Local-Global Blending Graph Neural ODE Network for Graph Classification

Minhyuk An[1,2] , **Sung-Bae Cho**[1,3]
[1]Yonsei University
[2]Department of Artificial Intelligence
[3]Department of Computer Science
{als7928, sbcho}@yonsei.ac.kr

Abstract

Effectively integrating both local and global information is crucial for graph classification, as it ensures that graph neural networks (GNNs) capture fine-grained local structures while leveraging the global topology. However, traditional GNNs primarily excel at extracting local patterns but often failing to capture the graph's global properties. Recent reaction-diffusion based methods appear well-suited for local-global integration but lack of stabilizing and balancing these features. In this work, we propose Blend-GNN, a novel model, designed to selectively and adaptively integrate both global and local patterns. We employ a reaction-diffusion process specifically designed to extract stable local and global patterns, enabling the model to effectively capture both node-level and graph-level aspects. These two representations are adaptively aggregated to achieve a balanced representation, effectively excluding redundant information by focusing on patterns most relevant to classification. Furthermore, the model offers interpretable insights into how global and local patterns synergistically contribute to graph classification, enhancing its explainability. Extensive experiments on 8 benchmark datasets demonstrate that our Blend-GNN outperforms the state-of-the-art models, achieving significant improvements up to 3.8%p over baselines. Additionally, further analyses validate the effectiveness of our approach, providing a deeper understanding of the local-global interplay.

1 Introduction

Graph neural networks (GNNs) have emerged as a powerful tool for handling graph-structured data by leveraging their ability to learn representations that capture both node features and graph topology [Cai *et al.*, 2018; Wu *et al.*, 2020; Beaini *et al.*, 2021; Veličković *et al.*, 2018]. Graph classification, a critical task in graph-based learning, involves predicting labels for entire graphs by leveraging their structural and attribute information. The ability to effectively integrate local patterns, derived from node neighborhoods, with global structures, reflecting the graph's overarching topology, is essential for building robust graph classification models [Veličković *et al.*, 2018]. Striking this balance is challenging, as most existing methods struggle to capture long-range dependencies while preserving critical local distinctions [Wei *et al.*, 2023].

Traditional graph neural networks (GNNs), such as graph convolutional networks (GCN) [Kipf and Welling, 2016] and graph attention networks (GAT) [Veličković *et al.*, 2018], rely on neighborhood aggregation mechanisms, where node features are updated by aggregating information from immediate neighbors. While this approach captures local patterns effectively, it often fails to propagate information over long distances, making it difficult to learn meaningful global representations [Yu *et al.*, 2024]. Deepening GNN layers to address this issue frequently leads to over-smoothing, where node representations become indistinguishable [Rusch *et al.*, 2023].

Recent works have sought to address these issues by introducing techniques that extend beyond traditional message-passing. Hierarchical pooling-based GNNs, such as DiffPool [Ying *et al.*, 2018], coarsen graphs into multi-scale representations by learning soft cluster assignments. It allows models to capture both local and global structures, enhancing their expressiveness for graph-level tasks. However, the reliance on dense cluster assignment matrices introduces significant computational overhead, making these methods less practical for large-scale graphs. Transformer-based GNNs, such as GraphTransformer [Dwivedi and Bresson, 2020], Graphormer [Ying *et al.*, 2021; Shi *et al.*, 2022], SAN [Kreuzer *et al.*, 2021], and GraphGPS [Rampášek *et al.*, 2022], leverage the global receptive field of Transformer architectures by incorporating local information. This allows the model to learn both local and global patterns, but the requirement for dataset dependant encodings and memory budget can limit their generalizability across diverse graph domains [Rampášek *et al.*, 2022]. Contrastive learning-based methods such as GraphCL [You *et al.*, 2020], AutoGCL [Yin *et al.*, 2022] and InfoGraph [Sun *et al.*, 2019] generate augmented graph-level and node-level representations and maximize mutual information between different levels of representations (e.g., nodes, edges, and subgraphs). While these methods align local and global representations [You *et al.*, 2020], their dependence on contrastive learning introduces

large and carefully curated datasets for effective training, as its performance depends heavily on generating meaningful positive and negative pairs. Concurrently, neural ODE-based GNNs such as GRAND [Chamberlain *et al.*, 2021] have emerged. GRAND applies the diffusion process with deep layers to graphs, effectively preventing oversmoothing and enabling the learning of robust global representations [Wei *et al.*, 2023]. GREAD [Chamberlain *et al.*, 2021; Choi *et al.*, 2023] improves upon GREAD by integrating diffusion process with reaction process which leads local patterns. By leveraging both local and global, it achieves strong performance on diverse graph types. Despite their promise, it lacks systematic mechanisms to stabilize the interplay between local and global features, which can lead to inefficiencies and suboptimal representations.

Building on these insights, we propose Blend-GNN[1], a novel GNN that adaptively integrates local and global information using a principled reaction-diffusion mechanism. Blend-GNN addresses the limitations of existing methods, including over-smoothing, long range dependencies [Dwivedi *et al.*, 2022c], and the lack of dynamic adaptivity in balancing local-global features. To ensure stability of reaction-driven representation, we propose a regularization term that minimizes the discrepancy between the initial encoding and its diffusion-transformed state. Afterward, the two representations from each diffusion and reaction are adaptively fused through a learnable method dynamically balances the contributions of global representation and local representation at both node and graph levels, ensuring task-specific adaptability, namely, blending.

