HyperDefender: A Robust Framework for Hyperbolic GNNs

Nikita Malik[1], **Rahul Gupta**[2], **Sandeep Kumar**[1,3,4]

[1]Bharti School of Telecommunication Technology and Management, Indian Institute of Technology, Delhi, India
[2]Department of Mathematics and Computing, Indian Institute of Technology, Delhi, India
[3]Department of Electrical Engineering, Indian Institute of Technology, Delhi, India
[4]Yardi School of Artificial Intelligence, Indian Institute of Technology, Delhi, India
bsz218185@iitd.ac.in, mt1200619@iitd.ac.in, ksandeep@iitd.ac.in

Abstract

Graph neural networks for hyperbolic space has emerged as a powerful tool for embedding datasets exhibiting a highly non-Euclidean latent anatomy e.g., graphs with hierarchical structures. While several Hyperbolic Graph Neural Networks (Hy-GNNs) have been developed to enhance the representation of hierarchical datasets, they remain susceptible to noise and adversarial attacks, posing serious risks in critical applications. The absence of robust Hy-GNN frameworks underscores a pressing problem. This research addresses this challenge by introducing HyperDefender—a robust and flexible approach designed to fortify Hy-GNNs against adversarial attacks and noises. HyperDefender aims to secure the reliability of applications that depend on the integrity of hierarchical graph-structured data in real-world scenarios. Experimental results demonstrate that HyperDefender significantly improves node classification accuracy across various attacks, effectively mitigating the performance degradation typically observed in Hy-GNNs when the hierarchy in original datasets is compromised.

Code — https://github.com/nikimal99/HyperDefender.git

1 Introduction

Graph Neural Networks (GNNs) extend traditional neural networks to graph-structured data (Xu et al. 2018; Velickovic et al. 2017; Kipf and Welling 2016). Despite the remarkable breakthroughs, the existing GNN models are still limited by the representation ability of Euclidean geometry, especially for datasets with hierarchical structures and highly non-Euclidean latent anatomy (Yang et al. 2022; Zitnik et al. 2019; Clauset, Moore, and Newman 2008). Numerous real-world datasets, such as disease propagation trees, flight networks (Chami et al. 2019), recommendation systems (Wang et al. 2021a), knowledge graphs (Wang et al. 2021b), drug molecules (Qu and Zou 2022), stock graph relations (Sawhney et al. 2021), medical ontology matching networks (Hao et al. 2021), and Human PPI networks (Li et al. 2023) have hierarchical structure.

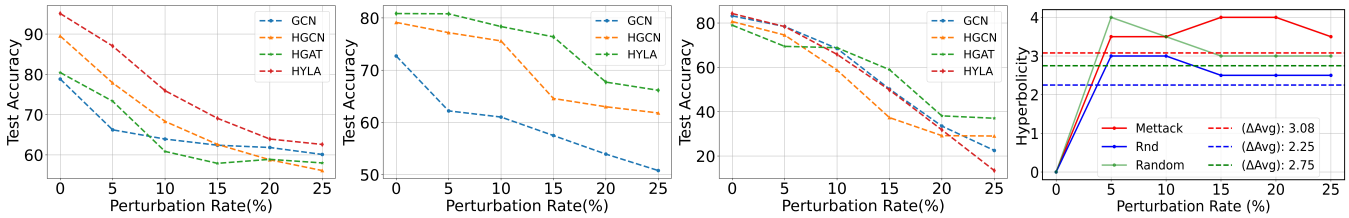
Recently, hyperbolic spaces due to their exponential growth properties have emerged as a better alternative, allowing for high-fidelity embedding of hierarchical structured data. Due to this several Hyperbolic Graph Neural Networks (Hy-GNNs) architectures have been developed to enhance the representation of hierarchical datasets. The theoretical and empirical superiority of hyperbolic space, characterized by constant negative curvature, is evident in capturing implicit hierarchies and outperforming Euclidean baselines. Motivated by these advantages, ongoing research focuses on developing machine learning models in hyperbolic space. Hy-GNN (Chami et al. 2019; Liu, Nickel, and Kiela 2019; Shimizu, Mukuta, and Harada 2020; Gulcehre et al. 2018), including HyLa-SGC (Yu and De Sa 2022), have demonstrated strong empirical results in tasks like node classification and link prediction, highlighting the efficacy of hyperbolic geometry in graph-based learning.

In summary, while Hy-GNNs have demonstrated remarkable performance in hierarchical graph representation learning (Peng et al. 2021; Yang et al. 2022), their reliability remains an open question. At this juncture, it is crucial to address the following question:

Are existing Hy-GNN models inherently robust against adversarial attacks and noise in the input data? If not, how can we enhance their robustness to ensure their effectiveness in practical scenarios?

To address these questions, we evaluate the vulnerability of Hy-GNNs (including HGAT (Zhang et al. 2021), HGCN (Chami et al. 2019), and HyLa-SGC) and GCNs under adversarial attacks, as illustrated in Figures 1a, 1b and 1c. The performance plots demonstrate a consistent decline for Hy-GNNs when exposed to adversaries, indicating that these models, similar to GNNs, lack inherent robustness against adversarial attacks. Moreover, adversarial attacks can compromise the inbuilt hierarchies (measured by Gromov δ-hyperbolicity, details in section 2) of such networks, as shown in Figure 1d.

This raises concerns about the utility of Hy-GNNs, as their performance degrade significantly under subtle perturbations in graphs. Such a lack of robustness poses severe consequences for critical applications (Hao et al. 2021; Sawhney et al. 2021) which must be considered to prevent vulnerabilities exploited by attackers. These hierarchical networks can be corrupted for discrediting information, gaining a competitive advantage, extortion, or financial gain. In practical scenarios, data like the disease networks depicting infection propagation are prone to human errors during



(a) (b) (c) (d)

Figure 1: Node classification performance under Mettack for (a) Airport, (b) Disease, and (c) Cora datasets. (d) depicts the hyperbolicity analysis of the disease network under various adversarial attacks with increasing perturbation rates, highlighting the disruption of hierarchical structure in the graph.

data collection and can face severe consequences if the data is corrupted. Additionally, random failures and malicious attacks on complex tree-like networks, in the form of node and edge-manipulations (Xu et al. 2020), necessitate strong robustness for practical networked systems.

