ASWT-SGNN: Adaptive Spectral Wavelet Transform-Based Self-Supervised Graph Neural Network

Ruyue Liu[1,2], **Rong Yin**[1,2*], **Yong Liu**[3], **Weiping Wang**[1]
[1]Institute of Information Engineering, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences
[3]Renmin University of China
{liuruyue, yinrong, wangweiping}@iie.ac.cn, liuyonggsai@ruc.edu.cn

Abstract

Graph Comparative Learning (GCL) is a self-supervised method that combines the advantages of Graph Convolutional Networks (GCNs) and comparative learning, making it promising for learning node representations. However, the GCN encoders used in these methods rely on the Fourier transform to learn fixed graph representations, which is inherently limited by the uncertainty principle involving spatial and spectral localization trade-offs. To overcome the inflexibility of existing methods and the computationally expensive eigen-decomposition and dense matrix multiplication, this paper proposes an Adaptive Spectral Wavelet Transform-based Self-Supervised Graph Neural Network (ASWT-SGNN). The proposed method employs spectral adaptive polynomials to approximate the filter function and optimize the wavelet using contrast loss. This design enables the creation of local filters in both spectral and spatial domains, allowing flexible aggregation of neighborhood information at various scales and facilitating controlled transformation between local and global information. Compared to existing methods, the proposed approach reduces computational complexity and addresses the limitation of graph convolutional neural networks, which are constrained by graph size and lack flexible control over the neighborhood aspect. Extensive experiments on eight benchmark datasets demonstrate that ASWT-SGNN accurately approximates the filter function in high-density spectral regions, avoiding costly eigen-decomposition. Furthermore, ASWT-SGNN achieves comparable performance to state-of-the-art models in node classification tasks.

Introduction

Graphs are essential in various real-world domains such as social networks, brain networks, transportation networks, and citation networks (Wang et al. 2016; Yu, Lee, and Sohn 2020). Recently, the emergence of graph neural networks (GNNs) (He et al. 2022; Wang et al. 2022b) has attracted much attention due to their success in applications involving graph-structured data such as node classification (Kipf and Welling 2016) and edge prediction (Hasanzadeh et al. 2019). Graph Comparative Learning (GCL) (Velickovic et al. 2019; Sun et al. 2019) combines the capabilities of GNN and comparative learning techniques, making it a promising paradigm in the field of graph analysis (He et al. 2020). Typically, GCL methods generate multiple views by randomly augmenting input data and optimize the GNN encoder by learning consistency across views. GCL reduces the dependence of graph representation learning on human annotations and achieves state-of-the-art performance in tasks such as node classification.

Most GCL methods utilize graph convolutional neural networks (GCNs) as encoders (Xie et al. 2022; Wang et al. 2022b). Similar to convolutional neural networks (CNNs) in computer vision, spectral GCNs use Fourier bases in the design of graph-based operators (Bruna et al. 2013). However, these operators are localized in the frequency rather than the spatial domain. Additionally, they require costly multiplications between eigen-decomposition and dense matrices, leading to high computational expenses. In order to address this issue and achieve spatial localization, methods such as ChebyNet (Defferrard, Bresson, and Vandergheynst 2016) and GCN (Kipf and Welling 2016) employ polynomial approximation. While GCN is widely adopted for graph problems due to its impressive performance and computational efficiency (Shi et al. 2020), it encounters limitations and challenges when applied to large graphs, especially in mini-batch settings (Zeng et al. 2019b). To overcome the scaling challenges of GCN on large graphs, researchers have proposed layer sampling methods (Ying et al. 2018; Kaler et al. 2022) and subgraph sampling methods (Zeng et al. 2019b,a). However, the filter size is determined by the size of the entire graph or the sampled subgraph, which restricts flexibility for inputs of different sizes. Although some flexible spatial methods have been proposed, their aggregators lack learnability and convolutional properties. Consequently, the problem of designing a flexible filter that combines the learnability of spatial methods and the convolutional properties of spectral methods still needs to be solved.

To address the issues of inflexible and unlearnable filters, as well as the limited applicability caused by high computational complexity in existing methods, this paper proposes a novel graph comparative learning paradigm based on the adaptive spectral wavelet transform. More specifically, a fast spectral adaptive approximation method is utilized to estimate the wavelet filter, and contrast loss is employed to optimize the wavelet scale directly. Additionally, the intro-

*Corresponding author.

duction of residual links mitigates over-smoothing during information aggregation. By avoiding the expensive eigen-decomposition of the graph Laplacian operator and enabling localization in both the spectral and spatial domains, this approach effectively overcomes the limitations of graph convolutional neural networks that are constrained by graph size and lack flexibility in controlling neighborhood aspects. In comparison to state-of-the-art methods, this work introduces the following innovations:

- We propose a novel self-supervised graph representation learning method based on sparse graph wavelets that creates localized filters in both the spectral and spatial domains. It reduces the computational complexity and addresses the limitation that graph convolutional neural networks cannot flexibly control the neighborhood aspect.
- Theoretically, we demonstrate that nodes with similar network neighborhoods and features exhibit similar ASWT-SGNN embeddings, providing a performance guarantee for the proposed method.
- Extensive experiments on eight benchmark datasets show that the proposed method reduces approximation errors in high-density spectral components without requiring expensive eigen-decomposition and achieves competitive performance with state-of-the-art models in node classification tasks.

Related Works

Graph Convolutional Neural Network

Following the success of CNNs in computer vision and natural language processing, researchers have sought to extend CNNs to the graph domain. The key challenge lies in defining the convolution operator for graphs. Graph convolutional neural networks can be broadly categorized into two main approaches: spectral and spatial. Spectral methods employ the graph Fourier transform to transfer signals from the spatial domain to the spectral domain, where convolution operations are performed. Spectral GNN (Bruna et al. 2013) is the first attempt to implement CNNs on graphs. ChebyNet (Defferrard, Bresson, and Vandergheynst 2016) introduce a parameterization method using Chebyshev polynomials for spectral filters, which enables fast localization of spectral filters. GCN (Kipf and Welling 2016) proposes a simplified version of ChebyNet, which achieves success in graph semi-supervised learning tasks. However, these spectral methods face challenges with generalization, as they are limited by fixed graph sizes, and larger filter sizes result in increased computational and memory costs. Spatial methods draw inspiration from weighted summation in CNNs to extract spatial features from topological graphs. However, these methods employ unlearnable aggregators, and the localization of the convolution operation remains uncertain.

