Curriculum-Enhanced Residual Soft An-Isotropic Normalization for Over-Smoothness in Deep GNNs

Jin Li[1, 2], **Qirong Zhang**[1], **Shuling Xu**[1], **Xinlong Chen**[1], **Longkun Guo**[1, 3], **Yang-Geng Fu**[1*]

[1]College of Computer and Data Science, Fuzhou University, Fuzhou, China
[2]AI Thrust, Information Hub, HKUST (Guangzhou), Guangzhou, China
[3]Shandong Provincial Key Laboratory of Computer Networks,
Shandong Fundamental Research Center for Computer Science, Jinan, China
{jslijin2015, shuling_xu}@outlook.com, {jiusizhang1, fjxinlong, longkun.guo}@gmail.com, fu@fzu.edu.cn

Abstract

Despite Graph neural networks' significant performance gain over many classic techniques in various graph-related downstream tasks, their successes are restricted in shallow models due to over-smoothness and the difficulties of optimizations among many other issues. In this paper, to alleviate the over-smoothing issue, we propose a soft graph normalization method to preserve the diversities of node embeddings and prevent indiscrimination due to possible over-closeness. Combined with residual connections, we analyze the reason why the method can effectively capture the knowledge in both input graph structures and node features even with deep networks. Additionally, inspired by Curriculum Learning that learns easy examples before the hard ones, we propose a novel label-smoothing-based learning framework to enhance the optimization of deep GNNs, which iteratively smooths labels in an auxiliary graph and constructs many gradual non-smooth tasks for extracting increasingly complex knowledge and gradually discriminating nodes from coarse to fine. The method arguably reduces the risk of overfitting and generalizes better results. Finally, extensive experiments are carried out to demonstrate the effectiveness and potential of the proposed model and learning framework through comparison with twelve existing baselines including the state-of-the-art methods on twelve real-world node classification benchmarks.

1 Introduction

Graph neural networks (GNNs) (Wu et al. 2020) are widely used state-of-the-art techniques to solve many tasks on graph (*e.g.*, semi-supervised node classification (Feng et al. 2020), link prediction (Yun et al. 2021), graph classification (Xie et al. 2022), and community detection (Liu et al. 2021), etc). Also, GNNs have achieved outstanding results recently in many domains including texts (Fei, Zhang, and Zhou 2021), images (Guan et al. 2022), traffic (Choi et al. 2022), molecule (Han et al. 2022), and even electroencephalogram (*i.e.*, EEG) (Demir et al. 2021) compared to classic methods (*e.g.*, CNN and RNN). They are previously derived in spectral domain based on the eigen-decomposition of graph Laplacian (*e.g.*, Spectral-CNN (Bruna et al. 2014) and ChebNet (Defferrard, Bresson, and Vandergheynst 2016)). Graph Convolutional Network (GCN) (Kipf and Welling 2017) is proposed to accelerate it via a linear approximation of universal filters on graph signals and also give an intuitive interpretation on spatial domain, *i.e.*, *message passing*, which is further developed by SGC (Wu et al. 2019), GAT (Veličković et al. 2018), GIN (Xu et al. 2019), and MPNN (Gilmer et al. 2017).

More recently, some deep GNNs (Chen et al. 2020; Li et al. 2019) are proposed to further improve the expressive power of GNNs in light of successes of other deep neural networks, *e.g.*, CNN and RNN. But unfortunately, GNNs are not easy to go deep and often suffer from severe performance degradation due to, *e.g.*, over-smoothness (Liu, Gao, and Ji 2020), difficulty in optimization (Yang et al. 2020), memory limitation (Li et al. 2021), time consumption (Li et al. 2021), and over-squashing (Topping et al. 2022). In this paper, we mainly focus on the first two problems, *i.e.*, improvements will be devised with respect to the following two aspects: 1) structures; and 2) the learning process.

In order to alleviate over-smoothness, various kinds of techniques (Chen et al. 2022) are proposed to prevent over-closeness of node embeddings or reduce the extent of aggregations including graph normalization (Zhou et al. 2021), residual connections (Chen et al. 2020), and random dropping (Huang et al. 2020), etc. Existing graph normalization techniques directly operate with norms, means, variances or distances of embeddings, and can effectively reduce the over-closeness. However, it's still unclear what or how much knowledge they can preserve in deep layers via only these numeric-related operations. Some residual connections methods are proven to keep useful feature semantics as GNNs go deep, but they may risk missing structural knowledge, *e.g.*, GCNII (Chen et al. 2020). Random dropping approaches are devised mainly for regularization without theoretical guarantees for alleviating information loss in over-smoothness and seldom perform competitively compared to other state-of-the-arts.

Thus for structures, we propose *R-SoftGraphAIN*, a residual connections-based soft graph normalization layer, which can be viewed as a combination of a *novel* soft graph normalization operation and two *improved* residual connections. Compared to Pairnorm (Zhao and Akoglu 2020), it can normalize embeddings in an an-isotropic manner instead of equally treating all nodes. Compared to GCNII, it can be

*Corresponding author

shown to preserve relatively high frequent structural knowledge instead of over-emphasizing features and can be viewed as a generalization of GCNII. In Sec. 3.1, theoretical analysis is carried out to show its following characteristics as the depth approaches infinity: 1) The mean distance of pairwise node embeddings will be kept nearly constant similar to Pairnorm; 2) The diversities of these signals are maximized; 3) It tends to extract the most d lowest frequent components of structural knowledge; 4) It never forgets the original feature's information. Experimental evaluations prove its effectiveness due to significant performance gain compared to others.

On the other hand, in order to ease the optimization of deep GNNs, we borrow ideas from Curriculum Learning (CL) (Wang, Chen, and Zhu 2022; Soviany et al. 2022). In CL, models are encouraged to first learn from easy examples and then examples with gradually increased difficulty (Bengio et al. 2009). The idea has been generalized to devise better curriculum applied to various scenarios in many domains including texts and images via, *e.g.*, designing multifarious tasks with increasing difficulties (Caubrière et al. 2019), gradually unleashing expressive powers of models (Sinha, Garg, and Larochelle 2020), or defining *pacing* functions to decide to which extent to learn from a task (Hacohen and Weinshall 2019). However, graphs contain specific structures and semi-supervised learning has a special setting, which may require a careful curriculum design, but few prior works focus on this (see Sec. 2). Therefore in this paper, we give a simple yet effective label-smooth-based example called *SmoothCurriculum*. More specifically, we first employ *label propagation* to estimate unknown labels, and all labels are iteratively smoothed in an auxiliary graph built via a pre-trained teacher model in order to construct gradual non-smooth tasks. From the analysis in Sec. 3.2, this learning framework encourages graph encoders to extract increasingly complex knowledge and learn to gradually discriminate nodes from coarse to fine[1], which intuitively emphasizes relatively global knowledge and alleviates possible label noise, thus reducing the risk of overfitting and generalizing better. Obvious performance improvements across various real-world datasets reveal the potential of this simple framework.

