Nonlinear Laplacians Improve Signed-Directed Graph Learning

Ali Parviz[1,2] * **Yuichi Yoshida**[3] †
[1]Mila - Quebec AI Institute, [2]New Jersey Institute of Technology, [3]National Institute of Informatics

Abstract

While signed-directed graphs have been studied using linear Laplacians in the design of graph neural networks, relatively little research has focused on developing non-linear Laplacian operators for such networks. We introduce a non-linear Laplacian operator specific to signed and directed networks (NLSD). This non-linear operator extends the concepts of the signed Laplacian for signed graphs and the Laplacian for directed graphs. The NLSD calculates node-specific potentials based on features More precisely, if the potential discrepancy is not aligned with the edge direction, we ignore it (and vice versa) leveraging message-passing techniques only across edges where potential discrepancies align with the edge's direction. Utilizing this novel operator, we propose an efficient spectral GNN framework (NLSD-GNN). We conducted comprehensive evaluations focusing on node classification and link prediction, examining scenarios involving signed, directional, or both types of information. Our findings reveal that this spectral GNN framework not only integrates signed and directional data effectively but also achieves superior performance across diverse datasets.

1 Introduction

Multiple real-life systems entail interactions that may be positive or negative and not always symmetrical He et al. [2024]. Such systems are best represented as undirected or directed graphs with *signed* edges, i.e. edges with an additional (+/-) sign attribute. In social networks, for example, users may share or oppose political views, trust or distrust recommendations from others, or harbor feelings of liking or disliking toward each other Kumar et al. [2016], Huang et al. [2021]. These diverse relationships can be effectively represented by signed graphs, where edges are assigned either positive or negative values to reflect affinity or conflict, and directional information could model the influence of one person on another Ju et al. [2020]. Signed-directed networks are also prevalent in contexts such as analyzing time-series data Bennett et al. [2022], identifying influential groups within social networks, and generating rankings through pairwise comparisons He et al. [2022a]. Moreover, these networks provide an intuitive framework for analyzing group conflicts and enhancing positive influence during opinion formation Zheng et al. [2015].

Not surprisingly, directed and signed graphs have attracted significant attention in graph representation learning. Graph Neural Networks (GNNs) in this context mimic standard GNNs and employ message passing, or equivalently, local aggregation of node features. The operators used by standard GNNs have well-understood spectral properties, which when seen globally relate to the combinatorial structure of the graph. However, the generalization of these operators to the signed-directed setting is far from obvious; indeed much of the existing work revolves around definitions of various *linear operators* for implementing message-passing. Table 3 summarizes the important aspects of these

*This work was done when the author was visiting the National Institute of Informatics (NII).

†Supported by JSPS KAKENHI Grant Number 20H05965

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: New Perspectives in Advancing Graph Machine Learning.

different Laplacians. Our work relies on a fragment of spectral theory that defines *nonlinear* Laplacian operators to encode directed graphs, and proves their expressivity in capturing -via Cheeger inequalities- combinatorial properties of the underlying directed graph Yoshida [2016]. More specifically, these Laplacians can be viewed as functions of a feature vector x, that combine two steps: (i) Edge dropouts that enforce message-passing only along directed edges (u, v) where $x_u \geq x_v$, (ii) The action of a symmetrized Laplacian on the subgraph induced by the edge dropouts. We begin by extending the definition of Yoshida [2016] and propose a nonlinear Laplacian that takes into account both edge direction and sign. While the extension is natural, it has not been discussed before.

The theoretical appeal and applied value of these nonlinear Laplacians motivate two questions: **Q1:** Can these nonlinear Laplacians be extended to directed graphs with additional edge signs? **Q2:** Can they be leveraged in the context of graph learning? We provide positive answers to both of these questions. We begin by extending the definition of Yoshida [2016] and propose a non-linear Laplacian that takes into account both edge direction and sign. While the extension is natural, it has not been discussed before. Using these nonlinear Laplacians in the context of GNNs, while simultaneously aiming for computational efficiency, is a trickier problem. In our method NLSD-GNN, we introduce a module that projects the node feature vectors $X \in \mathbb{R}^{n\times d}$ to scalar values. The module can be a fixed random projection (for efficiency), or a parametric trainable function, and it can be viewed as a function that assigns *potentials* to the nodes of the graph. Then, these potentials values are used to select a subgraph of the input graph, induced by the signed-directed edges. The subgraph is symmetrized into the signed-undirected counterpart and self-loops are added adequately. Then, the standard *linear* operator of the signed-undirected graph is used for the message propagation. Experimental validation shows that our methods improve upon the best known results on several benchmarks for transductive node classification and link prediction.

2 Preliminary

Let $G = (V, E, w)$ be a (weighted) directed graph, where V represents the set of vertices, E is the set of directed edges, and $w : E \to \mathbb{R}_{\geq 0}$ is the weight function. Although self-loops are allowed, we assume there are no parallel edges. For a vertex $v \in V$, the (weighted) *indegree* and *outdegree* of v is defined to be $d^-(v) := \sum_{e=(u,v)\in E} w(e)$ and $d^+(v) = \sum_{e=(v,u)\in E} w(e)$, respectively. The *total degree* of v is $d(v) := d^-(v) + d^+(v)$, which is the total weight of edges incident to v.

A signed graph $G = (V, E^+, E^-, w)$ consists of a vertex set V, two sets E^+, E^- of undirected edges, and a weight function $w : E^+ \cup E^- \to \mathbb{R}_{>0}$. Similarly, a *signed-directed graph* $G = (V, E^+, E^-, w)$ consists of a vertex set V, two sets $E^{\mp}, E^-$ of directed edges, and a weight function $w : E^+ \cup E^- \to \mathbb{R}_{\geq 0}$. We call edges in E^+ and E^- *positive edges* and *negative edges*, respectively. The *total degree* of v is the total weight of edges incident to v, including both positive and negative ones. We discussed the related in the Appendix A

3 Non Linear Signed-Directed Laplacian

We review the existing Laplacians for undirected, directed, signed and propose a novel non-linear Laplacian for signed-directed graphs.

Laplacian for undirected graphs. Let $G = (V, E, w)$ be a weighted undirected graph. The *degree matrix* of G is the diagonal matrix $D_G \in \mathbb{R}^{V\times V}$, where $(D_G)_{vv}$ is the (weighted) degree of $v \in V$, i.e., $\sum_{e\in E: v\in e} w(e)$. The *adjacency matrix* of G is a matrix $A_G \in \mathbb{R}^{V\times V}$, where $(A_G)_{uv} = (A_G)_{vu} = w(\{u, v\})$ if $\{u, v\} \in E$ and $(A_G)_{uv} = (A_G)_{vu} = 0$ otherwise. The *Laplacian* $L_G \in \mathbb{R}^{V\times V}$ of G is defined to be $D_G - A_G$. It is well known that its quadratic form satisfies $Q_G(x) := x^\top L_G x = \sum_{e=\{u,v\}\in E} w(e)(x_u - x_v)^2$, and hence for a set $S \subseteq V$, $Q(\mathbf{1}_S)$ gives the cut weight of the vertex set S, where $\mathbf{1}_S \in \mathbb{R}^V$ is the characteristic vector of S. The normalized Laplacian $\mathcal{L}_G$ is defined as $\mathcal{L}_G := D_G^{-1/2} L_G D_G^{-1/2}$.

Laplacian for directed graphs. Here, we define nonlinear Laplacian for directed graphs Yoshida [2016]:

Definition 1. Consider a directed graph $G = (V, E, w)$. The *Laplacian* $L_G : \mathbb{R}^V \to \mathbb{R}^V$ is defined to output a vector $L_G(x)$ for a vector $x \in \mathbb{R}^V$ via the following procedure:

1. Construct an undirected graph G_x on the vertex set V as follows: For each directed edge $e = (u, v) \in E$, include an undirected edge $\{u, v\}$ of weight $w(e)$ in G_x if $x_u \geq x_v$, and introduce self-loops $\{u, u\}$ and $\{v, v\}$ of weight $w(e)/2$ otherwise.
2. Let $L_{G_x} \in \mathbb{R}^{V \times V}$ be the Laplacian matrix for the undirected graph G_x.
3. The vector $L_G(x)$ is then derived as $L_{G_x} x$.

