Graph Neural Networks without Propagation

Liang Yang
Qiuliang Zhang
yangliang@vip.qq.com
3463194784@qq.com
School of Artificial Intelligence
Hebei University of Technology
Tianjin, China

Runjie Shi
Wenmiao Zhou
shirunjie2020@163.com
zhou_wen_miao@163.com
School of Artificial Intelligence
Hebei University of Technology
Tianjin, China

Bingxin Niu
niubingxin666@163.com
School of Artificial Intelligence
Hebei University of Technology
Tianjin, China

Chuan Wang*
wangchuan@iie.ac.cn
State Key Laboratory of Information
Security, IIE, CAS
Beijing, China

Xiaochun Cao
caoxiaochun@mail.sysu.edu.cn
School of Cyber Science and
Technology, Shenzhen Campus of Sun
Yat-sen University
Shenzhen, China

Dongxiao He*
hedongxiao@tju.edu.cn
College of Intelligence and
Computing
Tianjin University
Tianjin, China

Zhen Wang
w-zhen@nwpu.edu.cn
School of Artificial Intelligence,
OPtics and ElectroNics (iOPEN),
School of Cybersecurity,
Northwestern Polytechnical
University
Xi'an, China

Yuanfang Guo
andyguo@buaa.edu.cn
School of Computer Science and
Engineering, Beihang University
Beijing, China

ABSTRACT

Due to the simplicity, intuition and explanation, most Graph Neural Networks (GNNs) are proposed by following the pipeline of message passing. Although they achieve superior performances in many tasks, propagation-based GNNs possess three essential drawbacks. Firstly, the propagation tends to produce smooth effect, which meets the inductive bias of homophily, and causes two serious issues: over-smoothing issue and performance drop on networks with heterophily. Secondly, the propagations to each node are irrelevant, which prevents GNNs from modeling high-order relation, and cause the GNNs fragile to the attributes noises. Thirdly, propagation-based GNNs may be fragile to topology noise, since they heavily relay on propagation over the topology. Therefore, the propagation, as the key component of most GNNs, may be the essence of some serious issues in GNNs. To get to the root of these issue, this paper attempts to replace the propagation with a novel *local* operation. Quantitative experimental analysis reveals: 1) the existence of low-rank characteristic in the node attributes from ego-networks and 2) the performance improvement by reducing its rank. Motivated by this finding, this paper propose the Low-Rank GNNs, whose key component is the low-rank attribute matrix approximation in ego-network. The graph topology is employed to construct the ego-networks instead of message propagation, which is sensitive to topology noises. The proposed Low-Rank GNNs posses some attractive characteristics, including robust to topology and attribute noises, parameter-free and parallelizable. Experimental evaluations demonstrate the superior performance, robustness to noises and universality of the proposed Low-Rank GNNs.

CCS CONCEPTS

• **Computing methodologies → Neural networks**.

KEYWORDS

Graph Neural Network, low-rank matrix approximation, RPCA, propagation
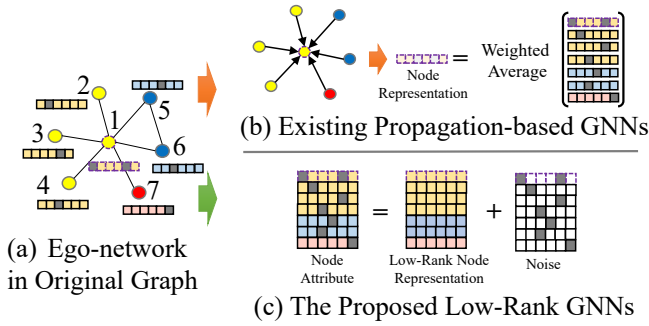
1 INTRODUCTION

Originated from spectral graph theory [14], graph neural networks (GNNs), which apply deep neural networks in graph domain, have become a powerful tool in modeling irregular data [37, 49]. They

---

*Corresponding authors.

(a) Ego-network in Original Graph

(b) Existing Propagation-based GNNs

(c) The Proposed Low-Rank GNNs

Figure 1: Comparison between existing propagation-based GNNs and the proposed Low-Rank GNNs without propagation in an ego-network. (a) Ego-network in original graph. Colors of nodes stand for categories, while center node, i.e. node 1, is highlight with purple dashed border. (b) Existing propagation-based GNNs are equivalent to weighted averaging of neighbourhood nodes. They may lose information due to mixing nodes from different categories. (c) The proposed Low-Rank GNNs employ low-rank matrix decomposition, which keeps critical information by taking high-order relationship between nodes in the ego-network.

boost the performance in many tasks, such as node classification [12, 13] and link prediction [47], and been widely employed by many fields, such as machine learning [30], computer vision [11], natural language processing [8, 9] and information retrieval [20, 25], etc. GNNs can be designed from two different perspectives, i.e., spectral filtering and spatial message passing. Many classic GNNs, such as GCN [21], ChebyNet [15] and CayleyNet [23] are motivated from spectral graph filtering. However, due to the simplicity, intuition and explanation, most GNNs are proposed by following the pipeline of message passing [16]. Recent work bridges the gap between spectral and spatial domains in GNNs by demonstrating the equivalence between them [1].