The contribution of this paper can be summarized as follows:

- We propose a residual connections-favored soft graph normalization structure (called *R-SoftGraphAIN*) for preserving knowledge from both input graph topology and features and retaining the diversities of node embeddings in deep layers to consequently alleviate over-smoothness.
- We design a novel label-smoothing-based curriculum learning framework (called *SmoothCurriculum*) to ease the difficulty of optimization of deep GNNs and better their generalization via implicit coarse-to-fine node discrimination.
- Extensive experiments were carried out to demonstrate the effectiveness and potential of our method compared to twelve existing baselines including state-of-the-arts.

[1]E.g., for collecting residential information of a person, first the information of the country and then the city he lives in will be collected.

2 Preliminaries and Related Work

Notation Let $G = (V, E)$ be an undirected graph with node set V and edge set E, where $n = |V|, m = |E|$ represent the numbers of its nodes and edges respectively. We denote by $A \in \{0, 1\}^{n\times n}$ and $X \in \mathbb{R}^{n\times d}$ its adjacency and feature matrix where node i has feature $x_i = X_{i,:} \in \mathbb{R}^d$ and a ground-truth label $y_i = Y_i \in \mathbb{N}$. Define $I_n \in \mathbb{R}^{n\times n}$ as an identity matrix, $\mathbf{0_n}, \mathbf{1_n} \in \mathbb{R}^{n\times 1}$ as all-zero/one vectors.

GCN and SGC GCN can be formulated as follows:

$$H^{(0)} = X, \quad H^{(l+1)} = \sigma\left(\hat{A}H^{(l)}W^{(l)}\right) \in \mathbb{R}^{n\times d}. \quad (1)$$

SGC simplifies GCN by dropping its non-linear activation functions and its forward pass can be described as follows:

$$H^{(l)} = \hat{A}^l XW \in \mathbb{R}^{n\times d}, \quad \forall l \in [0, L), \quad (2)$$

where L is the number of layers and $\sigma(\cdot)$ denotes a non-linear activation function (*e.g.*, ReLU, or Softmax for the last layer). $\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$ is the symmetrically normalized matrix of $\tilde{A} = A + I_n$ and diagonal matrix $\tilde{D}_{i,i} = \sum_{j=1}^{n}\tilde{A}_{i,j}$. Sometimes, the probability transition matrix $\hat{A}_{rw} = \tilde{D}^{-1}\tilde{A}$ is employed for aggregating. $H^{(l)} \in \mathbb{R}^{n\times d}$ and $W, W^{(l)} \in \mathbb{R}^{d\times d}$ represent the embeddings and the trainable parameters.

Deep GNNs and Over-Smoothness Majority of GNN variants are shallow networks (*e.g.*, no more than three layers), thus restricting their expressive power and limiting distant message passing. Some prior works make efforts to deepen GNNs via modifications or tricks which can be categorized into three classes: residual connections, graph normalization (*e.g.*, Pairnorm (Zhao and Akoglu 2020), Nodenorm (Zhou et al. 2021), Meannorm (Yang et al. 2020)), and random dropping (*e.g.*, DropEdge (Rong et al. 2020) and DropNode (Huang et al. 2020)). Residual connections contain common residual connection (from last layer) (Li et al. 2019), initial connection (from the first layer) (Chen et al. 2020), dense connection (from every layer) (Liu, Gao, and Ji 2020; Luan et al. 2019), and jump connection (from every layer to the last layer only) (Xu et al. 2018). See supplementary materials or (Chen et al. 2022) for some more related work or a more detailed survey. Our method appropriately combines *improved* residual connections and a *novel* soft graph normalization enabling effective feature and structural knowledge extraction and preservation even with a sufficiently large depth.

Curriculum Learning CL has become a popular kind of training strategies for networks in many applications including texts (Liu et al. 2020), images (Zhou, Wang, and Bilmes 2020), speeches (Wang et al. 2020), reinforcement learning (Narvekar et al. 2020), etc. The basic idea is to give examples from easy to hard, and has been developed a lot (Wang, Chen, and Zhu 2022), but few are designed specifically for graph-rated tasks (Wang et al. 2021; Chu et al. 2021). But note that besides over-smoothness, another non-negligible cause hindering deep GNNs is just the difficult optimization. Thus in this work, we hope to ease it via a novel curriculum learning framework based on iterative label smoothing on an auxiliary graph. Here we define a *curriculum* as a *task sequence*

$\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_{n_T}$ with gradually increasing difficulties where $\mathcal{T}_i = \left(D^{(i)}, f_\theta^{(i)}, \ell^{(i)}(\cdot), p^{(i)}\right)$ denotes a task meaning that a model $f_\theta^{(i)}$ learns from the data $D^{(i)}$ with a loss function $\ell^{(i)}(\cdot)$ and *pacing* strategy $p^{(i)}$ (*e.g.*, the time it spends). $f_\theta^{(i)}$ is some modified or restricted version of f_θ.

3 SmoothCurriculum-Improved R-SoftGraphAIN

In this section, we propose a novel model for alleviating the over-smoothness of graph neural networks by incorporating two main ingredients (*i.e.*, *R-SoftGraphAIN* for GNN structures, and the adaptive curriculum design). Although treated as a whole for reporting their performance in Sec. 4, we individually introduce each ingredient in the following for better clarity and briefness.

3.1 R-SoftGraphAIN

We will describe our normalization method and show how it can be improved via residual and initial connections in the following paragraphs.

Spectral Analysis on Over-Smoothness Over-closeness (*i.e.*, embeddings are too close) is the external manifestation of over-smoothness leading to indiscrimination via a classifier. But it's just a direct cause instead of an essential reason analyzed by priors in the spectral domain on SGC as follows:

$$\begin{aligned} h^{(L)} &= \hat{A}^L x = U\Lambda^L U^T x = \sum_{i=1}^{n} \lambda_i^L u_i u_i^T x, \\ \lim_{L\to\infty} h^{(L)} &= u_1 u_1^T x = \tilde{D}^{\frac{1}{2}} \mathbf{1}_n \mathbf{1}_n^T \tilde{D}^{\frac{1}{2}} x \propto \tilde{D}^{\frac{1}{2}} \mathbf{1}_n, \end{aligned} \tag{3}$$

where x is a signal, $\hat{A} = U\Lambda U^T$ contains decreasing eigenvalues $\{\lambda_i\}$ with respective eigenvectors $\{u_i\}$ and $\lambda_1 = 1 > \lambda_2$. The conclusion on GCN is similar with a very different non-trivial analysis (Oono and Suzuki 2020). The analysis tells us deep GNNs tend to: 1) hardly keep important structural knowledge, 2) gradually forget the semantics contained in features, thus essentially leading to over-smoothness.

