Adaptive Fusion of Global-to-Local Patterns for Graph Representation Learning

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Graph Neural Networks (GNNs) have achieved remarkable success in modeling local structures through message passing. However, their ability to capture long-range dependencies remains limited due to the inherently local nature of aggregation. To address this, recent studies have proposed Transformer-based GNNs that incorporate global attention mechanisms, enabling direct interactions between distant nodes. While these models improve access to global context, they often treat global and local information as parallel or independent, lacking a principled way to integrate the two. We argue that effective graph understanding requires a hierarchical modeling approach in which global structure guides the interpretation of local patterns. Without global context, structurally similar local subgraphs can be ambiguous or misleading. To address this, we propose a novel GNN that explicitly conditions local representation learning on a global embedding. Specifically, we first encode global features using graph diffusion, then induce local contrastive dynamics that refine node representations in a globally informed manner. Finally, our adaptive fusion mechanism integrates global and local signals based on graph topology and task characteristics. We validate our model on a wide range of benchmarks, including six graph classification datasets including small molecules, bioinformatics, and social networks, two long-range graph benchmark datasets, and four node classification datasets. Compared to both message-passing and Transformer-based GNNs, our model shows consistent performance gains, achieving up to 11.01%p improvement. In addition, detailed ablation studies, sensitivity analyses, and qualitative case studies confirm that our global-to-local design leads to more expressive graph representations.

1 Introduction

Graph-structured data are pervasive across domains such as social networks [49], molecular analysis [39], knowledge graphs, and recommender systems [38]. To model the rich relational structure inherent in such data, Graph Neural Networks (GNNs) have become the de facto standard. Message passing neural networks (MPNNs), in particular, achieve strong performance by recursively aggregating information from neighboring nodes [22, 6, 43, 2]. Yet, the locality of message passing imposes fundamental limitations on capturing long-range dependencies and high-level graph semantics [15]. To mitigate this, recent work has introduced Transformer-based GNNs that employ global attention to directly model pairwise interactions between nodes [42, 21, 46], regardless of their topological distance [36, 29]. Nonetheless, a key challenge remains: how to effectively fuse both global and local structural signals. While MPNNs capture fine-grained patterns in local neighborhoods, they struggle with global coherence. Conversely, Transformer-based architectures facilitate long-range dependencies but often neglect the structural inductive biases [12, 7, 10] encoded in local subgraphs [25].

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

While recent methods have attempted to address this limitation—for example, by incorporating inductive biases to inject global information by explicitly combining MPNNs or positional encodings [25, 21, 30, 7, 10]—these approaches generally treat global and local signals as separate auxiliary biases rather than modeling their interaction in a unified and learnable method. They typically adopt a monolithic or parallel scheme, integrating local inductive biases (e.g., message passing or structural encodings) as additional modules alongside global self-attention [25]. Such formulations lack hierarchical conditioning and do not align with human perceptual priors, where global understanding precedes detailed analysis [26]. As a result, they often fail to adaptively model structural features based on the specific demands of a given task or graph across multiple scales.

This presents a key challenge and motivation: (1) *how can we effectively capture fine-grained local information conditioned on broader global context?*; and (2) *how can we leverage both representations in a complementary and task-adaptive manner for accurate prediction?* Our approach differs in two important ways. First, we obtain initial embeddings that capture global structural information through graph diffusion, and subsequently induce a reaction process to extract local contrastive signals based on these global representations. Second, we propose an adaptive fusion mechanism that dynamically adjusts the contribution of global and local components. Rather than a fixed combination, the model learns to emphasize the appropriate structural information depending on the characteristics of the graph and the nature of the prediction task. Through this methodology, we enable effective graph representation learning and validate its effectiveness via comprehensive and detailed analyses. Our contributions can be summarized as follows:

- We propose a novel model that models global-to-local transitions continuously and fuses multi-scale information adaptively and show in-depth analyses on each component's role, showing how global and local signals are disentangled and integrated to improve predictive performance.
- We conduct comprehensive experiments across datasets spanning diverse domains and tasks, including graph classification, long-range benchmarks, and node classification, to verify the effectiveness and generality of our proposed method.

2 Preliminary

Definition 1 (Node classification)**.** *Given a single graph $G_1 = (\mathcal{V}, \mathcal{E}, X)$, where $X \in \mathbb{R}^{|\mathcal{V}| \times d}$ is the node feature matrix, the goal of node classification is to learn a function $f : \mathcal{V} \to \mathcal{Y}$ that predicts a label $y_v \in \mathcal{Y}$ for each node $v \in \mathcal{V}$.*

Definition 2 (Graph classification.)**.** *Given a set of graphs $G = \{G_1, G_2, \ldots, G_N\}$, where each graph $G_i = (\mathcal{V}_i, \mathcal{E}_i, X_i)$ consists of a node set $\mathcal{V}_i$, edge set $\mathcal{E}_i$, and node features $X_i \in \mathbb{R}^{|\mathcal{V}_i| \times d}$, the goal of graph classification is to learn a function $f : G \to \mathcal{Y}$ that predicts a label $y_i \in \mathcal{Y}$ for each graph G_i.*

3 Method

In this section, we propose a structured two-phase approach in which the global bias is first encoded through a diffusion process, followed by a conditional reaction mechanism that refines representations based on local contrasts (in §3.1). These two phases are then adaptively fused via an attention mechanism (in §3.2). This formulation ensures a clear functional disentanglement, enabling each module to learn a distinct inductive bias. Furthermore, the attention-based fusion allows fine-grained modulation of the contributions from global and local signals depending on the input graph and task context. We provide the overall architecture in Figure 1.

3.1 Continuous Global-to-Local Modeling

We adopt a continuous-depth modeling framework using Graph Neural Ordinary Differential Equations (Graph Neural ODEs). Given initial node features $h(0) \in \mathbb{R}^{|\mathcal{V}| \times d_h}$, the evolution of node representations is governed by the ODE: $\frac{dh(t)}{dt} = f(h(t), t; \theta), \quad h(0) = \epsilon(X; \theta_\epsilon)$, where ϵ is an encoder and f is a learnable dynamics function. The final representation is obtained by integrating over time t): $h(T) = h(0) + \int_0^T f(h(t), t; \theta)\, dt$. We use this for two major advantages: (1) it enables

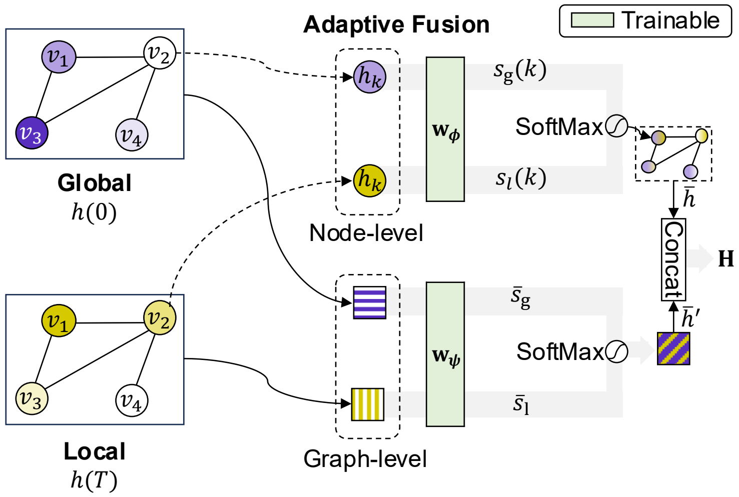


Figure 1: Illustration of our overall architecture.

gradual, controllable transitions in representation space; (2) it allows structural priors such as *global* and *local* inductive bias to be encoded via time-dependent dynamics.

We design the following sequential dynamics:

$$f(h,t) := \frac{dh(t)}{dt} = \begin{cases} h_{\text{global}}, & \text{if } t \in [0, T_1], \\ h_{\text{local}}, & \text{if } t \in (T_1, T], \end{cases} \tag{1}$$

where T is a hyperparameter. Our model f first encodes smooth, global topology and then refines local discriminative structure. We do not fix the value of T_1; instead, we allow the embedding to transition smoothly from global to local representations in an end-to-end manner. First, we use the following dynamics [9] as a basis.