Many efforts on spatial GNNs have been paid on the propagation, i.e., what should be propagated and how to propagate message. For example, Graph Attention Network (GAT) [34] shows the propagation weighs can be learned via self-attention mechanism, while Diverse Message Passing (DMP) [43] constructs the message with the element-wise product of the attributes from the two connected nodes. Although they achieve superior performances in many tasks, propagation-based GNNs possess three essential drawbacks.

Firstly, the propagation tends to produce smooth effect [24], which meets the inductive bias of homophily [44], and causes two serious issues. 1) It may cause the over-smoothing issue when performs multiple propagation by stacking multiple layers [24]. 2) Its performance may significantly drop on networks with heterophily, which are common in real world. Take Figure 1(a) as an example. The ego-network consists of 7 nodes from 3 categories, which are marked with yellow, blue and red, respectively. Nodes from the same category possess the similar attributes. As shown in Figure 1(b), existing propagation-based GNNs are equivalent to weighted average of the neighbourhood nodes. Since the neighbourhood is mixed with nodes from multiple categories, averaging between them may cause the loss of critical information (yellow content) in obtained node representation.

Secondly, the propagations to each node are irrelevant [41], since propagation weights are either predefined according to the topology or learned based on the contents of the two connected nodes. It prevents GNNs from modeling high-order relation, and cause the GNNs fragile to the attributes noises. Actually, the representation of one node should reflect global characteristics of its ego-network.

Thirdly, propagation-based GNNs may be NOT robust to topology noise. It is widely-accepted that there exist large amount of noises. Topology structure significantly impacts the performance, since GNNs relay on propagation over the topology. Although many graph structure learning methods, which refine the given topology and then propagate on the refined topology, has been proposed, they are exposed to overfitting due to the high model complexity.

Therefore, the propagation, as the key component of most GNNs, may be the essence of some serious issues in GNNs. To get to the root of these issue, this paper attempts to replace the propagation with a novel *local* operation. Firstly, the low-rank characteristic of the node attributes, which is employed in transductive classification in machine learning [17] and many tasks in computer vision [18], in ego-network are investigated. Quantitative experimental analysis demonstrates: 1) the existence of low-rank characteristic and 2) the performance improvement by reducing rank.

Motivated by this finding, this paper propose the Low-Rank GNNs, whose key component is the low-rank attribute matrix approximation in ego-network. The graph topology is employed to construct the ego-networks instead of message propagation, which is sensitive to topology noises. Specifically, representation for each node is obtained from low-rank attribute matrix approximation in its ego-network, which is implemented via the RPCA [6] and optimized by Alternating Direction Methods of Multipliers (ADMM) [4]. Then, representations of all nodes are fed into a multilayer perceptron (MLP), which is trained with the supervision from labeled nodes. Beneficial from the remarkable performance of low-rank approximation and the novel way of utilizing topology, the proposed Low-Rank GNNs posses some attractive characteristics, including robust to topology and attribute noises, parameter-free and parallelizable. Besides, Low-Rank GNNs can handle networks with heterophily by avoiding the smooth effect in propagation.

The main contributions of this paper are summarized as follows:

- We observe the low-rank characteristic of the collection of attributes in the ego-network for the first time.
- We propose a novel Low-Rank GNNs by replacing the attribute propagation in ego-network with low-rank approximation of attribute matrix.
- We analyze the attractive characteristics of the proposed Low-Rank GNNs, including robust to topology and attribute noises, handling networks with heterophily, parameter-free and parallelizable.
- We experimentally verify the effectiveness, robustness and universality of the Low-Rank GNNs.

2 PRELIMINARIES

This section provides the notations used in this paper, reviews the previous work on graph neural networks, and introduces the basic concept on low-rank matrix approximation, which forms the basis of the current work.

