Coupling Graph Neural Networks with Non-Integer Order Dynamics: A Robustness Study

Qiyu Kang[1]*, **Kai Zhao**[1]*, **Yang Song**[2], **Yihang Xie**[1],
Yanan Zhao[1], **Sijie Wang**[1], **Rui She**[1], **Wee Peng Tay**[1]
[1]Nanyang Technological University
[2]C3 AI, Singapore

Abstract

In this work, we rigorously investigate the robustness of graph neural fractional-order differential equation (FDE) models. This framework extends beyond traditional graph neural ordinary differential equation (ODE) models by implementing the time-fractional Caputo derivative. Utilizing fractional calculus allows our model to consider long-term dependencies during the feature updating process, diverging from the Markovian updates seen in traditional graph neural ODE models. The efficacy of FDE models in surpassing ODE models has been confirmed in a different submitted work, particularly in environments free from attacks or perturbations. While traditional graph neural ODE models have been verified to possess a degree of stability and resilience in the presence of adversarial attacks in existing literature, the robustness of graph neural FDE models, especially under adversarial conditions, remains largely unexplored. This paper undertakes a detailed assessment of the robustness of graph neural FDE models. We establish a theoretical foundation outlining the robustness features of graph neural FDE models, highlighting that they maintain more stringent output perturbation bounds in the face of input and functional disturbances, relative to their integer-order counterparts. Through rigorous experimental assessments, which include graph alteration scenarios and adversarial attack contexts, we empirically validate the improved robustness of graph neural FDE models against their conventional graph neural ODE counterparts.

1 Introduction

Graph Neural Networks (GNNs) have emerged as an influential tool capable of extracting meaningful representations from intricate datasets, such as social networks [1] and molecular structures [2]. Despite their impressive capability, GNNs have been found susceptible to adversarial attacks [3–5], with modifications or injections into the graph often causing significant degradation in performance. In real-world scenarios, it is common for data to be perturbed during the training or testing phases [6, 7], highlighting the importance of studying the robustness of GNNs. For instance, in financial systems, fraudulent activities may introduce slight perturbations into transactional data, making it paramount for the underlying models to remain robust against these adversarial changes. Similarly, in social networks, misinformation or the presence of bots can skew the data, which can subsequently impact the insights drawn from it. Therefore, the robustness of GNNs is not just a theoretical concern but a practical necessity. Several defensive strategies have been established to counteract the damaging implications of adversarial attacks on graph data. Approaches such as GARNET [8], GNN-Guard [9], RGCN [10], and Pro-GNN [11] are grounded in preprocessing techniques that aim to remove adversarial alterations to the structure before GNN training commences. Nonetheless, these methods

*First two authors contributed equally to this work. Contact: kang0080@e.ntu.edu.sg, kai.zhao@ntu.edu.sg.

often necessitate the exploration of graph structure properties, leading to higher computational costs. Furthermore, these strategies are more suitably tailored to combat poisoning attacks.

Recent advances have witnessed a growing use of dynamical system theory in designing and understanding GNNs. Models like CGNN [12], GRAND [13], GRAND++ [14], GraphCON [15], and CDE [16] employ neural ordinary differential equations (ODEs) to offer a dynamical system perspective on graph node feature evolution. Typically, these dynamics can be described by:

$$\frac{\mathrm{d}\mathbf{X}(t)}{\mathrm{d}t} = \mathcal{F}(\mathbf{W}, \mathbf{X}(t)). \tag{1}$$

In this formulation, $\mathbf{X}(t)$ represents the *evolving node features* with $\mathbf{X}(0)$ as the initial input features, while $\mathbf{W}$ is the graph's adjacency matrix. The function, $\mathcal{F}$, is specifically tailored for graph dynamics. As a case in point, [13] deploys an attention-based aggregation mechanism akin to heat diffusion on the graph. Building on this approach, [17] demonstrates the inherent robustness of the graph diffusion process within GRAND. Motivated by the Beltrami diffusion equation [18], they introduced a model based on the Beltrami flow (abbreviated as GraphBel) and designed for enhanced robustness, particularly in the face of topological perturbations.

Recent studies have ventured into the intersection of GNNs and fractional calculus [19]. One prominent example is the FRactional-Order graph Neural Dynamical network (FROND) framework [20]. Distinct from conventional GNNs grounded in ODEs, FROND leverages FDEs, with dynamics represented as:

$$D_t^{\beta}\mathbf{X}(t) = \mathcal{F}(\mathbf{W}, \mathbf{X}(t)),\ \beta \in (0, 1]. \tag{2}$$

The function $\mathcal{F}(\mathbf{W}, \mathbf{X}(t))$ maintains its form as in (1). The Caputo fractional derivative, denoted by D_t^{β}, infuses memory into the temporal dynamics (see Section 3.3 for more details). For $\beta = 1$, the equation reverts to the familiar first-order dynamics as in (1). The distinction lies in the fact that the conventional integer-order derivative measures the function's *instantaneous change rate*, concentrating on the proximate vicinity of the point. *In contrast, the fractional-order derivative [21] is influenced by the entire historical trajectory of the function,* which substantially diverges from the localized impact found in integer-order derivatives.

Incorporating a fractional derivative provides GNNs an avenue to mitigate the prevalent oversmoothing problems by enabling algebraic convergence [20], different from the standard exponential convergence. Further, with the integration of fractional dynamics, FROND can elevate performance on graph-structured datasets. Given its modularity, *FROND can effortlessly merge with existing graph ODE frameworks, potentially increasing their effectiveness, especially with diverse β values, without incorporating any additional training parameters to the underlying graph neural ODE models.* Critically, β acts as a proxy for the extent of memory in the feature dynamics: a value of $\beta = 1$ corresponds to Markovian dynamics, while $\beta < 1$ denotes non-Markovian dynamics. This nuance is further visualized in Fig. 1, where a $\beta < 1$ signifies nontrivial skip connections across model discretization timestamps.

Though FROND showcases proficiency in decoding complex graph data patterns, its robustness against adversarial perturbations remains an area of exploration. By broadening the order of time derivatives from integers to real numbers, fractional calculus can encapsulate more intricate dynamics and data relationships, such as long-range memory effects, where the system's current state is influenced by its comprehensive history, not merely its recent states. This capability augments a GNN's ability to more accurately represent the generative processes underlying graph structures and node features across layers, rendering them less susceptible to noise and perturbations. In this work, we delve deeply into the ramifications of the fractional order parameter β on the robustness attributes of graph neural FDEs. Our analysis suggests a monotonic relationship between the model's Lipschitz constant and the parameter β, with smaller β values indicating augmented robustness.

Our contributions can be encapsulated as follows:

- We investigate the robustness characteristics of FROND, providing a rigorous theoretical foundation. Our findings highlight that in the presence of input perturbations or function disturbances, FROND exhibit tighter output perturbation bounds relative to their integer-order counterparts.
- Through extensive experimental evaluations, including scenarios of graph modifications and adversarial injection attacks, we empirically demonstrate the superior robustness of FROND in contrast to conventional graph neural ODE models.

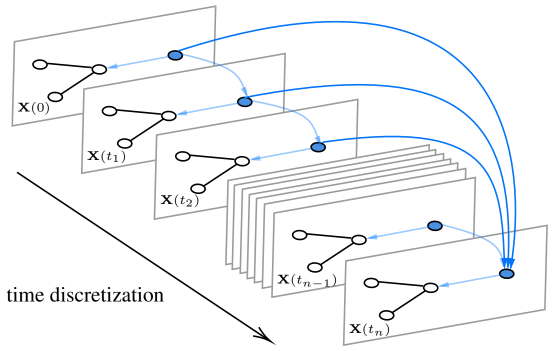


Figure 1: Unlike Euler discretization in graph neural ODE models, FROND integrates connections to historical times, enabling memory effects. The dark blue connections in FROND, absent in ODEs, have weights correlating with $\mu_{j,n}(\beta)$, as outlined in (10).

2 Related work

2.1 Graph Neural ODEs

In the seminal work of [13], the authors ingeniously modeled information propagation akin to the heat diffusion process of a substance. This novel perspective has provided profound insights into the dynamics of graph-structured data. Extending this paradigm, the Beltrami diffusion model was incorporated in [22] to enhance the rewiring mechanism within the graph structure. In a further advancement, GRAND++ [14] introduced a source term into the heat diffusion process, focusing on situations with low-labeling rates in graph deep learning. GraphCON [15] progressed dynamics on graphs from the first-order heat diffusion equation to the second-order system of controlled and damped oscillators, thereby mitigating the exploding and vanishing gradients problem in deep GNNs. Recent contributions based on graph neural diffusion [23, 24, 16] have effectively dealt with heterophily graphs. The paper [17] presented GraphBel, a diffusion process that has significantly augmented the robustness of graph models.

2.2 Adversarial Attacks and Defenses on Graphs

A plethora of research has consistently underscored the vulnerability of graph deep learning models to adversarial perturbations. Essentially, even inconspicuous alterations to the input data can misdirect a graph neural network into producing fallacious predictions. Adversarial attacks on GNNs typically fall into two categories based on the method of perturbation: Graph Modification Attacks (GMA) and Graph Injection Attacks (GIA). GMA involves manipulating the topology of a graph, primarily by adding or removing edges [25–29]. This category also encompasses perturbations to node features [5, 30, 28, 31, 32]. In contrast, Graph Injection Attacks (GIA) permit adversaries to incorporate malicious nodes into the original graph [33–38]. GIA is a stronger form of attack on graph data [38] as it introduces both structural and feature perturbations to the graph.

The defensive strategies employed in GNNs can be broadly categorized into pre-processing methods and the design of robust architectures. Methods such as GNN-GUARD [9], Pro-GNN [11],GARNET[8] and GCN-SVD [39] focus on cleansing or pruning the graph, with the aim of maintaining the integrity of the original adjacency matrix, thereby mitigating perturbations. On another front, methods like RGCN [10] and Soft-Median-GCN [40] are tailored to enhance the inherent architecture of GNNs, making them more resilient to feature perturbations. Distinctly, our approach diverges from these conventional defense mechanisms. **Instead of proposing an entirely new defensive technique, our focus is on bolstering the robustness of existing graph neural ODE models by seamlessly integrating the principles of FROND, without introducing any additional training parameters to the backbone graph neural ODE models.**

3 Preliminaries

3.1 Notation

Let us consider a graph $\mathcal{G} = (\mathcal{V}, \mathbf{W})$, in which $\mathcal{V} = 1, \ldots, N$ represents a set of N nodes. The $N \times N$ matrix $\mathbf{W} := (W_{ij})$ has elements W_{ij} indicating the edge weight between the i-th and j-th feature vectors with $W_{ij} = W_{ji}$. The node features at any given time t can be denoted by $\mathbf{X}(t) \in \mathbb{R}^{|\mathcal{V}| \times N}$, where N corresponds to the dimension of the node feature. In this matrix, the feature vector for the i-th vertex in $\mathcal{V}$ at time t can be represented as the i-th row of $\mathbf{X}(t)$, indicated by $\mathbf{x}_i^{\intercal}(t)$.