We evaluate our model on eight benchmark datasets with diverse domains and characteristics, and compare its performance against twelve baseline methods. The experimental results verify the effectiveness of our model by achieving the state-of-the-art performance, and highlight the importance of integrating local and global information at both node and graph levels in GNNs.

The key contributions of this work are as follows:

- We propose a novel model Blend-GNN that blends global information from diffusion processes and local information derived from reaction processes at both the node-level and graph-level.
- We empirically demonstrate significant performance gains across eight benchmark datasets, outperforming state-of-the-art baselines, while also showcasing how the integration of local and global information in our Blend-GNN contributes to graph classification at both the node and graph levels.
- It lays the groundwork for further exploration of local-global multi-level techniques in GNNs, with the potential to be applied in a wide range of real-world applications.

[1]Our source code will be available at https://example.com.

2 Preliminaries

2.1 Graph Classification

Given a graph $G = (V, E)$ where V and E are set of nodes and edges with node features $X \in \mathbb{R}^{|V| \times d_X}$, we aim to learn a classification model which predicts target class label y from the source G.

Graph classification often rely on mechanisms that model the flow and transformation of information across the graph structure. Among these mechanisms, diffusion and reaction processes play a fundamental role in capturing both smooth global patterns and distinct local features within graphs.

2.2 Encoding Layer

We use the following L-layer encoder network $\epsilon : \mathbb{R}^{d_X} \to \mathbb{R}^{d_h}$ to map the features of V to an initial encoding $h(0) \in \mathbb{R}^{|V| \times d_h}$, which serves as the starting point for further processing in neural networks, defined as:

$$\epsilon(x, \theta_\epsilon^{(1:L)}) = \theta^{(L)}\sigma(\theta_\epsilon^{(L-1)}\sigma(\dots\theta_\epsilon^{(2)}\sigma(\theta_\epsilon^{(1)}x))), \quad (1)$$

$$h(0) = \epsilon(X, \theta_\epsilon), \quad (2)$$

where $\theta_\epsilon^{(l)}$ represents the parameters of the l-th layer for $1 \leq l \leq L$ and σ is a non-linear activation function.

2.3 Diffusion on Graphs

Neural diffusion on a graph, introduced by [Chamberlain *et al.*, 2021], refers to the process by which information propagates between connected nodes, leading to a smoothing effect on node features. This process is mathematically modeled using the graph Laplacian, which encapsulates the connectivity structure of the graph. The diffusion process is governed by the differential equation:

$$\frac{df(t)}{dt} := -\alpha\tilde{L}h(t), \quad (3)$$

where f is a neural network parameterized by the parameters of the model, $h(t) \in \mathbb{R}^{|V| \times d_h}$ is the node encodings at time t, $\tilde{L}$ is the normalized graph Laplacian defined as $\tilde{L} = I - D^{-1/2}AD^{-1/2}$, and $\alpha \in \mathbb{R}^{d_h}$ is a learnable parameter controlling the rate of diffusion. Here, A represents the adjacency matrix of the graph, D is the degree matrix, and I is the identity matrix. This reduces disparities between connected nodes, effectively smoothing the features over the graph acting as a low-pass filter, propagating information in a way that diminishes high-frequency variations. This ensures that nodes with strong connections converge toward similar feature representations.

2.4 Reaction on Graphs

In contrast to diffusion, reaction processes emphasize feature amplification and differentiation between nodes. These processes often model non-linear interactions that highlight local structural properties of the graph [Choi *et al.*, 2023]. On a graph, reaction terms are frequently expressed as:

$$\tilde{A}_{ij} := \text{softmax}(\frac{(W_{\text{K}}h_i(0))^T(W_{\text{Q}}h_j(0))}{\sqrt{d_K}}), \quad (4)$$

$$\frac{df(t)}{dt} := \beta(\tilde{A} - \tilde{A}^2)h(t), \tag{5}$$

where $\tilde{A}_{i,j}$ is the (i,j)-th element of $\tilde{A}$, called attention weight matrix composed of learnable parameters that weigh the degree of information exchange between nodes [Chamberlain *et al.*, 2021; Choi *et al.*, 2023], $W_{\mathrm{K}} \in \mathbb{R}^{d_m}$ and $W_{\mathrm{Q}} \in \mathbb{R}^{d_m}$ are learnable parameters, d_K is the scale factor, and $\tilde{A}^2$ accounts for higher-order interactions between neighboring nodes. The parameter $\beta \in \mathbb{R}^{d_h}$ controls the intensity of the reaction process.

Unlike diffusion, which smooths node encodings, reaction processes focus on enhancing differences and separating node encodings based on their structural roles within the graph. It has an sharpening effect which allows the model to capture localized patterns, such as clusters or boundary distinctions [Choi *et al.*, 2023].