The resulting undirected graph G_x and the corresponding $L_G(x)$ are shown in Figure 1a. We note that the degree of any vertex $v \in V$ in G_x is exactly equal to the total degree of v in G.

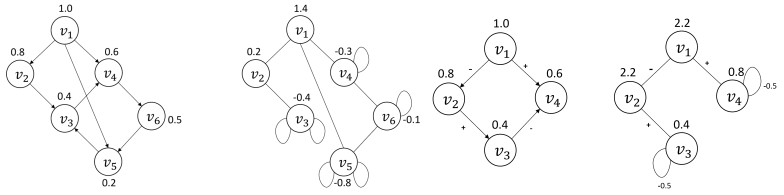
An important feature of the Laplacian for directed graphs is that its quadratic form satisfies $Q_G(x) := x^\top L_G x = \sum_{e=\{u,v\} \in E} w(e) \max\{x_u - x_v, 0\}^2$ Yoshida [2016]. Hence for a set $S \subseteq V$, $Q(\mathbf{1}_S)$ gives the cut weight of the vertex set S.

Laplacian for signed graphs. Let $G = (V, E^+, E^-, w)$ be a signed graph, and let $G^+ = (V, E^+, w)$ and $G^- = (V, E^-, w)$ be the undirected graphs consisting of positive and negative, respectively, edges. Then, let $L_G^+ \in \mathbb{R}^{V \times V}$ be the Laplacian of G^+, and $L_G^- \in \mathbb{R}^{V \times V}$ be the matrix defined as $L_G^- := D_{G^-} + A_{G^-} = 2D_{G^-} - L_{G^-}$. Then, the Laplacian $L_G \in \mathbb{R}^{V \times V}$ of G is $L_G = L_G^+ + L_G^-$ Hou et al. [2003]. We can observe that its quadratic form $Q_G(x) = x^\top L_G x$ can be written as

$$Q_G(x) = \sum_{e=\{u,v\} \in E^+} w(e)(x_u - x_v)^2 + \sum_{e=\{u,v\} \in E^-} w(e)(x_u + x_v)^2$$

Note that the first term is equal to the quadratic form of the Laplacian of G^+. The second term becomes small when the values for the endpoints of an edge have opposite signs. For $S \subseteq V$, $Q_G(\mathbf{1}_S - \mathbf{1}_{V \setminus S})$ is four times the total weight of positive edges cut by S plus the total weight of negative edges uncut by S.

3.1 Laplacian for Signed-Directed Graphs



(a) Directed graph (b) Signed-directed graph

Figure 1: The left figures show unsigned (1a) and signed (1b) directed graphs G and vectors x while the right figures show undirected graphs G_x and vectors $L_G(x)$.

We define nonlinear Laplacian for signed-directed graphs by naturally unifying Laplacian for directed graphs and that for signed graphs:

Definition 2. Consider a signed-directed graph $G = (V, E^+ \cup E^-, w)$. The *Laplacian* $L_G : \mathbb{R}^V \to \mathbb{R}^V$ is defined to output a vector $L_G(x)$ for a vector $x \in \mathbb{R}^V$ via the subsequent procedure:

1. Construct a signed-undirected graph G_x on the vertex set V as follows: For each edge $(u, v) \in E^+ \cup E^-$, include an undirected edge of the same sign $e = \{u, v\}$ in G_x of weight $w(e)$ if $x_u \geq x_v$, and introduce self-loops of the same sign $\{u, u\}$ and $\{v, v\}$ of weight $w(e)/2$ otherwise.
2. Let $L_{G_x} \in \mathbb{R}^{V \times V}$ be the Laplacian matrix for the signed-undirected graph G_x.
3. The vector $L_G(x)$ is then derived as $L_{G_x} x$.

Figure 1b shows an example of a signed-directed graph, a vector $x \in \mathbb{R}^V$, and the computation of the signed-undirected graph G_x and the corresponding $L_G(x)$.

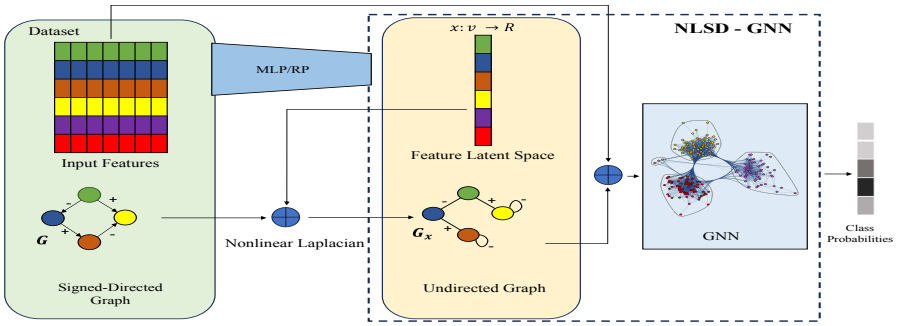


Figure 2: Overview of our workflow, NLSD-GNN, with a toy example. In the first block, we use an MLP or random projection to generate the feature latent space. In the next step, by using the nonlinear Laplacian transformation we calculate the counterpart signed-undirected graph by using the feature representation. The last block of the pipeline consists of passing the input features and the counterpart graph to spectral GNN layers to get the final prediction for the downstream task.

The *normalized Laplacian* $\mathcal{L}_G : \mathbb{R}^V \to \mathbb{R}^V$ of G is defined as $\mathcal{L}_G : x \mapsto D_G^{-1/2} L_G (D_G^{-1/2} x)$, where $D_G \in \mathbb{R}^{V \times V}$ is the diagonal matrix with total degree on the diagonal entries. For a signed-directed graph $G = (V, E^+, E^-, w)$, let $Q_G(x) = x^\top L_G(x)$ be its quadratic form. Then, we have

$$\begin{aligned} Q_G(x) = x^\top L^+_{G_x} x + x^\top L^-_{G_x} x \\ &= x^\top L_{G^+_x} x + x^\top (2D_{G^-_x} - L_{G^-_x}) x \\ &= x^\top L_{G^+_x} x + x^\top (2D_{G^-} - L_{G^-_x}) x \\ &= \sum_{e=(u,v)\in E^+} w(e) \max\{x_u - x_v, 0\}^2 \\ &\quad + \sum_{e=(u,v)\in E^-} w(e)(2x_u^2 + 2x_v^2 - \max\{x_u - x_v, 0\}^2). \end{aligned}$$

The second term can be written as

$$\sum_{e=(u,v)\in E^-} w(e) \cdot \begin{cases} (x_u + x_v)^2 & \text{if } x_u \geq x_v \\ 2x_u^2 + 2x_v^2 & \text{otherwise} \end{cases}$$

For $S \subseteq V$, $Q_G(\mathbf{1}_S - \mathbf{1}_{V \setminus S})$ is four times the total weight of positive edges cut by S plus the total weight of negative edges $(u, v) \in E^-$ such that $x_u \leq x_v$. Hence to minimize $Q_G(x)$, we want to embed vertices onto a one-dimensional line so that the tail vertices are to the left of the head vertices for positive edges and the tail vertices are to the right of the head vertices for negative edges.

When $\lambda \in \mathbb{R}$ and $v \in \mathbb{R}^V$ with $v \neq \mathbf{0}$ satisfy $L_G(v) = \lambda v$, we say that λ is an *eigenvalue* of L_G and that v is a corresponding eigenvector. The following theorems show that L_G and $\mathcal{L}_G$ satisfy properties analogous to the traditional graph Laplacians. The proofs are in Appendix B.