Graph Contrastive Learning

In graph analysis, contrastive learning was initially introduced by DGI (Velickovic et al. 2019) and InfoGraph (Sun et al. 2019), drawing inspiration from maximizing local-global mutual information. Building upon this, MVGRL (Hassani and Khasahmadi 2020) incorporate node diffusion into the graph comparison framework. GCA (Zhu et al. 2021) learns node representations by considering other nodes as negative samples, while BGRL (Thakoor et al. 2021) proposes a no-negative-sample model. CCA-SSG (Zhang et al. 2021) optimizes feature-level objectives in addition to instance-level differences. GRADE (Wang et al. 2022b) investigates fairness differences in comparative learning and proposes a novel approach to graph enhancement. Several surveys (Xie et al. 2022; Ding et al. 2022; Kumar, Rawat, and Chauhan 2022) summarize recent advancements in graph contrastive learning. Despite these methods' notable achievements, most rely on GCN and its variants as the base models, inheriting the limitations of GCN. These limitations restrict the performance of these graph contrastive learning methods in tasks that require preserving fine-grained node features.

Graph Wavelets

The wavelet transform exhibits favorable structural properties by utilizing finite length and attenuation basis functions. This approach effectively localizes signals within both the spatial and spectral domains. Additionally, it is noteworthy that the basis and its inverse in the wavelet transform often showcase sparsity, contributing to its utility and efficiency. To construct the wavelet transform on graphs, Hammond et al. (Hammond, Vandergheynst, and Gribonval 2011) propose a method that approximates the wavelet using Chebyshev polynomials. This approach effectively avoids the need for eigen-decomposition of the Laplace matrix. Building upon this, GWNN (Xu et al. 2019) redefines graph convolution based on graph wavelets, resulting in high efficiency and sparsity. M-GWCN (Behmanesh et al. 2022) applies the multi-scale graph wavelet transform to learn representations of multimodal data. Collectively, these works showcase the value of graph wavelets in signal processing on graphs. However, these methods primarily employ wavelet transforms in supervised or semi-supervised tasks and heavily rely on labeled data.

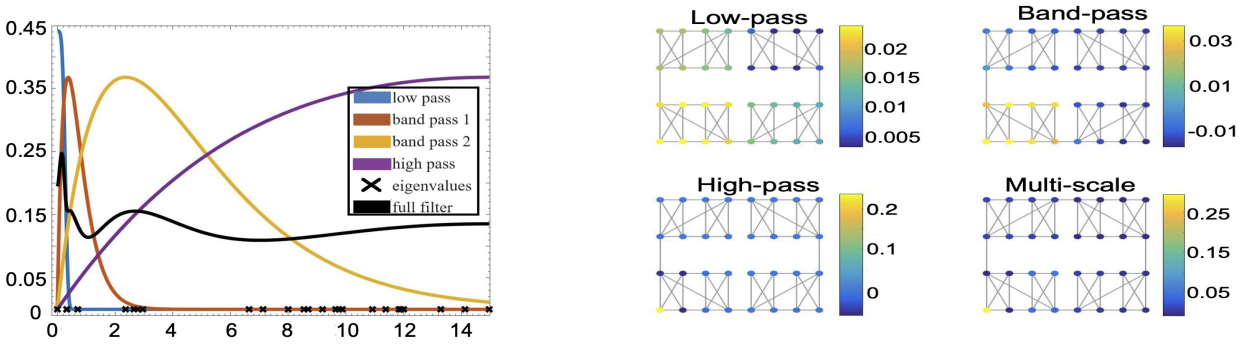
Preliminary

Graph Fourier Transform

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denotes a graph, where $\mathcal{V} = \{v_1, \cdots, v_N\}$ represents the set of nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ represents the set of edges. $\mathcal{G}$ is associated with a feature matrix $\boldsymbol{X} \in \mathbb{R}^{N \times p}$, $\boldsymbol{X} = [x_1, \ldots, x_n]$ and an adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$, where $\boldsymbol{A}_{ij} = 1$ if $(v_i, v_j) \in \mathcal{E}$ and $\boldsymbol{A}_{ij} = 0$ otherwise. We define the Laplacian matrix of the graph as $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$, where $\boldsymbol{D} = Diag(d_1, \ldots, d_N)$, $d_i = \sum_j \boldsymbol{A}_{ij}$. The symmetric normalized Laplacian is defined as $\boldsymbol{L}_{sym} = \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{L} \boldsymbol{D}^{-\frac{1}{2}} = \boldsymbol{U} \boldsymbol{\Lambda} \boldsymbol{U}^{\top}$. $\boldsymbol{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_N]$, where $\boldsymbol{u}_i \in \mathbb{R}^N$ denotes the i-th eigenvector of $\boldsymbol{L}_{sym}$ and $\boldsymbol{\Lambda} = Diag(\lambda_1, \ldots, \lambda_N)$ is the eigenvalue matrix.

For a discrete graph signal $\boldsymbol{X}$, its Fourier transform has the following form (Shuman et al. 2013):

$$\hat{f}(\lambda_\ell) = \sum_{i=1}^{N} \boldsymbol{x}_i \boldsymbol{U}_\ell^*(i) = \sum_{i=1}^{N} \boldsymbol{x}_i \boldsymbol{U}_\ell^T(i), \tag{1}$$



(a) Signal spectrum

(b) Filter signal

Figure 1: Visualize how wavelet filters can capture multiscale properties in graph signals and structures. We use graphs with two levels of clusters (4-node clusters and 8-node clusters) for demonstration. These clusters are reflected in the gaps (about 0 and 6) in the spectrum in Figure 1a, reflecting different eigenvalue clustering. The signal is obtained by filtering a random signal with the filter in Figure 1a, purposefully highlighting the three eigenvalue clusters. Figure 1b shows how the complete signal is decomposed into multiple filter components.

where $\boldsymbol{U}_\ell^*$ denotes the conjugate transposition of the eigenvalues λ_ℓ corresponding to the eigenvectors. Since the complex number is not involved in the scope of this work, it can be regarded as a common transposition, that is, $\boldsymbol{U}^* = \boldsymbol{U}^T$.

Graph Fourier transform provides a means to define the graph convolution operator using the convolution theorem. The filter signal can be mathematically expressed as:

$$f_o = \boldsymbol{U} g(\boldsymbol{\Lambda}) \boldsymbol{U}^T \boldsymbol{X}, \tag{2}$$

where $g(\boldsymbol{\Lambda})$ denotes the filter function on the eigenvalues, and $\boldsymbol{U} g(\boldsymbol{\Lambda}) \boldsymbol{U}^T$ is the graph filtering matrix.

