SigMaNet: One Laplacian to Rule Them All

Stefano Fiorini[1], **Stefano Coniglio**[2], **Michele Ciavotta**[1], **Enza Messina**[1]

[1] University of Milano-Bicocca, Milan, Italy,
[2] University of Bergamo, Bergamo, Italy
stefano.fiorini@unimib.it, stefano.coniglio@unibg.it, michele.ciavotta@unimib.it, enza.messina@unimib.it

Abstract

This paper introduces SigMaNet, a generalized Graph Convolutional Network (GCN) capable of handling both undirected and directed graphs with weights not restricted in sign nor magnitude. The cornerstone of SigMaNet is the *Sign-Magnetic Laplacian* (L^σ), a new Laplacian matrix that we introduce *ex novo* in this work. L^σ allows us to bridge a gap in the current literature by extending the theory of spectral GCNs to (directed) graphs with both positive and negative weights. L^σ exhibits several desirable properties not enjoyed by other Laplacian matrices on which several state-of-the-art architectures are based, among which encoding the edge direction and weight in a clear and natural way that is not negatively affected by the weight magnitude. L^σ is also completely parameter-free, which is not the case of other Laplacian operators such as, e.g., the *Magnetic Laplacian*. The versatility and the performance of our proposed approach is amply demonstrated via computational experiments. Indeed, our results show that, for at least a metric, SigMaNet achieves the best performance in 15 out of 21 cases and either the first- or second-best performance in 21 cases out of 21, even when compared to architectures that are either more complex or that, due to being designed for a narrower class of graphs, should—but do not—achieve a better performance.

Introduction

The dramatic advancements of neural networks and deep-learning have provided researchers and practitioners with extremely powerful analytics tools. Increasingly complex phenomena and processes which can often be modeled as graphs or networks, such as, e.g., social networks (Backstrom and Leskovec 2011), knowledge graphs (Zou 2020), protein interaction networks (Kashyap et al. 2018), or the World Wide Web (only to mention a few) can now be successfully addressed via Graph Convolutional Networks (GCNs). Compared with other approaches, GCNs effectively manage to represent the data and its interrelationships by explicitly capturing the topology of the underlying graph within a suitably-designed convolution operator.

In the literature, GCNs mostly belong to two categories: spectral based and spatial based (Wu et al. 2020). Spatial GCNs define the convolution operator as a localized aggregation operator (Wang et al. 2019) (although, rigorously speaking, such an operator may not always be called a convolution operator from a mathematical perspective). Differently, spectral GCNs define the convolution operator (a rigorous one in this case) as a function of the eigenvalue decomposition of the Laplacian matrix associated with the graph (Kipf and Welling 2017). In their basic definition, both methods assume the graph to be undirected and to feature nonnegative weights. For a comprehensive review, we refer the reader to (Zhang et al. 2019; Wu et al. 2020; He et al. 2022c).

Many real-world processes and phenomena can be modeled as directed graphs. While spatial GCNs have a natural extension to directed graphs, most spectral methods require the graph to be undirected and to feature nonnegative weights for their convolution operator to be well-defined. Indeed, due to being based on graph signal processing (Sandryhaila and Moura 2013; Chen et al. 2015), spectral GCNs require three fundamental properties to be satisfied which fail to hold when the graph is directed and/or features negative weights: *a)* the Laplacian matrix must be diagonalizable, i.e., it must admit an eigenvalue decomposition, *b)* the Laplacian matrix must be positive semidefinite, and *c)* the spectrum of the normalized Laplacian matrix must be upper-bounded by 2 (Kipf and Welling 2017; Wu et al. 2020).

In recent years, several extensions of the definition of Laplacian matrix have been proposed to overcome the first limitation, i.e., to handle directed graphs (see, e.g., the *Magnetic Laplacian* (Zhang et al. 2021b,a) and the *Approximate Digraph Laplacian* constructed via the PageRank matrix (Tong et al. 2020a)). Differently, no spectral techniques have been introduced so far to overcome the second limitation, i.e., to handle graphs with edge weights unrestricted in sign, which arise in many relevant applications (such as, e.g., those where the graph models credit/debit transactions between customers, like/dislike evaluations among users, or positive/negative user opinions). This paper aims at overcoming such a major limitation.

Main Contributions and Novelty of the Work

- We extend the theory of spectral-based GCNs by introducing SigMaNet, the first spectral GCN capable of handling both directed and undirected graphs with weights unrestricted in sign and of arbitrary magnitude.
- We introduce the *Sign-Magnetic Laplacian* matrix L^σ, a

new Laplacian matrix (which can be of independent interest beyond the scope of this paper) upon which SigMaNet is built. We show that L^σ satisfies all the properties that are needed for the definition of a convolution operator, among which being positive semidefinite regardless of the sign and magnitude of the edge weights. L^σ also exhibits useful structural properties, among which being positively homogeneous (thus being proportional to the magnitude of the graph weights) and allowing for a natural interpretation of the weight sign in terms of the edge direction—two properties that other Laplacian matrices do not enjoy. Compared with other proposals, the definition of L^σ is also parameter-free.

- We experiment with SigMaNet on four tasks using both real-world and synthetically-generated datasets. For the first task, we consider graphs with positive and negative edge weights, for which SigMaNet is the only spectral GCN that can be adopted, and find it to outperform the (more complex) state-of-the-art networks in 3 cases out of 5. For the other three tasks, we consider graphs with nonnegative weights, which allows us to compare SigMaNet with the other state-of-the-art spectral GCNs. While these GCNs are designed solely for graphs with nonnegative weights and, thus, one may expect that the wider applicability of SigMaNet may come at the price of an inferior performance, our experiments show that this is not the case, as SigMaNet is found to outperform the other GCNs in 11 cases out of 16. Across all four tasks, SigMaNet is consistently either the best performing or the second-best performing method according to at least a metric.

Preliminaries and Previous Works

For a given $n \in \mathbb{N}$, we denote by $[n]$ the set of integers $\{1, \dots, n\}$. For a given matrix M of appropriate dimensions with real eigenvalues, we denote its largest eigenvalue by $\lambda_{\max}(M)$. Throughout the paper, e and $ee^\top$ denote the all-one vector and matrix of appropriate dimensions. Undirected and directed graphs are denoted by $G = (V, E)$, where V is the set of vertices and E the set of edges. In the undirected (directed) case, E is a collection of unordered (ordered) pairs of elements in V. G is always assumed to be self-loop free.