Definition 3 (Reaction-diffusion dynamics)**.** *Let $h(0) = \epsilon(X, \theta_\epsilon)$ be the output of an encoder. Reaction-diffusion dynamics are:*

$$\frac{dh(t)}{dt} = -\alpha \tilde{L} h(t) + \beta(\tilde{A} - \tilde{A}^2) h(t), \tag{2}$$

where $\tilde{L}$ is the normalized Laplacian, $\tilde{A}$ is a soft-adjacency matrix, and $\alpha, \beta \in \mathbb{R}^{d_h}$ are learnable parameters controlling the intensity of each term. $-\alpha \tilde{L} h(t)$ serves as diffusion while $\beta(\tilde{A} - \tilde{A}^2)h(t)$ serves as reaction on graph.

In order to enforce Eq. 2 in a manner consistent with Eq. 1, we inject global structural information in $h(t) \in [0, T_1]$ with the following loss:

Definition 4 (Global embedding loss $\mathcal{L}_{GE}$)**.** *The model enforces $h(0)$ to approximate the global diffusion-only solution by minimizing the loss: $\mathcal{L}_{GE} := \frac{1}{|\mathcal{V}|} \sum_i \left\| h_i(0) - \int_0^T \alpha \tilde{L} h_i(t) dt \right\|^2$. This ensures that $h(t)$ at early time steps (i.e., near $t = 0$) reflects a globally smoothed representation, providing a globally coherent initialization for the subsequent process.*

Lemma 1. *Minimizing $\mathcal{L}_{GE}$ encourages the encoded $h(0)$ to approximate the equilibrium point of the globally smoothed representation, i.e., $h(0) \approx \int_0^T \alpha \tilde{L} h(t) dt$.*

Proof. Minimizing $\mathcal{L}_{GE}$ effectively suppresses high-frequency components in $h(0)$ by penalizing its deviation from $\int_0^T \alpha \tilde{L} h(t)\, dt$. Since the normalized Laplacian $\tilde{L}$ acts as a low-pass filter in the spectral domain, this encourages $h(0)$ to align with the the low-frequency spectrum associated with global smoothness. □

The detailed proof is provided in Appendix E.1. After enforcing the globally smoothed information in $t \in [0, T_1]$ with $\mathcal{L}_{GE}$, we now focus on $t \in [T_1, T]$ and provide the following theorem.

Theorem 1. *By definition 4 and minimizing $\mathcal{L}_{GE}$ as derived from lemma 1, $h(T)$ can be approximated by $h(T) = h(0) + \int_0^T -\alpha\tilde{L}h(t)dt + \int_0^T \beta(\tilde{A} - \tilde{A}^2)h(t)dt \approx \int_0^T \beta(\tilde{A} - \tilde{A}^2)h(t)dt$.*

The detailed proof is provided in Appendix E.2. This isolates the reaction term and yields a representation that is sensitive to fine-grained neighborhood contrasts and more adaptive to localized task objectives by the following lemma.

Lemma 2 (Graph reaction emphasizes local structural information)**.** *Let $\tilde{A}$ be the normalized adjacency matrix with self-loops, and let $h(t)$ be the representation of the node evolving over time under the ODE defined in Theorem 4. Then the reaction term $(\tilde{A} - \tilde{A}^2)h(t)$ emphasizes local structural deviations by enhancing node-wise contrasts among neighbors, thus inducing a high-frequency (local) signal in $h(T)$.*

Proof. We analyze the spectral and combinatorial properties of the reaction operator $(\tilde{A} - \tilde{A}^2)$. First, observe that $\tilde{A}$ is a row-normalized adjacency matrix with self-loops, such that $\tilde{A}_{ij} > 0$ only if nodes i and j are neighbors (or $i = j$). Thus, $\tilde{A}h(t)$ performs a first-order neighborhood aggregating of node features, a low-pass filtering operation in the spectral domain. Next, consider $\tilde{A}^2$, which captures second-order neighborhoods via: $\tilde{A}^2_{ij} = \sum_k \tilde{A}_{ik}\tilde{A}_{kj}$, which is non-zero if there exists a two-hop path from i to j. Hence, $\tilde{A}^2h(t)$ propagates features beyond immediate neighbors, smoothing over broader neighborhoods. Now, the difference $(\tilde{A} - \tilde{A}^2)$ subtracts second-order propagation from first-order propagation. This has two effects: (1) It cancels redundant or transitive similarity, e.g., when node i and node j are both strongly connected to a common neighbor k, reducing their similarity unless they are directly connected. (2) It accentuates edge-level contrasts where direct neighbors $i \sim j$ do not share many common second-hop paths (i.e., low transitive closure), which is common in boundary regions between structural clusters. Spectrally, $(\tilde{A} - \tilde{A}^2)$ behaves as a high-pass filter, as it suppresses low-frequency signals (smooth components shared across neighbors) and preserves sharp variations. Therefore, $(\tilde{A} - \tilde{A}^2)h(t)$ highlights localized deviations in structure—nodes with differing feature dynamics from their neighbors are amplified. Consequently, as $h(T)$ is largely governed by the accumulation of this term (cf. Theorem 4), it captures localized, contrastive patterns, validating its interpretation as a local representation. □

Corollary 1. *The representation $h(t)$ transitions from capturing global structural information at the initial stage ($t \approx 0$) to encoding more localized features near the final time step ($t \approx T$).*

We now solve the ODE using the Euler method [1, 8], a widely used numerical scheme for approximating solutions to differential equations: $h(t_{k+1}) = h(t_k) + \tau \cdot f(h, t_k)$, where time-step τ. Upon solving the ODE, we extract and utilize $h(0)$ as a global representation and $h(T)$ as a local representation.

3.2 Adaptive Fusion of Global and Local Representation Pairs

We now design the adaptive modeling of global and local embeddings based on the following conjecture:

Conjecture 1. *The relative importance of global and local information may vary across nodes or graphs, depending on the requirements of the downstream task. Thus, adaptive fusion of both information is essential to constructing expressive and task-relevant embeddings.*

Building on this conjecture, we dynamically adjust the contributions of the global embedding $h(0)$ and the local embedding $h(T)$ based on the task. We first define node-level attention scores as follows:

$$s_g(i) = \cos(h_i(0), w_\phi), \quad s_l(i) = \cos(h_i(T), w_\phi), \tag{3}$$

$$a_i = \text{softmax}([s_g(i), s_l(i)]), \tag{4}$$

$$h_i' = \text{Pool}(\alpha_i h_i(0) + (1 - a_i)h_i(T)), \tag{5}$$

where $\cos(\cdot, \cdot)$ denotes cosine similarity, and $w_\phi \in \mathbb{R}^{d_h}$ is a learnable vector that captures task-specific preferences over global and local representations.

Similarly, at the graph level, we compute attention weights using the pooled representations of global and local embeddings:

$$\bar{s}_g = \cos(\text{Pool}(h_i(0)), w_\psi), \quad \bar{s}_l = \cos(\text{Pool}(h_i(T)), w_\psi), \tag{6}$$

$$\bar{a} = \text{softmax}([\bar{s}_g, \bar{s}_l]), \tag{7}$$

$$\bar{h} = \bar{a} \cdot \text{Pool}(h_i(0)) + (1 - \bar{a}) \cdot \text{Pool}(h_i(T)), \tag{8}$$

where $w_\psi \in \mathbb{R}^{d_h}$ is a learnable vector for graph-level fusion.

Finally, the prediction representation is obtained by concatenating the graph-level and aggregated node-level features: $\mathbf{H} \in \mathbb{R}^{|\mathcal{V}| \times 2d_h} = [\sum_i h'_i, \bar{h}]$.

This mechanism enables the model to assess, for each instance, the relative alignment of global and local embeddings with task-specific query vectors (w_ϕ, w_ψ). The resulting attention scores modulate the contribution of each component, allowing adaptive fusion of structural signals across different scales.

