1.37	84.17 ± 0.26	91.04 ± 0.08	93.37 ± 0.26	93.55 ± 0.11	96.11 ± 0.08
GraphSAGE	83.92 ± 0.38	72.31 ± 0.83	84.96 ± 0.47	91.17 ± 0.24	93.16 ± 0.24	92.78 ± 0.79	96.37 ± 0.03
APPNP	84.76 ± 0.48	72.47 ± 0.29	85.35 ± 0.26	89.77 ± 0.24	93.61 ± 0.32	93.97 ± 0.17	96.61 ± 0.05
SGC	82.45 ± 0.13	72.54 ± 0.21	83.63 ± 0.14	89.02 ± 0.57	92.59 ± 0.33	93.12 ± 0.30	96.17 ± 0.09
NodeFormer	87.44 ± 0.83	77.12 ± 1.37	89.23 ± 0.29	87.64 ± 0.87	93.32 ± 0.63	94.77 ± 0.22	96.33 ± 0.26
SGFormer	85.72 ± 1.08	76.50 ± 1.93	88.95 ± 1.17	89.93 ± 1.61	93.92 ± 0.85	92.42 ± 1.15	96.18 ± 0.19
DUALFormer	**87.89 ± 0.09**	74.59 ± 0.30	88.51 ± 0.05	91.36 ± 0.06	94.67 ± 0.05	94.52 ± 0.18	**97.20 ± 0.02**
DNFormer(Ours)	86.35 ± 0.40	**77.36 ± 0.73**	**89.57 ± 0.31**	**92.13 ± 0.27**	**95.14 ± 0.25**	**95.28 ± 0.09**	96.95 ± 0.05

Table 2: Accuracy of node classification (mean ± standard deviation) averaged over 10 runs. The best results are bolded.

[Wu *et al.*, 2019], which rely on localized message passing or propagation-based aggregation to learn node representations. In addition, we compare DNFormer with four recent Graph Transformer (GT) architectures, including NodeFormer [Wu *et al.*, 2022], SGFormer [Wu *et al.*, 2023], Polynormer [Deng *et al.*, 2024], and DUALFormer [Zhuo *et al.*, 2025], which adopt efficient attention designs to model long-range dependencies and higher-order interactions on graphs, such as sparse or approximated global attention and hybrid local–global aggregation schemes. These baselines comprehensively cover both conventional GNN-based and Transformer-based graph learning paradigms, providing a basis for fair comparison.

Implementation Details. All experiments were conducted on a single server equipped with an NVIDIA RTX 4090 GPU (24 GB). The software environment is built on Python 3.10, PyTorch 2.4.1, and PyTorch Geometric 2.6.1. For each dataset, we adopt a fixed data split ratio of 60% / 20% / 20% for training, validation, and testing, respectively. Each experiment is repeated 10 times and we report the mean accuracy and standard deviation to ensure statistical reliability. All models are trained using the Adam optimizer.

4.2 Performance Analysis

We first summarize the main empirical finding: DNFormer delivers stable and competitive accuracy across diverse graphs, with particularly notable gains on datasets whose local neighborhoods exhibit strong structural regularities. These results suggest that improving the discriminability of neighborhood-restricted attention is a key factor for boosting node classification performance in practice.

As shown in Table 2, DNFormer achieves consistently strong performance across all seven benchmarks and outperforms both representative GNN baselines and recent Graph Transformer variants on most datasets. The gains are particularly clear on CiteSeer, CS, and the Amazon co-purchase graphs, where local neighborhoods often contain repeated structural patterns (e.g., hub-centric neighborhoods and triangle-rich regions). In such cases, standard local attention can become less discriminative because many neighbors share similar topology-induced scores, leading to low-contrast attention weights after softmax normalization. DNFormer addresses this issue by using differential local attention to cancel the shared neighborhood-level component in attention scores, making neighbor-specific differences more salient. This results in higher-contrast attention distributions within each neighborhood, which helps the model identify and aggregate more informative local signals. Overall, the results indicate that suppressing common-mode effects in local attention improves node classification across graphs with diverse structural and semantic characteristics.