Uncertainty Principle

The implementation of graph convolution using the Fourier transform lacks spatial localization despite its ability to achieve localization in the spectral domain. Additionally, its localization is significantly influenced by computational efficiency. We employ the spatial and spectral concentration metric proposed by Tsitsvero et al. (Tsitsvero, Barbarossa, and Di Lorenzo 2016) to establish a more comprehensive and precise notion of localization.

$$\frac{\|\boldsymbol{B}\boldsymbol{x}\|_2^2}{\|\boldsymbol{x}\|_2^2} = a^2, \frac{\|\boldsymbol{C}\boldsymbol{x}\|_2^2}{\|\boldsymbol{x}\|_2^2} = b^2, \tag{3}$$

where the square of the Euclidean paradigm $\|\boldsymbol{x}\|_2^2$ denotes the total energy of the signal $\boldsymbol{x}$. $\boldsymbol{B}$ is the diagonal matrix representing the node restriction operator. Given a node subset $\mathcal{S} \subseteq \mathcal{V}$, $\boldsymbol{B} = Diag\{\mathbb{I}(i \in \mathcal{S})\}$, where $\mathbb{I}(\cdot)$ is an indicator function. $\boldsymbol{C} = \boldsymbol{U}\Sigma_{\mathcal{F}}\boldsymbol{U}^{-1}$, is the band-limiting operator. Given a matrix $\boldsymbol{U}$ and a subset of frequency indices $\mathcal{F} \subseteq \mathcal{V}^*$, where $\mathcal{V}^* = \{1, \cdots, N\}$ denotes the set of all frequency indices. $\Sigma_{\mathcal{F}}$ is a diagonal matrix defined as $\Sigma_{\mathcal{F}} = Diag\{\mathbb{I}(i \in \mathcal{F})\}$.

More specifically, considering a pair of vertex sets $\mathcal{S}$ and frequency sets $\mathcal{F}$, where a^2 and b^2 represent the percentage of energy contained within the sets $\mathcal{S}$ and $\mathcal{F}$ respectively, our objective is to determine the balance between a and b and identify signals that can achieve all feasible pairs. The resulting uncertainty principle is formulated and presented in the theorem (Tsitsvero, Barbarossa, and Di Lorenzo 2016).

$$\cos^{-1} a + \cos^{-1} b \geq \cos^{-1} \lambda_{\max}(\boldsymbol{BC}), \tag{4}$$

where $\lambda_{max}(\boldsymbol{BC})$ is the maximum eigenvalue of $\boldsymbol{BC}$.

Considering the uncertainty principle, we retain the entire frequency band if we follow the graph localization method in Eq. (2). However, real graph signals show a non-uniform distribution in the frequency domain, which suggests that $\boldsymbol{C}$ can be chosen more efficiently. Moreover, a deeper network leads to a wider propagation of the signal in the graph domain, which limits the width of the frequency bands and leads to the dominance of low-frequency signals (smoothing signals), which produce over-smoothing.

Graph Wavelet Transform

As the graph Fourier transform, the graph wavelet transform also necessitates a set of suitable bases to map the graph signal to the spectral domain. In this case, we denote the wavelet operator as $\boldsymbol{\Psi}_s = \boldsymbol{U} g_s(\boldsymbol{\Lambda}) \boldsymbol{U}^\top$, where s is the scaling parameter. The wavelet transform breaks down a function g into a linear combination of basis functions localized in spatial and spectral. This paper employs the Heat Kernel wavelet as the low-pass filter (denoted as g^l). In contrast, the Mexican-Hat wavelet is the band-pass filter (denoted as g^b). These filters are defined as follows:

$$g_s^l(\lambda) = e^{-s\lambda}, \tag{5}$$

$$g_s^b(\lambda) = \frac{2}{\sqrt{3}\pi^{\frac{1}{4}}} \left(1 - (\lambda s)^2\right) e^{-\frac{(\lambda s)^2}{2}}. \tag{6}$$

In this work, we integrate filter functions to achieve the combined effect of a low-pass filter and a wide band-pass filter. Figure 1 exemplifies how the combination of low-pass and band-pass filters can generate a more intricate wavelet filter capable of capturing signal components at various

scales. This design enables us to obtain a richer representation of the signal, encompassing its frequency information more nuancedly.

$$g_\theta(\lambda) = g^l_{s_0}(\lambda) + \sum_{l=1}^{L} g^b_{s_l}(\lambda), \tag{7}$$

where $\theta = \{s_0, s_1, ..., s_l\}$ is the set of scale parameters.

Using the graph wavelet transform instead of the graph Fourier transform in Eq. (2), we get the graph convolution as follows:

$$f_o = \boldsymbol{\Psi} \boldsymbol{G} \boldsymbol{\Psi}^\top \boldsymbol{X}, \tag{8}$$

where $\boldsymbol{G}$ is a diagonal matrix, which acts as a filter, the scale set θ of wavelet coefficients is omitted for simplification.

Methodology

Wavelet Coefficients Approximation

The model formulation discussed in the preceding sections necessitates an eigen-decomposition of the Laplace operator of the input graph $\mathcal{G}$. However, this process presents a computational complexity of $\mathcal{O}(N^3)$, rendering it infeasible for larger graphs. In order to overcome this constraint, we employ the polynomial approximation method previously introduced by Hammond et al. (Hammond, Vandergheynst, and Gribonval 2011). This method involves expressing the wavelet filter as $g_\theta(\lambda) \approx p_\theta(\lambda) = \gamma_0 + \gamma_1\lambda + \cdots + \gamma_m\lambda^m$. This allows rewriting the wavelet operator as $\boldsymbol{\Psi_\theta} = \boldsymbol{U} p_\theta(\boldsymbol{\Lambda}) \boldsymbol{U}^\top = p_\theta(\boldsymbol{L}_{sym})$.

While existing methods rely on Chebyshev polynomial approximations, we aim to optimize scales. Therefore, we utilize a least squares approximation that can parameterize the set of wavelet scales θ, which can be expressed as follows:

$$\boldsymbol{\gamma}_\theta = (\boldsymbol{V}_{\boldsymbol{\Lambda}}^\top \boldsymbol{V}_{\boldsymbol{\Lambda}})^{-1} \boldsymbol{V}_{\boldsymbol{\Lambda}}^\top g_\theta(\boldsymbol{\Lambda}), \tag{9}$$

where $\boldsymbol{V}_{\boldsymbol{\Lambda}} \in \mathbb{R}^{N\times(m+1)}$ is the Vandermonde matrix of $\boldsymbol{\Lambda}$ from order 0 to order m, N is the number of eigenvalues.