Pairnorm (Zhao and Akoglu 2020) attempts to numerically solve over-closeness via direct manipulation on mean distance formulated as: $H^{(l+1)} = C\sqrt{n} \cdot H' / \|H'\|_F$, $H' = (I_n - 1/n \cdot \mathbf{1}_n \mathbf{1}_n^T)\hat{A}H^{(l)}$ with a constant $C > 0$. We conjecture its performance is limited even getting rid of indiscrimination due to normalizing embeddings: 1) only elementwisely and isotropically without modeling the complex relationships between nodes and between signals; 2) only numerically without interpretable knowledge preservation. These shortcomings motivate our *GraphAIN* considering normalization an-isotropically and in a distribution/knowledge-aware manner.

Soft Graph An-Isotropic Normalization As mentioned above, we propose *GraphAIN* to deal with the comprehensive distribution and keep diverse meaningful knowledge, which can be described as follows:

$$\begin{aligned} H_t &= B_{t-1}\left(B_{t-1}^T B_{t-1}\right)^{-\frac{1}{2}} \in \mathbb{R}^{n\times d}, \quad \forall\, t \ge 1, \\ B_{t-1} &= T \cdot \hat{A} H_{t-1} \in \mathbb{R}^{n\times d}, \quad H_0 = X, \end{aligned} \tag{4}$$

where $H_t = H^{(t)}$ and X denote the embedding matrix of the t-th layer and the original features. And $T = I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T \in \mathbb{R}^{n\times n}$ represents a centering operator and $P^{1/2}$ refers to the square root of a positive matrix P. The following statement theoretically justifies our idea that we are normalizing the covariance matrix of H instead of directly normalizing the embedding for an individual node or signal independently:

Theorem 1. $\forall\, t \ge 1$, *GraphAIN satisfies that:*
1) $\mathbf{1}_n^T B_t = \mathbf{1}_n^T H_t = \mathbf{0}_n$; *2)* $H_t^T H_t = I_d$;
3) $TH_t = H_t$; *4)* $B_t = \bar{A} H_t$;
where $\bar{A} = T\hat{A}T$ *denotes a doubly centered version of* $\hat{A}$.

Next theorem theoretically analyzes *GraphAIN* from an optimization perspective and gives a deep understanding of combination of normalization and aggregations in GNNs:

Theorem 2. *GraphAIN can be viewed as an iterative process in order to solve the following restricted optimization problem via* ***Projected Gradient Ascent*** *method:*

$$\max_{H} \quad f(H) = \frac{1}{2} \cdot \operatorname{tr}\left(H^T \bar{A} H\right), \quad s.t.\ H^T H = I_d, \tag{5}$$

where H is initialized to input graph signals $X \in \mathbb{R}^{n\times d}$ and set the step size $\eta = 1$.

All proofs can be found in supplementary materials. This theorem reveals what *GraphAIN* can learn. In fact, from another point of view, the optimal solution to this problem can be obtained via ***Lagrange multiplier*** method as follows:

$$\mathcal{L}(H, \Lambda_L) = \operatorname{tr}\left(H^T \bar{A} H\right) - \operatorname{tr}\left(\Lambda_L \left(H^T H - I_d\right)\right), \tag{6}$$

where $\mathcal{L}$ and Λ_L are the Lagrange function and multipliers. Let $\partial\mathcal{L}/\partial H = 0$, we get $\bar{A}H = \Lambda_L H$, which means that H tends to the eigenvectors corresponding to the top-d eigenvalues of $\bar{A}$ with sufficient steps. Thus similar to Spectral Clustering, it can capture essential structural knowledge due to the similarity between $\bar{A}$ and $\hat{A}$. In addition to spectral interpretations, we give some intuitions in spatial domain: 1) it can easily solve over-closeness, due to the facts that $\|H\|_F^2 = d$ and $\sum_i \sum_j \|H_{i,:} - H_{j,:}\|_2^2 = 2n \cdot \sum_i \|H_{i,:}\|_2^2 - 2 \cdot \|\sum_i H_{i,:}\|_2^2 = 2n \cdot d$ meaning the average pairwise distance is kept completely constant similar to Pairnorm; 2) the spatial variance in any direction is normalized to 1 for maximally preserving the diversity of knowledge in a circular distribution.

However, it still suffers from performance degradation due to the following four potential drawbacks: 1) too absolute; 2) numerical instability; 3) high time complexity; 4) risk of forgetting original features during iterations. The first three issues can be relieved via a soft version (*i.e.*, *SoftGraphAIN*):

$$H_t \approx B_{t-1}\left[a \cdot U_{d_0} \Lambda_{d_0}^{-\frac{1}{2}\cdot b} U_{d_0}^T + (1-a) \cdot I_d\right], \tag{7}$$

where $B_{t-1}^T B_{t-1} \approx U_{d_0} \Lambda_{d_0} U_{d_0}^T$ is the d_0-truncated SVD calculating only the top-$d_0 \le d$ eigenvectors and eigenvalues contained in $U_{d_0}, \Lambda_{d_0} \in \mathbb{R}^{d\times d}$, and $a, b \in [0, 1]$ are another hyper-parameters controlling the extent of normalizing. Formally, they transform the singular values $S \approx S_{d_0} = \Lambda_{d_0}^{1/2} \in \mathbb{R}^{d\times d}$ into $(1-a) \cdot S_{d_0} + a \cdot S_{d_0}^{1-b}$ thus flexibly reducing the absoluteness, possible noise in useless channels, risk of numerical zero-divisions, and empirical time-inefficiency.

Additional Residual Combination *R-SoftGraphAIN* can effectively alleviate the last drawback mentioned above via some residual and initial connections formulated as follows:

$$B_t = \alpha \cdot T\hat{A}H_t + \beta \cdot H_t + \gamma \cdot X \in \mathbb{R}^{n\times d},\ \forall\, t \geq 1, \quad (8)$$

where the non-negative hyper-parameters meet $\alpha+\beta+\gamma = 1$. Intuitively, the commonly used residual connections can alleviate gradient-vanishing and the initial ones are expected to constantly supplement some feature information during aggregations in case of oblivion. Moreover, the motivation can be theoretically justified via the following similar theorem:

Theorem 3. *Residual-favored GraphAIN can be viewed as an iterative process in order to solve the following restricted optimization problem via* ***Projected Gradient Ascent*** *method:*

$$\begin{aligned} \max_H\ f(H) =& \frac{1}{2} \cdot \mathrm{tr}\left(H^T \bar{A} H\right) - \frac{1}{2} \cdot \frac{\gamma}{\alpha} \cdot \|H - X\|_F^2 \\ & s.t. \quad H^T H = I_d, \end{aligned} \quad (9)$$

where H is initialized to input graph signals $X \in \mathbb{R}^{n\times d}$ and set the step size $\eta = \alpha \in [0, 1]$.