3.2 Graph Neural ODE Models

The existing body of literature presents a variety of graph dynamical models. Most of these models [13, 17, 22] take inspiration from transport processes like the heat equation with different diffusivities D to tailor their functionalities. Notably, the GraphCON model [15] conceptualizes graph nodes as coupled oscillators, assimilating the ensuing dynamics.
GRAND [13] incorporates the following dynamical system for graph learning:

$$\frac{\partial \mathbf{X}(t)}{\partial t} = \operatorname{div}(D(\mathbf{X}(t), t) \odot \nabla \mathbf{X}(t)) = (\mathbf{A}(\mathbf{X}(t)) - \mathbf{I})\mathbf{X}(t). \tag{3}$$

The initial condition $\mathbf{X}(0)$ is provided by the graph input features. Here, $\odot$ represents the element-wise product, and D is a diagonal matrix of size $|\mathcal{E}| \times |\mathcal{E}|$ with elements $\operatorname{diag}(a(\mathbf{x}_i(t), \mathbf{x}_j(t), t))$. The function $a(\cdot)$ serves as a similarity measure for vertex pairs. As such, the diffusion equation can be reframed as (3), where $\mathbf{A}(\mathbf{X}(t)) = (a(\mathbf{x}_i(t), \mathbf{x}_j(t)))$ constitutes a learnable attention matrix to depict the graph structure. $\mathbf{I}$ is the identity matrix.
By extending the concepts of Beltrami flow [18, 17], the stable graph neural ODE model GraphBel is formulated as:

$$\frac{\mathrm{d}\mathbf{X}(t)}{\mathrm{d}t} = (\mathbf{A}_{\mathbf{S}}(\mathbf{X}(t)) \odot \mathbf{B}_{\mathbf{S}}(\mathbf{X}(t)) - \Psi(\mathbf{X}(t)))\mathbf{X}(t), \tag{4}$$

where $\odot$ represents element-wise multiplication. Both $\mathbf{A}_{\mathbf{S}}(\cdot)$ and $\mathbf{B}_{\mathbf{S}}(\cdot)$ serve distinct purposes: the former acts as a learnable attention function, while the latter operates as a normalized vector map. $\boldsymbol{\Psi}(\mathbf{X}(t))$ is a diagonal matrix where $\Psi(\mathbf{x}_i, \mathbf{x}_j) = \sum_{\mathbf{x}_j} (\mathbf{A}_{\mathbf{S}} \odot \mathbf{B}_{\mathbf{S}})(\mathbf{x}_i, \mathbf{x}_j)$.
Using a graph coupled dynamical system, GraphCON [41] is given by

$$\frac{\partial \mathbf{Y}(t)}{\partial t} = \sigma(\mathbf{F}_\theta(\mathbf{X}(t), t)) - \gamma \mathbf{X}(t) - \alpha \mathbf{Y}(t), \ \frac{\partial \mathbf{X}(t)}{\partial t} = \mathbf{Y}(t), \tag{5}$$

where $\mathbf{F}_\theta(\cdot)$ is a learnable 1-neighborhood coupling function, σ denotes an activation function, γ and α are adjustable parameters.

Remark 1. *By leveraging the numerical solvers introduced in [42], one can efficiently solve* (3)*,* (4)*, and* (5) *where the initial* $\mathbf{X}(0)$ *represents the input features. This yields the terminal node embeddings, denoted as* $\mathbf{X}(T)$*, at time* T*. Subsequently,* $\mathbf{X}(T)$ *can be utilized for downstream tasks such as node classification or link prediction.*

3.3 Fractional-order Differential Equation (FDE)

Within the FDE framework, the fractional time derivative is typically characterized using the Caputo derivative [43], a prevalent choice for modeling real-world phenomena [19]. It is expressed as:

$$D_t^\beta f = \frac{1}{\Gamma(n-\beta)} \int_0^t (t-\tau)^{n-\beta-1} \frac{\mathrm{d}^n f}{\mathrm{d}\tau^n} \,\mathrm{d}\tau, \quad \beta > 0 \tag{6}$$

where β is the fractional order, n is the smallest integer greater than β, Γ is the gamma function, f is a scalar function defined over the interval $[0, b]$, and $\frac{\mathrm{d}^n f}{\mathrm{d}\tau^n}$ is the standard n-th order derivative. A distinguishing trait of the Caputo derivative is its capability to incorporate *memory effects*. This is underscored by observing that the fractional derivative at time t in (6) *aggregates historical states spanning the interval* $0 \le \tau \le t$. For the special case where $\beta = 1$, the definition collapses to the standard first-order derivative as $D_t^\beta f = \frac{\mathrm{d}f}{\mathrm{d}\tau}$. For a vector-valued function, the fractional derivative is defined component-wise for each dimension, similar to the first-order derivative. Thus, while our discussion centers on scalar functions in Sections 3.3 and 3.4, its extension to vector-valued functions is straightforward. A more detailed, self-contained exposition of the Caputo derivative can be found in the supplementary material.

A crucial concept in fractional calculus and its applications is the Mittag-Leffler function $E_\beta(z)$ [19]. This function has been recognized as the natural extension of the exponential function to the fractional domain, allowing the modeling of more complex phenomena. It is an integral part of the solutions to many fractional differential equations, thereby playing a significant role in the analysis and applications of such systems. Specifically, as per [19][Theorem 4.3], given $y(t) := E_n\left(\lambda t^\beta\right)$,

$x \geq 0$, then

$$D_t^\beta y(t) = \lambda y(t). \tag{7}$$

We present the formal definition of the Mittag-Leffler function below:

Definition 1 (Mittag-Leffler function)**.** *Let $\beta > 0$. The function E_β defined by*

$$E_\beta(z) := \sum_{j=0}^{\infty} \frac{z^j}{\Gamma(j\beta + 1)}, \tag{8}$$

whenever the series converges, is called the Mittag-Leffler function of order β.

This extension of the exponential function becomes more evident when considering the case $\beta = 1$, reducing the Mittag-Leffler function to the familiar exponential function:

$$E_1(z) = \sum_{j=0}^{\infty} \frac{z^j}{\Gamma(j+1)} = \sum_{j=0}^{\infty} \frac{z^j}{j!} = \exp(z). \tag{9}$$

It is also well-established that $\exp(z)$ acts as the eigenfunction of ODEs; specifically, $\exp(\lambda t)$ solves (7) for $\beta = 1$.

3.4 Numerical Solvers for FROND

Differing from works by [44–46] on FROND with integer $\beta = 1$, our research navigates through FDEs with non-integer β. We present the fractional Adams–Bashforth–Moulton method, elucidating time's role as a continuous layer index and revealing memory dependence emerging as dense or skip connections, owing to the non-local nature of fractional derivatives (refer to Fig. 1).

Basic predictor. Referencing [47], we first employ a preliminary numerical solver called "predictor" through time discretisation $t_j = jh$, where the discretisation parameter h is a small positive value:

$$\mathbf{X}^{\mathrm{P}}(t_n) = \sum_{j=0}^{\lceil\beta\rceil-1} \frac{t_n^j}{j!}\mathbf{X}^{(k)}(0) + \frac{1}{\Gamma(\beta)}\sum_{j=0}^{n-1} \mu_{j,n}(\beta)\mathcal{F}(\mathbf{W}, \mathbf{X}(t_j)), \tag{10}$$

with coefficients $\mu_{j,n}(\beta)$ outlined in [47][eq.17]. For $\beta = 1$, this method reduces to the Euler solver [44], where $\mu_{j,n} \equiv h$, resulting in $\mathbf{X}^{\mathrm{P}}(t_n) = \mathbf{X}^{\mathrm{P}}(t_{n-1}) + h\mathcal{F}(\mathbf{W}, \mathbf{X}(t_{n-1}))$.

4 Methodology

In this section, we analyze the output boundary of (11) under specific perturbations, leveraging the properties of the Mittag-Leffler function to illustrate the subdued alterations in the FDE output due to input disturbances. We present three theorems emphasizing the inherent resilience of the FROND paradigm:

- Theorem 1 [19] establishes the stability of the FROND model under small perturbations in the initial conditions of the FDE, which in our case, correspond to input feature changes.
- Theorem 2 [19] extends the discussion of robustness to include perturbations in the function that governs the system's dynamics. In the context of graph learning, such perturbations can be viewed as changes in the topology of the graph, which can occur due to the addition, deletion, or modification of edges.
- Finally, Theorem 3 provides important insights into how the choice of the fractional order β can influence the system's robustness, suggesting a pathway to further enhance FROND's resilience to perturbations.

Together, these results provide a strong theoretical basis for the robustness of FROND, setting the stage for its deployment in various practical applications.

4.1 Graph Fractional-order Differential Equations

Building upon the foundation of FDEs, recall that FROND incorporates the Caputo fractional-order time derivative into the model for feature evolution:

$$D_t^\beta \mathbf{X}(t) = \mathcal{F}(\mathbf{W}, \mathbf{X}(t)),\ \mathbf{X}(0) = \mathbf{X}_0,\ 0 < \beta \leq 1. \tag{11}$$

In (11), $D_t^\beta \mathbf{X}(t)$ denotes the fractional derivative of state $\mathbf{X}(t)$ concerning feature evolution time t, where $\beta \in (0, 1]$. This derivative instills memory effects and non-local interactions, enhancing the model's interpretative capacity. $\mathcal{F}(\mathbf{W}, \mathbf{X}(t))$ models the interaction between the weight matrix $\mathbf{W}$ and the state $\mathbf{X}(t)$ at time t. The model initializes at $\mathbf{X}(0) = \mathbf{X}_0$ and concludes at $\mathbf{X}(T)$ at time T.

One intrinsic characteristic of FROND is the long-memory property from fractional derivative. This property encapsulates the system's ability to "remember" its historical states over extended periods. This inherent memory effect contributes significantly to the robustness of the system, particularly when faced with perturbations. In contexts where disturbances or noise may influence the system, the extended memory of FROND can act as a buffer, absorbing and integrating these disturbances over time, thereby reducing their immediate impact. This attribute directly ties the long-memory feature to the enhanced robustness against perturbations, making FROND especially valuable in applications requiring stability against external disruptions.