Generalized Convolution Matrices

Let $M \in \mathbb{C}^{n\times n}$, $n \in \mathbb{N}$, be a positive semidefinite Hermitian matrix[1] with eigenvalue (or spectral) decomposition $M = U\Lambda U^*$, where $\Lambda \in \mathbb{R}^{n\times n}$ is the diagonal matrix whose elements are the (real, as M is Hermitian) eigenvalues of M, $U \in \mathbb{C}^{n\times n}$, and U^* is U's complex conjugate. For each $i \in [n]$, the i-th column of U coincides with the i-th eigenvector of M corresponding to its i-th eigenvalue Λ_{ii}. The columns of U form a basis of $\mathbb{C}^n$. We assume $\lambda_{\max}(M) \leq 2$.

Given a *signal* $x \in \mathbb{C}^n$, let $\widehat{x}$ be its discrete Fourier transform with basis U, i.e., $\widehat{x} = U^*x$. As $U^{-1} = U^*$, the transform is invertible and the inverse transform of $\widehat{x}$ reads $x = U\widehat{x}$. Given a *filter* $y \in \mathbb{C}^n$, the transform of its convolution with x satisfies the relationship $\widehat{y * x} = \widehat{y} \odot \widehat{x} = \mathrm{Diag}(\widehat{y})\widehat{x}$, where $*$ and $\odot$ denote the convolution and the Hadamard (or component-wise) product, respectively. Applying the inverse transform, we have $y * x = U\,\mathrm{Diag}(\widehat{y})U^*x$. Letting $\Sigma := \mathrm{Diag}(\widehat{y})$, we call a *generalized convolution matrix* the matrix $Y := U\Sigma U^*$, as $y * x = Yx$.

Let $\tilde{\Lambda} := \frac{2}{\lambda_{\max}(M)}\Lambda - I$ be the normalization of Λ. As $UU^* = I$, the same normalization applied to M leads to $\tilde{M} = U\tilde{\Lambda}U^* = \frac{2}{\lambda_{\max}}M - I$. Following (Hammond, Vandergheynst, and Gribonval 2011; Kipf and Welling 2017), we assume that y is such that the entries of $\widehat{y}$ are real-valued polynomials in $\tilde{\Lambda}$, i.e., that $\widehat{y}_i = \sum_{k=0}^K \theta_k T_k(\tilde{\lambda}_i), i \in [n]$, where $\theta_0, \dots, \theta_K \in \mathbb{R}$, $K \in \mathbb{N}$, and T_k is the Chebyshev polynomial of the first kind of order k. T_k is recursively defined as $T_0(x) = 1, T_1(x) = x$, and $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$ for $k \geq 2$, with $x \in \mathbb{R} \cap [-1, 1]$. Thus, we rewrite Σ as $\Sigma = \mathrm{Diag}(\widehat{y}) = \sum_{k=0}^K \theta_k T_k(\tilde{\Lambda})$, where T_k is applied component-wise to $\tilde{\Lambda}$, i.e., $(T_k(\tilde{\Lambda}))_{ij} = T_k(\tilde{\Lambda}_{ij})$ for all $i, j \in [n]$. With this, the convolution of x by y can be rewritten as $Yx = U\Sigma U^*x = U\left(\sum_{k=0}^K \theta_k T_k(\tilde{\Lambda})\right)U^*x$. Since, as it is easy to verify, $(U\tilde{\Lambda}U^*)^k = U\tilde{\Lambda}^k U^*$ holds for all $k \in \mathbb{N}$, one can also verify that $Yx = U\left(\sum_{k=0}^K \theta_k T_k(\tilde{\Lambda})\right)U^*x = \sum_{k=0}^K \theta_k T_k(U\tilde{\Lambda}U^*)x = \sum_{k=0}^K \theta_k T_k(\tilde{M})x$.

Assuming $\lambda_{\max} = 2$, we have $\tilde{M} = M - I$. Letting $K = 1$ and $\theta_1 = -\theta_0$, deduce:

$$y * x = Yx = (\theta_0 I - \theta_0(M - I))x = \theta_0(2I - M)x. \quad (1)$$

If M is chosen so as to express the topology of the graph and x coincides with the graph features, Eq. (1) represents the convolution operation underlying a spectral GCN. M should satisfy three properties for Eq. (1) to apply: *i)* it should admit an eigenvalue decomposition, *ii)* it should be positive semidefinite, and *iii)* its spectrum should be upper-bounded by 2. Examples of M are given in the following subsections.

Spectral Convolutions for Undirected Graphs

Let $G = (V, E)$ be an undirected graph with $n = |V|$ without weights nor signs associated with its edges and let $A \in \{0, 1\}^{n\times n}$ be its adjacency matrix, with $A_{ij} = 1$ if and only if $\{i, j\} \in E$. The Laplacian matrix of G is defined as:

$$L := D - A,$$

where $D := \mathrm{Diag}(Ae)$ is a diagonal matrix and, for each $i \in V$, D_{ii} equal to the degree of node i (Chung and Graham 1997). The normalized Laplacian matrix is defined as:

$$\begin{aligned} L_{\text{norm}} :&= D^{-\frac{1}{2}}LD^{-\frac{1}{2}} = D^{-\frac{1}{2}}(D - A)D^{-\frac{1}{2}} \\ &= I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}. \end{aligned}$$

L_{norm} satisfies many properties, among which *i)*, *ii)* and *iii)*.

The spectral convolution on the undirected graph G introduced by Kipf and Welling (2017) is obtained by letting $M := L_{\text{norm}}$ and defining Y as done before. Eq. (1) becomes:

$$\begin{aligned} y * x = Yx &= \theta_0(2I - (I - D^{-\frac{1}{2}}AD^{-\frac{1}{2}})x \\ &= \theta_0(I + D^{-\frac{1}{2}}AD^{-\frac{1}{2}})x. \end{aligned} \quad (2)$$

[1] A matrix is called Hermitian if its real part is symmetric and its imaginary part skew-symmetric.

To alleviate numerical instabilities and exploding/vanishing gradients when training a GCN built on Eq. (2), Kipf and Welling (2017) suggest the adoption of the following modified equation with a modified convolution matrix $\tilde{Y}$:

$$y * x = \tilde{Y}x = \theta_0(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}})x, \tag{3}$$

where $\tilde{A} := A + I$ and $\tilde{D} := \mathrm{Diag}(\tilde{A}e)$.

Drawbacks The Laplacian matrix L is well defined only for undirected graphs with (if any) nonnegative weights for two reasons. *i*) If G is a directed graph, the adjacency matrix A (and, thus, L) is, in the general case, not symmetric; thus, L may not admit an eigenvalue decomposition. *ii*) If G is directed, the sum of the columns of A (out degree) is not necessarily identical to the sum of its rows (in degree); thus, the matrix D is not well defined (as $\mathrm{Diag}(Ae) \neq \mathrm{Diag}(e^\top A)$). If G is undirected but features edges $\{i, j\} \in E$ with a negative weight $w_{ij} < 0$, L is not necessarily positive semidefinite as, even if $\mathrm{Diag}(Ae) = \mathrm{Diag}(e^\top A)$, $D^{-\frac{1}{2}} \notin I\!R^{n\times n}$ if $D_{ii} < 0$.