3 Proposed Method

3.1 Global Initial Encoding

To achieve a unified local-global representation on graph-structured data, the combination of two terms is a choice for tasks that require learning a nuanced balance of global and local properties. The governing equation and the solution is:

$$\frac{df(t)}{dt} = -\alpha\tilde{L}h(t) + \beta(\tilde{A} - \tilde{A}^2)h(t), \tag{6}$$

$$h(T) = h(0) + \int_0^T \left(-\alpha\tilde{L}h(t) + \beta(\tilde{A} - \tilde{A}^2)h(t)\right) dt. \tag{7}$$

It is solved by the Euler method [Atkinson *et al.*, 2009] is a first-order numerical solver for ODEs with a given initial value. It is defined as:

$$h(t_{k+1}) = h(t_k) + \tau \cdot f(h(t_k), t_k, \theta_{rd}) \tag{8}$$

where $h(t_k)$ is hidden representation at time t_k, τ is the step size, which plays a crucial role in the accuracy of the Euler method (i.e., a smaller τ leads to a more accurate $h(T)$ but requires more computational resources, while a larger τ reduces computational overhead with the sacrifice of accuracy). By solving Eq. 7, we can extract both local and global information. However, its success critically depends on the state of the initial encoding $h(0)$. If $h(0)$ is poorly arbitrary, it can result in instability or suboptimal representations. To achieve more stable and effective results, we suggest the initital encoding $h(0)$ satisfies:

$$h(0) + \int_0^T -\alpha\tilde{L}h(t)dt = 0, \tag{9}$$

which indicates that $h(0)$ is perfectly balanced with its diffusion-transformed state. In this condition, the diffusion term ensures that $h(0)$ is smoothed consistently with the graph's global topology, serving as a stable foundation for the reaction term. This stability prevents noise amplification or over-sharpening during the reaction process, leading to more reliable and interpretable local-global representations. By minimizing the mean squared error (MSE) between $h(0)$ and its diffusion-transformed state, the model explicitly trains $h(0)$ to approximate the graph's global properties. What we refer to as global encoding is defined as follows:

$$\mathcal{L}_{\mathrm{GE}} = \frac{1}{|V|}\sum_{i=1}^{|V|} \|h_i(0) + \int_0^T -\alpha\tilde{L}h_i(t)dt\|^2. \tag{10}$$

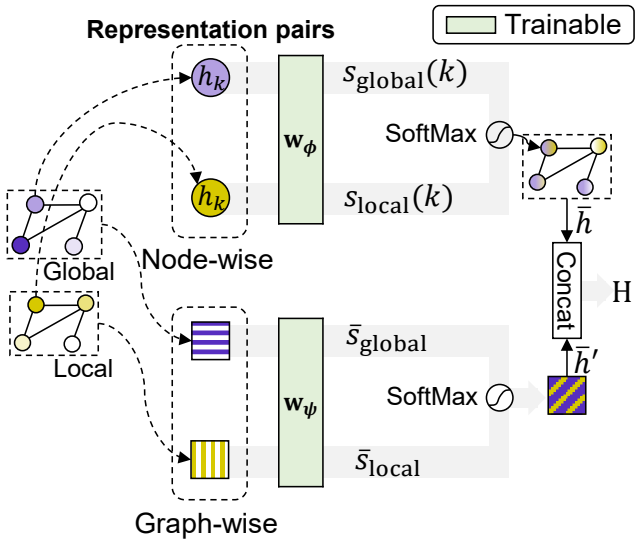


Figure 1: Overview of the local-global adaptive blending. This performs node-level and graph-level blending based on the two representation pairs, global from diffusion process and local from reaction process, to obtain the final representation **H** for prediction.

3.2 Global-to-Local Representation via Reaction

By initializing $h(0)$ with a diffusion-dominated global representation, the need for further diffusion diminishes, allowing the reaction term to primarily govern the encoding evolution. This effectively simplifies the computation of h(T) as:

$$h(T) = \int_0^T \beta(\tilde{A} - \tilde{A}^2)h(t)dt, \tag{11}$$

where $h(0)$ is absorbed into the initial conditions of the reaction dynamics. In this reformulation, $h(T)$ emerges purely from the reaction term, driven by the global coherence already embedded in $h(0)$.

3.3 Local-Global Adaptive Blending

The blending operates at both the node-level and graph-level to selectively combine global and local information encoded in $h(0) \in \mathbb{R}^{|V|\times d_h}$ and $h(T) \in \mathbb{R}^{|V|\times d_h}$. The overview of the method is shown in Figure 1. They are representations derived from the initial node features (as global information) after L layers of encoder ϵ and the final node features (as local information) $h(T)$ after T time of neural reaction-diffusion process. For each node v_i, the node-level scores $s_{\mathrm{global}}(i)$ and $s_{\mathrm{local}}(i)$ corresponding to the global vector $h(0)$ and the local

vector $h(T)$ are derived as follows:

$$s_{\text{global}}(i) = \frac{\langle h_i(0), \mathbf{w}_\phi \rangle}{||h_i(0)|| \cdot ||\mathbf{w}_\phi||}, \tag{12}$$

$$s_{\text{local}}(i) = \frac{\langle h_i(T), \mathbf{w}_\phi \rangle}{||h_i(T)|| \cdot ||\mathbf{w}_\phi||}, \tag{13}$$

where $h_i(0)$ and $h_i(T)$ are the i-th rows of $h(0)$ and $h(T)$, representing the feature vectors of node v_i, $\mathbf{w}_\phi \in \mathbb{R}^{d_h}$ is a learnable vector, and $\langle \cdot \rangle$ is dot-product operator. Next, we obtain the blending weights for each node:

$$\alpha_{\text{global}}(i) = \frac{\exp(s_{\text{global}}(i))}{\exp(s_{\text{global}}(i)) + \exp(s_{\text{local}}(i))}, \tag{14}$$

$$\alpha_{\text{local}}(i) = \frac{\exp(s_{\text{local}}(i))}{\exp(s_{\text{global}}(i)) + \exp(s_{\text{local}}(i))} \tag{15}$$

where $\alpha_{\text{global}}(i)$ and $\alpha_{\text{local}}(i)$ are the softmax-normalized weights for the global and local representations, respectively, for node v_i. The combined representation h'_i for node v_i is then computed as a weighted sum of the global and local representations:

$$h'_i = \alpha_{\text{global}}(i) s_{\text{global}}(i) + \alpha_{\text{local}}(i) s_{\text{local}}(i). \tag{16}$$

This equation ensures that the final representation h'_i for each node v_i is a blend of the global and local information.