4.2 Robustness of FROND under Perturbation

In this subsection, we delve into a theoretical analysis of model output perturbations. We begin by highlighting two central theorems from [19], which provide bounds on output perturbations, grounded in the properties of the Mittag-Leffler function. Following this, we analyze the bound outlined in Theorem 3, shedding light on the notion that a smaller β contributes to enhanced robustness of the model, particularly when faced with perturbations in input node features and graph topology.

Theorem 1. *[19, Theorem 6.20] Let $\mathbf{X}(t)$ be the solution of the initial value problem* (11)*, and let $\tilde{\mathbf{X}}(t)$ be the solution of the initial value problem*

$$D_t^\beta \tilde{\mathbf{X}}(t) = \mathcal{F}(\mathbf{W}, \tilde{\mathbf{X}}(t)), \quad \tilde{\mathbf{X}}(0) = \tilde{\mathbf{X}}_0, \tag{12}$$

where $\varepsilon := \|\mathbf{X}_0 - \tilde{\mathbf{X}}_0\|$. Then, if ε is sufficiently small, there exists some $h > 0$ such that both the functions $\mathbf{X}$ and $\tilde{\mathbf{X}}$ are defined on $[0, h]$, and

$$\sup_{0 \leq t \leq h} \|\mathbf{X}(t) - \tilde{\mathbf{X}}(t)\| = \varepsilon E_\beta(Lh^\beta) \tag{13}$$

where L is the Lipschitz constant of $\mathcal{F}$.

Remark 2. *Theorem 1 underlines the stability of the FDE* (11)*, demonstrating that for small perturbations in the input feature, the discrepancy between the solutions of the original and perturbed FDE systems is bounded. In the context of our FROND model, this means that minor perturbations in the input features cause only limited changes in the model's output, thus contributing to the robustness of the model against feature noise.*

Theorem 2. *[19, Theorem 6.21] Let $\mathbf{X}(t)$ be the solution of the initial value problem* (11)*, and let $\tilde{\mathbf{X}}(t)$ be the solution of the initial value problem*

$$D_t^\beta \tilde{\mathbf{X}}(t) = \tilde{\mathcal{F}}(\tilde{\mathbf{W}}, \tilde{\mathbf{X}}(t)), \quad \tilde{\mathbf{X}}(0) = \mathbf{X}_0 \tag{14}$$

Moreover, let $\varepsilon := \|\mathcal{F}(\mathbf{W}, \mathbf{X}(t)) - \tilde{\mathcal{F}}(\tilde{\mathbf{W}}, \tilde{\mathbf{X}}(t))\|$. Then, if ε is sufficiently small, there exists some $h > 0$ such that both the functions $\tilde{\mathcal{F}}$ and $\mathcal{F}$ are defined on $[0, h]$, and

$$\sup_{0 \leq t \leq h} \|\mathbf{X}(t) - \tilde{\mathbf{X}}(t)\| = \varepsilon E_\beta(Lh^\beta), \tag{15}$$

where L is the Lipschitz constant of $\mathcal{F}$.

Remark 3. *Theorem 2 affirms the robustness of fractional differential equations to perturbations in the function that defines the system dynamics. In the setting of FROND, this relates to the model's resilience against changes in the graph structure, thereby reinforcing the model's reliability when dealing with dynamic graph data or when the graph's structure is subject to uncertainty. These perturbations can be interpreted as topological changes in the graph structure, such as edge additions, deletions, or modifications.*

Theorem 3. *Let $f(\beta) = E_\beta(LT^\beta)$. For any $\epsilon > 0$, if T is sufficiently large and $L < 1$, $f(\beta)$ is monotonically increasing on the interval $[\epsilon, 1]$.*

Remark 4. *See the supplementary material for the proof. Together with Theorems 1 and 2, Theorem 3 shows that the fractional order β of the FROND plays a crucial role in the model's robustness. With an appropriately chosen β, the model can reduce the discrepancy between the clean and perturbed states, thereby improving the robustness. Particularly, a smaller β is associated with a smaller discrepancy, signifying enhanced robustness of FROND against perturbations. Please refer to Fig. 3 for an illustration. The monotonicity suggests that a larger β culminates in a larger Lipschitz constant for the FROND solution at time T. Consequently, with a larger β, one can anticipate a larger perturbed output at time T when subjected to the same input/graph topology perturbations.*

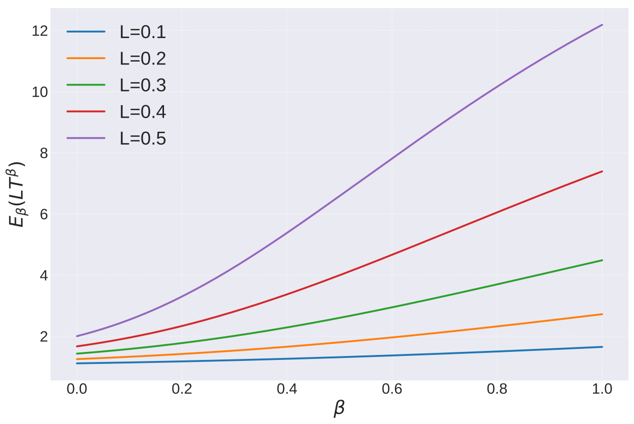


Figure 2: Plot of the Mittag-Leffler function $E_\beta(LT^\beta)$ against β with $T = 10$. Distinctively, for varying L, it displays monotonic increase over interval $[\epsilon, 1]$.

4.3 Algorithms

Our proposed approach enhances the robustness of integer-order graph neural diffusion models by introducing fractional-order derivatives into the model framework. Specifically, we extend three prominent graph neural diffusion models, GRAND, GraphBel, and GraphCON through this method.

We upgrade the GRAND framework with a fractional-order derivative, resulting in the Fractional-GRAND (F-GRAND) model. The F-GRAND formulation is as follows:

$$D_t^\beta \mathbf{X}(t) = (\mathbf{A}(\mathbf{X}(t)) - \mathbf{I})\mathbf{X}(t). \tag{16}$$

Following a similar approach, the GraphBel model is modified to incorporate a fractional-order derivative, resulting in the Fractional-GraphBel (F-GraphBel) model. This model is expressed as:

$$D_t^\beta \mathbf{X}(t) = (\mathbf{A_S}(\mathbf{X}(t)) \odot \mathbf{B_S}(\mathbf{X}(t)) - \Psi(\mathbf{X}(t)))\mathbf{X}(t). \tag{17}$$

Additionally, we introduce the Fractional-GraphCON (F-GraphCON) model, described by the following equations:

$$D_t^\beta \mathbf{Y}(t) = \sigma(\mathbf{F}_\theta(\mathbf{X}(t), t)) - \gamma \mathbf{X}(t) - \alpha \mathbf{Y}(t), \quad D_t^\beta \mathbf{X}(t) = \mathbf{Y}(t). \tag{18}$$

The order β of these fractional derivatives serves as a hyperparameter, introducing a new layer of flexibility to these models. This flexibility allows for adaptation to specific data characteristics, further enhancing the robustness of the learning process.

Table 1: Node classification accuracy (%) under **modification, poisoning, non-targeted** attack (Metattack) in **transductive** learning. The best and the second-best result for each criterion are highlighted in **red** and **blue** respectively.

Dataset	Ptb Rate(%)	F-GRAND	GRAND	F-GraphBel	GraphBel	F-GraphCON	GraphCON	GAT	GCN	RGCN	GCN-SVD
Cora	0	81.25±0.89	82.24±1.82	79.05±0.73	80.28±0.87	80.91±0.54	83.10±0.79	83.97±0.65	83.50±0.44	83.09±0.44	80.63±0.45
	5	78.84±0.57	78.97±0.49	76.10±0.74	77.70±0.66	77.80±0.44	77.90±1.14	80.44±0.74	76.55±0.79	77.42±0.39	78.39±0.54
	10	76.61±0.68	75.02±1.25	74.03±0.47	74.30±0.88	74.63±1.42	72.53±1.08	70.39±1.28	70.39±1.28	72.22±0.38	71.47±0.83
	15	73.42±0.97	71.43±1.09	73.01±0.75	72.14±0.69	73.01±0.78	69.83±0.68	65.10±0.71	65.10±0.71	66.82±0.39	66.69±1.18
	20	69.27±2.10	60.53±1.99	69.35±1.23	65.41±0.99	69.23±1.35	57.28±1.62	59.56±2.72	59.56±2.72	59.27±0.37	58.94±1.13
	25	64.47±1.83	55.26±2.14	67.63±0.93	62.31±1.13	65.27±1.33	53.17±1.52	47.53±1.96	47.53±1.96	50.51±0.78	52.06±1.19
Citeseer	0	71.37±1.34	71.50±1.10	68.90±1.15	69.46±1.15	71.49±0.71	70.48±1.18	73.26±0.83	71.96±0.55	71.20±0 83	70.65±0.32
	5	71.47±0.96	71.04±1.15	68.36±0.93	68.45±1.02	70.77±1.15	69.75±1.63	72.89±0.83	70.88±0.62	70.50±0.43	68.84±0.72
	10	69.76±0.71	68.88±0.60	67.22±1.52	66.72±1.31	69.54±0.82	67.40±1.78	70.63±0.48	67.55±0.89	67.71±0.30	68.87±0.62
	15	67.94±1.42	66.35±1.37	63.56±1.95	63.63±1.67	67.37±0.87	65.78±1.97	69.02±1.09	64.52±1.11	65.69±0.37	63.26±0.96
	20	64.18±0.93	58.71±1.42	63.38±0.96	58.90±0.84	66.52±0.68	56.79±1.46	61.04±1.52	62.03±3.49	62.49±1.22	58.55±1.09
	25	65.46±1.12	60.15±1.37	64.60±0.48	61.24±1.28	66.72±1.12	57.30±1.38	61.85±1.12	56.94±2.09	55.35±0.66	57.18±1.87
Pubmed	0	87.28±0.23	85.06±0.26	86.34±0.15	84.02±0.26	87.12±0.21	84.65±0.13	83.73±0.40	87.19±0.09	86.16±0.18	83.44±0.21
	5	87.05±0.17	84.11±0.30	86.17±0.12	83.91±0.26	86.72±0.23	83.06±0.22	78.00±0.44	83.09±0.13	81.08±0.20	83.41±0.15
	10	86.74±0.23	84.24±0.18	86.01±0.18	84.62±0.26	86.64±0.20	82.25±0.12	74.93±0.38	81.21±0.09	77.51±0.27	83.27±0.21
	15	86.51±0.14	83.74±0.34	85.92±0.13	84.83±0.20	86.40±0.14	81.26±0.33	71.13±0.51	78.66±0.12	73.91±0.25	83.10±0.18
	20	86.50±0.12	83.58±0.20	85.73±0.18	84.89±0.45	86.32±0.12	81.58±0.41	68.21±0.96	77.35±0.19	71.18±0.31	83.01±0.22
	25	86.47±0.15	83.66±0.25	86.11±0.30	85.07±0.15	86.15±0.26	80.75±0.32	65.41±0.77	75.50±0.17	67.95±0.15	82.72±0.18

5 Experiments

To empirically validate the robustness of FROND, we carry out a series of experiments where real-world graphs are subjected to various attack methods. The objective of these experiments is to showcase that FROND, even in the face of such adversarial perturbations, maintains stable

Table 2: Node classification accuracy (%) on graph **injection, evasion, non-targeted** attack in **inductive** learning. The best and the second-best result for each criterion are highlighted in red and blue respectively.