Accurate calculation of the eigenvalues requires an expensive eigen-decomposition of the graph Laplace operator, which is not feasible in our case. As an alternative, we transform the set of eigenvalues into a sequence of linearly spaced points $\boldsymbol{\xi} = \{\xi_i\}_{i=1}^K$ within the interval $[0, 2]$ on the spectral domain. However, in graphs that exhibit multiscale features, the eigenvalues do not follow a uniform distribution on the spectral domain. Instead, they display spectral gaps corresponding to different scales within the data. To address this non-uniform distribution, we incorporate the estimated spectral density $\boldsymbol{\omega}$ as the weight for each of the K sample points $\boldsymbol{\xi}$ on the spectral domain (Fan et al. 2020). Consequently, we can compute the weighted least squares coefficients,

$$\boldsymbol{\gamma}_\theta = (\boldsymbol{V}_{\boldsymbol{\xi}}^\top Diag(\boldsymbol{\omega}) \boldsymbol{V}_{\boldsymbol{\xi}})^{-1} \boldsymbol{V}_{\boldsymbol{\xi}}^T Diag(\boldsymbol{\omega}) g_\theta(\boldsymbol{\xi}), \tag{10}$$

where the spectral density $\boldsymbol{\omega} = \{\omega_j\}_{j=1}^K$, and $\omega_i = \frac{1}{N}\sum_{j=1}^N \{\mathbb{I}(\lambda_j = \xi_i)\}$.

The goal of spectral density estimation is to approximate the density function without expensive graph Laplacian eigen-decomposition. To achieve this, we determine the number of eigenvalues less than or equal to each ξ_i in the set $\boldsymbol{\xi}$. It can be achieved by computing an approximation to the trace of the eigen-projection matrix $\boldsymbol{P}$ (Di Napoli, Polizzi, and Saad 2016). In practice, directly obtaining the projector $\boldsymbol{P}$ is often not feasible. However, it can be approximated efficiently using polynomials or rational functions of the Laplacian operator $\boldsymbol{L}$. In this approximation, we interpret $\boldsymbol{P}$ as a step function of $\boldsymbol{L}$, which can be expressed as follows:

$$\boldsymbol{P}_{\xi_i} = h(\boldsymbol{L}), \text{ where } h(\lambda) = \begin{cases} 1, & \text{if } \lambda \le \xi_i \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

Although it is impossible to compute $h(\lambda)$ exactly cheaply, it can be approximated using a finite sum of Jackson-Chebyshev polynomials, denoted as $\phi(\lambda)$. Please refer to Appendix A (Liu et al. 2023) for detailed information on the approximation method. In this form, it becomes possible to estimate the trace of $\boldsymbol{P}$ by an estimator developed by Hutchinson (Hutchinson 1989) and further improved more recently (Tang and Saad 2012). Hutchinson's stochastic estimator relies solely on matrix-vector products to approximate the matrix trajectory. The key idea is to utilize Rademacher random variables, where each entry of a randomly generated vector $\boldsymbol{R} \in \mathbb{R}^N$ takes on the values -1 and 1 with equal probability of $\frac{1}{2}$. Thus, an estimate of the trace $tr(\boldsymbol{P})$ can be obtained by generating n_r samples of random vectors $\boldsymbol{R}_k$, $k = 1, \cdots, n_r$ and computing the average over these samples. The estimator can be expressed as follows:

$$tr(\boldsymbol{P}) \approx \frac{1}{n_r}\sum_{k=1}^{n_r} \boldsymbol{R}_k^\top \boldsymbol{P} \boldsymbol{R}_k \approx \frac{1}{n_r}\sum_{k=1}^{n_r} \boldsymbol{R}_k^\top \phi(\boldsymbol{L}_{sym}) \boldsymbol{R}_k. \tag{12}$$

We obtain the approximation $\boldsymbol{\Omega}$ to the cumulative spectral density function.

$$\boldsymbol{\Omega} = \left\{ \left(\xi_i, \frac{1}{N} \left[\frac{1}{n_r}\sum_{k=1}^{n_r} \boldsymbol{R}_k^\top \phi(\boldsymbol{L}_{sym}) \boldsymbol{R}_k \right] \right) \right\}_{i=1}^{K}. \tag{13}$$

Finally, the cumulative spectral density $\boldsymbol{\Omega}$ is differentiated to obtain an approximation of the spectral density $\boldsymbol{\omega}$, ie $\boldsymbol{\omega} = \frac{d}{d\boldsymbol{\xi}}\boldsymbol{\Omega}$. Using the estimated spectral density $\boldsymbol{\omega}$ as a weight for each sample point ξ_i on the spectral domain, the weighted least squares coefficient $\boldsymbol{\gamma}_\theta$ can be calculated by substituting it into Eq. (10).

These coefficients are used to approximate the wavelet filter matrix $\boldsymbol{\Psi}_\theta$, which can be expressed as follows:

$$\boldsymbol{\Psi}_\theta = \gamma_0 \boldsymbol{I} + \gamma_1 \boldsymbol{L}_{sym} + \cdots + \gamma_m \boldsymbol{L}_{sym}^m. \tag{14}$$

Encoder

In this paper, we construct a graph wavelet multilayer convolutional network (ASWT-SGNN). Based on the above, we define the l-th layer of ASWT-SGNN as

$$\boldsymbol{H}^{l+1} = \sigma(\boldsymbol{H}^{l'} \boldsymbol{W}^l), \tag{15}$$

where σ is the activation function, $\boldsymbol{H}^{l'} \in \mathbb{R}^{N\times p}$ is the results of graph convolution of the l-th layer, $\boldsymbol{W}^l \in \mathbb{R}^{p\times q}$ is

the weight of the l-th layer, p is the number of features in current layer, and q is the number of features in next layer. $\boldsymbol{H}^{l'}$ in Eq. (15) is described as follows:

$$\boldsymbol{H}^{l'} = \alpha \boldsymbol{F}^{l} \boldsymbol{H}^{l} + (1-\alpha)\boldsymbol{H}^{l}, \tag{16}$$

where $1-\alpha$ represents the proportion of the original features $\boldsymbol{H}^{l}$ in l-th layer, and $0 \leq \alpha \leq 1$. By incorporating this residual connection, we guarantee that regardless of the number of layers we add, the resulting representation will always retain a portion of the initial features. To enhance the encoder's capability to aggregate both local and global information effectively, we define the diffusion operator $\boldsymbol{F}^{l}$ of l-th layer in Eq. (16) as

$$\boldsymbol{F}^{l} = \beta(\boldsymbol{\Psi}_{\theta} \boldsymbol{G}^{l} \boldsymbol{\Psi}_{\boldsymbol{\theta}}^{\top}) + (1-\beta)(\boldsymbol{D}^{-\frac{1}{2}} \tilde{\boldsymbol{A}} \boldsymbol{D}^{-\frac{1}{2}}), \tag{17}$$

where β stands for the ratio of the graph wavelet term and $0 \leq \beta \leq 1$. θ is the learnable multi-scale parameters. $\boldsymbol{G}^{l}$ is the learnable diagonal filter matrix of l-th layer. $\tilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}$ represents the adjacency matrix of the self-loop graph of $\mathcal{G}$, where $\boldsymbol{I}$ is the identity matrix.

