Reaction–Diffusion-based Global–Local Graph Fusion Network for Graph Mining

Anonymous submission

Abstract. Graph neural networks often struggle to balance global dependency modeling and preservation of local structural detail. Message-passing models emphasize local neighborhoods but fail to capture long-range interactions, whereas transformer-based graph architectures introduce global attention at the cost of fine-grained distinctions. Inspired by reaction–diffusion dynamics, where diffusion enforces global coherence and reaction amplifies local variation, this paper proposes a global–local graph fusion network that models representation learning as a gradual transition from a globally smooth initialization to locally refined patterns. The proposed method implements this transition through a reaction–diffusion ordinary differential equation on graphs and extracts paired global and local embeddings, which are adaptively fused by an attention-based module. Experiments on six graph classification datasets, two long-range graph benchmarks, and four node classification datasets show that it consistently outperforms strong MPNN and transformer baselines, with gains up to 11.01%. Ablation and analysis studies indicate that both the global–local decomposition and the fusion mechanism are critical to improvements in graph mining performance.

Keywords: Graph Neural Network · Representation Learning · Graph Mining · Neural Ordinary Differential Equations.

1 Introduction

Graph-structured data are ubiquitous in diverse domains such as social networks, molecular analysis, and recommender systems [24,34], making expressive and scalable graph mining methods essential. Graph neural networks (GNNs) have become a dominant framework for learning from such data by propagating information along edges. Message passing neural networks (MPNNs) achieve strong performance through recursive neighborhood aggregation [20,2], but their inherently local nature limits generalization across graphs with varying connectivity patterns [13]. Transformer-based GNNs mitigate this limitation by using global attention mechanisms that enable direct interactions among distant nodes [37,25], but they often blur fine-grained local structural distinctions [10,9]. This tension between global context and local expressiveness remains a central design challenge in graph representation learning.

We draw inspiration from the Turing reaction–diffusion mechanism [17,26] to view this challenge as an interaction between diffusion and reaction processes

on graphs. In biological systems, diffusion enforces spatial uniformity by spreading concentration gradients, while reaction amplifies local differences to form structured patterns. Similarly, graph diffusion promotes global smoothness by aligning the node embeddings with the low-frequency topology [41], while graph reaction encourages discriminative variations that emphasize local differences between adjacent nodes. The interplay between these two processes produces representations that combine global coherence with locally diverse patterns.

Based on this principle, this paper proposes a global–local graph fusion network (GLGFN) that models graph representation learning as a continuous transition from global diffusion to local reaction. The model is instantiated as a reaction–diffusion neural ordinary differential equation (ODE) on graphs, where the initial state $t = 0$ encodes globally diffused information and the terminal state $t = T$ encodes locally refined patterns. These two states are explicitly extracted as paired global and local embeddings.

To integrate these complementary embeddings, GLGFN employs an attention-based fusion mechanism that adaptively balances global smoothness and local expressiveness according to the downstream task and graph structure. We will show the effectiveness of the proposed method with extensive experiments on six graph classification datasets, two long-range graph benchmarks, and four node classification datasets. In addition, additional ablations and analyses will highlight the importance of both the reaction–diffusion decomposition and the adaptive fusion module for graph mining tasks.

2 Related Works

2.1 Message Passing GNNs

Graph neural networks based on message passing update node representations by aggregating features from local neighborhoods. GCN [16] applies spectral filtering to linearly combine neighboring features, while GraphSAGE [14] introduces learnable aggregation functions to support inductive inference. GAT [33] employs attention over neighbors to assign different importances, and GatedGCN [5] incorporates edge gating to modulate information flow. Although these models effectively capture local structures, their reliance on shallow neighborhoods makes it difficult to encode long-range dependencies and global graph properties.

To enrich global information, several methods introduce hierarchical pooling or global objectives. DiffPool [40] learns soft cluster assignments with differentiable pooling, constructing hierarchical graph representations that capture coarse-grained structure. InfoGraph [31] maximizes mutual information between node and graph embeddings in a contrastive framework to learn informative graph-level representations. These methods extend the receptive field beyond immediate neighborhoods, but they do not explicitly disentangle global and local inductive biases.