Despite the significant advancements in adversarial attacks and defenses for GNNs, the robustness of Hy-GNNs remains largely unexplored. This study aims to bridge this gap, offering the following key contributions:

- We assess the susceptibility of existing GNNs and Hy-GNNs to various poisoning and evasion attacks, revealing that neither the standalone Hy-GNN architectures nor traditional Euclidean defense methods are sufficient to mitigate adversarial attacks and noise in hierarchical datasets.
- We perform a Gromov δ-hyperbolicity analysis on hierarchical graph networks, examining how different adversarial attacks compromise the hierarchical integrity of the data.
- We introduce HyperDefender, the first flexible and practical framework that leverages the strengths of various Euclidean defenses to create robust hyperbolic methods. This innovative approach not only withstands adversarial attacks but also restores the lost hierarchy in compromised datasets.
- We rigorously test the HyperDefender's performance under adversarial conditions, benchmarking it against existing Euclidean defense models in node classification tasks to provide deep insights into its effectiveness.

Outline: This paper is structured as follows: Section 2 presents the problem formulation and background. Our proposed framework is elaborated in Section 3, and experimental results on adversarial attacks, along with comparative analysis with existing methods, are discussed in Section 4.

2 Problem Formulation and Background

Let $\mathcal{G} = (\mathbf{A}, \mathbf{X}, \delta)$ be a hierarchical graph, where $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ is the set of M-dimensional node features, and δ denotes Gromov-hyperbolicity (Väisälä 2005). The adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ has elements $\boldsymbol{A}_{uv} \in \{0, 1\}$ indicating the presence of edge e_{uv} between nodes u and v. We consider the node classification task, where a Hy-GNN f_h classifies nodes into C categories. The goal is to learn a function $f_{h,\theta} : \mathcal{V}_L \to Y_L$ that predicts labels Y_L for unlabeled nodes $\mathcal{V}_L$. The objective function is formulated as:

$$\min_{\theta} \mathcal{L}(\theta, A, X, Y_L) = \sum_{v_i \in V_L} \ell\left(f_\theta(X, A)_i, y_i\right), \quad (1)$$

where θ represents the parameters of $f_{h,\theta}$, $f_{h,\theta}(X, A)_i$ is the prediction for node v_i, and $\ell(\cdot, \cdot)$ measures the difference between the prediction and the true label, such as cross-entropy. In an adversarial scenario, the graph $\mathcal{G}$ is perturbed, resulting in $\mathcal{G}'$. Training Hy-GNNs with noisy graph information may not yield a reliable predictor function $f_{h,\theta}$. Moreover, the unique properties of hyperbolic space pose challenges for Hy-GNN models (Yu and De Sa 2022), which often rely on mappings between tangent space and the hyperbolic manifold. Unlike Euclidean space, hyperbolic space lacks certain vector space properties, complicating fundamental operations such as addition and multiplication. Consequently, optimizing Hy-GNN architectures within hyperbolic space present significant difficulties for developing defense mechanisms.

Problem statement. *Formally, considering a hierarchical graph structure $\mathcal{G}' = (A_n, X, \delta_n)$ targeted by adversaries, where A_n denotes the perturbed adjacency matrix $A_n \in \mathbb{R}^{N\times N}$, and δ_n is the Gromov-hyperbolicty value of the attacked graph, our objective is: to learn the embeddings in hyperbolic space that optimize the node classification performance of Hy-GNN against adversarial attacks.*

Goal. The goal is to develop a range of flexible and practical robust Hy-GNNS methods. To achieve this, we envision a two-stage solution: (i) learning hyperbolic embeddings to approximate euclidean features, and (ii) applying a euclidean defense mechanism. A schematic outline of the proposed framework is presented in Figure 2 and detailed in Section 3. Before delving into the framework, we first provide background on adversarial attacks and hyperbolicity.

Adversarial attacks and defense mechanisms. Adversarial attacks (Dai et al. 2018) can be categorized into poisoning attacks and evasion attacks. In poisoning attacks (Zügner, Akbarnejad, and Günnemann 2018), the attacker perturbs the training graph, while in evasion attacks, the attacker perturbs the graph during testing. According to the attacker's goal, these attacks can be further divided into (1) Targeted attacks: where the attacker misclassifies spe-