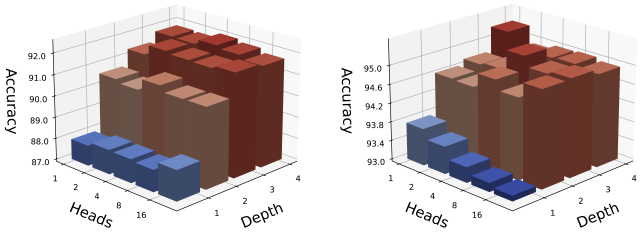
4.3 Ablation Study

We then verify the contribution of each design choice by ablating the two core components of DNFormer. The ablation results consistently show that both differential local attention and edge-adaptive modulation contribute to the final performance, and they complement each other in improving discriminative local aggregation.

To examine the contribution of each component in DNFormer, we conduct ablation studies by removing (i) the differential attention operation and (ii) the edge-adaptive modulation module. The results are summarized in Table 3. First, removing differential attention (denoted as DNFormer$_{w/o\ \text{diff}}$) consistently degrades performance across all datasets, with the largest drops observed on CiteSeer and CS. This supports our motivation that differential attention plays a key role in suppressing the shared, topology-induced component that tends to dominate neighborhood-restricted attention. Without the subtraction-based formulation, the model reverts to a standard single-sco local attention scheme, where attention weights are more likely to become uniform within a neighborhood, making them less effective at distinguishing informative neighbors. Second, removing edge-adaptive modulation (denoted as DNFormer$_{w/o\ \text{adapt}}$) also leads to noticeable performance declines, although the magnitude is generally smaller than that observed with differential attention removal. This suggests that learning an edge-specific scaling factor for the subtractive component improves flexibility: different edges and neighborhoods may require different degrees of common-mode suppression, depending on local structure and feature semantics. The effect is more evident on denser graphs such as Computers and Photo, where neighborhoods are larger and structural regularities are stronger. Overall, both components contribute positively and complement each other, and the full DNFormer yields the best results across all benchmarks.

Model	Cora	CiteSeer	PubMed	Computers	Photo	CS	Physics
DNFormer	**86.35 ± 0.40**	**77.36 ± 0.73**	**89.57 ± 0.31**	**92.13 ± 0.27**	**95.14 ± 0.25**	**95.28 ± 0.09**	**96.95 ± 0.05**
DNFormer$_{w/o\ \text{diff}}$	86.13 ± 0.69	76.82 ± 0.61	89.33 ± 0.23	92.10 ± 0.12	95.03 ± 0.14	94.59 ± 0.10	96.92 ± 0.08
DNFormer$_{w/o\ \text{adapt}}$	86.24 ± 0.38	77.03 ± 0.41	89.26 ± 0.09	91.94 ± 0.29	94.99 ± 0.26	94.74 ± 0.09	96.93 ± 0.05

Table 3: Ablation studies of DNFormer. The best results are in bold.



(a) Computers (b) Photo

Figure 2: Hyperparameter sensitivity of DNFormer to the number of attention heads and model depth.

4.4 Hyper-parameter Analysis

We further study the robustness of DNFormer with respect to architectural hyperparameters by varying the number of attention heads and model depth on Computers and Photo.

As shown in Figure 2, for the number of heads, we observe that small head counts (e.g., 1–2 heads) generally perform best, while using many heads (e.g., 8–16) tends to reduce accuracy. A plausible explanation is that excessive head partitioning fragments the feature space and weakens the per-head signal-to-noise ratio in neighborhood-restricted attention, making it harder to preserve meaningful neighbor-specific differences after normalization. In contrast, increasing depth provides a consistent improvement trend from 1 to 3–4 layers on both datasets, suggesting that stacking multiple DNFormer layers helps integrate multi-hop context while maintaining discriminative local aggregation through common-mode suppression. The improvement becomes less pronounced beyond 3 layers, indicating diminishing returns. On the whole, the performance landscape is smooth, implying that our proposed DNFormer is relatively stable under moderate hyperparameter perturbations. In practice, we find that 2 heads and 2–3 layers provide a strong trade-off between accuracy and efficiency.

