Mitigating Over-squashing without Rewiring: A Sheaf Effective Resistance Perspective

André Ribeiro
Getulio Vargas Foundation
andre.guimaraes@fgv.br

Germano Barcelos
Federal University of Viçosa
gegenbarcelos@gmail.com

Amauri H. Souza
Federal Institute of Ceará, 2δ AI
amauriholanda@ifce.edu.br

Diego Mesquita
Getulio Vargas Foundation, 2δ AI
diego.mesquita@fgv.br

Ana Luiza Tenório
Getulio Vargas Foundation, UFRRJ
anatenorio@ufrrj.br

Abstract

Graph Neural Networks (GNNs) often struggle to capture long-range dependencies due to over-squashing — a phenomenon in which the repeated compression of node embeddings into finite-size messages causes representations to collapse. Over-squashing is most often diagnosed as a property of the graph topology, with effective resistance serving as a principled measure of the bottleneck. We provide a complementary view on the matter: building on cellular sheaves, we introduce sheaf effective resistance, a generalization of effective resistance that depends on the sheaf attached to the graph, and we prove that for flat vector bundles, the over-squashing sensitivity in the Jacobian sense is upper bounded by a quantity related to the sheaf effective resistance between the nodes. The bottleneck thus need not lie in the graph itself: it can be relocated, and reduced, by adjusting the sheaf. We instantiate this idea in FlatNSD, a simple message-passing variant of Neural Sheaf Diffusion, and show that it implicitly learns to modulate total sheaf effective resistance, performing well on benchmarks designed to stress over-squashing without altering the original graph topology.

1 Introduction

Graph neural networks (GNNs) have become the dominant approach for learning on relational data, with applications ranging from molecular chemistry [Duvenaud et al., 2015, Gilmer et al., 2017] to recommendation [Ying et al., 2018] and physical simulation [Sanchez-Gonzalez et al., 2020]. Their performance, however, is constrained by two well-documented pathologies: over-smoothing, where iterated message passing makes node embeddings nearly indistinguishable, and over-squashing, where information from exponentially many nodes is compressed into fixed-size vectors, hindering long-range dependencies. Recent work by Bodnar et al. [2022] introduced Neural Sheaf Diffusion (NSD), a generalization of Graph Convolutional Networks [Kipf and Welling, 2017] that uses cellular sheaves to mitigate over-smoothing while accommodating heterophilic graphs.

This raises a natural question: **can sheaf-based models also mitigate over-squashing?** The role of sheaves in this setting has been explored by Bamberger et al. [2025], who introduced flat vector bundles as a scalable parameterization of sheaves and proposed BuNN, a sheaf-inspired architecture

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

that steps outside the message-passing paradigm to address over-squashing. Building on their flat-bundle formulation, we show that the message-passing paradigm itself is sufficient to alleviate over-squashing, and we introduce FlatNSD as a simple NSD variant to support this claim.

Our analysis builds on a tool from the rewiring literature: effective resistance. The prevailing view treats over-squashing as a topological bottleneck. Through a Jacobian sensitivity analysis, Topping et al. [2022] linked the issue to edges of high negative curvature; Black et al. [2023] introduced effective resistance as a metric for over-squashing and proposed adding edges to reduce it; and Di Giovanni et al. [2023] further showed that over-squashing primarily affects distant nodes. Together, these works established graph topology as the main driver of over-squashing.

On the other hand, we have sheaves: a cellular sheaf $\mathcal{F}$ on a fixed graph is mathematical structure that assigns a vector space, the *stalk*, to each vertex and edge, together with linear *restriction maps* for each incident vertex–edge pair. Whereas Hansen and Gebhart [2020] considered fixed restriction maps, Bodnar et al. [2022] proposed learning them end-to-end, yielding a flexible model that adapts to the task. Information propagation in NSD is governed by the sheaf Laplacian, which generalizes the graph Laplacian and is determined by the restriction maps. This suggests a different perspective on over-squashing: the graph itself need not be modified; one can instead adjust the sheaf $\mathcal{F}$ assigned to it. Intuitively, restriction maps modulate the communication channels between nodes, so an appropriate choice of $\mathcal{F}$ can play the role traditionally assigned to edge rewiring. This marks a shift in perspective: **the bottleneck responsible for over-squashing need not lie in the graph, but in the sheaf attached to it**. Rather than asking *which edges to add/remove*, we ask *which restriction maps to learn*.

To make this precise, we generalize the classical effective resistance and introduce *sheaf effective resistance* $\mathrm{R}_{\mathcal{F}}(u, v)$, which quantifies the ease of transferring information between nodes u and v on a fixed graph. Our results show that lower $\mathrm{R}_{\mathcal{F}}(u, v)$ corresponds to less compression of messages (Theorem 4.1). Empirically, evaluations on synthetic data reveal a strong inverse relationship between total sheaf effective resistance and predictive accuracy (Figure 2). When using SAGE to learn the restriction maps, FlatNSD produces Laplacians with low sheaf effective resistance and correspondingly higher accuracy (Figure 1). Importantly, as we analyze in Section B.2, our modifications mitigate over-squashing without inducing over-smoothing — addressing a natural concern given the well-known tension between the two phenomena. Our theoretical results extend beyond FlatNSD to a broader class of MPNNs that we refer to as SNNs (see Equation (2)).

In summary, our main contributions are the following:

1. We derive closed-form equations for the sheaf effective resistance (Section 4) and prove that the sheaf effective resistance for flat vector bundles upper bounds the over-squashing sensitivity (Theorem 4.1).
2. Through synthetic experiments that control over-squashing, we support the theory observing that accuracy increases while sheaf effective resistance decreases (Figure 1 and Figure 2).
3. Empirical evaluations on real-world tasks show that FlatNSD offers a robust alternative, performing competitively alongside established rewiring and graph transformer baselines.
4. We identify key architectural choices driving SNN performance (i.e., the role of restriction-map parameterization and stalk dimension) providing guidance for future sheaf-based models.

2 Background

In this section, we briefly review cellular sheaves and their use in GNNs. Moreover, we overview how effective resistance was previously studied to avoid over-squashing.

Notation. Let $G = (V, E)$ be a connected, undirected, unweighted graph with n vertices and m edges. Let A be the adjacency matrix of G and D its degree matrix. The graph Laplacian of G is $L_G = D - A$ and its normalized version by $\Delta_G = I - \hat{A} = D^{-1/2} L_G D^{-1/2}$, with $\hat{A} = D^{-1/2} A D^{-1/2}$.

Cellular Sheaves. A cellular sheaf $\mathcal{F}$ over a graph G associates (i) vector spaces $\mathcal{F}(v)$ and $\mathcal{F}(e)$ to each vertex $v \in V$ and each edge $e \in E$, respectively; (ii) linear maps $\mathcal{F}_{v \trianglelefteq e} : \mathcal{F}(v) \to \mathcal{F}(e)$ to each incident vertex-edge pair $v \trianglelefteq e$. The vector spaces are called **stalks**, and the linear maps are called **restriction maps**. For simplicity, it is standard to assume that all stalks are isomorphic to $\mathbb{R}^d$. If $d = 1$ and all restriction maps are the identity map, we have the so called **trivial sheaf**.

In GNNs, diffusion is often governed by the graph Laplacian. In SNNs, the sheaf Laplacian plays an analogous role, but its definition involves several intermediate steps.

Sheaf Laplacian. We start by defining spaces of direct sums $C^0(G,\mathcal{F}) = \bigoplus_{v\in V}\mathcal{F}(v)$ and $C^1(G,\mathcal{F}) = \bigoplus_{e\in E}\mathcal{F}(e)$ to consider a linear map $\delta : C^0(G,\mathcal{F}) \to C^1(G,\mathcal{F})$ given by $(\delta\mathbf{X})_e = \mathcal{F}_{u\trianglelefteq e}\mathbf{x}_u - \mathcal{F}_{v\trianglelefteq e}\mathbf{x}_v$ for each $e \in E$, where an orientation $e = v \to u$ is chosen, and $\mathbf{x}_v$ refers to a vector in $\mathcal{F}(v)$, i.e. a row in $\mathbf{X}$. When the sheaf is trivial, this operator coincides with $B^\top$, where B is the incidence matrix of G. Then, the sheaf Laplacian is defined by $\delta^\top\delta$, generalizing the Laplacian description as $BB^\top$. Equivalently, we also have that the **sheaf Laplacian** of $\mathcal{F}$ is the linear operator $L_\mathcal{F} : C^0(G,\mathcal{F}) \to C^0(G,\mathcal{F})$ given by $L_\mathcal{F}(\mathbf{X})_v := \sum_{v,u\trianglelefteq e}\mathcal{F}^\top_{v\trianglelefteq e}\left(\mathcal{F}_{v\trianglelefteq e}\mathbf{x}_v - \mathcal{F}_{u\trianglelefteq e}\mathbf{x}_u\right), \forall v \in V$.

The sheaf Laplacian $L_\mathcal{F}$ is a positive semidefinite matrix with diagonal blocks $(L_\mathcal{F})_{vv} = \sum_{v\trianglelefteq e}\mathcal{F}^\top_{v\trianglelefteq e}\mathcal{F}_{v\trianglelefteq e}$ and non-diagonal blocks $(L_\mathcal{F})_{vu} = (L_\mathcal{F})^\top_{vu} = -\mathcal{F}^\top_{v\trianglelefteq e}\mathcal{F}_{u\trianglelefteq e}$. It can also be written as $L_\mathcal{F} = D_\mathcal{F} - A_\mathcal{F}$, where $D_\mathcal{F}$ is the block-diagonal sheaf-degree matrix, and $A_\mathcal{F}$ the sheaf-adjacency matrix. Its normalized version is $\Delta_\mathcal{F} = D_\mathcal{F}^{-\frac{1}{2}}L_\mathcal{F}D_\mathcal{F}^{-\frac{1}{2}}$.

Neural Sheaf Diffusion (NSD). One of the first, and currently the most popular SNN model was introduced in Bodnar et al. [2022] as follows:

$$\mathbf{H}_{t+1} = \mathbf{H}_t - \sigma(\Delta_{\mathcal{F}(t)}(\mathbf{I}\otimes\mathbf{W}_{1,t})\mathbf{H}_t\mathbf{W}_{2,t}), \tag{1}$$

where $\mathbf{H}_0 \in \mathbb{R}^{nd\times h}$ is the matrix of initial node features projected into h channels, $\mathbf{W}_{1,t}$ and $\mathbf{W}_{2,t}$ are weight square matrices of dimensions d and h, respectively, $\otimes$ is the Kronecker product, and σ is an activation. Furthermore, the Laplacian is learned locally through the graph features: each restriction map $F_{v\trianglelefteq e}$ is learned using a function $\Phi : \mathbb{R}^{2d} \to \mathbb{R}^{d\times d}$, with $\mathcal{F}_{v\trianglelefteq e:=(v,u)} = \Phi(\mathbf{x}_v,\mathbf{x}_u)$.

Classes of sheaves. Bodnar et al. [2022] studied three classes of sheaves based on the properties of their restriction maps: diagonal, orthogonal, and general (i.e, any invertible linear map). The best balance between scalability and empirical performance was achieved by orthogonal restriction maps. We call this class of sheaves **vector bundles**. More recently, further improving the computational complexity of SNNs, Bamberger et al. [2025] presented **flat vector bundles** as part of their Bundle Neural Networks (BuNNs) model. They consider the special case of flat vector bundles in which to every node v is assigned an orthogonal map $\mathcal{F}_v$, which replaces $\mathcal{F}_{v\trianglelefteq e}$ for any adjacent edge e. Instead of building the model directly over the sheaf Laplacian, BuNN uses the bundle heat kernel.

Effective Resistance. Originally defined by Kirchhoff [1847], effective resistance is the voltage difference between two points in an electrical network when a unit current is applied. In the realm of GNNs, it serves as a measure of proximity and connectivity within a graph, and it was recently incorporated to mitigate over-squashing [Black et al., 2023]. Formally, the effective resistance between nodes u and v is defined by $\mathrm{R}(u,v) = (1_u - 1_v)^\top L_G^+(1_u - 1_v)$, where 1_v is an indicator vector, i.e., a binary vector of length n and in this vector, only the entry corresponding to node v is 1, and L^+ is the pseudo-inverse of the graph Laplacian.

Over-squashing. The connection between over-squashing and effective resistance passes through the work from Topping et al. [2022], which proposed an upper bound on the 2-norm of the Jacobian between node features, $\|\partial h_u^{(r)}/\partial x_v\|$, with $h_u^{(r)}$ and x_v both being vectors. This Jacobian quantifies how the initial feature x_v at node v affects $h_u^{(r)}$ at node u in the r-th GNN layer. Then, Black et al. [2023] proved that if $\mathrm{R}(u,v)$ decreases, then the upper bound increases, allowing for an improved information flow between u and v. More precisely, using ϕ_t and ψ_t to denote, respectively, learnable update and message functions at layer t, they obtained:

Theorem 2.1 (Theorem 3.3 Black et al. [2023])**.** *Let G be a non-bipartite graph, and let $u,v \in V$. Suppose $\|\nabla\phi_l\| \leqslant \alpha$ and $\max\{\|\nabla\psi_l\|,1\} \leqslant \beta$. Define $d_{\min} = \min\{d_u,d_v\}$ and $d_{\max} = \max\{d_u,d_v\}$. Let $\max\{|\mu_2|,...,|\mu_n|\} \leqslant \mu$. Then*

$$\left\|\frac{\partial h_u^{(r)}}{\partial x_v}\right\| \leqslant (2\alpha\beta)^r\frac{d_{max}}{2}\left(\frac{2}{d_{min}}\left(r+1+\frac{\mu^{r+1}}{1-\mu}\right) - \mathrm{R}(u,v)\right).$$

Black et al. [2023] rewiring technique adds new edges between certain nodes, aiming to decrease the total effective resistance — the sum of the effective resistances between all pairs of nodes. This brings significant improvements in empirical results when compared with GNNs with no rewiring.

Rewiring is a standard method to address over-squashing by improving the connectivity of a graph and potentially easing structural bottlenecks. Its general goal is to create new connections that enable a more efficient and balanced flow of information within the network. We focus on the comparison with Black et al. [2023] because they also work with effective resistance but other strategies can be applied: for instance, spectral gap [Karhadkar et al., 2023, Banerjee et al., 2022], or different notions of curvature [Fesser and Weber, 2024, Giraldo et al., 2023].

3 Sheaf Neural Networks via Message Passing

We analyze over-squashing through the lens of the Jacobian, in the same fashion as Theorem 2.1. Therefore, we need to describe the NSD model in the message passing framework.

Recall that for a graph whose nodes u have neighborhoods $N_u = \{v \in V \mid (v,u) \in E\}$ and features $\mathbf{h}_u^{(0)} = \mathbf{x}_u$, a message passing layer is defined as $\mathbf{h}_v^{(t+1)} = \phi_t\Big(\mathbf{h}_v^{(t)}, \bigoplus_{u \in N_v} \psi_t\left(\mathbf{h}_u^{(t)}, \mathbf{h}_v^{(t)}\right)\Big)$, where ψ_t is the **message function**, ϕ_t is the **update function**, and $\oplus$ is a permutation-invariant aggregation function.

The NSD model described in Equation (1) is a message-passing GNN where the message function is updated for each layer since it is given by $\psi_t(\mathbf{h}_u, \mathbf{h}_v) = D_v^{-\frac{1}{2}} \mathcal{F}_{v \trianglelefteq e}^\top \left(\mathcal{F}_{v \trianglelefteq e} D_v^{-\frac{1}{2}} \mathbf{h}_v - \mathcal{F}_{u \trianglelefteq e} D_u^{-\frac{1}{2}} \mathbf{h}_u\right)$, if we ignore the weights, and the aggregation is the sum. This simple observation provides significantly more efficient implementation and accommodates directed graph inputs, which the original code from Bodnar et al. [2022] does not. Moreover, if $\mathcal{F}$ is a flat vector bundle, then

$$D_v^{-\frac{1}{2}} \mathcal{F}_v^\top \mathcal{F}_u D_u^{-\frac{1}{2}} = \frac{1}{\sqrt{d_u d_v}} \mathcal{F}_v^\top \mathcal{F}_u = \hat{A}_{uv} \mathcal{F}_v^\top \mathcal{F}_u.$$

Thus, we can write the FlatNSD model, which is NSD restricted to flat vector bundles, as

$$\mathbf{h}_v^{(t+1)} = \phi_t \left(\mathbf{h}_v^{(t)}, \sum_{u \in N_v} \hat{A}_{uv} \psi_t \left(\mathbf{h}_u^{(t)} \right) \right), \tag{2}$$

where $\psi_t(\mathbf{h}_u) = -\mathcal{F}_u \mathbf{h}_u$, up to weight matrices, $\hat{A}_{uv}$ is the normalized adjacency matrix of the graph, and $\mathcal{F}_v^\top$ is applied by ϕ_t, since $\mathcal{F}_v$ is linear. Therefore, the FlatNSD model is a Message Passing Neural Network (MPNN) that fits in the framework explored by Black et al. [2023], allowing us to apply some of their results. In the following sections, we show that restricting attention to flat vector bundles yields additional computational and theoretical benefits. In Section C, we provide a summary with our main contributions, distinguishing what is novel from prior work. We also provide a comparison of FlatNSD with existing sheaf architectures to better highlight its role as a tool to empirically verify our theoretical claims rather than a novel model.

4 Sheaf effective resistance

We introduce sheaf effective resistance (sheaf ER) and theoretically connect it with over-squashing.

Definition 4.1. The sheaf effective resistance between nodes u and v is defined by

$$\mathrm{R}_{\mathcal{F}}(u,v) = (\mathbb{1}_u - \mathbb{1}_v)^\top L_{\mathcal{F}}^+ (\mathbb{1}_u - \mathbb{1}_v),$$

where $\mathbb{1}_v$ is a vector of dimension nd where the d entries corresponding to node v is 1. $L_{\mathcal{F}}^+$ is the pseudo-inverse of the sheaf Laplacian.