3.3 Training

The final representation $\mathbf{H}$ is mapped to a prediction vector $\hat{y}$ using fully connected (FC) layers, where $\hat{y} = \text{FC}(\mathbf{H})$. The ground-truth labels y are represented using one-hot encoding and the. Total training loss function is $\mathcal{L}_{\text{total}} = \lambda_{\text{cls}}\mathcal{L}_{\text{cls}} + \lambda_{GE}\mathcal{L}_{\text{GE}}$, where $\mathcal{L}_{\text{cls}} = -\sum_{i=1}^{\text{C}} y_i \log(\hat{y}_i)$ is the cross-entropy loss for C classes [50]. We set λ_{cls} to 10 in all experiments. The ODE gradients are computed using the adjoint method introduced by [8]. This enables end-to-end training with negligible memory overhead while preserving the temporal dynamics necessary to learn meaningful transitions from global to local structure.

4 Experiment

4.1 Setup

Datasets. To ensure diversity, we test on a diverse set of real-world graph classification datasets, including 1 small molecules dataset PTC_MR [35], 2 bioinformatics datasets (D&D [11] , PROTEINS [4]) and 3 social networks datasets (IMDB-B [44], IMDB-M [44], REDDIT-B [44]). We also conducted experiments on the long-range graph benchmark [13] (LRGB), a recently introduced suite designed to assess the ability of GNNs to capture long-range dependencies in complex graph structures. Specifically, we test on Peptides-func and Peptides-struct datasets from the LRGB. These require models to effectively capture and utilize global information to handle not only local interactions but also distant node interactions across the entire graph. We further evaluate our model on three widely-used homophilous citation network datasets for node classification: Cora [27], CiteSeer [3], PubMed [32] and a heterophilous Wikipedia network dataset Chameleon [31]. More details about the datasets are in Appendix B.

Implementation details. In all experiments, we use Adam [18] as the optimizer. The learning rate is initialized at 0.001 and reduced by 1% every 20 epochs. The weight decay is set to 1e-5. The number of encoding layer is set to 2. For graph classification tasks, we follow a standard 10-fold cross-validation by randomly splitting the dataset into 10 splits and reporting the mean accuracy across folds. On the LRGB, we followed the evaluation scheme from [30] and adopt the train/validation/test splits provided by the benchmark and perform 4 independent runs using different random seeds. For node classification experiments, we follow the evaluation scheme proposed in [24], splitting the nodes into 60% for training, 20% for validation, and 20% for testing, except for the Chameleon dataset, where the splits are provided. We repeat each experiment with 10 different random seeds and report the mean and standard deviation of semi-supervised test performance. More hyperparameter settings and details are provided in Appendix C.

Comparison baselines. To evaluate the effectiveness of our model and ensure a comprehensive evaluation, we compare our model against a diverse set of strong state-of-the-art baselines encompassing both classical MPNN-based models and recent Transformer-based graph models. For MPNN-based GNNs, we include GCN [19], GraphSAGE [16], GAT [37], GatedGCN [5],

Table 1: Comparative performance on graph classification datasets. The mean ± s.d. of 10 cross-validation folds are reported. The top is in **bold** and the second is underlined.

Model	PTC_MR	D&D	PROTEINS	IMDB-B	IMDB-M	REDDIT-B
GCN	62.3±5.7	79.1±3.1	75.9±2.8	73.3±5.3	51.2±5.1	89.3±3.3
GraphSAGE	–	65.8±4.9	65.9±2.7	72.4±3.6	49.9±5.0	84.3±1.9
GAT	–	–	74.7±2.2	75.8±2.3	47.8±3.1	–
DiffPool	63.4±1.0	76.9±4.4	75.2±4.0	70.1±6.3	47.2±1.8	89.1±1.6
InfoGraph	61.7±1.4	72.9±1.8	74.4±0.3	73.0±0.9	36.7±0.8	82.5±1.4
GraphTrans	–	75.2±4.8	73.9±3.8	73.1±2.1	–	88.6±1.3
SAN	–	–	74.1±3.1	72.1±2.3	–	–
Graphormer	71.4±5.2	–	76.3±2.7	70.3±0.9	48.9±2.0	–
GraphGPS	–	76.0±1.5	75.8±2.3	77.4±0.6	–	88.4±1.2
NAGphormer	66.5±5.6	–	74.6±3.0	74.7±4.1	51.7±3.5	–
SGFormer	65.2±4.2	–	74.6±3.0	74.7±4.1	**56.4±3.4**	–
Gradformer	–	–	77.5±1.9	77.1±0.5	–	–
Ours	**74.7±8.3**	**83.4±2.6**	**80.0±3.7**	**78.2±4.3**	54.8±2.6	**90.8±2.0**

DiffPool [48], GINE [17], and InfoGraph [34]. For Transformer based GNNs, we include GraphTrans [42], SAN [21], Graphormer [46], GraphGPS [30], NAGphormer [7], NodeFormer [40], Exphormer [33], GOAT [20], SGFormer [41], Polynormer [10], and Gradformer [23]. The baseline results are taken from [23, 30, 45, 24].

4.2 Model Performance

Graph classification results. Table 1 shows that our method consistently achieves the best performance across all six graph classification datasets, outperforming both MPNN-based (e.g., GCN, GAT) and Transformer-based (e.g., GraphGPS, SAN, SGFormer) baselines. In particular, our model demonstrates remarkable performance on large-scale datasets with many nodes and edges (reter to Table 1), such as D&D (83.4%) and REDDIT-B (90.8%), surpassing the second-best models by +3.3% and +1.5%, respectively. These results highlight the strength of our global-to-local modeling in capturing complex structural patterns essential for accurate graph-level prediction.

Node classification results. Table 2 shows that our model achieves the best performance across all four node classification datasets, including homophilous graphs (Cora, CiteSeer, PubMed) and the heterophilous graph Chameleon. While our method outperforms existing MPNN and Transformer-based baselines on standard benchmarks like Cora (86.72%), CiteSeer (75.57%), and PubMed (88.99%), the most notable gain is observed on Chameleon, where our model achieves 57.29%—a substantial +11.01% improvement over the second-best method. This highlights the strength of our approach in handling both homophilous and heterophilous structures. Notably, our model outperforms all other Transformer-based baselines designed to capture both local and global information, including Polynormer, which adopts a local-to-global attention scheme in contrast to our global-to-local modeling strategy—demonstrating the effectiveness of our method.

Table 2: Performance on node classification datasets. The mean ± s.d. of 10 splits are reported.

Model	Cora	CiteSeer	PubMed	Chameleon
GCN	85.10±0.67	73.14±1.54	81.12±0.52	46.28±3.40
GraphSAGE	83.88±0.65	72.26±0.55	79.72±0.50	44.81±4.74
GAT	84.46±0.55	72.22±0.84	80.28±0.52	44.13±4.17
GraphGPS	83.87±0.96	72.73±1.28	79.94±0.43	41.55±3.91
NAGphormer	80.92±1.17	70.59±0.89	80.14±1.06	–
NodeFormer	82.73±0.75	72.37±1.27	79.59±0.92	36.38±3.85
Exphormer	83.29±1.14	71.85±1.19	79.67±0.67	–
GOAT	83.26±1.24	72.21±1.12	79.60±1.05	–
SGFormer	84.82±0.85	72.72±1.30	80.60±0.49	45.21±3.72
Polynormer	83.43±0.89	72.19±1.03	79.63±0.40	41.97±3.18
Ours	**86.72**±1.11	**75.57**±1.41	**88.99**±0.43	**57.29**±1.72

Long-range benchmark results. Table 3 reports results on the LRGB benchmark, where Peptides-struct and Peptides-func datasets with a large graph diameter (longest shortest path) of 56 (refer to Table 1), requiring effective modeling of long-range dependencies. In such settings, capturing both global context across distant nodes and preserving local structural nuances is critical. Our model achieves the lowest MAE on Peptides-struct (0.2498) and performs competitively on Peptides-func (63.12 AP), closely matching the best-performing baseline GraphGPS (63.55 AP). These results demonstrate that our model effectively balances long-range global signal propagation with localized information, validating its robustness in large-diameter graph scenarios.