2.2 Transformer-based GNNs

To overcome the locality of message passing, recent work adapts Transformer architectures to graphs by combining structural encodings with global self-attention. GraphTrans [37] augments standard GNN layers with a permutation-invariant Transformer that models long-range dependencies on top of locally aggregated features. SAN [19] uses Laplacian eigenvectors as learnable positional encodings to inject spectral structure into a fully connected Transformer, and Graphormer [39] integrates centrality, shortest-path, and edge encodings directly into attention computations to improve structural awareness.

More recent models further refine the treatment of global and local signals. GraphGPS [27] combines a message passing backbone with a global attention module and organizes structural encodings into local, global, and relative components. GOAT [18] introduces a dual-attention mechanism with scalable global attention and sampled local attention to handle large and heterophilous graphs. Polynormer [10] models node interactions via polynomial filters controlled by attention and implements separate local and global equivariant attention modules fused by a local-to-global attention scheme. Although these Transformer-based GNNs improve global dependency modeling, they typically treat global and local processing within a single attention stack rather than explicitly modeling and fusing distinct global and local representation stages, which is the focus of this work.

3 Proposed Method

We propose a two-phase architecture in which a diffusion process first encodes a global bias and a subsequent reaction process refines representations based on local contrasts (§3.1). The resulting global and local embeddings are then fused by an attention module (§3.2). This design enforces a functional separation between global and local inductive biases while allowing the model to adaptively weight their contributions per instance.

3.1 Continuous Global-to-Local Modeling

We adopt a continuous-depth framework based on graph neural ordinary differential equation (ODE). Given initial node features X, an encoder ϵ produces

$$h(0) = \epsilon(X; \theta_\epsilon) \in \mathbb{R}^{|\mathcal{V}| \times d_h},$$

and the representation evolves according to

$$\frac{dh(t)}{dt} = f(h(t), t; \theta), \quad h(T) = h(0) + \int_0^T f(h(t), t; \theta)\, dt. \tag{1}$$

This formulation allows gradual transitions in representation space and supports time-dependent dynamics that encode both global and local priors.

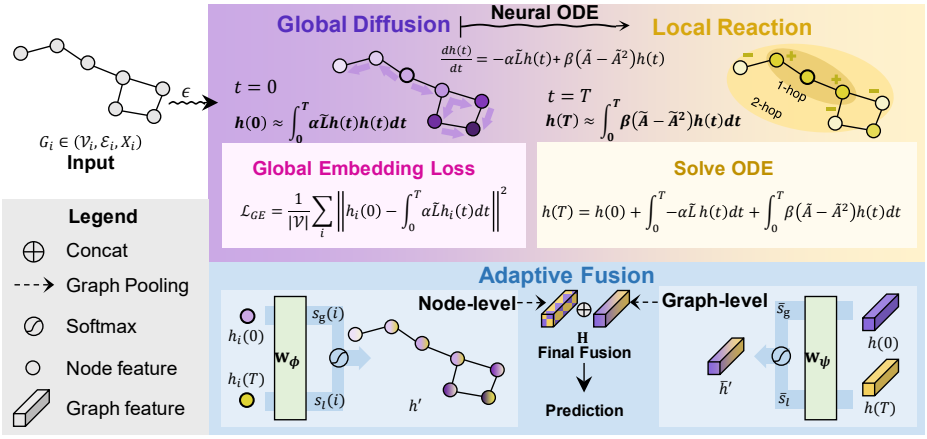


Fig. 1. Illustration of the overall architecture.

We instantiate f as reaction–diffusion dynamics [8]. Given $h(0) = \epsilon(X; \theta_\epsilon)$, the evolution is defined as

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

where $\tilde{L}$ is the normalized Laplacian, $\tilde{A} \in [0,1]^{|\mathcal{V}|\times|\mathcal{V}|}$ is a soft adjacency, and $\alpha, \beta \in \mathbb{R}^{d_h}$ are learnable diffusion and reaction coefficients.

To explicitly encode a globally smoothed initialization, we regularize $h(0)$ to the diffusion-only solution. We introduce the global encoding loss

$$\mathcal{L}_{GE} := \frac{1}{|\mathcal{V}|}\sum_i \left\| h_i(0) - \int_0^T \alpha\tilde{L}h_i(t)\,dt \right\|^2. \tag{3}$$

Minimizing $\mathcal{L}_{GE}$ encourages $h(0)$ to approximate a globally smoothed equilibrium of the diffusion term.