If $\mathcal{F}$ is the trivial sheaf, then $d = 1$ and sheaf effective resistance coincides with effective resistance.

Example 4.1. *Consider the graph with only two vertices u and v, and an edge e between them.*

1. *If $\mathcal{F}$ is the trivial sheaf, then $R_{\mathcal{F}}(u,v) = R(u,v) = 1$.*
2. *If $\mathcal{F}$ is a sheaf such that all stalks are isomorphic to $\mathbb{R}^2$, $\mathcal{F}_{u \trianglelefteq e} = \begin{pmatrix} \cos(x) & -\sin(x) \\ \sin(x) & \cos(x) \end{pmatrix}$, and $\mathcal{F}_{v \trianglelefteq e} = \begin{pmatrix} \cos(y) & -\sin(y) \\ \sin(y) & \cos(y) \end{pmatrix}$, then $R_{\mathcal{F}}(u,v) = \cos(x-y) + 1$.*

In Section E, we provide more details to perform the calculations above. Observe that **by simply changing the sheaf, without changing the graph, we can reduce or increase the effective resistance** depending only on the restriction maps. Larger worked examples can be found in Section A

From now on, we focus on flat vector bundles, deriving formulas for their effective resistance and proving the relationship with over-squashing sensitivity. Importantly, this restriction is highly practical: prior work shows vector bundles optimally balance scalability and performance [Bodnar et al., 2022], with the *flat* constraint further enhancing efficiency [Bamberger et al., 2025]. Additionally, our preliminary experiments with general sheaves revealed significant numerical instability. Thus, flat vector bundles emerge as the most robust and practical choice for sheaf-based networks.

First, observe that for flat vector bundles, the sheaf Laplacian can be written as $L_{\mathcal{F}} = \mathbf{O}^\top (L_G \otimes I_d)\mathbf{O}$, where $\mathbf{O} = \mathrm{blockdiag}(\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_n) \in \mathbb{R}^{nd \times nd}$ [Bamberger et al., 2025]. In the following, we consider a flat vector bundle $\mathcal{F}$ over G and L_G the graph Laplacian.

Proposition 4.1. *The pseudo-inverse of $L_{\mathcal{F}}$ is $L_{\mathcal{F}}^+ = \mathbf{O}^\top (L_G^+ \otimes I_d)\mathbf{O}$.*

Crucially, this provides the following formula to compute $R_{\mathcal{F}}(u, v)$.

Proposition 4.2. *Let $\mathbf{1}_d \in \mathbb{R}^d$ be a vector of ones, and $M_{uv} = \mathbf{1}_d^\top \mathcal{F}_u^\top \mathcal{F}_v \mathbf{1}_d$. Then*

$$\mathrm{R}_{\mathcal{F}}(u, v) = d((L_G^+)_{uu} + (L_G^+)_{vv}) - 2M_{uv}(L_G^+)_{uv}.$$

If $\mathcal{F}$ is the trivial sheaf, we obtain $\mathrm{R}(u, v) = (L_G^+)_{uu} + (L_G^+)_{vv} - 2(L_G^+)_{uv}$. So we have generalized a classical result. Importantly, this also allows us to write $\mathrm{R}_{\mathcal{F}}(u, v)$ in terms of $\mathrm{R}(u, v)$:

$$\mathrm{R}_{\mathcal{F}}(u, v) = d\,\mathrm{R}(u, v) + 2(d - M_{uv})(L_G^+)_{uv}. \tag{3}$$

Note that computing $R_{\mathcal{F}}$ requires only a single pseudo-inverse calculation, as the sheaf effective resistance for any flat vector bundle $\mathcal{F}$ depends on the pseudo-inverse of the **graph** Laplacian, which remains unchanged since we do not rewire the graph.

Theorem 4.1 below formalizes the relationship between sheaf effective resistance and over-squashing.

Theorem 4.1. *Let G be a non-bipartite graph, and let $u, v \in V$. Suppose $\|\nabla \phi_l\| \leqslant \alpha$ and $\max\{\|\nabla \psi_t\|, 1\}| \leqslant \beta$. Define $d_{\min} = \min\{d_u, d_v\}$, $d_{\max} = \max\{d_u, d_v\}$, and fix d the dimension of the stalks. Let $\max\{|\mu_2|, ..., |\mu_n|\} \leqslant \mu$. If $(L_G^+)_{uv} \geqslant 0$, then*

$$\left\| \frac{\partial h_u^{(r)}}{\partial x_v} \right\| \leqslant (2\alpha\beta)^r \frac{d_{max}}{2} \left(\frac{2}{d_{min}} \left(r + 1 + \frac{\mu^{r+1}}{1 - \mu} \right) + 4(L_G^+)_{uv} - \frac{\mathrm{R}_{\mathcal{F}}(u, v)}{d} \right).$$

If $(L_G^+)_{uv} < 0$, then

$$\left\| \frac{\partial h_u^{(r)}}{\partial x_v} \right\| \leqslant (2\alpha\beta)^r \frac{d_{max}}{2} \left(\frac{2}{d_{min}} \left(r + 1 + \frac{\mu^{r+1}}{1 - \mu} \right) - \frac{\mathrm{R}_{\mathcal{F}}(u, v)}{d} \right).$$

Theorem 4.1 shows that the propagation of messages can be controlled by the sheaf effective resistance. If the sheaf is fixed (the absence of a sheaf is equivalent to fixing the trivial sheaf), we must alter the graph to decrease the effective resistance. However, with varying sheaves, we can keep the original graph and decrease the sheaf effective resistance. Moreover, this provides a perspective change: **effective resistance measures communication capacity of the sheaf associated to the graph rather than to the graph itself.**

To align theory with experiments, we calculate total sheaf effective resistance $\mathrm{R}_{\mathcal{F}}$, i.e., the sum of the sheaf effective resistance $\mathrm{R}_{\mathcal{F}}(u, v)$ for all pairs of nodes. In the following, R denotes the total (graph) effective resistance.

Theorem 4.2. *Let $\langle \cdot, \cdot \rangle_F$ denote the Frobenius inner product and M as in Proposition 4.2. Then*

$$\mathrm{R}_{\mathcal{F}} = dn\,\mathrm{Tr}(L_G^+) - \langle M, L_G^+ \rangle_F = d\,\mathrm{R} - \langle M, L_G^+ \rangle_F.$$

Importantly, if $\mathcal{F}$ is the trivial sheaf, we obtain $\mathrm{R} = n\,\mathrm{Tr}(L_G^+)$. This is another classical result, so our approach is well-founded. We use the above in our implementation to analyze the total sheaf effective resistance in our experiments, using $M = SS^\top$, where $S_u = \mathbf{1}_d^\top \mathcal{F}_u^\top$. This allows us to calculate the Frobenius inner product using the cyclic property of the trace while avoiding building M. Furthermore, we characterize when flat bundles strictly decrease sheaf effective resistance.

Proposition 4.3. *Let G be a connected graph with at least two nodes and $\mathcal{F}$ a flat $O(d)$-bundle over G. Then $\frac{R_{\mathcal{F}}}{d} \leqslant R$, with equality holding if and only if $\mathcal{F}_v \mathbf{1}_d = c$ for some $c \in \mathbb{R}^d$ and all $v \in V$. In particular, for every $d \geqslant 1$, there exists a flat $O(d)$-bundle $\mathcal{F}$ such that $\frac{R_{\mathcal{F}}}{d} < R$.*

We proceed with a corollary of Theorem 4.1 regarding the total sheaf effective resistance.

Corollary 4.1. *Let G be a non-bipartite graph, and let $u, v \in V$. Suppose $\|\nabla\phi_l\| \leqslant \alpha$ and $\max\{\|\nabla\psi_t\|, 1\}| \leqslant \beta$. Let $d_{\min} = \min\{d_u, d_v\}$, $d_{\max} = \max\{d_u, d_v\}$, and d be the minimum stalk dimension. Let $\max\{|\mu_2|, ..., |\mu_n|\} \leqslant \mu$. Then*

$$\sum_{u \neq v} \left\| \frac{\partial h_u^{(r)}}{\partial x_v} \right\| \leqslant (2\alpha\beta)^r \frac{d_{max}}{2} \left(\frac{n(n-1)}{d_{min}} \left(r + 1 + \frac{\mu^{r+1}}{1-\mu} \right) - \frac{\mathrm{R}_{\mathcal{F}}}{d} \right).$$

Importantly, FlatNSD does not require computing the sheaf ER or any pseudo-inverse. Rather, we use the sheaf ER as an analytical tool to understand the performance of our FlatNSD model, since $\mathrm{R}_{\mathcal{F}}(u, v)$ is related to $\left\| \partial h_u^{(r)} / \partial x_v \right\|$, which measures sensitivity to over-squashing.

In the previous results, we assumed non-bipartite graphs, similar to Black et al. [2023]. We now provide extensions relating graph ER to the Jacobian of MP-based GNNs as in Equation (2) to bipartite graphs, with natural extensions to sheaf ER. We begin by constructing a formula for the Laplacian pseudo-inverse on bipartite graphs.

Lemma 4.1. *Let G be a connected bipartite graph. Then $\Delta_G^+ = \frac{1}{2}P_- + \sum_{k=0}^{\infty}(\hat{A}^k - P_+ - (-1)^k P_-)$, where P_+ and P_- are the outer product of the eigenvectors with eigenvalues 1 and -1, respectively.*

The next step is to find a formula for the effective resistance of bipartite graphs.

Proposition 4.4. *Let G be a connected bipartite graph, $b_{uv} = \mathbb{1}_u - \mathbb{1}_v$, $z_{uv} = D^{-1/2} b_{uv}$, and $\gamma_{uv} = z_{uv}^\top P_- z_{uv}$. Then*

$$R(u, v) = \frac{\gamma_{uv}}{2} + \sum_{k=0}^{\infty} \left(\frac{(\hat{A}^k)_{uu}}{d_u} + \frac{(\hat{A}^k)_{vv}}{d_v} - \frac{2(\hat{A}^k)_{uv}}{\sqrt{d_u d_v}} - (-1)^k \gamma_{uv} \right)$$

Now, we proceed to relate the above formula for the effective resistance with the Jacobian bound.

Theorem 4.3. *Let G be a connected bipartite graph. Let $u, v \in V = U \sqcup W$, $\|\nabla\phi_l\| \leqslant \alpha$ and $\max\{\|\nabla\psi_l\|, 1\| \leqslant \beta$, $d_{\min} = \min\{d_u, d_v\}$, $d_{\max} = \max\{d_u, d_v\}$, and $\overline{\mu} = \max_{i \neq 1} |\mu_i| < 1$. Then*

$$\left\| \frac{\partial h_u^{(r)}}{\partial x_v} \right\| \leqslant (2\alpha\beta)^r \frac{d_{\max}}{2} \left(\frac{2}{d_{\min}} \left(r + 1 + \frac{\overline{\mu}^{r+1}}{1-\overline{\mu}} \right) + \frac{1}{2}\gamma_{uv} - R(u, v) \right).$$

The corollary for the total effective resistance follows from this theorem, where the additional term accounting for bipartiteness becomes $\frac{4|U||W|}{\mathrm{vol}(G)}$ (see Remark E.1).

Remark 4.1 (Geometric interpretation under isospectrality). A natural question arises from the decomposition $L_{\mathcal{F}} = \mathbf{O}^\top (L_G \otimes I_d)\mathbf{O}$: how can FlatNSD mitigate over-squashing if its diffusion operator has essentially the same spectral properties as the graph Laplacian? The key lies in the rotations performed by the restriction maps. Define $\xi = \frac{\mathbf{1}_d}{\sqrt{d}}$ and $g_v = \mathcal{F}_v \xi \in \mathbb{S}^{d-1}$. Then

$$M_{uv} = \mathbf{1}_d^\top \mathcal{F}_u^\top \mathcal{F}_v \mathbf{1}_d = d\xi^\top \mathcal{F}_u^\top \mathcal{F}_v \xi = d g_u^\top g_v = d\cos\theta_{uv},$$

where θ_{uv} is the angle between g_u and g_v. Thus, Proposition 4.2 can be rewritten as $\mathrm{R}_{\mathcal{F}}(u, v) = d\big((L_G^+)_{uu} + (L_G^+)_{vv} - 2\cos(\theta_{uv})(L_G^+)_{uv}\big)$. When $g_u = g_v$, we recover the effective resistance of the trivial d-dimensional flat bundle. If $g_u \perp g_v$, the cross-term disappears, reducing effective resistance relative to the aligned configuration when $(L_G^+)_{uv} < 0$. In this case, $g_u = -g_v$ gives the largest reduction. When $(L_G^+)_{uv} > 0$, alignment minimizes the effective resistance.

Conclusion. Sheaf effective resistance upper bounds over-squashing, as measured by the Jacobian sensitivity. This suggests that learned sheaves can enhance representational capacity compared to the trivial sheaves provided by the data.

5 Synthetic Experiments

In this section, we test FlatNSD on three different synthetic datasets designed to address over-squashing. Our goal is to endorse that a simple SNN such as FlatNSD can alleviate it. We provide experiments on the short-range over-squashing regime [Mishayev et al., 2025] in Section B.5.

TreeNeighborsMatch. To test FlatNSD ability to deal with over-squashing, we reproduce the TreeNeighborsMatch problem proposed by Alon and Yahav [2021], using the same framework. The TreeNeighborsMatch is a synthetic dataset aimed at evaluating a GNN's ability to handle over-squashing, assessed through training performance. The underlying graph is a binary tree of depth r, and the task requires propagating information from the leaves to the root. Both the root and each leaf are attached to a set of *blue neighbors*. The target label is determined by the unique leaf whose number of blue neighbors matches that of the root. We provide the parameters used for this task in Table 4. In Figure 1, we evaluate FlatNSD under two parameterizations of the restriction maps: an MLP (as in NSD) and SAGE (as in BuNN and CSNN Ribeiro et al. [2026]). FlatNSD-SAGE outperforms all classical baselines (GCN, GIN, GAT, GGNN) as well as BuNN, matching the perfect accuracy of CSNN, a concurrent sheaf-based model. As a point of comparison, NSD with orthogonal maps attains 0.91 accuracy at $r = 3$ and 0.05 at $r = 4$. To better understand how and why FlatNSD improves performance on this task, we provide an additional experiment with a controlled intervention on this task in Section B.6

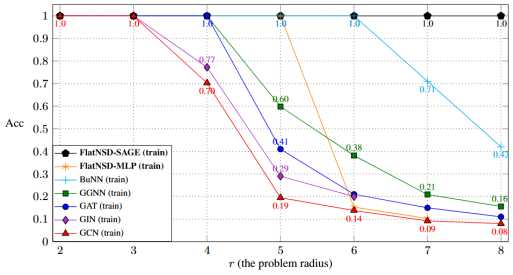


Figure 1: Training accuracy for the TreeNeighborsMatch task. Results for GCN, GIN, GAT, and GGNN are from Alon and Yahav [2021]. Results for BuNN are from Bamberger et al. [2025]. Only FlatNSD-SAGE achieves perfect accuracy for all r.

Conclusion. Restriction map parametrization plays a major role in NSD's performance and its ability to mitigate over-squashing. In particular, using flat bundles and structure-aware parameterizations (e.g., via GNNs) significantly boosts NSD's performance in long-range dependency tasks.

Effective Resistance. By Theorem 4.1, lower sheaf ER reduces sensitivity to over-squashing. Consequently, we expect a strong inverse relationship between accuracy and total ER in the TreeNeighborsMatch task, measured via Spearman correlation (Figure 2). For $r = 8$, FlatNSD-SAGE reaches 100% accuracy using both $d = 2$ and $d = 4$. However, $d = 4$ converges much faster (25 steps vs. 143 for $d = 2$) and maintains a strong inverse correlation across all layers. The weaker correlation for $d = 2$ reflects its slower convergence, indicating the model struggles to approximate an optimal sheaf. Full results for all r are detailed in Section B.4, together with Spearman correlation between Jacobian norm and total sheaf ER.

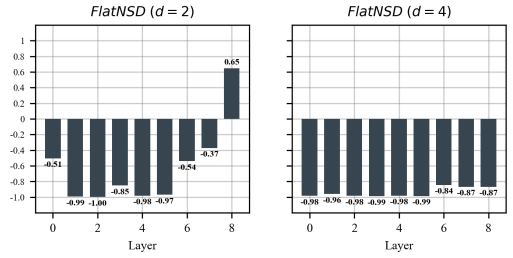


Figure 2: Correlation between Train Accuracy and Sheaf ER on FlatNSD-SAGE with $r = 8$ for $d = 2$ and $d = 4$. The network struggles to find an optimal sheaf when $d = 2$, leading to higher variance for the ER throughout the layers. For $d = 4$, the effective resistance is more uniform throughout the layers, while also achieving faster convergence.

We further substantiate that FlatNSD mitigates over-squashing by testing its performance on ECHO-Synth [Miglior et al., 2025] and Graph Transfer [Di Giovanni et al., 2023], both capturing long-range interactions.

Table 1: Performance (MAE) of FlatNSD. Darker green indicates lower error.

Model	`diam` ↓	`ecc` ↓	`sssp` ↓
A-DGN	1.151 ± 0.038	4.981 ± 0.037	1.176 ± 0.140
DRew	1.243 ± 0.047	4.651 ± 0.020	1.279 ± 0.011
GraphCON	2.969 ± 0.189	5.474 ± 0.001	5.734 ± 0.011
GCN	3.832 ± 0.262	5.233 ± 0.034	2.102 ± 0.094
GCNII	2.005 ± 0.093	5.241 ± 0.030	2.128 ± 0.429
GIN	1.630 ± 0.161	4.869 ± 0.092	2.234 ± 0.271
GPS	2.160 ± 0.098	4.758 ± 0.021	0.472 ± 0.050
GRIT	1.014 ± 0.046	5.091 ± 0.158	0.121 ± 0.013
PH-DGN	1.627 ± 0.398	5.068 ± 0.126	1.323 ± 0.485
SWAN	1.121 ± 0.070	4.840 ± 0.045	0.896 ± 0.232
CSNN	$\mathbf{1.001} \pm 0.068$	4.527 ± 0.022	$\mathbf{0.044} \pm 0.036$
$\text{FlatNSD}_{\text{SAGE}}$	1.132 ± 0.095	$\mathbf{3.138} \pm 0.052$	0.198 ± 0.082

ECHO-Synth. The ECHO-Synth dataset challenges GNNs with long-range tasks across six topological families (diameters 17–40). Since this setting induces different types of structural compression, it is ideal for evaluating FlatNSD's capacity to alleviate over-squashing. Following Miglior et al. [2025], we benchmark FlatNSD against their original models, with the addition of CSNN [Ribeiro et al., 2026] to assess the capacity of a recent, more engineered sheaf model. We use the same experimental setup and hyperparameter selection routine (details in Section F).