Spectral Convolutions for Directed Graphs

Since the adjacency matrix of a directed graph is asymmetric, the Laplacian matrix L defined before does not enjoy properties *i)*, *ii)* and *iii)* and, therefore, it is not possible to directly apply Eq. (2) to define a spectral graph convolution. Alternative approaches such as those of Tong et al. (2020b) and Tong et al. (2020a) (which split the adjacency matrix A into a collection of symmetric matrices in such a way that the information regarding the direction of the edges is not lost) are known, but they typically come at the cost of increasing the size and complexity of the neural network.

A more direct way to encode the directional information of the edges is resorting to complex-valued matrices that are Hermitian. Indeed, albeit asymmetric in the general case, Hermitian matrices admit an eigenvalue decomposition with real eigenvalues. The *Magnetic Laplacian* is one such matrix. It was first introduced in particle physics and quantum mechanics by Lieb and Loss (1993) and then applied in the context of community detection by Fanuel et al. (2018), in graph signal processing by Furutani et al. (2019) and, lastly, in the context of spectral GCNs by Zhang et al. (2021b,a).

Let $A_s := \frac{1}{2}\left(A + A^\top\right)$ be the symmetrized version of A and let $D_s := \mathrm{Diag}(A_s e)$. The *Magnetic Laplacian* is defined as the following Hermitian positive semidefinite matrix:

$$L^{(q)} := D_s - H^{(q)}, \qquad \text{with}$$

$$H^{(q)} := A_s \odot \exp\left(\mathbf{i}\,\Theta^{(q)}\right), \Theta^{(q)} := 2\pi q\left(A - A^\top\right),$$

where $\mathbf{i}$ is the square root of the negative unit, i.e., $\mathbf{i} = \sqrt{-1}$, and $\exp\left(\mathbf{i}\,\Theta^{(q)}\right) = \cos(\Theta^{(q)}) + \mathbf{i}\sin(\Theta^{(q)})$, where $\cos(\cdot)$ and $\sin(\cdot)$ are applied component-wise. Θ is a phase matrix that captures the directional information of the edges. The parameter $q \geq 0$ represents the electric charge. It is typically set to values smaller than 1 such as $\left[0, \frac{1}{4}\right]$ as in Zhang et al. (2021b) or $\left[0, \frac{1}{2}\right]$ as in Fanuel, Alaíz, and Suykens (2017). If $q = 0$, $\Theta^{(q)} = 0$ and $L^{(q)}$ boils down to the Laplacian matrix L defined on the "symmetrized" version of the graph with adjacency matrix A_s (which, crucially, renders G completely undirected and its directional information is lost).

For unweighted directed graphs where $A \in \{0, 1\}^{n\times n}$, $H^{(q)}$ straightforwardly captures the graph's directional information. Assuming $q = 0.25$, we have $H^{(q)}_{ij} = H^{(q)}_{ji} = 1 + \mathbf{i}0$ if $(i, j), (j, i) \in E$ and $H^{(q)}_{ij} = 0 + \mathbf{i}\frac{1}{2}$ and $H^{(q)}_{ji} = 0 - \mathbf{i}\frac{1}{2}$ if $(i, j) \in E \wedge (j, i) \notin E$. This way, digons (pairs of antiparallel edges) are represented as single undirected edges in the real part of $H^{(q)}$ whereas any other edge is represented in the imaginary part of $H^{(q)}$ with a sign encoding its direction.

Drawbacks The *Magnetic Laplacian* $L^{(q)}$ suffers from two drawbacks. Drawback #1 is that $L^{(q)}$ is well defined only for graphs with nonnegative weights. Indeed, if $(D_s)_{ii} < 0$ for some $i \in V$, in the general case $L^{(q)}$ is not positive semidefinite and $D_s^{-\frac{1}{2}}$ does not belong to $I\!R^{n\times n}$. Drawback #2 is that, even when restricted to graphs with nonnegative weights, $L^{(q)}$ exhibits a crucial sign-pattern inconsistency if the edge weights are sufficiently large. Indeed, while for unweighted graphs $L^{(q)}$ always captures the directional information of the edges by the sign of the imaginary part of $H^{(q)}$, this is not necessarily the case for weighted graphs, where the sign pattern of $H^{(q)}$ can drastically change irrespective of the edge direction by just scaling the edge weights by a positive constant. To see this, assume, for instance, $(i, j) \in E$ and $(j, i) \notin E$ with $A_{ij} = 1$. Then, we obtain: $H^{(0.25)}_{ij} = 0.40\cdot 0.31 + \mathbf{i}0.40\cdot 0.95$ and $H^{(0.25)}_{ji} = 0.40\cdot 0.31 - \mathbf{i}0.40\cdot 0.95$ by scaling A_{ij} by 0.8; $H^{(0.25)}_{ij} = -1 + \mathbf{i}0$ by scaling A_{ij} by 2; $H^{(0.25)}_{ij} = 0 + \mathbf{i}\frac{5}{2}$ by scaling A_{ij} by 5; and $H^{(0.25)}_{ij} = \frac{36}{2} + \mathbf{i}0$ by scaling A_{ij} by 36. This shows that $L^{(q)}$ is not robust to scaling and that, in it, the edge direction information can easily be lost.

Our Proposal: The Sign-Magnetic Laplacian and SigMaNet

In this section, we extend the theory underlying spectral GCNs by introducing the *Sign-Magnetic Laplacian* matrix, a positive semidefinite Hermitian matrix that well captures the directional as well as the weight information of any directed graph with weights unrestricted in sign nor magnitude without suffering from the two drawbacks we outlined before.