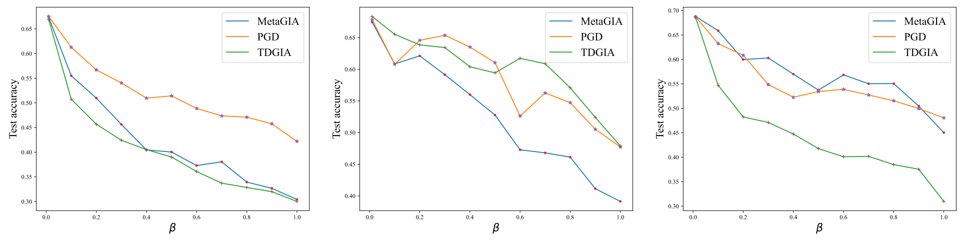
Dataset	Attack	F-GRAND	GRAND	F-GraphBel	GraphBel	F-GraphCON	GraphCON	GAT	GraphSAGE	GCN
Cora	*clean*	86.44±0.31	85.87±0.59	77.55±0.79	79.07±0.46	82.42±0.89	83.10±0.63	86.37±0.56	81.73±0.62	85.09±0.26
	PGD	56.38±6.39	36.80±1.86	69.50±2.83	63.93±3.88	56.70±4.36	48.38±2.44	38.82±2.48	42.10±0.55	40.11±0.70
	TDGIA	54.88±6.72	40.0±3.52	56.94±1.82	53.22±2.95	54.24±2.54	46.43±2.82	32.76±3.30	41.36±0.95	40.43±1.76
	MetaGIA	53.36±5.31	37.89±1.56	71.98±1.32	66.74±3.23	63.97±2.09	52.21±2.71	42.23±4.19	47.23±0.73	42.52±0.90
Citeseer	*clean*	71.91±0.43	72.52±0.73	71.09±0.30	74.75±0.28	73.50±0.43	72.07±0.93	73.10±0.39	72.68±0.58	74.48±0.66
	PGD	61.26±1.23	42.20±2.77	60.78±2.37	47.73±5.87	54.47±1.0	37.71±7.0	35.12±12.44	33.38±0.58	30.49±0.80
	TDGIA	50.74±1.20	30.02±1.33	65.52±0.55	47.88±1.83	54.71±1.69	30.93±3.00	28.64±4.05	29.05±1.45	28.88±2.07
	MetaGIA	55.50±1.72	30.42±1.87	60.85±1.88	39.13±1.19	48.82±3.27	29.09±2.01	30.17±2.71	32.95±0.66	32.74±1.00
Computers	*clean*	92.61±0.20	92.53±0.34	88.02±0.24	88.12±0.33	91.86±0.38	91.30±0.20	91.42±0.22	92.33±0.31	91.83±0.25
	PGD	89.90±1.33	70.45±11.03	87.60±0.33	87.38±0.37	91.36±0.74	81.28±7.99	38.82±5.53	39.54±1.49	33.43±0.21
	TDGIA	84.71±1.52	65.45±14.30	87.81±0.28	87.67±0.40	90.45±0.71	68.70±15.67	42.04±9.01	41.38±1.52	39.83±3.15
	MetaGIA	87.50±3.17	70.01±9.32	87.37±0.23	87.77±0.22	90.51±0.88	82.43±8.42	41.86±8.33	46.27±2.20	34.03±0.36
Pubmed	*clean*	88.39±0.47	88.44±0.34	89.51±0.12	88.18±1.89	90.30±0.11	88.09±0.32	87.41±1.73	88.71±0.37	88.46±0.20
	PGD	59.62±11.66	44.61±2.78	82.09±0.83	67.81±12.23	51.16±6.04	45.85±1.97	48.94±12.99	44.62±6.49	39.03±0.10
	TDGIA	54.31±2.38	46.26±1.32	82.72±0.47	68.66±10.64	55.50±4.03	45.57±2.02	47.56±3.11	47.61±0.91	42.64±1.41
	MetaGIA	61.62±9.05	44.07±2.11	79.16±0.87	64.64±9.70	52.03±5.53	45.81±2.81	44.75±2.53	42.39±0.53	40.42±0.17

performance in downstream tasks, without the need for any additional preprocessing steps to handle the perturbed data. For a comprehensive and fair evaluation, we perform two distinct evaluations: a poisoning Graph Modification Attack (GMA), where training occurs directly on the perturbed graph; and an evasion attack for Graph Injection Attack (GIA), taking place during the inference phase.

The core aim of this paper is to explore the robustness provided by the graph neural FDE model, asserting that fractional methods are inherently more robust than their integer-order dynamic system GNN counterparts. Thus, our comparisons mainly target foundational integer-order graph neural ODE models, minimally incorporating non-ODE-based models like GCN [48], GAT [49], and GraphSAGE [50]. Notably, fractional-order GNNs can also merge with other defense techniques such as adversarial training and preprocessing strategies, discussed further in the supplementary material.

5.1 GMA

Our experimental setup involves the execution of graph modification adversarial attacks employing the Metattack method [30]. Within the Metattack paradigm, the graph's adjacency matrix is perceived not just as a static structure but as a malleable hyperparameter. This perspective allows for attack optimization through meta-gradients to effectively address the inherent bilevel problem. For the sake of ensuring a consistent and unbiased comparative landscape, our experiments strictly conform to the attack parameters as outlined in the paper [11]. To achieve a comprehensive evaluation, we vary the perturbation rate, representing the proportion of edge modifications. We source the perturbed graph data from the comprehensive DeepRobust library [51]. The perturbation rate is adjusted in consistent increments of 5%, starting from an untouched graph (0%) and extending up to significant alterations at 25%. This approach provides insights into the effects and resilience of our model across a spectrum of adversarial conditions.



(a) F-GRAND (b) F-GraphBel (c) F-GraphCON

Figure 3: The impact of β on the robust accuracy.

5.2 GIA

As elucidated in [38], GIA presents a considerably potent challenge to GNNs because of its ability to introduce new nodes and establish new edges within the original graph. Executing a GIA entails a two-step process: the injection of nodes and the subsequent update of features. During the node injection phase, new edges are established for the inserted nodes, driven by either gradient data or heuristic methods. Drawing inspiration from the methods proposed in [38], we have incorporated three distinct GIA techniques: PGD-GIA, TDGIA, and MetaGIA. The PGD-GIA method predominantly relies on a randomized approach for node injection. Once these nodes are in place, their features are meticulously curated using the Projected Gradient Descent (PGD) algorithm [52]. The Topological Deficiency Graph Injection Attack (TDGIA) [35] exploits inherent topological weaknesses in graph structures. This approach harnesses these vulnerabilities to guide edge creation, optimizing a specific loss function to devise suitable features. MetaGIA [38], however, presents a more comprehensive approach. By considering both the adjacency matrix and node features as hyperparameters awaiting optimization, it champions an iterative strategy. This strategy continually refines the adjacency matrix and node features, leaning heavily on gradient information to guide these refinements.

We conduct inductive learning for GIA in line with the data partitioning approach of the GRB framework [53], allocating 60% for training, 10% for validation, and 20% for testing purposes. To maintain a balanced attack landscape, we pre-process the data using methods from [53], which involve excluding the 5% of nodes with the lowest degrees (more susceptible to attacks) and the 5% with the highest degrees (more resistant to attacks).

5.3 Results

Table 1 presents the results of GMA for transductive learning. As can be observed from the table, our proposed fractional-order methods outperform the original GRAND, GraphBel, and GraphCON in terms of robustness accuracy. These results validate and resonate with our theoretical findings, as discussed in Theorem 3. Notably, these empirical observations underscore the capability of the FROND paradigm in enhancing a system's resilience, particularly when faced with input perturbations. The GIA results are presented in Table 2. We note that the fractional-order approach significantly improves post-attack accuracy relative to its integer-order graph neural ODE counterparts. Among these neural ODE models, [17] demonstrated that GraphBel possesses superior robustness, which is further amplified by our fractional-order differential technique.

5.4 Ablation Study

5.4.1 Influence of β

We assess the robustness accuracy of our fractional-order method across varying β values. The findings are depicted in Fig. 3. A discernible trend emerges: as β increases, the accuracy under the three GIA methods diminishes. This observation aligns with our theoretical insights presented in Theorem 3.

5.4.2 Model Complexity

A comparison of inference times between our models and the baseline models is presented in Table 3. The results indicate that fractional-based models have similar inference times to graph neural ODE models. Notably, fractional-based models maintain the same training parameters as integer ODE models, avoiding any extra parameters. These findings highlight the efficiency and flexibility of our approach.

Table 3: Inference time of models on the Cora dataset: integral time $T = 10$ and step size of 1.

Model	F-GRAND	GRAND	F-GraphBel	GraphBel	F-GraphCON	GraphCON
Inf. Time(s)	18.74	16.40	64.90	78.51	21.33	18.27

6 Conclusion

In this paper, we have undertaken a comprehensive exploration of robustness against adversarial attacks within the framework of the graph neural FDE model, leading to substantial insights. Our investigation has yielded significant revelations, notably demonstrating the heightened robustness of the graph neural FDE model when compared to existing graph neural ODE models. Moreover, our work has contributed theoretical clarity, shedding light on the underlying reasons behind the heightened robustness of the graph neural FDE models in contrast to the graph neural ODE counterparts.

7 Acknowledgments and Disclosure of Funding

This research is supported by the Singapore Ministry of Education Academic Research Fund Tier 2 grant MOE-T2EP20220-0002, and the National Research Foundation, Singapore and Infocomm Media Development Authority under its Future Communications Research and Development Programme. To improve the readability, parts of this paper have been grammatically revised using ChatGPT [54].