Results. Table 1 reports MAE across the three property prediction tasks. FlatNSD$_{\text{SAGE}}$ demonstrates reliable effectiveness in handling long-range dependencies, establishing an impressive performance on the `ecc` task, outperforming the next best baseline (CSNN) by over 30%. Furthermore, it consistently ranks among the top performers on the remaining tasks, securing the third-best result on `sssp` and remaining highly competitive on `diam`. These results confirm FlatNSD's capacity to effectively alleviate over-squashing across diverse structural bottlenecks. Furthermore, CSNN is also effective in these tasks, achieving the lowest MAE on `diam` and `sssp`, highlighting the effectiveness of a more intricate sheaf-based model regarding over-squashing mitigation.

Graph Transfer. We consider three graph topologies (CrossedRing, Ring, and CliquePath), each parameterized by the shortest-path distance r between a designated source–target pair. The target node is assigned a one-hot label, and every other node receives a constant unitary feature. The model is trained to reproduce the target's label at the source. Since information must travel a distance r through the graph, this task isolates over-squashing as the dominant failure mode, with accuracy expected to decrease as r grows.

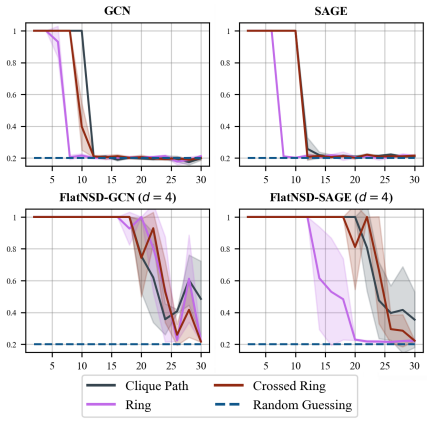


Figure 3: FlatNSD delays accuracy drop for all three topologies, demonstrating its ability of alleviating over-squashing.

Results. Figure 3 shows that with FlatNSD accuracy decreases after longer distances from source to target, in comparison with GCN and SAGE, for all three topologies. In Section B.3, we show that FlatNSD also outperforms GIN and GAT, and discuss that for $d = 4$ the performance is better than for $d = 2$, suggesting the importance of stalk dimension on tasks sensitive to over-squashing, in alignment with our results for the TreeNeighborsMatch (Figure 1).

Summary. FlatNSD was designed as a simple message-passing sheaf neural network that theoretically and empirically mitigates over-squashing across a diverse range of graph topologies. Rather than pursuing state-of-the-art performance, our overarching goal is to highlight SNNs as a fundamentally robust and promising direction for future architecture design.

6 Real-world experiments

Node Classification. To further show that sheaf models are a robust alternative, we also evaluate FlatNSD on the five heterophilic graphs proposed by Platonov et al. [2023]. For binary classification datasets, we report AUROC, while for multiclass datasets we report accuracy.

Table 2: Performance comparison on datasets from Platonov et al. [2023]. AUROC is reported for minesweeper, tolokers and questions, accuracy is reported for the remaining datasets.

Model Hom level	roman-empire 0.05	amazon-ratings 0.38	minesweeper 0.68	tolokers 0.59	questions 0.84
GCN	73.69 ± 0.74	48.70 ± 0.63	89.75 ± 0.52	83.64 ± 0.67	76.09 ± 1.27
SAGE	85.74 ± 0.67	53.63 ± 0.39	93.51 ± 0.57	82.43 ± 0.44	76.44 ± 0.62
GAT	80.87 ± 0.30	49.09 ± 0.63	92.01 ± 0.68	83.70 ± 0.47	77.43 ± 1.20
GAT-sep	88.75 ± 0.41	52.70 ± 0.62	93.91 ± 0.35	83.78 ± 0.43	76.79 ± 0.71
GT	86.51 ± 0.73	51.17 ± 0.66	91.85 ± 0.76	83.23 ± 0.64	77.95 ± 0.68
GT-sep	87.32 ± 0.39	52.18 ± 0.80	92.29 ± 0.47	82.52 ± 0.92	78.05 ± 0.93
BuNN	91.75 ± 0.39	**53.74 ± 0.51**	98.99 ± 0.16	84.78 ± 0.80	78.75 ± 1.09
CSNN	**92.63 ± 0.50**	52.07 ± 1.00	**99.07 ± 0.25**	**85.45 ± 0.53**	**79.31 ± 1.22**
FlatNSD$_{\text{SAGE}}$	92.26 ± 0.46	50.51 ± 0.70	98.45 ± 0.34	84.90 ± 0.83	76.44 ± 1.25

Following Platonov et al. [2023], who showed that classical GNNs often match or outperform architectures specifically designed for heterophilic graphs, we adopt their reference baselines: GCN [Kipf and Welling, 2017], GraphSAGE [Hamilton et al., 2017], GAT [Veličković et al., 2017], and GT [Shi et al., 2020], together with the GAT-sep and GT-sep variants, which concatenate a node's representation with the mean of its neighbors rather than summing them [Zhu et al., 2020]. We also compare FlatNSD against the recently proposed sheaf models BuNN [Bamberger et al., 2025], and CSNN [Ribeiro et al., 2026]. All three sheaf models considered use SAGE to learn the restriction maps.

Result. As shown in Table 2, sheaf-based models are uniformly robust on heterophilic datasets. While more sophisticated variants surpass FlatNSD — BuNN by departing from the message-passing paradigm, and CSNN by coupling two cooperative sheaves to refine node communication within message passing — FlatNSD remains within close range of both. This supports our broader claim: the message-passing paradigm itself, when equipped with appropriate restriction maps, is already sufficient to alleviate over-squashing.

Graph classification. We validate our theoretical claims through the Spearman correlation of sheaf effective resistance and accuracy on graph classification graphs Enzymes, IMDB, Mutag, and Proteins from the TUDatasets Morris et al. [2020]. Our objective is to elucidate how cellular sheaves can alleviate over-squashing without changing the topology of the graph, but rather the structure of its underlying sheaf. In Table 3 we conduct the same analysis for small graphs with a node classification task for Cora, Citeseer Yang et al. [2016], Texas, Cornell, Wisconsin Pei et al. [2020], and Chameleon Rozemberczki et al. [2021].

Experimental setup. Aiming to prioritize fairness and understandability instead of obtaining the highest accuracy, we follow Nguyen et al. [2023], Liang et al. [2025], that tune only rewiring-related hyperparameters, and freeze GNN-specific parameters and tune only the sheaf-related ones. We compare FlatNSD against GCN, GIN, SAGE, and seven rewiring methods using these three GNNs as backbone models: SDRF Topping et al. [2022], FOSR Karhadkar et al. [2023], BORF Nguyen et al. [2023], DR Attali et al. [2024], GTR Black et al. [2023], LASER Barbero et al. [2024], and GOKU Liang et al. [2025]. Section F provides details on our experimental setup.

Effective Resistance. We use Spearman rank correlation coefficient to measure the correlation between accuracy and effective resistance. Since real-world data faces multiple challenges beyond requiring over-squashing mitigation, we do not expect the inverse relation to be as strong as in the controlled synthetic experiment reported in Figure 2. Nevertheless, the effective resistance is considerably correlated with training accuracy in these datasets, in alignment with the theory we developed.

Figure 4: Performance of FlatNSD and GCN, GIN, and SAGE with different rewiring methods for both node and graph classification tasks. The method with the highest accuracy is highlighted in **bold**. DR requires graphs to have at least 3 nodes, but some graphs from Enzymes have only 2.

Method	Enzymes	IMDB	Mutag	Proteins	Rel Avg Imp
GCN+NONE	25.5 ± 1.3	49.3 ± 1.0	68.8 ± 2.1	70.6 ± 1.0	-
GIN+NONE	33.5 ± 1.3	67.7 ± 1.4	76.1 ± 3.1	69.5 ± 1.4	-
SAGE+NONE	31.5 ± 1.3	46.9 ± 0.8	72.2 ± 1.9	70.9 ± 0.9	-
GCN+SDRF	26.1 ± 1.1	49.1 ± 0.9	70.5 ± 2.1	71.4 ± 0.8	+1.39%
GIN+SDRF	32.4 ± 1.3	69.4 ± 1.4	79.5 ± 2.6	71.4 ± 0.8	+1.61%
SAGE+SDRF	32.3 ± 1.5	47.1 ± 0.8	78.2 ± 1.7	69.9 ± 0.8	+2.47%
GCN+FOSR	27.4 ± 1.1	49.6 ± 0.8	75.6 ± 1.7	72.3 ± 0.9	+5.09%
GIN+FOSR	28.8 ± 1.0	70.6 ± 1.3	74.8 ± 1.5	73.7 ± 0.8	-1.35%
SAGE+FOSR	32.3 ± 1.5	47.1 ± 0.8	73.6 ± 2.0	69.9 ± 0.8	+0.87%
GCN+BORF	24.7 ± 1.0	50.1 ± 0.9	75.8 ± 1.9	71.0 ± 0.8	+2.31%
GIN+BORF	31.4 ± 1.5	70.5 ± 1.3	78.2 ± 1.6	71.9 ± 1.3	+1.02%
SAGE+BORF	26.5 ± 1.1	47.4 ± 0.7	69.8 ± 2.0	73.5 ± 0.8	-3.62%
GCN+DR	-	47.0 ± 0.7	80.1 ± 1.8	72.2 ± 0.8	+4.68%
GIN+DR	-	64.8 ± 0.8	74.5 ± 2.8	**74.3 ± 0.8**	+0.17%
SAGE+DR	-	47.3 ± 0.7	72.7 ± 2.0	70.5 ± 0.8	+0.33%
GCN+GTR	27.4 ± 1.1	49.5 ± 1.0	78.9 ± 1.8	72.4 ± 1.2	+6.27%
GIN+GTR	28.4 ± 1.8	70.1 ± 1.2	78.5 ± 3.5	73.3 ± 0.9	-0.76%
SAGE+GTR	32.2 ± 1.1	47.4 ± 0.7	80.2 ± 1.6	74.2 ± 0.7	+4.76%
GCN+LASER	27.6 ± 1.3	50.3 ± 1.3	78.8 ± 1.6	71.8 ± 1.6	+6.62%
GIN+LASER	35.3 ± 1.3	68.6 ± 1.2	76.1 ± 2.4	72.1 ± 0.7	+2.61%
SAGE+LASER	31.9 ± 1.4	47.6 ± 0.7	75.5 ± 1.7	71.1 ± 0.8	+1.90%
GCN+GOKU	27.6 ± 1.2	49.8 ± 0.7	81.0 ± 2.0	71.9 ± 0.8	+7.21%
GIN+GOKU	33.8 ± 1.2	**71.3 ± 0.9**	78.4 ± 2.5	73.9 ± 1.0	+3.89%
SAGE+GOKU	32.9 ± 1.2	47.2 ± 0.8	78.2 ± 1.7	72.8 ± 0.9	+4.02%
$\text{FlatNSD}_{\text{GCN}}$	45.1 ± 1.2	65.7 ± 0.9	77.6 ± 1.8	72.2 ± 0.9	**+31.30%**
$\text{FlatNSD}_{\text{GIN}}$	**47.4 ± 1.3**	68.5 ± 0.9	**86.2 ± 1.6**	71.5 ± 0.8	+14.71%
$\text{FlatNSD}_{\text{SAGE}}$	43.8 ± 1.5	66.5 ± 0.9	80.7 ± 1.7	70.6 ± 0.9	+23.05%

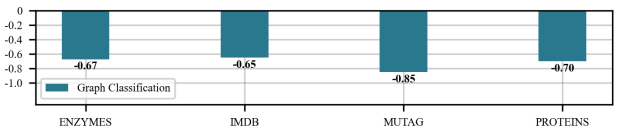


Figure 5: Spearman Correlation between Sheaf Effective Resistance and Train Accuracy on FlatNSD-SAGE.

Results. Figure 4 demonstrates that FlatNSD is highly robust: while not always the top performer on individual datasets, it presents a substantially higher relative average improvement. Crucially, FlatNSD robustness may be explained by its general efficacy outside of purely over-squashing-dominated regimes. We highlight this perspective, recalling the discussion on Section B.2 about the existence of sheaves that mitigate over-smoothing and over-squashing simultaneously.

7 Related work

Sheaf Neural Networks. Building on the contributions of Bodnar et al. [2022], who demonstrated the advantages of the NSD model in handling heterophilic graphs and mitigating over-smoothing, we investigated the over-squashing phenomenon within SNNs. Bamberger et al. [2025] introduced flat vector bundles together with BuNN, an architecture that departs from the message-passing paradigm to address over-squashing. Our results suggest that the message-passing paradigm itself, when equipped with flat bundles, is already a strong contender: FlatNSD performs competitively with

BuNN on the TreeNeighborsMatch task (Figure 1) and on node classification with larger graphs (Table 2). Notably, much of the variation across sheaf-based models seems to be explained by the choice of restriction-map parameterization rather than by the underlying architecture (Figure 1). Direct head-to-head comparisons remain limited as the BuNN code was not publicly available at the time of submission. Crucially, Ribeiro et al. [2026] introduced sheaves on directed graphs to allow a selective behavior of each node, creating the CSNN model, which also handles over-squashing. Other recently proposed sheaf-based models [Fiorini et al., 2026, Mule et al., 2026] also formulate directed sheaves but do not discuss data compression.

Graph Rewiring. Graph rewiring is a popular strategy to address over-squashing. Part of its success is theoretically explained by Di Giovanni et al. [2023] and empirically validated by different authors, using different heuristics Black et al. [2023], Topping et al. [2022], Karhadkar et al. [2023], Barbero et al. [2024], Liang et al. [2025]. Moreover, relational GNNs are tools to improve rewiring methods that we exclude from our experiments, similarly to Nguyen et al. [2023], Liang et al. [2025]. This choice intends to isolate the contribution of FlatNSD and to consider graph rewiring methods and SNNs without further performance-oriented modifications.

Over-squashing. There is a variety of strategies to overcome over-squashing and the related problem of long-range propagation. They include but do not restrict to: DE-GNN approaches [Gravina et al., 2025, 2023], exploiting port-Hamiltonian dynamics [Heilig et al., 2025], adapting messages in the information flow [Errica et al., 2025, Finkelshtein et al., 2024], and Graph Transformers [Shirzad et al., 2023, Rampášek et al., 2022, Kreuzer et al., 2021]. Our experiments establish SNNs as a principled member of this toolkit for mitigating over-squashing.

8 Conclusions

In this work, we established a theoretical link between Sheaf Neural Networks and over-squashing mitigation through the novel concept of Sheaf Effective Resistance. We demonstrated that FlatNSD can implicitly optimize restriction maps to facilitate information flow, all without altering the underlying graph topology. Our empirical analysis reveals that FlatNSD is a robust model, validating the efficacy of SNNs across diverse structural regimes.

Future works. Over-squashing is a well-known issue in GNNs, explored theoretically and empirically through various lenses. Many approaches rely on concepts deeply tied to effective resistance, such as the spectral gap, the Cheeger constant, and commute time. Notably, Topping et al. [2022] introduce an edge-based metric called Balanced Forman curvature, linking negatively curved edges to over-squashing and proving that a positive lower bound on curvature controls the spectral gap. We envision that defining a notion of curvature for graph sheaves could be intrinsically related to the sheaf spectral gap. This perspective further positions sheaves as a powerful tool to alleviate over-squashing: bottlenecks traditionally viewed as intrinsic to the graph structure might be better understood—and resolved—as properties of the associated sheaf.

Limitations. We recall that our goal with this work was to analyze how FlatNSD can deal with over-squashing. To this end, we keep the model as simple as possible. Our experiments and theoretical results contribute to support that sheaves, through the lens of sheaf effective resistance, provide a framework genuinely capable of mitigating over-squashing. Nevertheless, the FlatNSD model can be strengthened. For instance, Bamberger et al. [2025] considered a parallel dimension for multiple sheaves instead of a single sheaf per layer, which led to improved long-range interactions; Fiorini et al. [2026] generalizes the sheaf Laplacian even further with sheaves designed to explicitly account for directed graphs; Ribeiro et al. [2026] introduces a sheaf based model where each node can learn whether it propagates or receives messages from/to its neighbors in each layer.

Computational Complexity. Sheaf Neural Networks main drawback is its computational cost, but we address that the FlatNSD model we used in this work presents significant improvements in comparison to the original NSD model (Section G), showing positive perspectives in this realm. Nevertheless, in Section H we provide the runtime of FlatNSD and GCN with different rewiring methods, where we concluded that FlatNSD is considerable faster than graph rewiring methods for the tested node classification tasks while keeps the same order of magnitude for the tested graph classification tasks. This shows that FlatNSD is a versatile and balanced model.

Societal Impact. We foresee no immediate societal or ethical concerns arising from this work.

Acknowledgments

We acknowledge the support of the Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro (FAPERJ) (SEI-260003/020348/2025, SEI-260003/020694/2025) and the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) (404336/2023-0, 305692/2025-9, 445170/2024-7).

References

Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. In *International Conference on Learning Representations*, 2021.

Hugo Attali, Davide Buscaldi, and Nathalie Pernelle. Delaunay graph: Addressing over-squashing and over-smoothing using delaunay triangulation. In *Forty-first International Conference on Machine Learning*, 2024.