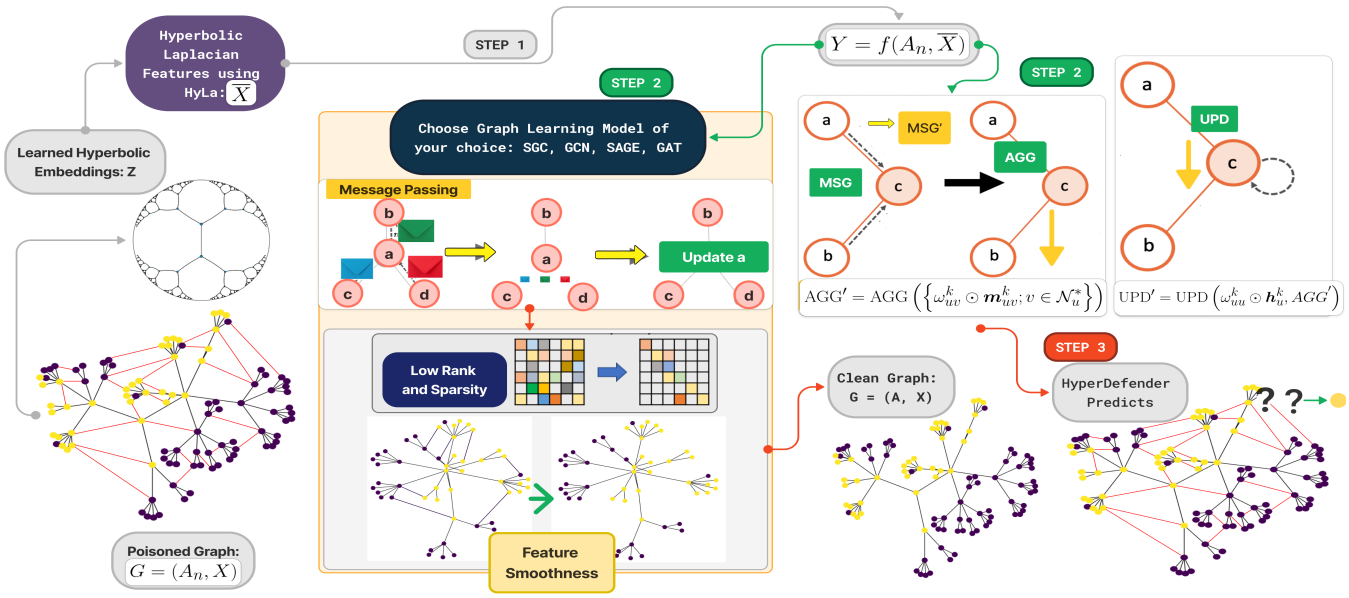


Figure 2: The figure showcases the HyperDefender framework, which consists of three steps: Step 1 involves learning hyperbolic embeddings (Z) of the perturbed data $\mathcal{G}' = (A_n, X)$ and estimating HyLa features using Z. In Step 2, a Euclidean defense method is applied to $\mathcal{G}' = (A_n, \overline{X})$. Finally, in Step 3, the framework either retrieves the clean graph or achieves high performance on the perturbed graph through various defense methods.

cific nodes, (2) Non-targeted attacks: where the attacker degrades the model's overall performance on all test data, and (3) Random attacks: where the attacker randomly injects fake edges into the graph. Adversarial attacks present a significant threat to deep learning models, GNNs (Bojchevski and Günnemann 2019; Dai et al. 2018; Jin et al. 2021) as well as Hy-GNNs. Many recent efforts have concentrated on enhancing GNN robustness (Zhao et al. 2023; Ennadir et al. 2024; Jin et al. 2020; Zhang and Zitnik 2020; Runwal, Vivek, and Kumar 2022; Wu et al. 2019) but none of them considers how to defend hierarchical graphs. These methods doesn't work well for hierarchical graphs as shown in Section 4.

Hyperbolicity of graphs. Real-world networks are rarely exact trees (Abu-Ata and Dragan 2016), which is why a measure of tree-likeness, such as Gromov's δ-hyperbolicity (Väisälä 2005), is useful. A lower δ signifies a more hierarchical graph dataset, with $\delta = 0$ indicating a tree structure.

Definition 1: Let G be a graph with node set V, and let $\ell(u, v)$ be the shortest path distance between nodes u and v. G is δ-hyperbolic if, for any nodes $w, x, y, z \in V$, ordered such that $\ell(w, x) + \ell(y, z) \geq \ell(w, y) + \ell(x, z) \geq \ell(w, z) + \ell(x, y)$, the following condition holds:

$$(\ell(w, x) + \ell(y, z)) - (\ell(w, y) + \ell(x, z)) \leq 2\delta$$

A detailed explanation is provided in Appendix A.2. Gromov's δ-hyperbolicity remains consistent across graphs exhibiting hyperbolic geometry, regardless of their size, making it a valuable metric for evaluating real-world graphs. In this study, we examined how increasing levels of perturbation through various adversarial attacks affect the hyperbolicity value as shown in Fig 1d. Consequently, changes in empirical δ-hyperbolicity can indicate whether a graph has been perturbed, often making it less hierarchical. A key objective of defense mechanism is to relearn the structure and restore true δ-hyperbolicity. Thus, if δ represents the hyperbolicity of the original graph, δ_n that of the noisy graph, and δ_{clean} that of the restored graph, the defense model aims to achieve $\delta_{clean} \approx \delta$.

3 Proposed Hyperdefender Framework

The proposed framework HyperDefender consists of the following two main components:

1. *Estimation of hyperbolic embeddings to estimate the Euclidean features (or HyLa features) $\overline{X}$;.*
2. *Using the estimated $\overline{X}$ and A_n as an input to any euclidean defense mechanism $\mathcal{E}$ to get $\mathcal{H}$ i.e., hyperbolic defense method.*

The first component being an essential part of a robust Hy-GNN architecture that helps to understand the hierarchical graph better. While the latter is designed to provide defense against attacks.

Stage 1: Euclidean features from hyperbolic embeddings. We first discuss the estimation of $\overline{X}$. It is a feature-extracted architecture proposed in (Yu and De Sa 2022) which we utilise to embed the graph nodes or features into hyperbolic space. It maps the hyperbolic embeddings to Euclidean features $\overline{\mathbf{X}}$ via the kernel transformation. Concretely, to map from hyperbolic space $\mathbb{H}_n$ to euclidean space $\mathbb{R}^D$, we perform the following steps:

1. Draw D independent samples $\lambda_1, \ldots, \lambda_D$ from ρ where ρ is gaussian,
2. D independent samples $\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_D$ uniform from $\partial\mathcal{B}^n$ (the boundary of the ball $\mathcal{B}^n$) and,
3. D independent samples $b_1, \ldots, b_D$ uniform from $[0, 2\pi]$ and,
4. Then output a feature map ϕ, the kth coordinate of which is $\phi_k(\boldsymbol{x}) = \frac{1}{\sqrt{D}}\mathrm{HyLa}_{\lambda_k, b_k, \boldsymbol{\omega}_k}(\boldsymbol{x})$,
5. It is easy to see that this will yield feature vectors with $\mathbb{E}[\langle\phi(\boldsymbol{x}), \phi(\boldsymbol{y})\rangle] = k(\boldsymbol{x}, \boldsymbol{y})$,

where, the kernel $k(\boldsymbol{x}, \boldsymbol{y})$ is defined as:

$$k(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{2}\int_{-\infty}^{\infty} {}_2F_1\left(\frac{n-1}{2} + i\lambda, \frac{n-1}{2} - i\lambda; \frac{n}{2}; \frac{1}{2}\left(1 - \cosh\left(d_{\mathbb{H}}(\boldsymbol{x}, \boldsymbol{y})\right)\right)\right) \cdot \rho(\lambda) d\lambda. \tag{2}$$

which is selected at random from some distribution $\rho(\lambda)$. The resulting kernel is an isometry invariant kernel that depends on the distribution of λ (Yu and De Sa 2022; Rahimi and Recht 2007).

