Joint Diffusion Processes as an Inductive Bias in Sheaf Neural Networks

Ferran Hernandez Caralt[*] [1] **Guillermo Bernárdez Gil** [2] **Iulia Duta** [3] **Pietro Liò** [3] **Eduard Alarcón Cot** [4]

Abstract

Sheaf Neural Networks (SNNs) naturally extend Graph Neural Networks (GNNs) by endowing a cellular sheaf over the graph, equipping nodes and edges with vector spaces and defining linear mappings between them. While the attached geometric structure has proven to be useful in analyzing heterophily and oversmoothing, so far the methods by which the sheaf is computed do not always guarantee a good performance in such settings. In this work, drawing inspiration from opinion dynamics concepts, we propose two novel sheaf learning approaches that *(i)* provide a more intuitive understanding of the involved structure maps, *(ii)* introduce a useful inductive bias for heterophily and oversmoothing, and *(iii)* infer the sheaf in a way that does not scale with the number of features, thus using fewer learnable parameters than existing methods. In our evaluation, we show the limitations of the real-world benchmarks used so far on SNNs, and design a new synthetic task –leveraging the symmetries of n-dimensional ellipsoids– that enables us to better assess the strengths and weaknesses of sheaf-based models. Our extensive experimentation on these novel datasets reveals valuable insights into the scenarios and contexts where basic SNNs and our proposed approaches can be beneficial.

1. Introduction

Graph Neural Networks (GNNs) (Scarselli et al., 2008; Kipf & Welling, 2017; Veličković et al., 2018) have become very popular as a way to model and process relational data, demonstrating remarkable performance in a wide range of applications and tasks (Zhou et al., 2020; Qiu et al., 2018; Rusek et al., 2019). Nonetheless, two main problems frequently appear when dealing with GNNs: they perform poorly on heterophilic settings (Zhu et al., 2020; Luan et al., 2022), and exhibit over-smoothing behaviour (Nt & Maehara, 2019; Oono & Suzuki, 2019). The first problem emerges because most models implicitly assume graph homophily (i.e. edges are expected to connect similar nodes), whereas the second one has to do with the tendency of deep GNNs to produce features too uniform to be useful.

Sheaf Neural Networks (SNNs) (Hansen & Gebhart, 2020), originally designed as a natural generalization of GNNs with a *possibly* non-trivial underlying graph ”geometry”, have been proven a powerful tool to analyze the aforementioned issues (Bodnar et al., 2022). In particular, SNNs endow a (cellular) sheaf (Curry, 2014) over the graph, equipping each node and edge with a vector space and defining a linear application between these spaces for each incident edge-node. The particular sheaf choice is reflected in the graph Laplacian (Hansen & Ghrist, 2019) operator, the properties of the corresponding diffusion equation, and the convolutional models that discretise the equation (Bodnar et al., 2022). At first, the sheaf was defined through domain knowledge (Hansen & Gebhart, 2020), but recent papers have proposed new ways to work with this structure, such as learning the sheaf using a learnable parametric function (Bodnar et al., 2022), inferring the graph connection Laplacian directly from data at preprocessing time (Barbero et al., 2022), using the wave equation on sheaves to design the model (Suk et al., 2022), or even introducing non-linearities in the process (Zaghen, 2024). However, the sheaf structure is typically inferred through universal approximators (e.g. via a Multi-Layer Perceptron (MLP)) that do not intrinsically incorporate an inductive bias for heterophilic data.

[*]Work done while the author was at the University of Cambridge [1]FME, FIB and ETSETB, Universitat Politècnica de Catalunya-BarcelonaTech, Barcelona, Spain [2]Department of Electrical and Computer Engineering, UC Santa Barbara, Santa Barbara, United States of America [3]Department of Computer Science and Technology, University of Cambridge, Cambridge, United Kingdom [4]Department of Electronic Engineering, Universitat Politècnica de Catalunya-BarcelonaTech, Spain, Barcelona. Correspondence to: Ferran Hernandez Caralt <ferran.hernandez.caralt@estudiantat.upc.edu, ferranhernandezc@gmail.com>.

Proceedings of the Geometry-grounded Representation Learning and Generative Modeling Workshop (GRaM) at the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 251, 2024. Copyright 2024 by the author(s).

Contributions: Motivated by this fact, in this work we propose new SNN variants specifically **tailored for learning on heterophilic data**, and which learn the sheaf **explicitly** to prevent oversmoothing. Drawing inspiration from (Hansen & Ghrist, 2021) and some of its opinion dynamics-based interpretations of sheaves, we introduce a SNN formulation whose theoretical guarantees of class separation **no longer depend on learnable parameters**. As a practical result of this, our proposed models are able to compute the sheaf with a significantly lower number of parameters compared to existing SNN models. Lastly, and in order to overcome the current limitations of SNN benchmarks, we also propose a **novel synthetic framework** to evaluate the performance of sheaf-based approaches on node classification tasks. In this new framework, we leverage symmetries of the surface of n-dimensional ellipsoids to create non-noisy distinguishable classes **where, even under high homophily, GCNs suffer from oversmoothing**, thus making the separation of classes a hard problem for regular GNNs. In this new setting, we show how the different approaches to learn the sheaf may be advantageous or disadvantageous depending on the characteristics of the data.

2. Background

Let us begin contextualizing SNNs by briefly revisiting one of the most popular GNNs: the Graph Convolutional Network (GCN) (Kipf & Welling, 2017). The t layer of a GCN is typically formulated as

$$X(t) = X(t-1) - \sigma(\hat{D}^{-\frac{1}{2}} L \hat{D}^{-\frac{1}{2}} X(t-1) W(t)), \quad (1)$$

where $W(t)$ and $X(t)$ represent the weights and node features at step t, σ a non-linear activation function, $L = I - A$ the graph Laplacian, and $\hat{D}$ the node degree matrix of $A + I$. We note that this can be seen as using an MLP on top of the following Ordinary Differential Equation (ODE) over the original node signal:

$$\frac{d}{dt} X = -LX, \quad (2)$$

which in turn satisfies the following property:

Proposition 2.1. *Let $G = (V, E)$ be a graph with an associated graph Laplacian L and $X(t)$ a node signal satisfying (2), then $lim_{t \to \infty} X(t)$ is constant on all connected components.*

Proof in Appendix A.

This behaviour, root of the so-called **oversmoothing** phenomena, is a common issue when handling graph classification tasks with multiple-layered GNNs (Chen et al., 2020), preventing them from building discriminative representations for different classes. In particular, since the ODE-based diffusion process is locally defined through graph connectivity, oversmoothing problems are even more compelling when dealing with heterophilic data (Yan et al., 2022) (i.e. graphs where nodes from different classes tend to be connected). To address this particular context, the authors of (Bodnar et al., 2022) proposed to learn a sheaf-structure on the graph. Before reviewing the results of this work, though, let us first introduce the concept of a cellular sheaf and some useful notation:

Definition 2.2. Given an undirected graph $G = (V, E)$, we will call $(G, \mathcal{F})$ a cellular sheaf when $\mathcal{F}$

- assigns, for each $v \in V$ and $e \in E$, the corresponding vector spaces $\mathcal{F}(v)$ and $\mathcal{F}(e)$, which are called *stalks*;
- defines a linear map $\mathcal{F}_{v \trianglelefteq e} : \mathcal{F}(v) \to \mathcal{F}(e)$ for each incident $v \trianglelefteq e$ node-edge pair, known as *restriction maps*.

Definition 2.3. We will call the direct sum of all the node vector spaces the space of *0-cochains*, $C^0(G; \mathcal{F}) = \bigoplus_{v \in V} \mathcal{F}(v)$. Analogously we can define the space of *1-cochains*, $C^1(G; \mathcal{F}) = \bigoplus_{e \in E} \mathcal{F}(e)$. x_u and x_e will be the projections unto the spaces $\mathcal{F}(u), \mathcal{F}(e)$.