The above derivation primarily deals with the blending at the node level. However, the mechanism can be generalized to perform graph-level blending by considering global and local representations aggregated over the whole graph. Assume that $\bar{h}(0) = \sum_{i=1}^{|V|} h_i(0)$ and $\bar{h}(T) = \sum_{i=1}^{|V|} h_i(T)$ represent the aggregated global and local representations of the G, respectively (e.g., sum-pooling, mean-pooling, and max-pooling). The global-level scores $\bar{s}_{\text{global}}$ and $\bar{s}_{\text{local}}$ are derived similarly:

$$\bar{s}_{\text{global}} = \frac{\langle \bar{h}(0), \mathbf{w}_\psi \rangle}{||\bar{h}(0)|| \cdot ||\mathbf{w}_\psi||}, \tag{17}$$

$$\bar{s}_{\text{local}} = \frac{\langle \bar{h}(T), \mathbf{w}_\psi \rangle}{||\bar{h}(T)|| \cdot ||\mathbf{w}_\psi||}, \tag{18}$$

where $\mathbf{w}_\psi \in \mathbb{R}^{d_h}$ is a learnable vector. These scores are normalized with the softmax function:

$$\bar{\alpha}_{\text{global}} = \frac{\exp(\bar{s}_{\text{global}})}{\exp(\bar{s}_{\text{global}} + \exp(\bar{s}_{\text{local}})}, \tag{19}$$

$$\bar{\alpha}_{\text{local}} = \frac{\exp(\bar{s}_{\text{local}})}{\exp(\bar{s}_{\text{global}} + \exp(\bar{s}_{\text{local}})}. \tag{20}$$

Finally, the graph-level representation $\bar{h}'$ and the final representation $\mathbf{H}$ are derived as:

$$\bar{h}' = \bar{\alpha}_{\text{global}} \bar{s}_{\text{global}} + \bar{\alpha}_{\text{local}} \bar{s}_{\text{local}}. \tag{21}$$

$$\mathbf{H} = [\sum_{i=1}^{|V|} h'_i, \bar{h}'], \tag{22}$$

where $\mathbf{H} \in \mathbb{R}^{|V| \times 2d_h}$ is the concatenation of the results from node-level blending and graph-level blending, where the contribution of each part is determined by the learned vector $\mathbf{w}_\phi$ and $\mathbf{w}_\psi$.

3.4 Training Objective

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. The training objective is as follows:

$$\mathcal{L}_{\text{cls}} = -\sum_{i=1}^{C} y_i \log(\hat{y}_i), \qquad \text{for } C \text{ classes}, \tag{23}$$

$$\mathcal{L}_{\text{total}} = \lambda_{\text{cls}} \mathcal{L}_{\text{cls}} + \lambda_{\text{GE}} \mathcal{L}_{\text{GE}}, \tag{24}$$

where $\mathcal{L}_{\text{cls}}$ is the cross-entropy loss [Zhang and Sabuncu, 2018]. We set λ_{cls} to 10 in all experiments. On the other hand, we optimize f using the adjoint method [Chen *et al.*, 2018] by solving the ODE backward in $T \to 0$, which efficiently computes gradients. The adjoint method provides a memory efficient, scalable, and accurate way to compute gradients.

4 Experiments

4.1 Datasets

Graph classification datasets. To ensure diversity, we test on a diverse set of real-world graph classification datasets, including 1 small molecules dataset PTC_MR, 2 bioinformatics datasets (D&D, PROTEINS) and 3 social networks datasets (IMDB-B, IMDB-M, REDDIT-B). These are collected from the TUDataset [Morris *et al.*, 2020], a collection of graph classification benchmark datasets.

Long-range graph benchmark. We also conducted experiments on the long-range graph benchmark (LRGB) [Dwivedi *et al.*, 2022c], 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 accross the entire graph.

We summarize the dataset details in Table 1 and further details are reported in Appendix. To further investigate the structural properties of the dataset, we analyze the average graph diameter, which provides insights into the long-range connectivity of the graphs.

Table 1: Dataset specification.