Jacob Bamberger, Federico Barbero, Xiaowen Dong, and Michael Bronstein. Bundle neural networks for message diffusion on graphs. In *The Thirteenth International Conference on Learning Representations*, 2025.

Pradeep Kr Banerjee, Kedar Karhadkar, Yu Guang Wang, Uri Alon, and Guido Montúfar. Oversquashing in gnns through the lens of information contraction and graph expansion. In *2022 58th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1–8. IEEE, 2022.

Federico Barbero, Ameya Velingker, Amin Saberi, Michael M Bronstein, and Francesco Di Giovanni. Locality-aware graph rewiring in gnns. In *The Twelfth International Conference on Learning Representations*, 2024.

Mitchell Black, Zhengchao Wan, Amir Nayyeri, and Yusu Wang. Understanding oversquashing in gnns through the lens of effective resistance. In *International Conference on Machine Learning*, pages 2528–2547. PMLR, 2023.

Cristian Bodnar, Francesco Di Giovanni, Benjamin Chamberlain, Pietro Lio, and Michael Bronstein. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns. *Advances in Neural Information Processing Systems*, 35:18527–18541, 2022.

Chen Cai and Yusu Wang. A note on over-smoothing for graph neural networks. *arXiv preprint arXiv:2006.13318*, 2020.

Francesco Di Giovanni, Lorenzo Giusti, Federico Barbero, Giulia Luise, Pietro Lio, and Michael M. Bronstein. On over-squashing in message passing neural networks: The impact of width, depth, and topology. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 7865–7885. PMLR, 23–29 Jul 2023. URL `https://proceedings.mlr.press/v202/di-giovanni23a.html`.

David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. *Advances in neural information processing systems (NeurIPS)*, 2015.

Federico Errica, Henrik Christiansen, Viktor Zaverkin, Takashi Maruyama, Mathias Niepert, and Francesco Alesiani. Adaptive message passing: A general framework to mitigate oversmoothing, oversquashing, and underreaching. In *International Conference on Machine Learning*, pages 15490–15515. PMLR, 2025.

Lukas Fesser and Melanie Weber. Mitigating over-smoothing and over-squashing using augmentations of forman-ricci curvature. In *Learning on Graphs Conference*, pages 19–1. PMLR, 2024.

Ben Finkelshtein, Xingyue Huang, Michael M. Bronstein, and Ismail Ilkan Ceylan. Cooperative graph neural networks. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International*

Conference on Machine Learning, volume 235 of *Proceedings of Machine Learning Research*, pages 13633–13659. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/finkelshtein24a.html.

Stefano Fiorini, Hakan Aktas, Iulia Duta, Stefano Coniglio, Pietro Morerio, Alessio Del Bue, and Pietro Liò. Sheaves reloaded: A directional awakening. In *The Fourteenth International Conference on Learning Representations*, 2026.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning (ICML)*, 2017.

Jhony H Giraldo, Konstantinos Skianis, Thierry Bouwmans, and Fragkiskos D Malliaros. On the trade-off between over-smoothing and over-squashing in deep graph neural networks. In *Proceedings of the 32nd ACM international conference on information and knowledge management*, pages 566–576, 2023.

Alessio Gravina, Davide Bacciu, and Claudio Gallicchio. Anti-symmetric dgn: a stable architecture for deep graph networks. In *The Eleventh International Conference on Learning Representations*, 2023.

Alessio Gravina, Moshe Eliasof, Claudio Gallicchio, Davide Bacciu, and Carola-Bibiane Schönlieb. On oversquashing in graph neural networks through the lens of dynamical systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 2025.

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

Jakob Hansen and Thomas Gebhart. Sheaf neural networks. *NeurIPS Workshop on Topological Data Analysis and Beyond*, 2020.

Jakob Hansen and Robert Ghrist. Toward a spectral theory of cellular sheaves. *Journal of Applied and Computational Topology*, 3(4):315–358, 2019.

Simon Heilig, Alessio Gravina, Alessandro Trenta, Claudio Gallicchio, Davide Bacciu, et al. Port-hamiltonian architectural bias for long-range propagation in deep graph networks. In *13th International Conference on Learning Representations*. International Conference on Learning Representations (ICLR), 2025.

Adarsh Jamadandi, Celia Rubio-Madrigal, and Rebekka Burkholz. Spectral graph pruning against over-squashing and over-smoothing. *Advances in Neural Information Processing Systems*, 37: 10348–10379, 2024.

Kedar Karhadkar, Pradeep Kr Banerjee, and Guido Montufar. Fosr: First-order spectral rewiring for addressing oversquashing in gnns. In *The Eleventh International Conference on Learning Representations*, 2023.

T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

Gustav Kirchhoff. Ueber die auflösung der gleichungen, auf welche man bei der untersuchung der linearen vertheilung galvanischer ströme geführt wird. *Annalen der Physik*, 148(12):497–508, 1847.

Devin Kreuzer, Dominique Beaini, Will Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph transformers with spectral attention. *Advances in neural information processing systems*, 34:21618–21629, 2021.

Langzhang Liang, Fanchen Bu, Zixing Song, Zenglin Xu, Shirui Pan, and Kijung Shin. Mitigating over-squashing in graph neural networks by spectrum-preserving sparsification. *Forty-second International Conference on Machine Learning*, 2025.

Zakaria Mhammedi, Andrew Hellicar, Ashfaqur Rahman, and James Bailey. Efficient orthogonal parametrisation of recurrent neural networks using householder reflections. In *International Conference on Machine Learning*, pages 2401–2409. PMLR, 2017.

Luca Miglior, Matteo Tolloso, Alessio Gravina, and Davide Bacciu. Can you hear me now? a benchmark for long-range graph propagation. *arXiv preprint arXiv:2512.17762*, 2025.

Yaaqov Mishayev, Yonatan Sverdlov, Tal Amir, and Nadav Dym. Short-range oversquashing. *arXiv preprint arXiv:2511.20406*, 2025.

Christopher Morris, Nils M Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs. *arXiv preprint arXiv:2007.08663*, 2020.

Emanuele Mule, Stefano Fiorini, Antonio Purificato, Federico Siciliano, Stefano Coniglio, and Fabrizio Silvestri. Directional sheaf hypergraph networks: Unifying learning on directed and undirected hypergraphs. In *The Fourteenth International Conference on Learning Representations*, 2026.

Khang Nguyen, Nong Minh Hieu, Vinh Duc Nguyen, Nhat Ho, Stanley Osher, and Tan Minh Nguyen. Revisiting over-smoothing and over-squashing using ollivier-ricci curvature. In *International Conference on Machine Learning*, pages 25956–25979. PMLR, 2023.

Anton Obukhov. Efficient householder transformation in pytorch, 2021. URL `www.github.com/toshas/torch-householder`. Version: 1.0.1, DOI: 10.5281/zenodo.5068733.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. *arXiv preprint arXiv:2002.05287*, 2020.

Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of gnns under heterophily: Are we really making progress? *arXiv preprint arXiv:2302.11640*, 2023.

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

André Ribeiro, Ana Luiza Tenório, Juan Belieni, Amauri H Souza, and Diego Mesquita. Cooperative sheaf neural networks. In *The Fourteenth International Conference on Learning Representations*, 2026.

Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding. *Journal of Complex Networks*, 9(2):cnab014, 2021.

Danial Saber and Amirali Salehi-Abari. Empirical study of over-squashing in gnns and causal estimation of rewiring strategies. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 2525–2534, 2025.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *International conference on machine learning (ICML)*, 2020.

Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjin Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification. *arXiv preprint arXiv:2009.03509*, 2020.

Hamed Shirzad, Ameya Velingker, Balaji Venkatachalam, Danica J Sutherland, and Ali Kemal Sinop. Exphormer: Sparse transformers for graphs. In *International Conference on Machine Learning*, pages 31613–31632. PMLR, 2023.

Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. In *International Conference on Learning Representations*, 2022.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Zhilin Yang, William Cohen, and Ruslan Salakhudinov. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, pages 40–48. PMLR, 2016.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *ACM SIGKDD international conference on knowledge discovery & data mining (KDD)*, 2018.

Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. *Advances in neural information processing systems (NeurIPS)*, 2020.

A Additional Examples

To give a better intuition of how the choice of sheaves affects effective resistance, we provide two larger examples, building on the graph effective resistance examples from Black et al. [2023].

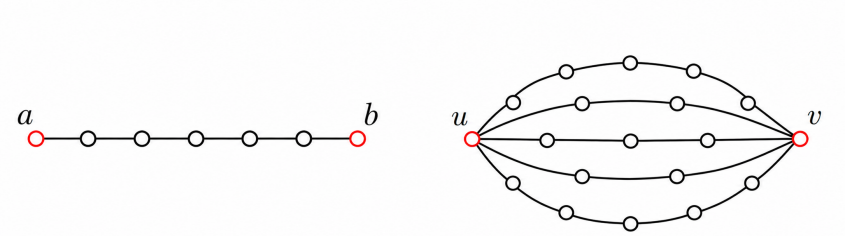


Figure 6: The two examples from Black et al. [2023]. For these graphs, we have $R(a,b) = 6$ (left) and $R(a,b) = \frac{10}{9}$ (right)

Consider first the two graphs above: nodes a, b connected by a path of length 6, and nodes a, b connected by four disjoint paths of lengths $5, 4, 4, 5$. Their graph effective resistances are $R(a,b) = 6$ and $R(a,b) = \frac{10}{9}$, respectively. From their Laplacian pseudo-inverses, we obtain $(L_G^+)_{ab} = -\frac{8}{7}$ and $(L_G^+)_{ab} = -\frac{185}{1152}$.

Let $Q(k) = \begin{pmatrix} \cos k & -\sin k \\ \sin k & \cos k \end{pmatrix}$, and define $\mathcal{F}_a = Q(x)$, $\mathcal{F}_b = Q(y)$, with identity maps on all remaining nodes. Using Proposition 4.2, we obtain

$$R_{\mathcal{F}}(a,b) = \frac{52 + 32\cos(y-x)}{7} \quad \text{and} \quad R_{\mathcal{F}}(a,b) = \frac{455 + 185\cos(y-x)}{288},$$

for the first and second graphs, respectively. Hence, for $y - x \in \{0, \frac{\pi}{2}, \pi\}$, we have

$$R_{\mathcal{F}}(a,b) \in \left\{12, \frac{52}{7}, \frac{20}{7}\right\} \quad \text{and} \quad R_{\mathcal{F}}(a,b) \in \left\{\frac{20}{9}, \frac{455}{288}, \frac{15}{16}\right\},$$

for first and second graphs, respectively. Since $(L_G^+)_{ab} < 0$ in both cases, rotations can only lower the endpoint resistance relative to the aligned configuration.

Importantly, however, the effect of the sheaf need not be uniform throughout the graph. Let w be the first internal node adjacent to a in the path graph. In this case, $(L_G^+)_{aw} = 1$ and $R(a,w) = 1$, yielding $R_{\mathcal{F}}(a,w) = 6 - 4\cos x$. Thus, every non-zero rotation at a increases the resistance between a and its neighbor. For example, taking $\mathcal{F}_a = -I$ and all other maps equal to the identity gives $R_{\mathcal{F}}(a,b) = \frac{20}{7}$ and $R_{\mathcal{F}}(a,w) = 10$, in contrast to $R_{\mathcal{F}}(a,b) = 12$ and $R_{\mathcal{F}}(a,w) = 2$ for the trivial sheaf. Therefore, a nontrivial sheaf can simultaneously lower the resistance between a distant pair while increasing it between nearby nodes. This is consistent with the role of the restriction maps: information can be transmitted efficiently from a to b even when it cannot be directly decoded by intermediate nodes along the communication path.

B Additional Experiments

B.1 Rewiring against FlatNSD on node classification

We additionally test the performance of FlatNSD against rewiring methods on the node classification benchmarks used in Nguyen et al. [2023], Liang et al. [2025]. It achieves the best performance in 4 out

of 6 datasets, and a generally higher overall improvement of the backbone GNN compared to baselines. We notice that Cora and Citeseer are homophilic graphs, an environment where it is already known that sheaf-based models do not flourish Bodnar et al. [2022]. In contrast, graph rewiring techniques often struggle to improve their backbone GNNs on node classification tasks. This relates to the fact that graph rewiring methods primarily target over-squashing, whereas FlatNSD is more versatile.

Table 3: Performance of FlatNSD and GCN, GIN, and SAGE with different rewiring methods for both node and graph classification tasks. The method with the higher accuracy is highlighted in **bold**. DR requires graphs to have at least 3 nodes, but some graphs from Enzymes have only 2.

Method	Cora	Citeseer	Texas	Cornell	Wisconsin	Chameleon	Rel Avg Improv
GCN+NONE	86.7 ± 0.3	72.3 ± 0.3	44.2 ± 1.5	41.5 ± 1.8	44.6 ± 1.4	59.2 ± 0.6	-
GIN+NONE	77.5 ± 1.2	59.3 ± 1.3	46.4 ± 4.5	40.2 ± 4.8	42.1 ± 3.6	56.6 ± 0.8	-
SAGE+NONE	87.1 ± 0.1	75.0 ± 0.1	82.4 ± 0.3	73.2 ± 0.3	77.3 ± 0.9	66.1 ± 0.9	-
GCN+SDRF	86.3 ± 0.3	72.6 ± 0.3	43.9 ± 1.6	42.2 ± 1.5	46.2 ± 1.2	59.4 ± 0.5	+0.81%
GIN+SDRF	77.2 ± 0.8	59.9 ± 1.2	44.5 ± 3.9	38.2 ± 4.1	43.1 ± 1.6	58.1 ± 1.4	-0.57%
SAGE+SDRF	87.1 ± 0.7	74.1 ± 0.7	81.9 ± 2.9	72.4 ± 2.5	76.5 ± 1.8	65.2 ± 1.2	-0.88%
GCN+FOSR	85.9 ± 0.3	72.3 ± 0.3	46.0 ± 1.6	40.2 ± 1.6	48.3 ± 1.3	59.3 ± 0.6	+1.41%
GIN+FOSR	78.2 ± 0.8	61.4 ± 1.5	43.7 ± 4.1	39.4 ± 3.8	43.7 ± 2.8	59.3 ± 0.6	+0.87%
SAGE+FOSR	87.5 ± 0.6	74.7 ± 0.8	81.1 ± 2.0	71.1 ± 2.2	75.4 ± 1.1	66.2 ± 0.8	-1.12%
GCN+BORF	**87.5 ± 0.2**	73.8 ± 0.2	49.4 ± 1.2	50.8 ± 1.1	50.3 ± 0.9	61.5 ± 0.4	+8.97%
GIN+BORF	77.6 ± 0.9	60.8 ± 0.2	49.9 ± 3.4	39.9 ± 3.9	46.7 ± 2.2	59.2 ± 0.4	+4.16%
SAGE+BORF	87.3 ± 0.6	73.5 ± 0.6	73.5 ± 2.3	73.0 ± 3.1	78.2 ± 3.7	64.9 ± 0.7	-2.25%
GCN+DR	78.4 ± 1.2	69.5 ± 1.6	67.8 ± 2.5	57.8 ± 1.9	62.8 ± 2.1	58.6 ± 0.8	+19.84%
GIN+DR	64.6 ± 1.6	50.8 ± 2.0	57.3 ± 2.8	50.1 ± 2.7	50.1 ± 3.0	60.2 ± 0.8	+7.08%
SAGE+DR	87.5 ± 0.4	74.2 ± 0.7	81.1 ± 3.7	76.2 ± 2.4	77.1 ± 2.9	**66.8 ± 1.1**	+0.45%
GCN+GTR	87.3 ± 0.4	72.4 ± 0.3	45.9 ± 1.9	50.8 ± 1.6	46.7 ± 1.5	57.6 ± 0.8	+4.85%
GIN+GTR	78.6 ± 1.3	62.6 ± 1.9	49.5 ± 2.9	39.4 ± 2.3	44.2 ± 2.4	57.1 ± 1.2	+2.92%
SAGE+GTR	87.5 ± 0.6	74.3 ± 0.7	76.8 ± 4.3	71.1 ± 3.0	78.0 ± 2.5	65.6 ± 1.1	-1.66%
GCN+LASER	86.9 ± 1.1	72.6 ± 0.6	45.9 ± 2.6	42.7 ± 2.6	46.0 ± 2.6	43.5 ± 1.0	-2.67%
GIN+LASER	79.1 ± 1.0	66.5 ± 1.3	46.5 ± 4.5	44.5 ± 3.8	46.1 ± 4.6	59.8 ± 2.2	+6.71%
SAGE+LASER	84.9 ± 0.5	74.5 ± 0.4	81.3 ± 3.1	76.5 ± 3.1	79.8 ± 2.5	54.7 ± 0.9	-2.34%
GCN+GOKU	86.8 ± 0.3	73.6 ± 0.2	72.4 ± 2.2	69.4 ± 2.1	68.8 ± 1.4	63.2 ± 0.4	+32.33%
GIN+GOKU	78.4 ± 0.5	63.6 ± 1.3	60.2 ± 2.6	49.5 ± 3.5	57.6 ± 3.1	62.1 ± 0.6	+17.97%
SAGE+GOKU	87.1 ± 0.5	74.1 ± 0.3	81.4 ± 2.1	71.9 ± 4.1	80.2 ± 2.1	66.1 ± 0.6	-0.07%
$\text{FlatNSD}_{\text{GCN}}$	76.7 ± 0.4	68.5 ± 0.3	86.3 ± 1.1	79.4 ± 1.3	**86.2 ± 0.8**	51.6 ± 0.5	+41.70%
$\text{FlatNSD}_{\text{GIN}}$	76.6 ± 0.3	70.0 ± 0.3	84.4 ± 1.3	78.1 ± 1.3	**86.2 ± 1.0**	52.5 ± 0.4	+48.43%
$\text{FlatNSD}_{\text{SAGE}}$	83.4 ± 0.3	**76.0 ± 0.3**	**86.9 ± 1.0**	**80.5 ± 1.2**	85.4 ± 0.9	52.5 ± 0.4	+0.40%

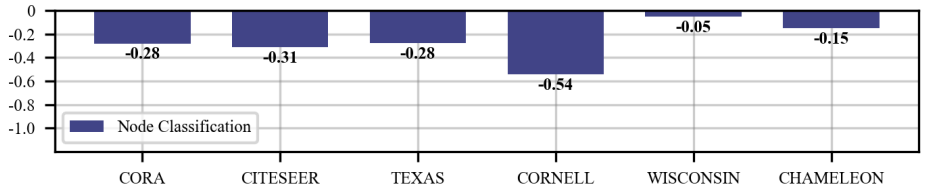


Figure 7: The figure shows the Spearman Correlation between Sheaf Effective Resistance and Train Accuracy on FlatNSD-SAGE. The correlation is weaker on these node classification tasks, aligning with the common understanding they are not a suitable setting for studying over-squashing.