4.3 Model Analysis

Ablation studies. Table 4 presents the results of an ablation study to analyze the contributions of key components. It shows the accuracy of the model under different configurations, highlighting the individual and combined contributions of each component to the overall performance. In the last row, the full method achieves the highest accuracy. The role of global encoding is evident when it is not applied as $\lambda_{GE} = 0$, leading to a drop in accuracy (See the first, third, seventh row). While global encoding λ_{GE} does not show an effect when used with global alone, a synergistic effect can be observed with Local (See the second row). This is evidenced by the improved performance compared to the case without global encoding. We also observe that our adaptive fusion at the node and graph levels play a crucial role in the model's performance. Disabling node-level fusion (in the sixth row) results in a slight drop in accuracy. Similarly, removing graph-level fusion (in the fifth row) reduces the accuracy, reflecting its importance.

Table 3: Performance on long-range graph benchmark. The mean ± s.d. of 4 runs with different random seeds are reported.

Model	**Peptides-struct**	**Peptides-func**
	MAE↓	AP↑
GCN	0.3496±0.0013	0.5930±0.0023
GINE	0.3547±0.0045	0.5498±0.0079
GatedGCN	0.3420±0.0013	0.5864±0.0077
GatedGCN+RWSE	0.3357±0.0006	0.6069±0.0035
Transformer+LapPE	0.2529±0.0016	0.6326±0.0126
SAN+LapPE	0.2683±0.0043	0.6384±0.0121
SAN+RWSE	0.2545±0.0012	0.6439±0.0075
GraphGPS	0.2500±0.0005	**0.6535±0.0041**
Ours	**0.2498±0.0020**	0.6312±0.0056

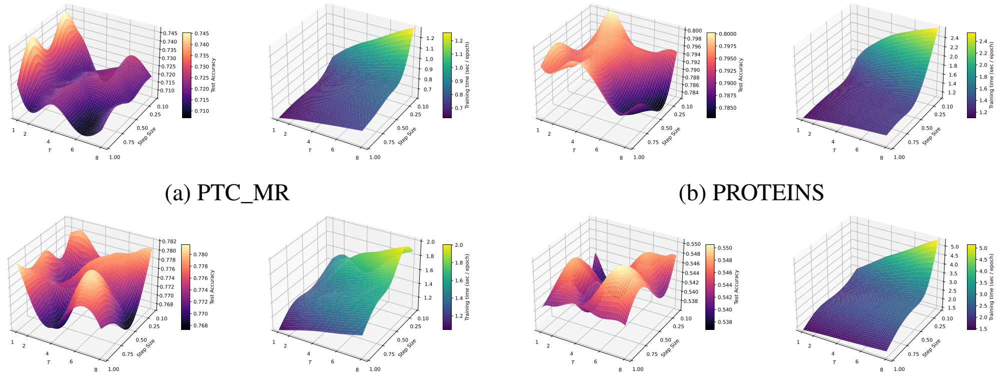
Table 4: Ablation study. **#** refers to the rank based on test accuracy.

$h(0)$	$\mathcal{L}_{GE}$	$h(T)$	**Adaptive Fusion**		**PROTEINS**		**IMDB-B**		**IMDB-M**		**Chameleon**	
Global		Local	Node	Graph	**Accuracy ↑**	**#**	**Accuracy ↑**	**#**	**Accuracy ↑**	**#**	**Accuracy ↑**	**#**
✓					79.1±3.9	6	77.4±4.4	4	53.9±4.0	5	53.18±2.15	7
✓	✓				79.1±4.7	6	76.3±4.4	8	53.9±4.2	5	53.82±2.70	6
✓		✓			78.7±3.5	8	76.8±4.8	7	53.7±3.5	7	55.00±1.12	8
✓	✓	✓			79.3±5.0	4	77.0±3.8	6	54.3±4.4	2	55.07±2.28	5
✓	✓	✓	✓		79.3±4.7	4	77.2±4.4	2	54.1±4.0	4	55.88±1.46	2
✓	✓	✓		✓	79.7±4.7	2	77.4±3.8	4	53.5±4.2	8	55.46±3.10	4
✓		✓	✓	✓	79.4±3.7	3	77.7±3.6	3	54.3±3.5	2	55.57±2.34	3
✓	✓	✓	✓	✓	**80.0±3.7**	**1**	**78.2±4.3**	**1**	**54.8±3.6**	**1**	**57.29±1.72**	**1**

Global vs. local analysis. To empirically assess whether the global and local embeddings from Lemma 1 and Theorem1 are indeed effectively disentangled and extracted, we define and use a metric to quantify the level of local structural information captured by a node embedding.

Definition 5 (One-hop local similarity)**.** *Given a graph G with node embeddings $h \in \mathbb{R}^{|\mathcal{V}| \times d_h}$, we compute the cosine similarity between each node and its one-hop neighbors. For each node v, the local similarity score is defined as $sim(v) := \frac{1}{|\mathcal{N}(v)|} \sum_{u \in \mathcal{N}(v)} \cos(h_v, h_u)$, where $\mathcal{N}(v)$ denotes the set of one-hop neighbors of v. The average similarity over all nodes in the graph is then used as a measure of local consistency $\frac{1}{|V'|} \sum_{v \in V'} sim(v), \quad V' = \{v \in V \mid |\mathcal{N}(v)| > 0\}$.*

High cosine similarity between neighboring nodes often indicates successful representation learning in homophilic graphs, where connected nodes typically belong to the same class. This also suggests the formation of coherent local clusters aligned with the underlying graph structure. We assess this metric on both $h(0)$ and $h(T)$. As shown in Figure 2, across eight datasets, the representation $h(T)$ consistently yields higher one-hop neighbor similarity compared to $h(0)$. The results show a consistent increase in local similarity from $h(0)$to $h(T)$, implying that $h(T)$ captures localized cluster patterns, whereas $h(0)$ shows weak intra-cluster cohesion but suggests that it captures unique functional role and topological position within the graph. Complementarily, Figure 3 shows t-SNE visualizations of embeddings of $h(0)$ and $h(T)$. The observed boundary and between them in the embedding space further confirm that our method effectively disentangles and captures global and local patterns at different stages.



(a) PTC_MR (b) PROTEINS

(c) IMDB-B (d) IMDB-M

Figure 4: 3D visualization of sensitivity results w.r.t $T \in [1, 8]$ and step size $\tau \in [0.1, 1]$. Left: test accuracy. Right: training time per epoch.

Sensitivity analysis. The sensitivity analysis presented in Figure 4 evaluates the effects of two hyperparameters, T and step size τ, on test accuracy and training time. Training time per epoch increases monotonically with larger ODE depth T and smaller step size τ. This is because a smaller step size leads to a finer discretization of the ODE trajectory, which requires more numerical integration steps to reach the terminal time T. As a result, the total number of function evaluations scales approximately with T/τ, thereby increasing the computational cost. Our results show that while training time increases steadily with T/τ, test accuracy does not follow a monotonic trend. In terms of test accuracy, we observe that the model exhibits stable learning behavior under varying ODE configurations. Specifically, on the PTC_MR dataset (Figure 4a), the difference between the highest and lowest test accuracy is approximately 3%p. This small variation indicates that the model exhibits stable learning behavior under varying ODE configurations.

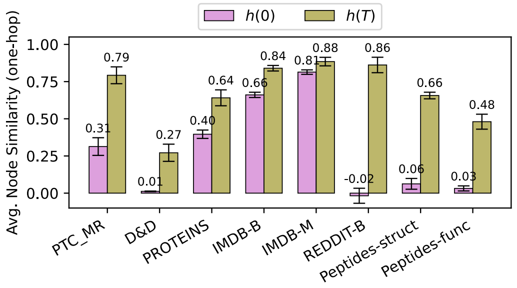
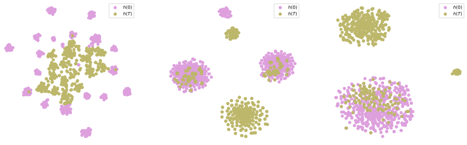


Figure 2: Comparison of average one-hop node similarity between extracted $h(0)$ and $h(T)$ from the test set. Mean and standard deviation are reported from 10-fold cross-validation.