Sign-Magnetic Laplacian

We introduce the following Hermitian matrix, which we refer to as the *Sign-Magnetic Laplacian*:

$$L^\sigma := \bar{D}_s - H^\sigma, \qquad \text{with}$$

$$H^\sigma := A_s \odot \left(ee^\top - \mathrm{sgn}(|A - A^\top|) + \mathbf{i}\,\mathrm{sgn}\left(|A| - |A^\top|\right)\right),$$

where $A_s := \frac{1}{2}\left(A + A^\top\right)$, $\bar{D}_s := \mathrm{Diag}(|A_s|\,e)$, and $\mathrm{sgn} : I\!R \to \{-1, 0, 1\}$ is the signum function (applied component-wise). Let us illustrate the way the graph topology and its weights are stored in H^σ. H^σ encodes the direction and weight of every edge $(i, j) \in E$ that does not have an antiparallel edge (j, i) purely in its imaginary part

by $H^\sigma_{ij} = -H^\sigma_{ji} = 0 + \mathbf{i}\frac{1}{2}A_{ij}$. Pairs of antiparallel edges with $A_{ij} = A_{ji}$ are encoded purely in the real part by $H^\sigma_{ij} = H^\sigma_{ji} = \frac{1}{2}(A_{ij} + A_{ji}) + \mathbf{i}0$ (as if they coincided with an undirected edge of the same weight). Differently, pairs of antiparallel edges with $A_{ij} \neq A_{ji}$ are encoded purely in the imaginary part by $H^\sigma_{ij} = -H^\sigma_{ji} = 0 + \mathbf{i}\frac{1}{2}(A_{ij} + A_{ji})$ if $|A_{ij}| > |A_{ji}|$ and $H^\sigma_{ij} = -H^\sigma_{ji} = 0 - \mathbf{i}\frac{1}{2}(A_{ij} + A_{ji})$ if $|A_{ij}| < |A_{ji}|$. We define the normalized version of L^σ as:

$$L^\sigma_{\text{norm}} := \bar{D}_s^{-\frac{1}{2}} L^\sigma \bar{D}_s^{-\frac{1}{2}} = I - \bar{D}_s^{-\frac{1}{2}} H^\sigma \bar{D}_s^{-\frac{1}{2}}. \quad (4)$$

One can show that both L^σ and L^σ_{norm} are Hermitian by construction. Therefore, they admit an eigenvalue decomposition and, thus, satisfy property *i)*.

L^σ is defined in such a way that, if G is unweighted, it mirrors the behavior of $L^{(q)}$ with $q = 0.25$:

Theorem 1. *If $A \in \{0,1\}^{n\times n}$ and $q = 0.25$, $L^\sigma = L^{(q)}$.*

In contrast with $L^{(q)}$, L^σ does not suffer from drawback #1 as it is well-defined even when G features negative weights.

With the following two results, we show that L^σ and L^σ_{norm} enjoy the two remaining properties *ii)* and *iii)* that are required for the construction of a convolution operator:

Theorem 2. *L^σ and L^σ_{norm} are positive semidefinite.*

Theorem 3. *$\lambda_{max}(L^\sigma_{norm}) \leq 2$.*

With the next result, we show that L^σ encodes the topology of G (including its directions) and the weights of its edges in such a way that it is always proportional to the magnitude of A (i.e., to the magnitude of graph weights):

Theorem 4. *Given a constant $\alpha \in \mathbb{R}^+$, L^σ satisfies the homogeneity property $L^\sigma(\alpha A) = \alpha L^\sigma(A)$, where $L^\sigma(\alpha A)$ and $L^\sigma(A)$ are the Sign-Magnetic Laplacian matrices of a directed graph with, respectively, adjacency matrix $\alpha A \in \mathbb{R}^{n\times n}$ and $A \in \mathbb{R}^{n\times n}$.*

Theorem 4 shows that L^σ is robust to scaling applied to the weights of G. From it, we deduce the following result, which shows that L^σ does not suffer from drawback #2:

Corollary 1. *The sign-pattern of L^σ is uniquely determined by the topology of G and, thus, L^σ does not suffer from any sign-pattern inconsistencies.*

Lastly, we show that L^σ satisfies the following invariant:

Theorem 5. *Given a weighted digon-free directed graph $G = (V, E)$ and a directed edge $(i, j) \in E$ of weight w_{ij}, let $G' = (V, E')$ be a graph obtained by reversing the direction of (i, j) in G into (j, i) and flipping the weight sign with $w_{ji} = -w_{ij}$. Letting $L^\sigma(G)$ and $L^\sigma(G')$ be the L^σ matrix defined on G and G', resp., we have: $L^\sigma(G) = L^\sigma(G')$.*

Theorem 5 shows that the behavior of L^σ is consistent with applications where the graph models a flow relationship in which flipping the sign of an edge coincides with flipping its direction. This applies to, among others, scenarios where the weights represent flow values such as cash flows, where it is reasonable to assume that a negative flow from i to j correspond to a positive flow from j to i.

SigMaNet's Architecture

For the spectral convolution operator to be well defined the Laplacian matrix must satisfy properties *i)*,*ii)*, and *iii)*. As the hermiticity of L^σ_{norm}, Theorem 2, and Theorem 3 show that L^σ_{norm} enjoys these properties, Eq. (2) can be rewritten as:

$$Yx = \theta_0 \left(I + \bar{D}_s^{-\frac{1}{2}} H^\sigma \bar{D}_s^{-\frac{1}{2}}\right) x.$$

Following Kipf and Welling (2017) to avoid numerical instabilities, we apply Eq. (3) with $\tilde{D}_s^{-\frac{1}{2}} \tilde{H}^\sigma \tilde{D}_s^{-\frac{1}{2}}$ *in lieu* of $I + \bar{D}_s^{-\frac{1}{2}} H^\sigma \bar{D}_s^{-\frac{1}{2}}$, where $\tilde{H}^\sigma$ and $\tilde{D}_s$ are defined based on $\tilde{A} := A + I$ rather than A. We generalize the feature vector signal $x \in \mathbb{C}^{n\times 1}$ to a feature matrix signal $X \in \mathbb{C}^{n\times c}$ with c input channels (i.e., a c-dimensional feature vector for every node of the graph). Letting $\Theta \in \mathbb{C}^{c\times f}$ be a matrix of learnable filter parameters with f filters and ϕ be an activation function applied component-wise to the input matrix, the output $Z^\sigma \in \mathbb{C}^{n\times f}$ of SigMaNet's convolutional layer is:

$$Z^\sigma(X) = \phi(\tilde{D}_s^{-\frac{1}{2}} \tilde{H}^\sigma \tilde{D}_s^{-\frac{1}{2}} X\Theta). \quad (5)$$

Since the argument of ϕ is a complex matrix and, thus, traditional activation functions cannot be directly adopted, we follow Zhang et al. (2021b) and rely on a complex version of the *ReLU* activation function which is defined for a given $z \in \mathbb{C}$ as $\phi(z) = z$ if $\Re(z) \geq 0$ and $\phi(z) = 0$ otherwise. As the output of the convolutional layer Z^σ is complex-valued, to coerce it into the reals without information loss we apply an *unwind* operation by which $Z^\sigma(X) \in \mathbb{C}^{n\times f}$ is transformed into $[\Re(Z^\sigma(X)); \Im(Z^\sigma(X)] \in \mathbb{R}^{n\times 2f}$. To obtain the final result based on the task at hand, we apply either a linear layer with weights W or a $1D$ convolution.

