DeepGRAND: Deep Graph Neural Diffusion

Khang Nguyen*
FSOFT AI Center
Ho Chi Minh City, Vietnam
khang.nguyenhoang.vn@gmail.com

Hieu Nong*
FSOFT AI Center
Hanoi, Vietnam
hieubkvn123@gmail.com

Khuong Nguyen
FSOFT AI Center
Ho Chi Minh City, Vietnam
khuongnd@gmail.com

Tan M. Nguyen**
University of California, Los Angeles
Los Angeles, USA
tanmnguyen89@ucla.edu

Vinh Nguyen**
FSOFT AI Center
Ho Chi Minh City, Vietnam
vinhnguyen.maths@gmail.com

***Abstract*—Graph neural networks (GNNs) have achieved remarkable success in numerous domains. Nevertheless, many popular GNNs are known to suffer from over-smoothing. This phenomenon causes node features to become indistinguishable and hurts the classification accuracy as layer depth increases, which limits their effectiveness at capturing complex and long range graph interactions. We propose the Deep Graph Neural Diffusion (DeepGRAND), a continuous-depth graph neural network that is based on the diffusion process on graphs that theoretically alleviates the over-smoothing issue. DeepGRAND pertubes the learnable graph diffusivity and re-scales the underlying diffusion equation by a data-dependent term. We empirically show that DeepGRAND mitigates the accuracy drop-off caused by over-smoothing and surpasses the best accuracy achieved by popular GNNs on various graph deep learning benchmarks. We further demonstrate the advantage of DeepGRAND over many existing graph neural networks in the low label rate regimes.**

***Index Terms*—deep learning, graph neural networks, diffusion**

I. Introduction

Graph neural networks (GNNs) and machine learning on graphs [1] have been successfully applied in a wide range of applications including physical modeling [2], recommender systems [3], and social networks [4]. Recently, advanced GNNs have been developed to further improve the performance of the models and extend their application, which include graph convolutional networks (GCNs) [5], GraphSAGE [6], message passing neural networks (MPNNs) [2], and graph attention networks (GATs) [7]. A well-known problem of GNNs is that the performance of the model decreases significantly with increasing depth. This phenomenon is a common plight of most GNN architectures, and it is referred to as the over-smoothing issue of GNNs [8], [9].

Partial differential equations (PDEs) have been extensively studied and used in a variety of applications, including image processing [10] and computer graphics [11]. Interpreting GNNs as discretization schemes of the underlying diffusion PDE, [12] proposes Graph Neural Diffusion (GRAND), a novel class of continuous-depth GNNs. This framework allows the vast literature on PDEs to be used to improve GNN performance. Despite its strengths, GRAND does not escape the plight of over-smoothing.

Main contributions. In this paper, we propose the Deep Graph Neural Diffusion (DeepGRAND), a novel continuous-depth graph neural network that improves on various aspects of the baseline GRAND [12]. At its core, DeepGRAND introduces a data-dependent scaling term and a perturbation to the diffusion dynamics to alter the spectral and convergent characteristics of the process. With this design, DeepGRAND attains the following advantages:

1) DeepGRAND inherits the diffusive characteristic of GRAND while significantly mitigating the over-smoothing issue.
2) DeepGRAND achieves remarkably better performance than popular GNNs and GRAND variants when fewer nodes are labeled as training data, meriting its use in low-labeling rates situations.
3) Feature representation under the dynamics of DeepGRAND is guaranteed to remain bounded, ensuring numerical stability.

II. Background

A. Graph neural networks and the over-smoothing issue

The majority of the literature on GNNs can be derived from a common message passing scheme, forming a large class of MPNNs that differ from each other by how information is propagated and aggregated in each model [13]. An update rule for all MPNNs can be written in the form

$$\boldsymbol{H}_u = \xi\left(\boldsymbol{X}_u, \bigoplus_{v\in\mathcal{N}_u} \mu(\boldsymbol{X}_u, \boldsymbol{X}_v)\right),$$

where μ is a learnable message function, $\bigoplus$ is a permutation-invariant aggregate function, and ξ is the update function. Three popular MPNNs are GCN [5], GraphSAGE [6], and GAT [7]. In these specific GNN designs, the update rule can be further simplified to

$$\boldsymbol{H}_u = \sigma\left(\sum_{v\in\mathcal{N}_u\cup\{u\}} a_{uv}\mu(\boldsymbol{X}_v)\right), \tag{1}$$

*: co-first authors, **: co-last authors

where a is either given by the normalized augmented adjacency matrix (GCN), a random sampling scheme (GraphSAGE), or the attention mechanism (GAT), μ is a linear transformation, and σ is an activation function.

It is noted that a GNN with more message passing layers suffers from over-smoothing [8], which makes learned features become indistinguishable and hurts the classification accuracy. This phenomenon goes against the common understanding that the deeper the model, the better its learning capacity is [14]. Due to over-smoothing, GNNs are constrained to having small depths and encounter difficulty in capturing long-range dependencies.