This theoretically reveals that *R-SoftGraphAIN* never forgets the original features as GNNs go deep, simultaneously relieving two essential reasons in Sec. 3.1 and thus alleviating over-smoothness. Furthermore, from this we can get some intuitions on the roles of α, β, γ: α allows a sufficiently small step size ensuring better convergence, γ estimates the contribution of features. β can give some freedom to α and γ. In order to further improve its performance, we generalize these connections as *fuzzy* connections. We use Eq. 7 and Eq. 8 to substitute Eq. 8, and replace X in Eq. 8 by H_1 due to possible misalignment in dimensions. A GCN-based implementation of the whole structure is summarized in Algorithm 1 in supplementary materials with a line-by-line description therein.

Relations to Others In this paragraph, we detailedly compare ours with other related methods. SGC suffers from over-smoothness due to both structural and feature knowledge loss. Pairnorm numerically solves over-closeness without interpretable knowledge preservation. Meannorm and Spectral Clustering (SC) can keep the 2-th and lowest d frequent components in structures respectively while keeping little feature information. GCNII proves to be a universal approximator of any function on features, but it ignores structural semantics. Compared to them, ours can keep both features and structural knowledge inheriting both advantages. From another perspective, Pairnorm, Meannorm, and Nodenorm (Zhou et al. 2021) are only element-wise, signal-independent, and node-independent, respectively. However, our method considers the comprehensive distributions and effectively models the relationships between nodes and between signals, thus uttermost preserving the diversity of the embeddings. Fig. 3 in supplementary materials shows our superiority, where ours is similar to and even outperform SC while others suffer from over-smoothness to different extents.

3.2 SmoothCurriculum

In this section, we propose a *simple* yet *effective* curriculum learning framework based on *label-smoothing* on an auxiliary graph to ease the hardness of optimizing the proposed *R-SoftGraphAIN*. Inspired by Curriculum Learning, the key idea of our framework is first to learn the low-frequent knowledge contained in labels before the high-frequent ones, and then to employ an easy-to-hard learning process that favors a better generalization. The framework will be described in detail regarding several of its important modules (*e.g.*, label estimation and smoothing, graph construction, and curriculum designs), intuitions, and interpretations.

Label Estimation and Auxiliary Graph Deep GNNs are powerful yet risk overfitting due to limited labeled data, especially in the semi-supervised setting. Thus we hope to enlarge the training set via label estimation. One of the most commonly used classic techniques is *Label Propagation*, whose iterative process is: $f \leftarrow P \cdot f, f_L \leftarrow Y_L$ initializing $f_L = Y_L, f_U = \mathbf{0}$. Furthermore, its limit can be formulated as: $Y_U^{(e)} = \lim f_U = (I - P_{UU})^{-1} P_{UL} Y_L$, where U, L represent unlabeled and labeled node sets, $P = D^{-1}A$, P_{UL} is a sub-matrix of P respect to lines U and columns L, and $Y_L, Y_U^{(e)}$ are the known and estimated labels. But it suffers from two shortcomings: 1) impossible propagation due to possible disconnectivity; 2) impractical matrix inversion with a large $|U|$. Thus, we estimate Y_U *implicitly* via a teacher model $f_t(\cdot)$ pre-trained on the labeled data $D_o = (X_L, Y_L)$, which can distill shared knowledge from distant nodes or disconnected components to favor more accurate estimation.

Moreover, we expect to capture and encode the similarities of nodes' ground-truth labels into the structure or communities of an auxiliary graph G_{aux}. It can be built as follows: 1) $G_{aux} = G$, input graph for graphs with noisy or missing features; 2) $G_{aux} = G_f$, a KNN-graph built according to node features for graphs with heterophily; 3) $G_{aux} = G_e$, a KNN-graph built according to node embeddings output by the teacher $f_t(\cdot)$ for others. More specifically, a KNN-graph $G(\mathbf{h})$ of a set of vectors $h_1, \cdots, h_n$ is built as follows: link every h_i to its top-k nearest vectors via a KNN algorithm with *Gaussian* distance, drop the edge directions, and then calculate the weights via similarity scores $W_{i,j}^{(aux)} = \mathrm{ReLU}\left(h_i^T h_j\right)^{\gamma'}$ with a distribution-controlling hyper-parameter $\gamma' > 0$ for edge weights $W^{(aux)} \in \mathbb{R}^{n\times n}$.

Label Smoothing and Curriculum Design To get multi-scale label signals, we iteratively smooth labels on G_{aux} with initial signal $Y^{[0]} = Y$ from $f_t(\cdot)$ as: $Y^{[i+1]} = P_{aux} \cdot Y^{[i]},\ \forall\, i \in [0, n_T)$, where $P_{aux} = D_{aux}^{-1} W^{(aux)}$ and D_{aux} are the probability and degree matrix on G_{aux}, respectively. Note that this simplified Label Propagation without fixing f_L will definitely encounter over-smoothness similar to SGC, but it's just what we desire (see next paragraph). After that, a curriculum $\mathcal{C}$ can be defined as $\mathcal{T}_0, \mathcal{T}_1, \cdots, \mathcal{T}_{n_T}$, where task $\mathcal{T}_i = \left(D^{(i)}, f_\theta, \ell(\cdot), p^{(i)}\right)$, *i.e.*, the graph encoder f_θ and the loss $\ell(\cdot)$ are shared in all tasks but the training data $D^{(i)} = \left(X, Y^{(i)}\right)$ and pacing strategies $p^{(i)}$ vary. Here, we prepare $Y^{(i)} = Y^{[n_T - i]},\ \forall\, i \in [0, n_T]$. In other words, the encoder f_θ will be encouraged to learn tasks from $\mathcal{T}_0$ to $\mathcal{T}_{n_T}$ where easy tasks containing easy data $D^{(i)}$ are solved before the relatively harder ones with *even paces*. And finally, it will be fine-tuned to solve the original task with $D_o = (X_L, Y_L)$.

Analysis and Interpretations Next we give some analysis and intuitions on what *SmoothCurriculum* exactly does and how it guides the training in the following aspects: 1) optimizing from convex to non-convex: as claimed in (Bengio et al. 2009; Wang, Chen, and Zhu 2022), curriculum learning with priority for easy tasks can be equivalently understood as landscape smoothing for empirical loss contributing to a more convex optimization problem, which guides models to find a local minima with less vibration and better generalizability. 2) learning spectral knowledge from low- to high-frequency: $Y^{[i]} = (P_{aux})^i Y^{[0]} = U\Lambda^i U^T Y^{[0]}$ and $\Lambda^i = \mathrm{diag}\left(\lambda_1^i, \lambda_2^i, \cdots, \lambda_n^i\right)$ with always decreasing eigenvalues. Let i vary *decreasingly*, and consider important values $\{i_j, j \in [1, n]\}$ where at time i_j, $Y^{[i_j]} \approx \sum_{t=1}^{j} \lambda_t^{i_j} u_t u_t^T Y^{[0]}$ with the dominating top-j eigenvalues $\{\lambda_k^i, k \in [1, j]\}$. Then from $Y^{[i_j]}$ to $Y^{[i_{j+1}]}$, old knowledge will be reviewed due to $\lambda_t^{i_j} \leq \lambda_t^{i_{j+1}}$ and some relative high-frequent component $u_{j+1} u_{j+1}^T Y^{[0]}$ as new information will be injected to label signals. Additionally, if we independently consider a single signal $y^{[0]}$, then $u_{j+1} u_{j+1}^T y^{[0]} = \left(u_{j+1}^T y^{[0]}\right) u_{j+1} \propto u_{j+1}$ introducing a new channel for spectral embedding encoding some new details leading to more complex clustering structures. Thus models can learn to discriminate nodes from coarse to fine. 3) learning spatial knowledge from global to local: Intuitively, the shared commonsense is illustrated first due to $\lim_{i\to\infty} (P_{aux})^i Y^{[0]} = \mathbf{1}_n \mathbf{1}_n^T Y^{[0]}$, which encodes the global label frequency. Then the pieces of information in big communities, small communities, and local environments are presented in order because over-smoothness happens quickly in high-density regions but slowly otherwise. In other words, it also spatially favors coarse-to-fine node discrimination via perceiving the multi-scale community structure or density varieties of G_{aux}.