Definition 2.4. We will call the space of global sections the subspace $H^0(G; \mathcal{F}) \subseteq C^0(G; \mathcal{F})$ that satisfies

$$H^0(G; \mathcal{F}) = \{x \in C^0(G; \mathcal{F}) | \mathcal{F}_{u \trianglelefteq e} x_u = \mathcal{F}_{v \trianglelefteq e} x_v, \ \forall e \in E\}.$$

From what we've seen so far we can define a new linear map in the following way:

Definition 2.5. We define the coboundary map $\delta : C^0(G; \mathcal{F}) \to C^1(G; \mathcal{F})$ where[1]

$$(\delta x)_e = \mathcal{F}_{u \trianglelefteq e} x_u - \mathcal{F}_{v \trianglelefteq e} x_v.$$

Now, with this extra structure, a new Laplacian may be defined in the following way:

Definition 2.6. We define the *sheaf laplacian* $\Delta_{\mathcal{F}} : C^0(G; \mathcal{F}) \to C^0(G; \mathcal{F})$ as $\Delta_{\mathcal{F}} = \delta^T \delta$. Given a node signal $X(t)$ on a graph $G = (V, E)$ with an associated sheaf $\mathcal{F}$, then its sheaf Laplacian applied to node u would be:

$$(\Delta_{\mathcal{F}} x)_u = \sum_{u, v \trianglelefteq e} \mathcal{F}^T_{u \trianglelefteq e} (\mathcal{F}_{u \trianglelefteq e} x_u - \mathcal{F}_{v \trianglelefteq e} x_v). \quad (3)$$

This Laplacian can be used like the graph Laplacian in (2) to define an ODE over $X \in C^0(G; \mathcal{F})$:

$$\frac{d}{dt} X = -\Delta_{\mathcal{F}} X, \quad (4)$$

resulting in a generalization of GCNs when discretised and used as a diffusion operator to define SNNs:

[1] Note that an arbitrary orientation of the edges must be defined. However, the particular choice is irrelevant for our purposes.

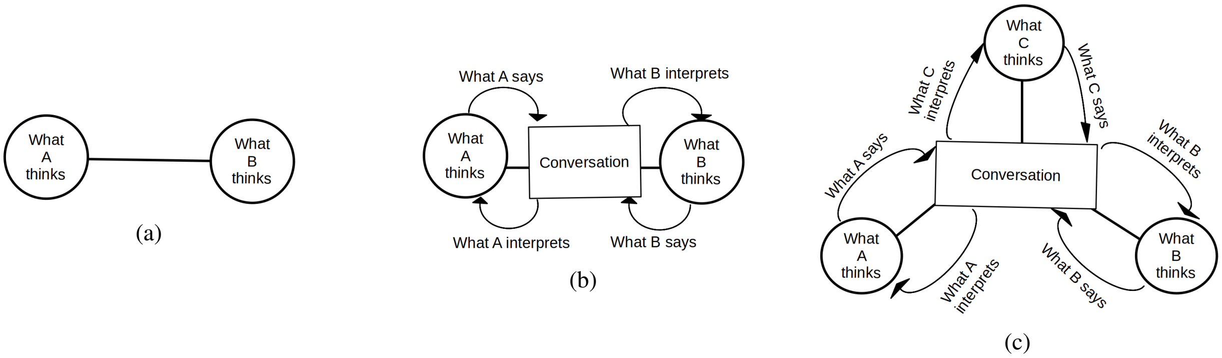

Figure 1. Visual representations of (a) a pair-wise interaction modelled by a graph, (b) a pair-wise interaction modelled by a sheaf over a graph, and (c) a higher-order interaction modelled by a sheaf over a hypergraph.

Definition 2.7. Let us denote $X(t) \in \mathbb{R}^{nd} \times \mathbb{R}^{f}$, where n is the number of nodes, d is the dimension of the stalks, f the number of feature channels, and t the layer of the neural network. The update equation that corresponds to the t layer of a SNN is:

$$X(t+1) = X(t) - \sigma(\Delta_{\mathcal{F}(t)}(I_n \otimes W_1(t))X(t)W_2(t)) \quad (5)$$

where the restriction maps of $\mathcal{F}(t)$ are computed like (Bodnar et al., 2022) –i.e. $\mathcal{F}_{u \trianglelefteq e}(t) = MLP(x_u||x_v)$ with $v \trianglelefteq e$–, and the matrices $W_1(t), W_2(t)$ consist of learnable parameters that act like convolutions.

The key point of this SNN is to learn a Laplacian with a richer set of equilibrium points that is no longer limited by a proposition like 2.1, thus avoiding oversmoothing. In this regard, we have the following result:

Proposition 2.8. *(Bodnar et al., 2022) Let G be a graph with an associated sheaf $\mathcal{F}$ and a node signal $X(t)$. If $X(t)$ satisfies (4), then $lim_{t\to\infty} X(t) \in H^0(G; \mathcal{F})$.*

In particular, this means that instead of converging to a configuration where $x_u = x_v$ for all nodes u, v, we now converge to a configuration where $\mathcal{F}_{u \trianglelefteq e} x_u = \mathcal{F}_{v \trianglelefteq e} x_v$. So, as intended, by considering the right restriction maps we should be able to have different classes be connected in our graph WITH different features. It is important to note, however, that unless the maps are orthogonal we have no guarantees that $H^0(G; \mathcal{F}) \neq \{0\}$ (Bodnar et al., 2022).

Remark 2.9. In order to learn these restriction maps, existing SNNs typically set $\mathcal{F}_{u \trianglelefteq e} = MLP(x_u||x_v)$, getting a sheaf universal approximation result (Bodnar et al., 2022). Nevertheless, this choice does not introduce any inductive bias to ensure that learnt maps are able to deal with heterophily and oversmoothing.

3. Opinion Dynamics Inspired Sheaf Neural Networks

Addressing the point raised in Remark 2.9, this section explores new ways of predicting the sheaf with a stronger inductive bias towards heterophilic data. To do so, we turn to opinion dynamics –the field that studies how opinions in a group of people spread– (Xia et al., 2011) and put our focus on some ODEs proposed in (Hansen & Ghrist, 2021) to analyze opinions' exchanges from a cellular sheaf perspective.

3.1. Motivation

Remark 3.1. If we encode a social network as a simple graph, the edges can only represent if two nodes interact or if they don't interact.

With this remark, it is straightforward to see that a naive graph-based modeling of opinion dynamics would prevent us from capturing the different types of interactions between different types of opinions. In other words, using a graph to spread opinions represents a myopic way of handling the complexity of human communication. Instead of blindly exchanging their private opinions (Fig. 1a), humans engage in conversations, where they have the freedom to express different nuances of their beliefs and draw their own conclusions based on how the conversation goes (Fig. 1b). Precisely, cellular sheaves offer us the necessary tools to formalise this, providing with an intuitive interpretation of the sheaf (Hansen & Ghrist, 2021): x_u now represents the node's private opinion, while $\mathcal{F}_{u \trianglelefteq e} x_u$ is the public opinion it decides to share with node $v \trianglelefteq e$. It is worth noting that the same rationale can be applied to hypergraphs, having more than 2 people participate in a conversation (Fig. 1c).

Bearing this in mind, we observe that Graph Diffusion (2) is a process that makes the node's true private opinions agree, while in Sheaf Diffusion (4) the agreement is achieved at the level of node's public opinions. This means that current variants of SNNs assume *(i)* a fixed communication space, and *(ii)* a diffusion process that modifies the private opinion to reach a consensus in that space. However, a more natural approach would be to allow the communication channel to evolve simultaneously, which is also important to guarantee the convergence to non-trivial agreements. As we know from Proposition 2.8, Eq. (4) must converge to a point satisfying a system of linear equations, but in general, this system might not have non-zero solutions. In this regard, we recall that (Bodnar et al., 2022) only proved the existence of non-trivial solutions when considering orthogonal restriction maps.