Dataset	# $\|G\|$	Avg. # $\|V\|$	Avg. # $\|E\|$	d_X	Avg. diameter	Prediction task	Metric
PTC_MR	344	14.29	14.69	18	7.52	2-class	Accuracy
D&D	1,178	284.32	715.66	82	19.90	2-class	Accuracy
PROTEINS	1,113	39.06	72.82	3	11.57	2-class	Accuracy
IMDB-B	1,000	19.77	96.53	–	1.86	2-class	Accuracy
IMDB-M	1,500	13.00	65.94	–	1.47	3-class	Accuracy
REDDIT-B	2,000	429.63	497.75	–	9.72	2-class	Accuracy
Peptides-struct	15,535	150.9	307.3	9	56.97	11-task	AP
Peptides-func	15,535	150.9	307.3	9	56.97	10-task	MAE

4.2 Implementation Details

For a fair comparison, 10-fold cross-validation [Wong and Yeh, 2019] is used to report the mean and standard deviation for each experiment. In all experiments, we use

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	0.623±0.057	0.791±0.031	0.759±0.028	0.733±0.053	0.512±0.051	0.893±0.033
GraphSAGE	–	0.658±0.049	0.659±0.027	0.724±0.036	0.499±0.050	0.843±0.019
GAT	–	–	0.747±0.022	0.758±0.023	0.478±0.031	–
GIN-0	0.646±0.070	0.776±0.050	0.762±0.028	0.751±0.034	0.523±0.028	**0.924±0.025**
DiffPool	0.634±0.010	0.769±0.044	0.752±0.040	0.701±0.063	0.472±0.018	0.891±0.016
GraphCL	–	0.786±0.004	0.744±0.005	0.711±0.004	–	0.895±0.008
InfoGraph	0.617±0.014	0.729±0.018	0.744±0.003	0.730±0.009	0.367±0.008	0.825±0.014
AutoGCL	–	0.776±0.006	0.758±0.036	0.733±0.004	–	0.886±0.015
Graphormer	0.714±0.052	–	0.763±0.027	0.703±0.009	0.489±0.020	–
GREAD	0.718±0.069	0.790±0.026	0.787±0.035	0.771±0.043	0.539±0.036	0.854±0.031
Blend-GNN (Ours)	**0.724±0.064**	**0.829±0.025**	**0.800±0.037**	**0.788±0.033**	**0.548±0.026**	0.908±0.020

Adam [Kingma and Ba, 2017] as the optimizer and train the model for 1000 epochs in each fold. We fixed the batch size to 128. 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 L is set to 2. For the D&D and REDDIT-B, the hidden dimension d_h is set to 64, while 128 for the remaining experiments. Further details can be found in Appendix.

4.3 Comparison Baselines

For evaluating the effectiveness of Blend-GNN on graph classification benchmarks, we compare against a diverse set of baseline models. These are selected to represent a wide range of approaches, including traditional message-passing methods (GCN, GraphSAGE, GAT, GIN-0), hierarchical pooling strategies (DiffPool), self-supervised and contrastive learning frameworks (GraphCL, InfoGraph, AutoGCL), and advanced attention-based Graphormer and reaction-diffusion-based GREAD. Together, these baselines provide a comprehensive benchmark against state-of-the-art techniques for capturing local and global patterns in graphs.

For LRGB, we adopt GINE [Hu *et al.*, 2019], GatedGCN [Bresson and Laurent, 2017], GraphTransformer, SAN, and GraphGPS as baselines. These methods are chosen because they are explicitly designed to address long-range dependencies, a critical challenge in many graph-based tasks. They incorporate Laplacian positional encoding (LapPE) [Dwivedi *et al.*, 2022a] and random walk structural encoding (RWSE) and global aggregation strategies [Dwivedi and Bresson, 2020; Kreuzer *et al.*, 2021; Beaini *et al.*, 2021; Wang *et al.*, 2022; Lim *et al.*, 2022; Rampášek *et al.*, 2022; Dwivedi *et al.*, 2022b].

4.4 Comparative Performance

Graph classification results. Our Blend-GNN achieves the best performance on five out of six datasets and competitive results on the remaining dataset REDDIT-B, as shown in Table 2. It shows the generalizability of our approach across diverse graph domains, including small molecules, bioinformatics, social networks. On the D&D dataset, we observe a performance improvement of up to 3.8%p compared to the baseline. This indicates that Blend-GNN achieves strong results even on datasets with a high average number of edges. Although it does not achieve the best performance on the REDDIT-B, it shows comparable results to the baseline, showing its effectiveness even on datasets with a large average number of nodes. Notably, Blend-GNN achieves outstanding performance even on PTC_MR and IMDB-M, which have a relatively small average number of nodes and edges.

LRGB results. Table 3 shows that Blend-GNN is comparable to or even superior to Transformer-based GNNs such as GraphTransformer, SAN, and GraphGPS, which leverage a global receptive field of Transformer [Dwivedi and Bresson, 2020; Parmar *et al.*, 2018; Wei *et al.*, 2023] to capture long-range dependencies by explicitly encoding the local positional information of nodes. It highlights our model's ability to handle not only local but also global graph-level structural information. On the Peptides-func task, while Transformer-based methods achieve higher average precision (AP) of 0.6535, Blend-GNN remains highly competitive with an AP of 0.6312. This further shows that Blend-GNN effectively captures global dependencies and delivers robust performance across diverse tasks.

From the results, we observe that Blend-GNN performs well not only on datasets with low diameters, such as IMDB-B and IMDB-M, but also on datasets with high diameters, such as LRGB (see Table 1). This demonstrates the effectiveness of our model across diverse domains and a wide range of graph structures.