4 Experimental Results

In this section, we conduct extensive experiments to evaluate the effectiveness of our method (applied with GCN, GAT, and GIN) by comparing it with twelve baselines on twelve real-world graph benchmarks on semi-supervised node classification tasks. Note that our method can be applied to more sophisticated spatial propagation-based GNN backbones to further improve its performance, but we prefer basic ones to keep it simple and evaluate its potential. Due to space limitations, some experimental details are given in supplementary materials including dataset descriptions, implementations, omitted results (*e.g.*, with other layers, with different splits, comparisons with more baselines on heterophilous graphs, as well as standard errors), hyper-parameters (searching spaces and specific configurations), and some more visualizations.

Experimental Settings They are performed on an Ubuntu system with a single GeForce RTX 2080Ti GPU (12GB Memory) and 40 Intel(R) Xeon(R) Silver 4210 CPUs. And the proposed model is implemented by Pytorch (Paszke et al. 2019) and optimized with Adam Optimizer. For a fair comparison, twelve real-world public benchmarks are chosen, including two kinds: 1) eight graphs with homophily: four widely used scientific citation networks (*i.e.*, Core, Citeseer, Pubmed (Sen et al. 2008), and a large-scale graph OGBN-ArXiv (Hu et al. 2020)), scientific co-authorship networks Physics and CS (Mernyei and Cangea 2020), as well as Amazon purchasing system Computers and Photo (Shchur et al. 2018); 2) four graphs with heterophily: webpage datasets Texas, Wisconsin, and Cornell (Pei et al. 2020) as well as an actor co-occurrence network Actor (Tang et al. 2009). Their statistics and adopted splits are summarized in Tab. 4 in supplementary materials. We adopt the standard semi-supervised training/validation/testing splits for them following prior works (Kipf and Welling 2017; Chen et al. 2020, 2022). Furthermore, twelve baselines or state-of-the-art GNN models are applied for comparison including four vanilla classic models (GCN, SGC, GAT, and GIN), two spectral-based methods (ChebNet (Defferrard, Bresson, and Vandergheynst 2016) and BernNet (He et al. 2021)), a normalization-based method Pairnorm (Zhao and Akoglu 2020), some residual connections-based methods including GCNII (Chen et al. 2020), GPRGNN (Chien et al. 2021), APPNP (Gasteiger, Bojchevski, and Günnemann 2019), JKNet (Xu et al. 2018), and DAGNN (Liu, Gao, and Ji 2020). For a fair comparison with spectral-based baselines, we view the orders of Laplacian used in filters as the depths.

Node Classification with Homophily and Heterophily We call the proposed method applied to GCN, GAT, and GIN *Ours(GCN)*, *Ours(GAT)*, and *Ours(GIN)*, respectively, where Ours(GCN) is the default, *i.e.*, *Ours*. We run each experiment five times with different initializations, and report the average accuracies in Tab. 1 and Tab. 2 with varied numbers of layers on these benchmarks. The standard errors and results with some other layers are given in supplementary materials. From Tab. 1 and 2, it is shown that our model consistently achieved the best results against these state-of-the-art counterparts in almost all listed layers. Notably, for Amazon Computers Dataset, we get a performance improvement compared to DAGNN of more than 6.8% and 7.4% in 32 and 64 layers, respectively. As observed from Tab. 2, our method outperforms any other listed model by very large margins on four heterophilous graphs and the large-scale graph OGBN-ArXiv. In supplementary materials, we also provide results with fully supervised random splits compared to the listed counterparts (see Tab. 16) and more baselines on these heterophilous graphs (see Tab. 17). These results demonstrate its potential to alleviate over-smoothness in deep layers.

Node Classification with Noisy Features Sometimes features can provide enough meaningful supervision signals for node prediction, which veils the ability of a GNN for *structure understanding*. In this subsection, we evaluate our model in a challenging task called *Node Classification with Noisy Features* where all node features are substituted by noise sampled from Standard Normal Distribution $\mathcal{N}(0, 1)$ while only the structure of input graph remains. This task is more difficult than that in (Zhao and Akoglu 2020), since: 1) The feature substitution is conducted for all nodes instead of the nodes out of the training set only. 2) We make the features noisy instead of replacing them with zeros. Intuitively, this task tests *how deeply GNNs can understand the input structure*, *i.e.*, whether they can capture more useful structural knowledge for alleviating the adverse effect of noise in

Method	Cora		Citeseer		Pubmed		CS		Physics		Computers		Avg.Rank
#Layes	32	64	32	64	32	64	32	64	32	64	32	64	
GCN	31.90	27.56	36.66	25.40	44.22	32.65	41.29	34.23	79.87	75.34	58.30	37.58	12.75
SGC	63.84	55.39	67.50	63.08	70.70	65.33	70.52	72.51	91.46	90.77	37.44	37.50	10.33
ChebNet	31.90	20.63	33.43	24.90	48.67	45.37	29.28	23.24	70.35	50.74	58.58	50.12	12.67
GAT	72.28	31.92	59.08	22.90	78.72	41.88	85.85	12.83	91.87	17.75	76.05	37.18	10.92
GIN	60.92	31.90	47.32	23.10	72.94	40.23	52.90	20.79	83.02	24.98	39.18	37.50	12.42
Pairnorm	65.00	66.24	44.20	41.48	72.12	71.72	72.71	68.62	88.51	89.11	74.96	74.35	9.67
GCNII	**85.29**	**85.34**	73.24	73.00	79.81	79.88	71.67	72.11	93.15	92.79	37.56	37.50	6.67
JKNet	73.23	72.54	50.68	52.22	63.77	69.10	81.82	82.84	90.92	89.88	67.99	67.78	9.17
GPRGNN	83.13	82.48	71.01	70.96	78.46	78.92	89.56	89.33	93.49	93.26	41.94	78.30	6.83
DAGNN	83.39	82.16	72.59	71.00	80.58	80.44	89.60	89.47	93.31	93.52	79.73	79.23	4.92
APPNP	83.68	83.66	72.13	72.02	80.24	80.08	91.61	91.58	93.75	91.61	43.02	41.42	5.67
BernNet	81.38	17.72	70.82	39.10	70.24	32.86	91.54	9.20	92.27	19.38	81.06	12.81	10.67
Ours(GCN)	85.12	84.87	74.42	**74.50**	**81.28**	**81.58**	**92.11**	**92.05**	94.22	94.20	**85.21**	**85.13**	1.42
Ours(GAT)	84.60	84.68	74.26	74.04	80.46	80.20	91.30	91.26	94.02	94.02	84.82	85.04	3.33
Ours(GIN)	84.18	83.80	**74.88**	74.16	80.32	80.94	91.79	91.71	**94.56**	**94.53**	85.16	**85.13**	2.17