Under this regularization, the terminal state can be approximated as

$$h(T) \approx \int_0^T \beta(\tilde{A} - \tilde{A}^2)h(t)\,dt, \tag{4}$$

so that the reaction term dominates the residual evolution after global diffusion.

The operator $(\tilde{A} - \tilde{A}^2)$ cancels transitive similarity and amplifies contrasts between nodes that are directly connected but poorly supported by common two-hop neighbors, which induces high-frequency (local) signals in $h(T)$. Intuitively, the diffusion term $-\tilde{L}h(t)$ acts as a low-pass filter that aligns the embeddings with the low-frequency topology, while the reaction term $(\tilde{A} - \tilde{A}^2)h(t)$ highlights the boundary regions and local deviations. As a result, $h(0)$ encodes a globally coherent structure and $h(T)$ encodes locally discriminative patterns, producing a global–local pair from a single ODE trajectory.

We discretize the ODE with the Euler method [1,7]:

$$h(t_{k+1}) = h(t_k) + \tau \cdot f(h(t_k), t_k), \tag{5}$$

with step size τ. The embeddings at $t = 0$ and $t = T$ are extracted and used as global and local representations, respectively.

3.2 Adaptive Fusion of Global and Local Representations

We now define an adaptive fusion mechanism for the global embedding $h(0)$ and the local embedding $h(T)$. The relative importance of global and local information is expected to vary across nodes and graphs depending on the downstream task. Adaptive fusion of both sources is therefore essential for constructing expressive and task-relevant embeddings.

At the node level, we compute attention scores that compare each embedding with a learnable task vector $w_\phi \in \mathbb{R}^{d_h}$:

$$\begin{aligned} s_g(i) &= \cos(h_i(0), w_\phi), \quad s_l(i) = \cos(h_i(T), w_\phi), \\ a_i &= \mathrm{softmax}([s_g(i), s_l(i)]), \quad h'_i = \mathrm{Pool}(a_i^{(g)} h_i(0) + a_i^{(l)} h_i(T)), \end{aligned} \tag{6}$$

where $a_i^{(g)}$ and $a_i^{(l)}$ are the global and local components of the 2D vector a_i, and $\mathrm{Pool}(\cdot)$ is a permutation-invariant aggregation over feature channels if needed.

At the graph level, we define an analogous fusion using pooled global and local embeddings:

$$\begin{aligned} \bar{s}_g &= \cos(\mathrm{Pool}(h(0)), w_\psi), \quad \bar{s}_l = \cos(\mathrm{Pool}(h(T)), w_\psi), \\ \bar{a} &= \mathrm{softmax}([\bar{s}_g, \bar{s}_l]), \quad \bar{h} = \bar{a}^{(g)} \mathrm{Pool}(h(0)) + \bar{a}^{(l)} \mathrm{Pool}(h(T)), \end{aligned} \tag{7}$$

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

The final representation is obtained by concatenating the aggregated node-level and graph-level features:

$$\mathbf{H} \in \mathbb{R}^{2d_h} = \left[\sum_i h'_i,\ \bar{h} \right]. \tag{8}$$

This mechanism allows the model to modulate the contribution of global and local components based on their alignment with learned task-specific queries at both node and graph levels.

3.3 Training

The representation $\mathbf{H}$ is passed through fully connected layers to produce predictions $\hat{y} = \mathrm{FC}(\mathbf{H})$. The total loss is

$$\mathcal{L}_{\text{total}} = \lambda_{\text{cls}} \mathcal{L}_{\text{cls}} + \lambda_{GE} \mathcal{L}_{GE}, \tag{9}$$

where $\mathcal{L}_{\text{cls}} = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$ is the cross-entropy loss for the C classes and λ_{cls} and λ_{GE} are the weighting coefficients (we set $\lambda_{\text{cls}} = 10$ in all experiments). Gradients through the ODE solver are computed using the adjoint method [7], which enables end-to-end training with controlled memory cost.

Table 1. Dataset specification.

Dataset	# $\|G\|$	Avg. $\|\mathcal{V}\|$	Avg. $\|\mathcal{E}\|$	# features	Avg. diameter	Prediction	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

4 Experiment

4.1 Setup

Datasets. We evaluate on six graph classification benchmarks: one molecular dataset (PTC_MR [32]), two bioinformatics datasets (D&D [11], PROTEINS [4]), and three social network datasets (IMDB-B, IMDB-M, REDDIT-B [38]). For long-range dependency evaluation, we use Peptides-func and Peptides-struct from the LRGB benchmark [12]. For node classification, we use three homophilous citation networks (Cora [23], CiteSeer [3], PubMed [29]) and the heterophilous Chameleon dataset [28].