Over the past few years, a flurry of research into understanding and alleviating the over-smoothing issue has been conducted [9], [15]–[17].

B. Graph Neural Diffusion

GRAND is a continuous-depth architecture for deep learning on graphs proposed by [12]. It draws inspiration from the heat diffusion process in mathematical physics and follows the same vein as other PDE-inspired neural networks [15], [18], [19]. Central to the formulation of GRAND is the diffusion equation on graphs

$$\frac{\partial \boldsymbol{X}(t)}{\partial t} = \operatorname{div}[\boldsymbol{G}(\boldsymbol{X}(t), t)\nabla \boldsymbol{X}(t)], \tag{2}$$

where $\boldsymbol{G} = \operatorname{diag}\left(a(\boldsymbol{X}_i(t), \boldsymbol{X}_j(t), t)\right)$ is a diagonal matrix giving the diffusivity between connected vertices, which describes the thermal conductance property of the graph.

Let $\boldsymbol{X} \in \mathbb{R}^{n\times d}$ be the matrix representing the initial d features of all n nodes in the graph. The architecture utilises the encoder-decoder design given by $\boldsymbol{Y} = \psi(\boldsymbol{X}(T))$, where for all T, $\boldsymbol{X}(T) \in \mathbb{R}^{n\times d}$ is given by

$$\boldsymbol{X}(T) = \boldsymbol{X}(0) + \int_0^T \frac{\partial \boldsymbol{X}}{\partial t}(t)dt, \quad \text{with } \boldsymbol{X}(0) = \phi(\boldsymbol{X}). \tag{3}$$

In the simplest case when $\boldsymbol{G}$ is only dependent on the initial node features, the differential in (3) simplifies to

$$\frac{\partial \boldsymbol{X}}{\partial t}(t) = (\boldsymbol{A}(\boldsymbol{X}) - \boldsymbol{I})\boldsymbol{X}(t), \tag{4}$$

where $\boldsymbol{A}(\boldsymbol{X}) = [(a(\boldsymbol{X}_i(t), \boldsymbol{X}_j(t), t))]$ is an $n \times n$ matrix with the same structure as the adjacency matrix of the graph. From now on, we omit the term $\boldsymbol{X}$ when writing $\boldsymbol{A}(\boldsymbol{X})$. The entries of $\boldsymbol{A}$ are exactly those in $\boldsymbol{G}$, and thus determine the diffusivity. Furthermore, $\boldsymbol{A}$ can be informally thought of as the attention weight between vertices. Building upon this heuristic, GRAND models the attention matrix $\boldsymbol{A}$ in (4) by the multi-head self-attention mechanism, where

$$\boldsymbol{A} = \frac{1}{h}\sum_{l=1}^{h} \boldsymbol{A}^l(\boldsymbol{X}) \tag{5}$$

with h being the number of heads and the attention matrix $\boldsymbol{A}^l(\boldsymbol{X}) = (a^l(\boldsymbol{X}_i, \boldsymbol{X}_j))$, for $l = 1, \ldots, h$. This specific implementation is called GRAND-l since it becomes a linear matrix differential equation once $\boldsymbol{A}$ is calculated from the initial node features. The authors of [12] also proposed GRAND-nl, an implementation of GRAND where $\boldsymbol{G}$ is dependent on the current node features at each step.

As has been pointed out by [12], [20], many GNN architectures, including GAT and GCN, can be formalized as a discretization scheme of (2) if no non-linearity is used between the layers. The intuition behind the connection can readily be seen from (1), where each subsequent node data is computed similarly or equal to a weighted average of the neighboring node features. This gives rise to the diffusive nature of these architectures.

III. Does GRAND suffer from over-smoothing?

Various works [9], [15] have attributed the occurrence of the over-smoothing issue to the exponential convergence of node representations. We formally define this phenomenon for continuous depth graph neural networks.

Definition 1. *Let $\boldsymbol{X}(t) \in \mathbb{R}^{n\times d}$ denote the feature representation at time $t \geq 0$. $\boldsymbol{X}$ is said to experience over-smoothing if there exists a vector $\boldsymbol{v} \in \mathbb{R}^d$ and constants $C_1, C_2 > 0$ such that for $\boldsymbol{V} = (\boldsymbol{v}, \boldsymbol{v}, \ldots, \boldsymbol{v})^\top$*

$$\|\boldsymbol{X}(t) - \boldsymbol{V}\|_\infty \leq C_1 e^{-C_2 t}. \tag{6}$$

This definition is similar in spirit to the one given by [15].