Theorem 1. *For any signed directed graph G, The operators L_G and $\mathcal{L}_G$ are positive semidefinite, i.e., all the eigenvalues are nonnegative.*

Theorem 2. *For any signed directed graph G, all the eigenvalues of $\mathcal{L}_G$ are in the range* $[0, 2]$.

4 NLSD-GNN architecture

In this section, we show how to use a normalized or unnormalized nonlinear Laplacian introduced in Definition 2, to define convolution on a signed-directed.

4.1 Spectral Convolution on Graphs.

Signal on graphs can be filtered using the eigendecomposition of graph Laplacian: $\mathcal{L} = U \Lambda U^T$, where U contains all of the eigenvectors of $\mathcal{L}$ and $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)$ is a diagonal matrix

Algorithm 1 Algorithm for NLSD-GNN-Fast

Input: Signed-directed graph $G = (V, E^+ \cup E^-, w)$, features $H \in \mathbb{R}^{N \times d}$, labeled vertices V_L
Output: Labels for all nodes in $V - V_L$

Apply random projection $x = H \cdot r$ where $r \in \mathbb{R}^{d \times 1}$
Graph Transformation:
Construct signed-undirected graph G_x on vertex set V
for each edge (u, v) in $E^+ \cup E^-$ **do**
  if $x_u \geq x_v$ **then**
    Include edge $e = \{u, v\}$ in G_x with weight $w(e)$
  else
    Introduce self-loops of the same sign $\{u, u\}$ and $\{v, v\}$ of weight $w(e)/2$
  end if
end for
Set $\tilde{A}_{G_x}$ as the adjacency matrix of G_x
for each epoch τ **do**
  for layer $\ell = 1$ to 2 **do**
    Compute $Z = \text{softmax}(\tilde{A}_{G_x}\text{ReLU}(\tilde{A}_{G_x} H\Theta^{(1)})\Theta^{(2)})$
    Update parameters Θ_τ to minimize cross-entropy loss
  end for
  Label nodes in $V - V_L$ using Z
end for

consisting of the corresponding eigenvalues as its diagonal. Given a signal h , the graph Fourier transform of h can be defined as $\hat{h} = U^T h$, and its inverse operation as $h = U\hat{h}$. The spectral convolution filter $\mathcal{F}$ can be formulated as: $\mathcal{F} * h = U f(\Lambda) U^T h$, where $f(\Lambda)$ is the graph convolution filter function corresponding to the signal frequency Λ . By defining the graph convolution kernel as:

$$\mathcal{F} = U f(\Lambda) U^T, \tag{1}$$

The following formulation connects the spectral-based and spatial-based GNNs in a unified form Balcilar et al. [2020]:

$$H^{(l+1)} = \sigma\left(\sum_{k=1}^{K} \mathcal{F}_k H^{(l)} W^{(l,k)}\right), \tag{2}$$

where $\mathcal{F}_k$ denotes the k-th graph convolution kernel, σ denotes the nonlinear activation function such as ReLU, $H^{(l)}$ is the node representations at l-th layer, and $W^{(l,k)}$ is the learnable parameter matrix.

4.2 The NLSD-GNN architecture

We now introduce our architecture, NLSD-GNN. At the ℓ-th layer, each node is associated with a F_ℓ-dimensional feature vector in a graph, and the feature *matrix* for all nodes is represented as $H^{(\ell)} \in \mathbb{R}^{N \times F_\ell}$. Following the Laplacian construction steps outlined in Definition 2, we require a *vector* $x \in \mathbb{R}^N$ to constructing an undirected counterpart of the graph. To effectively reduce the dimensionality of our node feature matrix $H^{(\ell)}$, we consider the following two schemes:

NLSD-GNN-MLP. We employ a simple MLP architecture with a tanh as an activation function between the layers except for the last layer. At each layer, we progressively halve the dimensionality until reaching dimensionality of one. Now, we can map the original directed graph to its undirected counterpart by leveraging this one-dimensional feature vector. Figure 2 illustrates our architecture.

NLSD-GNN-Fast. As a faster alternative, we also propose an architecture inspired by Louis [2014], Yadati et al. [2019], which creates the feature vector by leveraging a simple random projection.

Let $G = (V, E^+, E^-, w)$ be a signed-directed graph. Then, we train our GNN using the original feature matrix and the nonlinear Laplacian graph generated in the previous step as inputs. Following Equation (2), to generate the ℓ-th layer from the $(\ell - 1)$-th layer, we compute the matrix $H^{(\ell)}$ as follows:

$$H^{(\ell)} = \sigma\left(\tilde{D}_G^{-\frac{1}{2}} \tilde{A}_{G_x} \tilde{D}_G^{-\frac{1}{2}} H^{(\ell-1)} \Theta^{(\ell)}\right), \tag{3}$$

Table 1: Node classification accuracy (%) and its standard deviation for directed graphs. Top three models are color-coded as First, Second, Third.

Type	Method	Cornell	Texas	Wisconsin	Cora-ML	CiteSeer
Spectral	GCN	59.0 ± 6.4	58.7 ± 3.8	55.9 ± 5.4	82.0 ± 1.1	66.0 ± 1.5
	ChebNet	76.3 ± 3.2	79.2 ± 7.5	81.6 ± 6.3	80.0 ± 1.8	66.7 ± 1.6
Spatial	APPNP	58.7 ± 4.0	57.0 ± 4.8	51.8 ± 7.4	82.6 ± 1.4	66.9 ± 1.8
	SAGE	80.0 ± 6.1	84.3 ± 5.5	83.1 ± 4.8	82.3 ± 1.2	66.0 ± 1.5
	GIN	57.9 ± 5.7	65.2 ± 6.5	58.2 ± 5.1	78.1 ± 2.0	63.3 ± 2.5
	GAT	57.6 ± 4.9	61.1 ± 5.0	54.1 ± 4.2	81.9 ± 1.0	67.3 ± 1.3
Directed	DGCN	67.3 ± 4.3	71.7 ± 7.4	65.5 ± 4.7	81.3 ± 1.4	66.3 ± 2.0
	Digraph	66.8 ± 6.2	64.9 ± 8.1	59.6 ± 3.8	79.4 ± 1.8	62.6 ± 2.2
	DiGraphIB	64.4 ± 9.0	64.9 ± 13.7	64.1 ± 7.0	79.3 ± 1.2	61.1 ± 1.7
	MagNet	74.0 ± 4.5	83.3 ± 6.1	85.7 ± 3.2	79.8 ± 2.5	67.5 ± 1.8
NLSD-GNN-MLP		78.1 ± 4.5	85.5 ± 5.2	86.8 ± 4.3	79.3 ± 1.2	67.6 ± 2.6
NLSD-GNN-Fast		77.8 ± 3.9	85.4 ± 4.2	87.2 ± 3.5	82.2 ± 2.5	68.3 ± 1.2

Table 2: Test accuracy (%) comparison the signed-directed link prediction. Top three models are color-coded as First, Second.