Table 1: Results of node classification tasks on Cora, Citeseer, Pubmed, CS, Physics, and Computers

Method	Photo		Texas		Wisconsin		Cornell		Actor		OGBN-ArXiv		Avg.Rank
#Layes	32	64	32	64	32	64	32	64	32	64	32	64	
GCN	58.47	50.21	62.16	62.16	57.84	57.84	56.76	56.76	25.16	25.16	46.38	42.95	10.00
SGC	26.08	24.57	56.41	56.96	51.29	52.16	58.57	55.41	26.17	25.88	34.22	23.14	11.67
ChebNet	65.28	64,83	64.86	64.86	52.94	52.94	55.86	52.25	25.46	25.46	41.00	35.16	10.33
GAT	83.73	25.36	65.41	64.86	53.73	53.33	54.05	55.14	25.54	25.62	59.78	36.63	9.33
GIN	65.98	25.27	62.17	60.00	47.06	50.98	54.59	55.14	24.36	23.45	65.17	60.56	11.33
PairNorm	82.66	79.55	41.08	40.68	52.84	52.94	36.89	40.68	24.33	23.23	63.32	43.57	11.91
GCNII	62.95	65.12	69.19	65.41	70.31	59.02	74.16	56.92	34.28	34.64	72.60	70.07	5.33
JKNet	78.42	79.73	61.08	66.49	52.76	56.08	57.30	51.49	28.80	28.26	66.31	65.80	8.25
GPRGNN	91.74	91.28	62.27	61.08	71.35	64.90	58.27	52.16	29.88	32.43	70.18	69.98	6.25
DAGNN	89.96	87.86	57.68	60.27	50.84	51.76	58.43	52.43	27.73	25.45	71.46	70.58	8.92
APPNP	59.62	63.63	60.68	64.32	54.24	59.90	58.43	54.69	28.65	28.19	66.94	66.90	8.25
BernNet	91.59	16.53	61.08	16.76	63.53	24.71	52.43	11.89	28.19	20.66	45.16	37.18	11.92
Ours(GCN)	**92.06**	**92.03**	**85.41**	**84.86**	**83.14**	**84.71**	**82.70**	**82.62**	**38.42**	**38.49**	74.07	73.95	1.33
Ours(GAT)	91.98	92.00	78.92	78.38	73.73	74.90	77.84	81.62	34.97	35.54	74.37	74.41	2.50
Ours(GIN)	92.03	91.98	82.57	83.12	81.78	81.20	81.08	81.08	34.51	34.50	**75.02**	**74.85**	2.25

Table 2: Results of node classification tasks on Photo, heterophilous graphs (*e.g.*, Texas), and a large-scale graph OGBN-ArXiv

features. We adopt our standard model *Ours(GCN)* itself as the teacher and the original graph as the auxiliary graph. As observed from Fig. 1, our model outperforms most evaluated baselines in nearly all layers by a significant margin, showing its effective extraction and preservation of structural semantics. While SGC with no more than 64 layers can achieve better results in some layers, it suffers from over-smoothness severely with sufficient large layers (*e.g.*, 10^3 or 10^4 layers).

Ablation, Hyper-Parameter Studies, and Visualizations In order to demonstrate the effects of each individual part of our model and learning framework, we conduct extensive ablation studies following (Chen et al. 2020) and report the results on seven graph benchmarks in Tab. 3. In the following, we take *Ours(GCN)* as our standard model and independently drop each of the five parts: SoftGraphAIN (SG), residual connections (RC), R-SoftGraphAIN (R-SG), label smoothing (LS), and the holistic curriculum learning framework (CL), where *w.o. X* means that we drop the part *X*. From Tab. 3, we observe that every part contributes a portion to the performance gain, among which R-SG is the most significant since it reduces the risk of features and structure forgetting simultaneously. To facilitate a better understanding, we plot the varying effects of softly normalizing extents (the hyper-parameter a in Eq. 7) in Fig. 2, from which we can see a comprehensive ascending-and-then-descending trend, showing the benefits of this soft version compared to the hard one. In supplementary materials, we study some other hyper-parameters (*e.g.*, α and k_{KNN}), and detailedly visualize the embeddings produced by our method and some counterparts.

Discussion On the Time and Space Complexities The theoretical time complexity is analysed $O(Ld^2d_0)$ where $d, d_0 \ll n$ if partial-SVD or truncated-SVD (Halko, Martinsson, and Tropp 2011) is utilized. And it would become $O(Ld^3)$ with a full SVD decomposition. However, it is ef-

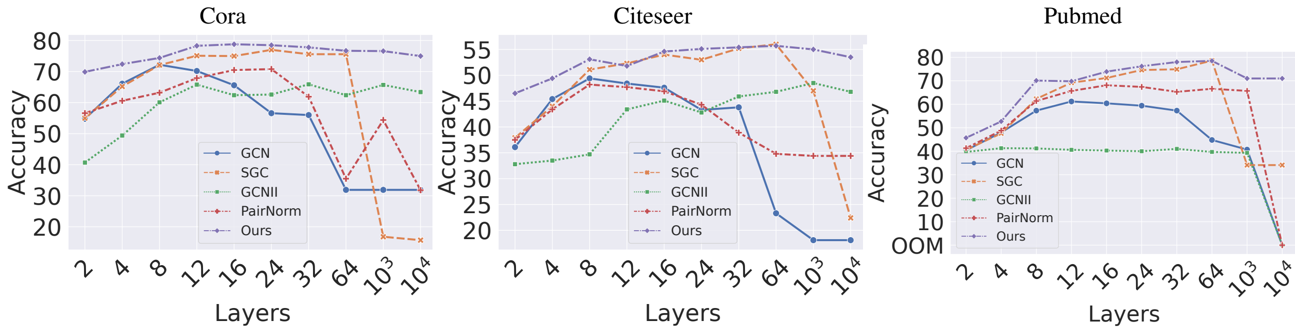

Figure 1: Results of different models with varying layers in node classification tasks with noisy features