Proposition 1. *The GRAND-l dynamics given by equations* (3)*,* (4)*, and* (5) *cause $\boldsymbol{X}$ to experience over-smoothing.*

Sketch of proof. $\boldsymbol{A}$ is a right-stochastic matrix with positive entries, hence its Perron-Frobenius eigenvalue is $\alpha_1 = 1$. Moreover, α_1 is a simple eigenvalue and its one-dimensional eigenspace has the basis $\boldsymbol{u} = (1, 1, \ldots, 1)$. Suppose $\{\alpha_1, \alpha_2, \ldots, \alpha_k\}$ is the complex spectrum of $\boldsymbol{A}$. That is, they are all complex eigenvalues of $\boldsymbol{A}$. The matrix $\boldsymbol{A} - \boldsymbol{I}$ has eigenvalues $\beta_i = \alpha_i - 1$ for all $i = \overline{1, k}$, so $\beta_1 = 0$ and $\operatorname{Re}\beta_i < 0$ for all $i = \overline{2, k}$. A Jordan canonical form $\boldsymbol{J}$ of $\boldsymbol{A} - \boldsymbol{I}$ consists of a single entry 0 on the top left most corner, followed by several Jordan blocks $\boldsymbol{J}_i$ corresponding to eigenvalues β_i on the diagonal of the matrix. Let $\boldsymbol{P}$ be the change of basis matrix such that $\boldsymbol{J} = \boldsymbol{P}^{-1}(\boldsymbol{A} - \boldsymbol{I})\boldsymbol{P}$ and the first column of $\boldsymbol{P}$ is $\boldsymbol{u}$ and let $\boldsymbol{Z}(t) = \boldsymbol{P}^{-1}\boldsymbol{X}(t)$. We can rewrite (4) as

$$\frac{\partial \boldsymbol{Z}}{\partial t}(t) = \boldsymbol{J}\boldsymbol{Z}(t).$$

The solution to this matrix differential equation is

$$\boldsymbol{Z}(t) = \exp(t\boldsymbol{J})\boldsymbol{Z}(0).$$

Since $\operatorname{Re}\beta_i < 0$ for all $i = \overline{2, k}$, we readily obtain

$$\lim_{t\to\infty} \boldsymbol{Z}(t) = \lim_{t\to\infty} \exp(t\boldsymbol{J})\boldsymbol{Z}(0) = (\boldsymbol{Z}(0)_{1,1}, \cdots, \boldsymbol{Z}(0)_{1,d})^\top$$

where the convergence is exponential in ∞-norm with regard to t. Recall that the first column of $\boldsymbol{P} = \boldsymbol{u}$, we have

$$\lim_{t\to\infty} \boldsymbol{X}(t) = \boldsymbol{P} \lim_{t\to\infty} \boldsymbol{Z}(t) = \boldsymbol{V}$$

for

$$\boldsymbol{V} = \begin{pmatrix} \boldsymbol{Z}(0)_{1,1} & \boldsymbol{Z}(0)_{1,2} & \dots & \boldsymbol{Z}(0)_{1,d} \\ \boldsymbol{Z}(0)_{1,1} & \boldsymbol{Z}(0)_{1,2} & \dots & \boldsymbol{Z}(0)_{1,d} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{Z}(0)_{1,1} & \boldsymbol{Z}(0)_{1,2} & \dots & \boldsymbol{Z}(0)_{1,d} \end{pmatrix}.$$

We deduce that $\|\boldsymbol{X}(t) - \boldsymbol{V}\|_\infty$ exponentially converges to 0, and we can choose C_1, C_2 suitably so that the inequality (6) is satisfied. □

Proposition 1 conclusively shows that GRAND-l still suffers from over-smoothing. Of course, if we consider more general variants of GRAND, the above arguments no longer hold. Although we can not rigorously assert that over-smoothing affect all implementations of GRAND, it can be argued that the occurrence of this phenomenon in GRAND intuitively makes sense since diffusion have the tendency to "even out" distribution over time. Hence, a purely diffusion based model like GRAND is inherently ill-suited for deep networks.

IV. DeepGRAND: Deeper Graph Neural Diffusion

A. Model formulation

We propose a new model of continuous Graph Neural Network based on GRAND capable of learning at much higher depth (DeepGRAND). It leverages a perturbation to the diffusivity matrix and a scaling factor to make the model both more stable and more resilient to the over-smoothing issue.

Denote the i-th column of $\boldsymbol{X}$ by $\boldsymbol{X}_i$. With the usual 2-norm, we define the column-wise norm matrix $\langle \boldsymbol{X}(t) \rangle^\alpha \in \mathbb{R}^{n \times d}$ as

$$\langle \boldsymbol{X}(t) \rangle^\alpha = \begin{pmatrix} \|\boldsymbol{X}_1\|^\alpha & \|\boldsymbol{X}_2\|^\alpha & \dots & \|\boldsymbol{X}_d\|^\alpha \\ \|\boldsymbol{X}_1\|^\alpha & \|\boldsymbol{X}_2\|^\alpha & \dots & \|\boldsymbol{X}_d\|^\alpha \\ \vdots & \vdots & \ddots & \vdots \\ \|\boldsymbol{X}_1\|^\alpha & \|\boldsymbol{X}_2\|^\alpha & \dots & \|\boldsymbol{X}_d\|^\alpha \end{pmatrix}, \quad (7)$$

for some constant $\alpha > 0$. The dynamics of DeepGRAND is given by (3), with $\frac{\partial \boldsymbol{X}}{\partial t}$ given by

$$\frac{\partial \boldsymbol{X}}{\partial t}(t) = (\boldsymbol{A} - (1+\epsilon)\boldsymbol{I})\, \boldsymbol{X}(t) \odot \langle \boldsymbol{X}(t) \rangle^\alpha, \quad (8)$$

where $\odot$ is the Hadamard product. Hence, our proposed dynamics differs from GRAND's dynamics by the perturbation ϵ and the data-dependent scaling term $\langle \boldsymbol{X}(t) \rangle^\alpha$.