Dataset	Task	SGCN	SDGNN	SiGAT	SNEA	SSSNET	SigMaNet	MSGNN	NLSD-GNN
BitCoin-Alpha	SP	64.7±0.9	64.5±1.1	62.9±0.9	64.1±1.3	67.4±1.1	47.8±3.9	71.3±1.2	72.7±0.8
	DP	60.4±1.7	61.5±1.0	61.9±1.9	60.9±1.7	68.1±2.3	49.4±3.1	72.5±1.5	70.5±0.5
	3C	81.4±0.5	79.2±0.9	77.1±0.7	83.2±0.5	78.3±4.7	37.4±16.7	84.4±0.6	85.6±1.6
	4C	51.1±0.8	52.5±1.1	49.3±0.7	52.4±1.8	54.3±2.9	20.6±6.3	58.5±0.7	58.7±0.3
	5C	79.5±0.3	78.2±0.5	76.5±0.3	81.1±0.3	77.9±0.3	34.2±6.5	81.9±0.9	82.6±0.3
BitCoin-OTC	SP	65.6±0.9	65.3±1.2	62.8±1.3	67.7±0.5	70.1±1.2	50.0±2.3	73.0±1.4	73.8±0.4
	DP	63.8±1.2	63.2±1.5	64.0±2.0	65.3±1.2	69.6±1.0	48.4±4.9	71.8±1.1	71.8±0.7
	3C	79.0±0.7	77.3±0.7	73.6±0.7	82.2±0.4	76.9±1.1	26.8±10.9	83.3±0.7	84.2±1.7
	4C	51.5±0.4	55.3±0.8	51.2±1.8	56.9±0.7	57.0±2.0	23.3±7.4	59.8±0.7	60.5±0.8
	5C	77.4±0.7	77.3±0.8	74.1±0.5	80.5±0.5	74.0±1.6	25.9±6.2	80.9±0.9	82.5±0.3
Slashdot	SP	74.7±0.5	74.1±0.7	64.0±1.3	70.6±1.0	86.6±2.2	57.9±5.3	92.4±0.2	93.2±0.6
	DP	74.8±0.9	74.2±1.4	62.8±0.9	71.1±1.1	87.8±1.0	53.0±4.0	93.1±0.1	93.3±0.1
	3C	69.7±0.3	66.3±1.8	49.1±1.2	72.5±0.7	79.3±1.2	42.0±7.9	86.1±0.3	86.8±0.5
	4C	63.2±0.3	64.0±0.7	53.4±0.2	60.5±0.6	72.7±0.6	25.7±8.9	78.2±0.3	79.1±1.3
	5C	64.4±0.3	62.6±2.0	44.4±1.4	66.4±0.5	70.4±0.7	19.3±8.6	76.8±0.6	78.4±0.3
Epinions	SP	62.9±0.5	67.7±0.8	63.6±0.5	66.5±1.0	78.5±2.1	53.3±10.6	85.4±0.5	86.1±0.4
	DP	61.7±0.5	67.9±0.6	63.6±0.8	66.4±1.2	73.9±6.2	49.0±3.2	86.3±0.3	86.7±0.6
	3C	70.3±0.8	73.2±0.8	52.3±1.3	72.8±0.2	72.7±2.0	30.5±8.3	83.1±0.5	85.2±0.7
	4C	66.7±1.2	71.0±0.6	62.3±0.5	69.5±0.7	70.2±5.2	29.9±6.4	78.7±0.9	79.7±0.1
	5C	73.5±0.8	76.6±0.7	52.9±0.7	74.2±0.1	70.3±4.6	22.1±6.1	80.5±0.5	81.3±0.1

where $x \in \mathbb{R}^N$ is obtained via an MLP for NLSD-GNN-MLP and a random projection for NLSD-GNN-Fast. Following the convolutional layers, we transform the matrix $H^{(L)} \in \mathbb{R}^{N \times F_L}$ into $N \times n_c$ matrix by applying a linear layer and then apply a weight matrix $W^{(L+1)} \in \mathbb{R}^{F_L \times n_c}$ from the right, where n_c is the number of classes. Finally, we apply softmax on the output of the final layer. In our experiments, we set L to 2 or 3. For link prediction, we apply the same method up to the last layer, then concatenate the rows associated with pairs of nodes to generate the edge features. we discussed the complexity of our method in Appendix.

5 Experiments

The node classification task consists in predicting the class label to which each node belongs. For this task, we only consider directed graphs with nonnegative edge weights, node classification is carried out in a semi-supervised setting. We employ 10 random data splits for all the datasets. Ten folds are generated randomly for each dataset. For the datasets Cornell, Texas, and Wisconsin, we used the settings from He et al. [2022b]. In all experiments, we used the normalized nonlinear Laplacian and implemented NLSD-GNN with convolution defined as in Equation (3), which means our network may be viewed as the nonlinear Laplacian generalization of ChebNet.

Link prediction typically aims to determine the presence of a connection between two nodes in unsigned and undirected graphs. However, this task extends into more complex scenarios involving signed and/or directed networks. The initial task, known as link sign prediction (SP), predicts the existence of a directed edge from v_i to v_j and seeks to classify the edge as positive or negative, determining whether $(v_i, v_j) \in E^+$ or $(v_i, v_j) \in E^-$. The next challenge, direction prediction (DP), involves predicting the directionality of the edge, i.e., whether $(v_i, v_j) \in E$ or $(v_j, v_i) \in E$, under the assumption that only one direction is valid. Further complexities are introduced through multi-class prediction tasks. The three-class challenge (3C) considers three possibilities: $(v_i, v_j) \in E$, $(v_j, v_i) \in E$, or neither. In the four-class scenario (4C), the predictions expand to include both positive and negative relationships in both directions: $(v_i, v_j) \in E^+$, $(v_i, v_j) \in E^-$, $(v_j, v_i) \in E^+$, and $(v_j, v_i) \in E^-$. The most complex, the five-class format (5C), adds an additional category where neither edge exists between v_i and v_j. Performance across these tasks is assessed using classification accuracy. It is noteworthy that while the (SP), (DP), and (3C) tasks require methods capable of discerning either signed or directed aspects, the (4C) and (5C) tasks necessitate methods that can adeptly handle both signs and directions of the edges.

5.1 Results & Discussion

Node Classification. As demonstrated in Table 1, on homophilic datasets, both variants of NLSD-GNN consistently rank among the top three models across the datasets. Notably, NLSD-GNN outperforms MagNet and other non-spectral methods, achieving the best results on three out of five datasets and placing second on the Cornell dataset. This superior performance underscores the effectiveness of the nonlinear Laplacian in capturing directional information. On the Cora-ML dataset, NLSD-GNN places third, slightly behind SAGE. In heterophilic datasets presented in Table 8, NLSD-GNN surpasses MagNet and similar spectral GNN methodologies. However, it falls short against GNNs specifically designed for directed heterophilic environments, such as HoloNet and Dir-GNN detailed in Koke and Cremers [2023], Rossi et al. [2024]. Moreover, these targeted heterophilic methods demonstrate minimal gains—and occasionally a slight decline—in performance when applied to homophilic datasets with directional data. It is also important to note that the models cited in Koke and Cremers [2023], Rossi et al. [2024] cannot handle signed-directed graphs. Additional results involving various baselines can be found in the Appendix.

Link Prediction on Signed-Directed Graphs. Table 2 summarizes the experimental results for link prediction on signed-directed graphs. We observe that NLSD-GNN-Fast outperforms all the baseline methods on almost all the datasets, and MSGNN generally achieves suboptimal results, followed by SSSNET, SNEA and SDGNN. In some cases especially For the SP task, it is observed that relying solely on signed attributes, excluding weighted features, yields favorable results. In summary, developing features that distinctly address the positive and negative subgraphs proves beneficial, and incorporating directional information generally enhances performance. Table 2 is instrumental in assessing the performance of the NLSD-GNN model, specifically examining how it handles signs and directions both individually and in combination. This constitutes the core of our ablation study on the NLSD-GNN's capabilities. While simpler tasks such as (SP), (DP), and (3C) evaluate the model's ability to address either signed or directed attributes separately, more intricate tasks like (4C) and (5C) challenge the model to effectively manage both attributes at the same time. We have the following observations: the performance of NLSD-GNN in task (4C) diminishes compared to task (5C), suggesting that effectively quantifying the influence of non-existent links and capturing the structural nuances of signed-directed graphs are crucial for modeling interactions between positive and negative links. Additionally, we observe that integrating directionality into the nonlinear Laplacian matrix generally enhances outcomes in tasks related to directionality (DP, 3C, 4C, 5C).

6 Conclusion

In this work, we introduced a spectral GNN based on a novel nonlinear signed-directed Laplacian and explored its application in node classification and link prediction tasks that consider both edge sign and directionality. The NLSD-GNN not only matches or surpasses the performance of leading GNNs but also operates more efficiently on real-world datasets.