Implementation details. We use Adam optimizer with an initial learning rate of 0.001, reduced by 1% every 20 epochs, weight decay 10^{-5}, and two encoding layers in all experiments. For graph classification, we adopt 10-fold cross-validation and report mean accuracy. For LRGB, we use the official train/validation/test splits [27] and report the mean and standard deviation in four random seeds. For node classification, we follow the protocol in [22] with 60/20/20 splits (training/validation/test), except for Chameleon where predefined splits are used, and report the mean and standard deviation over 10 seeds.

Comparison baselines. We compare the proposed method with MPNN-based GNNs and Transformer-based graph models. MPNN baselines include GCN [16], GraphSAGE [14], GAT [33], GatedGCN [5], DiffPool [40], GINE [15], and InfoGraph [31]. Transformer-based baselines include GraphTrans [37], SAN [19], Graphormer [39], GraphGPS [27], NAGphormer [6], NodeFormer [35], Exphormer [30], GOAT [18], SGFormer [36], Polynormer [10], and Gradformer [21]. Baseline results are taken from [21,27,22].

Table 2. 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**

4.2 Model Performance

Graph classification results. Table 2 shows that the proposed method achieves the best performance in all six graph classification datasets, surpassing both the MPNN-based and Transformer-based baselines. The gains are especially pronounced on large graphs such as D&D (83.4%, +3.3% over the second best) and REDDIT-B (90.8%, +1.5%), indicating that global–local modeling is effective for complex graph structure.

Node classification results. Table 3 shows that the proposed method obtains the best accuracy in all four node classification datasets. The improvement is particularly large in Chameleon (57.29%, +11.01% over GCN), suggesting that explicit global–local decomposition is beneficial in both homophilous and heterophilous settings.

Long-range benchmark results. In the LRGB benchmark (Table 4), the proposed method achieves the lowest MAE in Peptides-struct and competitive AP in Peptides-func, close to GraphGPS. These results indicate that the method handles long-range dependencies effectively while preserving local structural information.

4.3 Model Analysis

Ablation studies. Table 5 shows that each component contributes to the final performance. Removing $\mathcal{L}_{GE}$ or $h(T)$ degrades accuracy, which confirms the benefit of explicitly separating diffusion and reaction. Disabling node- or graph-level fusion also reduces performance, indicating that adaptive weighting at both levels is important.

Table 3. 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
Ours	**86.72**±1.11	**75.57**±1.41	**88.99**±0.43	**57.29**±1.72

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

Model	Peptides-struct MAE↓	Peptides-func 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
GraphGPS	0.2500±0.0005	**0.6535±0.0041**
Ours	**0.2498±0.0020**	0.6312±0.0056

Table 5. 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	Acc. ↑	#	Acc. ↑	#	Acc. ↑	#	Acc. ↑	#
✓					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–local analysis. To quantify local structural information, we compute a one-hop local similarity score. Given a graph $G = (\mathcal{V}, \mathcal{E})$ with node embeddings $h \in \mathbb{R}^{|\mathcal{V}| \times d_h}$, the local similarity of a node $v \in \mathcal{V}$ is defined as

$$\mathrm{sim}(v) := \frac{1}{|\mathcal{N}(v)|} \sum_{u \in \mathcal{N}(v)} \cos(h_v, h_u), \tag{10}$$

where $\mathcal{N}(v)$ denotes the one-hop neighbors of v. The corresponding graph-level statistic is the average over nodes with at least one neighbor:

$$\frac{1}{|\mathcal{V}'|} \sum_{v \in \mathcal{V}'} \mathrm{sim}(v), \quad \mathcal{V}' = \{v \in \mathcal{V} \mid |\mathcal{N}(v)| > 0\}. \tag{11}$$

Figure 2 shows that $h(T)$ consistently exhibits higher one-hop similarity than $h(0)$, which is consistent with its interpretation as a cluster-aware, locally smoothed representation. The t-SNE plots in Figure 3 indicate that $h(0)$ and $h(T)$ occupy complementary regions in the embedding space, supporting the global–local disentanglement.