4.5 Ablation Studies

Table 4 presents the results of an ablation study conducted on PROTEINS 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 first row, the full method achieves the highest accuracy. The role of global encoding is evident when $\lambda_{\text{GE}} = 0$, leading to a drop in accuracy (See the first, second, fifth and sixth row, which is GREAD). It demonstrates the importance of global encoding in enhancing the local representation. We further observe deeper insights into the influence of global

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
GraphTransformer+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**
GREAD	0.2683±0.0021	0.6186±0.0049
Blend-GNN (Ours)	**0.2498±0.0020**	0.6312±0.0056

Table 4: Ablation studies on PROTEINS.

Diffusion	Global encoding	Reaction	Blending-level		Accuracy↑
			Node	Graph	
✓	✓	✓	✓	✓	0.800±0.037
✓	–	✓	✓	✓	0.794±0.037
✓	✓	✓	–	✓	0.797±0.047
✓	✓	✓	✓	–	0.793±0.047
✓	✓	✓	–	–	0.793±0.050
✓	–	✓	–	–	0.787±0.035
✓	✓	–	–	–	0.791±0.047
✓	–	–	–	–	0.791±0.039

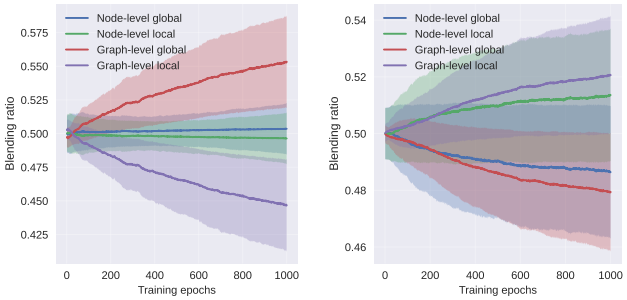
encoding from the fifth to eight row. While global encoding shows no effect when used with diffusion alone, a synergistic effect can be observed when combined with reaction. This is evidenced by the improved performance compared to the case without global encoding.

We also observe that our blending at the node and graph levels play a crucial role in the model's performance. Disabling node-level blending (in the third row) results in a slight drop in accuracy. Similarly, removing graph-level blending (in the fourth) row) reduces the accuracy, reflecting its importance.

4.6 Analysis of Blending Ratios

Figure 2 shows the observed trends of blending ratios for global and local at both the node and graph levels over 1000 training epochs for the PTC_MR and IMDB-B. It indicates the relative contributions of global and local features to the model's predictions. Specifically, the decreasing ratio signifies the exclusion of redundant information by focusing on patterns most relevant to classification. The shaded regions represent the variance across graphs and the batch of nodes.

For the PTC_MR, shown in Figure 2a, the ratios reveal distinct trends at both node and graph levels, with a clear divergence in the importance of global and local information as training progresses. The ratio at the node-level remains stable around 50%p throughout training, indicating that both local and global representations are important for node-level tasks in the PTC_MR. At the graph level, the ratios exhibit a contrasting dynamic. The contribution of global information increases substantially, with the ratio rising from 50%p to approximately 55%p by the final epoch. It highlights the growing importance of high-level structural patterns for graph-level predictions. Conversely, the ratio for local graph-level information steadily decreases, dropping to around 45%p. It suggests that local information at the graph level become less critical as the model increasingly leverages global representations.



(a) PTC_MR (b) IMDB-B

Figure 2: The observed blending ratio of global and local information according to the training epochs.

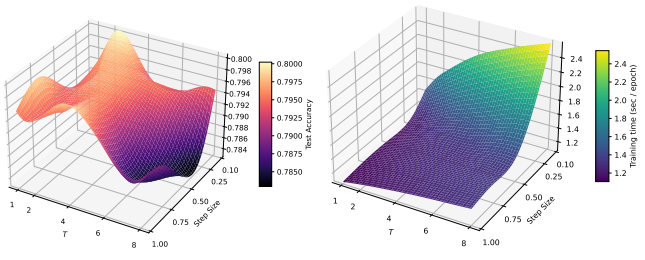
In the IMDB-B, shown in Figure 2b, the ratios demonstrate a markedly different behavior, particularly in the relative contributions of global and local information at both levels. The both ratios for global information decreases modestly, dropping from approximately 50%p to around 48%p by the final epoch. It underscores the growing importance of local information in the IMDB-B, where the interactions between individual nodes may play a more significant role. In fact, as shown in Table 1, IMDB-B graphs have relatively small node counts, a high number of edges and low diameter, resulting in a high connectivity, which suggests that local information plays a crucial role.

The contrasting trends in ratios between two datasets shows the domain-specific nature of global and local integration. In PTC_MR, the growing reliance on global graph-level information suggests that capturing graph-level patterns is critical for achieving higher performance. It implies that we can consider designing models for PTC_MR to incorporate mechanisms that emphasize global graph-level representations. For IMDB-B, we can consider focusing on extracting local features as it implies that global information is less critical.

4.7 Sensitivity Analyses

The sensitivity analysis presented in Figure 3 evaluates the effects of two hyperparameters, T and step size τ , on test accuracy and training time. These results provide insights into the trade-offs between computational efficiency and predictive performance.