B. How DeepGRAND overcomes the over-smoothing issue

To explain the motivation behind DeepGRAND, we first note that the convergence property of all nodes to a pre-determined feature vector is not necessarily an undesirable trait. It guarantees that our model will remain bounded and not "explode", which helps to ensure numerical stability. Our perturbation by ϵ serves to strengthen this behavior. Its purpose is to reduce the real part of all eigenvalues to negative values, making all node representations converge to 0 in the long run.

We observe that the rate of convergence is the chief factor in determining the range of effective depth. If the model converges very slowly, it is clear that we can train it at high depth without ever having to worry about over-smoothing. As such, if we can control the convergence rate, we would be able to alleviate the over-smoothing issue. The addition of the term $\langle \boldsymbol{X}(t) \rangle^\alpha \in \mathbb{R}^{n \times d}$ serves this purpose. As the node representations come close to their limits, this term acts as a scaling factor to slow down the convergent process.

We present an exact bound for the convergent rate of our model. We heuristically note that a symmetric matrix will almost surely have a simple spectrum. This characteristic has been made precise in some classes of matrices with certain distributions [21]. We will utilize this assumption in the next proposition.

TABLE I
PERFORMANCE OF DEEPGRAND VS. GRAND AT DIFFERENT DEPTHS. THE BEST RESULTS ARE HIGHLIGHTED IN BOLD.

Dataset	Depth	GRAND-l	GRAND-nl	DeepGRAND
Cora	4	83.5 ± 0.4	83.3 ± 0.3	**84.7 ± 0.2**
	16	83.5 ± 0.5	82.5 ± 0.4	**84.5 ± 0.3**
	32	81.8 ± 0.5	81.7 ± 0.3	**84.3 ± 0.4**
	64	76.5 ± 0.6	78.7 ± 1.1	**82.1 ± 0.7**
	128	68.6 ± 0.4	74.6 ± 0.5	**76.4 ± 0.9**
Citeseer	4	68.5 ± 1.7	69.0 ± 1.5	**75.4 ± 0.6**
	16	60.4 ± 2.0	60.7 ± 1.8	**75.0 ± 0.6**
	32	60.1 ± 3.7	58.8 ± 1.6	**74.4 ± 1.1**
	64	55.1 ± 2.6	55.7 ± 2.3	**71.0 ± 1.1**
	128	50.5 ± 2.6	52.9 ± 1.8	**67.9 ± 0.8**
Pubmed	4	77.8 ± 0.1	77.5 ± 0.3	**79.6 ± 1.0**
	16	77.9 ± 0.4	77.8 ± 0.2	**79.5 ± 0.6**
	32	76.7 ± 0.3	78.4 ± 0.4	**79.3 ± 0.4**
	64	74.7 ± 0.5	73.3 ± 1.2	**79.1 ± 0.6**
	128	NA	NA	**78.7 ± 0.5**
Computers	1	85.3 ± 0.6	85.7 ± 0.1	**85.9 ± 0.3**
	2	85.6 ± 2.9	85.0 ± 0.7	**85.8 ± 0.3**
	4	85.4 ± 0.5	84.2 ± 0.5	**85.5 ± 0.5**
	8	55.9 ± 0.3	52.1 ± 0.4	**85.1 ± 0.3**
	16	40.4 ± 1.0	42.1 ± 3.0	**78.7 ± 2.2**
	32	4.2 ± 4.1	8.2 ± 5.1	**48.9 ± 0.3**
Photo	1	93.6 ± 0.2	93.4 ± 0.3	**93.9 ± 0.3**
	2	93.7 ± 0.2	93.3 ± 0.3	**93.8 ± 0.3**
	4	93.5 ± 0.3	92.3 ± 0.4	**93.8 ± 0.2**
	8	926 ± 0.5	92.5 ± 0.6	**93.5 ± 0.4**
	16	83.6 ± 1.7	93.3 ± 0.4	**93.3 ± 0.4**
	32	25.5 ± 5.6	45.2 ± 7.3	**91.1 ± 1.0**
CoauthorCS	1	90.3 ± 1.1	91.1 ± 0.4	**91.4 ± 0.2**
	2	90.3 ± 0.9	90.8 ± 0.1	**91.2 ± 0.7**
	4	90.4 ± 0.7	89.9 ± 0.8	**90.9 ± 0.6**
	8	85.1 ± 1.8	87.2 ± 1.0	**90.7 ± 0.7**
	16	65.2 ± 14.5	77.7 ± 3.7	**88.4 ± 1.1**
	32	48.0 ± 8.9	49.7 ± 12.1	**79.7 ± 2.4**