Method	#Layers	Cora	Citeseer	Pubmed	CS	Physics	Computers	Photo
w.o. SG	32	83.30±0.12	72.98±0.50	80.26±0.98	91.79±0.07	94.23±0.25	82.34	90.26
	64	84.60±0.29	72.62±0.98	80.30±0.23	91.70±0.07	94.25±0.13	84.50	91.69
w.o. RC	32	82.88±0.79	72.82±1.01	78.84±0.59	91.40±0.26	92.99±0.19	83.78	91.42
	64	82.40±0.33	71.04±0.55	78.60±0.27	89.03±0.62	92.59±0.63	84.25	91.14
w.o. R-SG	32	40.22±5.71	29.32±3.47	46.28±4.91	51.61±18.23	77.20±11.95	61.52	83.77
	64	36.74±4.36	27.70±2.94	28.61±3.64	28.51±4.70	60.94±8.23	49.94	70.09
w.o. LS	32	83.96±0.65	73.64±0.68	81.04±0.21	92.02±0.05	94.10±0.11	84.52	91.50
	64	84.00±0.22	74.34±0.79	80.86±0.79	91.95±0.08	94.06±0.08	84.91	91.64
w.o. CL	32	82.40±0.65	73.20±0.77	79.34±0.39	89.60±0.20	92.46±0.61	84.00	90.58
	64	82.58±0.87	72.54±0.42	79.00±0.54	89.23±0.12	92.22±0.56	84.44	90.41
Ours(GCN)	32	85.12±0.15	74.42±0.26	81.28±0.18	92.50±0.09	94.45±0.06	85.21	92.06
	64	84.87±0.26	74.50±0.23	81.58±0.66	92.41±0.07	94.52±0.04	85.13	92.03

Table 3: Ablation studies on seven benchmarks including Cora, Citeseer, Pubmed, CS, Physics, Computers, and Photo

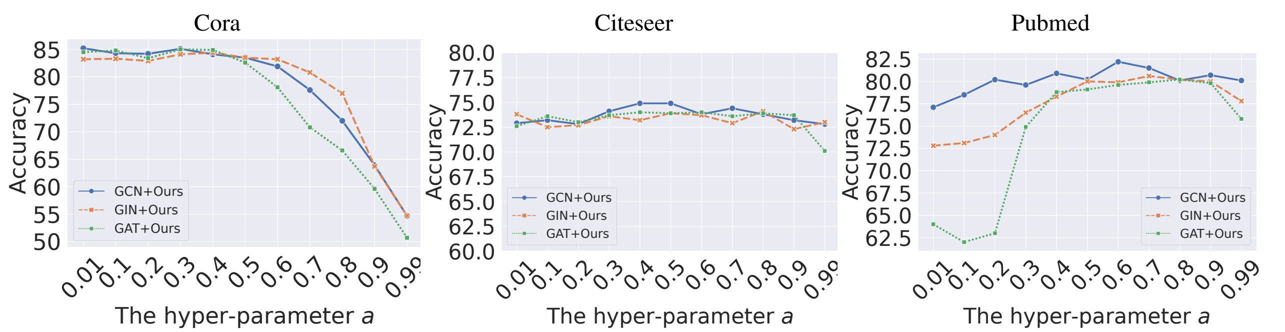

Figure 2: Results of Ours(GCN/GAT/GIN) against the varying hyper-parameter $a \in [0, 1]$ controlling the normalizing extent

ficient under the high-parallelizability implemented via Pytorch. The theoretical space complexity is $O\left(L\left(n+d\right)d\right)$.

5 Conclusion

In this paper, we propose *R-SoftGraphAIN* to alleviate the over-smoothness of deep GNNs, by novelly employing soft normalization of the covariance matrix with appropriately incorporated residual connections. We show in theory that the technique can maximally preserve the diversities of knowledge from both structures and features even at a sufficiently large depth against over-smoothness. Furthermore, in order to ease the difficulty of the optimization of deep GNNs, a label-smoothing-based curriculum learning framework (called *SmoothCurriculum*) is proposed to intuitively encourage the encoder to digest knowledge from low- to high-frequency and to learn to discriminate nodes from coarse to fine. Extensive experiments were carried out against semi-supervised node classification tasks to show the effectiveness of our model by demonstrating its practical performance gain compared to twelve state-of-the-art baselines on twelve real-world graph benchmarks. In future work, we will explore more applications of our method such as link prediction, graph classification, and community detection tasks.

Acknowledgements

This work is supported by the National Science Foundation of China (Nos. 12271098) and Taishan Scholars Young Expert Project of Shandong Province (No. tsqn202211215) and the University-Industry Cooperation Project of Fujian Province, China (2023H6008). The complete version of this paper can be found at https://arxiv.org/abs/2312.08221 with all codes at https://github.com/jslijin/Research-Paper-Codes/.

References

Bengio, Y.; Louradour, J.; Collobert, R.; and Weston, J. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, 41–48.

Bruna, J.; Zaremba, W.; Szlam, A.; and LeCun, Y. 2014. Spectral Networks and Locally Connected Networks on Graphs. arXiv:1312.6203.

Caubrière, A.; Tomashenko, N.; Laurent, A.; Morin, E.; Camelin, N.; and Estève, Y. 2019. Curriculum-based transfer learning for an effective end-to-end spoken language understanding and domain portability. In *20th Annual Conference of the International Speech Communication Association (InterSpeech)*, 1198–1202.

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.

Chen, T.; Zhou, K.; Duan, K.; Zheng, W.; Wang, P.; Hu, X.; and Wang, Z. 2022. Bag of Tricks for Training Deeper Graph Neural Networks: A Comprehensive Benchmark Study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, DOI:10.1109/TPAMI.2022.3174515.

Chien, E.; Peng, J.; Li, P.; and Milenkovic, O. 2021. Adaptive Universal Generalized PageRank Graph Neural Network. In *International Conference on Learning Representations*.

Choi, J.; Choi, H.; Hwang, J.; and Park, N. 2022. Graph neural controlled differential equations for traffic forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 6367–6374.

Chu, G.; Wang, X.; Shi, C.; and Jiang, X. 2021. CuCo: Graph Representation with Curriculum Contrastive Learning. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 2300–2306.

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.

Demir, A.; Koike-Akino, T.; Wang, Y.; Haruna, M.; and Erdogmus, D. 2021. EEG-GNN: Graph Neural Networks for Classification of Electroencephalogram (EEG) Signals. In *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, 1061–1067.

Fei, Z.; Zhang, Q.; and Zhou, Y. 2021. Iterative GNN-based decoder for question generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2573–2582.

Feng, W.; Zhang, J.; Dong, Y.; Han, Y.; Luan, H.; Xu, Q.; Yang, Q.; Kharlamov, E.; and Tang, J. 2020. Graph random neural networks for semi-supervised learning on graphs. *Advances in neural information processing systems*, 33: 22092–22103.

Gasteiger, J.; Bojchevski, A.; and Günnemann, S. 2019. Predict then Propagate: Graph Neural Networks meet Personalized PageRank. In *International Conference on Learning Representations*.