References

Yixuan He, Xitong Zhang, Junjie Huang, Benedek Rozemberczki, Mihai Cucuringu, and Gesine Reinert. Pytorch geometric signed directed: A software package on graph neural networks for signed and directed graphs. In *Learning on Graphs Conference*, pages 12–1, 2024.

Srijan Kumar, Francesca Spezzano, VS Subrahmanian, and Christos Faloutsos. Edge weight prediction in weighted signed networks. In *Proceedings of the 2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 221–230, 2016.

Junjie Huang, Huawei Shen, Liang Hou, and Xueqi Cheng. Sdgnn: Learning node representation for signed directed networks. In *Proceedings of the AAAI conference on Artificial Intelligence (AAAI)*, volume 35, pages 196–203, 2021.

Weijia Ju, Ling Chen, Bin Li, Wei Liu, Jun Sheng, and Yuwei Wang. A new algorithm for positive influence maximization in signed networks. *Information Sciences*, 512:1571–1591, 2020.

Stefanos Bennett, Mihai Cucuringu, and Gesine Reinert. Detection and clustering of lead-lag networks for multivariate time series with an application to financial markets, 2022.

Yixuan He, Quan Gan, David Wipf, Gesine D Reinert, Junchi Yan, and Mihai Cucuringu. Gnnrank: Learning global rankings from pairwise comparisons via directed graph neural networks. In *International Conference on Machine Learning (ICML)*, pages 8581–8612, 2022a.

Quan Zheng, David B Skillicorn, and Olivier Walther. Signed directed social network analysis applied to group conflict. In *Proceedings of the 2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pages 1007–1014, 2015.

Yuichi Yoshida. Nonlinear Laplacian for digraphs and its applications to network analysis. *Proceedings of the 9th ACM International Conference on Web Search and Data Mining (WSDM)*, pages 483–492, 2016.

Yaoping Hou, Jiongsheng Li, and Yongliang Pan. On the Laplacian eigenvalues of signed graphs. *Linear and Multilinear Algebra*, 51(1):21–30, 2003.

M. Balcilar, Guillaume Renton, Pierre Héroux, Benoit Gaüzère, Sébastien Adam, and Paul Honeine. Bridging the gap between spectral and spatial domains in graph neural networks. *ArXiv*, abs/2003.11702, 2020.

Anand Louis. Hypergraph markov operators, eigenvalues and approximation algorithms. *Proceedings of the 47th Annual ACM symposium on Theory of Computing*, pages 713–722, 2014.

Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Pratim Talukdar. HyperGCN: A new method for training graph convolutional networks on hypergraphs. *Advances in Neural Information Processing Systems*, 32, 2019.

Yixuan He, Xitong Zhang, Junjie Huang, Mihai Cucuringu, and Gesine Reinert. Pytorch geometric signed directed: A survey and software on graph neural networks for signed and directed graphs. *ArXiv*, abs/2202.10793, 2022b.

Christian Koke and Daniel Cremers. Holonets: Spectral convolutions do extend to directed graphs. *ArXiv*, abs/2310.02232, 2023.

Emanuele Rossi, Bertrand Charpentier, Francesco Di Giovanni, Fabrizio Frasca, Stephan Günnemann, and Michael M Bronstein. Edge directionality improves learning on heterophilic graphs. In *Learning on Graphs Conference*, pages 25–1. PMLR, 2024.

Yi Ma, Jianye Hao, Yaodong Yang, Han Li, Junqi Jin, and Guangyong Chen. Spectral-based graph convolutional network for directed graphs. arXiv:1907.08990, 2019.

Zekun Tong, Yuxuan Liang, Changsheng Sun, David S. Rosenblum, and Andrew Lim. Directed graph convolutional network. arXiv preprint arXiv:2004.13970, 2020a.

Z. Tong, Yuxuan Liang, Changsheng Sun, Xinke Li, David S. Rosenblum, and A. Lim. Digraph inception convolutional networks. *Advances in Neural Information Processing Systems*, 33: 17907–17918, 2020b.

Xitong Zhang, Yixuan He, Nathan Brugnone, Michael Perlmutter, and Matthew J. Hirn. Magnet: A neural network for directed graphs. *Advances in Neural Information Processing Systems*, 34: 27003–27015, 2021.

Simon Geisler, Yujia Li, Daniel Jaymin Mankowitz, Ali Taylan Cemgil, Stephan Gunnemann, and Cosmin Paduraru. Transformers meet directed graphs. *ArXiv*, abs/2302.00049, 2023.

Sohir Maskey, Raffaele Paolino, Aras Bacho, and Gitta Kutyniok. A fractional graph Laplacian approach to oversmoothing. *ArXiv*, abs/2305.13084, 2023.

Sohir Maskey, Ali Parviz, Maximilian Thiessen, Hannes Stärk, Ylli Sadikaj, and Haggai Maron. Generalized Laplacian positional encoding for graph representation learning. *arXiv preprint arXiv:2210.15956*, 2022.

Yixuan He, Michael Permultter, Gesine Reinert, and Mihai Cucuringu. Msgnn: A spectral graph neural network based on a novel magnetic signed Laplacian. In *Learning on Graphs Conference*, 2022c.

Tyler Derr, Yao Ma, and Jiliang Tang. Signed graph convolutional networks. In *Proceedings of the 2018 IEEE International Conference on Data Mining (ICDM)*, pages 929–934, 2018.

Frank Harary. On the notion of balance of a signed graph. *Michigan Mathematical Journal*, 2(2): 143–146, 1953.

Junjie Huang, Huawei Shen, Liang Hou, and Xueqi Cheng. Signed graph attention networks. In *Proceedings of the 28th International Conference on Artificial Neural Networks (ICANN)*, pages 566–577, 2019.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio', and Yoshua Bengio. Graph attention networks. *ArXiv*, abs/1710.10903, 2017.

Yu Li, Yuan Tian, Jiawei Zhang, and Yi Chang. Learning signed network embedding via graph attention. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 34, pages 4772–4779, 2020.

Yixuan He, Gesine Reinert, Songchao Wang, and Mihai Cucuringu. Sssnet: semi-supervised signed network clustering. In *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*, pages 244–252, 2022d.

Yixuan He, Michael Perlmutter, Gesine Reinert, and Mihai Cucuringu. Msgnn: A spectral graph neural network based on a novel magnetic signed Laplacian. In *Learning on Graphs Conference*, pages 40–1, 2022e.

Yu Li, Meng Qu, Jian Tang, and Yi Chang. Signed laplacian graph neural networks. In *AAAI Conference on Artificial Intelligence*, 2023. URL `https://api.semanticscholar.org/CorpusID:259715656`.

Stefano Fiorini, Stefano Coniglio, Michele Ciavotta, and Enza Messina. Sigmanet: One Laplacian to rule them all. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 37, pages 7568–7576, 2023.

Rahul Singh and Yongxin Chen. Signed graph neural networks: A frequency perspective. *arXiv preprint arXiv:2208.07323*, 2022.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *Proceedings of the 15th Extended Semantic Web Conference (ESWC)*, pages 593–607, 2018.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. Composition-based multi-relational graph convolutional networks. *arXiv preprint arXiv:1911.03082*, 2019.

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.

Derek Lim, Felix Hohne, Xiuyu Li, Sijia Huang, Vaishnavi Gupta, Omkar Bhalerao, and Ser-Nam Lim. Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. In *Neural Information Processing Systems*, 2021. URL https://api.semanticscholar.org/CorpusID:239998578.

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*, abs/2302.11640, 2023. URL https://api.semanticscholar.org/CorpusID:257102689.