2.1 Notations

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a graph with node set $\mathcal{V} = \{v_1, v_2, \cdots, v_N\}$ and edge set $\mathcal{E}$, where N is the number of nodes. The topology of graph $\mathcal{G}$ can be represented by its adjacency matrix $\mathbf{A} = [a_{ij}] \in \{0, 1\}^{N \times N}$, where $a_{ij} = 1$ if and only if there exists an edge $e_{ij} = (v_i, v_j)$ between nodes v_i and v_j. The degree matrix $\mathbf{D}$ is a diagonal matrix with diagonal element $d_i = \sum_{i=1}^{N} a_{ij}$ as the degree of node v_i. $\mathcal{N}(v_i) = \{v_j | (v_i, v_j) \in \mathcal{E}\}$ stands for the neighbourhoods of node v_i. Let $\mathcal{G}_i = (\mathcal{V}_i, \mathcal{E}_i)$ represents the ego-network around node v_i, where $\mathcal{V}_i = \mathcal{N}(v_i) \cup v_i$ and $\mathcal{E}_i$ denotes edges between nodes in $\mathcal{V}_i$. $\mathbf{X} \in \mathbb{R}^{N \times F}$ and $\mathbf{H} \in \mathbb{R}^{N \times F'}$ denote the collections of node attributes and representations with the ith rows, i.e., $\mathbf{x}_i \in \mathbb{R}^F$ and $\mathbf{h}_i \in \mathbb{R}^{F'}$, corresponding to node v_i, where F and F' stand for the dimensions of attribute and representation. For convenience, $\mathbf{X}_i \in \mathbb{R}^{(d_i+1) \times F}$ and $\mathbf{H}_i \in \mathbb{R}^{(d_i+1) \times F'}$ denote the collections of node attributes and representations of ego-network around node v_i, i.e., $\mathcal{G}_i$ Note that $\mathbf{X} \in \mathbb{R}^{N \times F}$ and $\mathbf{H} \in \mathbb{R}^{N \times F'}$ can be used to represent the collections of data samples and their representations on non-graph dataset, respectively.

2.2 Graph Neural Networks

Most of the Graph Neural Networks (GNNs) follow an aggregation-combination strategy [16], where each node representation is iteratively updated by aggregating node representations in the local neighbourhoods and combining the aggregated representations with the node representation itself as

$$\bar{\mathbf{h}}_v^k = \text{AGGREGATE}^k \left(\left\{ \mathbf{h}_u^{k-1} | u \in \mathcal{N}(v) \right\} \right), \tag{1}$$

$$\mathbf{h}_v^k = \text{COMBINATE}^k \left(\mathbf{h}_v^{k-1}, \bar{\mathbf{h}}_v^k \right), \tag{2}$$

where $\bar{\mathbf{h}}_v^k$ stands for the aggregated representation from local neighbourhoods. Besides of the concatenation based implementation, such as GraphSAGE [19] and H2GCN [50], averaging (or summation) has been widely adopted to implement $\text{COMBINATE}^k(\cdot, \cdot)$, such as GCN [21], GAT [34], GIN [38], etc. Except for the MAX and LSTM implementations in GraphSAGE [19], most of the GNNs utilize averaging function to implement AGGREGATE^k. Therefore, they can be unified as

$$\mathbf{h}_v^k = \sigma \left(\left(c_{vv}^k \mathbf{h}_v^{k-1} + \sum_{u \in \mathcal{N}(v)} c_{uv}^k \mathbf{h}_u^{k-1} \right) \mathbf{W}^k \right), \tag{3}$$

where $\mathbf{W}^k$ represents the learnable parameters and $\sigma(\cdot)$ denotes the nonlinear mapping function. Note that the scalar c_{uv} is the averaging weight. For example, GCN [21] sets $c_{uv}^k = 1/(\sqrt{(d_u + 1)(d_v + 1)})$, GIN [38] sets $c_{uv}^k = 1$ for $u \neq v$ and $c_{vv}^k = 1 + \epsilon^k$, and GAT [34] learns non-negative c_{uv}^k based on the attention mechanism. Recently, to handle the network with heterophily via high-passing filtering, GPRGNN [13] sets $c_{uv}^k = \gamma^k/(\sqrt{(d_u + 1)(d_v + 1)}$ with γ^k being a learnable real value, while FAGCN [2] directly relaxes the learnable c_{uv} in GAT to real value.

2.3 Low-Rank Matrix Approximation

Since most real data are corrupted with noise, how to remove noise and reveal the structure of the data is a critical problem in many areas, such as signal processing, computer vision, pattern recognition and social modeling [10, 48]. Low-rank matrix approximation is a class of widely-used methods to find the underlying structure of the given data. They operate under the assumptions that the underlying structure of the data lies on a low dimensional subspace and the high dimension of the observed data is often due to noises. If the rank of the low dimensional subspace is known, matrix factorization strategy can be employed. It factorizes data matrix $\mathbf{X} \in \mathbb{R}^{N \times F}$ into two low-dimensional matrices $\mathbf{U} \in \mathbb{R}^{N \times K}$ and $\mathbf{V} \in \mathbb{R}^{F \times K}$ by minimizing $\text{argmin}_{\mathbf{U}, \mathbf{V}} \text{Dis}(\mathbf{X}, \mathbf{UV}')$, where $k \ll \min(m, n)$ is the dimension of the latent space or the rank of the underlying data and $\text{Dis}(\mathbf{X}, \mathbf{UV}')$ denotes the error between the original data $\mathbf{X}$ and the reconstructed data $\mathbf{UV}'$ under some specific distance metric, such as KL-divergence, ℓ_1 norm and Frobenius norm. The most serious limitation of this kind of methods is that the dimension of the latent space, i.e., K, must be pre-determined. In reality, nevertheless, it is often difficult to determine K in advance.