Optimization Objective

Typical GCL methods involve generating augmented views and subsequently optimizing the congruence between their encoded representations. In this paper, we generate two augmented graphs, $\boldsymbol{z}$ and $\boldsymbol{o}$, by using feature augmentation. We randomly sample the mask vector $\boldsymbol{m_f} \in \{0,1\}^{B}$ to hide part of the dimensions in the node feature. Each element in mask $\boldsymbol{m_f}$ is sampled from Bernoulli distribution $Ber(1-f_d)$, where the hyperparameter f_d is the feature descent rate. Therefore, the augmented node feature $\hat{\boldsymbol{X}}$ calculated by the following formula:

$$\hat{\boldsymbol{X}} = [\boldsymbol{x}_1 \circ \boldsymbol{m}_f, \boldsymbol{x}_2 \circ \boldsymbol{m}_f, \cdots, \boldsymbol{x}_N \circ \boldsymbol{m}_f]. \tag{18}$$

We use the comparison objective for the node representation of the two graph augmentation obtained. For node v_a, the node representations from different graph augmentation $\boldsymbol{z}_a$ and $\boldsymbol{o}_a$ form a positive pair. In contrast, the node representations of other nodes in the two graph augmentation are considered negative pairs. Therefore, we define the paired objective of each positive pair $(\boldsymbol{z}_a, \boldsymbol{o}_a)$ as

$$\mathcal{L}_a(\boldsymbol{z}, \boldsymbol{o}) = \log \frac{e^{\theta(\boldsymbol{z}_a, \boldsymbol{o}_a)}}{e^{\theta(\boldsymbol{z}_a, \boldsymbol{o}_a)} + \sum_{b \neq a}(e^{\theta(\boldsymbol{z}_a, \boldsymbol{o}_b)} + e^{\theta(\boldsymbol{z}_a, \boldsymbol{z}_b)})}, \tag{19}$$

where the critic $\theta(\boldsymbol{z}, \boldsymbol{o})$ is defined as $sim(\boldsymbol{H_z}, \boldsymbol{H_o})$, and $sim(\cdot,\cdot)$ refers to cosine similarity function. $\boldsymbol{H_z}$ is the graph embedding generated by graph augmentation through the proposed method ASWT-SGNN. The overall objective to be maximized is the average of all positive pairs,i.e., $\mathcal{L} = -\frac{1}{2N}\sum_{a=1}^{N}[\mathcal{L}_a(\boldsymbol{z}, \boldsymbol{o}) + \mathcal{L}_a(\boldsymbol{o}, \boldsymbol{z})]$.

Complexity Analysis

The first step of ASWT-SGNN is wavelet coefficients approximation. Eq. (8) shows that the immediate solution requires eigen-decomposition of the Laplace matrix to obtain the eigenvalues and eigenvectors of the matrix. However, the complexity of the direct solution is very high. For example, the time complexity of the quick response (QR) algorithm is $\mathcal{O}(N^3)$, and the space complexity is $\mathcal{O}(N^2)$. Therefore, we use the least squares approximation to approximate the solution in this step. If the m-order Chebyshev polynomial approximation is used, the sum of the complexity of each item in the polynomial is calculated as $\mathcal{O}(m \times |E|)$.

The second step of ASWT-SGNN is to use the wavelet transform for graph convolution. Spectral CNN (Bruna et al. 2013) has high parameter complexity $\mathcal{O}(N \times p \times q)$. ChebyNet (Defferrard, Bresson, and Vandergheynst 2016) approximates the convolution kernel by the polynomial function of the diagonal matrix of the Laplace eigenvalue, reducing the parameter complexity to $\mathcal{O}(m \times p \times q)$, where m is the order of the polynomial function. GCN (Kipf and Welling 2016) simplifies ChebyNet by setting m=1. In this paper, the feature transformation is performed first, and the parameter complexity is $\mathcal{O}(p \times q)$. Then the graph convolution is performed, and the parameter complexity is $\mathcal{O}(N)$.

Theoretical Analysis

Lemma 1. *Consider nodes v_a and v_b within a graph, characterized by their similarity in features or labels. Consequently, their k-hop neighbors exhibit a one-to-one mapping, specifically $\mathcal{N}_k(b) = \pi(\mathcal{N}_k(a))$. In this context, it holds true that $|\boldsymbol{\Psi}_{am} - \boldsymbol{\Psi}_{b\pi(m)}| \leq 2\epsilon$, where $\boldsymbol{\Psi}_{am}$ signifies the wavelet coefficient between nodes v_a and v_m.*

Theorem 1. *If the graph following the above assumption and Lemma 1, the expectation of embedding is given by:*

$$\mathbb{E}[\boldsymbol{F}_i] = \boldsymbol{W}\mathbb{E}_{y \sim P(y_i), \boldsymbol{x} \sim P_y(\boldsymbol{x})}[\boldsymbol{\Gamma}_i \boldsymbol{x}], \tag{20}$$

where $\boldsymbol{\Gamma} = \boldsymbol{\Psi}\boldsymbol{G}\boldsymbol{\Psi}^{\top}$, $\boldsymbol{\Gamma}_i$ is the i-th row of $\boldsymbol{\Gamma}$. In this context, we simplify the model by excluding the residual connection component. With probability at least $1-\delta$ over the distribution for the graph, we have:

$$\|\boldsymbol{H}_i - \mathbb{E}[\boldsymbol{H}_i]\|_2 \leq \sqrt{\frac{\sigma^2_{\max}(\boldsymbol{W}) p \log(2p/\delta)}{2\boldsymbol{N}\|\boldsymbol{\Gamma}_i \boldsymbol{x}\|_{\psi_2}}}, \tag{21}$$

where the sub-gaussian norms $\|\boldsymbol{\Gamma}_i \boldsymbol{x}\|_{\psi_2} = \min \|\boldsymbol{\Gamma}_i \boldsymbol{x}_{i,d}\|_{\psi_2}$, p is the dimension of features, $d \in [1,p]$ and $\sigma^2_{max}(\boldsymbol{W})$ is the largest singular value of $\boldsymbol{W}$.

Proof. The proof is shown in Appendix A (Liu et al. 2023). □

The theorem stated above indicates that the proposed method can map nodes with the same label to an area centered around the expectation in the embedding space. This holds true for any graph in which each node's feature and neighborhood pattern are sampled from distributions that depend on the node label.