Proposition 2. *Assuming $\boldsymbol{A}$ is right-stochastic, symmetric and has a simple spectrum. With the dynamics given in* (8)*, we have the bound for each column $\boldsymbol{X}_i$*

$$\|\boldsymbol{X}_i(T)\| \geq \left((2+\epsilon)\alpha T + \|\boldsymbol{X}_i(0)\|^{-\alpha}\right)^{\frac{-1}{\alpha}}, \quad (9)$$

$$\|\boldsymbol{X}_i(T)\| \leq \left(\epsilon\alpha T + \|\boldsymbol{X}_i(0)\|^{-\alpha}\right)^{\frac{-1}{\alpha}}. \quad (10)$$

Proof. Let $\boldsymbol{Y}_i(t) = \|\boldsymbol{X}_i(t)\|^4$, we have

$$\begin{aligned} \boldsymbol{Y}_i'(t) &= 4\|\boldsymbol{X}_i(t)\|^2 \langle \boldsymbol{X}_i'(t), \boldsymbol{X}_i(t) \rangle \\ &= 4\|\boldsymbol{X}_i(t)\|^2 \langle (\boldsymbol{A} - (1+\epsilon)\boldsymbol{I})\boldsymbol{X}_i(t) \|\boldsymbol{X}_i\|^\alpha, \boldsymbol{X}_i(t) \rangle \\ &= 4\|\boldsymbol{X}_i(t)\|^{2+\alpha} \langle (\boldsymbol{A} - (1+\epsilon)\boldsymbol{I})\boldsymbol{X}_i(t), \boldsymbol{X}_i(t) \rangle \\ &= 4\|\boldsymbol{X}_i(t)\|^{2+\alpha} \langle \boldsymbol{A}\boldsymbol{X}_i(t), \boldsymbol{X}_i(t) \rangle - 4(1+\epsilon)\|\boldsymbol{X}_i(t)\|^{4+\alpha} \end{aligned}$$

TABLE II

PERFORMANCE OF GRAND, OTHER POPULAR GNNS, AND DEEPGRAND AT DIFFERENT LABEL RATES. THE BEST RESULTS ARE HIGHLIGHTED IN BOLD. (NOTE: THE RESULTS FOR GRAND++-L, GCN, GAT, AND GRAPHSAGE ARE IMPORTED FROM [20])