Considering, e.g., the task of predicting the class of an edge, SigMaNet is defined as:

$$\text{softmax}\left(\text{unwind}\left(Z^{\sigma(2)}\left(Z^{\sigma(1)}\left(X^{(0)}\right)\right)\right)W\right),$$

where $X^{(0)} \in \mathbb{R}^{n\times c}$ is the input feature matrix, $Z^{\sigma(1)} \in \mathbb{C}^{n\times f_1}$ and $Z^{\sigma(2)} \in \mathbb{C}^{n\times f_2}$ are the spectral graph convolutional layers, $W \in \mathbb{R}^{2f_2\times d}$ are the weights of the linear layer (with d being the number of classes), and softmax : $\mathbb{R}^d \to \mathbb{R}^d$ is the normalized exponential activation function.

SigMaNet features a flexible architecture that differs from other spectral GCNs in the literature (e.g., MagNet (Zhang et al. 2021b)) mainly in the way the convolutional layer is defined. As such, it can easily be applied to a variety of tasks in an almost task-agnostic way (provided that one defines a suitable loss function) while architectures such as MagNet are suitable only for tasks whose graph has nonnegative edge weights. As L^σ is entirely parameter-free, SigMaNet does not require any fine-tunings to optimize the propagation of topological information through the network, differently from, e.g., DiGraph (Tong et al. 2020a) and MagNet.

Complexity of SigMaNet

Assuming, as done in our experiments, that SigMaNet features two graph-convolutional layers with f_1 and f_2 filters, each defined as in Eq. (5) and c features per node, the complexity of SigMaNet is $O(nc(n + f_1) + nf_1(n + f_2) + \Gamma)$,

where Γ accounts for the complexity of the last (task-specific) layer. For the four tasks that we consider in the next section, we have $\Gamma = m^{\text{train}} f_2 d$ for the first three (link sign/direction/existence prediction), where m^{train} is the number of edges in the training set and d is the number of classes, and $\Gamma = n f_2 d$ for the last one (node classification). The complexity is quadratic in n and, assuming $O(f_1) = O(f_2) = O(c) = O(d)$, also quadratic in the feature/class space.

We remark that, while enjoying a wider applicability due to being able to handle graphs with edge weights unrestricted in sign), SigMaNet features half the weights of MagNet.

Numerical Experiments

In this section, we report on a set of computational experiments carried out on four tasks: *link sign prediction*, *link existence prediction*, *link direction prediction*, and *node classification*. The experiments are conducted to assess the performance of SigMaNet on graphs with weights unrestricted in sign on which no other spectral GCNs can be applied (link sign prediction) and to compare it to other state-of-the-art spectral and spatial approaches on graphs with nonnegative weights (link existence/direction prediction and node classification). The code is available on GitHub.[2]

For the link sign prediction task, we compare SigMaNet with three categories of methods: *i)* signed network embedding: SiNE (Wang et al. 2017), SIGNet (Islam, Aditya Prakash, and Ramakrishnan 2018), BESIDE (Chen et al. 2018); *ii)* Feature Engineering: FeExtra (Leskovec, Huttenlocher, and Kleinberg 2010a); and *iii)* signed graph neural networks: SGCN (Derr, Ma, and Tang 2018), SiGAT (Huang et al. 2019), and SDGNN (Huang et al. 2021). For the link prediction and node classification tasks, we compare SigMaNet with the following three categories of methods: *i)* spectral methods designed for undirected graph: ChebNet (Defferrard, Bresson, and Vandergheynst 2016), GCN (Kipf and Welling 2017); *ii)* spectral methods designed for directed graphs: DGCN (Tong et al. 2020b), DiGraph (Tong et al. 2020a), DiGCL (Tong et al. 2021), and MagNet (Zhang et al. 2021b), and *iii)* spatial methods: APPNP (Klicpera, Bojchevski, and Günnemann 2019), SAGE (Hamilton, Ying, and Leskovec 2017), GIN (Xu et al. 2018), GAT (Veličković et al. 2017), and SSSNET (He et al. 2022b).[3]

Throughout this section's tables, the best results are reported in **boldface** and the second best are underlined.

Datasets

We test SigMaNet on six real-world datasets from the literature: `Bitcoin-OTC` and `Bitcoin Alpha` (Kumar et al. 2016); `Slashdot` and `Epinions` (Leskovec, Huttenlocher, and Kleinberg 2010b); `WikiRfa` (West et al. 2014); and `Telegram` (Bovet and Grindrod 2020). In order to better assess SigMaNet's performance as the density of the graph increases, in three tasks we also consider a synthetic set of graphs generated via a direct stochastic block model (DSBM) with (unlike what is done in (Zhang et al. 2021b)) edge weights greater than 1. These datasets are generated by varying: *i)* the number of nodes n; *ii)* the number of clusters C; *iii)* the probability α_{ij} to create an undirected edge between nodes i and j belonging to different clusters ; *iv)* the probability α_{ii} to create an undirected edge between two nodes in the same cluster, and *v)* the probability β_{ij} of an edge taking a certain direction. Each node is labeled with the index of the cluster it belongs to.

Link Sign Prediction

The link sign prediction task is a classification problem designed for graphs with both positive and negative edge weights. It consists of predicting the sign of the edges in the graph and, thus, for such task SigMaNet is the only spectral-based GCN that can be used.

For this task, we adopt the `Bitcoin Alpha`, `Bitcoin-OTC`, `WikiRfa`, `Slashdot`, and `Epinions` datasets, which are directed graphs with weights of unrestricted sign (necessary for the task to be applicable) and of arbitrary magnitude, with the sole exception of the last two, whose weights satisfy $A \in \{-1, 0, +1\}^{n \times n}$. In these five datasets, the classes of positive and negative weighted edges are imbalanced (i.e., nearly 80% are positive edges). The experiments are run with k-cross validation with $k = 5$, reporting the average score obtained across the k splits. Connectivity is maintained when building each training set by guaranteeing that the graph used for training in each fold contain a spanning tree. Following (Huang et al. 2021), we we adopt a 80%-20% training-testing split.

The results are reported in Table 1.[4] We observe that SigMaNet clearly outperforms all competitors on the three datasets whose graphs have unrestricted weights, i.e., `Bitcoin Alpha`, `Bitcoin-OTC`, and `WikiRfa`. On graphs with unit weights, i.e., `Slashdot` and `Epinions`, its performance, while marginally worse, is still in line with the best methods. This suggests the relevance of Corollary 1 towards SigMaNet's performance. We remark that the latter is achieved in spite of SigMaNet being less complex than the deep neural networks we compared it to here, which feature two sequentially-applied neural networks (one producing a set of embeddings from which the other one predicts the link sign via a logistics regression).