Experiments

Experimental Setting

Datasets We evaluate the approach on eight benchmark datasets, which have been widely used in GCL methods. Specifically, citation datasets include Cora, CiteSeer

Method	Datasets								
	Cora	CiteSeer	PubMed	Computers	Photo	CS	Physics	WikiCS	Avg.
GCN	81.6±0.2	70.3±0.4	79.3±0.2	84.5±0.3	91.6±0.3	93.1±0.3	93.7±0.2	73.0±0.1	83.4
GAT	83.1±0.3	72.4±0.3	79.5±0.1	85.8±0.1	91.7±0.2	89.5±0.3	93.5±0.3	72.6±0.3	83.5
GWNN	82.8±0.3	71.8±0.2	79.9±0.3	85.6±0.2	92.5±0.1	92.6±0.4	92.7±0.2	72.8±0.3	83.8
GCNII	85.5±0.4	73.4±0.6	80.4±0.3	87.6±0.4	92.7±0.2	92.8±0.4	93.5±0.2	74.7±0.3	85.1
GMI	83.3±0.2	72.6±0.2	79.8±0.4	82.2±0.1	90.7±0.2	92.6±0.2	94.3±0.4	74.9±0.2	83.8
MVGRL	83.1±0.2	72.3±0.5	80.3±0.5	87.5±0.1	91.7±0.1	92.1±0.3	95.1±0.2	77.5±0.1	84.9
GCA-SSG	83.9±0.4	73.1±0.3	81.3±0.4	88.4±0.3	89.5±0.1	92.4±0.1	93.4±0.2	78.2±0.3	85.0
GRADE	84.0±0.3	72.4±0.4	82.7±0.3	84.7±0.1	92.6±0.1	92.7±0.4	93.7±0.2	78.1±0.2	85.1
AF-GCL	83.1±0.1	71.9±0.4	79.0±0.7	**89.6±0.2**	92.5±0.3	92.0±0.1	95.2±0.2	79.0±0.5	85.3
MA-GCL	83.3±0.4	73.6±0.4	83.5±0.7	88.8±0.3	93.8±0.3	92.5±0.4	94.8±0.5	78.7±0.5	86.1
GraphMAE	84.2±0.4	73.4±0.4	81.1±0.4	89.5±0.1	93.2±0.1	92.7±0.2	94.3±0.4	78.9±0.2	85.9
MaskGAE	84.3±0.4	73.8±0.8	83.6±0.5	89.5±0.1	93.3±0.1	92.7±0.5	94.1±0.4	78.4±0.2	86.2
ASWT-SGNN(Semi)	**88.1±0.4**	**81.5±0.3**	**85.2±0.6**	89.4±0.4	**93.8±0.2**	93.2±0.3	94.9±0.4	**79.8±0.5**	**88.2**
ASWT-SGNN(CL)	86.2±0.6	73.9±0.1	84.9±0.3	89.2±0.3	93.5±0.2	**93.5±0.3**	**95.4±0.3**	79.5±0.3	87.0

Table 1: Accuracy (%) on the eight datasets for the node classification. The best result is bold, and the second best is underlined.

and PubMed (Yang, Cohen, and Salakhudinov 2016), co-purchase and co-author datasets include Photo, Computers, CS and Physics (Suresh et al. 2021). Wikipedia dataset includes WikiCS (Mernyei and Cangea 2020).

Baselines We consider several baseline methods for the node classification task. These include semi-supervised learning methods like GCN (Kipf and Welling 2016), GAT (Velickovic et al. 2017) and GCNII (Chen et al. 2020) and wavelet neural network GWNN (Xu et al. 2019). Furthermore, we evaluate six GCL methods and two graph generation learning methods, which are GMI (Peng et al. 2020), MVGRL (Hassani and Khasahmadi 2020), GCA-SSG (Zhang et al. 2021), GRADE (Wang et al. 2022b), AF-GCL (Wang et al. 2022a), MA-GCL (Gong, Yang, and Shi 2023), GraphMAE (Hou et al. 2022), and MaskGAE (Li et al. 2023). These methods represent state-of-the-art approaches in the field of node classification tasks.

Evaluation protocol. For the ASWT-SGNN model, node representations are learned unsupervised using a 2-layer model. Following that, a linear classifier is applied as a post-processing step for assessment. The dataset is randomly partitioned, with 20% of nodes allocated to the training set, another 20% to the validation set, and the remaining 60% to the test set. To ensure the robustness of our findings, we conducted the experiments five times for each dataset, each time with different random seeds. The results include both the average accuracy and the corresponding standard deviation. All experiments use PyTorch on a server with four e NVIDIA A40 GPUs. ASWT-SGNN utilizes the Adam Optimizer with a learning rate of 0.001. The specific hyperparameters are as follows: the number of sampling points in the spectral domain, K, is set to 20, the feature update ratio, α, is set to 0.8, and the wavelet terms ratio, β, is set to 0.4.

Wavelet Operator Approximation Experiments

The proposed method of approximating wavelet operators is applied to real-world graph data. We evaluate its performance using two distinct metrics: the similarity between the approximated wavelet filter g_θ and the precisely computed wavelet filter, and the Mean Absolute Error (MAE) between the approximated wavelet operator $\mathbf{\Psi}_\theta$ and the actual wavelet operator. Figure 2 presents a specific example showing the accurate and approximate multiscale filters, computed precisely and approximately, demonstrating a significant overlap between them. The MAEs for different scaled training sets are depicted in Figure 3. The experimental results indicate that the proposed wavelet operator approximation method reduces the approximation error in the high-density spectral domain, while avoiding the need for computationally expensive eigen-decomposition.

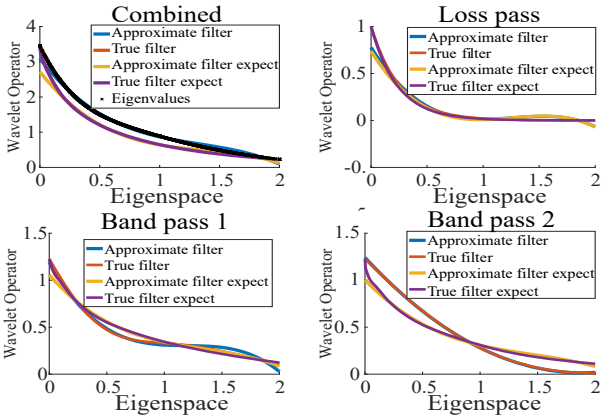


Figure 2: Wavelet filters g_θ for the Cora dataset obtained by actual wavelet and polynomial approximation.

Node Classification

Table 1 displays the results of node classification accuracy. Notably, our proposed ASWT-SGNN achieves state-of-the-art (SOTA) performance on six out of the eight graphical benchmarks. Specifically, compared to other self-supervised methods across the eight datasets, ASWT-SGNN outperforms the best self-supervised method, MaskGAE, by an average of 0.8%, and it outperforms the worst self-supervised

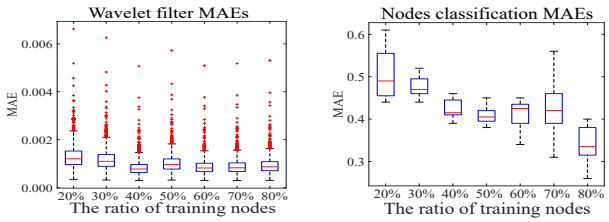


Figure 3: MAE between the approximated and the actual wavelet operator at the eigenvalues (left). MAE between the predicted labels and the actual labels of the nodes on the Cora dataset (right).