When the rank of the data is not given, we may directly approximate $\mathbf{X} \in \mathbb{R}^{N \times F}$ with a low-rank matrix $\mathbf{H} \in \mathbb{R}^{N \times F}$, which is the idea of low-rank approximation [6], by minimizing

$$\underset{\mathbf{H}}{\text{argmin}} \, \text{Dis}(\mathbf{X}, \mathbf{H}) + \lambda \, \text{rank}(\mathbf{H}),$$

where $\text{rank}(\mathbf{H})$ is the rank of the matrix $\mathbf{H}$ and λ is a parameter for tradeoff between the two terms. Since the rank function is nonconvex, we can alternatively minimize its convex surrogate as

$$\underset{\mathbf{H}}{\text{argmin}} \, \text{Dis}(\mathbf{X}, \mathbf{H}) + \lambda ||\mathbf{H}||_*, \tag{4}$$

where $||\mathbf{H}||_*$ is the trace norm of $\mathbf{H}$, i.e., the sum of the singular values of $\mathbf{H}$. The most well-known approach for low-rank approximation with unknown rank is Robust PCA (RPCA) [6], which has been used for background subtraction, texture repair and subspace segmentation. RPCA uses the ℓ_0 norm to measure the difference between the original data $\mathbf{X}$ and the low-rank approximation $\mathbf{H}$, i.e., $\text{Dis}(\mathbf{X}, \mathbf{H}) = ||\mathbf{X} - \mathbf{H}||_0$, where $||\mathbf{X}||_0$ is the number of nonzeros in $\mathbf{X}$. As before, RPCA directly optimizes the $||\mathbf{X}||_1$ for its convexity instead of $||\mathbf{X}||_0$ and for the equivalence of the following two problems under rather broad conditions (the error matrix $\mathbf{X} - \mathbf{H}$ is sufficiently sparse relative to the rank of $\mathbf{H}$) [6]:

$$\underset{\mathbf{H}}{\text{argmin}} \, ||\mathbf{X} - \mathbf{H}||_0 + \lambda \, \text{rank}(\mathbf{H}),$$

$$\underset{\mathbf{H}}{\text{argmin}} \, ||\mathbf{X} - \mathbf{H}||_1 + \lambda ||\mathbf{H}||_*. \tag{5}$$

Since the nosies have been separated from data, the obtained low-rank approximation $\mathbf{H}$ achieves better performance and generalization on many tasks, such as subspace clustering [26] and background extraction [7].

3 METHODOLOGY

In this section, the Graph Neural Networks without propagation are given. Firstly, the low-rank characteristic of node attributes in ego-network, which motivates the proposed method, is analyzed. Secondly, the ego-network low-rank approximation is proposed to replace the widely-used propagation. Finally, the Low-Rank

(a) Low-rank Structure Destroy
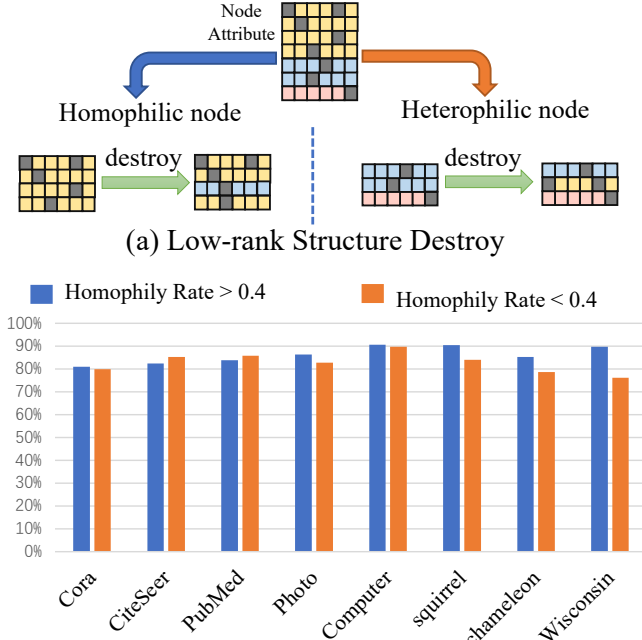
(b) Percentage of nodes with nuclear norm increased

Figure 2: Low-rank characteristic of the collection of attributes in ego-networks. (a) Low-rank structure destroy procedure. (b) Percentage of nodes with nuclear norm increased on 8 networks. The blue and orange bars stand for homophilic nodes (homophily rate > 0.4) and heterophilic nodes (homophily rate < 0.4), respectively.