5 Related Work

Graph Neural Networks. Graph Neural Networks (GNNs) [Gilmer *et al.*, 2017] are a standard paradigm for graph representation learning, where node representations are updated by aggregating messages from local neighborhoods. Classical models such as GCN [Kipf and Welling, 2017] and GraphSAGE [Hamilton *et al.*, 2017] achieve strong performance, but capturing long-range dependencies often requires deep message passing, which can suffer from over-smoothing [Oono and Suzuki, 2020] and over-squashing [Alon and Yahav, 2021]. These limitations motivate attention-based architectures that model graph interactions more flexibly than fixed neighborhood aggregation. Meanwhile, attention-based models can decouple information mixing from rigid multi-hop propagation, offering a more direct mechanism to integrate distant evidence when equipped with appropriate structural inductive biases.

Graph Transformers. Transformer architectures have been widely applied to various structured and sequential data analysis tasks, such as malicious domain name classification in cybersecurity [Ding *et al.*, 2023]. Graph Transformers (GTs) adapt self-attention to graphs for adaptive interaction modeling [Ying *et al.*, 2021; Dwivedi and Bresson, 2020; Kreuzer *et al.*, 2021]. Many GTs incorporate topology into attention via structural encodings or bias terms, such as shortest-path and centrality biases in Graphormer [Ying *et al.*, 2021], positional encodings and local message passing in GraphGPS [Dwivedi and Bresson, 2020], and structure-biased attention in SAN [Kreuzer *et al.*, 2021]. However, full self-attention over all node pairs is expensive on large graphs, so many recent GTs adopt efficient designs, including sparse, sampled, or neighborhood-restricted (local) attention [Deng *et al.*, 2024; Shirzad *et al.*, 2023] and hybrid local–global schemes [Xing *et al.*, 2024; Wu *et al.*, 2023]. While local attention improves scalability, many neighbors share similar local topology, which can introduce a shared component in attention scores and reduce score contrast after softmax normalization. DNFormer addresses this by subtracting multiple topology-aware scores to cancel the shared component and further uses edge-adaptive modulation to adjust the suppression strength across local contexts.

6 Conclusion

We have proposed DNFormer, a improved Graph Transformer, to introduce differential local attention to mitigate the common-mode effect in neighborhood-restricted attention. The core idea is to compute multiple topology-aware attention scores within the same neighborhood and combine them via a subtractive operation, which cancels the effect of shared topology-induced components and highlights neighbor-specific differences. We further incorporated an edge-adaptive modulation module, enabling the model to adjust the degree of common-mode suppression to different local contexts while retaining the scalability of local attention. Extensive experiments on seven node classification benchmarks demonstrate that DNFormer consistently outperforms representative GNNs and recent Graph Transformers. Future work includes extending DNFormer to heterogeneous and dynamic graphs and exploring its applicability to edge-level and graph-level learning tasks.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62422210, No. 62276187, No. 62402337 and No. 62261045), the Hong Kong RGC theme-based Strategic Target Grant Scheme (STG STG1/M-501/23-N), the Hong Kong Global STEM Professor Scheme, and the Hong Kong Jockey Club Charities Trust.

References

[Alon and Yahav, 2021] Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. In *Proceedings of the 9th International Conference on Learning Representations*, 2021.

[Chen *et al.*, 2023] Jinsong Chen, Kaiyuan Gao, Gaichao Li, and Kun He. Nagphormer: A tokenized graph transformer for node classification in large graphs. In *Proceedings of the The 11th International Conference on Learning Representations*, 2023.

[Deng *et al.*, 2024] Chenhui Deng, Zichao Yue, and Zhiru Zhang. Polynormer: Polynomial-expressive graph transformer in linear time. In *Proceedings of the International Conference on Learning Representations*, 2024.

[Ding *et al.*, 2023] Ling Ding, Peng Du, Haiwei Hou, Jian Zhang, Di Jin, and Shifei Ding. Botnet DGA domain name classification using transformer network with hybrid embedding. *Big Data Res.*, 33:100395, 2023.