This approach only manipulates the input of the graph-based learning model and hence, this architecture can be used with any standard Euclidean defense model, $\boldsymbol{\mathcal{E}}$. Here, the model parameters and the hyperbolic embeddings are learned simultaneously with the backpropagation.

Algorithm 1: HyperDefender Framework

Input: $G' = (A_n, X, Y, \delta_n)$, $\boldsymbol{\mathcal{E}}$
Output: $\boldsymbol{\mathcal{H}_\theta}$ (parameters of $\boldsymbol{\mathcal{H}}$)
Stage 1: Estimate HyLa features $\overline{X}$
Stage 2: Apply $\boldsymbol{\mathcal{E}} \rightarrow (\overline{X}, A_n)$
Return $\boldsymbol{\mathcal{H}_\theta}$

Stage 2: Applying defense measures. The defense operates through two primary strategies: **(i)** The first strategy simultaneously learns a clean graph structure A^* constrained to $\mathcal{A} = \{0, 1\}^{N \times N}$ and learns the model parameters $\boldsymbol{\mathcal{H}_\theta}$ (refer, Equation 1), formalized as:

$$\arg\min_{\boldsymbol{\mathcal{H}_\theta}} L\left(\boldsymbol{\mathcal{H}_\theta}, A^*, \overline{X}, y_i\right), \tag{3}$$

$$\text{and } A^* = \arg\min_{A} L_{nr}\left(A_n, \overline{X}, A\right) \tag{4}$$

where the model is trained on a clean graph adjacency A^*, obtained by simultaneously minimizing a noise removal objective function $L_{nr}\left(A_n, \overline{X}, A\right)$. This strategy is employed in both the Pro-HyLa and RWL-HyLa defense methods. For training these models we used joint learning strategy where we are simulatneously learning the clean structure of graph, the hyla features and the model parameters.

(ii) The second strategy involves determining the prediction $\ddot{y}'_{\mathrm{h},u}$ on the attacked graph $\mathcal{G}'$ that minimizes the difference:

$$\ddot{y}'_{\mathrm{h},u} = \arg\min_{\ddot{y}'_{\mathrm{h},u}} \left(f'_{\mathrm{h},u}\left(\mathcal{G}'\right) - f_{\mathrm{h},u}(\mathcal{G})\right) \tag{5}$$

Here, $f'_{\mathrm{h},u}\left(\mathcal{G}'\right)$ represents the prediction of Hy-GNN trained on $\mathcal{G}'$. In this case, $f'_{\mathrm{h}}\left(\mathcal{G}'\right) = \boldsymbol{\mathcal{H}}$. Here, $f_{\mathrm{h}}\left(\mathcal{G}\right)$ denotes a hypothetical prediction that the Hy-GNN would make if it had access to clean graph $\mathcal{G}$. This strategy forms the basis of the HyLa-Guard approach. Here we are jointly learning the hyla features and model parameters for this model.

Next, we present the hyperbolic defense strategies ($\boldsymbol{\mathcal{H}}$), which we developed using these two essential components. Here we give a thorough formulation for Pro-HyLa.

Pro-HyLa. The final objective function of Pro-HyLa is given as follows:

$$\arg min_{s.t., A=A^T;\, A\in\mathcal{A}, \boldsymbol{\mathcal{H}_\theta}} L_{total} = L_{nr}(A_n, \overline{X}, A) + \gamma L \tag{6}$$

where L is the loss function defined in Equation 1 for node classification, controlled by a predefined parameter γ. Overall, L_{total} constitutes the loss of Pro-HyLa with learnable parameters $\boldsymbol{\mathcal{H}_\theta}$ and clean adjacency A. Finally, the noise removal objective function is defined as:

$$L_{nr}(A_n, X, A) = \|A_n - A\|_F^2 + \alpha\|A\|_1 + \beta\|A\|_* + \lambda\, tr(\overline{X}^T\widehat{L}\overline{X}) \tag{7}$$

It is remarked that adversarial attacks focus on introducing unnoticeable perturbations and connecting two nodes with dissimilar features (McPherson, Smith-Lovin, and Cook 2001). The formulation in Equation 7 is a potential solution for providing robustness against such types of attacks. More concretely, the term $\|A_n - A\|_F^2$ tries to maintain the structure of the learned graph by ensuring that the learned Adjacency matrix does not deviate too far from the original input matrix.

Next, by minimizing $tr(\overline{X}^T\hat{L}\overline{X}) = \frac{1}{2}\sum_{i,j=1}^{N} A_{i,j}(x_i - x_j)^2$, we aim to ensure that the edges introduced between two nodes with dissimilar features should be removed or down-weighted. λ is a predefined parameter to control the contribution from feature smoothness. α and β are predefined parameters that control the contributions of the properties of sparsity and low rank, respectively. The added advantage of this formulation (7) lies in its ability to utilize

Algorithm 2: Pro-HyLa

Input: A_n, X, Y_L, Hyper-parameters $\alpha, \beta, \gamma, \lambda, \tau$, Learning rate η, η', HyLa feature dimension d_1
Output: A, $\boldsymbol{\mathcal{H}_\theta}$
Initialize $A \leftarrow A_n$ and randomly initialize $\boldsymbol{\mathcal{H}_\theta}$
Initialize $Z \in \mathbb{R}^{n \times d_0}$ {hyperbolic embeddings}
while Stopping condition not met **do**
  $A \leftarrow \mathrm{PS}(\mathrm{prox}_{\eta\alpha}(\mathrm{prox}_{\eta\beta}(A - \eta\nabla_A\Phi)))$
  for $i = 1$ to τ **do**
    compute $\overline{X}$ HyLa features
    Update $\boldsymbol{\mathcal{H}_\theta}$ parameter using $\mathcal{L}$:
    $\delta \leftarrow \frac{\partial\mathcal{L}(\boldsymbol{\mathcal{H}_\theta}, A, \overline{X}, Y_L)}{\partial\boldsymbol{\mathcal{H}_\theta}}$
    $\boldsymbol{\mathcal{H}_\theta} \leftarrow \boldsymbol{\mathcal{H}_\theta} - \eta'\delta$
  end for
end while
Return A, $\boldsymbol{\mathcal{H}_\theta}$

Data	δ	Size	Density	Edges	Classes
Cora	3	(2485, 1433)	16.42	5069	7
Disease	0	(1044, 1000)	7.5	1043	2
Airport	1.5	(3188, 11)	36.7	18630	4

Table 1: Dataset Specifications

information from L to steer the graph learning process, serving as a defense mechanism against adversarial attacks. This is particularly effective since graph adversarial attacks aim to maximize L.