Quantitative analysis of adaptive fusion weights. In Figure 5, we analyze the learned node- and graph-level weights a_i and $\bar{a}$ from Eqs. (6) and (7). The model places more graph-level weight on global features in datasets such

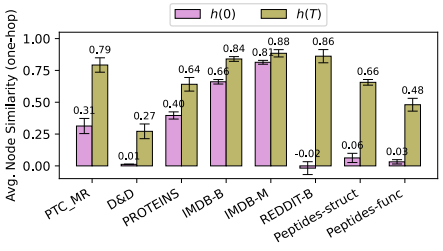
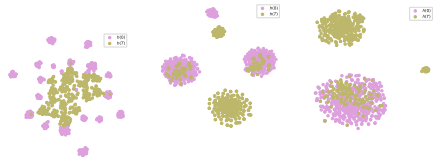


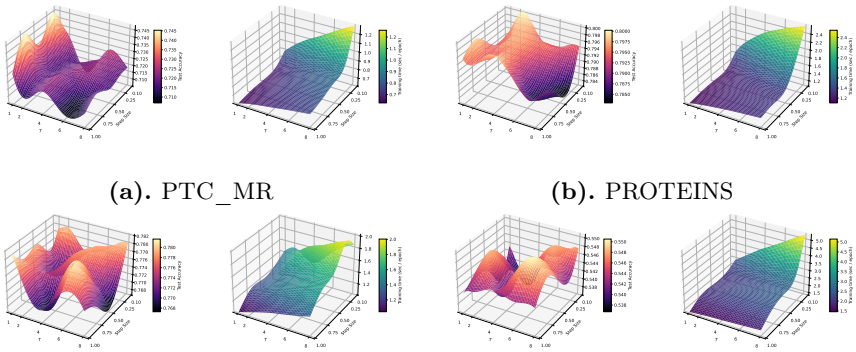
Fig. 2. Average one-hop node similarity for $h(0)$ and $h(T)$ on the test sets. Mean and standard deviation are reported over 10 folds.



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

Fig. 3. t-SNE visualization of node embeddings from $h(0)$ (pink) and $h(T)$ (brown).

as PTC_MR and REDDIT-B, while local features dominate in others. This behavior is consistent with the task-dependent fusion assumption in Section 3.2 and shows that the fusion module adapts to structural characteristics specific to the dataset.

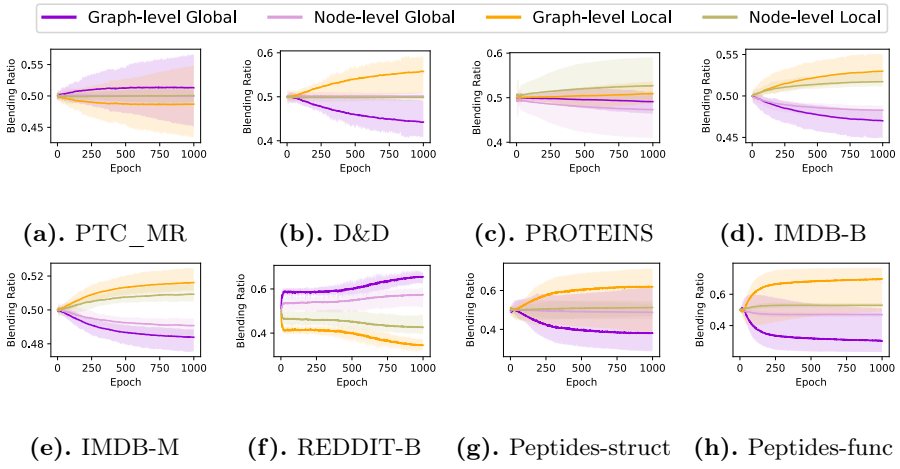


(a). PTC_MR **(b).** PROTEINS

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

Fig. 4. Sensitivity with respect to $T \in [1, 8]$ and step size $\tau \in [0.1, 1]$. Left: test accuracy. Right: training time per epoch.

Sensitivity analysis. Figure 4 shows that the training time increases monotonically with larger depth of ODE T and smaller step size τ, as expected from the increase in the number of function evaluations. In contrast, the accuracy of the test remains relatively stable over a broad range of (T, τ); for example, the accuracy variation in PTC_MR stays within about 3 percentage points. This indicates that the proposed method is robust to the choices of ODE hyperparameters.