Results. Table 3 shows that FlatNSD is a robust model across both node and graph classification tasks (Figure 4). The exceptions are Cora, Citeseer, and Chameleon. Cora and Citeseer are homophilic graphs, a regime where sheaf-based models are known to underperform their classical counterparts [Bodnar et al., 2022]; the drop relative to vanilla GCN on these datasets is therefore expected rather than surprising. Chameleon is known to contain duplicate nodes Platonov et al. [2023], which interfere with model performance. Overall, on the considered heterophilic datasets, FlatNSD is competitive with or superior to graph rewiring.

Effective Resistance. The Spearman correlation shows a weak correlation for these node classification. This aligns with the view that the graph classification tasks performed in Figure 4 is more suitable for studying over-squashing: it requires aggregating node information into a graph-level embedding, thereby amplifying information compression.

Accordingly, Saber and Salehi-Abari [2025] suggest that rewiring strategies primarily pay off when over-squashing is severe and global, but the standard node classification benchmarks used here likely suffer only from mild compression.Therefore, although this might attenuate the effectiveness of FlatNSD-SAGE regarding over-squashing mitigation on this set of problems, the model still achieves good performance in terms of raw accuracy, demonstrating its general efficacy outside of purely over-squashing-dominated regimes. We highlight this perspective, recalling the discussion on Section B.2 about the existence of sheaves that mitigate over-smoothing and over-squashing simultaneously.

B.2 Over-squashing and Over-smoothing

Most graph rewiring techniques address over-squashing by reducing effective resistance or, equivalently, enlarging the spectral gap and the Cheeger constant Black et al. [2023]. However, modifying the edge set of a graph also typically causes standard GNNs to yield increasingly similar node representations as the number of layers grows. In other words, mitigating over-squashing is often accompanied by a worsening of the over-smoothing effect.

To overcome this, R-GNNs are used instead of vanilla GNNs Karhadkar et al. [2023], Jamadandi et al. [2024]. Rougly speaking, R-GNNs models can keep information from the original graph and use it through learned weights to keep balance between over-squashing and over-smoothing.

On the other hand, it is well known that NSD can circumvent over-smoothing. The question is: *are sheaves able to simultaneously mitigate over-squashing and over-smoothing?*

For simplicity, we consider a connected graph $G = (V, E)$ with n vertices and two classes $A, B \subseteq V$. Let α_e for all $e \in E$ be arbitrary non-negative scalars. According to Bodnar et al. [2022, Proposition 10], a sheaf whose stalks have dimension one equipped with invertible restriction defined by

$$\mathcal{F}_{v \trianglelefteq e} = -\alpha_e \text{ if } v \in A \text{ and } \mathcal{F}_{u \trianglelefteq e} = \alpha_e \text{ if } u \in B \tag{4}$$

induces a diffusion that can linearly separate the classes of G for almost all initial conditions. Therefore, even if we increase the number of layers, in such circumstances, the feature nodes will not converge to the same value.

We show that equipping a graph with the sheaves above can also reduce the total effective resistance, consequently reducing network sensitivity to over-squashing by Theorem 4.1.

Theorem B.1. *Let G be a connected graph with n vertices and two classes $A, B \subseteq V$. If $\mathcal{F}$ is a flat vector bundle as in Equation* (4)*, then* $\mathrm{R}_{\mathcal{F}} \leqslant \mathrm{R}$.

Proof. Since $d = 1$, we only have to prove that $\langle M, L_G^+ \rangle_F \geqslant 0$.

The flat and the orthogonal conditions force that $\alpha_e = 1$ for every $e \in E$. So $\mathcal{F}_v = -1$ if $v \in A$, $\mathcal{F}_u = 1$, if $u \in B$, and then $M_{uv} = \pm 1$. Let $\beta = (\beta_1, \ldots, \beta_n)$, where $\beta_i = \mathcal{F}_i$, for all $i \in V$. Then $M = \beta\beta^\top$. Consequently,

$$\langle M, L_G^+ \rangle = \mathrm{Tr}(M^\top L_G^+) = \mathrm{Tr}(\beta\beta^\top L_G^+) = \mathrm{Tr}(\beta^\top L_G^+ \beta) = \beta^\top L_G^+ \beta,$$

where we use the trace property that $\mathrm{Tr}(AB) = \mathrm{Tr}(BA)$, with $A = \beta$ and $B = \beta^\top L_G^+$.

Next, we consider the spectral decomposition $L_G = Q\Lambda Q^\top$ where $\Lambda = \mathrm{diag}(0, \lambda_2, \ldots, \lambda_{\mathrm{n}})$ with $0 = \lambda_1 \leqslant \lambda_2 \leqslant \cdots \leqslant \lambda_n$ eigenvalues of L_G. Then $\Lambda^+ = \mathrm{diag}(0, \frac{1}{\lambda_2}, \ldots, \frac{1}{\lambda_{\mathrm{n}}})$ and $L_G^+ = Q\Lambda^+ Q^\top$.

Therefore,

$$\langle M, L_G^+ \rangle = \beta^\top L_G^+ \beta = \sum_{k=2}^{n} \frac{(q_k^\top \beta)^2}{\lambda_k} \geqslant 0.$$

□

Therefore, if the number of classes of G is $C = 2$, there is a sheaf that handles both over-smoothing and over-squashing. For graphs with $C > 2$ classes, Bodnar et al. [2022] argued that sheaves whose stalks have dimension $d \geqslant C$ can mitigate over-smoothing. Our experiments in Figure 4 suggest that a higher dimensional version of the above theorem holds, since FlatNSD presents good performance

on real-world datasets for node and graph classification tasks, where both over-smoothing and over-squashing may co-exist.

Conclusion. Sheaves can alleviate over-smoothing leveraging the existence of a non-trivial equilibrium space of the diffusion process [Bodnar et al., 2022]. Within the same class of sheaves, we can also mitigate over-squashing by lowering the total sheaf effective resistance.

To substantiate the theoretical insight, we include experiments demonstrating that FlatNSD can handle over-smoothing and over-squashing simultaneously.

As explained by Alon and Yahav [2021], there is no reason to expect over-smoothing to occur in the TreeNeighborsMatch problem. To the best of our knowledge, there is no synthetic benchmark specifically designed to evaluate the ability of GNNs to cope with both over-squashing and over-smoothing. We therefore propose GridTransfer as a task to study this setting.

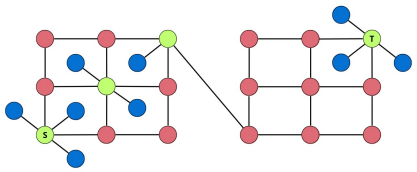


Figure 8: Example of graph in the GridTransfer task. Two grids are linked by an edge connecting two nodes in a diagonal. The task is to transfer information from one of the nodes in the diagonal of the node in the path to the most-distant node in the other grid.

GridTransfer. We construct a graph by taking two $n \times n$ grids and adding a single edge that connects one node on the outermost diagonal of the first grid to one node on the outermost diagonal of the second grid. The task is to transfer information from a *target node* in the first grid to a node in the second grid located at the opposite corner relative to the endpoint of the inter-grid connection.

The nodes along the main diagonal of each grid act as *green nodes*, as in the NEIGHBORSMATCH task, and each green node is connected to a number of *blue neighbors*. The label is determined by matching the number of blue neighbors: the green node whose number of blue neighbors equals that of the target node contains its true label.

We adopt the same model configurations as in TREENEIGHBORSMATCH and choose the network depth as $4(n-1)+1$, which corresponds to the worst-case receptive-field requirement for transferring information across the two-grid construction. Since this depth increases linearly with n, larger grids induce much deeper networks, where over-smoothing is expected to be more pronounced. Results are shown in Figure 9. In comparison with TREENEIGHBORSMATCH, we observe that FlatNSD-SAGE remains robust, indicating that over-smoothing has limited impact on its performance in this setting.

Dirichlet Energy. Cai and Wang [2020] used Dirichlet energy, given by the quadratic form $E_G(\mathbf{X}) := \mathrm{tr}(\mathbf{X}^\top \Delta_G \mathbf{X})$, where Δ_G is the normalized Laplacian of the graph G, as a measure of "smoothness" of node embeddings.

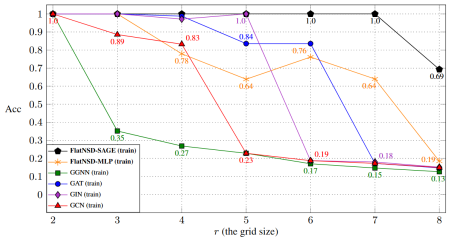


Figure 9: Accuracy of the models on the training set for the GridTransfer task. FlatNSD-SAGE is the only model to achieve accuracy 1 until $r = 7$.

From the perspective of Laplacian smoothing, deeper GCN propagation increasingly favors low-frequency signals, which generally lowers the Dirichlet energy and may cause node embeddings to become nearly constant on each connected component. Hence, mitigating over-smoothing should manifest as maintaining higher Dirichlet energy across layers, or equivalently, a less pronounced decrease.

In Figure 10, we observe that FlatNSD with restriction maps learned with a GCN increases the energy for Enzymes, IMDB, MUTAG, and Proteins. We find consistent empirical evidence that the learned restriction maps are associated with decreasing sheaf effective resistance during training in regimes sensitive to over-squashing. While this does not establish causality, it supports the theoretical link between resistance and information propagation and provides a diagnostic for when FlatNSD alleviates over-squashing – whilst still alleviating over-smoothing through the increase of the Dirichlet energy.

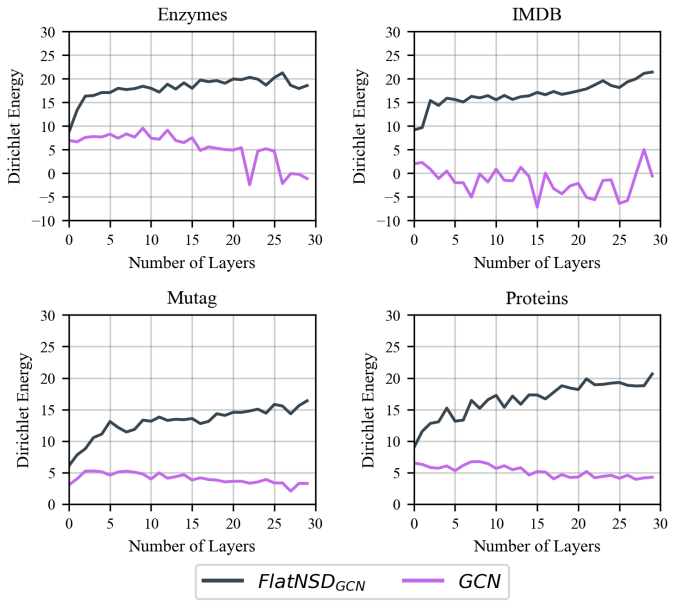


Figure 10: Log-Dirichlet Energy on the graph classification tasks of GCN and FlatNSD$_{\text{GCN}}$. The high Dirichlet Energy presented by FlatNSD-GCN against the low Dirichlet Energy presented by GCN is sign that FlatNSD alleviates over-smoothing even in datasets more strongly dominated by over-squashing.

B.3 Evaluating FlatNSD on Graph Transfer

The Graph Transfer task proposed by Di Giovanni et al. [2023] as a benchmark for over-squashing, evaluates a model's ability to propagate information from a target node to a source node. The task is defined on three graph topologies: RING, CROSSEDRING, and CLIQUEPATH, where r denotes the distance between the source and the target node.

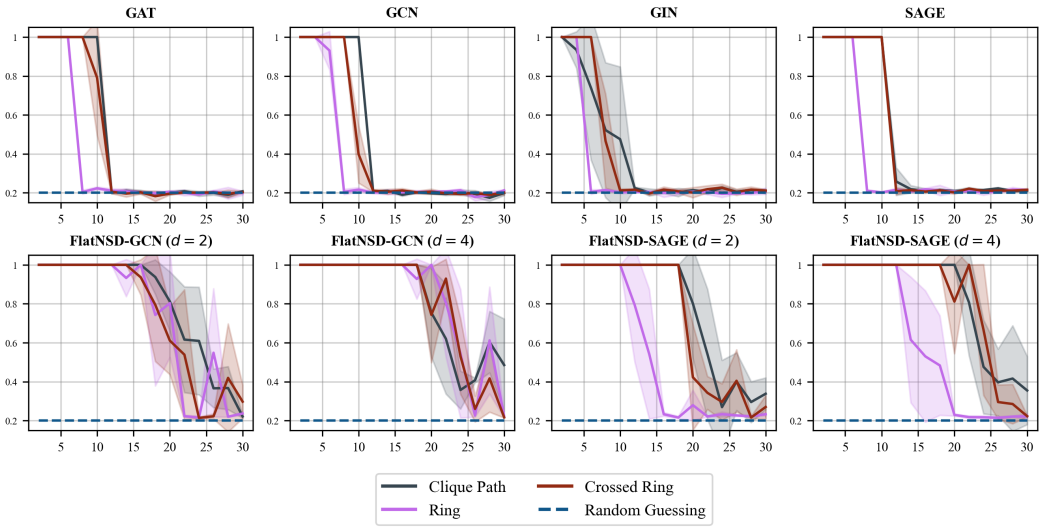


Figure 11: FlatNSD accuracy decreases only after longer distances from source to target, in comparison with GAT, GCN, GIN, and SAGE. Moreover, $d = 4$ presents better performance than $d = 2$, suggesting the importance of d for tasks sensitive to over-squashing.

RING graphs are simple cycles of size n in which the source and target nodes lie at distance $\lfloor n/2 \rfloor$. CROSSEDRING graphs are cycles of the same size augmented with chords/crosses between auxiliary nodes; the added edges do not shorten the source–target distance, which remains $\lfloor n/2 \rfloor$. CLIQUEPATH

graphs consist of a $\lfloor n/2 \rfloor$-clique connected to a path of length $\lfloor n/2 \rfloor$, with the source placed inside the clique and the target at the far end of the path; the source–target distance is $\lfloor n/2 \rfloor + 1$. The three topologies are designed to isolate the effect of structural connectivity at a fixed distance: although r is matched across topologies, over-squashing is most severe for CLIQUEPATH and least severe for CROSSEDRING.

Results. For all three topologies, FlatNSD can postpone the accuracy drop, i.e., in FlatNSD accuracy decreases only after bigger distances from source to target, in comparison with all non-sheaf based models (GAT, GCN, GIN, SAGE), providing further evidence that FlatNSD mitigates over-squashing. Moreover, $d = 4$ presents better performance than $d = 2$, suggesting the importance of stalk dimension d for tasks sensitive to over-squashing, in alignment with our results for the TreeNeighborsMatch task.

B.4 Spearman Correlation on the TreeNeighborsMatch task

Recall that Spearman correlation assesses the monotonic relationship between two variables and yields a value in $[-1, 1]$.

Accuracy. We compute Spearman rank correlation per layer between training accuracy in the TreeNeighborsMatch and the sheaf effective resistance across all radii r with $d = 4$. While the model does not consistently recover the resistance-minimizing sheaf at each layer, it reliably does so in the initial layers. The effective resistance reported in Figure 2 and Figure 12 is the mean of total

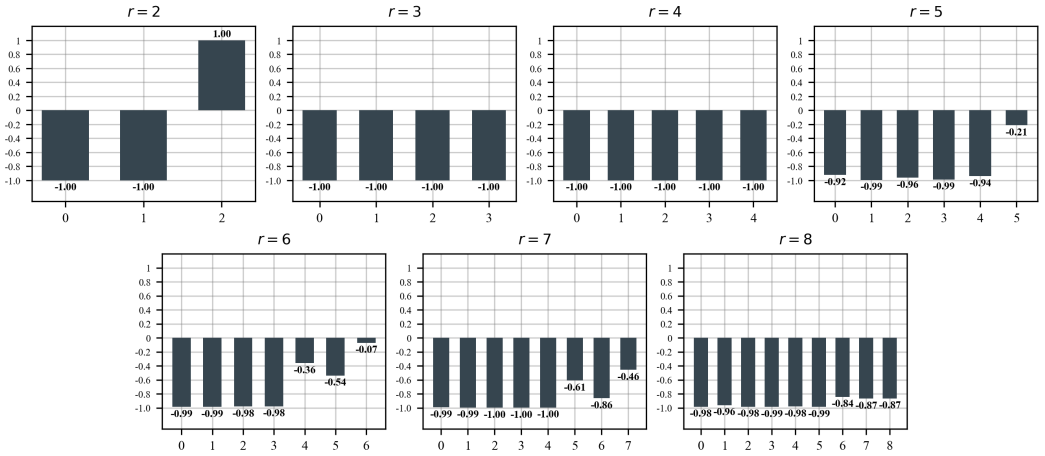


Figure 12: Spearman Correlation between total sheaf effective resistance and accuracy in TreeNeighborsMatch on each layer across the problem radius $2 \leqslant r \leqslant 8$ (left to right).

sheaf effective resistance throughout the batches in each epoch, for each layer. The correlation is then calculated over the pairs $(mean(R_{\mathcal{F}}^{batches}), train\ acc)$.