Link (Existence and Direction) Prediction

We consider the *existence prediction* task of predicting whether $(u, v) \in E$ for a pair of vertices $u, v \in V, u \neq v$ provided as input and the *direction prediction* task of predicting whether *a)* $(u, v) \in E$ or *b)* $(v, u) \in E$ or both.

[2] https://github.com/Stefa1994/SigMaNet.

[3] (He et al. 2022a) (appeared after the submission of this paper) reports experiments in which SigMaNet seems to perform poorly. In them, though, SigMaNet is used in a much simplied configuration than in our paper that only features four convolutional filters, whereas, in this work, the number of filters is chosen from $\{16, 32, 64\}$ via hyperameter optimization.

[4] Except for SigMaNet, the results are taken from (Huang et al. 2021). For SGCN, SiGAT, and SDGNN, we chose to report the results in (Huang et al. 2021) rather than those in (He et al. 2022c) as the former are better, and, thus, more challenging for SigMaNet.

		Signed Network Embedding			Feature Engineering	Graph Neural Network			
Dataset	Metric (%)	SiNE	SIGNet	BESIDE	FeExtra	SGCN	SiGAT	SDGNN	SigMaNet
Bitcoin Alpha	Micro-F1	94.58	94.22	94.89	94.86	92.56	94.56	94.91	**95.13**
	Binary-F1	97.16	96.96	97.32	97.30	96.07	97.14	97.29	**97.44**
	Macro-F1	68.69	69.65	73.00	71.67	63.67	70.26	73.90	**74.69**
	AUC	87.28	89.08	89.81	88.82	84.69	88.72	89.88	**92.46**
Bitcoin-OTC	Micro-F1	90.95	92.29	93.20	93.61	90.78	92.68	93.57	**94.49**
	Binary-F1	95.10	95.81	96.28	96.53	94.91	96.02	96.47	**97.02**
	Macro-F1	68.05	73.86	78.43	78.26	73.06	75.33	80.17	**80.53**
	AUC	85.71	89.35	91.52	91.21	87.55	90.55	91.24	**93.67**
WikiRfa	Micro-F1	83.38	83.84	85.89	83.46	84.89	84.57	86.27	**86.56**
	Binary-F1	89.72	90.01	91.17	89.87	90.69	90.42	91.42	**91.64**
	Macro-F1	73.19	73.84	78.03	72.35	75.27	75.35	78.49	**78.66**
	AUC	86.02	86.82	89.81	86.04	85.63	88.29	88.98	**90.53**
Slashdot	Micro-F1	82.65	83.89	85.90	84.72	82.96	84.94	**86.16**	85.03
	Binary-F1	89.18	89.83	91.05	90.70	89.26	90.55	**91.28**	90.59
	Macro-F1	72.73	75.54	**78.92**	73.99	74.03	76.71	**78.92**	77.63
	AUC	84.09	87.52	**90.17**	88.80	85.34	88.74	89.77	89.79
Epinions	Micro-F1	91.73	91.13	93.36	92.26	91.12	92.93	**93.55**	92.25
	Binary-F1	95.25	94.89	96.15	95.61	94.86	95.93	**96.28**	95.51
	Macro-F1	81.60	80.60	86.01	81.30	81.05	84.54	**86.10**	83.41
	AUC	88.72	90.95	93.51	94.17	87.45	93.33	94.11	**94.19**

Table 1: Link sign prediction results assessed with four metrics

For both tasks, we only consider graphs with nonnegative edge weights in order to compare SigMaNet not just to spatial GCNs as done before, but also to state-of-the-art spectral-based ones. As such GCNs are designed solely for graphs with nonnegative weights, one may expect that the wider applicability of SigMaNet should come at the price of inferior performances. Our experiments show that this is not the case.

We consider the `Telegram`, `Bitcoin Alpha`, `Bitcoin-OTC` datasets and synthetic DBSM graphs. The latter are generated with $n = 2500$, $C = 5$, $\alpha_{ii} = 0.1$, $\beta_{ij} = 0.2$, with inter-cluster density $\alpha_{ij} \in \{0.05, 0.08, 0.1\}$.[5] Following Zhang et al. (2021b), in each task we reserve 15% of the edges for testing, 5% for validation, and use the remaining ones for training. The experiments are run with k-cross validation with $k = 10$.

Tables 2 and 3 report the results obtained for the existence and direction prediction tasks, respectively. The tables show that, when compared to the other 10 methods, SigMaNet achieves the best performance on 9 datasets out of 12 and that it achieves either the first- or the second-best performance on 12 datasets out of 12. SigMaNet is also consistently better than the state of the art on the synthetic datasets. This is likely due to the positive homogeneity property (Theorem 4) as the synthetic datasets have a significantly wider range of weights ($[2, 1000]$) than the real-world ones (in `Telegram`, for instance, the mean and median weight is 2 and 20.7 and only 96.4% of the weights are smaller than 100).

[5]As spectral methods, except for SigMaNet, cannot handle graphs with negative weights, to be able to compare our proposal to them in a setting in which the latter can be applied, in these experiments we pre-process `Bitcoin-OTC` and `Bitcoin Alpha` by removing any edge with a negative weight—in the tables, these datasets are denoted by a '*'.

Node Classification

The node classification task consists in predicting the class label to which each node belongs. We only consider graphs with nonnegative edge weights so as to comparing SigMaNet not just to spatial GCNs but also to spectral-based ones. Also for this task we will show that the wider applicability of SigMaNet does not hinder its performances.

We consider the `Telegram` dataset as well as the three synthetic datasets. `Bitcoin-OTC` and `Bitcoin Alpha` dataset are not considered as they lack label information.[6] We rely on the standard 60%/20%/20% split for training/validation/testing across all datasets. The experiments are run with k-cross validation, with $k = 10$.

The results are reported in Table 4. SigMaNet achieves notable performance on all four datasets, especially on the synthetic ones as the graph density increases, where being able to rely on both edge direction and weight information seems to be paramount for a correct node labeling. This is confirmed by the extremely poor performance of ChebNet and GCN, which ignore the edge direction. When comparing SigMaNet to MagNet, SigMaNet achieves a consistently better performance of about 10% on average. This can be ascribed to the positive homogeneity property of SigMaNet, which circumvents the sign-pattern inconsistency MagNet suffers from (we recall that all these graphs have weights larger than 1), which is likely to reduce its ability to adequately propagate the information between nodes. We note that, while SSSNET outperforms SigMaNet once by about 4%, SigMaNet outperforms SSSNET by about 50% on `Telegram`. SSSNET's poor performance on this dataset is likely due to the lack of

[6]Differently from (Zhang et al. 2021b), where `Telegram` is preprocessed in such a way that it is transformed into an unweighted graph, in our experiments we retain its original weights.