Graph Neural Networks are introduced based on the proposed ego-network low-rank approximation.

3.1 Analysis and Motivations

This section first investigate whether the low-rank characteristic exists in ego-network. Then, it verifies whether the low-rank approximation boosts the performance.

3.1.1 Low-rank characteristic. This subsection, we tend to demonstrate the low-rank characteristic of the collection of attributes in ego-networks. In some computer vision tasks, such as background subtraction [6], the low-rank characteristic is very obvious. Unfortunately, it is difficult to directly show whether a collection attributes possesses this characteristic, although it has been employed in transductive classification [17]. To this end, we alternatively investigate whether the rank will increase when the low-rank structure is destroy. However, the rank of matrix is sensitive to floating-point operation, the rank of matrix is replaced with its convex surrogate, i.e., nuclear norm, as in the RPCA (Eq. (4)), which is the singular-values summation of the matrix.

The remaining problem is how to destroy the low-rank structure. The main assumption is that the collection of attribution from nodes, which belong to the same category, should possess the low-rank structure. The procedure is shown in Figure 2(a). For stable experiments, the nodes are divided into homopilic nodes and heterophilic ones, according to its local homophily rate [28]. Besides, the isolated nodes are removed.
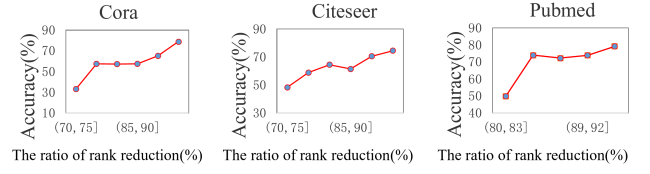
Figure 3: The percentage of nodes with performance improved as the percentage of rank reduced.

For homophilic nodes, i.e. nodes with homphily rate large than 0.4, the attribute of nodes, which belong to the same class as central node, are collected as low-rank structure, such as the yellow matrix in 2(a). The destruction procedure is to randomly replace one row with attribute of nodes from other classes, such as the yellow matrix with one blue row in Figure 2(a).

For heterophilic nodes. i.e. nodes with homphily rate smaller than 0.4, the attribute of nodes, which do NOT belong to the same class as central node, are collected as low-rank structure, such as the blue-pink matrix in 2(a). The destruction procedure is to randomly replace one row with attribute of central node, such as the matrix with one blue row, one yellow row, and one pink row in Figure 2(a).

We investigate the percentage of nodes where low-rank structure destruction cause an increase in nuclear norm. The results are shown in Figure 2(b). It can be observed that most of the nodes have nuclear norm increased. This indicates that most collections of attribute from eg-network benefit from low-rank recovery.

3.1.2 Performance Improvement. Next, we investigate whether the rank reduction boost the performance on 3 citation networks. To this end, we check whether the amount of rank reduced is related to performance improvement. Figure 3 gives the percentage of nodes with performance improvement calculated with the percentage of rank reduced. The positive correlation can be observed. Therefore, the rank reduction can boost the performance.

3.2 Ego-network Low-Rank Approximation

As discussion in the previous sub-section, the propagation in ego-network may cause the smooth effect, which leads to the loss of critical information and performance drop in networks with heterophily. To alleviate this issue, the low-rank matrix approximation is employed to model the characteristic of attributes in ego-network. As given in Section 2.1 (Notation), $\mathbf{X}_i \in \mathbb{R}^{(d_i+1) \times F}$ and $\mathbf{H}_i \in \mathbb{R}^{(d_i+1) \times F'}$ denote the collections of node attributes and representations of ego-network around node v_i, i.e. $\mathcal{G}_i$. Therefore, the formula of low-rank matrix approximation in Eq. (5) can be applied on $\mathbf{X}_i$, i.e.

$$\underset{\mathbf{H}_i}{\operatorname{argmin}} ||\mathbf{H}_i||_* + ||\mathbf{X}_i - \mathbf{H}_i||_1. \tag{6}$$