The strength of the proposed Pro-HyLa framework lies in its seamless integration of HyLa model training with graph reconstruction, functioning as an iterative process that jointly refines both the estimated graph A and the model parameters $\boldsymbol{\mathcal{H}_\theta}$. The workflow, detailed in Algorithm 2, unfolds as follows: In line 3, the estimated graph A is initialized as the perturbed graph A_n, and the model parameters $\boldsymbol{\mathcal{H}_\theta}$ are randomly initialized. Line 4 initializes the hyperbolic embeddings Z. From lines 5 to 13, A and $\boldsymbol{\mathcal{H}_\theta}$ are updated iteratively, alternating between updates on the clean structure. In line 6, the objective function is defined as $\phi = \|A_n - A\|_F^2 + \gamma L + \lambda\, tr(\overline{X}^T \widehat{L} \overline{X})$. In line 8, Hyla features $\overline{X}$ are estimated using the steps explained in Stage 1. Moreover, we used the original authors' implementation of HyLa (Yu and De Sa 2022), together with the SGC model, i.e., $f(A, \overline{X}) = \text{softmax}(A^K \overline{X} W)$.

Essentially, we learn the HyLa features $\overline{X}$ and use SGC as the graph learning model while ensuring other graph properties, such as feature smoothness, sparsity, and low-rank structure, are maintained as we learn a clean adjacency matrix A. Specifically, the model parameters are trained in each iteration, while the graph reconstruction model is updated every τ iterations.

4 Experiments

In this section, we evaluate the effectiveness of HyperDefender.

Task and Datasets. We comprehensively compare the node classification performance of HyperDefender methods with existing euclidean-based defense mechanisms as baselines: NoisyGNN (Zhao et al. 2023), HANG (Ennadir et al. 2024), Pro-GNN (Jin et al. 2020), GNNGUARD (Zhang and Zitnik 2020), RWL-GNN (Runwal, Vivek, and Kumar 2022), and Jaccard-GNN (Wu et al. 2019) and also a Hy-GNN model, i.e., HyLa-SGC. In addition, we analyze the impact of attacks on the hyperbolicity of graphs.

We evaluate our proposed approach on three diverse datasets: Cora (Sen et al. 2008), Disease (Anderson and May 1991) and Airport (Zhang and Chen 2018). Dataset statistics are summarized in Table 1. The hyperbolicity values correspond to our modified versions of the datasets.

Experiment Setup. We evaluate the node classification performance of HyperDefender against three types of attacks, i.e., non-targeted attacks, targeted attacks, and random attacks. We apply these attack methods to perturb the graph, followed by training HyperDefender and baseline models on the perturbed graph.

Performance analysis. In Table 2 and Figure 3, detailed performance results for each model are shown. Observations include:

- HyperDefender defense methods consistently outperform Euclidean-based defenses and also HyLa-SGC (Hy-GNN) model across various perturbation rates. Taking the airport data as an example, our models exhibit significant improvements over HyLa-SGC. At a 100% perturbation rate in random attack scenarios (Figure 3 first row), nearly all our HyperDefender models surpass HyLa-SGC by over 50%, and exceed 10% at 25% perturbation in Mettack. Also, Hyperbolic-based defenses demonstrate considerable superiority over Euclidean-based defenses on the Airport dataset in all the attacks.
- For the disease dataset, Pro-HyLa outperforms HyLa-SGC, Pro-GNN, and other Euclidean-based defense methods by a wider margin, even under substantial perturbations. In the Cora dataset, our method surpasses HyLa-SGC and Pro-GNN by over 58% and 6%, respectively, at a 25% perturbation rate. But in some scenarios, our defense methods show comparable performance to standard Euclidean-based methods, in the case of the Cora dataset.

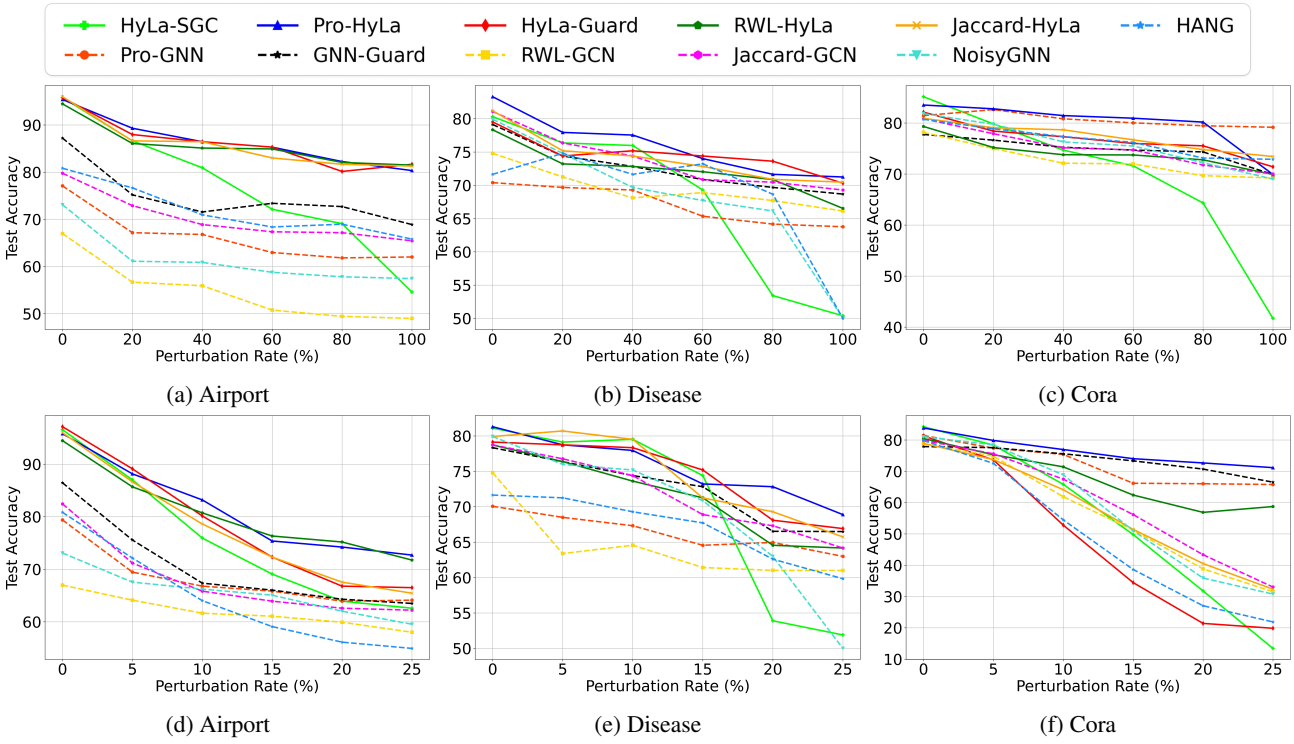
While Pro-HyLa emerges as a superior performer in cases of disease and airport datasets, RWL-HyLa shows good performance in cases of RND attack (Table 2) on Cora and disease. The results suggest that the more hierarchical the graph is, the more improvements will be gained with hyperbolic based defense methods. The average performance depicted in Figure 5 illustrates Pro-HyLa's superiority over the other three (GCN, Pro-GCN, and HyLa-SGC) models. Notably, Pro-HyLa stands out as the most effective among them, demonstrating superior performance compared to its counterparts.