Jacobian norm. Still in the TreeNeighborsMatch, we compute Spearman rank correlation between the Jacobian norm and the sheaf effective resistance. As predicted by the theory, we obtained a strong negative correlation between them. Moreover, the plots showing correlation with the Jacobian (Figure 13) and the plots showing correlation with accuracy (Figure 12) exhibit the same behavior. This result further supports the hypothesis that a reduction in sheaf effective resistance enhances the Jacobian sensitivity, thereby assisting in the mitigation of over-squashing.

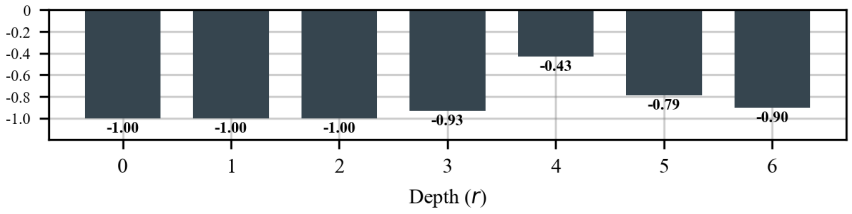


Figure 13: Spearman Correlation between total sheaf effective resistance and Jacobian norm for the TreeNeighborsMatch task on each layer across the problem radius $2 \leqslant r \leqslant 8$ (left to right).

B.5 Short-range Over-squashing

Mishayev et al. [2025] proposes a short-range over-squashing scenario, instantiated through the two-radius problem. In their experiments, SetTransformer achieves 100% accuracy across all tested graph sizes, while message-passing GNNs struggle to fit the data, with GAT being the best-performing architecture among GCN, GIN, and SAGE. We evaluate FlatNSD on the same task, fixing $d = 2$ and the remaining parameters as in Mishayev et al. [2025]. Notably, FlatNSD with a SetTransformer backbone and hidden dimension 32 achieves 100% accuracy across all graph sizes, with fast convergence (under 35 epochs).

To better isolate the contribution of the sheaf, we also evaluate FlatNSD with a SAGE backbone. Its performance starts to decay on the two-radius problem with 50 nodes, but remains above that of SAGE alone. This highlights the importance of choosing an appropriate parametrization for the backbone GNN, revealing a similar dichotomy to that observed between the parametrizations used in the TreeNeighborsMatch experiment.

B.6 Intervention on the TreeNeighborsMatch problem

To examine the distinction between correlation and mechanism more directly, we perform a controlled intervention on TreeNeighborsMatch with fixed stalk dimension. We compare a trivial flat bundle, with $\mathcal{F}_v = I_d$ at every node, against an oracle endpoint intervention in which all maps remain fixed to the identity except those at the root and the task-relevant leaf.

The trivial bundle attains the maximum possible total resistance. Under this configuration, the model fails to fit the training set, remaining near 0.045 training accuracy. By contrast, allowing nontrivial maps only at the two task-relevant endpoints substantially lowers their pairwise sheaf effective resistance, and the model converges almost immediately, after only two iterations. Since the graph topology and all remaining restriction maps are unchanged, this controlled intervention shows that modifying the flat bundle can remove a communication obstruction that persists under the maximum-resistance trivial bundle. This provides more direct evidence for the role of sheaf effective resistance in the mechanism underlying information propagation, beyond the correlation observed during learning.

C Contributions and the role of FlatNSD

The main contributions of this work are theoretical rather than architectural. In particular, we derive a closed-form expression for pairwise sheaf effective resistance for flat bundles (Proposition 4.2), a closed-form expression for total sheaf effective resistance, together with an efficient way of computing it (Theorem 4.2), and establish a connection between sheaf effective resistance and bounds on the message-passing Jacobian (Theorem 4.1). We further extend Theorem 2.1, Theorem 4.1, and Corollary 4.1 to bipartite graphs by extending the corresponding results for graph effective resistance (Lemma 4.1, Proposition 4.4, and Theorem 4.3). We also provide a result characterizing the existence of flat bundles that strictly decrease effective resistance (Proposition 4.3) and empirically study how sheaf effective resistance evolves during learning. FlatNSD is introduced as a simplified model used to isolate and verify these mechanisms empirically, rather than as the main contribution of the paper.

Our analysis builds upon several existing results. Sheaf effective resistance was introduced by Hansen and Ghrist [2019]; our contribution is to recognize and develop its use as a tool for understanding information propagation in sheaf-based GNNs. We further build upon the flat sheaf Laplacian decomposition of Bamberger et al. [2025], the connection between graph effective resistance and message-passing Jacobian bounds developed by Black et al. [2023], and the Neural Sheaf Diffusion (NSD) architecture of Bodnar et al. [2022]. The results developed here combine these ingredients to connect the geometry of learned sheaves, effective resistance, and over-squashing.

To further clarify the role of FlatNSD, we briefly compare it with the sheaf-based architectures considered throughout this work. FlatNSD intentionally removes architectural components that are not necessary for studying the resistance mechanism, while retaining the basic sheaf diffusion structure:

- **FlatNSD:** learns one orthogonal map $\mathcal{F}_v$ per vertex v and uses the same diffusion process as NSD. The resulting sheaf is a flat $O(d)$-bundle, which allows the decomposition $L_{\mathcal{F}} = \mathbf{O}^\top (L_G \otimes I_d)\mathbf{O}$ and makes the theoretical analysis particularly transparent.
- **Orthogonal NSD:** learns one orthogonal restriction map $\mathcal{F}_{v \trianglelefteq e}$ for each vertex-edge incidence pair $v \trianglelefteq e$. Its diffusion operator is the sheaf Laplacian obtained from the discretized heat equation on the sheaf. FlatNSD can be viewed as a constrained version of this construction in which the incidence maps are induced by a single map per node.
- **BuNN:** also learns one orthogonal map $\mathcal{F}_v$ per vertex, but uses the corresponding heat kernel as its diffusion operator. It therefore shares the same flat-bundle parameterization while differing in how diffusion is performed.
- **CSNN:** learns two conformal maps $\mathbf{S}_v$ and $\mathbf{T}_v$ for each node and constructs the sheaf on a directed graph. Its diffusion operator is formed from a product of in- and out-Laplacians, making it more expressive and structurally different from the flat orthogonal setting considered in our analysis.

These distinctions are important for interpreting the experimental results. FlatNSD is not proposed as a replacement for more expressive sheaf-based architectures. Instead, its restricted parameterization provides a controlled setting in which the theoretical predictions can be tested with as few additional mechanisms as possible. In this sense, FlatNSD serves as a basic sheaf model for validating the relationship between learned sheaf geometry, effective resistance, and over-squashing, while architectures such as NSD, BuNN, and CSNN incorporate additional modeling choices that may affect performance beyond this mechanism.

D More about sheaf effective resistance

Hansen and Ghrist [2019] introduced *cosheaf effective resistance* on a cell complex[1]. Here, we adapt their definition to a graph G with n vertices and $\mathcal{F}$ a sheaf over G whose stalks have dimension d.

Definition D.1. The sheaf effective resistance between nodes u and v is given by the solution to the optimization problem

$$\min_{c \in C^1(G,\mathcal{F})} \|c\|^2 \quad \text{s.t} \quad \delta^\top c = \mathbb{1}_u - \mathbb{1}_v, \tag{5}$$

where $\mathbb{1}_v$ is a binary vector of length nd where only the d entries corresponding to node v is 1.

If the sheaf is trivial, then the effective resistance is the solution to $\min\{\|c\|^2 \mid Bc = 1_u - 1_v\}$, where B is the incidence matrix of G. This is equivalent to $\mathrm{R}(u,v) = (1_u - 1_v)^\top L_G^+ (1_u - 1_v)$.

We check the equivalence between Definition D.1 and Definition 4.1 in the following proposition:

Proposition D.1. *Definition D.1 and Definition 4.1 are equivalent.*

Proof. Recall that for any matrix $A^\top A$, it holds that $x^\top (A^\top A)^+ x = \min\{\|y\|^2 \mid A^\top y = x\}$. Moreover, $L_{\mathcal{F}} = \delta^\top \delta$. Consequently:

$$\begin{aligned} \mathrm{R}_{\mathcal{F}}(u,v) &= (\mathbb{1}_u - \mathbb{1}_v)^\top L_{\mathcal{F}}^+ (\mathbb{1}_u - \mathbb{1}_v) \\ &= \min\{\|c\|^2 \mid \delta^\top c = (\mathbb{1}_u - \mathbb{1}_v)\}. \end{aligned}$$

□

In this work we also consider the total sheaf effective resistance, which $\mathrm{R}_{\mathcal{F}}$ is the sum of the sheaf effective resistance $\mathrm{R}_{\mathcal{F}}(u,v)$ for all pairs of nodes. With vertex-major indexing, write $L_{\mathcal{F}}^+$ as n^2 blocks $(L_{\mathcal{F}}^+)^{ij}$, each of dimension $d \times d$ in the following way:

$$L_{\mathcal{F}}^+ = \begin{pmatrix} (L_{\mathcal{F}}^+)^{11} & \dots & (L_{\mathcal{F}}^+)^{1n} \\ \vdots & \ddots & \vdots \\ (L_{\mathcal{F}}^+)^{n1} & \dots & (L_{\mathcal{F}}^+)^{nn} \end{pmatrix},$$

then a straightforward calculation leads to:

[1] A graph is a 1-dimensional cell complex. Cellular sheaves can be defined in this more general context, but considering them into GNNs realm brings complications we do not discuss in this work.

Proposition D.2. *Let $\langle \cdot, \cdot \rangle_F$ denote the Frobenius inner product, and $\mathbf{1}$ denote a matrix whose entries are all equal to one. Let $\mathcal{F}$ be a sheaf over a connected graph G. Then*

$$\mathrm{R}_{\mathcal{F}} = (n-1)\sum_{i \in V} \langle (L_{\mathcal{F}}^{+})^{ii}, \mathbf{1} \rangle_F - 2\sum_{i<j} \langle (L_{\mathcal{F}}^{+})^{ij}, \mathbf{1} \rangle_F,$$

where $\mathbf{1} \in \mathbb{R}^{n \times n}$ is a matrix of ones.

Proof. Here we use a numerical index notation i, j, which we believe will help to follow the argument. Observe that $R_{\mathcal{F}}(i,j) = \sum_{k,l=1}^{d} \left((L_{\mathcal{F}}^{+})_{kl}^{ii} + (L_{\mathcal{F}}^{+})_{kl}^{jj} \right) - 2 \sum_{k,l=1}^{d} (L_{\mathcal{F}}^{+})_{kl}^{ij}$, for each pair of nodes $i, j \in V$.

To calculate the sheaf effective resistance we need sum all $R_{\mathcal{F}}(i,j)$. So

$$\begin{aligned} R_{\mathcal{F}} &= (n-1)\left(\sum_{ij=1}^{d} (L_{\mathcal{F}}^{+})_{ij}^{11} + \cdots + \sum_{ij=1}^{d} (L_{\mathcal{F}}^{+})_{ij}^{nn} \right) - 2\left(\sum_{ij=1}^{d} (L_{\mathcal{F}}^{+})_{ij}^{12} + \cdots + (L_{\mathcal{F}}^{+})_{ij}^{1n} \right) \\ &\quad - 2\left((L_{\mathcal{F}}^{+})_{ij}^{23} + (L_{\mathcal{F}}^{+})_{ij}^{2n} \right) - \cdots - 2\sum_{ij=1}^{d} (L_{\mathcal{F}}^{+})_{ij}^{n-1n} \\ &= (n-1)\sum_{i \in V} \langle (L_{\mathcal{F}}^{+})^{ii}, \mathbf{1} \rangle_F - 2\sum_{i<j} \langle (L_{\mathcal{F}}^{+})^{ij}, \mathbf{1} \rangle_F. \end{aligned}$$

□

E Proofs and verifications

Example 4.1. Consider the graph with only two vertices u and v, and an edge e between them.

1. If $\mathcal{F}$ is trivial, the sheaf Laplacian is $\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$ whose pseudo-inverse is $\begin{pmatrix} 1/4 & -1/4 \\ -1/4 & 1/4 \end{pmatrix}$, $\mathbb{1}_u = (1,0)$, and $\mathbb{1}_v = (0,1)$. Thus, $R(u,v) = 1$.
2. If $\mathcal{F}$ is a sheaf such that all stalks are isomorphic to $\mathbb{R}^2$, $\mathcal{F}_{u \trianglelefteq e} = \begin{pmatrix} \cos(x) & -\sin(x) \\ \sin(x) & \cos(x) \end{pmatrix}$, and $\mathcal{F}_{v \trianglelefteq e} = \begin{pmatrix} \cos(y) & -\sin(y) \\ \sin(y) & \cos(y) \end{pmatrix}$, the sheaf Laplacian is
$$\begin{pmatrix} 1 & 0 & -\cos(x-y) & -\sin(x-y) \\ 0 & 1 & \sin(x-y) & -\cos(x-y) \\ -\cos(x-y) & \sin(x-y) & 1 & 0 \\ -\sin(x-y) & -\cos(x-y) & 0 & 1 \end{pmatrix},$$
whose pseudo-inverse is
$$\begin{pmatrix} 1/4 & 0 & -\cos(x-y)/4 & -\sin(x-y)/4 \\ 0 & 1/4 & \sin(x-y)/4 & -\cos(x-y)/4 \\ -\cos(x-y)/4 & \sin(x-y)/4 & 1/4 & 0 \\ -\sin(x-y)/4 & -\cos(x-y)/4 & 0 & 1/4 \end{pmatrix},$$
with $\mathbb{1}_u = (1,1,0,0)$, and $\mathbb{1}_v = (0,0,1,1)$. Thus, $R_{\mathcal{F}}(u,v) = \cos(x-y) + 1$.

Proposition 4.1. The pseudo-inverse of $L_{\mathcal{F}}$ is $L_{\mathcal{F}}^{+} = \mathbf{O}^{\top}(L_G^{+} \otimes I_d)\mathbf{O}$.

Proof. Using the fact that $(A \otimes B)^{+} = A^{+} \otimes B^{+}$, and that for orthogonal matrices U and V we have $(UAV)^{+} = V^{\top}A^{+}U^{\top}$, since the matrix $\mathbf{O}$ is orthogonal:

$$\begin{aligned} L_{\mathcal{F}}^{+} &= \left(\mathbf{O}^{\top}(L_G \otimes I_d)\mathbf{O} \right)^{+} \\ &= (\mathbf{O})^{+}(L_G \otimes I_d)^{+}(\mathbf{O}^{\top})^{+} \\ &= (\mathbf{O})^{-1}(L_G^{+} \otimes I_d^{+})(\mathbf{O}^{\top})^{-1} \\ &= \mathbf{O}^{\top}(L_G^{+} \otimes I_d)\mathbf{O}. \end{aligned}$$

□

Proposition 4.2. Let $\mathbf{1}_d \in \mathbb{R}^d$ be a vector of ones, and $M_{uv} = \mathbf{1}_d^\top \mathcal{F}_u^\top \mathcal{F}_v \mathbf{1}_d$. Then

$$\mathrm{R}_{\mathcal{F}}(u,v) = d((L_G^+)_{uu} + (L_G^+)_{vv}) - 2M_{uv}(L_G^+)_{uv}.$$

Proof. Using Definition 4.1 and Proposition 4.1 we obtain

$$R_{\mathcal{F}}(u,v) = (\mathbb{1}_u - \mathbb{1}_v)^\top L_{\mathcal{F}}^+ (\mathbb{1}_u - \mathbb{1}_v) = (\mathbb{1}_u - \mathbb{1}_v)^\top \mathbf{O}^\top (L_G^+ \otimes I_d)\mathbf{O}(\mathbb{1}_u - \mathbb{1}_v).$$

We first note that $\mathbf{O} = \sum_{i=1}^n (\mathbf{e}_i \mathbf{e}_i^\top) \otimes \mathcal{F}_i$ and $\mathbb{1}_u = \mathbf{e}_u \otimes \mathbf{1}_d$, where $\{\mathbf{e}_i\}_{i=1}^n$ is the standard basis of $\mathbb{R}^n$. Thus

$$\mathbf{O}\mathbb{1}_u = \left(\sum_{i=1}^n (\mathbf{e}_i \mathbf{e}_i^\top) \otimes \mathcal{F}_i\right)(\mathbf{e}_u \otimes \mathbf{1}_d) = \sum_{i=1}^n (\mathbf{e}_i \mathbf{e}_i^\top \mathbf{e}_u) \otimes (\mathcal{F}_i \mathbf{1}_d) = \mathbf{e}_u \otimes (\mathcal{F}_u \mathbf{1}_d).$$

Let $s_u = \mathcal{F}_u \mathbf{1}_d$, then $\|s_u\|_2^2 = \mathbf{1}_d^\top \mathcal{F}_u^\top \mathcal{F}_u \mathbf{1}_d = d$. Also, define $M_{uv} = s_u^\top s_v$. Therefore

$$\begin{aligned}
R_{\mathcal{F}}(u,v) &= ((\mathbf{e}_u \otimes s_u) - (\mathbf{e}_v \otimes s_v))^\top (L_G^+ \otimes I_d)((\mathbf{e}_u \otimes s_u) - (\mathbf{e}_v \otimes s_v)) \\
&= ((\mathbf{e}_u^\top \otimes s_u^\top) - (\mathbf{e}_v^\top \otimes s_v^\top))((L_G^+ \mathbf{e}_u \otimes s_u) - (L_G^+ \mathbf{e}_v \otimes s_v)) \\
&= (\mathbf{e}_u^\top L_G^+ \mathbf{e}_u \otimes s_u^\top s_u) - (\mathbf{e}_u^\top L_G^+ \mathbf{e}_v \otimes s_u^\top s_v) - (\mathbf{e}_v^\top L_G^+ \mathbf{e}_u \otimes s_v^\top s_u) + (\mathbf{e}_v^\top L_G^+ \mathbf{e}_v \otimes s_v^\top s_v) \\
&= (L_G^+)_{uu}\|s_u\|_2^2 - (L_G^+)_{uv} s_u^\top s_v - (L_G^+)_{vu} s_v^\top s_u + (L_G^+)_{vv}\|s_v\|_2^2 \\
&= d((L_G^+)_{uu} + (L_G^+)_{vv}) - 2M_{uv}(L_G^+)_{uv}.
\end{aligned}$$