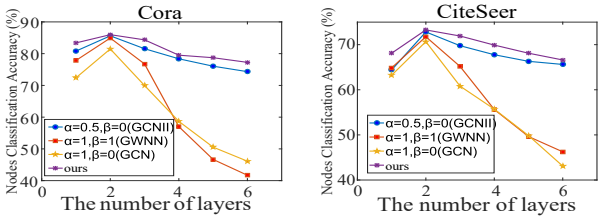


Figure 4: Ablation studies. The degree of over-smoothing under various parameter settings.

method, GMI, by 3.2%. Furthermore, experiments are carried out within a semi-supervised setting, revealing that the proposed method consistently outperforms the best semi-supervised benchmark, GCNII, by an average margin of 3.1%. Additionally, it surpasses the best self-supervised benchmark, MaskGAE, by an average of 2.0%. These results further demonstrate the effectiveness of the proposed method ASWT-SGNN in node classification tasks.

Ablation Studies

The proposed method incorporates two pivotal hyperparameters: α and β. By adjusting these parameters, ASWT-SGNN can be simplified to its core forms: when α is set to 1 and β is set to 0, ASWT-SGNN aligns with GCN; setting both α and β to 1 makes ASWT-SGNN behave similarly to GWNN (Xu et al. 2019); and when $\alpha \neq 1$ and β is set to 0, ASWT-SGNN exhibits similarities to GCNII (Chen et al. 2020). We comprehensively compare ASWT-SGNN and transformation models, including GCN, GWNN, and GCNII. Clear observations can be made from Figure 4: as the number of layers increases, the performance of GCN and GWNN significantly declines. In contrast, the performance of GCNII and ASWT-SGNN remains relatively stable even with more layers stacked. Notably, due to the incorporation of graph wavelet bases, ASWT-SGNN model outperforms GCNII in semi-supervised node classification tasks.

Other Experiments

To comprehensively investigate the impact of parameters α and β on model performance, we systematically vary their values from 0 to 1. The results are presented in Figure 5, illustrating several noticeable trends. When α is set to 0, indicating the exclusion of residual connections, accuracy is

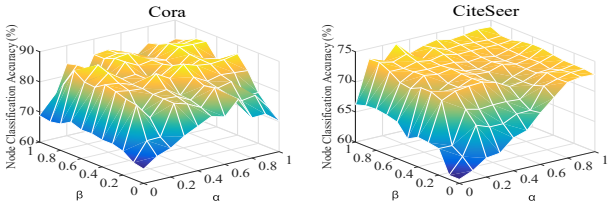
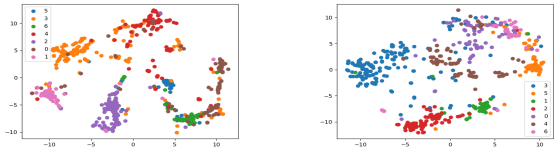


Figure 5: Classification accuracy of the proposed method at different parameter settings (α and β).



(a) ASWT-SGNN (b) MA-GCL

Figure 6: Intra-class distance heatmap and node embedding visualization on Cora dataset.

significantly decreased. On the other hand, larger values of α result in fixed node representations, leading to lower performance. β represents the proportion of the graph wavelet base; if it is tiny, it may not effectively extract local information. Conversely, excessively large values of β could result in the neglect of global information, ultimately reducing performance. These findings highlight the complex relationship between parameters α and β in shaping the model's performance. This delicate balance allows our model to synthesize local and global information, achieving optimal performance effectively.

Furthermore, we use t-SNE (Van der Maaten and Hinton 2008) to visualize the node embeddings. Figure 6 illustrates that compared to MA-GCL, ASWT-SGNN exhibits more pronounced gaps between different classes. This suggests that ASWL-SGNN captures more detailed class information and clarifies the boundaries between samples from different classes. Further extensive experimental analyses, including sparse, robustness, and uncertainty analyses, are exhaustively presented in Appendix B (Liu et al. 2023).

Conclusion

This paper introduces an adaptive graph wavelet self-supervised neural network called ASWT-SGNN. By utilizing multiple wavelet scales, the model integrates different levels of localization on the graph, enabling the capture of elements beyond the low-frequency ones. To avoid the expensive eigen-decomposition in the spectral domain, the model employs a polynomial approximation of the wavelet operator. Comprehensive experimental results demonstrate the competitiveness of the proposed method against state-of-the-art GCL models on real graph datasets. As our framework applies to all message-passing GNNs and polynomial graph filters, we plan to extend its application to more intricate graph neural architectures. Moreover, we will also consider larger datasets with the increase in GPU resources.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of China (No.62106259, No.62076234), Beijing Outstanding Young Scientist Program (NO.BJJWZYJH012019100020098), and Beijing Natural Science Foundation (No. 4222029).

References

Behmanesh, M.; Adibi, P.; Ehsani, S. M. S.; and Chanussot, J. 2022. Geometric multimodal deep learning with multi-scaled graph wavelet convolutional network. *IEEE Transactions on Neural Networks and Learning Systems*, 1–15.

Bruna, J.; Zaremba, W.; Szlam, A.; and LeCun, Y. 2013. Spectral networks and locally connected networks on graphs. arXiv:1312.6203.

Chen, M.; Wei, Z.; Huang, Z.; Ding, B.; and Li, Y. 2020. Simple and deep graph convolutional networks. In *International conference on machine learning*, 1725–1735. PMLR.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 3844–3852.

Di Napoli, E.; Polizzi, E.; and Saad, Y. 2016. Efficient estimation of eigenvalue counts in an interval. *Numerical Linear Algebra with Applications*, 23(4): 674–692.

Ding, K.; Xu, Z.; Tong, H.; and Liu, H. 2022. Data augmentation for deep graph learning: A survey. *ACM SIGKDD Explorations Newsletter*, 24(2): 61–77.

Du, S. S.; Hou, K.; Salakhutdinov, R. R.; Poczos, B.; Wang, R.; and Xu, K. 2019. Graph neural tangent kernel: Fusing graph neural networks with graph kernels. *Advances in neural information processing systems*, 32.

Fan, T.; Shuman, D. I.; Ubaru, S.; and Saad, Y. 2020. Spectrum-adapted polynomial approximation for matrix functions with applications in graph signal processing. *Algorithms*, 13(11): 295.

Gong, X.; Yang, C.; and Shi, C. 2023. Ma-gcl: Model augmentation tricks for graph contrastive learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 4284–4292.

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

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.

Hasanzadeh, A.; Hajiramezanali, E.; Narayanan, K.; Duffield, N.; Zhou, M.; and Qian, X. 2019. Semi-implicit graph variational auto-encoders. *Advances in neural information processing systems*, 32.