Table 3: Laplacians used in spectral GNNs and their applicability

Laplacian	Directed	Signed-Directed	No Extra Param
Ordinary Laplacian	×	×	✓
Magnetic Laplacian Zhang et al. [2021]	✓	×	×
Signed Laplacian Hou et al. [2003]	×	×	✓
Signed magnetic Laplacian He et al. [2022c]	✓	✓	×
Directed Laplacian Yoshida [2016]	✓	×	✓
Signed-directed Laplacian (This work)	✓	✓	✓

A Related Work

Inspired by GCN architecture, graph neural networks have been extended to directed graphs. DGCN Ma et al. [2019] uses a (linear) directed Laplacian associated with random walk, but this restricts the graph to be strongly connected which is not the case in most of the real-world scenarios. On the other hand, DiGCN Tong et al. [2020a] explores a method using a first-order proximity matrix and two second-order proximity matrices, and incorporates k-hop diffusion matrices. These matrices enable detailed neighborhood analysis and the construction of multiple Laplacians with a fusion operator for information integration. DiGraph Tong et al. [2020b] extended these ideas by adapting the directed Laplacian using a personalized PageRank matrix to be suitable for non-strongly connected graphs. They also incorporate higher-order receptive fields for enhanced data processing. MagNet Zhang et al. [2021] incorporates directional information in graphs through the use of the magnetic Laplacian, a complex Hermitian matrix. This matrix captures the undirected geometric structure in the magnitude of its entries and encodes directional information in their phase. Moreover, Geisler et al. [2023] extends Transformer GNNs for use in directed node/graph classification tasks, while Maskey et al. [2023, 2022] has utilized GNNs inspired by ordinary differential equations, extending the concept of oversmoothing to the context of directed graphs. Recently, Koke and Cremers [2023] tried to completely remove reliance on the graph Fourier transform and instead take the concept of learnable functions applied to characteristic operators as fundamental.

Recent research primarily explores unsigned graphs, which only include edges of positive signs. Yet, an expanding focus is on neural networks for signed and directed graphs, aimed at assessing link polarity. For instance, SGCN Derr et al. [2018] utilizes balance theory Harary [1953] for signed-undirected graphs, whereas SiGAT Huang et al. [2019], with a mechanism from Velickovic et al. [2017], employs a motif-based architecture for signed-directed graph embeddings. SDGNN Huang et al. [2021] and SNEA Li et al. [2020] further developed this area with new efficiencies and objective functions. In contrast, SSSNET He et al. [2022d], MSGNN He et al. [2022e], SLGNN Li et al. [2023] and SigMaNet Fiorini et al. [2023] explore semi-supervised clustering and spectral GNNs using signed magnetic Laplacians, respectively. Additionally, various GNNs Singh and Chen [2022], Schlichtkrull et al. [2018], Vashishth et al. [2019] address multi-relational graphs, which inherently accommodate diverse edge types. Signed graphs, particularly those unweighted or with finite weighting scales, offer simplified models of multi-relational graphs where negative edges are as significant as positive, but with fewer parameters needed for network training.

B Proof of Theorem 1 & 2

Let (λ, v) be an eigenpair of the Laplacian L_G of a signed-directed graph G. Suppose $\lambda < 0$. Then, we have

$$0 \leq Q_G(v) = v^\top L_G v = v^\top L_{G_v} v = v^\top(-\lambda)v = -\lambda\|v\|^2 < 0$$

which is a contradiction. Hence, L_G is positive semidefinite. A similar argument shows that the normalized Laplacian $\mathcal{L}_G$ is also positive semidefinite.

Now consider an operator $\mathcal{M} : x \mapsto 2x - \mathcal{L}_G(x)$. To show that all the eigenvalues of $\mathcal{L}_G$ is bounded by two, it suffices to show that $\mathcal{M}$ is positive semidefinite. This is equivalent to showing that the quadratic form of $\mathcal{M}$ is nonnegative. For a vertex $v \in V$, let d_v denote the total degree of v. Then, we have

$$
\begin{aligned}
x^\top \mathcal{M}(x) &= 2\|x\|^2 - x^\top D_G^{-1/2} Q_G(x) (D_G^{-1/2} x) \\
&= 2\|x\|^2 - \sum_{e=(u,v)\in E^+} w(e) \max\left\{ \frac{x_u}{d_u^{1/2}} - \frac{x_v}{d_v^{1/2}}, 0 \right\}^2 \\
&\quad - \sum_{e=(u,v)\in E^-} w(e) \left(\frac{2x_u^2}{d_u} + \frac{2x_v^2}{d_v} - \max\left\{ \frac{x_u}{d_u^{1/2}} - \frac{x_v}{d_v^{1/2}}, 0 \right\}^2 \right) \\
&= \sum_{e=(u,v)\in E^+} w(e) \left(\frac{2x_u^2}{d_u} + \frac{2x_v^2}{d_v} - \max\left\{ \frac{x_u}{d_u^{1/2}} - \frac{x_v}{d_v^{1/2}}, 0 \right\}^2 \right) \\
&\quad + \sum_{e=(u,v)\in E^-} w(e) \max\left\{ \frac{x_u}{d_u^{1/2}} - \frac{x_v}{d_v^{1/2}}, 0 \right\}^2 \\
&\geq 0.
\end{aligned}
$$

Hence, the claim holds. Note that the second-to-last expression is the quadratic form of the normalized Laplacian of a signed-directed graph obtained from G by flipping signs of edges.

C Further implementation details

We configured both ChebNet and MagNet and MSGNN with $K = 1$. Each model trained for up to 1500 epochs, employing early stopping when validation errors ceased decreasing after 500 epochs, applicable to both node classification and link prediction tasks. A dropout layer set at a rate of 0.5 precedes the final linear layer. Model selection was based on peak validation accuracy. Filter counts in the convolutional layers were tuned to $[16, 32, 48]$. Node classification learning rates were tested at $[1e^{-3}, 5e^{-3}, 1e^{-2}]$, and a more conservative rate of $1e^{-3}$ was applied to link prediction due to the larger sample sizes. Adam optimizer and $L2$ regularization (hyperparameter $5e^{-4}$) were utilized to mitigate overfitting. The best test performances were derived by grid-search based on validation accuracies. In link prediction, node features were derived from in-degrees and out-degrees to encode directional information from the adjacency matrix.

For link prediction task on signed-directed graphs, all GNNs are trained for 300 epochs. Loss functions from the original publications are employed for SGCN Derr et al. [2018], SNEA Li et al. [2020], SiGAT Huang et al. [2019], and SDGNN Huang et al. [2021], while cross-entropy loss L_{CE} is used for SigMaNet Fiorini et al. [2023], SSSNET He et al. [2022d], and MSGNN Fiorini et al. [2023]. For each input graph, 20% of edges are randomly set aside as test edges, and the remaining edges are used for training. Five random splits are generated for each graph. For SigMaNet, SSSNET , and MSGNN, node features are formed into four-dimensional vectors based on the signed in- and out-degrees, excluding test edges. Default settings from He et al. [2024] are utilized for SGCN , SNEA, SiGAT , and SDGNN.

Computational experiments were conducted on 1 compute nodes, equipped with 1 Nvidia Quadro RTX 8000 GPU, 380GB RAM, and 32 Intel Xeon E5-2660 v3 CPUs.

D Datasets

D.1 Node Classification

As illustrated in Table 5, we employ ten real datasets for node classification. We define a directed edge as follows: if $(u, v) \in E$ but $(v, u) \notin E$, then (u, v) is considered a directed edge. Conversely, if both $(u, v) \in E$ and $(v, u) \in E$, they are treated as undirected edges; undirected edges that are not self-loops are counted twice. For the citation datasets Cora-ML and Citeseer, we randomly select 20 nodes per class for training, 500 nodes for validation, and allocate the remainder for testing, as per. For Chameleon and Squirrel we use the fixed GEOM-GCN Pei et al. [2020] splits, for Arxiv-Year we use the splits provided in Lim et al. [2021], while for Roman-Empire we use the splits from Platonov et al. [2023]. We report the mean and standard deviation of the test accuracy, computed over 10 runs in all experiments..