References

[1] C. Huang, H. Xu, Y. Xu, P. Dai, L. Xia, M. Lu, L. Bo, H. Xing, X. Lai, and Y. Ye, "Knowledge-aware coupled graph neural network for social recommendation," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 5, 2021, pp. 4115–4122.

[2] Z. Guo, B. Nan, Y. Tian, O. Wiest, C. Zhang, and N. V. Chawla, "Graph-based molecular representation learning," *arXiv preprint arXiv:2207.04869*, 2022.

[3] H. Dai, H. Li, T. Tian, X. Huang, L. Wang, J. Zhu, and L. Song, "Adversarial attack on graph structured data," in *International conference on machine learning*. PMLR, 2018, pp. 1115–1124.

[4] J. Ma, S. Ding, and Q. Mei, "Towards more practical adversarial attacks on graph neural networks," *Advances in neural information processing systems*, vol. 33, pp. 4756–4766, 2020.

[5] D. Zügner, A. Akbarnejad, and S. Günnemann, "Adversarial attacks on neural networks for graph data," in *Proc. Int. Conf. Knowl. Discovery Data Mining*, 2018.

[6] E. Dai, T. Zhao, H. Zhu, J. Xu, Z. Guo, H. Liu, J. Tang, and S. Wang, "A comprehensive survey on trustworthy graph neural networks: Privacy, robustness, fairness, and explainability," *arXiv preprint arXiv:2204.08570*, 2022.

[7] D. Wang, J. Lin, P. Cui, Q. Jia, Z. Wang, Y. Fang, Q. Yu, J. Zhou, S. Yang, and Y. Qi, "A semi-supervised graph attentive network for financial fraud detection," in *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2019, pp. 598–607.

[8] C. Deng, X. Li, Z. Feng, and Z. Zhang, "Garnet: Reduced-rank topology learning for robust and scalable graph neural networks," in *Learning on Graphs Conference*. PMLR, 2022, pp. 3–1.

[9] X. Zhang and M. Zitnik, "Gnnguard: Defending graph neural networks against adversarial attacks," *Advances Neural Inf. Process. Syst.*, vol. 33, pp. 9263–9275, 2020.

[10] D. Zhu, Z. Zhang, P. Cui, and W. Zhu, "Robust graph convolutional networks against adversarial attacks," in *Proc. Int. Conf. Knowl. Discovery Data Mining*, 2019, pp. 1399–1407.

[11] W. Jin, Y. Ma, X. Liu, X. Tang, S. Wang, and J. Tang, "Graph structure learning for robust graph neural networks," in *Proc. Int. Conf. Knowl. Discovery Data Mining*, 2020, pp. 66–74.

[12] L.-P. Xhonneux, M. Qu, and J. Tang, "Continuous graph neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 10 432–10 441.

[13] B. P. Chamberlain, J. Rowbottom, M. Goronova, S. Webb, E. Rossi, and M. M. Bronstein, "Grand: Graph neural diffusion," in *Proc. Int. Conf. Mach. Learn.*, 2021.

[14] M. Thorpe, T. M. Nguyen, H. Xia, T. Strohmer, A. Bertozzi, S. Osher, and B. Wang, "Grand++: Graph neural diffusion with a source term," in *Proc. Int. Conf. Learn. Representations*, 2021.

[15] T. K. Rusch, B. P. Chamberlain, J. Rowbottom, S. Mishra, and M. M. Bronstein, "Graph-coupled oscillator networks," in *Proc. Int. Conf. Mach. Learn.*, 2022.

[16] K. Zhao, Q. Kang, Y. Song, R. She, S. Wang, and W. P. Tay, "Graph neural convection-diffusion with heterophily," in *Proc. International Joint Conference on Artificial Intelligence*, Macao, China, Aug 2023.

[17] Y. Song, Q. Kang, S. Wang, K. Zhao, and W. P. Tay, "On the robustness of graph neural diffusion to topology perturbations," in *Advances Neural Inf. Process. Syst.*, New Orleans, USA, Nov. 2022.

[18] N. Sochen, R. Kimmel, and R. Malladi, "A general framework for low level vision," *IEEE Trans. Image Process.*, vol. 7, no. 3, pp. 310–318, 1998.

[19] K. Diethelm and N. Ford, "The analysis of fractional differential equations," *Lect. Notes Math*, vol. 2004, pp. 3–12, 2010.

[20] Anonymous, "Fractional-order graph neural diffusion," in *Submitted paper*, 2023.

[21] V. E. Tarasov, *Fractional dynamics: applications of fractional calculus to dynamics of particles, fields and media.* Springer Science & Business Media, 2011.

[22] B. P. Chamberlain, J. Rowbottom, D. Eynard, F. Di Giovanni, D. Xiaowen, and M. M. Bronstein, "Beltrami flow and neural diffusion on graphs," in *Advances Neural Inf. Process. Syst.*, 2021.

[23] C. Bodnar, F. Di Giovanni, B. Chamberlain, P. Liò, and M. Bronstein, "Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns," *Advances in Neural Information Processing Systems*, vol. 35, pp. 18 527–18 541, 2022.

[24] J. Choi, S. Hong, N. Park, and S.-B. Cho, "Gread: Graph neural reaction-diffusion networks," in *ICML*, 2023.

[25] J. Chen, Y. Wu, X. Xu, Y. Chen, H. Zheng, and Q. Xuan, "Fast gradient attack on network embedding," *ArXiv*, 2018.

[26] M. Waniek, T. P. Michalak, M. J. Wooldridge, and T. Rahwan, "Hiding individuals and communities in a social network," *Nature Human Behaviour*, vol. 2, no. 1, pp. 139–147, 2018.

[27] J. Du, S. Zhang, G. Wu, J. M. F. Moura, and S. Kar, "Topology adaptive graph convolutional networks," *ArXiv*, vol. abs/1710.10370, 2017.

[28] Y. Ma, S. Wang, T. Derr, L. Wu, and J. Tang, "Graph adversarial attack via rewiring," in *Proc. Int. Conf. Knowl. Discovery Data Mining*, 2021, p. 1161–1169.

[29] S. Geisler, T. Schmidt, H. Şirin, D. Zügner, A. Bojchevski, and S. Günnemann, "Robustness of graph neural networks at scale," *Advances Neural Inf. Process. Syst.*, vol. 34, pp. 7637–7649, 2021.

[30] D. Zügner and S. Günnemann, "Adversarial attacks on graph neural networks via meta learning," in *Proc. Int. Conf. Learn. Representations*, 2019.

[31] J. Ma, J. Deng, and Q. Mei, "Adversarial attack on graph neural networks as an influence maximization problem," in *Proc. of the 15th ACM Int. Conf. Web Search and Data Min.*, 2022, pp. 675–685.

[32] B. Finkelshtein, C. Baskin, E. Zheltonozhskii, and U. Alon, "Single-node attacks for fooling graph neural networks," *Neurocomputing*, vol. 513, pp. 1–12, 2022.

[33] J. Wang, M. Luo, F. Suya, J. Li, Z. Yang, and Q. Zheng, "Scalable attack on graph data by injecting vicious nodes," *Data Mining Knowl. Discovery*, pp. 1 – 27, 2020.

[34] Q. Zheng, Y. Fei, Y. Li, Q. Liu, M. Hu, and Q. Sun. Kdd cup 2020 ml track 2 adversarial attacks and defense on academic graph 1st place solution. Accessed: May 1, 2022. [Online]. Available: https://github.com/Stanislas0/KDD_CUP_2020_MLTrack2_SPEIT

[35] X. Zou, Q. Zheng, Y. Dong, X. Guan, E. Kharlamov, J. Lu, and J. Tang, "Tdgia: Effective injection attacks on graph neural networks," in *Proc. Int. Conf. Knowl. Discovery Data Mining*, 2021, p. 2461–2471.

[36] Y. Sun, S. Wang, X. Tang, T.-Y. Hsieh, and V. Honavar, "Adversarial attacks on graph neural networks via node injections: A hierarchical reinforcement learning approach," in *Proc. Web Conf.*, 2020, p. 673–683.

[37] H. Hussain, M. Cao, S. Sikdar, D. Helic, E. Lex, M. Strohmaier, and R. Kern, "Adversarial inter-group link injection degrades the fairness of graph neural networks," *arXiv preprint arXiv:2209.05957*, 2022.

[38] Y. Chen, H. Yang, Y. Zhang, K. Ma, T. Liu, B. Han, and J. Cheng, "Understanding and improving graph injection attack by promoting unnoticeability," in *Proc. Int. Conf. Learn. Representations*, 2022.

[39] N. Entezari, S. A. Al-Sayouri, A. Darvishzadeh, and E. E. Papalexakis, "All you need is low (rank): Defending against adversarial attacks on graphs," in *Proc. Int. Conf. Web Search Data Mining*, 2020, p. 169–177.

[40] S. Geisler, T. Schmidt, H. Şirin, D. Zügner, A. Bojchevski, and S. Günnemann, "Robustness of graph neural networks at scale," in *Neural Information Processing Systems, NeurIPS*, 2021.

[41] T. K. Rusch, B. Chamberlain, J. Rowbottom, S. Mishra, and M. Bronstein, "Graph-coupled oscillator networks," in *International Conference on Machine Learning*. PMLR, 2022, pp. 18 888–18 909.

[42] R. T. Q. Chen, "torchdiffeq," 2018. [Online]. Available: https://github.com/rtqichen/torchdiffeq

[43] M. Caputo, "Linear models of dissipation whose q is almost frequency independent—ii," *Geophysical Journal International*, vol. 13, no. 5, pp. 529–539, 1967.

[44] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations," in *Advances Neural Inf. Process. Syst.*, 2018.

[45] A. Quaglino, M. Gallieri, J. Masci, and J. Koutník, "Snode: Spectral discretization of neural odes for system identification," in *Proc. Int. Conf. Learn. Representations*, 2019.

[46] H. Yan, J. Du, V. Y. Tan, and J. Feng, "On robustness of neural ordinary differential equations," in *Advances Neural Inf. Process. Syst.*, 2018, pp. 1–13.

[47] K. Diethelm, N. J. Ford, and A. D. Freed, "Detailed error analysis for a fractional adams method," *Numer. Algorithms*, vol. 36, pp. 31–52, 2004.

[48] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017.

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

[50] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances Neural Inf. Process. Syst.*, 2017.

[51] Y. Li, W. Jin, H. Xu, and J. Tang, "Deeprobust: A pytorch library for adversarial attacks and defenses," *arXiv preprint arXiv:2005.06149*, 2020.