Hassani, K.; and Khasahmadi, A. H. 2020. Contrastive multi-view representation learning on graphs. In *International conference on machine learning*, 4116–4126. PMLR.

He, D.; Liang, C.; Liu, H.; Wen, M.; Jiao, P.; and Feng, Z. 2022. Block modeling-guided graph convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 4022–4029.

He, K.; Fan, H.; Wu, Y.; Xie, S.; and Girshick, R. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9726–9735. IEEE Computer Society.

Hou, Z.; Liu, X.; Cen, Y.; Dong, Y.; Yang, H.; Wang, C.; and Tang, J. 2022. Graphmae: Self-supervised masked graph autoencoders. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 594–604.

Hutchinson, M. F. 1989. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3): 1059–1076.

Kaler, T.; Stathas, N.; Ouyang, A.; Iliopoulos, A.-S.; Schardl, T.; Leiserson, C. E.; and Chen, J. 2022. Accelerating training and inference of graph neural networks with fast sampling and pipelining. *Proceedings of Machine Learning and Systems*, 4: 172–189.

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

Kumar, P.; Rawat, P.; and Chauhan, S. 2022. Contrastive self-supervised learning: review, progress, challenges and future research directions. *International Journal of Multimedia Information Retrieval*, 11(4): 461–488.

Li, J.; Wu, R.; Sun, W.; Chen, L.; Tian, S.; Zhu, L.; Meng, C.; Zheng, Z.; and Wang, W. 2023. What's Behind the Mask: Understanding Masked Graph Modeling for Graph Autoencoders. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1268–1279.

Liu, R.; Yin, R.; Liu, Y.; and Wang, W. 2023. ASWT-SGNN: Adaptive Spectral Wavelet Transform-based Self-Supervised Graph Neural Network. *arXiv preprint arXiv:2312.05736*.

Mernyei, P.; and Cangea, C. 2020. Wiki-cs: A wikipedia-based benchmark for graph neural networks. arXiv:2007.02901.

Peng, Z.; Huang, W.; Luo, M.; Zheng, Q.; Rong, Y.; Xu, T.; and Huang, J. 2020. Graph representation learning via graphical mutual information maximization. In *Proceedings of The Web Conference 2020*, 259–270.

Shervashidze, N.; Schweitzer, P.; Van Leeuwen, E. J.; Mehlhorn, K.; and Borgwardt, K. M. 2011. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(9).

Shi, M.; Tang, Y.; Zhu, X.; and Liu, J. 2020. Multi-label graph convolutional network representation learning. *IEEE Transactions on Big Data*, 8(5): 1169–1181.

Shuman, D. I.; Narang, S. K.; Frossard, P.; Ortega, A.; and Vandergheynst, P. 2013. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE signal processing magazine*, 30(3): 83–98.

Sun, F.-Y.; Hoffmann, J.; Verma, V.; and Tang, J. 2019. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. arXiv:1908.01000.

Suresh, S.; Li, P.; Hao, C.; and Neville, J. 2021. Adversarial graph augmentation to improve graph contrastive learning. *Advances in Neural Information Processing Systems*, 34: 15920–15933.

Tang, J. M.; and Saad, Y. 2012. A probing method for computing the diagonal of a matrix inverse. *Numerical Linear Algebra with Applications*, 19(3): 485–501.

Thakoor, S.; Tallec, C.; Azar, M. G.; Munos, R.; Veličković, P.; and Valko, M. 2021. Bootstrapped representation learning on graphs. In *ICLR 2021 Workshop on Geometrical and Topological Representation Learning*.

Tsitsvero, M.; Barbarossa, S.; and Di Lorenzo, P. 2016. Signals on graphs: Uncertainty principle and sampling. *IEEE Transactions on Signal Processing*, 64(18): 4845–4860.

Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11).

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

Velickovic, P.; Fedus, W.; Hamilton, W. L.; Liò, P.; Bengio, Y.; and Hjelm, R. D. 2019. Deep Graph Infomax. *ICLR (Poster)*, 2(3): 4.

Wang, H.; Zhang, J.; Zhu, Q.; and Huang, W. 2022a. Augmentation-Free Graph Contrastive Learning with Performance Guarantee. arXiv:2204.04874.

Wang, R.; Wang, X.; Shi, C.; and Song, L. 2022b. Uncovering the Structural Fairness in Graph Contrastive Learning. *Advances in Neural Information Processing Systems*, 35: 32465–32473.

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.

Wang, Z.; Wang, C.; Pei, J.; Ye, X.; and Philip, S. Y. 2016. Causality Based Propagation History Ranking in Social Networks. In *IJCAI*, 3917–3923.

Xie, Y.; Xu, Z.; Zhang, J.; Wang, Z.; and Ji, S. 2022. Self-supervised learning of graph neural networks: A unified review. *IEEE transactions on pattern analysis and machine intelligence*, 45(2): 2412–2429.

Xu, B.; Shen, H.; Cao, Q.; Qiu, Y.; and Cheng, X. 2019. Graph wavelet neural network. arXiv:1904.07785.

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

Yanardag, P.; and Vishwanathan, S. 2015. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, 1365–1374.

Yang, Z.; Cohen, W.; and Salakhudinov, R. 2016. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, 40–48. PMLR.

Ying, Z.; You, J.; Morris, C.; Ren, X.; Hamilton, W.; and Leskovec, J. 2018. Hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems*, 31.

You, Y.; Chen, T.; Sui, Y.; Chen, T.; Wang, Z.; and Shen, Y. 2020. Graph contrastive learning with augmentations. *Advances in neural information processing systems*, 33: 5812–5823.

Yu, B.; Lee, Y.; and Sohn, K. 2020. Forecasting road traffic speeds by considering area-wide spatio-temporal dependencies based on a graph convolutional neural network (GCN). *Transportation research part C: emerging technologies*, 114: 189–204.

Zeng, H.; Zhou, H.; Srivastava, A.; Kannan, R.; and Prasanna, V. 2019a. Accurate, efficient and scalable graph embedding. In *2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 462–471. IEEE.

Zeng, H.; Zhou, H.; Srivastava, A.; Kannan, R.; and Prasanna, V. 2019b. Graphsaint: Graph sampling based inductive learning method. arXiv:1907.04931.

Zhang, H.; Wu, Q.; Yan, J.; Wipf, D.; and Yu, P. S. 2021. From canonical correlation analysis to self-supervised graph neural networks. *Advances in Neural Information Processing Systems*, 34: 76–89.

Zhu, Y.; Xu, Y.; Yu, F.; Liu, Q.; Wu, S.; and Wang, L. 2021. Graph contrastive learning with adaptive augmentation. In *Proceedings of the Web Conference 2021*, 2069–2080.