To facilitate the optimization, by denoting $\mathbf{S}_i = \mathbf{X}_i - \mathbf{H}_i$, Eq. (6) can be reformulated as

$$\underset{\mathbf{H}_i, \mathbf{S}_i}{\operatorname{argmin}} ||\mathbf{H}_i||_* + \lambda ||\mathbf{S}_i||_1 \tag{7}$$

$$s.t. \quad \mathbf{X}_i = \mathbf{H}_i + \mathbf{S}_i \tag{8}$$

By applying the Augmented Lagrangian Methods (ALM) [3], the constrained optimization problem in Eqs. (7) and (8) can be converted to

$$\mathcal{F}(\mathbf{H}_i, \mathbf{S}_i, \mathbf{Y}_i, \mu) = ||\mathbf{H}_i||_* + \lambda ||\mathbf{S}_i||_1 \quad + \quad < \mathbf{Y}_i, \mathbf{X}_i - \mathbf{H}_i - \mathbf{S}_i >$$
$$+ \quad \frac{\mu}{2} ||\mathbf{X}_i - \mathbf{H}_i - \mathbf{S}_i||_F^2, \quad (9)$$

where $\mathbf{Y}_i$ is the Lagrange multipliers for the constraint $\mathbf{X}_i = \mathbf{H}_i + \mathbf{S}_i$. $< \mathbf{Y}, \mathbf{X} > = tr(\mathbf{Y}\mathbf{X}')$ stands for the inner-product of matrix $\mathbf{Y}$ and $\mathbf{X}$. This objective function can be optimized via Alternating Direction Methods of Multipliers (ADMM) [4]. ADMM alternatively updates $\mathbf{H}_i$, $\mathbf{S}_i$ and $\mathbf{Y}_i$ by fixing others.

Update $\mathbf{S}_i$. By fixing $\mathbf{H}_i$ and $\mathbf{Y}_i$, Eq (9) can be converted to

$$\underset{\mathbf{S}_i}{\text{argmin}} \frac{\lambda}{\mu} ||\mathbf{S}_i||_1 + \frac{1}{2} ||\mathbf{Q}_i - \mathbf{S}_i||_F^2, \quad (10)$$

$$\mathbf{Q}_i = \mathbf{X}_i - \mathbf{H}_i - \frac{1}{\mu}\mathbf{Y}_i. \quad (11)$$

Eq. (10) has a closed-form element-wise solution via soft-thresholding [3, 40] as follows:

$$(\mathbf{S}_i)_{jk}^* = soft\left((\mathbf{Q}_i)_{jk}, \frac{\lambda}{\mu}\right) = sign\left((\mathbf{Q}_i)_{jk}\right) \cdot \max\left\{(\mathbf{Q}_i)_{jk} - \frac{\lambda}{\mu}, 0\right\} \quad (12)$$

where the soft-thresholding function is

$$soft(x, \lambda) = sign(x)\max\{|x| - \lambda, 0\} \quad (13)$$

Update $\mathbf{H}_i$. By fixing $\mathbf{S}_i$ and $\mathbf{Y}_i$, Eq (9) can be converted to

$$\underset{\mathbf{H}_i}{\text{argmin}} \frac{1}{\mu} ||\mathbf{H}_i||_* + \frac{1}{2} ||\mathbf{P}_i - \mathbf{H}_i||_F^2, \quad (14)$$

$$\mathbf{P}_i = \mathbf{X}_i - \mathbf{S}_i - \frac{1}{\mu}\mathbf{Y}_i. \quad (15)$$

Eq. (14) has a closed-form with singular-value thresholding [5] as

$$\mathbf{H}_i^* = \mathbf{U}_i soft\left(\Sigma_i, \frac{1}{\mu}\right)\mathbf{V}_i', \quad (16)$$

where $\mathbf{X}_i = \mathbf{U}_i\Sigma_i\mathbf{V}_i'$ is the singular-value decomposition (SVD) of matrix $\mathbf{X}_i$, and $soft(\cdot, \cdot)$ is the soft-thresholding [40] as in Eq. (13). The Lagrange multipliers $\mathbf{Y}_i$ can be update as

$$\mathbf{Y}_i = \mathbf{Y}_i + \mu(\mathbf{X}_i - \mathbf{H}_i - \mathbf{S}_i). \quad (17)$$

By gradually increasing μ and repeatedly update $\mathbf{H}_i$, $\mathbf{S}_i$ and $\mathbf{Y}_i$ until convergence, the low-rank matrix $\mathbf{H}_i^*$ can be obtained. Then, the row in $\mathbf{H}_i^*$ corresponding to the node v_i is extracted as $\mathbf{h}_i^*$ as the final representation of node v_i.

Remark 1: The main procedure to obtain the low-rank approximation is the singular-value thresholding in Eq. (16) . It is the low passing spectral filter in attribute space. Therefore, it explores the high-order relationship in ego-network, which is beyond the propagation in existing GNNs (pairwise relationship).

3.3 Low-Rank Graph Neural Networks

In previous section, one node representation is obtained via low-rank approximation in ego-network around it. By traversing the entire graph, extracting ego-network around all nodes, and performing low-rank approximation on each ego-network to obtain embedding of central node, all the node representations can be obtained. Then, the Low-Rank Graph Neural Networks (Low-Rank GNNs) are constructed by feeding the embeddings of all nodes

into a multilayer perceptron (MLP), and trained by using the cross-entropy objective function.