Dataset	# labeled	GRAND-l	GRAND-nl	GRAND++-l	GCN	GAT	GraphSage	DeepGRAND
Cora	20	82.9 ± 1.1	82.7±2.5	83.0 ± 1.4	82.1 ± 2.0	79.9 ± 2.3	80.0 ± 2.5	**84.2 ± 0.7**
	10	80.7 ± 2.2	80.5 ± 1.1	80.9 ± 3.0	78.8 ± 5.4	76.3 ± 4.9	75.0 ± 5.0	**82.9 ± 0.8**
	5	77.1 ± 3.1	77.7 ± 2.9	77.8 ± 4.5	73.9 ± 8.0	71.0 ± 5.7	68.1 ± 7.0	**80.9 ± 1.1**
	2	74.2 ± 5.6	69.4 ± 4.3	66.9 ± 10.0	60.9 ± 14.0	58.3± 13.6	54.0 ± 12.2	**76.5 ± 2.0**
	1	57.9± 8.1	55.9 ± 10.0	54.9 ± 16.1	47.7 ± 15.3	47.9± 15.4	43.0 ± 14.0	**70.2 ± 3.3**
Citeseer	20	71.7 ± 2.9	73.2 ± 3.1	73.5 ± 3.3	74.2 ± 2.9	73.2 ± 2.9	72.0 ± 2.8	**74.7 ± 0.8**
	10	66.3 ± 4.2	67.8 ± 4.0	72.3 ± 2.4	72.2 ± 3.5	71.4 ± 4.9	68.9 ± 5.1	**73.5 ± 0.8**
	5	69.0 ± 3.7	66.9 ± 4.6	70.0 ± 3.6	67.2 ± 4.2	67.4 ± 5.1	64.8 ± 5.2	**72.0 ± 1.0**
	2	58.4 ± 9.0	56.4 ± 5.5	65.0 ± 8.3	58.1 ± 9.8	55.6 ± 9.2	54.4 ± 11.4	**69.7 ± 3.0**
	1	49.7 ± 8.7	47.3 ± 6.7	**59.0 ± 9.6**	48.9 ± 10.2	50.3 ± 14.3	48.8 ± 11.5	58.4 ± 2.5
Pubmed	20	78.4 ± 0.5	75.2 ± 1.8	79.2 ± 1.4	76.9 ± 3.3	75.6 ± 4.1	74.6 ± 3.1	**79.5 ± 0.6**
	10	74.1 ± 2.0	74.2 ± 2.0	75.1 ± 3.9	72.6 ± 3.2	72.4 ± 3.5	70.7 ± 3.1	**78.9 ± 1.5**
	5	71.1 ± 1.9	72.1 ± 2.2	72.0 ± 1.9	68.7 ± 7.9	68.5 ± 5.8	66.1 ± 6.2	**77.1 ± 1.1**
	2	71.4 ± 3.9	65.5 ± 9.5	69.3 ± 4.9	60.5 ± 16.2	60.2 ± 14.4	59.0 ± 12.7	**72.3 ± 1.8**
	1	62.4 ± 7.6	63.5 ± 5.5	65.9 ± 4.9	58.6 ± 12.8	58.8 ± 12.8	55.5 ± 12.7	**70.0 ± 1.8**
Computers	20	84.0 ± 1.0	83.3 ± 1.2	85.7 ± 0.5	82.9 ± 1.5	80.1 ± 1.8	80.0 ± 1.0	**87.1 ± 0.5**
	10	82.3 ± 2.2	81.3 ± 3.0	83.0 ± 0.8	82.5 ± 0.7	76.0 ± 0.4	74.7 ± 1.3	**85.7 ± 1.1**
	5	78.7 ± 0.8	79.7 ± 2.0	**82.6 ± 0.6**	82.5 ± 1.0	71.4 ± 7.3	64.8 ± 1.6	82.4 ± 0.3
	2	66.2 ± 11.3	65.1 ± 8.3	76.5 ± 1.5	**76.9 ± 1.5**	65.1 ± 8.9	42.6 ± 42.9	76.6 ± 1.4
	1	49.8 ± 15.0	47.3 ± 11.2	67.7 ± 0.4	49.5 ± 1.7	37.1 ± 7.8	27.7 ± 2.4	**69.3 ± 2.7**
Photo	20	93.2 ± 0.4	91.8 ± 1.5	**93.6 ± 0.4**	92.0 ± 0.1	89.4 ± 2.5	91.3 ± 0.7	93.5 ± 0.7
	10	90.8 ± 1.4	89.0 ± 2.5	90.7 ± 1.2	90.4 ± 0.4	87.4 ± 2.4	84.4 ± 1.8	**92.3 ± 0.4**
	5	88.1 ± 2.6	88.3 ± 1.6	88.3 ± 1.2	88.9 ± 1.6	83.0 ± 3.6	78.3 ± 1.9	**90.5 ± 1.0**
	2	82.6 ± 3.0	80.6 ± 4.4	83.7 ± 0.9	83.6 ± 0.7	76.9 ± 4.9	51.9 ± 4.2	**85.1 ± 0.3**
	1	75.1 ± 5.4	76.3 ± 4.8	83.1 ± 0.8	82.9 ± 2.2	73.6 ± 8.2	45.4 ± 7.1	**83.3 ± 1.9**
CoauthorCS	20	91.0 ± 0.6	90.6 ± 1.0	90.8 ± 0.3	91.1 ± 0.4	80.0 ± 2.9	91.3 ± 0.4	**91.7 ± 0.6**
	10	89.0 ± 2.1	**90.0 ± 0.7**	86.9 ± 0.5	88.6 ± 0.5	74.7 ± 3.4	89.7 ± 0.4	89.8 ± 0.7
	5	84.2 ± 3.6	87.0 ± 2.0	84.8 ± 0.8	86.7 ± 0.4	71.7 ± 4.5	**89.1 ± 0.7**	88.2 ± 0.7
	2	75.2 ± 3.8	76.7 ± 6.85	76.5 ± 1.9	**83.6 ± 1.5**	63.1 ± 6.1	76.5 ± 1.3	82.1 ± 3.4
	1	56.6 ± 8.4	66.44 ± 8.17	60.3 ± 1.5	65.2 ± 2.3	51.1 ± 5.2	61.4 ± 1.4	**71.1 ± 1.7**

Since $\boldsymbol{A}$ is a right-stochastic and symmetric matrix, we have

$$\langle \boldsymbol{A}\boldsymbol{X}_i(t), \boldsymbol{X}_i(t)\rangle \leq \|\boldsymbol{A}\|\|\boldsymbol{X}_i(t)\|^2 = \|\boldsymbol{X}_i(t)\|^2,$$

and

$$-\langle \boldsymbol{A}\boldsymbol{X}_i(t), \boldsymbol{X}_i(t)\rangle \leq \|\boldsymbol{A}\|\|\boldsymbol{X}_i(t)\|^2 = \|\boldsymbol{X}_i(t)\|^2,$$

so that

$$-\|\boldsymbol{X}_i(t)\|^2 \leq \langle \boldsymbol{A}\boldsymbol{X}_i(t), \boldsymbol{X}_i(t)\rangle \leq \|\boldsymbol{X}_i(t)\|^2.$$