3.2. Learning to Lie

In our aim to model more intricate interactions, we turn to the ODE introduced and studied in (Hansen & Ghrist, 2021) as *Learning to Lie*. With this diffusion, we may overcome the aforementioned limitations by making the restriction maps evolve to fit the opinions instead of the other way around:

$$\frac{d}{dt}\mathcal{F}_{u\trianglelefteq e} = -(\mathcal{F}_{u\trianglelefteq e}x_u - \mathcal{F}_{v\trianglelefteq e}x_v)x_u^T. \tag{6}$$

Note that this ODE can be seen as a "dual" version of sheaf diffusion where we use the features x_u as maps by transposing them to diffuse the restriction maps; we can slightly rewrite Eq. (6) as

$$\frac{d}{dt}\mathcal{F}^T_{v\trianglelefteq e} = -x_u(x_u^T\mathcal{F}^T_{u\trianglelefteq e} - x_v^T\mathcal{F}^T_{v\trianglelefteq e}) \tag{7}$$

for a more explicit analogy (please see Figure 2 for a visual representation). In fact, given a sheaf Laplacian defined with the X signal, and denoting by $\mathcal{F}^*$ a matrix with all the restriction maps transposed, we can formulate this Dual Diffusion Process using the standard notation

$$\frac{d}{dt}\mathcal{F}^* = -\Delta_X\mathcal{F}^*. \tag{8}$$

As proven by (Hansen & Ghrist, 2021), in the limit this diffusion process achieves $\mathcal{F}_{u\trianglelefteq e}x_u = \mathcal{F}_{v\trianglelefteq e}x_v$ by adjusting the restriction maps to the signal. However, this might be still limiting for modelling complex interactions, as one node's private opinions cannot change at all.

3.3. Joint Opinion-Expression Diffusion

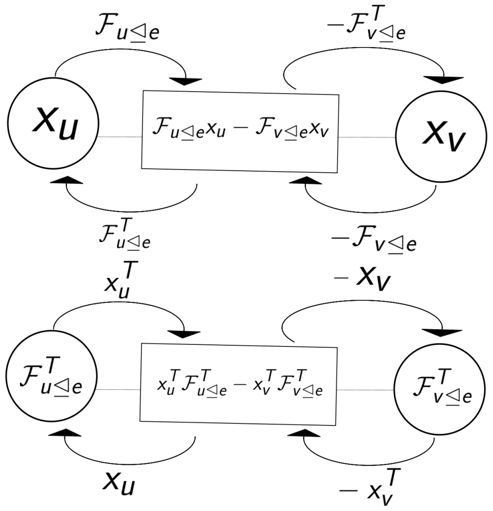
$\mathcal{F}_{u\trianglelefteq e}$
$-\mathcal{F}^T_{v\trianglelefteq e}$
x_u
$\mathcal{F}_{u\trianglelefteq e}x_u - \mathcal{F}_{v\trianglelefteq e}x_v$
x_v
$\mathcal{F}^T_{u\trianglelefteq e}$
$-\mathcal{F}_{v\trianglelefteq e}$
x_u^T
$-x_v$
$\mathcal{F}^T_{u\trianglelefteq e}$
$x_u^T\mathcal{F}^T_{u\trianglelefteq e} - x_v^T\mathcal{F}^T_{v\trianglelefteq e}$
$\mathcal{F}^T_{v\trianglelefteq e}$
x_u
$-x_v^T$

Figure 2. Top: a diagram representing Regular Sheaf Diffusion. Bottom: a representation of the *Learning to Lie* ODE diffusion.

To overcome the unrealistic rigidity of opinions of the previous model, the work of (Hansen & Ghrist, 2021) proposes a novel *joint opinion-expression diffusion* setting that puts together both the *Learning to Lie* and the Regular Sheaf Diffusion processes. From a theoretical perspective, this results in the following system of non-linear ODEs:

$$\begin{cases} \dfrac{d}{dt}\mathcal{F}^*(t) = -\beta\Delta_{X(t)}\mathcal{F}^*(t) \\ \dfrac{d}{dt}X(t) = -\alpha\Delta_{\mathcal{F}(t)}X(t) \end{cases} \tag{9}$$

where $\alpha, \beta \in \mathbb{R}$ are coefficients that control the diffusion strength of each ODE, prioritising the diffusion either over the restriction maps or over the features.

The following lemma can help us understand the inductive bias brought by Equation (9):

Lemma 3.2. *(Hansen & Ghrist, 2021) Let's consider $\Psi(X(t),\mathcal{F}(t)) = X(t)^T\delta(t)^T\delta(t)X(t)$ where $X(t),\mathcal{F}(t)$ is a solution of (9). Then $\Psi(X(t),\mathcal{F}(t)) \geq 0$ and $\frac{d\Psi(X(t),\mathcal{F}(t))}{dt} \leq 0$ with zero attained in both if and only if $\mathcal{F}_{u\trianglelefteq e}x_u = \mathcal{F}_{v\trianglelefteq e}x_v$ for all $u, v \trianglelefteq e$.*

This means that we may see this process as gradient descent on the sheaf Dirichlet energy (Bodnar et al., 2022), which means we are evolving towards a sheaf equilibrium leveraging both the node's features and the restriction maps. Using LaSalle's invariance principle (Hansen & Ghrist, 2021) we can prove that we actually converge to an equilibrium point with $\mathcal{F}_{u\trianglelefteq e}x_u = \mathcal{F}_{v\trianglelefteq e}x_v$ for all $u, v \trianglelefteq e$.

Even though we have ensured convergence to some point $(x_\infty, \delta_\infty)$, it is still necessary to check when $x_\infty \neq 0$. This is due to the fact that, if the trajectories of (9) all converged to zero, this system would also oversmooth the signal $X(t)$. To do this, we resort to the following result:

Theorem 3.3. *(Hansen & Ghrist, 2021) If the initial conditions of the system (9) are δ_0, x_0 and one of the diagonal blocks of $\alpha\delta_0^T\delta_0 - \beta x_0 x_0^T$ fails to be semidefinite, the trajectory of (9) converges to a point $(x_\infty, \delta_\infty)$ with $x_\infty \neq 0$.*

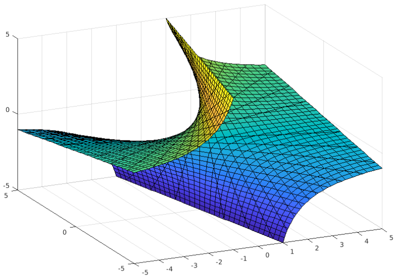

Figure 3. A visualization of the solutions of $zx = ty$, with $t = 1$ because the ones with $t = 0$ are trivially $t = 0$ and $y = 0$. This is, in fact, an affine projection of a projective variety. This corresponds with the set of equilibrium points of Eqs. (9) in the case of 1-dimensional stalks and a graph with only 2 adjacent nodes, so z, t would be the restriction maps while x, y would be the node's features. Sheaf diffusion only evolves x, y, so it may miss close points of equilibrium in the z axis.

This shows a clear advantage concerning regular sheaf diffusion, as we can easily check if the system will converge to a non-zero solution. In addition to this, we also have the following result:

Corollary 3.4. *(Hansen & Ghrist, 2021) If (x_0, δ_0) are some initial conditions, then there exists a k such that the trajectory of (9) with initial conditions (kx_0, δ_0) converges to a point $(x_\infty, \delta_\infty)$ with $x_\infty \neq 0$.*

Therefore, we can always guarantee the existence of non-zero solutions by just tuning one parameter. Note, however, that this is not the case for regular sheaf diffusion, where orthogonal restriction maps are necessary to get a similar guarantee. Another advantage is that the diffusion leverages both restriction maps and the node's features to reach a point of equilibrium. This is particularly useful in cases of heterophily, as this process may shut off communication among nodes with different opinions without the need of a learnable function.