3.4 Insights and Discussions

This section provides some insights and discussion towards robustness to topology and attribute noises, the characteristics of parameter-free and parallelizable, local and global information, and difference from existing Low-rank strategy in GNNs.

3.4.1 Robustness to Topology Noises. It is widely-accepted that there exist large amount of noises, i.e. edges between nodes from different classes. Topology structure significantly impacts the performance, since GNNs relay on propagation over the topology. Thus, the graph structure learning is an important topic in GNNs [35, 42]. Most existing graph structure learning methods focus on correcting and refining the existing topology and then propagating on the refined topology. However, the graph structure learning may increase the model complexity of GNNs, and thus is exposed to overfitting. Alternatively, the proposed Low-rank GNNs is without propagation. The give topology is only used to form the ego-network. Therefore, Low-rank GNNs tend to be robust to topology noises.

3.4.2 Robustness to Attribute Noises. Although the smooth effect of the propagation in existing GNNs can be robust to attribute noises, its also leads to the loss of information due to the propagation between two nodes, i.e., individual pairwise relationships. In contrary, the proposed Low-rank GNNs de-noise the node attributes by jointly considering all nodes in the ego-network. That is the denoising in Low-rank GNNs may take the high-order relationship in the ego-network. Therefore, Low-rank GNNs may be more robust to attribute noises.

3.4.3 Parameter-free. Note that above low-rank matrix approximation is parameter-free. Thus, this procedure does not need to be learned with label supervision, and can be perform in advance. This attractive characteristic is also possessed by SGC [36]. However, the proposed Low-rank GNNs are powerful than SGC, since it explores the diverse characteristic in different ego-networks.

3.4.4 Parallelization. Most existing GNNs need repeatedly propagation. Therefore, the neighbourhood explosion issue [?] prevents them from being parallelized. Fortunately, the proposed Low-rank GNNs only need the information of ego-network to obtain representation, and thus are easy to beparallelized.

4 EVALUATIONS

In this section, the performance of our proposed Low-Rank GNNs is experimentally evaluated on node classification task. Then, the robustness analysis of Low-Rank GNNs is provided for intuitive understanding. Finally, the comparison is conducted between existing propagation-based GNNs and the proposed Low-Rank GNNs to illustrate the superiority of low-rank approximation.

4.1 Experimental Setup

4.1.1 Datasets and splitting. For node classification task, four kinds of datasets are adopted to comprehensively evaluate the proposed Low-Rank GNNs. The statistics of datasets are shown in Table 1.

Table 1: Benchmark dataset statistics for node classification.

Dataset	Cora	Citeseer	Pubmed	Computer	Photo	Chameleon	Squirrel	Actor	Cornell	Texas	Wisconsin
# Nodes	2,708	3,327	19,717	13,752	7,650	2,277	5,201	7,600	183	183	251
# Edges	5,429	4,732	44,338	245,861	119,081	36,101	217,073	33,544	295	309	499
# Features	1,433	3,703	500	767	745	2,325	2,089	931	1,703	1,703	1,703
# Classes	7	6	3	10	8	5	5	5	5	5	5

Table 2: Mean Classification Accuracy on Heterophilic Datasets (Bold indicates the best, underlined indicates the second best).

Dataset	Chameleon	Squirrel	Actor	Cornell	Texas	Wisconsin
MLP	49.67±0.78	37.04±0.46	34.10±0.25	81.08±6.37	81.89±4.78	85.29±3.61
GCN	28.18±0.23	23.96±0.26	26.86±0.23	55.14±7.57	55.68±9.61	58.42±5.10
GAT	42.93±0.28	42.93±0.28	28.45±0.23	58.92±3.32	58.38±4.45	55.29±8.71
SGC	<u>63.02±0.43</u>	43.14±0.28	29.39±0.20	47.80±1.50	55.18±1.17	54.31±2.10
GCNII	60.61±2.00	37.85±2.76	36.18±0.61	74.86±2.73	69.46±1.86.	74.12±1.62
APPNP	54.30±0.34	33.29±1.72	31.71±0.70	**82.16±3.83**	82.43±1.72	84.51±2.40
JKNet	62.31±2.76	44.24±2.11	36.47±0.51	56.49±3.22	65.35±4.68	51.37±3.21
GPR-GNN	**67.48±1.98**	<u>49.93±1.34</u>	36.58±1.04	79.73±3.91	77.84±2.78	82.55±1.67
FAGCN	61.12±1.95	40.88±2.02	<u>36.81±0.26</u>	67.95±10.02	61.82±8.71	76.93±3.46
H2GCN-1	57.11±1.58	36.42±1.89	35.86±1.03	<u>82.16±4.80</u>	<u>84.86±6.77</u>	**86.67±4.69**
H2GCN-2	59.39±1.98	37.90±2.02	35.62±1.30	82.16±6.00	82.16±5.28	<u>85.88±4.22</u>
DMP	55.92±1.26	47.26±1.54	35.86±0.46	70.27±1.74	78.38±2.14	80.39±2.27
Low-Rank GNNs	62.71±2.06	**51.70±1.28**	**38.51±0.33**	78.48±0.36	**88.10±1.27**	83.13±1.90