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural message passing for quantum chemistry. In *International conference on machine learning*, 1263–1272.

Guan, Y.; Zhang, J.; Tian, K.; Yang, S.; Dong, P.; Xiang, J.; Yang, W.; Huang, J.; Zhang, Y.; and Han, X. 2022. Node-Aligned Graph Convolutional Network for Whole-Slide Image Representation and Classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 18813–18823.

Hacohen, G.; and Weinshall, D. 2019. On the power of curriculum learning in training deep networks. In *International Conference on Machine Learning*, volume 97, 2535–2544.

Halko, N.; Martinsson, P. G.; and Tropp, J. A. 2011. Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions. *SIAM Review*, 53(2): 217–288.

Han, P.; Zhao, P.; Lu, C.; Huang, J.; Wu, J.; Shang, S.; Yao, B.; and Zhang, X. 2022. GNN-Retro: Retrosynthetic Planning with Graph Neural Networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 4014–4021.

He, M.; Wei, Z.; Xu, H.; et al. 2021. Bernnet: Learning arbitrary graph spectral filters via bernstein approximation. *Advances in Neural Information Processing Systems*, 34: 14239–14251.

Hu, W.; Fey, M.; Zitnik, M.; Dong, Y.; Ren, H.; Liu, B.; Catasta, M.; and Leskovec, J. 2020. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33: 22118–22133.

Huang, W.; Rong, Y.; Xu, T.; Sun, F.; and Huang, J. 2020. Tackling over-smoothing for general graph convolutional networks. arXiv:2008.09864.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations*.

Li, G.; Müller, M.; Ghanem, B.; and Koltun, V. 2021. Training graph neural networks with 1000 layers. In *International conference on machine learning*, 6437–6449.

Li, G.; Muller, M.; Thabet, A.; and Ghanem, B. 2019. Deepgcns: Can gcns go as deep as cnns? In *Proceedings of the IEEE/CVF international conference on computer vision*, 9267–9276.

Liu, F.; Xue, S.; Wu, J.; Zhou, C.; Hu, W.; Paris, C.; Nepal, S.; Yang, J.; and Yu, P. S. 2021. Deep learning for community detection: progress, challenges and opportunities. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, 4981–4987.

Liu, M.; Gao, H.; and Ji, S. 2020. Towards deeper graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 338–348.

Liu, X.; Lai, H.; Wong, D. F.; and Chao, L. S. 2020. Norm-Based Curriculum Learning for Neural Machine Translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 427–436.

Luan, S.; Zhao, M.; Chang, X.-W.; and Precup, D. 2019. Break the ceiling: stronger multi-scale deep graph convolutional networks. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 10945–10955.

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

Narvekar, S.; Peng, B.; Leonetti, M.; Sinapov, J.; Taylor, M. E.; and Stone, P. 2020. Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey. *Journal of Machine Learning Research*, 21(181): 1–50.

Oono, K.; and Suzuki, T. 2020. Graph Neural Networks Exponentially Lose Expressive Power for Node Classification. In *International Conference on Learning Representations*.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. Pytorch: An imperative style,high-performance deep learning library. *Advances in neural information processing systems*, 32: 8024–8035.

Pei, H.; Wei, B.; Chang, K. C.-C.; Lei, Y.; and Yang, B. 2020. Geom-GCN: Geometric Graph Convolutional Networks. In *International Conference on Learning Representations*.

Rong, Y.; Huang, W.; Xu, T.; and Huang, J. 2020. DropEdge: Towards Deep Graph Convolutional Networks on Node Classification. In *International Conference on Learning Representations*.

Sen, P.; Namata, G.; Bilgic, M.; Getoor, L.; Galligher, B.; and Eliassi-Rad, T. 2008. Collective classification in network data. *AI magazine*, 29(3): 93–93.

Shchur, O.; Mumme, M.; Bojchevski, A.; and Günnemann, S. 2018. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*.

Sinha, S.; Garg, A.; and Larochelle, H. 2020. Curriculum by smoothing. *Advances in Neural Information Processing Systems*, 33: 21653–21664.

Soviany, P.; Ionescu, R. T.; Rota, P.; and Sebe, N. 2022. Curriculum Learning: A Survey. *International Journal of Computer Vision*, 130(6): 1526–1565.

Tang, J.; Sun, J.; Wang, C.; and Yang, Z. 2009. Social influence analysis in large-scale networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 807–816.

Topping, J.; Giovanni, F. D.; Chamberlain, B. P.; Dong, X.; and Bronstein, M. M. 2022. Understanding over-squashing and bottlenecks on graphs via curvature. In *International Conference on Learning Representations*.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *International Conference on Learning Representations*.

Wang, C.; Wu, Y.; Liu, S.; Zhou, M.; and Yang, Z. 2020. Curriculum Pre-training for End-to-End Speech Translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 3728–3738.

Wang, X.; Chen, Y.; and Zhu, W. 2022. A Survey on Curriculum Learning. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 44(9): 4555–4576.

Wang, Y.; Wang, W.; Liang, Y.; Cai, Y.; and Hooi, B. 2021. Curgraph: Curriculum learning for graph classification. In *Proceedings of the Web Conference 2021*, 1238–1248.

Wu, F.; Souza, A.; Zhang, T.; Fifty, C.; Yu, T.; and Weinberger, K. 2019. Simplifying graph convolutional networks. In *International conference on machine learning*, 6861–6871.

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.

Xie, Y.; Lv, S.; Qian, Y.; Wen, C.; and Liang, J. 2022. Active and Semi-Supervised Graph Neural Networks for Graph Classification. *IEEE Transactions on Big Data*, 8(4): 920–932.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019. How Powerful are Graph Neural Networks? In *International Conference on Learning Representations*.

Xu, K.; Li, C.; Tian, Y.; Sonobe, T.; Kawarabayashi, K.-i.; and Jegelka, S. 2018. Representation learning on graphs with jumping knowledge networks. In *International conference on machine learning*, 5453–5462.

Yang, C.; Wang, R.; Yao, S.; Liu, S.; and Abdelzaher, T. 2020. Revisiting Over-smoothing in Deep GCNs. arXiv:2003.13663.

Yun, S.; Kim, S.; Lee, J.; Kang, J.; and Kim, H. J. 2021. Neognns: Neighborhood overlap-aware graph neural networks for link prediction. *Advances in Neural Information Processing Systems*, 34: 13683–13694.

Zhao, L.; and Akoglu, L. 2020. PairNorm: Tackling Oversmoothing in GNNs. In *International Conference on Learning Representations*.

Zhou, K.; Dong, Y.; Wang, K.; Lee, W. S.; Hooi, B.; Xu, H.; and Feng, J. 2021. Understanding and resolving performance degradation in deep graph convolutional networks. *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2728–2737.

Zhou, T.; Wang, S.; and Bilmes, J. 2020. Curriculum learning by dynamic instance hardness. *Advances in Neural Information Processing Systems*, 33: 8602–8613.