Example. Figure 3 let us visualize the usefulness of diffusing both restriction maps and node's features by depicting the points of equilibrium of Eqs. 9 for a particular case example. On the one hand, the trajectory used in the Joint Diffusion has the freedom to move in any direction to reach a point of equilibrium. On the other hand, the trajectory used in classical Sheaf Diffusion (Equation 4) can only move in a horizontal plane, which means we may miss close points of equilibrium in the vertical axis.

3.4. Joint Diffusion Sheaf Neural Networks

With the previous dynamical system (9) in mind, we propose a new SNN model as the natural conclusion of this section:

Definition 3.5. With the setting described in the Definition 2.7 of SNNs, we define the t layer of a Joint diffusion Sheaf Neural Network (JdSNN) through the following equations:

$$\begin{cases} X(t+1) = X(t) - \\ \quad \sigma((\mathbf{I}_{nd} - \alpha\Delta_{\mathcal{F}}(t))(I_n \otimes W_1(t))XW_2(t)) \\ \mathcal{F}^*(t+1) = \mathcal{F}^*(t) - \\ \quad \sigma((\mathbf{I}_{2md} - \beta\Delta_{X(t)})(I_n \otimes W_1^*(t))\mathcal{F}^*W_2^*(t)) \end{cases} \tag{10}$$

where α, β are weights to give more importance to one of the diffusions depending on the problem.[2]

This new variant has an extra inductive bias towards learning on heterophilic data, as it may prevent oversmoothing of the signal x_u by "oversmoothing" the restriction maps $\mathcal{F}_{u \trianglelefteq e}$ instead. For instance, let us consider the case of two adjacent nodes u, v of different classes and significantly different features; whereas the second equation of (10) would make the restriction maps evolve to avoid mixing the features of u and v too much, it is unclear if –and how– the features from u, v would mix when using $\mathcal{F}_{u \trianglelefteq e} = MLP(x_u || x_v)$. Moreover, all of this is achieved with $2d^2$ parameters per diffusion layer coming from W_1^*, W_2^*, instead of the at least $2d^2c$ ones of usual SNNs (with d being the stalk dimension and c the number of original features and $2d^2c$ is the size of the weights matrix used to predict the restriction maps). Here it is important to note that the number of parameters in JdSNNs no longer scale with c, making this model suitable for contexts where extensive parameterizations may not be possible. Examples of this are small datasets or some scenarios of federated learning where the size of the features makes linear layers impossible to apply (Gabrielli et al., 2023).

However, this new model also presents some setbacks, as it is not possible to impose some constraints such as orthogonal or diagonal restriction maps, which have proved to be useful when training SNNs (Bodnar et al., 2022; Duta et al., 2024). We also lose the universal sheaf approximation result presented by (Bodnar et al., 2022), but this actually represents a trade-off between adding an inductive bias for heterophilic data instead of using the most general model.

3.5. Rotation Invariant Sheaf Neural Networks

As we mentioned in the previous section, JdSNNs gain an inductive bias at the cost of using more complex dynamics. In hopes of simplifying the dynamics while keeping the inductive bias, we introduce another alternative that explores a new way of learning the restriction maps within the usual SNN formulation of Definition 2.7.

In our proposal, instead of using $\mathcal{F}_{u \trianglelefteq e}(t+1) =$

[2] For example, if we know the dataset is heterophillic we may want to set $\beta > \alpha$ to prioritize the diffusion of the restriction maps, potentially preventing oversmoothing.

$MLP(x_u(t)||x_v(t))$, we consider the vector $x_e(t) = \mathcal{F}_{u\trianglelefteq e}(t)x_u(t) - \mathcal{F}_{v\trianglelefteq e}(t)x_v(t)$ –that represents the conversation in the edge space–, and set $\mathcal{F}_{u\trianglelefteq e}(t+1) = MLP(x_e(t)x_u(t)^T)$. With this approach, the number of parameters we use is d^4, which is more than JdSNNs but still does not scale with the total number of features c. In this model, instead of learning the restriction maps from all the features, we are assuming that they only depend on the relationship between the node's private opinion x_u and the public conversation x_e:

$$x_e x_u^T = \begin{pmatrix} < x_{e1}, x_{u1} > & \dots & < x_{e1}, x_{ud} > \\ \dots & \dots & \dots \\ < x_{ed}, x_{u1} > & \dots & < x_{ed}, x_{ud} > \end{pmatrix},$$

where x_{ei}, x_{uj} are the i-th and j-th row vectors of x_e, x_u. With this in mind, the following proposition can be easily proven:

Proposition 3.6. *The restriction maps of a SNN learnt using $\mathcal{F}_{u\trianglelefteq e}(t+1) = MLP((\mathcal{F}_{u\trianglelefteq e}(t)x_u - \mathcal{F}_{v\trianglelefteq e}(t)x_v)x_u^T)$ are feature-wise rotation invariant w.r.t X. That is, given Q an orthogonal matrix, then the restriction maps computed for X and QX will be equal.*

Proof in Appendix B.

Due to this result, we call this SNN variant **Rotation invariant Sheaf Neural Networks** (RiSNN). Please note that only the restriction maps are rotation invariant. However, it is possible to obtain a GNN with rotation equivariance by setting $\sigma = id$ and $W_2 = Id$. A more intuitive way of thinking about this model would be performing an attention-like mechanism on multiple cosine-similarities.

4. Benchmark Evaluation

In this section we test the introduced opinion-inspired SNN models under the benchmark defined in (Bodnar et al., 2022).[3]

Ablation Study For this evaluation, we consider a total of 4 different versions of our models: RiSNN and JdSNN are the ones we have already discussed. RiSNN with no time dependency (RiSNN-NoT) is a reinterpretation of the regular RiSNN by setting $x_e(t) = x_u(t) - x_v(t)$; the goal of this modification is to stabilize the gradient. JdSNN with no learnable weights (JdSNN-NoW) is a simplification of our full version where the restriction maps are computed without any parameters –i.e. obtained by removing the non-linearity, considering $W_1^*, W_2^* = Id$, and setting $\mathcal{F}_{u\trianglelefteq e}(0) = Id$ for all $u \in V, e \in E$. These 2 simpler variants are shown to achieve equivalent results in some cases.

Table 1 shows the results of our models against the best version of the architectures proposed in (Bodnar et al., 2022; Suk et al., 2022; Barbero et al., 2022; Zaghen, 2024). As we can see in Table 1, we mostly get statistically equivalent results to those of other methods while inferring the sheaf structure with significantly less parameters. Consequently, this shows that the inductive bias we have traded for learnable parameters is useful and does not negatively affect performance when applied to heterophilic datasets.

We note, though, that our models –especially JdSNN– do not perform greatly in two heterophilic datasets (Squirrel and Chameleon). However, they also happen to be the biggest ones, so overall this still aligns with what we have discussed so far: on the one hand, high amounts of data allow for a complex $MLP(x_u||x_v)$ that approximates $\mathcal{F}_{u\trianglelefteq e}$ accurately. On the other hand, when there is not enough data for a good approximation to be learned, the inductive bias we provide is capable of replacing these extensive parametrizations.

However, previous work (Platonov et al., 2023) highlights the limitations of these benchmarks for properly evaluating the model's abilities to capture heterophilic interactions. In fact, there is a pressing need in the graph community to find and/or design new datasets that can help to properly assess current model architectures, providing with meaningful insights about their actual capabilities in different settings.