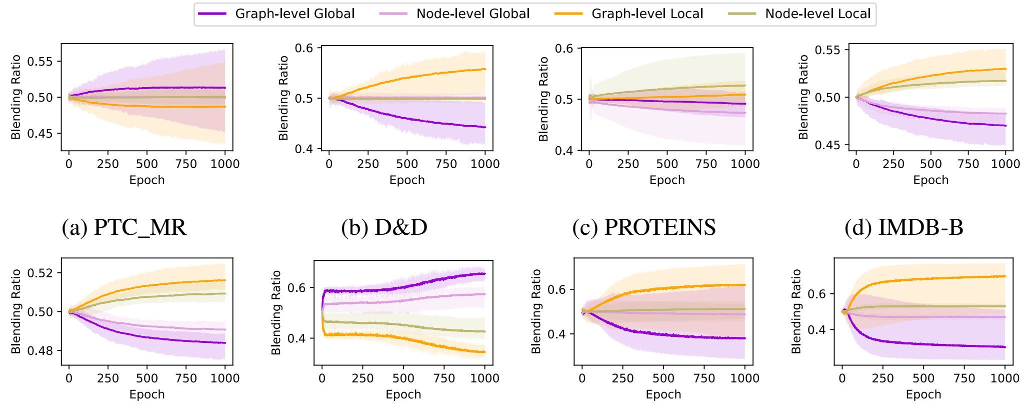


(a) D&D (b) PROTEINS (c) REDDIT-B

Figure 3: t-SNE visualization of node embeddings. Embeddings extracted from $h(0)$ are shown in pink, while those from $h(T)$ are shown in brown.

Quantitative analysis of adaptive fusion weights. We present quantitative evidence that supports conjecture 1, as shown in Figure 5. By analyzing the node- and graph-level weight a_i (in Eq. 4) and $\bar{a}$ (in Eq. 7) learned by our model, we observe that the model adaptively prioritizes global or local features depending on the graph domain and prediction task. For example, in the PTC_MR and REEDIT-B dataset, global features are weighted more heavily at the graph level, while in other datasets, local information dominates. This behavior empirically validates the need for task- and topology-dependent emphasis.

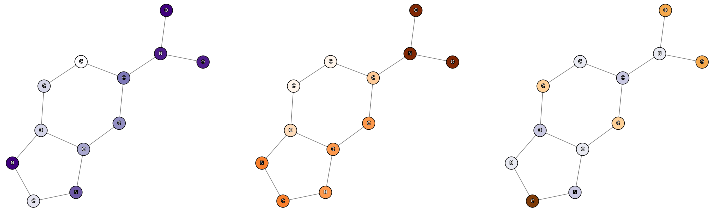
Case study. Figure 6 shows a visualization of a molecule from the PTC_MR dataset, where nodes correspond to atoms (carbon, nitrogen, oxygen) and edges represent chemical bonds. We visualize the node embeddings at three stages of our model: (a) global embedding $h(0)$, (b) local embedding $h(T)$, and (c) the adaptively fused embedding. In Figure 6a, we observe that that $h(0)$ encodes



(a) PTC_MR (b) D&D (c) PROTEINS (d) IMDB-B

(e) IMDB-M (f) REDDIT-B (g) Peptides-struct (h) Peptides-func

Figure 5: The contribution of global and local information according to the epochs. The shaded area denotes the minimum and maximum for all test samples, while the line represents the mean.



(a) $h(0)$ (b) $h(T)$ (c) Adaptive fusion

Figure 6: Visualization on PTC_MR.

topology-aware positional information rather than local structural similarity. The lack of localized color clustering implies that the embedding does not yet reflect neighborhood-level homogeneity, but rather captures a broad structural context within the molecular graph. In Figure 6b, $h(T)$ exhibits a different pattern. Nodes within structurally coherent subgraphs (e.g., rings and tails) display more uniform and distinguishable. This reflects that $h(T)$ emphasizes local substructural cluster patterns, indicating the model's ability to detect locally discriminative patterns. The fused embedding in Figure 6c, obtained via attention-based fusion of $h(0)$ and $h(T)$, reveals a more nuanced fusion of global and local signals. While certain atoms retain high local values in $h(T)$, others reflect the broader structural information seen in $h(0)$. This highlights the model's ability to adaptively balance the contribution of global versus local information based on the context, leading to enriched representations which are particularly valuable for tasks requiring both macro-level and micro-level discrimination.

5 Concluding remarks

In this paper, we proposed a novel GNN that performs continuous global-to-local modeling and adaptively fuses multi-scale structural signals. Motivated by the limitations of existing GNNs in jointly capturing global and local inductive biases, our method first encodes global structural information via graph diffusion, and then conditions local representations on the global context, followed by an adaptive fusion of the two to form a task-specific representation. Through extensive experiments on datasets spanning molecular graphs, social networks, long-range benchmarks, and both homophilous and heterophilous node classification tasks, we validated the robustness and generality of our approach. Detailed ablations and analyses further confirm the effectiveness of our method.

References

[1] K. Atkinson, W. Han, and D. E. Stewart. *Numerical solution of ordinary differential equations*, volume 81. John Wiley & Sons, 2009.

[2] D. Beaini, S. Passaro, V. Létourneau, W. Hamilton, G. Corso, and P. Liò. Directional graph networks. In *International Conference on Machine Learning*, pages 748–758. PMLR, 2021.

[3] A. Bojchevski and S. Günnemann. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. *arXiv preprint arXiv:1707.03815*, 2017.

[4] K. M. Borgwardt, C. S. Ong, S. Schönauer, S. Vishwanathan, A. J. Smola, and H.-P. Kriegel. Protein function prediction via graph kernels. *Bioinformatics*, 21(suppl_1):i47–i56, 2005.

[5] X. Bresson and T. Laurent. Residual gated graph convnets. *arXiv preprint arXiv:1711.07553*, 2017.

[6] H. Cai, V. W. Zheng, and K. C.-C. Chang. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE transactions on knowledge and data engineering*, 30(9):1616–1637, 2018.

[7] J. Chen, K. Gao, G. Li, and K. He. Nagphormer: A tokenized graph transformer for node classification in large graphs. *arXiv preprint arXiv:2206.04910*, 2022.

[8] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

[9] J. Choi, S. Hong, N. Park, and S.-B. Cho. Gread: Graph neural reaction-diffusion networks. In *International Conference on Machine Learning*, pages 5722–5747. PMLR, 2023.

[10] C. Deng, Z. Yue, and Z. Zhang. Polynormer: Polynomial-expressive graph transformer in linear time. *arXiv preprint arXiv:2403.01232*, 2024.

[11] P. D. Dobson and A. J. Doig. Distinguishing enzyme structures from non-enzymes without alignments. *Journal of molecular biology*, 330(4):771–783, 2003.

[12] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[13] V. P. Dwivedi, L. Rampášek, M. Galkin, A. Parviz, G. Wolf, A. T. Luu, and D. Beaini. Long range graph benchmark. In *Advances in Neural Information Processing Systems*, volume 35, pages 22326–22340, 2022.

[14] M. Fey and J. E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[15] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

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

[17] W. Hu, B. Liu, J. Gomes, M. Zitnik, P. Liang, V. Pande, and J. Leskovec. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*, 2019.

[18] D. P. Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907, 2016.

[20] 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*, pages 17375–17390. PMLR, 2023.

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

[22] S. Kumar, A. Mallik, A. Khetarpal, and B. S. Panda. Influence maximization in social networks using graph embedding and graph neural network. *Information Sciences*, 607:1617–1636, 2022.