Visualization. In Figure 4, we visualize the HyperDefender learned HyLa node features ($\overline{X}$) on both the original and perturbed airport dataset with t-SNE. This visualization demonstrates that HyperDefender methods achieve excellent label-class separation, as indicated by the different colors, whereas Euclidean-based defense methods fail to achieve this separation. All four HyperDefender methods: Pro-HyLa, HyLa-Guard, RWL-HyLa, and Jaccard-HyLa consistently achieve strong label-class separation, even on the original unperturbed dataset.

Since it is uncertain whether an input hierarchical graph has been attacked, a successful Hy-GNN defender must handle poisoned graphs without compromising performance on clean datasets. Table 2, also shows the classification accuracy of HyperDefender methods on clean graphs (at Ptb % = 0), confirming that *HyperDefender methods maintain performance comparable to HyLa-SGC when there is no*

Data	Ptb (%)	Baselines Standard Defense Models ($\mathcal{E}$)						Proposed Hyperbolic Defense Models ($\mathcal{H}$)			
		Pro-GNN	GNN-Guard	RWL-GNN	GCN-Jaccard	NoisyGNN	HANG	Pro-HyLa	HyLa-Guard	RWL-HyLa	HyLa-Jaccard
Airport	0	83.05	81.75	70.34	76.71	66.79	78.46	94.94	**96.06**	94.47	95.03
	2	81.67	79.46	68.09	75.38	66.79	76.48	**94.38**	94.35	90.03	93.89
	4	75.57	76.75	66.25	70.61	63.16	76.28	**93.82**	93.01	87.74	91.03
	6	73.47	75.03	64.65	70.41	63.35	76.09	**93.25**	88.35	84.50	86.83
	8	72.90	73.47	64.46	68.51	62.97	73.32	**92.69**	87.90	83.24	83.77
	10	68.89	68.53	63.50	68.70	60.30	72.53	**93.25**	86.03	83.05	82.44
Disease	0	68.50	69.44	67.71	79.52	79.92	71.65	**80.31**	78.74	78.72	**80.31**
	2	66.92	57.79	63.07	65.35	64.17	63.39	67.32	65.74	**71.26**	67.71
	4	61.41	53.46	58.74	59.05	56.69	55.91	61.41	55.90	**64.57**	53.93
	6	**67.71**	52.04	58.03	52.75	55.51	48.43	**58.26**	57.48	57.87	53.54
	8	59.05	50.07	52.20	53.14	53.93	41.34	**67.32**	51.96	62.20	52.75
	10	59.84	48.42	54.88	51.57	50.03	38.98	**62.59**	53.15	56.26	54.72
Cora	0	81.34	76.59	78.17	80.63	81.84	80.73	**84.20**	81.99	79.33	80.63
	2	77.62	73.29	68.56	70.87	73.03	71.93	**80.93**	75.28	75.15	76.65
	4	71.93	68.64	65.79	64.28	63.78	58.65	72.28	70.85	**73.79**	72.53
	6	67.76	66.47	62.78	56.99	54.22	46.28	67.60	61.82	**71.13**	68.20
	8	65.24	59.95	58.00	47.83	44.76	29.02	66.19	47.07	**68.41**	66.09
	10	61.62	56.63	57.19	43.76	42.90	23.19	61.56	45.40	**68.00**	63.98

Table 2: Node classification accuracy under targeted RND attacks. The best performance is highlighted in bold.



(a) Airport (b) Disease (c) Cora

(d) Airport (e) Disease (f) Cora

Figure 3: First row: Results of different models under random attack, Second row: Results of different models under mettack

attack. This indicates that HyperDefender does not weaken the learning ability of Hy-GNNs on clean datasets.

Analysis of Hyperbolicity. Gromov's δ-hyperbolicity, was employed to assess our model's performance. For each attack, the hierarchy of the dataset was compromised. We calculated the hyperbolicity values after attacking the graph dataset and compared them with the hyperbolicity values of the learned clean adjacency matrix. We present this data in Table 3, showcasing the average hyperbolicity values for each dataset against various attacks. Here, Δavg represents the mean of δ-hyperbolicity values for six different perturbations for each attack. Similar calculations were performed for the clean adjacency matrix learned using Pro-