[52] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[53] Q. Zheng, X. Zou, Y. Dong, Y. Cen, D. Yin, J. Xu, Y. Yang, and J. Tang, "Graph robustness benchmark: Benchmarking the adversarial robustness of graph machine learning," *Advances Neural Inf. Process. Syst. Track Datasets Benchmarks*, 2021.

[54] OpenAI, "Chatgpt-4," 2022, available at: https://www.openai.com (Accessed: 26 September 2023).

[55] K. Diethelm, *The analysis of fractional differential equations: an application-oriented exposition using differential operators of Caputo type*. Lect. Notes Math, 2010, vol. 2004.

[56] A. M. Cohen, *Inversion Formulae and Practical Results*. Boston, MA: Springer US, 2007.

[57] E. A. Coddington and N. Levinson, *Theory of ordinary differential equations*. McGraw-Hill, 1955.

[58] P. Hartman, *Ordinary differential equations*. SIAM, 2002.

[59] S. B. Yuste and L. Acedo, "An explicit finite difference method and a new von neumann-type stability analysis for fractional diffusion equations," *SIAM Journal on Numerical Analysis*, vol. 42, no. 5, pp. 1862–1874, 2005.

[60] B. D. Coleman and W. Noll, "Foundations of linear viscoelasticity," *Reviews of modern physics*, vol. 33, no. 2, p. 239, 1961.

[61] R. Almeida, N. R. Bastos, and M. T. T. Monteiro, "Modeling some real phenomena by fractional differential equations," *Mathematical Methods in the Applied Sciences*, vol. 39, no. 16, pp. 4846–4855, 2016.

[62] I. Podlubny, "Fractional-order systems and fractional-order controllers," *Institute of Experimental Physics, Slovak Academy of Sciences, Kosice*, vol. 12, no. 3, pp. 1–18, 1994.

[63] J. T. Machado, V. Kiryakova, and F. Mainardi, "Recent history of fractional calculus," *Communications in nonlinear science and numerical simulation*, vol. 16, no. 3, pp. 1140–1153, 2011.

[64] E. Scalas, R. Gorenflo, and F. Mainardi, "Fractional calculus and continuous-time finance," *Physica A: Statistical Mechanics and its Applications*, vol. 284, no. 1-4, pp. 376–384, 2000.

[65] R. Nigmatullin, "The realization of the generalized transfer equation in a medium with fractal geometry," *Physica status solidi (b)*, vol. 133, no. 1, pp. 425–430, 1986.

[66] B. B. Mandelbrot and B. B. Mandelbrot, *The fractal geometry of nature*. WH freeman New York, 1982, vol. 1.

[67] C. Ionescu, A. Lopes, D. Copot, J. T. Machado, and J. H. Bates, "The role of fractional calculus in modeling biological phenomena: A review," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 51, pp. 141–159, 2017.

[68] K. Diethelm and N. J. Ford, "Analysis of fractional differential equations," *J. Math. Anal. Appl.*, vol. 265, no. 2, pp. 229–248, 2002.

[69] D. Krapf, "Mechanisms underlying anomalous diffusion in the plasma membrane," *Current Topics Membranes*, vol. 75, pp. 167–207, 2015.

[70] Z. Liu, Y. Wang, Y. Luo, and C. Luo, "A regularized graph neural network based on approximate fractional order gradients," *Mathematics*, vol. 10, no. 8, p. 1320, 2022.

[71] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[72] H. Antil, R. Khatri, R. Löhner, and D. Verma, "Fractional deep neural network via constrained optimization," *Mach. Learn.: Sci. Tech.*, vol. 2, no. 1, p. 015003, 2020.

[73] G. Pang, L. Lu, and G. E. Karniadakis, "fpinns: Fractional physics-informed neural networks," *SIAM J. Sci. Comput.*, vol. 41, no. 4, pp. A2603–A2626, 2019.

[74] L. Guo, H. Wu, X. Yu, and T. Zhou, "Monte carlo fpinns: Deep learning method for forward and inverse problems involving high dimensional fractional partial differential equations," *Comput. Methods Appl. Mech. Eng.*, vol. 400, p. 115523, 2022.

[75] S. Wang, H. Zhang, and X. Jiang, "Fractional physics-informed neural networks for time-fractional phase field models," *Nonlinear Dyn.*, vol. 110, no. 3, pp. 2715–2739, 2022.

[76] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

This supplementary material complements the main body of our paper, providing additional details and supporting evidence for the assertions made therein. The structure of this document is as follows:

1. A comprehensive background on fractional calculus is detailed in Appendix A.
2. An expanded discussion on related work is provided in Appendix B.
3. Details of the fractional differential equation (FDE) solver used in our paper can be found in Appendix C.
4. Theoretical results from the main paper are rigorously proven in Appendix D.
5. Additional experiments, ablation studies, and dataset specifics are elaborated in Appendix E.

A Review of Caputo Fractional Calculus

We appreciate the need for a more accessible explanation of the Caputo time-fractional derivative and its derivation, as the mathematical intricacies may be challenging for some readers in the GNN community. To address this, we are providing a more comprehensive background in this section.

A.1 Caputo Fractional Derivative and Its Compatibility of Integer-order Derivative

The Caputo fractional derivative of a function $f(t)$ over an interval $[0, b]$, of a general positive order $\beta \in (0, \infty)$, is defined as follows:

$$D_t^\beta f(t) = \frac{1}{\Gamma(\lceil \beta \rceil - \beta)} \int_0^t (t-\tau)^{\lceil \beta \rceil - \beta - 1} f^{(\lceil \beta \rceil)}(\tau) \mathrm{d}\tau, \tag{19}$$

Here, $\lceil \beta \rceil$ is the smallest integer greater than or equal to β, $\Gamma(\cdot)$ symbolizes the gamma function, and $f^{(\lceil \beta \rceil)}(\tau)$ signifies the $\lceil \beta \rceil$-order derivative of f. Within this definition, it is presumed that $f^{(\lceil \beta \rceil)} \in L^1[0, b]$, i.e., $f^{(\lceil \beta \rceil)}$ is Lebesgue integrable, to ensure the well-defined nature of $D_t^\beta f(t)$ as per (19) [55]. When addressing a vector-valued function, the Caputo fractional derivative is defined on a component-by-component basis for each dimension, similar to the integer-order derivative. For ease of exposition, we explicitly handle the scalar case here, although all following results can be generalized to vector-valued functions. The Laplace transform for a general order $\beta \in (0, \infty)$ is presented in Theorem 7.1 [55] as:

$$\mathcal{L} D_t^\beta f(s) = s^\beta \mathcal{L} f(s) - \sum_{k=1}^{\lceil \beta \rceil} s^{\beta - k} f^{(k-1)}(0). \tag{20}$$

where we assume that $\mathcal{L} f$ exists on $[s_0, \infty)$ for some $s_0 \in \mathbb{R}$. In contrast, for the integer-order derivative $f^{(\beta)}$ when β is a positive integer, we also have the formulation (20), with the only difference being the range of β. Therefore, as β approaches some integer, the Laplace transform of the Caputo fractional derivative converges to the Laplace transform of the traditional integer-order derivative. *As a result, we can conclude that the Caputo fractional derivative operator generalizes the traditional integer-order derivative since their Laplace transforms coincide when β takes an integer value.* Furthermore, the inverse Laplace transform indicates the uniquely determined $D_t^\beta = f^{(\beta)}$ (in the sense of almost everywhere [56]).

Under specific reasonable conditions, we can directly present this generalization as follows. We suppose $f^{(\lceil \beta \rceil)}(t)$ (19) is continuously differentiable. In this context, integration by parts can be utilized to demonstrate that

$$\begin{aligned} D_t^\beta f(t) &= \frac{1}{\Gamma(\lceil\beta\rceil-\beta)}\Bigg(-\left[f^{(\lceil\beta\rceil)}(\tau)\frac{(t-\tau)^{\lceil\beta\rceil-\beta}}{\lceil\beta\rceil-\beta} \right]\Bigg|_0^t \\ &\qquad + \int_0^t f^{(\lceil\beta\rceil+1)}(\tau)\frac{(t-\tau)^{\lceil\beta\rceil-\beta}}{\lceil\beta\rceil-\beta}\mathrm{d}\tau \Bigg) \\ &= \frac{t^{\lceil\beta\rceil-\beta} f^{(\lceil\beta\rceil)}(0)}{\Gamma(\lceil\beta\rceil-\beta+1)} + \frac{1}{\Gamma(\lceil\beta\rceil-\beta+1)} \\ &\qquad \times \int_0^t (t-\tau)^{\lceil\beta\rceil-\beta} f^{(\lceil\beta\rceil+1)}(\tau)\mathrm{d}\tau \end{aligned} \tag{21}$$

When $\beta \to \lceil\beta\rceil$, we get the following

$$\begin{aligned} \lim_{\beta\to\lceil\beta\rceil} D_t^\beta f(t) &= f^{(\lceil\beta\rceil)}(0) + \int_0^t f^{(\lceil\beta\rceil+1)}(\tau)\mathrm{d}\tau \\ &= f^{(\lceil\beta\rceil)}(0) + f^{(\lceil\beta\rceil)}(t) - f^{(\lceil\beta\rceil)}(0) \\ &= f^{(\lceil\beta\rceil)}(t) \end{aligned} \tag{22}$$

In parallel to the integer-order derivative, given certain conditions ([55][Lemma 3.13]), the Caputo fractional derivative possesses the semigroup property as illustrated in [55][Lemma 3.13]:

$$D_t^\varepsilon D_t^n f = D_t^{n+\varepsilon} f. \tag{23}$$

The Caputo fractional derivative also exhibits linearity, but does not adhere to the same Leibniz and chain rules as its integer counterpart. As such properties are not utilized in our work, we refer interested readers to [55][Theorem 3.17 and Remark 3.5.]. We believe the above explanation facilitates understanding the relation between the Caputo derivative and its generalization of the integer-order derivative.

A.2 Caputo Fractional Differential Equations

In this section, we first discuss the initial conditions for fractional differential equations (FDEs) under the Caputo definition. Following this, we present the precise conditions for the existence and uniqueness of the solution to the FDEs. As we will see, these conditions closely align with those of ordinary differential equations, conditions which are widely assumed by all graph neural ODE works such as the recent contributions like GRAND [13], GraphCON [41], and GraphBel [17]. In short, all these graph neural ODE works can be seamlessly extended to fractional dynamics.

We first give the Caputo fractional taylor expansion: [55][Theorem 3.8.] Assume that $n \geq 0, m = \lceil n\rceil$, and f has absolutely continuous $(m-1)$-st derivative. Then

$$f(t) = \sum_{k=0}^{m-1} \frac{D_t^k f(0)}{k!} t^k + J^n D_t^n f(t). \tag{24}$$

Note the order in D_t^k here is still an integer. If we compare it with the classical integer-order Taylor expansion, it becomes evident that the Caputo derivative closely resembles the classical integer-order derivative in terms of Taylor expansion. This fact will influence the initial conditions for differential equations, as introduced in the following.