[23] C. Liu, Z. Yao, Y. Zhan, X. Ma, S. Pan, and W. Hu. Gradformer: Graph transformer with exponential decay. *arXiv preprint arXiv:2404.15729*, 2024.

[24] Y. Luo, L. Shi, and X.-M. Wu. Classic gnns are strong baselines: Reassessing gnns for node classification. In *Advances in Neural Information Processing Systems*, volume 37, pages 97650–97669, 2024.

[25] L. Ma, C. Lin, D. Lim, A. Romero-Soriano, P. K. Dokania, M. Coates, P. Torr, and S.-N. Lim. Graph inductive biases in transformers without message passing. In *International Conference on Machine Learning*, pages 23321–23337. PMLR, 2023.

[26] M. Martin. Local and global processing: The role of sparsity. *Memory & Cognition*, 7:476–484, 1979.

[27] A. K. McCallum, K. Nigam, J. Rennie, and K. Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3:127–163, 2000.

[28] C. Morris, N. M. Kriege, F. Bause, K. Kersting, P. Mutzel, and M. Neumann. Tudataset: A collection of benchmark datasets for learning with graphs. In *ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020)*, 2020.

[29] N. Parmar, A. Vaswani, J. Uszkoreit, L. Kaiser, N. Shazeer, A. Ku, and D. Tran. Image transformer. In *International conference on machine learning*, pages 4055–4064. PMLR, 2018.

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

[31] B. Rozemberczki, C. Allen, and R. Sarkar. Multi-scale attributed node embedding. *Journal of Complex Networks*, 9(2):cnab014, 2021.

[32] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

[33] H. Shirzad, A. Velingker, B. Venkatachalam, D. J. Sutherland, and A. K. Sinop. Exphormer: Sparse transformers for graphs. In *International Conference on Machine Learning*, pages 31613–31632. PMLR, 2023.

[34] F.-Y. Sun, J. Hoffman, V. Verma, and J. Tang. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In *International Conference on Learning Representations*, 2019.

[35] H. Toivonen, A. Srinivasan, R. D. King, S. Kramer, and C. Helma. Statistical evaluation of the predictive toxicology challenge 2000–2001. *Bioinformatics*, 19(10):1183–1193, 2003.

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

[37] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[38] L. Waikhom and R. Patgiri. A survey of graph neural networks in various learning paradigms: methods, applications, and challenges. *Artificial Intelligence Review*, 56(7):6295–6364, 2023.

[39] Y. Wang, Z. Li, and A. Barati Farimani. Graph neural networks for molecules. In *Machine learning in molecular sciences*, pages 21–66. Springer, 2023.

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

[41] 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. *Advances in Neural Information Processing Systems*, 36:64753–64773, 2023.

[42] Z. Wu, P. Jain, M. Wright, A. Mirhoseini, J. E. Gonzalez, and I. Stoica. Representing long-range context for graph neural networks with global attention. *Advances in neural information processing systems*, 34:13266–13279, 2021.

[43] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[44] P. Yanardag and S. Vishwanathan. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1365–1374, 2015.

[45] Z. Yao, C. Liu, X. Ma, M. Chen, J. Wu, X. Cai, B. Du, and W. Hu. Dual-perspective cross contrastive learning in graph transformers. *arXiv preprint arXiv:2406.00403*, 2024.

[46] 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? *Advances in neural information processing systems*, 34:28877–28888, 2021.

[47] Z. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32, 2019.

[48] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec. Hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems*, 31, 2018.

[49] M. Zhang and Y. Chen. Link prediction based on graph neural networks. *Advances in neural information processing systems*, 31, 2018.

[50] Z. Zhang and M. Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018.

A Computing environment and license

All experiments were conducted on Ubuntu 18.04.6 LTS, with an Intel(R) Xeon(R) CPU E5-2698 v4 (40 cores, 2.20GHz) and Tesla V100-DGXS GPUs (32GB). The datasets used in our experiments were accessed via the TUDataset [28] and the LRGB [13] dataset from PyG [14] dataset loader, which is publicly available. We manually fixed the seed for both the CPU and CUDA. For details, refer to the *"seed_everywhere"* function in the provided code, which sets the seeds in Python 3.10.13, NumPy 1.26.4, PyTorch 1.13.1, and CUDA 11.6.1.

B Datasets

Table 1: Dataset specification.

Dataset	# graphs	Avg. nodes	Avg. edges	# features	Avg. diameter	Prediction task	Metric
PTC_MR	344	14.29	14.69	18	7.52	2-graph-class	Accuracy
D&D	1,178	284.32	715.66	82	19.90	2-graph-class	Accuracy
PROTEINS	1,113	39.06	72.82	3	11.57	2-graph-class	Accuracy
IMDB-B	1,000	19.77	96.53	N/A	1.86	2-graph-class	Accuracy
IMDB-M	1,500	13.00	65.94	N/A	1.47	3-graph-class	Accuracy
REDDIT-B	2,000	429.63	497.75	N/A	3.02	2-graph-class	Accuracy
Peptides-struct	15,535	150.9	308.9	9	56.97	11-graph-task	AP
Peptides-func	15,535	150.9	307.3	9	56.97	10-graph-task	MAE
Cora	1	2,708	5,278	1,433	19	7-node-class	Accuracy
CiteSeer	1	3,327	4,522	3,703	28	6-node-class	Accuracy
PubMed	1	19,717	44,324	500	18	3-node-class	Accuracy
Chameleon	1	890	8,854	2,325	23	5-node-class	Accuracy

Small molecules datasets. PTC_MR dataset [35] includes 349 chemical compounds, which reports the carcinogenicity for male rats, 18 distinct node labels, and 2 classes.

Bioinformatics datasets. D&D (Dobson and Doig) dataset [11] comprises 1178 protein structures. Here, each protein is represented by a graph with amino acids as nodes. The classification task is to distinguish between enzymes and non-enzymes protein structures.

PROTEINS dataset [4] consists of 2 graph classes where nodes represent secondary structure elements (SSEs), connected either in the amino-acid sequence or in 3D space.

Social network datasets. IMDB-B dataset [44] is a movie collaboration dataset, where actor, actress and genre information across various movies on IMDB. IMDB-M dataset extends IMDB-B dataset into a multi-class version, containing a balanced set of ego-networks derived from movie genres: Comedy, Romance, and Sci-Fi. REDDIT-B dataset [47, 44] consists of 2,000 graphs, each graph representing an online discussion thread from Reddit. In these graphs, nodes represent users involved in a thread, and edges denote the replies between users, which are categorized based on the type of user interactions within each thread.

Peptides-struct and Peptides-func. The Peptides-struct and Peptides-func [13] datasets are designed to study the structural and functional properties of peptides retrieved from SATPdb. Peptides-struct focuses on predicting 3D structural properties, while Peptides-func aims to predict functional attributes.

Citation network datasets. The Cora dataset [27] consists of scientific publications categorized into seven machine learning topics, where each node represents a paper and edges indicate citation relationships. The node features are derived from a bag-of-words representation of the documents.

The CiteSeer dataset [3] comprises research papers classified into six categories, with citation links forming the graph structure. Each node is described by a sparse binary vector indicating the presence of specific words from a fixed vocabulary.

The PubMed dataset [32] contains biomedical papers classified into three classes, where edges denote citations between them. Node features are based on term frequency-inverse document frequency (TF-IDF) representations of the abstracts.

Wikipedia network datasets. The Chameleon dataset [31] contains web pages from Wikipedia, where each node represents a page and edges denote mutual hyperlinks. It is characterized by low homophily, meaning that connected nodes often belong to different classes, making it a challenging benchmark for GNNs that rely solely on local neighborhood information.

C Hyperparameters

Table 2: Hyperparameters.