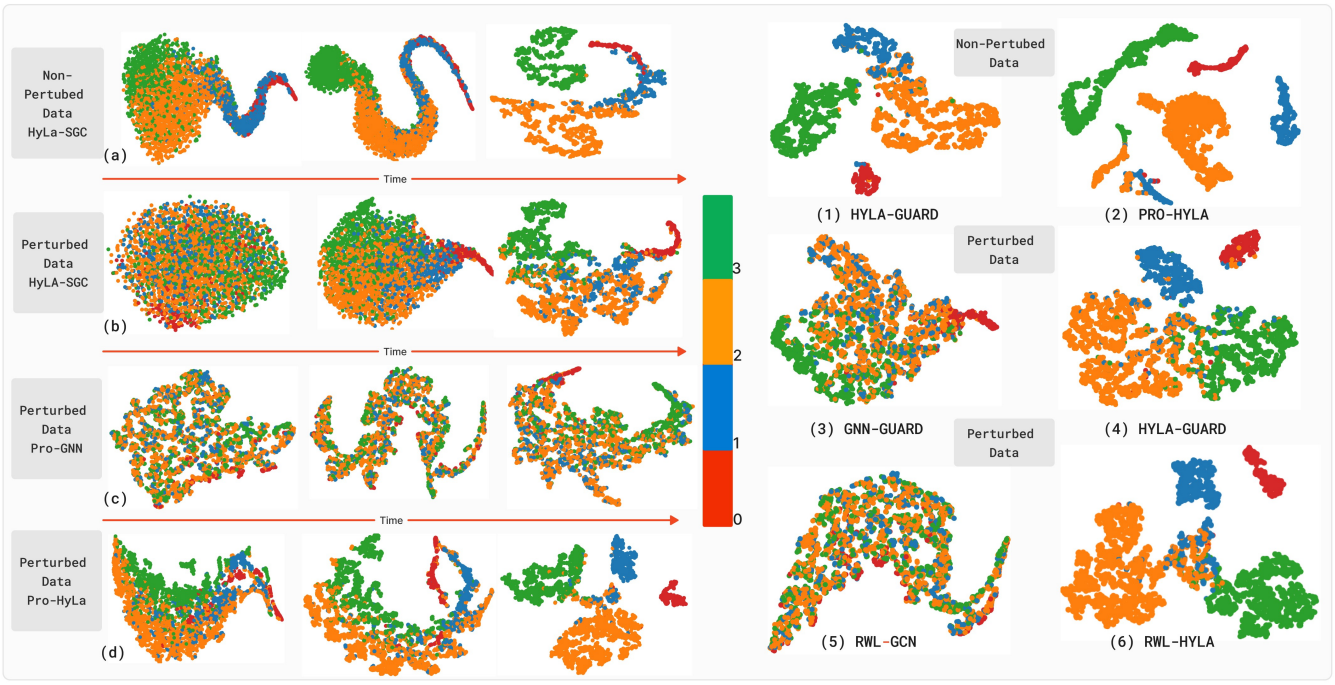


Figure 4: Evolution of latent space representations over time: Visualizing the transformation of HyLa features and node embeddings on the Airport dataset. Nodes of distinct classes are color-coded for clarity. On the left, (a) and (b) showcase HyLa-SGC's learned Hyla features for both unperturbed and perturbed data, while (c) and (d) present Pro-GNN and Pro-HyLa learned embeddings and HyLa features for perturbed data. On the right, (1) and (2) illustrate Hyla-Guard and Pro-HyLa learned node embeddings and HyLa features for the original data, while (3) and (4) compare embeddings and HyLa features of GNN-Guard and Hyla-Guard for perturbed data respectively. Similarly, (5) and (6) provide a comparison between RWL-GCN and RWL-HyLa, respectively.

Dataset	Attacks	After-attack	Pro-Hyla	RWL-Hyla	Jac-Hyla
Airport	Mettack	1.92	1.75	1.17	1.91
	RND	1.83	1.25	1.50	2.91
	Random	1.92	1.08	1.08	1.66
Disease	Mettack	3.08	0.83	2.83	2.83
	RND	2.25	1.25	2.17	2.16
	Random	2.75	1.92	2.58	2.66
Cora	Mettack	2.67	1.50	3.25	2.83
	RND	2.25	1.50	4.42	2.25
	Random	2.50	2.00	2.58	2.58

Table 3: Δavg values (closer to the original is better)

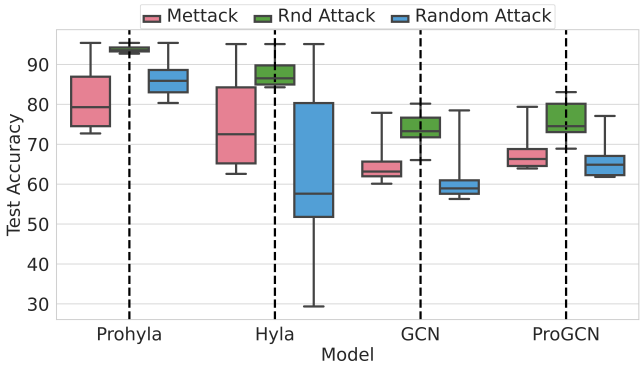


Figure 5: The figure illustrates the average performance of ProHyla's superiority over HyLa, GCN, and ProGCN.

HyLa, RWL-HyLa and Jaccard-HyLa. Observations indicate a restoration in the Δavg values for the clean adjacency matrix, suggesting that our framework, HyperDefender, is capable of partially recovering the hierarchy of hierarchical networks. Among all hyperbolic defense methods, Pro-HyLa emerges as the most effective in restoring the hierarchical structure of networks, showcasing superiority not only over Euclidean-based methods but also among other hyperbolic defense methods.

5 Conclusion

We introduce HyperDefender, a flexible and effective framework for enhancing the robustness of Hy-GNNs. Our experiments show that HyperDefender significantly outperforms HyLa-SGC and Euclidean-based defense methods. The analysis highlights Hy-GNNs' vulnerability to attacks and the critical need for HyperDefender in hierarchical networks. Notably, HyperDefender partially restores hierarchical structures in graphs. On datasets with minimal hierarchy, like Cora, HyperDefender achieves performance comparable to Euclidean defenses.

References

Abu-Ata, M.; and Dragan, F. F. 2016. Metric tree-like structures in real-world networks: an empirical study. *Networks*, 67(1): 49–68.

Anderson, R. M.; and May, R. M. 1991. *Infectious diseases of humans: dynamics and control*. Oxford university press.

Bojchevski, A.; and Günnemann, S. 2019. Adversarial attacks on node embeddings via graph poisoning. In *International Conference on Machine Learning*, 695–704. PMLR.