Assume that e is a given function with the property that $e = D_t^\beta g$, and then the solution of the Caputo differential equation is the form

$$D_t^\beta f = g \tag{25}$$

is given by

$$f(x) = e(x) + \sum_{j=1}^{\lceil\beta\rceil} c_j (x-a)^{\lceil\beta\rceil-j} \tag{26}$$

once more, with c_j^* as arbitrary constants. Thus, to obtain a unique solution, it is most logical to prescribe the values of integer order derivatives $f(0), D_t^1 f(0), \ldots, D_t^{\lceil\beta\rceil-1} f(0)$ in the Caputo setting, *mirroring the traditional ordinary differential equation.*

Next, we delve into a general Caputo fractional differential equation, presented as follows:

$$D_t^\beta y(t) = g(t, y(t)) \tag{27}$$

conjoined with suitable initial conditions. As hinted in (25) and (26), the initial conditions take the form:

$$D_t^k y(0) = y_0^{(k)}, \quad k = 0, 1, \ldots, \lceil\beta\rceil - 1. \tag{28}$$

Caputo existence and uniqueness theorem: [55][Theorem 6.8] Let $y_0^{(0)}, \ldots, y_0^{(m-1)} \in \mathbb{R}$ and $h^* > 0$. Define the set $G := [0, h^*] \times \mathbb{R}$ and let the function $g : G \to \mathbb{R}$ be continuous and fulfill a *Lipschitz condition* with respect to the second variable, i.e.

$$|g(x, y_1) - g(x, y_2)| \leq L |y_1 - y_2| \tag{29}$$

with some constant $L > 0$ independent of x, y_1, and y_2. Then there uniquely exists function $y \in C[0, h^*]$ solving the initial value problem (27) and (28).

For a point of reference, we also provide the well-known Picard–Lindelöf uniqueness theorem for ordinary differential equations.
Picard–Lindelöf theorem: [57, 58] Let $D \subseteq \mathbb{R} \times \mathbb{R}^n$ be a closed rectangle with $(t_0, y_0) \in \operatorname{int} D$, the interior of D. Let $g : D \to \mathbb{R}^n$ be a function that is continuous in t and *Lipschitz continuous* in y. Then, there exists some $\varepsilon > 0$ such that the initial value problem

$$y'(t) = g(t, y(t)), \quad y(t_0) = y_0.$$

has a unique solution $y(t)$ on the interval $[t_0, t_0 + \varepsilon]$.

This allows us to draw parallels between the existence and uniqueness theorem of the Caputo fractional differential equation and its integer-order ordinary differential equation equivalent. We also remind readers that standard neural networks, as compositions of linear maps and pointwise non-linear activation functions with bounded derivatives (such as fully-connected and convolutional networks), satisfy global Lipschitz continuity with respect to the input. For attention neural networks, which are compositions of softmax and matrix multiplication, we observe local Lipschitz continuity. To see this, suppose $\mathbf{v} = \operatorname{softmax}(\mathbf{u}) \in \mathbb{R}^{n\times 1}$. Then

$$\begin{aligned}\frac{\partial \mathbf{v}}{\partial \mathbf{u}} &= \operatorname{diag}(\mathbf{v}) - \mathbf{v}\mathbf{v}^\top \\ &= \begin{bmatrix} v_1(1-v_1) & -v_1 v_2 & \ldots & -v_1 v_n \\ -v_2 v_1 & v_2(1-v_2) & \ldots & -v_2 v_n \\ \vdots & \vdots & \ddots & \vdots \\ -v_n v_1 & -v_n v_2 & \ldots & v_n(1-v_n) \end{bmatrix}\end{aligned}$$

For bounded input, we always have a bounded Jacobian. All the graph neural ODE works, such as recent contributions like GRAND [13], GraphCON [41], and GraphBel [17], safely assume the uniqueness of the solution to ODEs.

B More Discussion of Related Work: Fractional Calculus and Deep Learning

In this section, we further discussion the applications of fractional calculus, with a particular emphasis on its implications in deep learning.

Recently, fractional calculus has garnered significant attention due to its myriad applications spanning diverse areas. Key domains where fractional calculus has demonstrated potential include numerical analysis [59], viscoelastic materials [60], population dynamics [61], control theory [62], signal processing [63], financial mathematics [64], and especially in characterizing porous and fractal systems [65–67]. Within these arenas, fractional-order differential equations have emerged as an enhanced alternative to their integer-ordered counterparts, serving as a robust mathematical tool for various system analyses [68]. For instance, fractional calculus has been pivotal in diffusion process studies, elucidating phenomena from protein diffusion in cellular structures [69] to complex biological processes [67].

In the landscape of deep learning, [70] introduced an innovative approach for GNN parameter optimization via fractional derivatives. This deviates significantly from the traditional use of integer-order derivatives in optimization algorithms such as SGD or Adam [71]. However, the crux of their approach is fundamentally different from ours. While they focus on leveraging fractional derivatives for gradient optimization, our emphasis is on the fractional-derivative evolution of node embeddings. In another vein, [72] draws from fractional calculus, specifically the L1 approximation of the Captou fractional derivative, to design a densely connected neural network. This design seeks to effectively manage non-smooth data and counter the vanishing gradient problem. Though our work orbits the same realm, our novelty lies in infusing fractional calculus into Graph ODE models, concentrating on the utility of fractional derivatives for evolving node embeddings, and highlighting its affinity with non-Markovian dynamic processes.

From the vantage of physics-informed machine learning, there exists a research trajectory dedicated to the formulation of neural networks anchored in physical principles, specifically tailored for solving fractional PDEs. A trailblazing contribution in this sphere is the Fractional Physics Informed Neural Networks (fPINNs) [73]. Subsequent explorations, including [74, 75], have expanded in this trajectory. It's pivotal to underline that these endeavors are distinctly different from our proposed methodology.

C Numerical Solvers for FDEs

In this section, we present further details about how to solve FDEs using the fractional Adams–Bashforth–Moulton method solvers from [47]. The predictor y_{k+1}^P is expressed as:

$$y_{k+1}^P = \sum_{j=0}^{\lceil\beta\rceil-1} \frac{t_{k+1}^j}{j!} y_0^{(j)} + \frac{1}{\Gamma(\beta)} \sum_{j=0}^{k} b_{j,k+1} f(t_j, y_j). \tag{30}$$

Here, k represents the current iteration or time step index in the discretization process. h is the step size or time interval between successive approximations with $t_j = hj$ and $\lceil\cdot\rceil$ represents the ceiling function, when $0 < \beta \leq 1$, we have $\lceil\beta\rceil = 1$. The coefficients $b_{j,k+1}$ are defined as follows:

$$b_{j,k+1} = \frac{h^\beta}{\beta}\left((k+1-j)^\beta - (k-j)^\beta\right), \tag{31}$$

Leveraging this prediction, a corrector term can be formulated to enhance the solver's numerical accuracy. This can be viewed as the fractional counterpart of the traditional one-step Adams–Moulton method. However, we do not employ this additional corrector term in our paper. We reserve the examination of the corrector solution and its impact on FROND for future work.

D Proof

In this section, we prove Theorem 3. For clarity, we restate the theorem for the reader's reference.

Theorem 4. *Let $f(\beta) = E_\beta(LT^\beta)$. For any $\varepsilon > 0$, we have that when T is large enough and $L < 1$, $f(\beta)$ is monotonically increasing on the interval $[\varepsilon, 1]$.*

Proof. First, we cite the equation from [62, eq.(140)]:

$$\begin{aligned} E_{\beta,\alpha}(z) = {} & \frac{1}{\beta} z^{(1-\alpha)/\beta} \exp\left(z^{1/\beta}\right) - \sum_{k=1}^{p} \frac{z^{-k}}{\Gamma(\alpha - \beta k)} \\ & + \frac{1}{2\pi\beta i z^p} \int_{\gamma(1,\varphi)} \exp\left(\zeta^{1/\beta}\right) \zeta^{(1-\alpha)/\beta + p} d\zeta, \quad (|\arg(z)| \leq \mu, |z| > 1). \end{aligned} \tag{32}$$

Here $E_{\beta,\alpha}(z)$ denotes the generalized two-parameter Mittag-Leffler function, defined as $E_{\beta,\alpha}(z) := \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(\beta k + \alpha)}$. Its connection to the one-parameter Mittag-Leffler function, as given in Definition 1, is captured by $E_\beta(z) = E_{\beta,1}(z)$. Note that $z \in \mathbb{C}$ lies in the complex plane. The integral contour $\gamma(1, \varphi)$ is detailed in [62, Figure 1.4, Sec 1.1.6]. The parameter μ is selected such that $\frac{\pi\beta}{2} < \mu < \min\{\pi, \pi\beta\}$, while and φ is chosen to satisfy $\frac{\pi\beta}{2} < \mu < \varphi \leq \min\{\pi, \pi\beta\}$. For a thorough understanding of this integral contour, we direct readers to the aforementioned reference.