	Existence prediction					
	Telegram	Bitcoin Alpha*	Bitcoin-OTC*	$\alpha_{ij} = 0.05$	$\alpha_{ij} = 0.08$	$\alpha_{ij} = 0.1$
ChebNet	75.30±1.54	81.93±0.64	82.07±0.38	50.24±0.35	50.21±0.33	50.25±0.34
GCN	67.88±1.39	81.53±0.57	81.65±0.35	50.26±0.30	50.24±0.26	50.18±0.26
APPNP	68.52±5.76	81.62±0.57	81.02±0.51	60.62±0.46	62.61±0.64	63.51±1.93
SAGE	85.36±1.27	82.74±0.48	83.28±0.65	60.92±0.82	61.50±4.05	62.77±1.50
GIN	72.37±3.57	74.64±5.43	77.75±1.15	57.52±4.47	55.50±5.14	55.25±7.14
GAT	78.37±2.11	82.60±0.43	83.43±0.52	55.97±2.58	54.37±0.89	50.24±0.35
DGCN	82.97±2.06	83.13±0.61	83.79±0.36	55.41±3.09	55.70±5.71	56.15±5.65
DiGraph	82.15±1.11	83.24±0.38	**84.77±0.83**	59.09±3.66	57.64±2.35	58.66±3.28
DiGCL	78.80±1.50	80.22±0.77	81.99±0.62	60.69±0.27	60.63±0.18	60.49±0.15
MagNet	**86.32±1.06**	83.26±0.50	84.14±0.44	61.27±0.19	63.81±0.20	64.93±0.43
SigMaNet	84.95±0.95	**83.28±0.54**	84.71±0.39	**62.25±0.31**	**64.48±0.17**	**65.49±0.31**

Table 2: Accuracy (%) on datasets of the existence prediction task

	Direction prediction					
	Telegram	Bitcoin Alpha*	Bitcoin-OTC*	$\alpha_{ij} = 0.05$	$\alpha_{ij} = 0.08$	$\alpha_{ij} = 0.1$
ChebNet	78.56±3.53	53.86±1.15	50.06±1.04	50.13±0.30	50.23±0.25	50.13±0.30
GCN	63.86±1.40	55.32±1.12	49.63±1.82	50.05±0.15	50.24±0.29	50.13±0.30
APPNP	75.70±9.08	57.14±1.03	52.61±1.63	66.42±1.35	70.25±1.46	71.93±0.47
SAGE	91.15±0.77	55.82±1.60	55.29±1.23	66.62±1.72	68.84±2.38	69.43±6.79
GIN	80.77±5.01	56.04±1.42	53.31±1.58	60.51±6.88	60.87±9.50	57.66±9.04
GAT	84.06±11.17	55.20±1.06	53.23±0.63	52.71±1.53	57.07±1.50	57.43±1.07
DGCN	89.81±1.20	56.35±0.84	54.06±0.90	55.97±2.58	62.64±6.91	65.53±6.73
DiGraph	87.46±0.84	**58.62±1.09**	56.37±1.29	65.51±1.71	67.09±1.65	67.43±2.10
DiGCL	82.98±1.72	55.98±0.91	56.42±0.59	67.34±0.33	66.92±0.26	66.24±0.29
MagNet	**91.65±0.79**	56.84±0.74	55.63±0.74	68.50±0.23	72.01±0.33	73.28±0.37
SigMaNet	91.20±0.65	56.90±0.60	**57.19±0.58**	**69.10±0.18**	**72.74±0.23**	**73.77±0.18**

Table 3: Accuracy (%) on datasets of the direction prediction task

	Node classification			
	Telegram	α_{ij} = 0.05	α_{ij} = 0.08	α_{ij} = 0.1
ChebNet	61.73±4.25	20.06±0.18	20.50±0.77	19.98±0.06
GCN	60.77±3.67	20.06±0.18	20.02±0.06	20.01±0.01
APPNP	55.19±6.26	33.46±7.43	34.72±14.98	36.16±14.92
SAGE	65.38±5.15	67.64±9.81	68.28±10.92	82.96±10.98
GIN	72.69±4.62	28.46±8.01	20.12±0.20	20.98±8.28
GAT	72.31±3.01	22.34±3.13	21.90±2.89	21.58±1.80
SSSNET	24.04±9.29	**91.04±3.60**	94.94±1.01	96.77±0.80
DGCN	71.15±6.32	30.02±6.57	30.22±11.94	28.40±8.62
DiGraph	71.16±5.57	53.84±14.28	38.50±12.20	34.78±9.94
DiGCL	64.62±4.50	19.51±1.21	20.24±0.84	19.98±0.45
MagNet	55.96±3.59	78.64±1.29	87.52±1.30	91.58±1.04
SigMaNet	**74.23±5.24**	87.44±0.99	**96.14±0.64**	**98.60±0.31**

Table 4: Testing accuracy (%) of node classification.

seed nodes that SSSNET needs to identify and target node classes (which are present in the DSBM graphs).

Conclusions

We have extended the applicability of spectral GCNs to (directed) graphs with edges of weight unrestricted in sign by introducing the *Sign-Magnetic Laplacian* matrix. Thanks to its properties, which we rigorously derived, we embedded the matrix into a generalized convolution operator which is the cornerstone of our proposed spectral GCN: SigMaNet, which is first spectral GCN capable of handling (directed) graphs with weights not restricted in sign nor magnitude. Compared with similar approaches presented in the literature, SigMaNet also does not suffer from any sign-pattern inconsistencies, making it capable to handle graphs with arbitrarily large weights (also in a completely parameter-free way and without preprocessing). Thanks to extensive numerical experiments, we have shown that, on graphs with negative weights where no other spectral GCN can be applied, SigMaNet's performance is either better or in line with more complex architectures, and that, on graphs with nonnegative weights where state-of-the-art spectral GCNs can be employed, SigMaNet's performance is consistently either the best or the second-best across all tasks.

Acknowledgements

This work has been partially supported by the project "ULTRA OPTYMAL - Urban Logistics and susTainable tRAnsportation: OPtimization under uncertainTY and MAchine Learning" funded by the MIUR Progetti di Ricerca di Rilevante Interesse Nazionale (PRIN) Bando 2020 - grant 20207C8T9M, the European Union's Horizon Europe research and innovation programme under grant agreement No 101070284 - enRichMyData, and the Alan Turing Institute under the EPSRC grants EP/W001381/1 and EP/N510129/1.

References

Backstrom, L.; and Leskovec, J. 2011. Supervised random walks: predicting and recommending links in social networks. In *Proceedings of the fourth ACM international conference on Web search and data mining*, 635–644.

Bovet, A.; and Grindrod, P. 2020. The activity of the far right on Telegram. *ResearchGate preprint*, DOI: 10.13140/RG.2.2.16700.05764: 1–19.