Hence, we deduce that

$$4\|\boldsymbol{X}_i(t)\|^{4+\alpha}(-2-\epsilon) \leq \boldsymbol{Y}_i'(t) \leq 4\|\boldsymbol{X}_i(t)\|^{4+\alpha}(-\epsilon). \quad (11)$$

Multiply both sides of (11) with $-\frac{\alpha}{4}\|\boldsymbol{X}_i(t)\|^{-4-\alpha} = -\frac{\alpha}{4}\boldsymbol{Y}_i(t)^{-1-\alpha/4}$, and by noting that $-\frac{\alpha}{4}\boldsymbol{Y}_i'\boldsymbol{Y}_i^{-1-\alpha/4} = (\boldsymbol{Y}_i^{-\alpha/4})'$, we have

$$\alpha(2+\epsilon) \geq (\boldsymbol{Y}_i^{-\alpha/4})' \geq \alpha\epsilon.$$

Integrate from 0 to T and rearranging the appropriate terms, we get

$$(2+\epsilon)\alpha T + \boldsymbol{Y}_i(0)^{-\alpha/4} \geq \boldsymbol{Y}_i(T)^{-\alpha/4} \geq \epsilon\alpha T + \boldsymbol{Y}_i(0)^{-\alpha/4}.$$

Finally, by noting that $\boldsymbol{Y}_i^{\frac{-\alpha}{4}} = \|\boldsymbol{X}_i\|^{-\alpha}$, we obtain the desired bounds. □

The inequalities (9), (10) ensure that each column $\boldsymbol{X}_i$ is bounded in norm on both sides by polynomial-like terms. Hence, it no longer converges at exponential speed. The over-smoothing problem is thus alleviated.

V. EXPERIMENTAL RESULTS

In this section, we conduct experiments to compare the performance of our proposed method DeepGRAND and the baseline GRAND. We show that DeepGRAND is able to achieve better accuracy when trained with higher depth and a limited number of labeled nodes per class. We also compare our results with those of GRAND++ [20] and popular GNNs architectures such as GCN [3], GAT [7], and GraphSage [6]. GRAND++ is a GRAND variant where a source term is added into the dynamics to reduce the effect of over-smoothing. In table II, we reported the results of GRAND++-l, the best-performing variant of GRAND++.

A. Experiment design

We evaluate our proposed method using the following experiments:

- **Performance with different depths:** For all benchmarks, we trained variants of GRAND and DeepGRAND using the same integration limits T across a wide range of values and compare the respective test accuracies.
- **Performance with different label rates:** For all benchmarks, we trained GRAND-l, GRAND-nl, other popular GNNs, and DeepGRAND with different numbers of labeled nodes per class.

Datasets. We conduct our experiments on a wide range of popular node classification datasets, including Cora [22], Citeseer [23], Pubmed [24], Computers [25], Photo [25], and CoauthorCS [26].

B. DeepGRAND is more adaptable to deep architectures

In table I, we provide empirical evidence for our argument that the dynamics of DeepGRAND is more resilient to deeper architectures. Specifically, we compared the change in performance of both GRAND and DeepGRAND as the integration limit T increases. Following [12], for Cora, Citeseer, and Pubmed, we used the default planetoid split with 10 random seeds per split. For Computers, Photo, and CoauthorCS, we split the datasets randomly and used 10 random seeds per split. For each seed, we used the default label rate of 20 labeled nodes per class. On all benchmarks, our proposed dynamics performed substantially better across all depths. Furthermore, the performance degradation as depth increases is less significant than that of GRAND, indicating that over-smoothing has been alleviated.

C. DeepGRAND is more resilient under limited labeled training data

In Table II, we provide empirical results to demonstrate that DeepGRAND achieves superior accuracies compared to other GNNs with a limited number of labeled nodes per class. For all of the benchmarks: Cora, Citeseer, Pubmed, Computers, Photo, and CoauthorCS, we used grid search to find the optimal T values and evaluate the performance under different numbers of labeled nodes per class. For each dataset, we experimented with 1, 2, 5, 10, and 20 labeled nodes per class and compared the test accuracies between GRAND, GRAND++, GCN, GAT, GraphSage, and DeepGRAND. Our results show that DeepGRAND outperforms both GRAND variants and common GNN architectures like GCN, GAT, and GraphSage.

VI. Conclusion

We propose DeepGRAND, a novel class of continuous-depth graph neural networks that leverage a data-dependent scaling term and a perturbation to the graph diffusivity to decrease the saturation rate of the underlying diffusion process, thus alleviating the over-smoothing issue. We also prove that the proposed method stabilizes the learning of the model by controlling the norm of the node representation. We theoretically and empirically show the advantage of DeepGRAND over GRAND and other popular GNNs in terms of resiliency to over-smoothing and overall performance. It is interesting to leverage advanced methods in improving the Neural ODEs [27] to further develop DeepGRAND.