□

Lemma E.1. *(Lemma 3.5 Black et al. [2023]) Let G be a non-bipartite graph. Let u and v be two vertices in G. Then*

$$\mathrm{R}(u,v) = \sum_{j=0}^{\infty} \left(\frac{1}{d_u}(\hat{A}^j)_{uu} + \frac{1}{d_v}(\hat{A}^j)_{vv} - \frac{2}{\sqrt{d_u d_v}}(\hat{A}^j)_{uv}\right).$$

Lemma E.2. *(Lemma 3.2 Black et al. [2023]) Let $u, v \in V$. Suppose $\|\nabla \phi_t\| \leqslant \alpha$ and $\max\{\|\nabla \psi_t\|, 1\} \leqslant \beta$, for all $t = 0, ..., r$, where ∇f denotes the Jacobian of a map f. Then*

$$\left\| \frac{\partial h_u^{(r)}}{\partial x_v} \right\| \leqslant (2\alpha\beta)^r \sum_{j=0}^{r} (\hat{A}^j)_{uv}.$$

Then we can prove:

Theorem 4.1. Let G be a non-bipartite graph, and let $u, v \in V$. Suppose $\|\nabla \phi_l\| \leqslant \alpha$ and $\max\{\|\nabla \psi_t\|, 1\}| \leqslant \beta$. Define $d_{\min} = \min\{d_u, d_v\}$, $d_{\max} = \max\{d_u, d_v\}$, and fix d the dimension of the stalks. Let $\max\{|\mu_2|, ..., |\mu_n|\} \leqslant \mu$. If $(L_G^+)_{uv} \geqslant 0$, then $\left\| \frac{\partial h_u^{(r)}}{\partial x_v} \right\|$ is upper bounded by

$$(2\alpha\beta)^r \frac{d_{max}}{2}\left(\frac{2}{d_{min}}\left(r + 1 + \frac{\mu^{r+1}}{1-\mu}\right) + 4(L_G^+)_{uv} - \frac{\mathrm{R}_{\mathcal{F}}(u,v)}{d}\right).$$

If $(L_G^+)_{uv} < 0$, it is upper bounded by

$$(2\alpha\beta)^r \frac{d_{max}}{2}\left(\frac{2}{d_{min}}\left(r + 1 + \frac{\mu^{r+1}}{1-\mu}\right) - \frac{\mathrm{R}_{\mathcal{F}}(u,v)}{d}\right).$$

Proof. By Lemma E.1 and Equation (3),

$$\mathrm{R}_{\mathcal{F}}(u,v) = d\sum_{j=0}^{\infty} \left(\frac{1}{d_u}(\hat{A}^j)_{uu} + \frac{1}{d_v}(\hat{A}^j)_{vv} - \frac{2}{\sqrt{d_u d_v}}(\hat{A}^j)_{uv}\right) + 2(d - M_{uv})(L_G^+)_{uv}.$$

So

$$\sum_{j=0}^{r}(\hat{A}^j)_{uv} = \frac{\sqrt{d_u d_v}}{2}\left(\sum_{j=0}^{\infty}\left(\frac{1}{d_u}(\hat{A}^j)_{uu} + \frac{1}{d_v}(\hat{A}^j)_{vv}\right) - \sum_{j=r+1}^{\infty}\frac{2}{\sqrt{d_u d_v}}(\hat{A}^j)_{uv} + \frac{2}{d}(d - M_{uv})(L_G^+)_{uv} - \frac{1}{d}\,\mathrm{R}_{\mathcal{F}}(u,v)\right)$$

By Lemma E.2,

$$\begin{aligned}\left\|\frac{\partial h_u^{(r)}}{\partial x_v}\right\| &\leqslant (2\alpha\beta)^r \sum_{j=0}^{r}(\hat{A}^j)_{uv} \\ &\leqslant (2\alpha\beta)^r \frac{d_{max}}{2}\left(\sum_{j=0}^{\infty}\left(\frac{1}{d_u}(\hat{A}^j)_{uu} + \frac{1}{d_v}(\hat{A}^j)_{vv}\right) - \sum_{j=r+1}^{\infty}\frac{2}{\sqrt{d_u d_v}}(\hat{A}^j)_{uv} + \frac{2}{d}(d - M_{uv})(L_G^+)_{uv} - \frac{1}{d}\,\mathrm{R}_{\mathcal{F}}(u,v)\right)\end{aligned}$$

The proof of Theorem 3.3 in Black et al. [2023] shows that $\sum_{j=0}^{\infty}\left(\frac{1}{d_u}(\hat{A}^j)_{uu} + \frac{1}{d_v}(\hat{A}^j)_{vv}\right) - \sum_{j=r+1}^{\infty}\frac{2}{\sqrt{d_u d_v}}(\hat{A}^j)_{uv}$ is bounded by $\frac{2}{d_{min}}\left((r+1) + \frac{\mu^{r+1}}{1-\mu}\right)$. Since $-d \leqslant M_{uv} \leqslant d$, if $(L_G^+)_{uv} \geqslant 0$ we obtain

$$\left\|\frac{\partial h_u^{(r)}}{\partial x_v}\right\| \leqslant (2\alpha\beta)^r \frac{d_{max}}{2}\left(\frac{2}{d_{min}}\left(r + 1 + \frac{\mu^{r+1}}{1-\mu}\right) + 4(L_G^+)_{uv} - \frac{\mathrm{R}_{\mathcal{F}}(u,v)}{d}\right).$$

If $(L_G^+)_{uv} < 0$, we obtain

$$\left\|\frac{\partial h_u^{(r)}}{\partial x_v}\right\| \leqslant (2\alpha\beta)^r \frac{d_{max}}{2}\left(\frac{2}{d_{min}}\left(r + 1 + \frac{\mu^{r+1}}{1-\mu}\right) - \frac{\mathrm{R}_{\mathcal{F}}(u,v)}{d}\right).$$

□

Theorem 4.2. Let $\langle\cdot,\cdot\rangle_F$ denote the Frobenius inner product and define M as in Proposition 4.2. Then

$$\mathrm{R}_{\mathcal{F}} = dn\,\mathrm{Tr}(L_G^+) - \langle M, L_G^+\rangle_F = d\,\mathrm{R} - \langle M, L_G^+\rangle_F$$

Proof. On one hand, by Proposition 4.2, we have

$$\begin{aligned}\mathrm{R}_{\mathcal{F}} &= d\sum_{u<v}\left((L_G^+)_{uu} + (L_G^+)_{vv}\right) - 2\sum_{u<v} M_{uv}(L_G^+)_{uv} \\ &= d(n-1)\sum_{u\in V}(L_G^+)_{uu} - 2\sum_{u<v} M_{uv}(L_G^+)_{uv} \\ &= d(n-1)\,\mathrm{Tr}(L_G^+) - 2\sum_{u<v} M_{uv}(L_G^+)_{uv}\end{aligned}$$

On the other hand, observe that

$$\begin{aligned}\langle M, L_G^+\rangle_F = \sum_{u,v\in V} M_{uv}(L_G^+)_{uv} &= \sum_{u\in V} M_{uu}(L_G^+)_{uu} + \sum_{u\neq v} M_{uv}(L_G^+)_{uv} \\ &= d\,\mathrm{Tr}(L_G^+) + 2\sum_{u<v} M_{uv}(L_G^+)_{uv},\end{aligned}$$

where $u < v$ means we are considering only the upper triangular part of the matrix, and the equality holds because both L_G^+ and M are symmetric. Thus

$$2\sum_{u<v} M_{uv}(L_G^+)_{uv} = \langle M, L_G^+\rangle_F - d\,\mathrm{Tr}(L_G^+),$$

Combining the above, we obtain

$$\mathrm{R}_{\mathcal{F}} = d(n-1)\,\mathrm{Tr}(L_G^+) - (\langle M, L_G^+\rangle_F - d\,\mathrm{Tr}(L_G^+)) = dn\,\mathrm{Tr}(L_G^+) - \langle M, L_G^+\rangle_F.$$

□

Proposition 4.3. Let G be a connected graph with at least two nodes and $\mathcal{F}$ a flat $O(d)$-bundle over G. Then $\frac{R_{\mathcal{F}}}{d} \leqslant R$, with equality holding if and only if $\mathcal{F}_v\mathbf{1}_d = c$ for some $c \in \mathbb{R}^d$ and all $v \in V$. In particular, for every $d \geqslant 1$, there exists a flat $O(d)$-bundle $\mathcal{F}$ such that $\frac{R_{\mathcal{F}}}{d} < R$.

Proof. Recall that S is a matrix with $S_u = \mathbf{1}_d^\top \mathcal{F}_u^\top$ and $M = SS^\top$, therefore

$$\langle M, L_G^+\rangle = \mathrm{Tr}(S^\top L_G^+ S) \geqslant 0,$$

since $L_G^+ \succeq 0$, and then $\frac{R_{\mathcal{F}}}{d} \leqslant R$ by Theorem 4.2. If the underlying graph G is connected, $\ker L_G = (\mathbf{1}_n)$, thus $\frac{R_{\mathcal{F}}}{d} = R$ precisely when $\mathcal{F}_v\mathbf{1}_d = c$, for some $c \in \mathbb{R}^d$ and all $v \in V$. Therefore the inequality is strict whenever $\mathcal{F}_v\mathbf{1}_d$ is non-constant. We proceed to show a family of sheaves for which effective resistance is decreased.

Consider any non-trivial vertex partition $V = A \sqcup B$ and define $\varepsilon_v = \begin{cases} +1, & v \in A, \\ -1, & v \in B \end{cases}$. Set $\mathcal{F}_v = \mathrm{diag}(\varepsilon_v, 1, \ldots, 1) \in O(d)$, then for two nodes $v \in A$ and $u \in B$ we get $S = [\varepsilon \quad \mathbf{1}_n \quad \ldots \quad \mathbf{1}_n]$, therefore $\mathrm{Tr}(S^\top L_G^+ S) = \varepsilon^\top L_G^+ \varepsilon + \sum_{j=2}^d \mathbf{1}_n^\top L_G^+ \mathbf{1}_n = \varepsilon^\top L_G^+ \varepsilon > 0$, since ε is non-constant. By Theorem 4.2, we conclude that $R_{\mathcal{F}} = dR - \mathrm{Tr}(S^\top L_G^+ S) \implies \frac{R_{\mathcal{F}}}{d} < R$. □

Corollary 4.1. Let G be a non-bipartite graph, and let $u, v \in V$. Suppose $\|\nabla\phi_l\| \leqslant \alpha$ and $\max\{\|\nabla\psi_t\|, 1\}| \leqslant \beta$. Define $d_{\min} = \min\{d_u, d_v\}$, $d_{\max} = \max\{d_u, d_v\}$, and let d be the dimension of the stalks. Let $\max\{|\mu_2|, ..., |\mu_n|\} \leqslant \mu$. Then

$$\sum_{u\neq v\in V} \left\| \frac{\partial h_u^{(r)}}{\partial x_v} \right\| \leqslant (2\alpha\beta)^r \frac{d_{max}}{2} \left(\frac{n(n-1)}{d_{min}} \left(r + 1 + \frac{\mu^{r+1}}{1-\mu} \right) - \frac{\mathrm{R}_{\mathcal{F}}}{d} \right).$$

Proof. Observe that

$$\begin{aligned}
&\sum_{u\neq v\in V} (2\alpha\beta)^{r+1} \frac{d_{max}}{2} \left(\frac{2}{d_{min}} \left(r + 1 + \frac{\mu^{r+1}}{1-\mu} \right) + 4(L_G^+)_{uv} - \frac{\mathrm{R}_{\mathcal{F}}(u,v)}{d} \right) \\
&= (2\alpha\beta)^r \frac{d_{max}}{2} \left(\frac{n(n-1)}{d_{min}} \left(r + 1 + \frac{\mu^{r+1}}{1-\mu} \right) - 4\,\mathrm{Tr}(L_G^+) - \frac{2\,\mathrm{R}_{\mathcal{F}}}{d} \right) \\
&\leqslant (2\alpha\beta)^r \frac{d_{max}}{2} \left(\frac{n(n-1)}{d_{min}} \left(r + 1 + \frac{\mu^{r+1}}{1-\mu} \right) - \frac{\mathrm{R}_{\mathcal{F}}}{d} \right),
\end{aligned}$$

because the sum of each line in (L_G^+) equals to 0, $\mathrm{Tr}(L_G^+) \geqslant 0$, and $\mathrm{R}_{\mathcal{F}} \leqslant 2\,\mathrm{R}_{\mathcal{F}}$.

Similarly,

$$\begin{aligned}
&\sum_{u\neq v\in V} (2\alpha\beta)^{r+1} \frac{d_{max}}{2} \left(\frac{2}{d_{min}} \left(r + 1 + \frac{\mu^{r+1}}{1-\mu} \right) - \frac{\mathrm{R}_{\mathcal{F}}(u,v)}{d} \right) \\
&= (2\alpha\beta)^r \frac{d_{max}}{2} \left(\frac{n(n-1)}{d_{min}} \left(r + 1 + \frac{\mu^{r+1}}{1-\mu} \right) - \frac{2\,\mathrm{R}_{\mathcal{F}}}{d} \right) \\
&\leqslant (2\alpha\beta)^r \frac{d_{max}}{2} \left(\frac{n(n-1)}{d_{min}} \left(r + 1 + \frac{\mu^{r+1}}{1-\mu} \right) - \frac{\mathrm{R}_{\mathcal{F}}}{d} \right).
\end{aligned}$$

□

Lemma 4.1. Let G be a connected bipartite graph. Then $\Delta_G^+ = \frac{1}{2}P_- + \sum_{k=0}^{\infty}(\hat{A}^k - P_+ - (-1)^k P_-)$, where P_+ and P_- are the outer product of the eigenvectors with eigenvalues 1 and -1, respectively.

Proof. Let $G = (U \sqcup W, E)$, $\varepsilon_v = \begin{cases} +1, & u \in W \\ -1, & v \in U \end{cases}$, and $\mathrm{vol}(G) = \sum_{v\in V} d_v$. The kernel of the Laplacian is $\ker L_G = (\mathbf{1}_n)$, and the symmetrically normalized Laplacian is $\Delta_G = I - D^{-1/2}AD^{-1/2} = I - \hat{A}$. Thus

$$x \in \ker \Delta_G \iff (I - \hat{A})x = 0 \iff \hat{A}x = x.$$

We also have

$$x \in \ker \Delta_G \iff D^{-1/2}L_G D^{-1/2}x = 0 \iff L_G(D^{-1/2}x) = 0 \iff D^{-1/2}x \in \ker L_G.$$

We conclude that if $x \in \ker \Delta_G$, then $x = cD^{1/2}\mathbf{1}_n$, and therefore the normalized eigenvector of $\hat{A}$ corresponding to the eigenvalue 1 is $q_+ = \frac{D^{1/2}\mathbf{1}_n}{\sqrt{\mathrm{vol}(G)}}$. Now let $(u, v) \in E$, then $\varepsilon_u = -\varepsilon_v$, thus

$$(A\varepsilon)_v = \sum_{u\in\mathcal{N}(v)} \varepsilon_u = -d_v\varepsilon_v,$$

i.e. $A\varepsilon = -D\varepsilon$. This yields

$$\hat{A}D^{1/2}\varepsilon = D^{-1/2}A\varepsilon = D^{-1/2}(-D\varepsilon) = -D^{1/2}\varepsilon,$$

and we conclude that the normalized eigenvector with eigenvalue -1 is $q_- = \frac{D^{1/2}\varepsilon}{\sqrt{\mathrm{vol}(G)}}$. Defining $P_+ = q_+q_+^\top$, $P_- = q_-q_-^\top$ and $P_i = q_iq_i^\top$, where the q's are the eigenvectors of $\hat{A}$, we get

$$\hat{A} = Q\,\mathrm{diag}(1, -1, \mu_2, \ldots, \mu_{n-1})\,Q^\top = P_+ - P_- + \sum_{i=2}^{n-1}\mu_i P_i.$$

Furthermore, since $B = \{q_+, q_-, q_2, \ldots, q_{n-1}\}$ forms an orthonormal basis, we get $I = P_+ + P_- + \sum_{i=2}^{n-1} P_i$, therefore

$$\Delta_G = I - \hat{A} = 2P_- + \sum_{i=2}^{n-1}(1 - \mu_i)P_i \implies \Delta_G^+ = \frac{1}{2}P_- + \sum_{i=2}^{n-1}\frac{1}{1-\mu_i}P_i.$$

Since $|\mu_i| < 1$, we have $\frac{1}{1-\mu_i} = \sum_{k=0}^{\infty}\mu_i^k$, thus

$$\Delta_G^+ = \frac{1}{2}P_- + \sum_{i=2}^{n-1}\sum_{k=0}^{\infty}\mu_i^k P_i = \frac{1}{2}P_- + \sum_{k=0}^{\infty}\sum_{i=2}^{n-1}\mu_i^k P_i.$$

But $\hat{A}^k = P_+ + (-1)^k P_- + \sum_{i=2}^{n-1}\mu_i^k P_i \implies \sum_{i=2}^{n-1}\mu_i^k P_i = \hat{A}^k - P_+ - (-1)^k P_-$, which yields

$$\Delta_G^+ = \frac{1}{2}P_- + \sum_{k=0}^{\infty}(\hat{A}^k - P_+ - (-1)^k P_-).$$

□

Proposition 4.4. Let G be a connected bipartite graph, $b_{uv} = \mathbb{1}_u - \mathbb{1}_v$, $z_{uv} = D^{-1/2}b_{uv}$, and $\gamma_{uv} = z_{uv}^\top P_- z_{uv}$. Then

$$R(u, v) = \frac{\gamma_{uv}}{2} + \sum_{k=0}^{\infty}\left(\frac{(\hat{A}^k)_{uu}}{d_u} + \frac{(\hat{A}^k)_{vv}}{d_v} - \frac{2(\hat{A}^k)_{uv}}{\sqrt{d_u d_v}} - (-1)^k\gamma_{uv}\right)$$