Chen, S.; Varma, R.; Sandryhaila, A.; and Kovačević, J. 2015. Discrete Signal Processing on Graphs: Sampling Theory. *IEEE transactions on signal processing*, 63(24): 6510–6523.

Chen, Y.; Qian, T.; Liu, H.; and Sun, K. 2018. "Bridge": Enhanced Signed Directed Network Embedding. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 773–782.

Chung, F. R.; and Graham, F. C. 1997. *Spectral graph theory*. 92. American Mathematical Soc.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29.

Derr, T.; Ma, Y.; and Tang, J. 2018. Signed graph convolutional networks. In *2018 IEEE International Conference on Data Mining (ICDM)*, 929–934. IEEE.

Fanuel, M.; Alaíz, C. M.; Fernández, Á.; and Suykens, J. A. 2018. Magnetic eigenmaps for the visualization of directed networks. *Applied and Computational Harmonic Analysis*, 44(1): 189–199.

Fanuel, M.; Alaíz, C. M.; and Suykens, J. A. K. 2017. Magnetic eigenmaps for community detection in directed networks. *Physical Review E*, 95(2).

Furutani, S.; Shibahara, T.; Akiyama, M.; Hato, K.; and Aida, M. 2019. Graph signal processing for directed graphs based on the Hermitian Laplacian. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 447–463. Springer.

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30: 1–11.

Hammond, D. K.; Vandergheynst, P.; and Gribonval, R. 2011. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2): 129–150.

He, Y.; Perlmutter, M.; Reinert, G.; and Cucuringu, M. 2022a. MSGNN: A Spectral Graph Neural Network Based on a Novel Magnetic Signed Laplacian. In *Proceedings of the 1st Learning on Graphs (LoG) conference*, 1–13.

He, Y.; Reinert, G.; Wang, S.; and Cucuringu, M. 2022b. SSSNET: Semi-Supervised Signed Network Clustering. In *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*, 244–252. SIAM.

He, Y.; Zhang, X.; Huang, J.; Cucuringu, M.; and Reinert, G. 2022c. PyTorch Geometric Signed Directed: A Survey and Software on Graph Neural Networks for Signed and Directed Graphs. *arXiv preprint arXiv:2202.10793*.

Huang, J.; Shen, H.; Hou, L.; and Cheng, X. 2019. Signed graph attention networks. In *International Conference on Artificial Neural Networks*, 566–577. Springer.

Huang, J.; Shen, H.; Hou, L.; and Cheng, X. 2021. SDGNN: Learning Node Representation for Signed Directed Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(1): 196–203.

Islam, M. R.; Aditya Prakash, B.; and Ramakrishnan, N. 2018. SIGNet: Scalable Embeddings for Signed Networks. In Phung, D.; Tseng, V. S.; Webb, G. I.; Ho, B.; Ganji, M.; and Rashidi, L., eds., *Advances in Knowledge Discovery and Data Mining*, 157–169. Springer International Publishing.

Kashyap, S.; Kumar, S.; Agarwal, V.; Misra, D. P.; Phadke, S. R.; and Kapoor, A. 2018. Protein protein interaction network analysis of differentially expressed genes to understand involved biological processes in coronary artery disease and its different severity. *Gene Reports*, 12: 50–60.

Kipf, T. N.; and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*.

Klicpera, J.; Bojchevski, A.; and Günnemann, S. 2019. Predict then propagate: Graph neural networks meet personalized pagerank. In *Proceedings of the 7th International Conference on Learning Representations*, 1–15.

Kumar, S.; Spezzano, F.; Subrahmanian, V. S.; and Faloutsos, C. 2016. Edge Weight Prediction in Weighted Signed Networks. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, 221–230.

Leskovec, J.; Huttenlocher, D.; and Kleinberg, J. 2010a. Predicting positive and negative links in online social networks. In *Proceedings of the 19th international conference on World wide web*, 641–650.

Leskovec, J.; Huttenlocher, D.; and Kleinberg, J. 2010b. Signed networks in social media. In *Proceedings of the SIGCHI conference on human factors in computing systems*, 1361–1370.

Lieb, E. H.; and Loss, M. 1993. Fluxes, Laplacians, and Kasteleyn's theorem. In *Statistical Mechanics*, 457–483. Springer.

Sandryhaila, A.; and Moura, J. M. 2013. Discrete signal processing on graphs. *IEEE transactions on signal processing*, 61(7): 1644–1656.

Tong, Z.; Liang, Y.; Ding, H.; Dai, Y.; Li, X.; and Wang, C. 2021. Directed graph contrastive learning. *Advances in Neural Information Processing Systems*, 34: 19580–19593.

Tong, Z.; Liang, Y.; Sun, C.; Li, X.; Rosenblum, D. S.; and Lim, A. 2020a. Digraph inception convolutional networks. *Advances in Neural Information Processing Systems*, 2020-December(NeurIPS): 1–12.

Tong, Z.; Liang, Y.; Sun, C.; Rosenblum, D. S.; and Lim, A. 2020b. Directed Graph Convolutional Network. arXiv:2004.13970.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. In *Proceedings of the 5th International Conference on Learning Representations*, 1–12.

Wang, S.; Tang, J.; Aggarwal, C.; Chang, Y.; and Liu, H. 2017. Signed network embedding in social media. In *Proceedings of the 2017 SIAM international conference on data mining*, 327–335. SIAM.

Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S. E.; Bronstein, M. M.; and Solomon, J. M. 2019. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5): 1–12.

West, R.; Paskov, H. S.; Leskovec, J.; and Potts, C. 2014. Exploiting social network structure for person-to-person sentiment analysis. *Transactions of the Association for Computational Linguistics*, 2: 297–310.

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Philip, S. Y. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1): 4–24.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2018. How Powerful are Graph Neural Networks? In *Proceedings of the 6th International Conference on Learning Representations*, 1–17.

Zhang, J.; Hui, B.; Harn, P.-W.; Sun, M.-T.; and Ku, W.-S. 2021a. sMGC: A Complex-Valued Graph Convolutional Network via Magnetic Laplacian for Directed Graphs. arXiv:2110.07570.

Zhang, S.; Tong, H.; Xu, J.; and Maciejewski, R. 2019. Graph convolutional networks: a comprehensive review. *Computational Social Networks*, 6(1): 1–23.

Zhang, X.; He, Y.; Brugnone, N.; Perlmutter, M.; and Hirn, M. 2021b. Magnet: A neural network for directed graphs. *Advances in Neural Information Processing Systems*, 34: 27003–27015.

Zou, X. 2020. A survey on application of knowledge graph. In *Journal of Physics: Conference Series*, volume 1487, 012016. IOP Publishing.