Algorithm 2 Simple MLP for Graph Feature Reduction

Input: H, the initial feature matrix of the graph (dimensions $N \times D$)
Output: x, the one-dimensional feature vector (dimensions $N \times 1$)
Initialize $H_{\text{current}} = H$
Set $D_{\text{current}} = D$
while $D_{\text{current}} > 1$ **do**
  Halve the dimensionality at each layer: $D_{\text{next}} = \frac{D_{\text{current}}}{2}$
  for $i = 1$ to D_{next} **do**
    Apply MLP layer: $H_{\text{next}}[:, i] = \tanh(H_{\text{current}} \cdot W[:, i] + b[i])$
  end for
  Update H_{current} to H_{next}
  Update D_{current} to D_{next}
end while
Final transformation: $x = H_{\text{current}} \cdot W_{\text{final}} + b_{\text{final}}$
Return x

Table 4: Statistics of the datasets used in Node Classification Task.

Dataset	# Nodes	# Edges	# Feat.	# C	% Unidirectional Edges	Edge Hom.
CiteSeer	4,230	5,358	602	6	99.61	0.949
Cora-ML	2,995	8,416	2,879	7	96.84	0.792
Chameleon	2,277	36,101	2,325	5	85.01	0.235
Squirrel	5,201	217,073	2,089	5	90.60	0.223
Arxiv-year	169,343	1,166,243	128	40	99.27	0.221
Roman-Empire	22,662	44,363	300	18	65.24	0.050
Cornell	183	295	1,703	5	86.90	0.050
Texas	183	309	1,703	5	76.60	0.050
Wisconsin	251	499	1,703	5	77.90	0.050

Table 5: Summary of various datasets with link statistics for Signed-Directed Link prediction

Dataset	# Nodes	# Links	% Positive Links	% Negative Links
Bitcoin-OTC	5,872	21,431	85.29	14.71
Bitcoin-Alpha	3,772	14,077	90.69	9.31
Slashdot	82,140	549,202	77.40	22.6
Epinions	131,828	841,372	85.30	14.7

D.2 Link prediction on Signed-Directed graphs

We assess the performance of our NLSD-GNN using four well-known signed-directed graph datasets. The datasets include Bitcoin-Alpha and Bitcoin-OTC, which are networks of Bitcoin traders who assign trust or distrust labels to each other. Another dataset is Slashdot, a network where individuals label one another as friends or foes, derived from interactions on the Slashdot tech news site. The fourth dataset, Epinions, comprises a network from the consumer review site Epinions, where users express trust or distrust towards others. These datasets are summarized in Table 6, highlighting the significant imbalance between the numbers of positive and negative links, where positive links overwhelmingly predominate. In the results presented in Table 9, Following the setting of the Zhang et al. [2021] on the directed graphs, we develop our datasets for training, validation, and testing, which consist of vertex pairs from the graph, we implement the following procedures:

1. **Existence Prediction:** For a pair (u, v), the label is 0 if $(u, v) \in E$, and 1 otherwise. Class proportions are 25% and 75% for scenarios including undirected and multi-edges, and 50% each when considering only directed edges.
2. **Direction Prediction:** For an ordered node pair (u, v), we assign label 0 if $(u, v) \in E$ and label 1 if $(v, u) \in E$, provided either $(u, v) \in E$ or $(v, u) \in E$. Both edge types maintain a 50% proportion.

Table 6: Node classification accuracy (%) and its standard deviation for real-world directed heterophilic graphs. The numbers below network names indicate the degree of homophiliy. Top three models are color-coded as First, Second, Third.

Type	Method	Squirrel *0.223*	Chameleon *0.235*	Arxiv-Year *0.221*	Roman-Empire *0.05*
Baseline	MLP	28.77 ± 1.56	46.21 ± 2.99	36.70 ± 0.21	64.94 ± 0.62
	GCN	53.43 ± 2.01	64.82 ± 2.24	46.02 ± 0.26	73.69 ± 0.74
NON-GNN	H2GCN	37.90 ± 2.02	59.39 ± 1.98	49.09 ± 0.10	60.11 ± 0.52
	GPR-GNN	54.35 ± 0.87	62.85 ± 2.90	45.07 ± 0.21	64.85 ± 0.27
	LINKX	61.81 ± 1.80	68.42 ± 1.38	56.00 ± 0.17	37.55 ± 0.36
	FSGNN	74.10 ± 1.89	78.27 ± 1.28	50.47 ± 0.21	79.92 ± 0.56
	ACM-GCN	67.40 ± 2.21	74.76 ± 2.20	47.37 ± 0.59	69.66 ± 0.62
	GLOGNN	57.88 ± 1.76	71.21 ± 1.84	54.79 ± 0.25	59.63 ± 0.69
	Grad. Gating	64.26 ± 2.38	71.40 ± 2.38	63.30 ± 1.84	82.16 ± 0.78
Directed	DIGCN	37.74 ± 1.54	52.24 ± 3.65	OOM	52.71 ± 0.32
	MagNeT	39.01 ± 1.93	58.22 ± 2.87	60.29 ± 0.27	88.07 ± 0.27
	DIR-GNN	75.31 ± 1.92	79.71 ± 1.26	64.08 ± 0.26	91.23 ± 0.32
	HoloNet	76.71 ±1.92	80.33 ± 1.19	64.43 ± 0.28	92.24 ± 0.43
NLSD-GNN-Fast (ours)		68.34 ± 1.37	75.27 ± 1.42	62.13 ± 0.1	88.22 ± 0.43

3. **Three-Class Link Prediction:** For a pair (u, v), the labels are 0 if $(u, v) \in E$, 1 if $(v, u) \in E$, and 2 if neither condition is met, with respective proportions of 25%, 25%, and 50%.

4. **Direction Prediction by Three Classes Training:** This task builds on the third, evaluating only when $(u, v) \in E$ or $(v, u) \in E$.

Ten random folds were generated for all datasets. We allocate 15% and 5% of edges for testing and validation across all datasets. Also, we concatenate the original features, with the in-degree and out-degree as the node features in order to allow the models to learn structural information from the adjacency matrix directly. The connectivity is maintained by getting the undirected minimum spanning tree before removing edges for validation and testing. For the results in the main text, undirected edges and, if they exist, pairs of vertices with multiple edges between them, may be placed in the training/validation/testing sets.

E Ablation Study and discussion

Table 8 compares different variants of NLSD-GNN on the link prediction tasks. In our analysis, we evaluate the impact of including sign information in input node features and the consideration of edge weights. Typically, the calculation of degrees with signed edge weights yields net degrees, where edge weights are summed directly. This approach allows opposing signs, such as -1 and +1, to neutralize each other. In contrast, the features aggregating absolute values of edge weights are labeled as T' in the table. Each configuration is represented by a tuple: (inclusion of signed features, inclusion of weighted features), where "T" and "F" signify "True" and "False," respectively. Here, "T" indicates the sum of entries in the adjacency matrix, whereas "T'" represents the sum of the absolute values of these entries. Based on the results there is no significant difference between the two options, T and "T'", as weight features when signed features are excluded. Also, the results suggest that the use of unit-magnitude weights may be either advantageous or detrimental, varying by dataset and task.

Moreover, We achieve the best performance on the direction prediction task on all of the datasets as shown in Table 9. In comparison to the MagNet, NLSD-GNN obtains a $\sim 9\%$ improvement on Cornell and a $\sim 7\%$ improvement on Wisconsin, while on Cora-ML and CiteSeer it improves by $\sim$ %3 and %5 respectively. This emphasizes the role of the nonlinear Laplacian in our model. On the existence prediction, while MagNet has a better performance than our model, our model in three out of four datasets, is still in the top three models narrowly behind the second.

Table 7: Comparison of link prediction test performance across different variants of NLSD-GNN. The ablation study investigates the effects of incorporating signed and weighted features, where "T" and "F" denote "True" and "False," respectively. For weighted features, "T" indicates the sum of the entries in the adjacency matrix, whereas "T'" represents the sum of their absolute values.