Chami, I.; Ying, Z.; Ré, C.; and Leskovec, J. 2019. Hyperbolic graph convolutional neural networks. *Advances in neural information processing systems*, 32.

Clauset, A.; Moore, C.; and Newman, M. E. 2008. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453(7191): 98–101.

Dai, H.; Li, H.; Tian, T.; Huang, X.; Wang, L.; Zhu, J.; and Song, L. 2018. Adversarial attack on graph structured data. In *International conference on machine learning*, 1115–1124. PMLR.

Ennadir, S.; Abbahaddou, Y.; Lutzeyer, J. F.; Vazirgiannis, M.; and Boström, H. 2024. A Simple and Yet Fairly Effective Defense for Graph Neural Networks. arXiv:2402.13987.

Gulcehre, C.; Denil, M.; Malinowski, M.; Razavi, A.; Pascanu, R.; Hermann, K. M.; Battaglia, P.; Bapst, V.; Raposo, D.; Santoro, A.; et al. 2018. Hyperbolic attention networks. *arXiv preprint arXiv:1805.09786*.

Hao, J.; Lei, C.; Efthymiou, V.; Quamar, A.; Özcan, F.; Sun, Y.; and Wang, W. 2021. Medto: Medical data to ontology matching using hybrid graph neural networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2946–2954.

Jin, W.; Li, Y.; Xu, H.; Wang, Y.; Ji, S.; Aggarwal, C.; and Tang, J. 2021. Adversarial attacks and defenses on graphs. *ACM SIGKDD Explorations Newsletter*, 22(2): 19–34.

Jin, W.; Ma, Y.; Liu, X.; Tang, X.; Wang, S.; and Tang, J. 2020. Graph structure learning for robust graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 66–74.

Kipf, T. N.; and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Li, N.; Yang, Z.; Yang, Y.; Wang, J.; and Lin, H. 2023. Hyperbolic hierarchical knowledge graph embeddings for biological entities. *Journal of Biomedical Informatics*, 147: 104503.

Liu, Q.; Nickel, M.; and Kiela, D. 2019. Hyperbolic graph neural networks. *Advances in neural information processing systems*, 32.

McPherson, M.; Smith-Lovin, L.; and Cook, J. M. 2001. Birds of a feather: Homophily in social networks. *Annual review of sociology*, 27(1): 415–444.

Peng, W.; Varanka, T.; Mostafa, A.; Shi, H.; and Zhao, G. 2021. Hyperbolic deep neural networks: A survey. *IEEE Transactions on pattern analysis and machine intelligence*, 44(12): 10023–10044.

Qu, E.; and Zou, D. 2022. Hyperbolic neural networks for molecular generation. *arXiv preprint arXiv:2201.12825*.

Rahimi, A.; and Recht, B. 2007. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20.

Runwal, B.; Vivek; and Kumar, S. 2022. Robust Graph Neural Networks using Weighted Graph Laplacian. arXiv:2208.01853.

Sawhney, R.; Agarwal, S.; Wadhwa, A.; and Shah, R. 2021. Exploring the scale-free nature of stock markets: Hyperbolic graph learning for algorithmic trading. In *Proceedings of the Web Conference 2021*, 11–22.

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

Shimizu, R.; Mukuta, Y.; and Harada, T. 2020. Hyperbolic neural networks++. *arXiv preprint arXiv:2006.08210*.

Väisälä, J. 2005. Gromov hyperbolic spaces. *Expositiones Mathematicae*, 23(3): 187–231.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y.; et al. 2017. Graph attention networks. *stat*, 1050(20): 10–48550.

Wang, H.; Lian, D.; Tong, H.; Liu, Q.; Huang, Z.; and Chen, E. 2021a. Hypersorec: Exploiting hyperbolic user and item representations with multiple aspects for social-aware recommendation. *ACM Transactions on Information Systems (TOIS)*, 40(2): 1–28.

Wang, S.; Wei, X.; Dos Santos, C. N.; Wang, Z.; Nallapati, R.; Arnold, A.; and Philip, S. Y. 2021b. Knowledge graph representation via hierarchical hyperbolic neural graph embedding. In *2021 IEEE International Conference on Big Data (Big Data)*, 540–549. IEEE.

Wu, H.; Wang, C.; Tyshetskiy, Y.; Docherty, A.; Lu, K.; and Zhu, L. 2019. Adversarial examples on graph data: Deep insights into attack and defense. *arXiv preprint arXiv:1903.01610*.

Xu, H.; Ma, Y.; Liu, H.-C.; Deb, D.; Liu, H.; Tang, J.-L.; and Jain, A. K. 2020. Adversarial attacks and defenses in images, graphs and text: A review. *International Journal of Automation and Computing*, 17: 151–178.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*.

Yang, M.; Zhou, M.; Li, Z.; Liu, J.; Pan, L.; Xiong, H.; and King, I. 2022. Hyperbolic graph neural networks: a review of methods and applications. *arXiv preprint arXiv:2202.13852*.

Yu, T.; and De Sa, C. 2022. Random Laplacian Features for Learning with Hyperbolic Space. *arXiv preprint arXiv:2202.06854*.

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

Zhang, X.; and Zitnik, M. 2020. Gnnguard: Defending graph neural networks against adversarial attacks. *Advances in neural information processing systems*, 33: 9263–9275.

Zhang, Y.; Wang, X.; Shi, C.; Jiang, X.; and Ye, Y. 2021. Hyperbolic graph attention network. *IEEE Transactions on Big Data*, 8(6): 1690–1701.

Zhao, K.; Kang, Q.; Song, Y.; She, R.; Wang, S.; and Tay, W. P. 2023. Adversarial Robustness in Graph Neural Networks: A Hamiltonian Approach. arXiv:2310.06396.

Zitnik, M.; Sosič, R.; Feldman, M. W.; and Leskovec, J. 2019. Evolution of resilience in protein interactomes across the tree of life. *Proceedings of the National Academy of Sciences*, 116(10): 4426–4433.

Zügner, D.; Akbarnejad, A.; and Günnemann, S. 2018. Adversarial attacks on neural networks for graph data. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2847–2856.