[Dwivedi and Bresson, 2020] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*, 2020.

[Dwivedi *et al.*, 2023] Vijay Prakash Dwivedi, Yozen Liu, Anh Tuan Luu, Xavier Bresson, Neil Shah, and Tong Zhao. Graph transformers for large graphs. *arXiv preprint arXiv:2312.11109*, 2023.

[Gilmer *et al.*, 2017] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 1263–1272, 2017.

[Hamilton *et al.*, 2017] William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 1024–1034, 2017.

[Kipf and Welling, 2017] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference on Learning Representations*, 2017.

[Klicpera *et al.*, 2019] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *Proceedings of the 7th International Conference on Learning Representations*, 2019.

[Kreuzer *et al.*, 2021] Devin Kreuzer, Dominique Beaini, William L. Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph transformers with spectral attention. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 21618–21629, 2021.

[Li *et al.*, 2018] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018.

[McAuley *et al.*, 2015] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 43–52, 2015.

[Oono and Suzuki, 2020] Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *Proceedings of the 8th International Conference on Learning Representations*, 2020.

[Sen *et al.*, 2008] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. Collective classification in network data. *AI Mag.*, 29(3):93–106, 2008.

[Shirzad *et al.*, 2023] Hamed Shirzad, Ameya Velingker, Balaji Venkatachalam, Danica J Sutherland, and Ali Kemal Sinop. Exphormer: Sparse transformers for graphs. In *Proceedings of the International Conference on Machine Learning*, pages 31613–31632, 2023.

[Sinha *et al.*, 2015] Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June Hsu, and Kuansan Wang. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th International Conference on World Wide Web*, pages 243–246, 2015.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[Velickovic *et al.*, 2018] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *Proceedings of the 6th International Conference on Learning Representations*, 2018.

[Wu *et al.*, 2019] Felix Wu, Amauri H. Souza Jr., Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Q. Weinberger. Simplifying graph convolutional networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 6861–6871, 2019.

[Wu *et al.*, 2022] Qitian Wu, Wentao Zhao, Zenan Li, David P. Wipf, and Junchi Yan. Nodeformer: A scalable graph structure learning transformer for node classification. In *Proceedings of the Advances in Neural Information Processing Systems*, 2022.

[Wu *et al.*, 2023] Qitian Wu, Wentao Zhao, Chenxiao Yang, Hengrui Zhang, Fan Nie, Haitian Jiang, Yatao Bian, and Junchi Yan. Sgformer: Simplifying and empowering

transformers for large-graph representations. In *Proceedings of the Advances in Neural Information Processing Systems*, 2023.

[Xing *et al.*, 2024] Yujie Xing, Xiao Wang, Yibo Li, Hai Huang, and Chuan Shi. Less is more: on the over-globalizing problem in graph transformers. In *Proceedings of the 41st International Conference on Machine Learning*, pages 54656–54672, 2024.

[Xu *et al.*, 2019] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *Proceedings of the 7th International Conference on Learning Representations*, 2019.

[Ying *et al.*, 2021] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? In *Proceedings of the Advances in Neural Information Processing Systems*, pages 28877–28888, 2021.

[You *et al.*, 2020] Jiaxuan You, Zhitao Ying, and Jure Leskovec. Design space for graph neural networks. In *Proceedings of the Advances in Neural Information Processing Systems*, 2020.

[Zhang and Chen, 2018] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. *Proceedings of the Advances in Neural Information Processing Systems*, 31, 2018.

[Zhang *et al.*, 2018] Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. An end-to-end deep learning architecture for graph classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[Zhuo *et al.*, 2025] Jiaming Zhuo, Yuwei Liu, Yintong Lu, Ziyi Ma, Kun Fu, Chuan Wang, Yuanfang Guo, Zhen Wang, Xiaochun Cao, and Liang Yang. Dualformer: Dual graph transformer. In *Proceedings of the 13th International Conference on Learning Representations*, 2025.