5. Synthetic Evaluation

Arguably, the need of meaningful datasets to test SNNs' capabilities is even more crucial than in the broader graph domain given its recent formalization and its particular geometric intricacies. However, apart from evaluating them on graph benchmarks, previous sheaf-based works only assess their proposed models considering either denoising tasks (Zaghen, 2024; Bodnar et al., 2022), or heterophilic scenarios with unrealistic single-feature $d = 1$ setups (Hansen & Gebhart, 2020; Bodnar et al., 2022).[4]

Given this context, this section presents a novel method to produce synthetic datasets, which we then leverage to evaluate different aspects and capabilities of SNNs –and those of our introduced variants. The proposed pipeline is divided into two main processes –feature and edge generation–, and is designed to address the following research questions:

1. How does node features' noise affect performance?
2. What is the impact of the ratio of intra-class edges vs inter-class edges?
3. How does the amount of data available correlate with performance for different SNN variants?

[3] Except the film dataset, as equivalent results to (Bodnar et al., 2022) can be obtained ignoring the graph structure.

[4] By construction our model relies on $d > 1$, the most usual case in real-world scenarios.

Table 1. Accuracy obtained by the considered models on the node classification benchmark originally used by (Bodnar et al., 2022), sorted by homophily degree. Top three models are coloured by First, Second, Third. Our models are: RiSNN-NoT, RiSNN, JdSNN, and JdSNN-NoW.

Dataset	**Texas**	**Wisconsin**	**Squirrel**	**Chameleon**	**Cornell**	**Citeseer**	**Pubmed**	**Cora**
#Nodes	**183**	**251**	**5201**	**2277**	**183**	**3327**	**18717**	**2708**
#Edges	**295**	**466**	**198493**	**31421**	**280**	**4676**	**44327**	**5278**
#Homophily	**0.11**	**0.21**	**0.22**	**0.23**	**0.3**	**0.74**	**0.80**	**0.81**
RiSNN-NoT	87.89±4.28	88.04±2.39	51.24±1.71	66.58±1.81	82.97±6.17	75.07±1.56	87.91±0.55	85.86±1.31
RiSNN	86.84±3.72	87.84±2.60	53.30±3.30	65.15±2.40	85.95±6.14	76.23±1.81	88.00±0.42	85.27±1.11
JdSNN-NoW	87.30±4.53	88.43±2.83	51.28±1.80	66.45±3.46	84.59±6.95	75.93±1.41	88.09±0.49	84.39±1.47
JdSNN	87.37±5.10	89.22±3.42	49.89±1.71	66.40±2.33	85.41±4.55	73.27±1.86	88.19±0.55	85.43±1.73
Conn-NSD	86.16±2.24	88.73±4.47	45.19±1.57	65.21±2.04	85.95±7.72	75.61±1.93	89.28±0.38	83.74±2.19
Best-NSD	85.95±5.51	89.41±4.74	56.34±1.32	68.68±1.73	86.49±7.35	77.14±1.85	89.49±0.40	87.30±1.15
Best-NSP	87.03±5.51	89.02±3.84	50.11±2.03	62.85±1.98	76.49±5.28	76.85±1.48	89.42±0.33	87.38±1.14
Best-BC-NLSD	87.57±5.43	89.41±2.66	54.62±2.82	66.54±1.05	87.30±6.74	76.03±1.56	89.39±0.43	87.00±1.23
Best-MLP-NLSD	86.22±3.91	89.02±3.19	51.96±2.65	65.37±2.73	87.03±4.49	76.11±1.81	89.60±0.29	86.38±1.20

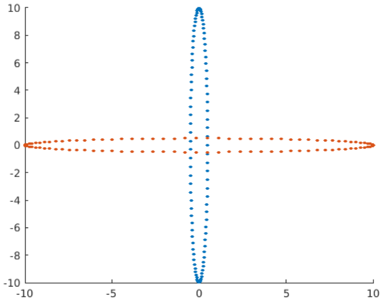

Figure 4. Graphic plot of two classes, red and blue, generated by sampling a 2-dimensional ellipsoid's surface. In this representation, it's easy to see how they're not linearly separable but distinct, and that the expected value for both of them is zero.

5.1. Synthetic Features Generation

As we previously mentioned, we aim to measure the impact of noise in our models as well as the impact of heterophily and the amount of data available. This means that generating classes by adding noise to different expected values is not a viable option, as we would only be able to answer the first question. Consequently, we turn to manifold sampling to generate complex patterns for the different classes.

In particular, we choose the surface of different n-dimensional ellipsoids for each class, all sharing one central point as can be seen in Fig. 4. The reason behind this choice relies on their symmetrical properties, producing classes that are not linearly separable while at the same time, they all share the same expected value. This second condition is key as it ensures that non-trivial aggregation will be necessary.

5.2. Synthetic Edge Generation

Regarding the edge generation, we use a variation of the Watts-Strogatz model (Watts & Strogatz, 1998) that also introduces inter-class correlations. But before describing the pipeline, let us first introduce the involved notation: N denotes the number of nodes, K the desired degree of each node (should be an even number), $p \in [0, 1]$ a probability-based parameter, n_c the number of node classes, and $R^c \in \mathcal{M}_{n_c \times n_c}$ the matrix such that $R^c_{ij} = Pr(\{node_1, node_2\} \in E | node_1 \in class_i, node_2 \in class_j)$. We also consider a het coefficient representing the level of heterophily in the graph, which is defined as the probability that a node is connected to another class. This in turn allows us to write the matrix of inter-class correlations as:

$$R^c = \left(1 - \frac{n_c + 1}{n_c - 1} het\right) Id + \frac{het}{n_c - 1} * \mathbf{1}_{\boldsymbol{n_c}} \mathbf{1}_{\boldsymbol{n_c}}{}^T,$$

with $1 - het$ in the diagonal entries, and $\frac{het}{n_c - 1}$ in the rest.

Bearing this notation in mind, the steps to generate the final graph connectivity of the data are as follows:

1. Label the nodes from 0 to $N - 1$ and assign each node a class uniformly at random.

2. Construct a regular ring lattice. That is $(i, j) \in E \Leftrightarrow 0 < |i - j| mod(N - 1 - K/2) \leq K/2$.

3. For each node $i = 0, ..., N - 1$ we consider its $K/2$ rightmost edges and with probability p rewire it to another node. To choose the node we first sample a class from the distribution of the i-th row of R^c, and then we choose uniformly at random a node of said class that is not connected to node i.

5.3. Experimental setup

With the focus on assessing how the restriction maps are learned, we set $W_1 = Id, W_2 = Id$, and $\sigma = Id$ in the setting introduced in Definition 2.7, as done in the work (Bodnar et al., 2022).

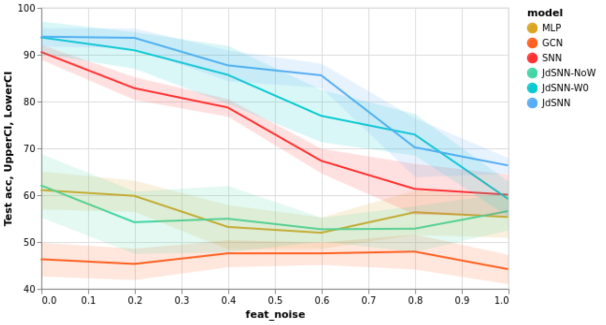

Figure 5. JdSNNs variants' accuracy results with a 95% confidence interval when increasing the percentage of feature Gaussian noise in the data. We may observe how our methods are more robust to noise than regular SNNs. We can also observe how GCNs and MLPs underfit the data.

Analogously to our previous Benchmark Evaluation, in these experiments we also test the full JdSNN and RiSNN models, as well as their respective simpler variants JdSNNnoW and RiSNN-NoT (please see Section 4). Additionally, here we also consider another joint diffusion variation, JdSNN-W0, which initializes the restriction maps with a learnable function –$\mathcal{F}^*(0) = MLP(x_u||x_v)$– but does the Dual Diffusion without any parameters –$\mathcal{F}^*(t+1) = \mathcal{F}^*(t) - \Delta_{X(t)}\mathcal{F}^*(t)$.