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

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

Fig. 5. Global and local contributions over training epochs. The shaded area indicates the range across test samples and the line indicates the mean.

5 Concluding Remarks

We propose a reaction–diffusion-based graph neural network that decomposes representation learning into a diffusion-driven global stage and a reaction-driven local stage, followed by adaptive fusion. The proposed method produces paired global and local embeddings from a single ODE trajectory and learns to weight them according to the task and graph structure. Experiments on graph classification, long-range benchmarks, and node classification, including homophilous and heterophilous graphs, show consistent improvements over strong MPNN and Transformer baselines. Ablation, similarity, fusion-weight, and sensitivity analyses support the interpretation of the proposed method as a robust and controllable global–local modeling framework, although the decomposition into global and local components remains coarse and is currently characterized only by attention weights and similarity statistics. The present analysis does not identify specific substructures or node roles that drive performance, such as functional groups in molecular graphs or motifs in social networks, and extending the proposed method with finer-grained attribution or probing tools to uncover these structural factors is an important direction for future work.

References

1. Atkinson, K., Han, W., Stewart, D.E.: Numerical solution of ordinary differential equations, vol. 81. John Wiley & Sons (2009)
2. Beaini, D., Passaro, S., Létourneau, V., Hamilton, W., Corso, G., Liò, P.: Directional graph networks. In: International Conference on Machine Learning. pp. 748–758. PMLR (2021)

3. Bojchevski, A., Günnemann, S.: Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. arXiv preprint arXiv:1707.03815 (2017)
4. Borgwardt, K.M., Ong, C.S., Schönauer, S., Vishwanathan, S., Smola, A.J., Kriegel, H.P.: Protein function prediction via graph kernels. Bioinformatics **21**(suppl_1), i47–i56 (2005)
5. Bresson, X., Laurent, T.: Residual gated graph convnets. arXiv preprint arXiv:1711.07553 (2017)
6. Chen, J., Gao, K., Li, G., He, K.: Nagphormer: A tokenized graph transformer for node classification in large graphs. arXiv preprint arXiv:2206.04910 (2022)
7. Chen, R.T., Rubanova, Y., Bettencourt, J., Duvenaud, D.K.: Neural ordinary differential equations. Advances in neural information processing systems **31** (2018)
8. Choi, J., Hong, S., Park, N., Cho, S.B.: Gread: Graph neural reaction-diffusion networks. In: International Conference on Machine Learning. pp. 5722–5747. PMLR (2023)
9. Choi, J., Park, S., Park, S., Cho, S.B., Park, N.: Are graph transformers necessary? efficient long-range message passing with fractal nodes in mpnns. arXiv preprint arXiv:2511.13010 (2025)
10. Deng, C., Yue, Z., Zhang, Z.: Polynormer: Polynomial-expressive graph transformer in linear time. arXiv preprint arXiv:2403.01232 (2024)
11. Dobson, P.D., Doig, A.J.: Distinguishing enzyme structures from non-enzymes without alignments. Journal of molecular biology **330**(4), 771–783 (2003)
12. Dwivedi, V.P., Rampášek, L., Galkin, M., Parviz, A., Wolf, G., Luu, A.T., Beaini, D.: Long range graph benchmark. In: Advances in Neural Information Processing Systems. vol. 35, pp. 22326–22340 (2022)
13. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: International conference on machine learning. pp. 1263–1272. PMLR (2017)
14. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. Advances in neural information processing systems **30** (2017)
15. Hu, W., Liu, B., Gomes, J., Zitnik, M., Liang, P., Pande, V., Leskovec, J.: Strategies for pre-training graph neural networks. arXiv preprint arXiv:1905.12265 (2019)
16. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. CoRR **abs/1609.02907** (2016)
17. Kondo, S., Miura, T.: Reaction-diffusion model as a framework for understanding biological pattern formation. science **329**(5999), 1616–1620 (2010)
18. Kong, K., Chen, J., Kirchenbauer, J., Ni, R., Bruss, C.B., Goldstein, T.: Goat: A global transformer on large-scale graphs. In: International Conference on Machine Learning. pp. 17375–17390. PMLR (2023)
19. Kreuzer, D., Beaini, D., Hamilton, W., Létourneau, V., Tossou, P.: Rethinking graph transformers with spectral attention. Advances in Neural Information Processing Systems **34**, 21618–21629 (2021)
20. Kumar, S., Mallik, A., Khetarpal, A., Panda, B.S.: Influence maximization in social networks using graph embedding and graph neural network. Information Sciences **607**, 1617–1636 (2022)
21. Liu, C., Yao, Z., Zhan, Y., Ma, X., Pan, S., Hu, W.: Gradformer: Graph transformer with exponential decay. arXiv preprint arXiv:2404.15729 (2024)
22. Luo, Y., Shi, L., Wu, X.M.: Classic gnns are strong baselines: Reassessing gnns for node classification. In: Advances in Neural Information Processing Systems. vol. 37, pp. 97650–97669 (2024)
23. McCallum, A.K., Nigam, K., Rennie, J., Seymore, K.: Automating the construction of internet portals with machine learning. Information Retrieval **3**, 127–163 (2000)