Data Set	Link Task	(F, F)	(F, T)	(F, T')	(T, F)	(T, T)
BitCoin-Alpha	SP	71.6±0.9	71.6±1.6	70.2±1.1	71.5±0.9	72.7±0.8
	DP	74.8±1.0	71.8±1.6	71.6±1.8	70.3±0.5	70.5±0.5
	3C	85.8±0.9	82.2±0.4	84.4±0.6	83.6±0.6	85.6±1.6
	4C	58.8±1.1	54.2±0.3	56.6±1.6	58.4±1.4	58.7±0.3
	5C	83.3±0.6	81.0±0.5	81.9±0.5	83.2±0.3	82.6±0.3
BitCoin-OTC	SP	73.7±1.5	72.0±0.3	73.3±0.8	74.1±0.7	73.8±0.4
	DP	74.8±0.7	74±0.1	72.6±1.4	75.2±0.4	71.8±0.7
	3C	85.0±0.5	84.9±0.8	83.3±1.0	84.8±0.9	84.2±1.7
	4C	61.4±0.4	57.4±0.9	55.9±2.1	64.5±0.4	60.5±0.8
	5C	82.4±0.8	77.0±2.6	80.0±0.7	81.8±0.4	82.5±0.3
Slashdot	SP	93.1±0.0	92.0±0.1	93.0±0.1	92.8±0.1	93.2±0.6
	DP	92.1±0.1	91.1±0.1	93.1±0.1	92.0±0.1	93.3±0.1
	3C	84.2±0.2	85.3±0.4	86.2±0.3	85.2±0.2	86.8±0.5
	4C	72.3±1.1	72.1±0.1	70.7±1.2	79.0±0.6	79.1±1.3
	5C	74.8±0.3	72.1±0.6	72.8±0.3	78.5±0.6	78.4±0.3
Epinions	SP	85.4±0.1	86.6±0.1	86.4±0.1	84.6±0.2	86.1±0.4
	DP	86.1±0.5	86.1±0.7	86.1±0.4	83.3±0.1	86.7±0.6
	3C	83.5±0.2	83.3±0.3	83.6±0.4	82.6±0.3	85.2±0.7
	4C	76.7±0.3	77.3±0.3	76.2±0.4	79.1±1.5	79.7±0.1
	5C	78.5±0.5	78.1±0.2	77.3±0.2	80.6±0.2	81.3±0.1

Table 8: Link prediction accuracy (%) for directed graph. Top three models are color-coded as First, Second, Third.

Method	Direction prediction				Existence prediction			
	Cornell	Wisconsin	Cora-ML	CiteSeer	Cornell	Wisconsin	Cora-ML	CiteSeer
GCN	56.2 ± 8.7	71.0 ± 4.0	79.8 ± 1.1	68.9 ± 2.8	75.1 ± 1.4	75.1 ± 1.9	81.6 ± 0.5	76.9 ± 0.5
ChebNet	71.0 ± 5.5	67.5 ± 4.5	72.7 ± 1.5	68.0 ± 1.6	80.1 ± 2.3	82.5 ± 1.9	80.0 ± 0.6	77.4 ± 0.4
APPNP	69.5 ± 9.0	75.1 ± 3.5	83.7 ± 0.7	77.9 ± 1.6	74.9 ± 1.5	75.7 ± 2.2	82.5 ± 0.6	78.6 ± 0.7
SAGE	75.2 ± 11.0	72.0 ± 3.5	68.2 ± 0.8	68.7 ± 1.5	79.8 ± 2.4	77.3 ± 2.9	75.0 ± 0.0	74.1 ± 1.0
GIN	69.3 ± 6.0	74.8 ± 3.7	83.2 ± 0.9	76.3 ± 1.4	74.5 ± 2.1	76.2 ± 1.9	82.5 ± 0.7	77.9 ± 0.7
GAT	67.9 ± 11.1	53.2 ± 2.6	50.0 ± 0.1	50.6 ± 0.5	77.9 ± 3.2	74.6 ± 0.0	75.0 ± 0.0	75.0 ± 0.0
DGCN	80.7 ± 6.3	74.5 ± 7.2	79.6 ± 1.5	78.5 ± 2.3	80.0 ± 3.9	82.8 ± 2.0	82.1 ± 0.5	81.2 ± 0.4
DiGraph	79.3 ± 1.9	82.3 ± 4.9	80.8 ± 1.1	81.0 ± 1.1	80.6 ± 2.5	82.8 ± 2.6	81.8 ± 0.5	82.2 ± 0.6
DiGraphIB	79.8 ± 4.8	82.0 ± 4.9	83.4 ± 1.1	82.5 ± 1.3	80.5 ± 3.6	82.4 ± 2.2	82.2 ± 0.5	81.0 ± 0.5
MagNet	80.7 ± 2.7	83.6 ± 2.8	86.1 ± 0.9	85.1 ± 0.8	80.6 ± 3.8	82.9 ± 2.6	82.8 ± 0.7	79.9 ± 0.5
NLSD-GNN-MLP	88.4 ± 3.2	89.9 ± 4.2	89.8 ± 2.3	92.2 ± 3.0	80.4 ± 3.2	81.9 ± 1.1	82.8 ± 3.3	80.1 ± 0.7
NLSD-GNN-Fast	89.1 ± 5.6	90.3 ± 4.0	89.3 ± 0.1	90.1 ± 0.3	79.9 ± 1.2	82.5 ± 4.7	82.4 ± 2.3	79.8 ± 1.5

F Complexity Analysis

In this section, we evaluate the efficiency of our model against some of the directed baselines. Assuming the DirGCN configuration for the DirGNN baseline (with higher complexities possible for other configurations), consider a graph G with N nodes and M edges, input features of dimension d , and F hidden dimensions in our architectures. The computational complexities are as follows:

- **DiGCN:** Typically, In its full form, DiGCN has a $O(N^2)$ complexity, which is reduced to a $O(MFd)$ complexity in its approximate propagation scheme, which is used in practice.
- **DirGNN:** Similarly, DirGNN possesses a $O(MFd)$. In both cases this stems from the sparse-dense matrix multiplications that facilitate the forward function.
- **MagNet, MSGNN:** Both implement their forward functions through sparse-dense matrix multiplications, resulting in $O(MFd)$ complexity.

- **NLSD-GNN:** NLSD-GNN is similarly implemented via sparse-dense matrix multiplications; resulting in an $O(MFd)$ complexity.

Number of trainable parameters:

Compared to MSGNN, our **NLSD-GNN-MLP**, assuming equivalent network width and depth, introduces a slightly higher number of learnable parameters primarily in the MLP part. However, this increase is negligible, especially when employing a shallow MLP configuration in practical scenarios. Furthermore, for the **NLSD-GNN-FAST** variant, the total count of parameters aligns closely with that of MSGNN, showing no significant difference in parameter complexity between these models. in Algorithm 2, we present a simple pseudocode for how we can use an MLP for feature reduction to obtain a one-dimensional vector for use in our non-linear Laplacian. This demonstrates that enhancements in our model do not necessarily come at the cost of increased computational overhead.

G Limitations and Ethical Considerations:

In this work, we introduced a spectral GNN based on a novel nonlinear signed-directed Laplacian and explored its application in node classification and link prediction tasks that consider both edge sign and directionality. The NLSD-GNN not only matches or surpasses the performance of leading GNNs but also operates more efficiently on real-world datasets. Future work will investigate additional properties of our proposed Laplacian and extend its application to temporal and dynamic graphs, where node features and edge information may evolve over time.

Our method extends spectral graph convolutional networks to signed-directed graphs, though scalability to larger graphs remains a challenge for future development. Additionally, the potential of NLSD-GNN in heterophilic graph contexts has not been explored, indicating further avenues for research. The societal impact of this method aligns with that of other graph neural network algorithms, neither significantly greater nor lesser.