References

[1] Bronstein, M., Bruna, J., LeCun, Y., Szlam, A. & Vandergheynst, P. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*. **34**, 18-42 (2017)

[2] Gilmer, J., Schoenholz, S., Riley, P., Vinyals, O. & Dahl, G. Neural message passing for quantum chemistry. *International Conference On Machine Learning*. pp. 1263-1272 (2017)

[3] Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W. & Leskovec, J. Graph convolutional neural networks for web-scale recommender systems. *Proceedings Of The 24th ACM SIGKDD International Conference On Knowledge Discovery And Data Mining*. pp. 974-983 (2018)

[4] Qiu, J., Tang, J., Ma, H., Dong, Y., Wang, K. & Tang, J. DeepInf: Social Influence Prediction with Deep Learning. *Proceedings Of The 24th ACM SIGKDD International Conference On Knowledge Discovery And Data Mining (KDD'18)*. (2018)

[5] Kipf, T. & Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. *Proceedings Of The 5th International Conference On Learning Representations*. (2017)

[6] Hamilton, W., Ying, R. & Leskovec, J. Inductive Representation Learning on Large Graphs. *Proceedings Of The 31st International Conference On Neural Information Processing Systems*. pp. 1025-1035 (2017)

[7] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P. & Bengio, Y. Graph Attention Networks. *International Conference On Learning Representations*. (2018)

[8] Li, Q., Han, Z. & Wu, X. Deeper Insights into Graph Convolutional Networks for Semi-Supervised Learning. (arXiv,2018)

[9] Oono, K. & Suzuki, T. Graph Neural Networks Exponentially Lose Expressive Power for Node Classification. *International Conference On Learning Representations*. (2020)

[10] Perona, P. & Malik, J. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions On Pattern Analysis And Machine Intelligence*. **12**, 629-639 (1990)

[11] Sun, J., Ovsjanikov, M. & Guibas, L. A Concise and Provably Informative Multi-Scale Signature Based on Heat Diffusion. *Comput. Graph. Forum*. **28** pp. 1383-1392 (2009,7)

[12] Chamberlain, B., Rowbottom, J., Goronova, M., Webb, S., Rossi, E. & Bronstein, M. GRAND: Graph Neural Diffusion. *Proceedings Of The 38th International Conference On Machine Learning, (ICML) 2021, 18-24 July 2021, Virtual Event*. (2021)

[13] Bronstein, M., Bruna, J., Cohen, T. & Veličković, P. Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges.

[14] Li, G., Müller, M., Ghanem, B. & Koltun, V. Training Graph Neural Networks with 1000 layers. *International Conference On Machine Learning (ICML)*. (2021)

[15] Rusch, T., Chamberlain, B., Rowbottom, J., Mishra, S. & Bronstein, M. Graph-Coupled Oscillator Networks. *Proceedings Of The 39th International Conference On Machine Learning*. **162** pp. 18888-18909 (2022,7,17)

[16] Zhao, L. & Akoglu, L. PairNorm: Tackling Oversmoothing in GNNs. *International Conference On Learning Representations*. (2020)

[17] Nguyen, K., Nguyen, T., Nong, H., Nguyen, V., Ho, N. & Osher, S. Revisiting Over-smoothing and Over-squashing Using Ollivier-Ricci Curvature. *International Conference On Machine Learning (ICML)*. (2023), in press.

[18] Chamberlain, B., Rowbottom, J., Eynard, D., Giovanni, F., Dong, X. & Bronstein, M. Beltrami Flow and Neural Diffusion on Graphs. *Advances In Neural Information Processing Systems*. (2021)

[19] Eliasof, M., Haber, E. & Treister, E. PDE-GCN: Novel Architectures for Graph Neural Networks Motivated by Partial Differential Equations. *Advances In Neural Information Processing Systems*. (2021)

[20] Thorpe, M., Nguyen, T., Xia, H., Strohmer, T., Bertozzi, A., Osher, S. & Wang, B. GRAND++: Graph Neural Diffusion with A Source Term. *International Conference On Learning Representations*. (2022)

[21] Tao, T. & Vu, V. Random matrices have simple spectrum. *Combinatorica*. (2014,12)

[22] McCallum, A., Nigam, K., Rennie, J. & Seymore, K. Automating the Construction of Internet Portals with Machine Learning. *Information Retrieval*. **3**, 127-163 (2000,7)

[23] Giles, C., Bollacker, K. & Lawrence, S. CiteSeer. *Proceedings Of The Third ACM Conference On Digital Libraries - DL '98*. (1998)

[24] Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B. & Eliassi-Rad, T. Collective Classification in Network Data. *AI Magazine*. **29**, 93 (2008,9)

[25] McAuley, J., Targett, C., Shi, Q. & Hengel, A. Image-based Recommendations on Styles and Substitutes. (arXiv,2015)

[26] Monti, F., Boscaini, D., Masci, J., Rodolà, E., Svoboda, J. & Bronstein, M. Geometric deep learning on graphs and manifolds using mixture model CNNs. (arXiv,2016)

[27] Chen, R., Rubanova, Y., Bettencourt, J. & Duvenaud, D. Neural Ordinary Differential Equations. *Advances In Neural Information Processing Systems*. **31** (2018)