Accuracy analysis. Figure 3a shows the relationship between test accuracy and the two hyperparameters. T , represented on the horizontal axis, increases from left to right, while the step size τ decreases from bottom to top. Test accuracy, visualized along the Z-axis (color gradient), demonstrates a highly non-linear dependence on both T and τ. A



(a) Test accuracy (b) Training time

Figure 3: 3D visualization of sensitivity results w.r.t $T \in [1, 8]$ and step size $\tau \in [0.1, 1]$ on PROTEINS.

key observation is the presence of a peak in test accuracy at approximately $T \approx 2$ and $\tau \approx 0.1$, where the accuracy reaches its maximum value of approximately 0.8. This region suggests that moderate values of both T and τ are optimal for achieving the best performance. However, the accuracy sharply declines in regions with excessively high or low values of τ , emphasizing the sensitivity of the model to the step size. The results also indicate that accuracy becomes lower as T increases beyond $T > 4$, likely due to underfitting as it needs a tight time step.

Training time analysis. Figure 3b shows the impact of T and τ on training time, measured in seconds per epoch. Training time increases with T , as expected, since more iterations correspond to longer computation. However, τ also influences training time. Smaller values of τ lead to slower convergence and, consequently, longer training times. The minimum training time (11.02 seconds per epoch) is observed at the lowest $T = 1$ and the largest $\tau = 1$. Conversely, training time grows significantly as T increases or τ becomes smaller $\tau < 0.5$, reflecting the higher computational cost of fine-grained optimization or excessive iterations.

Practical implications. We observe the importance of carefully tuning T and τ to achieve the desired balance between performance and efficiency. It highlights a clear trade-off between test accuracy and training time. While increasing T and decreasing τ can improve test accuracy, these come at the cost of longer training times. The optimal region for balancing accuracy and efficiency appears to lie around $T \in [2, 4]$ and $\tau \in [0.1, 0.25]$, where test accuracy is maximized without excessive computational overhead.

5 Related Works

Message-passing-based GNNs. The foundational approaches in graph neural networks, such as GCN [Kipf and Welling, 2016], GAT [Veličković *et al.*, 2018], GIN [Xu *et al.*, 2018], and GraphSAGE [Hamilton *et al.*, 2017], rely on message-passing mechanisms. GCN use a spectral perspective to propagate information by applying convolutional filters on graph Laplacians. GAT enhance the flexibility of neighborhood aggregation by introducing attention mechanisms (limited to local neighborhoods) to assign importance to neighboring nodes. GIN address the expressiveness of GNNs by ensuring injectivity in aggregation functions. GraphSAGE combines information from a neighbors using functions like mean, pooling, or LSTM, enabling the model to capture more powerful local graph structures. However, their reliance on neighborhood aggregation leads to over-smoothing when multiple layers are stacked, rendering node representations indistinguishable.

Hierarchical pooling-based GNNs. Hierarchical pooling methods aim to capture both local and global information by iteratively coarsening the graph. DiffPool [Ying *et al.*, 2018] uses learned soft cluster assignments to aggregate node features into hierarchies, enabling multi-scale graph representations.

Transformer-based GNNs. GraphTransformer [Dwivedi and Bresson, 2020], SAN [Kreuzer *et al.*, 2021] and Graphormer [Ying *et al.*, 2021; Shi *et al.*, 2022] build upon the Transformer architecture by incorporating structural encodings to model graph-structured data. Unlike GAT, it leverages the advantages of the Transformer's global receptive field by encoding the local positional information of nodes, enabling the model to learn both local and global structures. Recently, hybrid method such as GraphGPS [Rampášek *et al.*, 2022] has emerged, which combines message-passing mechanisms and Transformers to learn both local and global information.

Contrastive learning-based GNNs. Contrastive learning methods have gained significant traction for unsupervised and self-supervised graph representation learning. GraphCL [You *et al.*, 2020] generates augmented views of graphs using operations like node dropping, edge perturbation, and subgraph sampling. AutoGCL [Yin *et al.*, 2022] improves upon GraphCL by introducing learnable view generators, allowing the model to adapt node-level augmentation policies to specific datasets dynamically. InfoGraph [Sun *et al.*, 2019] maximizes the mutual information between graph-level representations and their substructures, such as nodes and edges, to capture hierarchical dependencies.

Reaction-diffusion in GNNs. GREAD [Choi *et al.*, 2023] introduces a approach to graph representation learning by leveraging reaction-diffusion equations. Specifically, it utilizes Turing patterns [Turing, 1990] to capture long-range structure and local structure within both heterophilic and homophilic graphs.

6 Concluding Remarks and Future Work

In this work, we propose a novel Blend-GNN that effectively balances local and global information at both the node and graph-level. By initializing node representations through a diffusion-dominated global encoding, our Blend-GNN ensures stability and coherence, allowing the model to focus on refining localized patterns while effectively blending both local and global contexts. Experimental results show that Blend-GNN achieves significant improvements across diverse graph domains and prediction tasks. Exploring the integration of multimodal data, such as combining graph with text, images, could further enhance Blend-GNN's applicability to real-world scenarios. In the future, we aim to develop method that enable the model to effectively leverage complementary information across modalities.

Ethical Statement

We adhere to fundamental ethical principles, ensuring the responsible use of datasets and the minimization of potential societal harms. All datasets utilized are obtained from publicly available sources and used in compliance with relevant privacy, copyright.

References

[Atkinson *et al.*, 2009] Kendall Atkinson, Weimin Han, and David E Stewart. *Numerical solution of ordinary differential equations*, volume 81. John Wiley & Sons, 2009.

[Beaini *et al.*, 2021] Dominique Beaini, Saro Passaro, Vincent Létourneau, Will Hamilton, Gabriele Corso, and Pietro Liò. Directional graph networks. In *International Conference on Machine Learning*, pages 748–758. PMLR, 2021.