We compare these models against the following baselines: VanillaSheaf, that is $\mathcal{F}_{u \trianglelefteq e} = Id$ for all $u \in V, e \in E$; a regular MLP with one layer and id as activation function; and a SNN with diagonal restriction mappings.

Remark 5.1. For the Joint Diffusion based models, we wish to test the separation power of each of the diffusion processes. To this end, in the experiments involving JdSNNs, only 1 feature channel and 3-dimensional stalks are considered –i.e. the hidden features have shape 3×1. Conversely, for RiSNNs evaluations we use 3-dimensional stalks and 5 feature channels to fully utilize the scalar products in $x_e x_u^T$.

5.4. Results

This section describes the main takeaways of our extensive evaluation of the generated synthetic datasets, with the focus on answering the fundamental research questions raised at the beginning of this section. Apart from Figure 5 above, we also include in Appendix C the plots associated with the rest of performed experiments (Figures 6, 7 and 8).

Joint diffusion without any parameters Overall, the JdSNN version without parameters, JdSNN-noW, has an advantage over GCNs (Figures 6, 7 and 8). However, in our task one layer oversmooths the representation, so setting $\mathcal{F}(0) = Id$ is a bottleneck to the model's performance.

The inductive bias towards heterophily Our proposed models, and specially the Joint Diffusion variants, show a clear bias towards heterophilic data as seen in Figure 7. When increasing the number of inter-class edges the difficulty of the problem also increases, as more different types of interactions must be modelled. In this aspect, the performance of standard SNNs decreases significantly more than JdSNNs.

Joint Diffusion in cases of noisy data JdSNNs are more robust to noise in the cases of small dimensionality of the data. As one may observe in Fig. 5, the version using only a learnable initialization and the full model are significantly better than a regular SNN when adding noise, up to the point where the features are 70% noise.

Rotation invariant SNNs in cases of noisy data RiSNNs in general are more sensitive to noisy features than SNNs as seen in Figure 6. Nonetheless, the RiSNN-NoTime version is slightly better than the original one. In cases of non-noisy data, the original formulation of RiSNNs tends to have a slight advantage over the version that doesn't use the restriction maps of the previous layer.

Amount of available data and the performance of SNNs SNNs' performance is greatly increased in problems with a large number of edges as seen in Figure 8. This allows the MLP which computes the restriction maps to get a better approximation. The inductive bias introduce by our variants, JdSNN and RiSNN, restricts the possible relationships between nodes, so they have an advantage when the sheaf structure cannot be accurately computed 8.

6. Conclusions

This work proposes and evaluates new ways of inferring the sheaf structure of a graph taking inspiration from complex opinion dynamics processes. Through these, we show how using opinion dynamics interpretations of sheaves may lead to a more intuitive understanding of SNNs and how they operate. When benchmarked, our new proposed models present a clear inductive bias towards heterophily, and they are able to achieve equivalent performance to regular SNNs while using fewer parameters to infer the sheaf structure. In particular, this reduction of the number of parameters opens the door to applying sheaf-based methods in domains where extensive parameterizations cannot be used. Furthermore, the geometrical properties of the introduced RiSNN model may enable the application of sheaves to geometrical deep learning problems.

Last but not least, this paper also proposes a novel way of evaluating SNNs *(i)* using the surface of n-dimensional ellipsoids to generate features, and *(ii)* generating graph connectivities with different degrees of heterophily. From extensive experimentation on generated datasets, we observe that SNNs benefit from having graphs with lots of edges, while our introduced variants take the lead in graphs with low connectivity.

Overall, we reckon that our introduced models and synthetic data pipelines can pave the way for interesting future research in this exciting area, fostering a deeper understanding of the inner workings and capabilities of SNNs.[5]

Acknowledgments

The authors would like to thank the Private Foundation Pere Mir-Puig and CFIS for the funding provided to Ferran Hernandez to facilitate his stay at the University of Cambridge.

References

Barbero, F., Bodnar, C., de Ocáriz Borde, H. S., Bronstein, M., Veličković, P., and Liò, P. Sheaf neural networks with connection laplacians. In *Topological, Algebraic and Geometric Learning Workshops 2022*, pp. 28–36. PMLR, 2022.

Bodnar, C., Di Giovanni, F., Chamberlain, B., Lio, P., and Bronstein, M. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns. *Advances in Neural Information Processing Systems*, 35: 18527–18541, 2022.

Chen, D., Lin, Y., Li, W., Li, P., Zhou, J., and Sun, X. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 3438–3445, 2020.

Curry, J. M. *Sheaves, cosheaves and applications*. University of Pennsylvania, 2014.

Duta, I., Cassarà, G., Silvestri, F., and Liò, P. Sheaf hypergraph networks. *Advances in Neural Information Processing Systems*, 36, 2024.

Duval, A., Mathis, S. V., Joshi, C. K., Schmidt, V., Miret, S., Malliaros, F. D., Cohen, T., Liò, P., Bengio, Y., and Bronstein, M. A hitchhiker's guide to geometric gnns for 3d atomic systems. *arXiv preprint arXiv:2312.07511*, 2023.

Gabrielli, E., Pica, G., and Tolomei, G. A survey on decentralized federated learning. *arXiv preprint arXiv:2308.04604*, 2023.

Hansen, J. and Gebhart, T. Sheaf neural networks. *arXiv preprint arXiv:2012.06333*, 2020.

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

Hansen, J. and Ghrist, R. Opinion dynamics on discourse sheaves. *SIAM Journal on Applied Mathematics*, 81(5): 2033–2060, 2021.

Kipf, T. N. and Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*, 2017.

Luan, S., Hua, C., Lu, Q., Zhu, J., Zhao, M., Zhang, S., Chang, X.-W., and Precup, D. Revisiting heterophily for graph neural networks. *Advances in neural information processing systems*, 35:1362–1375, 2022.

Nt, H. and Maehara, T. Revisiting graph neural networks: All we have is low-pass filters. *arXiv preprint arXiv:1905.09550*, 2019.

Oono, K. and Suzuki, T. Graph neural networks exponentially lose expressive power for node classification. *arXiv preprint arXiv:1905.10947*, 2019.

Platonov, O., Kuznedelev, D., Babenko, A., and Prokhorenkova, L. Characterizing graph datasets for node classification: Homophily-heterophily dichotomy and beyond. *Advances in Neural Information Processing Systems*, 36, 2023.

Qiu, J., Tang, J., Ma, H., Dong, Y., Wang, K., and Tang, J. Deepinf: Social influence prediction with deep learning. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2110–2119, 2018.

Rusek, K., Suárez-Varela, J., Mestres, A., Barlet-Ros, P., and Cabellos-Aparicio, A. Unveiling the potential of graph neural networks for network modeling and optimization in sdn. In *Proceedings of the 2019 ACM Symposium on SDN Research*, pp. 140–151, 2019.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

Suk, J., Giusti, L., Hemo, T., Lopez, M., Barmpas, K., and Bodnar, C. Surfing on the neural sheaf. In *NeurIPS 2022 Workshop on Symmetry and Geometry in Neural Representations*, 2022.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph Attention Networks. In *ICLR*, 2018.

[5] A further discussion of limitations and future work can be found in Appendices D and E, respectively.

Watts, D. J. and Strogatz, S. H. Collective dynamics of ‘small-world’networks. *nature*, 393(6684):440–442, 1998.

Xia, H., Wang, H., and Xuan, Z. Opinion dynamics: A multidisciplinary review and perspective on future research. *International Journal of Knowledge and Systems Science (IJKSS)*, 2(4):72–91, 2011.

Yan, Y., Hashemi, M., Swersky, K., Yang, Y., and Koutra, D. Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks. In *2022 IEEE International Conference on Data Mining (ICDM)*, pp. 1287–1292. IEEE, 2022.

Zaghen, O. Nonlinear sheaf diffusion in graph neural networks. *arXiv preprint arXiv:2403.00337*, 2024.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. Graph neural networks: A review of methods and applications. *AI open*, 1:57–81, 2020.

Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L., and Koutra, D. Beyond homophily in graph neural networks: Current limitations and effective designs. *Advances in neural information processing systems*, 33:7793–7804, 2020.

A. Proof of Proposition 2.1

For this proof, we use that $lim_{t\to\infty}X(t)$ is the orthogonal projection of $X(t)$ to $ker(L)$. So, we will prove that $ker(L)$ only contains vectors constant in all connected components. Given $v \in ker(L)$, we compute Lv:

$$Lv = \begin{bmatrix} \sum\limits_{j|(1,j)\in E} (v_1 - v_j) \\ \sum\limits_{j|(2,j)\in E} (v_2 - v_j) \\ \vdots \\ \sum\limits_{j|(n,j)\in E} (v_n - v_j) \end{bmatrix} \tag{11}$$

As $v \in ker(L)$, $Lv = 0$, so $v^t Lv = 0$. Expanding $v^t Lv$ yields

$$v^t Lv = \sum_{i=1}^{n} \sum_{j|(i,j)\in E} v_i(v_i - v_j)$$

Now, if we select $i < j, (i,j) \in E$, the sum contains the terms $v_i(v_i - v_j)$ and $v_j(v_j - v_i)$ exactly once. So, by grouping terms we get $(v_i - v_j)^2$. This directly implies: $v^t Lv = \sum\limits_{i<j,(i,j)\in E} (v_i - v_j)^2$. This and this quantity is zero if and only if $v_i = v_j \forall \{i,j\} \in E$, which concludes our proof. ■

B. Proof of Proposition 3.6

Let's suppose $X' = XQ$ with Q an orthogonal matrix, we proceed by induction on the number of layers. Since $t = 0$ is not defined, we set it to a constant (Id) the rotation trivially does not affect this. Then if $\mathcal{F}'_{u\trianglelefteq e}(t) = \mathcal{F}_{u\trianglelefteq e}(t)$

$$\begin{aligned} \mathcal{F}'_{u\trianglelefteq e}(t+1) &= MLP((\mathcal{F}'_{u\trianglelefteq e}(t)x'_u - \mathcal{F}'_{v\trianglelefteq e}(t)x'_v)x'^T_u) \\ &= MLP((\mathcal{F}'_{u\trianglelefteq e}(t)x'_uQ - \mathcal{F}'_{v\trianglelefteq e}(t)x'_vQ)Q^T x'^T_u) \\ &= MLP((\mathcal{F}_{u\trianglelefteq e}(t)x_uQ - \mathcal{F}_{v\trianglelefteq e}(t)x_v)x^T_u) \\ &= \mathcal{F}_{u\trianglelefteq e}(t+1) \end{aligned}$$

Where we have used both our induction hypothesis and $QQ^T = Id$. ■

C. Synthetic Experiments Results

In this section, we can observe the results which answer the three questions presented in Section 5:

- Figure 6: How does noise impact the performance of SNNs and their variants?
- Figure 7: How does the ratio of intra-class edges vs inter-class edges impact the performance of SNNs and their variants?
- Figure 8: How does the amount of data available impact the performance of SNNs and their variants?

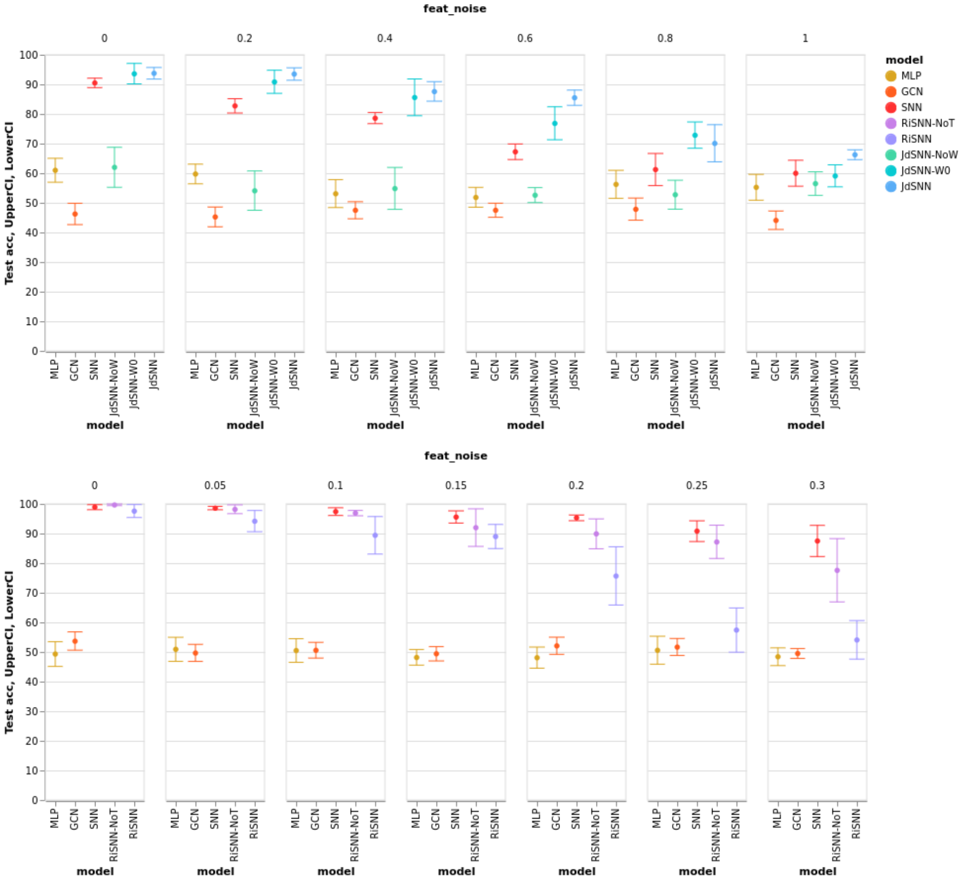

Figure 6. JdSNNs variants' results at the top and RiSNNs variants' results at the bottom when increasing feature Gaussian noise in the data