For $\alpha = 1$ and $0 < \beta \leq 1$, we have that for any positive integer p,

$$\begin{aligned} E_\beta(z) = E_{\beta,1}(z) = \frac{1}{\beta}\exp\left(z^{1/\beta}\right) - \sum_{k=1}^{p} \frac{z^{-k}}{\Gamma(1-\beta k)} \\ + \frac{1}{2\pi\beta i z^p}\int_{\gamma(1,\varphi)} \exp\left(\zeta^{1/\beta}\right)\zeta^p d\zeta \quad (|z|>1). \end{aligned} \tag{33}$$

Substitute $z = LT^\beta$, we have

$$\begin{aligned} E_\beta(LT^\beta) = \frac{1}{\beta}\exp\left(L^{1/\beta}T\right) - \sum_{k=1}^{p} \frac{(LT^\beta)^{-k}}{\Gamma(1-\beta k)} \\ + \frac{1}{2\pi\beta i\left(L\left(T^\beta\right)\right)^p}\int_{\gamma(1,\varphi)} \exp\left(\zeta^{1/\beta}\right)\zeta^p d\zeta \quad (|z|>1). \end{aligned} \tag{34}$$

first term: For the first term $\frac{1}{\beta}\exp\left(L^{1/\beta}T\right)$, the derivative with respect to β is

$$w(\beta) = -\frac{\exp(L^{1/\beta}T)}{\beta^2} - \frac{\exp(L^{1/\beta}T)L^{1/\beta}T\log(L)}{\beta^3} = -\frac{\exp(L^{1/\beta}T)}{\beta^3}\left(\beta + L^{1/\beta}T\log(L)\right)$$

We have that when T sufficiently large, $\beta + L^{1/\beta}T\log(L)$ would be negative due to $L < 1$. So the full first term is positive. We can prove that for any constant $M > 0$, we have $w(\beta) > M$ when T is large enough:

$$w(\beta) > -\exp(L^{1/\beta}T)\left(1 + L^{1/\beta}T\log(L)\right),$$

where the right-hand side is unbounded when $T \to \infty$.

second term: For $p = 1$, it is clear that $\frac{(LT^\beta)^{-1}}{\Gamma(1-\beta)}$ is increasing.

third term: Let us first deal with the integral in (33)

$$I_p(z) = \frac{1}{2\pi\beta i z^p}\int_{\gamma(1,\varphi)} \exp\left(\zeta^{1/\beta}\right)\zeta^p d\zeta$$

for large $|z|$ and $|\arg(z)| \leq \mu$. For large $|z|$ and $|\arg(z)| \leq \mu$ we have

$$\min_{\zeta\in\gamma(1,\varphi)} |\zeta - z| = |z|\sin(\varphi-\mu),$$

and therefore for large $|z|$ and $|\arg(z)| \leq \mu$ we have

$$|I_p(z)| \leq \frac{|z|^{-1-p}}{2\pi\beta\sin(\varphi-\mu)}\int_{\gamma(1,\varphi)} \left|\exp\left(\zeta^{1/\beta}\right)\right| |\zeta^p|\, d\zeta.$$

The integral on the right-hand side converges, because for ζ such that $\arg(\zeta) = \pm\varphi$ and $|\zeta| \geq 1$ the following holds:

$$\left|\exp\left(\zeta^{1/\beta}\right)\right| = \exp\left(|\zeta|^{1/\beta}\cos\left(\frac{\varphi}{\beta}\right)\right) \tag{35}$$

where $\cos(\varphi/\beta) < 0$ due to the chosen of φ. We have that the integration is bounded and $I_p(z)$ is bounded.

Given the function:

$$f(\beta) = \frac{1}{2\pi\beta i\left(L\left(T^\beta\right)\right)^p}\int_{\gamma(1,\varphi)} \exp\left(\zeta^{1/\beta}\right)\zeta^p d\zeta,$$

we are going to take the derivative with respect to β. Let's separate the function into three parts:

1. $g(\beta) = \frac{1}{2\pi\beta i(L(T^\beta))^p}$
2. $h(\zeta,\beta) = \exp\left(\zeta^{1/\beta}\right)$
3. $j(\zeta) = \zeta^p$

And we have $f(\beta) = g(\beta) \cdot \int_{\gamma(1,\varphi)} h(\zeta,\beta) \cdot j(\zeta) d\zeta$ Now the derivative $df(\beta)/d\beta$ will be:

$$\frac{df(\beta)}{d\beta} = g'(\beta) \cdot \int_{\gamma(1,\varphi)} h(\zeta,\beta) \cdot j(\zeta) d\zeta + g(\beta) \cdot \int_{\gamma(1,\varphi)} h'(\zeta,\beta) \cdot j(\zeta) d\zeta$$

Let's find the derivatives $g'(\beta)$ and $h'(\zeta,\beta)$: For $g'(\beta)$, we use the chain rule:

$$g'(\beta) = -\frac{\left(LT^{\beta}\right)^{-p} (1 + \beta p \log(T))}{\beta^2 2\pi i}$$

For $h'(\zeta,\beta)$, we also use the chain rule:

$$h'(\zeta,\beta) = -\frac{\ln(\zeta)}{\beta^2} \exp\left(\zeta^{1/\beta}\right) \zeta^{1/\beta}$$

So the final formula for $df(\beta)/d\beta$ is:

$$\begin{aligned}\frac{df(\beta)}{d\beta} &= -\frac{1 + \beta p \log(T)}{2\pi i \beta^2 (LT^{\beta})^p} \cdot \int_{\gamma(1,\varphi)} \exp\left(\zeta^{1/\beta}\right) \zeta^p d\zeta \\ &+ \frac{1}{2\pi i \beta \left(L\left(T^{\beta}\right)\right)^p} \cdot \int_{\gamma(1,\varphi)} \left[-\frac{\ln(\zeta)}{\beta^2} \exp\left(\zeta^{1/\beta}\right) \zeta^{1/\beta}\right] \zeta^p d\zeta\end{aligned}$$

We now turn to the case when $p = 1$ and get the following:

$$\begin{aligned}\frac{df(\beta)}{d\beta} &= -\frac{1 + \beta \log(T)}{2\pi i \beta^2 (LT^{\beta})} \cdot \int_{\gamma(1,\varphi)} \exp\left(\zeta^{1/\beta}\right) \zeta d\zeta \\ &+ \frac{1}{2\pi i \beta^3 (LT^{\beta})} \cdot \int_{\gamma(1,\varphi)} \left[-\ln(\zeta) \exp\left(\zeta^{1/\beta}\right) \zeta^{1/\beta}\right] \zeta d\zeta\end{aligned}$$

For both integrations, it is bounded according to (35). Next we see that $|\frac{df(\beta)}{d\beta}|$ is uniformly converge to 0 w.r.t. T over the interval $[\varepsilon, 1]$, i.e. for any $\delta > 0$, we have $|\frac{df(\beta)}{d\beta}| \leq \delta$ over the interval $[\varepsilon, 1]$ when T is sufficiently large.

The proof is now complete if we combine the three terms to conclude the derivative of $E_{\beta}(LT^{\beta})$ w.r.t. β is positive over $[\varepsilon, 1]$ for large enough T. □

E Experimets

E.1 Datasets

The statistics of the datasets used in our experiments are presented in Table 4. The attack budgets for the GIA, as shown in Table 2, are detailed in Table 5. We adhered to the attack budgets specified in the paper [38] for GIA.

Dataset	# Nodes	# Edges	# Features	# Classes
Cora	2708	5429	1433	7
Citeseer	3327	4732	3703	6
PubMed	19717	44338	500	3
Computers	13,752	245,861	767	10

Table 4: Dataset Statistics

E.2 White-box attack

White-box attacks, which directly target the model, are stronger than the black-box attacks used in Table 2. To demonstrate that our graph neural FDE model can consistently improve the robustness of graph neural ODE models, we also conducted white-box GIA. The results are presented in Table 6. Although the accuracy under white-box GIA is lower than under black-box GIA, our graph

Dataset	max # Nodes	max # Edges
Cora	60	20
Citeseer	90	10
PubMed	200	100
Computers	300	150

Table 5: Attack budget for GIA

neural FDE models still outperform the graph neural ODE models. This observation aligns with our theoretical findings presented in Section 4 of our main paper. Our graph neural FDE models indeed enhance the robustness of neural ODE models under both black-box and white-box scenarios.

Dataset	Attack	F-GRAND	F-GRAND	F-GraphBel	GraphBel	F-GraphCON	GraphCON
Computers	*clean*	90.0±0.05	92.78±0.13	88.36±1.05	90.14±0.27	89.99±0.28	91.70±0.25
	PGD	75.29±1.17	16.44±0.11	86.35±0.10	67.04±1.28	71.64±2.33	13.11±4.73
	TDGIA	71.99±0.73	15.10±0.76	86.21±0.21	53.75±2.84	66.35±1.94	4.33±4.21

Table 6: Node classification accuracy (%) on graph **injection, evasion, non-targeted, white-box** attack in **inductive** learning.

E.3 Adversarial training

Adversarial training (AT), as demonstrated in the paper [76], is an effective strategy for mitigating attacks. It entails the incorporation of perturbations or noise during the training process to bolster the robustness of the model. This approach serves as a general framework that can be applied to any model. In this work, we employ the Projected Gradient Descent (PGD) adversarial training method (AT-PGD), as outlined in [76], to train our graph neural FDE models and enhance their robustness.

The results of AT are presented in Table 7. It is evident that AT-PGD significantly improves the robustness of our neural FDE models. These findings demonstrate that our neural FDE models can be effectively combined with other defense mechanisms.

E.4 Pre-processing methods

As mentioned in our main paper, several methods [9], [11], [8] employ preprocessing techniques to prune or rewire the graph structure, thereby removing malicious edges or nodes. In this work, we integrate our graph neural FDE models with GNNGUARD [9] to further enhance their performance. GNNGUARD identifies suspicious nodes or edges and refines the edge weights to mitigate the influence of these suspicious edges.

We introduce the models F-GraphCON-GUARD and GraphCON-GUARD by incorporating the GNNGUARD preprocessing techniques into the graph neural FDE model and neural ODE model, respectively. The results after adversarial attacks are presented in Table 8. As can be observed, GNNGUARD enhances the robust accuracy in both cases. However, our F-GraphCON-GUARD performs better, as the base F-GraphCON model already exhibits superior robustness compared to GraphCON. This demonstrates that our neural FDE model can be seamlessly integrated with preprocessing methods to further improve robustness.

Dataset	Attack	F-GRAND-AT	F-GraphBel-AT	F-GraphCON-AT
Citeseer	*clean*	72.0±0.31	71.37±0.80	65.99±0.16
	PGD	71.16±0.84	71.25±0.24	65.93±0.10
	TDGIA	71.26±0.73	70.01±0.65	64.94±0.26
	MetaGIA	71.58±0.63	70.95±0.85	65.73±0.42

Table 7: Node classification accuracy (%) on graph **injection, evasion, non-targeted, black-box** attack in **inductive** learning.

Dataset	Attack	F-GraphCON-GUARD	F-GraphCON	GraphCON-GUARD	GraphCON
Citeseer	*clean*	71.22±0.61	73.50±0.34	71.93±0.82	72.07±0.93
	PGD	62.28±1.86	54.47±1.0	46.52±3.37	37.71±7.0
	TDGIA	63.36±1.34	54.71±1.69	51.52±1.71	30.93±3.0
	MetaGIA	59.73±2.26	48.82±3.27	51.70±3.40	29.09±2.01
Pubmed	*clean*	90.06±0.07	90.30±0.11	89.60±0.17	88.09±0.32
	PGD	83.39±3.38	51.16±6.04	63.55±2.19	45.85±1.97
	TDGIA	73.20±1.44	55.50±4.03	51.73±2.94	45.57±2.02
	MetaGIA	79.97±5.01	52.03±5.53	62.44±3.71	45.81±2.81

Table 8: Node classification accuracy (%) on graph **injection, evasion, non-targeted** attack in **inductive** learning.