[Bresson and Laurent, 2017] Xavier Bresson and Thomas Laurent. Residual gated graph convnets. *arXiv preprint arXiv:1711.07553*, 2017.

[Cai *et al.*, 2018] Hongyun Cai, Vincent W Zheng, and Kevin Chen-Chuan Chang. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE transactions on knowledge and data engineering*, 30(9):1616–1637, 2018.

[Chamberlain *et al.*, 2021] Ben Chamberlain, James Rowbottom, Maria I Gorinova, Michael Bronstein, Stefan Webb, and Emanuele Rossi. Grand: Graph neural diffusion. In *International conference on machine learning*, pages 1407–1418. PMLR, 2021.

[Chen *et al.*, 2018] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

[Choi *et al.*, 2023] Jeongwhan Choi, Seoyoung Hong, Noseong Park, and Sung-Bae Cho. Gread: Graph neural reaction-diffusion networks. In *International Conference on Machine Learning*, pages 5722–5747. PMLR, 2023.

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

[Dwivedi *et al.*, 2022a] Vijay Prakash Dwivedi, Chaitanya K. Joshi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks, 2022.

[Dwivedi *et al.*, 2022b] Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Graph neural networks with learnable structural and positional representations, 2022.

[Dwivedi *et al.*, 2022c] Vijay Prakash Dwivedi, Ladislav Rampášek, Michael Galkin, Ali Parviz, Guy Wolf, Anh Tuan Luu, and Dominique Beaini. Long range graph benchmark. *Advances in Neural Information Processing Systems*, 35:22326–22340, 2022.

[Hamilton *et al.*, 2017] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[Hu *et al.*, 2019] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*, 2019.

[Kingma and Ba, 2017] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

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

[Kreuzer *et al.*, 2021] Devin Kreuzer, Dominique Beaini, Will Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph transformers with spectral attention. *Advances in Neural Information Processing Systems*, 34:21618–21629, 2021.

[Lim *et al.*, 2022] Derek Lim, Joshua Robinson, Lingxiao Zhao, Tess Smidt, Suvrit Sra, Haggai Maron, and Stefanie Jegelka. Sign and basis invariant networks for spectral graph representation learning. *arXiv preprint arXiv:2202.13013*, 2022.

[Morris *et al.*, 2020] Christopher Morris, Nils M. Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs. In *ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020)*, 2020.

[Parmar *et al.*, 2018] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *International conference on machine learning*, pages 4055–4064. PMLR, 2018.

[Rampášek *et al.*, 2022] Ladislav Rampášek, Michael Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. *Advances in Neural Information Processing Systems*, 35:14501–14515, 2022.

[Rusch *et al.*, 2023] T Konstantin Rusch, Michael M Bronstein, and Siddhartha Mishra. A survey on oversmoothing in graph neural networks. *arXiv preprint arXiv:2303.10993*, 2023.

[Shi *et al.*, 2022] Yu Shi, Shuxin Zheng, Guolin Ke, Yifei Shen, Jiacheng You, Jiyan He, Shengjie Luo, Chang Liu, Di He, and Tie-Yan Liu. Benchmarking graphormer on large-scale molecular modeling datasets. *arXiv preprint arXiv:2203.04810*, 2022.

[Sun *et al.*, 2019] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. *arXiv preprint arXiv:1908.01000*, 2019.

[Turing, 1990] Alan Mathison Turing. The chemical basis of morphogenesis. *Bulletin of mathematical biology*, 52:153–197, 1990.

[Veličković *et al.*, 2018] Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. *arXiv preprint arXiv:1809.10341*, 2018.

[Veličković *et al.*, 2018] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018.

[Wang *et al.*, 2022] Haorui Wang, Haoteng Yin, Muhan Zhang, and Pan Li. Equivariant and stable positional encoding for more powerful graph neural networks. *arXiv preprint arXiv:2203.00199*, 2022.

[Wei *et al.*, 2023] Lanning Wei, Zhiqiang He, Huan Zhao, and Quanming Yao. Search to capture long-range dependency with stacking gnns for graph classification. In *Proceedings of the ACM Web Conference 2023*, pages 588–598, 2023.

[Wong and Yeh, 2019] Tzu-Tsung Wong and Po-Yang Yeh. Reliable accuracy estimates from k-fold cross validation. *IEEE Transactions on Knowledge and Data Engineering*, 32(8):1586–1594, 2019.

[Wu *et al.*, 2020] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[Xu *et al.*, 2018] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

[Yin *et al.*, 2022] Yihang Yin, Qingzhong Wang, Siyu Huang, Haoyi Xiong, and Xiang Zhang. Autogcl: Automated graph contrastive learning via learnable view generators. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 8892–8900, 2022.

[Ying *et al.*, 2018] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems*, 31, 2018.

[Ying *et al.*, 2021] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? *Advances in neural information processing systems*, 34:28877–28888, 2021.

[You *et al.*, 2020] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *Advances in neural information processing systems*, 33:5812–5823, 2020.

[Yu *et al.*, 2024] Zhizhi Yu, Bin Feng, Dongxiao He, Zizhen Wang, Yuxiao Huang, and Zhiyong Feng. Lg-gnn: local-global adaptive graph neural network for modeling both homophily and heterophily. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 2515–2523, 2024.

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