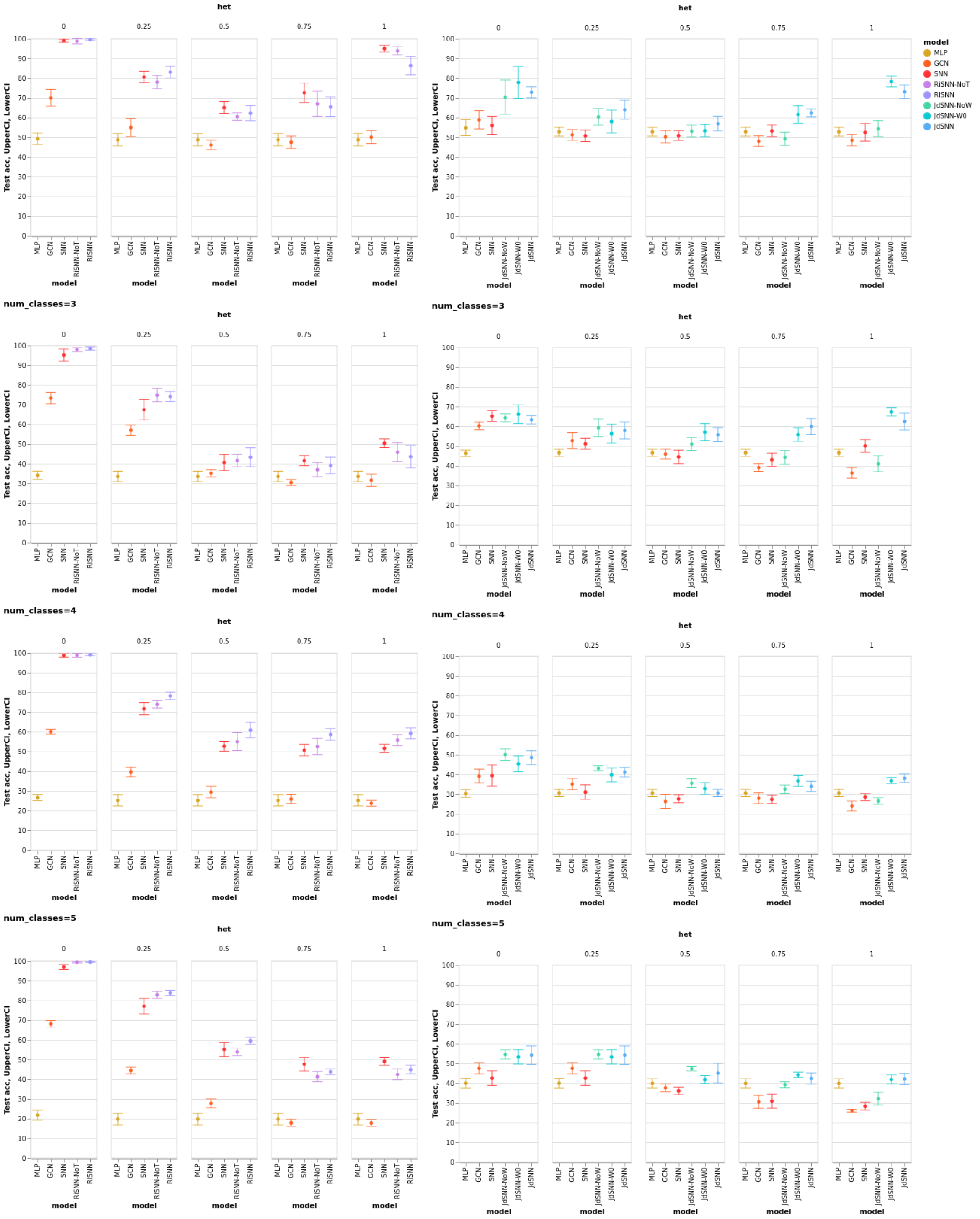

Figure 7. RiSNNs variants' results on the right and JdSNNs variants' results on the right when increasing the number of classes and the heterophily coefficient of the data

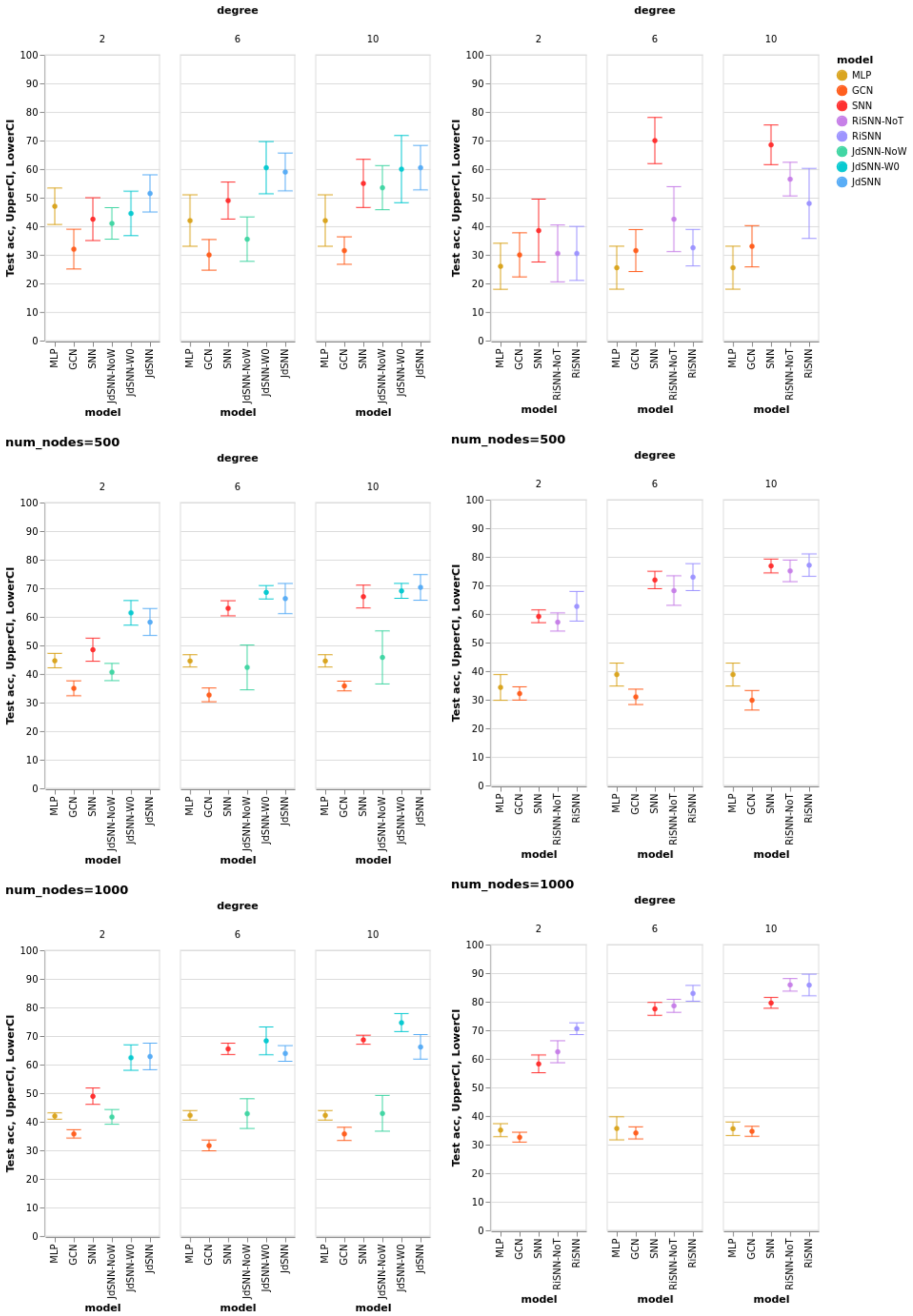

Figure 8. JdSNNs variants' results on the left and RiSNNs variants' results on the right when increasing the amount of nodes and edges of the data.

D. Limitations

The main limitation of the models proposed is that they are more complex to backpropagate; for example, using the restriction maps to define the normalization matrix D as in (Bodnar et al., 2022) can lead to exploding or undefined gradients. For this reason, it is necessary to detach D from the backpropagation algorithm.

Another limitation is that they are usually slower than SNNs. As was shown in (Bodnar et al., 2022), the time complexity of an SNN is $\mathcal{O}(nc^2 + mc)$ if the restriction maps are diagonal and $\mathcal{O}(n(c^2 + d^3) + m(c + d^3))$ if they're general, where n is the number of nodes, m the number of edges, c the total number of features, and d the stalk dimension. When it comes to our variants, JdSNNs and RiSNNs both have a time complexity of $\mathcal{O}(n(c^2 + d^3) + md(c + d^3))$. This implies that using these models with large d or m will lead to noticeably slower execution times. Nonetheless, we believe there is room for improvement in this aspect.

E. Future Work

SNNss are a relatively new GNN (Hansen & Gebhart, 2020; Bodnar et al., 2022), so there are many interesting research directions. In this section, we highlight a couple of them that we consider particularly interesting. The first one is related to federated learning while the latter with geometric deep learning. In federated learning, there have been approaches which are graph-based (Gabrielli et al., 2023). In this context, there are cases with millions of features, consequently, using regular MLPs on those features is not viable as a basic linear layer would have too many parameters. This implies that regular SNNs may not be used, but the new variants that have been proposed, RiSNN and JdSNN, have a number of parameters which only scales with the stalk dimension d which means sheaves may be used directly in federated learning. In some geometric problems on graphs, some extra properties regarding rotation are necessary (Duval et al., 2023). While this might prevent from implementing regular SNNs, RiSNNs might be more easily adaptable to be used in said problems.