Proof. The effective resistance can be defined as $z_{uv}^\top\Delta_G^+ z_{uv}$, where $z_{uv} = D^{-1/2}(\mathbb{1}_u - \mathbb{1}_v)$. Therefore, by the previous lemma

$$R(u, v) = z_{uv}^\top\left(\frac{1}{2}P_- + \sum_{k=0}^{\infty}(\hat{A}^k - P_+ - (-1)^k P_-)\right)z_{uv}.$$

Notice that $q_+^\top z_{uv} = \frac{\mathbf{1}_n^\top D^{1/2}}{\sqrt{\mathrm{vol}(G)}} D^{-1/2}(\mathbb{1}_u - \mathbb{1}_v) = \frac{1}{\sqrt{\mathrm{vol}(G)}} \mathbf{1}_n^\top (\mathbb{1}_u - \mathbb{1}_v) = 0$, thus $z_{uv}^\top P_+ z_{uv} = 0$. Define $\gamma_{uv} = z_{uv}^\top P_- z_{uv}$, then

$$R(u,v) = \frac{\gamma_{uv}}{2} + \sum_{k=0}^{\infty} (z_{uv}^\top \hat{A}^k z_{uv} - (-1)^k \gamma_{uv}).$$

Now, notice that

$$z_{uv}^\top \hat{A}^k z_{uv} = \left(\frac{\mathbb{1}_u^\top}{\sqrt{d_u}} - \frac{\mathbb{1}_v^\top}{\sqrt{d_v}} \right) \hat{A}^k \left(\frac{\mathbb{1}_u}{\sqrt{d_u}} - \frac{\mathbb{1}_v}{\sqrt{d_v}} \right) = \frac{(\hat{A}^k)_{uu}}{d_u} + \frac{(\hat{A}^k)_{vv}}{d_v} - 2\frac{(\hat{A}^k)_{uv}}{\sqrt{d_u d_v}},$$

thus

$$R(u,v) = \frac{\gamma_{uv}}{2} + \sum_{k=0}^{\infty} \left(\frac{(\hat{A}^k)_{uu}}{d_u} + \frac{(\hat{A}^k)_{vv}}{d_v} - \frac{2(\hat{A}^k)_{uv}}{\sqrt{d_u d_v}} - (-1)^k \gamma_{uv} \right).$$

□

Theorem 4.3. Let G be a connected bipartite graph. Let $u, v \in V = U \sqcup W$, $\|\nabla \phi_l\| \leqslant \alpha$ and $\max\{\|\nabla \psi_l\|, 1\| \leqslant \beta$, $d_{\min} = \min\{d_u, d_v\}$, $d_{\max} = \max\{d_u, d_v\}$, and $\overline{\mu} = \max_{i=2,\dots,n-1} |\mu_i| < 1$. Then

$$\left\| \frac{\partial h_u^{(r)}}{\partial x_v} \right\| \leqslant (2\alpha\beta)^r \frac{d_{\max}}{2} \left(\frac{2}{d_{\min}} \left(r + 1 + \frac{\overline{\mu}^{r+1}}{1-\overline{\mu}} \right) + \frac{1}{2}\gamma_{uv} - R(u,v) \right).$$

Proof. We have

$$R(u,v) = \frac{\gamma_{uv}}{2} + \sum_{k=0}^{r} \left(z_{uv}^\top \hat{A}^k z_{uv} - (-1)^k \gamma_{uv} \right) + T_r(u,v),$$

where $T_r(u,v) = \sum_{k=r+1}^{\infty} \left(z_{uv}^\top \hat{A}^k z_{uv} - (-1)^k \gamma_{uv} \right)$. We can write z_{uv} in the orthonormal basis as

$$z_{uv} = \left(q_-^\top z_{uv} \right) q_- + \sum_{i=2}^{n-1} \left(q_i^\top z_{uv} \right) q_i,$$

since $z_{uv} \perp q_+$. Recalling that $\hat{A}^k q_- = (-1)^k q_-$, $\hat{A}^k q_i = \mu_i^k q_i$, $q_- \perp q_i$ for every node, and $q_i \perp q_j$ for $i \neq j$, we get:

$$\begin{aligned} z_{uv}^\top \hat{A}^k z_{uv} &= \left[\left(q_-^\top z_{uv} \right) q_-^\top + \sum_{i=2}^{n-1} \left(q_i^\top z_{uv} \right) q_i^\top \right] \hat{A}^k \left[\left(q_-^\top z_{uv} \right) q_- + \sum_{i=2}^{n-1} \left(q_i^\top z_{uv} \right) q_i \right] \\ &= \left(q_-^\top z_{uv} \right)^2 (-1)^k + \sum_{i=2}^{n-1} (q_i^\top z)^2 \mu_i^k, \end{aligned}$$

thus $z_{uv}^\top \hat{A}^k z_{uv} - \gamma_{uv}(-1)^k = \sum_{i=2}^{n-1} (q_i^\top z)^2 \mu_i^k$. Substituting into $T_r(u,v)$:

$$T_r(u,v) = \sum_{k=r+1}^{\infty} \sum_{i=2}^{n-1} (q_i^\top z)^2 \mu_i^k = \sum_{i=2}^{n-1} (q_i^\top z)^2 \sum_{k=r+1}^{\infty} \mu_i^k = \sum_{i=2}^{n-1} (q_i^\top z)^2 \frac{\mu_i^{r+1}}{1-\mu_i},$$

since $|\mu_i| < 1$ for all $i = 2, \dots, n-1$ implies $\frac{\mu_i^{r+1}}{1-\mu_i} = \sum_{k=r+1}^{\infty} \mu_i^k$. Notice that, by Parseval's identity,

$$\|z_{uv}\|^2 = (q_+^\top z_{uv})^2 + (q_-^\top z_{uv})^2 + \sum_{i=2}^{n-1} (q_i^\top z_{uv})^2 = (q_-^\top z_{uv})^2 + \sum_{i=2}^{n-1} (q_i^\top z_{uv})^2,$$

since $z_{uv} \perp q_+$. Therefore, if $\overline{\mu} = \max_{i=2,\dots,n-1} |\mu_i|$ then

$$|T_r(u,v)| \leqslant \frac{\overline{\mu}^{r+1}}{1-\overline{\mu}} \sum_{i=2}^{n-1} (q_i^\top z_{uv})^2 \leqslant \frac{\overline{\mu}^{r+1}}{1-\overline{\mu}} \|z_{uv}\|^2 = \left(\frac{1}{d_u} + \frac{1}{d_v} \right) \frac{\overline{\mu}^{r+1}}{1-\overline{\mu}}.$$

We now combine this with the proposition above:

$$R(u,v) = \frac{\gamma_{uv}}{2} + \sum_{k=0}^{r}\left(\frac{(\hat{A}^k)_{uu}}{d_u} + \frac{(\hat{A}^k)_{vv}}{d_v} - \frac{2(\hat{A}^k)_{uv}}{\sqrt{d_u d_v}} - (-1)^k\gamma_{uv}\right) + T_r(u,v),$$

which yields

$$\sum_{k=0}^{r}(\hat{A}^k)_{uv} = \frac{\sqrt{d_u d_v}}{2}\left(\sum_{k=0}^{r}\left(\frac{(\hat{A}^k)_{uu}}{d_u} + \frac{(\hat{A}^k)_{vv}}{d_v} - (-1)^k\gamma_{uv}\right) + \frac{\gamma_{uv}}{2} - R(u,v) + T_r(u,v)\right).$$

Now, we have $\sum_{k=0}^{r}(-1)^k = \frac{1+(-1)^r}{2}$ and $\sum(-1)^k\gamma_{uv} = (\frac{1}{2} - \frac{1+(-1)^r}{2})\gamma_{uv} = \frac{(-1)^{r+1}}{2}\gamma_{uv} \leqslant \frac{1}{2}\gamma_{uv}$. Combining everything:

$$\begin{aligned}\sum_{k=0}^{r}(\hat{A}^k)_{uv} &\leqslant \frac{\sqrt{d_u d_v}}{2}\left(\sum_{k=0}^{r}\left(\frac{(\hat{A}^k)_{uu}}{d_u} + \frac{(\hat{A}^k)_{vv}}{d_v}\right) + \frac{1}{2}\gamma_{uv} - R(u,v) + \left(\frac{1}{d_u} + \frac{1}{d_v}\right)\frac{\overline{\mu}^{r+1}}{1-\overline{\mu}}\right)\\ &\leqslant \frac{d_{\max}}{2}\left(\sum_{k=0}^{r}\left(\frac{(\hat{A}^k)_{uu}}{d_u} + \frac{(\hat{A}^k)_{vv}}{d_v}\right) + \frac{1}{2}\gamma_{uv} - R(u,v) + \frac{2}{d_{\min}}\frac{\overline{\mu}^{r+1}}{1-\overline{\mu}}\right)\end{aligned}$$

Notice that

$$|(\hat{A}^k)_{uu}| = |\mathbb{1}_u^\top \hat{A}^k \mathbb{1}_u| \leqslant \|\mathbb{1}_u^\top\|_2 \|\hat{A}^k\|_2 \|\mathbb{1}_u\|_2 \leqslant \|\hat{A}^k\|_2,$$

and since $\|\hat{A}\|_2 = 1$ we get $\|\hat{A}^k\|_2 \leqslant \|\hat{A}\|_2^k = 1$. Thus $(\hat{A}^k)_{uu} = |(\hat{A}^k)_{uu}| \leqslant 1$ and

$$\sum_{k=0}^{r}\left(\frac{(\hat{A}^k)_{uu}}{d_u} + \frac{(\hat{A}^k)_{vv}}{d_v}\right) \leqslant \sum_{k=0}^{r}\left(\frac{1}{d_u} + \frac{1}{d_v}\right) \leqslant \sum_{k=0}^{r}\frac{2}{d_{\min}} = \frac{2}{d_{\min}}(r+1).$$

We conclude that

$$\sum_{k=0}^{r}(\hat{A}^k)_{uv} \leqslant \frac{d_{\max}}{2}\left(\frac{2}{d_{\min}}\left(r+1+\frac{\overline{\mu}^{r+1}}{1-\overline{\mu}}\right) + \frac{1}{2}\gamma_{uv} - R(u,v)\right).$$

Plugging this into Lemma 3.2 of Black et al. [2023] yields the desired inequality. □

Remark E.1. The corollary for the total effective resistance follows from the above theorem, where the additional term accounting for bipartiteness becomes $\frac{4|U||W|}{\mathrm{vol}(G)}$, since

$$q_- z_{uv} = \frac{\varepsilon^\top D^{1/2}}{\sqrt{\mathrm{vol}(G)}} D^{-1/2}(\mathbb{1}_u - \mathbb{1}_v) = \frac{\varepsilon_u - \varepsilon_v}{\sqrt{\mathrm{vol}(G)}},$$

thus $\gamma_{uv} = z_{uv}^\top P_- z_{uv} = \frac{(\varepsilon_u - \varepsilon_v)^2}{\mathrm{vol}(G)} = \begin{cases} 0, & u, v \text{ on the same side} \\ \frac{4}{\mathrm{vol}(G)}, & \text{otherwise} \end{cases}$.

F Experimental details

In this section, we provide additional details on the experimental setup, such as hyperparameter configurations and computational resources. For the real-world datasets related to over-squashing, we choose $d \in \{2,3,4,5\}$, three options of GNN architectures to learn restriction maps — GCN, GIN, and SAGE — with $\ell_{\mathrm{GNN}} \in \{1,2,3,4,5\}$. We perform grid search for the hyperparameter configurations reported in Table 4. The code used for the experiments can be found in `https://github.com/andrerg02/Sheaf_Effective_Resistance`

For the ECHO-Synth benchmarks, we reproduce the experimental setup of Miglior et al. [2025], performing Bayesian Optimization based on a GP prior in the chosen hyperparameter space. We fix the same range for hidden dimension, learning rate and weight decay proposed by the authors, layers from 10 to 40, stalk dimension and number of layers for a fixed SAGE backbone from 2 to 6, and backbone hidden dimension from 4 to 64.

Compute resources. Hyperparameter search for Table 2 was performed on an internal cluster equipped with NVIDIA V100, A100, and H100 GPUs. Experiments for Table 3 and Figure 5 were run on a node with an Intel Xeon CPU (12 cores @ 2.20GHz) and a single NVIDIA A100-SXM4 (80GB) GPU. The remaining experiments were carried out on a workstation with two NVIDIA GeForce RTX 4090 (24GB).

Table 4: Model hyperparameter configurations used across all datasets. The Node and Graph Classification columns refer to Figure 4, the TreeNeighborsMatch column refers to Figure 1, and the Heterophilic Graphs column refers to Table 2.

Parameter	Node Class.	Graph Class.	TreeNghbrsMatch	Heteroph. Graphs
sheaf dimension	2–5	2–5	4	2–5
# layers	3	4	$r+1$	1–5
# hidden channels	128	64	32	32, 64
GNN architecture	gcn, gin, sage	gcn, gin, sage	sage	sage
# of GNN layers	1–5	1–5	r+1	0–3
GNN dimension	128	64	32	32, 64
dropout	0.5	0.0	0.0	0.2
left weights	✓	✓	✓	✓, ×
right weights	✓	✓	✓	✓, ×
learning rate	0.001	0.001	0.001	0.02 , 0.002

G Computational Complexity

Here we compare the time complexity of FlatNSD against the time complexity of NSD with orthogonal maps. Let n be the number of vertices, m the number of edges, d the dimension of the stalks, h the number of channels, and $c = d \times h$ dimension.

NSD. Bodnar et al. [2022] shows that the time complexity of NSD with orthogonal restrictions maps is $\mathbf{O}(\mathbf{n}(\mathbf{c^2} + \mathbf{d^3}) + \mathbf{m}(\mathbf{cd^2} + \mathbf{d^3}))$, where $d \in \{1, 2, 3, 4, 5\}$.

Importantly, Bodnar et al. [2022] learns orthogonal maps from a composition of Householder reflections Mhammedi et al. [2017]. We use the same strategy to build the orthogonal maps in FlatNSD.

FlatNSD. We derive the time complexity of FlatNSD as follows: we need $O(d^2 n)$ to embed graph features in the sheaf stalks, $O(mcd)$ for the message-passing, $O(ncd)$ whenever we apply W_1, and $O(nch)$ whenever we apply W_2.

Thus, the total time complexity is $\mathbf{O}(\mathbf{cn}(\mathbf{d} + \mathbf{h}) + \mathbf{mcd})$, where $d \in \{2, 3, 4, 5\}$.

The computational gain of FlatNSD comes from learning a single restriction map to each node instead of two restriction maps to each edges (i.e., we have flat vector bundles instead of merely vector bundles), and from an optimized implementation based on message-passing.

Table 5: Total runtime in seconds of FlatNSD and GCN with different rewiring methods for both node and graph classification tasks. DR requires graphs to have at least 3 nodes, but some graphs from Enzymes have only 2 nodes.

Method	Cora	Citeseer	Texas	Cornell	Wisconsin	Chameleon	Enzymes	IMDB	Mutag	Proteins
GCN+SDRF	926.86	815.83	640.10	676.62	676.62	2087.36	**2502.20**	**2307.62**	**710.50**	**3598.63**
GCN+FOSR	945.13	825.15	522.83	677.56	677.56	2158.64	2601.29	2319.26	760.12	5962.85
GCN+BORF	1082.56	835.79	775.45	785.78	727.22	2093.88	4213.78	6410.12	1216.25	7330.80
GCN+DR	920.43	813.23	641.02	697.87	666.79	2148.66	-	2339.10	730.28	3637.15
GCN+GTR	898.48	791.14	638.70	710.84	647.71	2117.94	2589.34	2247.56	736.23	4023.95
GCN+LASER	1150.83	820.52	609.81	746.45	786.21	2143.54	2633.41	2296.16	741.60	4370.87
GCN+GOKU	934.48	805.99	648.15	662.81	679.77	2332.12	3780.77	3807.83	1078.83	4040.95
$\text{FlatNSD}_{\text{GCN}}$	**399.96**	**419.63**	**341.45**	**338.67**	**344.83**	**398.92**	4093.99	6674.47	1309.02	8031.02

H Runtime Comparison

We provide Table 5 to compare the total runtime of FlatNSD against GCN coupled with seven graph rewiring strategies. On node classification benchmarks, FlatNSD exhibits superior efficiency,

reducing runtime by up to $5\times$ on dense datasets like Chameleon compared to SDRF. Regarding graph classification tasks (Enzymes, IMDB, Mutag, and Proteins), while FlatNSD incurs a higher computational cost due to per-graph processing overhead, its total runtime remains within the same order of magnitude as the baseline methods.

Table 6: Model parameters configurations used on runtime comparison experiment.

Parameter	Node Classification	Graph Classification
sheaf dimension	4	4
# layers	2	2
# hidden channels	32	64
GNN architecture	GCN	GCN
# of GNN layers	5	5
GNN dimension	32	32
dropout	0.5	0.0
left weights	✓	✓
right weights	✓	✓
learning rate	0.001	0.001

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The constructions and contributions outlined in both the abstract and the introduction are developed in the main text.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes],

   Justification: The limitations of the paper are discussed in Section 8.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes].

Justification: The theoretical results are proved in Section E.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We explain the experimental setup in main text, and provide additional details in Section F.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We plan to release the code via a public repository once the paper is accepted.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Experimental details are described in Section F.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report mean and standard deviation for the experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Compute resources were described in Section F.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Authors have reviewed the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We present impacts in Section 8.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Our implementation builds on the work of Bodnar et al. [2022], which is distributed under the permissive Apache License. We also use the torch householder library Obukhov [2021].

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs were employed solely for grammar and style corrections throughout the manuscript.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.