	PTC_MR	D&D	PROTEINS	IMDB-B	IMDB-M	REDDIT-B	Peptides-struct	Peptides-func	Cora	CiteSeer	PubMed	Chameleon
Epochs	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
Batch size	128	128	128	128	128	128	128	128	2,708	3,327	19,717	890
Optimizer	Adam	Adam	Adam	Adam	Adam	Adam	Adam	Adam	Adam	Adam	Adam	Adam
Weight decay	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5	1e-5
Dropout rate	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.5	0.5	0.5	0.5
lr	1e-3	1e-3	1e-3	1e-3	1e-3	1e-3	1e-3	1e-3	1e-3	1e-3	1e-3	1e-3
lr decay ratio	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%	1%
lr decay step size	20	20	20	20	20	20	20	20	20	20	20	20
Hidden dimension d_h	128	64	128	128	64	128	128	128	256	256	256	256
Encoder (MLP) layers	2	2	2	2	2	2	2	2	2	2	2	2
ODE T	1.00	2.00	2.00	1.00	2.00	4.00	4.00	2.00	8.00	8.00	2.00	8.00
ODE τ	0.75	0.10	0.10	1.00	0.75	0.25	0.25	0.10	0.10	0.10	0.10	0.25
# Params	190K	71K	137K	220K	73K	60K	189K	188K	1.04M	2.20M	230K	701K
sec/epoch	0.6493	29.749	1.750	1.036	2.021	1.377	4.4206	5.5593	0.1165	0.1204	0.1363	0.0563

Our hyperparameter settings are shown in Table 2. For experiments on the LRGB dataset, where other models set the number of parameters to around ∼500K [21, 30], we opted for a lower parameter count of approximately ∼200K.

D Computational Complexity

The space complexity of our model is primarily dominated by the computation of ODE. For a graph with $|\mathcal{E}|$ edges and hidden dimension d_h, the storage and computation of the graph diffusion operator (e.g., normalized Laplacian $\tilde{L}$) scales as $\mathcal{O}(|\mathcal{E}|d_h))$, assuming a sparse graph.

We now analyze the time complexity of our proposed ODE in Eq. (2). The ODE is solved using the Euler method with T/τ steps, and each step involves evaluating a linear combination of the diffusion term and a contrastive reaction term.

The diffusion term $-\alpha\tilde{L}h(t)$ involves sparse matrix multiplication, with time complexity $\mathcal{O}(|\mathcal{E}|d_h))$ per step. The reaction term $\beta(\tilde{A}-\tilde{A}^2)h(t)$ requires computing a two-hop contrast operator. For sparse $\tilde{A}$, the multiplication $\tilde{A}^2\mathbf{h}(t)$ can be done in $\mathcal{O}(|\mathcal{E}_2|d_{\max})$, where $|\mathcal{E}_2| = \frac{1}{2}\sum_{v\in\mathcal{V}}|\mathcal{N}_2(v)|$ is the total number of two-hop edges and $d_{\max}$ is the maximum degree. Thus, the overall time complexity per integration step is:

$$\mathcal{O}((|\mathcal{E}|+|\mathcal{E}_2|)d_h+|\mathcal{E}_2|d_{\max}). \tag{9}$$

Multiplying by the number of integration steps T/τ, the total time complexity becomes:

$$\mathcal{O}((|\mathcal{E}|d_h+|\mathcal{E}_2|d_h+|\mathcal{E}_2|d_{\max})T/\tau). \tag{10}$$

Our adaptive fusion module performs hierarchical integration of the global embedding $h(0)$ and the local embedding $h(T)$ using a lightweight attention mechanism at both node and graph levels.

At the node level, we compute two cosine similarity scores per node between the node embeddings and a learnable vector $w_\phi \in \mathbb{R}^{d_h}$, followed by a softmax operation:

$$s_g(i) = \cos(h_i(0), w_\phi), \quad s_l(i) = \cos(h_i(T), w_\phi), \quad a_i = \text{softmax}([s_g(i), s_l(i)]) \tag{11}$$

Computing cosine similarity between two d_h-dimensional vectors requires $\mathcal{O}(d_h)$ operations, and this is done for all $n = |\mathcal{V}|$ nodes and both $h_i(0)$ and $h_i(T)$, resulting in $\mathcal{O}(nd_h)$. The weighted combination $\alpha_i h_i(0) + (1-\alpha_i)h_i(T)$ also requires $\mathcal{O}(d_h)$ per node, giving another $\mathcal{O}(nd_h)$. The final node-level fusion cost is thus $\mathcal{O}(nd_h)$.

At the graph level, we perform pooling (e.g., mean or sum) over node embeddings, compute two cosine similarities with a learnable vector $w_\psi \in \mathbb{R}^{d_h}$, and apply a softmax to obtain the fusion weight:

$$\bar{s}_g = \cos(\text{Pool}(h(0)), w_\psi), \quad \bar{s}_l = \cos(\text{Pool}(h(T)), w_\psi) \tag{12}$$

Since each of these involves only one pooled vector, the cost is negligible and constant:

$$\mathcal{O}(d_h) \tag{13}$$

The final fusion representation $\bar{h}$ is computed as a convex combination of pooled vectors, also in $\mathcal{O}(d_h)$.

The final prediction representation is the concatenation of the sum of node-level fused vectors $\sum_i h'_i$ and the graph-level vector $\bar{h}$, both in $\mathbb{R}^{d_h}$, resulting in a $\mathbb{R}^{2d_h}$ representation.

Therefore, the total complexity of the adaptive fusion module is:

$$\mathcal{O}(nd_h) \tag{14}$$

which is linear in the number of nodes and negligible.

E Proof

E.1 Proof of Lemma 1

Proof. Recall the definition of the global embedding loss:

$$\mathcal{L}_{GE} := \frac{1}{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{V}|} \left\| h_i(0) - \int_0^T \alpha \tilde{L} h_i(t)\, dt \right\|^2 . \tag{15}$$

where $h(t) \in \mathbb{R}^{|\mathcal{V}| \times d_h}$ denotes the node representation at time t, and $\tilde{L} := I - \tilde{A}$ is the normalized Laplacian, with $\tilde{A} := D^{-1/2} A_{\text{raw}} D^{-1/2}$.

Let us expand the loss:

$$\mathcal{L}_{GE} = \frac{1}{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{V}|} \left\| h_i(0) - \left(\int_0^T \alpha \left(h_i(t) - \sum_j \tilde{A}_{ij} h_j(t) \right) dt \right) \right\|^2 . \tag{16}$$

This penalizes deviation between $h_i(0)$ and a temporally accumulated diffusion signal, where each node's feature is compared with its smoothed neighborhood. In particular, the integrand $\tilde{L}h(t) = h(t) - \tilde{A}h(t)$ captures the discrepancy between each node's current feature and the aggregated feature of its neighbors. Minimizing $\mathcal{L}_{GE}$ therefore encourages:

$$h_i(0) \approx \int_0^T \alpha \left(h_i(t) - \sum_j \tilde{A}_{ij} h_j(t) \right) dt, \tag{17}$$

which implies:

$$h_i(0) + \int_0^T \alpha \sum_j \tilde{A}_{ij} h_j(t) dt \approx \int_0^T \alpha h_i(t) dt. \tag{18}$$

This means that $h_i(0)$ serves to absorb the difference between the raw node signal and the smoothed neighborhood signal over time. If $h_i(t)$ is relatively smooth over time and space (i.e., neighbors $j \sim i$ have similar representations), then this integral will be small, and $h_i(0)$ must be close to its own local average. More generally, the optimal $h(0)$ under this loss will minimize discrepancy with the diffusion dynamics governed by $\tilde{L}$, which suppresses high-frequency components.

From the spectral perspective, since $\tilde{L} = I - \tilde{A}$ is positive semi-definite and diagonally dominant, it acts as a low-pass filter in the graph Fourier domain. The term $\tilde{L}h(t)$ extracts the high-frequency residuals of $h(t)$, and thus minimizing $\|h(0) - \int_0^T \alpha\tilde{L}h(t)dt\|$ forces $h(0)$ to resemble the smoothed version of $h(t)$, i.e., one where high-frequency differences across neighboring nodes are attenuated.

In summary, minimizing $\mathcal{L}_{GE}$ encourages $h(0)$ to act as a globally consistent initialization that reflects the early-stage behavior of a diffusion process, dominated by low-frequency structure. This aligns with the intuition that global information should precede local specialization.