24. Moon, H.J., Cho, S.B.: Exploring implicit biological heterogeneity in asd diagnosis using a multi-head attention graph neural network. Journal of Integrative Neuroscience **23**(7), 135 (2024)
25. Moon, H.J., Cho, S.B.: Traffic prediction by graph transformer embedded with subgraphs. Expert Systems with Applications **272**, 126799 (2025)
26. Muolo, R., Giambagli, L., Nakao, H., Fanelli, D., Carletti, T.: Turing patterns on discrete topologies: from networks to higher-order structures. In: Proceedings A. vol. 480, p. 20240235. The Royal Society (2024)
27. Rampášek, L., Galkin, M., Dwivedi, V.P., Luu, A.T., Wolf, G., Beaini, D.: Recipe for a general, powerful, scalable graph transformer. Advances in Neural Information Processing Systems **35**, 14501–14515 (2022)
28. Rozemberczki, B., Allen, C., Sarkar, R.: Multi-scale attributed node embedding. Journal of Complex Networks **9**(2), cnab014 (2021)
29. Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., Eliassi-Rad, T.: Collective classification in network data. AI magazine **29**(3), 93–93 (2008)
30. Shirzad, H., Velingker, A., Venkatachalam, B., Sutherland, D.J., Sinop, A.K.: Exphormer: Sparse transformers for graphs. In: International Conference on Machine Learning. pp. 31613–31632. PMLR (2023)
31. Sun, F.Y., Hoffman, J., Verma, V., Tang, J.: Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In: International Conference on Learning Representations (2019)
32. Toivonen, H., Srinivasan, A., King, R.D., Kramer, S., Helma, C.: Statistical evaluation of the predictive toxicology challenge 2000–2001. Bioinformatics **19**(10), 1183–1193 (2003)
33. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
34. Waikhom, L., Patgiri, R.: A survey of graph neural networks in various learning paradigms: methods, applications, and challenges. Artificial Intelligence Review **56**(7), 6295–6364 (2023)
35. Wu, Q., Zhao, W., Li, Z., Wipf, D.P., Yan, J.: Nodeformer: A scalable graph structure learning transformer for node classification. Advances in Neural Information Processing Systems **35**, 27387–27401 (2022)
36. Wu, Q., Zhao, W., Yang, C., Zhang, H., Nie, F., Jiang, H., Bian, Y., Yan, J.: Sgformer: Simplifying and empowering transformers for large-graph representations. Advances in Neural Information Processing Systems **36**, 64753–64773 (2023)
37. Wu, Z., Jain, P., Wright, M., Mirhoseini, A., Gonzalez, J.E., Stoica, I.: Representing long-range context for graph neural networks with global attention. Advances in neural information processing systems **34**, 13266–13279 (2021)
38. Yanardag, P., Vishwanathan, S.: Deep graph kernels. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining. pp. 1365–1374 (2015)
39. Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y., Liu, T.Y.: Do transformers really perform badly for graph representation? Advances in neural information processing systems **34**, 28877–28888 (2021)
40. Ying, Z., You, J., Morris, C., Ren, X., Hamilton, W., Leskovec, J.: Hierarchical graph representation learning with differentiable pooling. Advances in neural information processing systems **31** (2018)
41. Zhao, J., Dong, Y., Ding, M., Kharlamov, E., Tang, J.: Adaptive diffusion in graph neural networks. Advances in neural information processing systems **34**, 23321–23333 (2021)