□

E.2 Proof of Theorem 1

Proof. By Definition 4 and Lemma 1, minimizing $\mathcal{L}_{GE}$ enforces $h(0) \approx \int_0^T \alpha\tilde{L}h(t)dt$. Now consider the full evolution of the representation $h(T)$ governed by the differential equation:

$$\frac{dh}{dt} = -\alpha\tilde{L}h(t) + \beta(\tilde{A} - \tilde{A}^2)h(t), \tag{19}$$

integrated over time:

$$h(T) = h(0) + \int_0^T \left(-\alpha\tilde{L}h(t) + \beta(\tilde{A} - \tilde{A}^2)h(t)\right) dt. \tag{20}$$

Substitute the approximation from Lemma 1:

$$\int_0^T \alpha\tilde{L}h(t)dt \approx h(0) \quad \Rightarrow \quad -\int_0^T \alpha\tilde{L}h(t)dt \approx -h(0). \tag{21}$$

Therefore,

$$h(T) \approx h(0) - h(0) + \int_0^T \beta(\tilde{A} - \tilde{A}^2)h(t)dt = \int_0^T \beta(\tilde{A} - \tilde{A}^2)h(t)dt. \tag{22}$$

This shows that the final representation $h(T)$ is dominated by the reaction term, which captures local structural contrasts through the operator $\tilde{A} - \tilde{A}^2$. The term $\tilde{A}h(t)$ propagates information from immediate neighbors, while $\tilde{A}^2h(t)$ propagates from 2-hop neighbors; their difference emphasizes neighborhood-level variations, making $h(T)$ sensitive to localized patterns.

□

F Related Work

Message passing GNNs. A wide range of Graph Neural Networks (GNNs) rely on the message passing paradigm, where node representations are iteratively updated by aggregating features from their local neighbors. GCN [19] introduces a spectral message passing approach that linearly combines neighboring node features, effectively capturing local structures but struggling to propagate long-range dependencies due to the shallow receptive field. GraphSAGE [16] extends this by introducing learnable aggregation functions (e.g., mean, LSTM), improving inductive capability while still focusing on localized neighborhoods. GAT [37] further enhances the local aggregation process via attention mechanisms, allowing nodes to weigh neighbor contributions dynamically, yet its attention scope remains confined to immediate neighbors. GatedGCN [5] employs edge gating mechanisms in the message passing process to modulate information flow between nodes, introducing nonlinearity and structural adaptability. While all these methods excel at modeling local interactions, their limited depth and local aggregation hinder their ability to capture global graph properties. To address this, hierarchical global pooling mechanisms have been proposed. DiffPool [48] introduces a differentiable graph pooling method that learns soft node cluster assignments, enabling hierarchical abstraction of the graph and capturing coarse-grained global structure. Meanwhile, InfoGraph [34] adopts a contrastive learning framework to learn graph-level representations by maximizing mutual information between node and global graph embeddings.

Transformer-based GNNs. To overcome the limitations of local message passing in traditional GNNs, a growing body of work has adapted the Transformer architecture to graph-structured data. GraphTrans [42] introduces an architecture that combines standard GNN layers for local representation with a permutation-invariant Transformer module for modeling global, long-range dependencies. This two-stage design enables the model to first capture neighborhood-level features and subsequently reason over the entire graph via self-attention. SAN (Spectral Attention Network) [21] utilizes Laplacian eigenvectors as learnable positional encodings (LPE), which encode both absolute and relative structural roles of nodes. By feeding these encodings into a fully-connected Transformer, SAN enables global attention while preserving local topology through spectral structure. Graphormer [46] further improves structural awareness by integrating centrality encoding, shortest-path-based spatial encoding, and edge encoding directly into the attention mechanisms. These components allow Graphormer to model both local connectivity and global distance-aware interactions. GraphGPS [30] proposes a modular framework that decouples local and global processing: a message passing component captures localized interactions while a global attention module enables scalable long-range reasoning. By organizing positional and structural encodings into local, global, and relative categories, GraphGPS systematically balances local bias with global expressivity. GOAT [20] addresses scalability and heterophily challenges by introducing a dual-attention mechanism: a scalable approximation of global attention via dimensionality reduction and a sampled local attention that avoids recursive smoothing. Lastly, Polynormer [10] explores a novel perspective by modeling node interactions through high-degree polynomial functions controlled by attention scores. It constructs separate local and global equivariant attention modules, then fuses them through a linear local-to-global attention mechanism.

G Discussion on the Limitations

While our model enables an interpretable decomposition of node representations into global and local components through the learned attention weights, the interpretation remains at a coarse level. Specifically, although we can quantify the relative contribution of global versus local signals for each node or graph, our method does not provide fine-grained and domain-specific insights such as which specific substructures (e.g., functional groups in molecules, motifs in social networks) or node roles (e.g., hubs, bridges, boundary nodes) are being emphasized by the global or local components. Developing more precise attribution tools or probing techniques to identify these structural drivers remains an important direction for future research.

H Broader impacts

Societal impact. Our method advances the capability of graph neural networks to more accurately capture both global and local patterns in complex graph-structured data. This can enhance performance across a wide range of applications, including drug discovery, molecular property prediction, and social network analysis. In domains such as healthcare and scientific research, improved modeling of relational structures could accelerate innovation and decision-making. Furthermore, by promoting structured and interpretable fusion of hierarchical signals, our model may contribute to more transparent and trustworthy AI systems in various fields.

Possible harms and negative societal impact. We do not foresee any direct societal harms arising from our proposed method. The method is designed as a general-purpose graph representation learning without dependence on sensitive data.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: Yes, the main claims in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Yes, the discussion of the limitations is providied in Appendix G.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

Justification: Yes, we provided the full proof in Appendix E.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: Our experimental results are fully reproducible and provide the details in Appendix.

Guidelines:

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

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

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

Answer: [Yes]

Justification: Yes, all datasets used in our experiments are publicly available. The details are provided in the appendix B.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Yes, we describe the dataset splits in Section 4.1, and provide detailed hyperparameters in Appendix C.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: For experiments involving cross-validation or multiple runs, we report the mean and standard deviation, or the minimum and maximum values.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Yes, we provide the training time in Figure 4 and conducted computational complexity analysis.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: Yes, our paper adheres to the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: Yes, we discuss both potential positive societal impacts and negative societal impacts of our work in H.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [NA]

    Justification: Our model poses no potential risk of misuse.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Yes, we provide the detailed computing environment and license in Appendix A.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [No]

    Justification: No assets are introduced in the paper.

    Guidelines:

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

14. **Crowdsourcing and research with human subjects**

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

    Answer: [NA]

    Justification: Crowdsourcing experiments and research with human subjects are not included in the paper.

    Guidelines:

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

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

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

    Answer: [NA]

    Justification: Crowdsourcing experiments and research with human subjects are not included in the paper.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: We only used LLMs for grammar correction and formatting purposes; they had no influence on our core methodology.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Supplementary Material

Technical appendix. We have included a technical appendix **in the main paper**, covering the computing environment and license, datasets, hyperparameter configuration, computational complexity analysis, proofs, related work, and discussions on the limitations and broader impacts.

Source code & data. We provide our source code in the zip file and the instructions are included in the accompanying README.MD file. We plan to release our code on GITHUB and the URL in the future, making it publicly available. When running the command, all datasets used in our experiments is automatically downloaded, so an internet connection is required.

Typographical corrections. We list minor typographical corrections identified in the main text, which do not affect the results or conclusions. We will address them in the revision for clarity and completeness.

- Page 2, Line 65: “Priliminary” → “Preliminary”.
- Page 4, Line 120: "defined in Theorem 4" → "defined in Definition 3".
- Page 4, Line 138: "(cf. Theorem 4)" → "(cf. Lemma 1)".
- Page 6, Line 232: "Polyformer" → "Polynormer".
- Page 6, Line 239: "GraphGPS (63.55 AP)" → "GraphGPS (65.35 AP)".