- **Citation Networks.** Cora, Pubmed and Citeseer are citation networks originally introduced in [27, 31], which are among the most widely used benchmarks for semi-supervised node classification.
- **Co-purchase Networks.** Computers and Photo are two networks of Amazon co-purchase relationships [32]. In these networks, nodes represent goods and edges stand for the connected two goods being frequently bought together.
- **Web page Networks.** Squirrel and Chameleon are subgraphs of web pages in Wikipedia discussing the corresponding topics, collected by [29]. Texas, Wisconsin and Cornell are graphs representing links between web pages of the corresponding universities, originally collected by the CMU WebKB project.
- **Co-occurrence Networks.** Actor is a graph representing actor co-occurrence in Wikipedia pages, processed by [28] based on the film-director-actor-writer network in [33].

For Cora, Citeseer and Pubmed, we use 20 labeled nodes per class for training, 500 nodes for validation and 1000 nodes for testing. Details can be found in [2]. For Computer and Photo, we use 20 labeled nodes per class for training, 30 nodes per class for validation and the rest nodes for testing. For Chameleon, Squirrel, Actor, Cornell, Texas and Wisconsin, we randomly split nodes of each class in to 60%, 20% and 20% for training, validation and testing, and run on test sets over 10 random splits, as suggested in [28].

4.1.2 Baselines. To verify the effectiveness of the proposed Low-Rank GNNs on node classification task, 11 methods are employed as the baselines with default hyper-parameters. They are divided into 3 categories:

- Classic GNN models for node classification task include vanilla GCN [21], GAT [34] and SGC [36].
- Deep GNNs designed to tackle over-smoothing issue include GCNII [12], APPNP [22] and JKNet [39].
- Models designed for networks with heterophily include GPR-GNN [13], FAGCN [2], H2GCN [50] and DMP [43].

4.1.3 Parameter settings. We implement Low-Rank GNNs based on Pytorch. To ensure fair comparisons, we set the hidden size as 64, the learning rate lr=0.01 and dropout rate d=0.3 for all the models. Besides, early stopping with a patience of 200 epochs and $L2$ regularization with coefficient of 0.01 is employed to prevent overfitting. We set the model layer K=2 of Low-Rank GNNs on Cora and Citeseer, and set K=1 for the other datasets.

4.2 Node Classification

4.2.1 Results analysis. Results on homophilic datasets are summarized in Table 3 and Results on homophilic datasets are summarized in Table 2. It can be observed that the proposed Low-Rank GNNs achieves new remarkable state-of-the-art results on Citeseer, Computer, Photo, Squirrel, Actor and Texas, which demonstrates the superiority of it.

To validate the performance of Low-Rank GNNs, three strong baselines: GCNII, APPNP and JKNet, which can act as deep models, are compared with Low-Rank GNNs. Note that Low-Rank GNNs significantly outperforms other state-of-the-art deep models e.g., the accuracy of LRGNN are 12.21% and 8.18% higher than those of APPNP on Computer and Photo, and obtains competitive results on most real-world datasets. This demonstrates that shallow-layer

Table 3: Mean Classification Accuracy on Homophilic Datasets (Bold indicates the best, underlined indicates the second best).

Dataset	Cora	Citeseer	Pubmed	Computer	Photo
MLP	58.20±2.10	59.10±2.30	70.00±2.10	44.90±5.80	69.60±3.80
GCN	81.50±1.30	70.30±0.28	77.80±2.90	76.30±2.40	87.30±1.20
GAT	81.80±1.30	70.80±0.26	78.50±0.27	78.00±1.90	85.70±1.70
SGC	81.00±0.00	71.90±0.10	78.90±0.00	74.40±0.01	86.40±0.00
GCNII	**85.50±0.50**	73.40±0.60	**80.20±0.40**	57.11±13.92	63.03±4.43
APPNP	83.30±0.00	71.80±0.00	80.10±0.00	71.69±4.67	83.62±3.73
JKNet	81.10±0.00	69.80±0.00	78.10±0.00	64.08±2.10	78.10±7.07
GPR-GNN	80.55±1.05	68.57±1.22	77.02±2.59	81.71±2.84	91.58±0.87
FAGCN	77.80±0.66	69.81±0.80	76.74±0.66	77.47±2.70	87.61±4.80
H2GCN-1	79.63±0.11	65.75±0.49	77.60±0.14	OOM	OOM
H2GCN-2	80.23±0.20	69.97±0.66	78.79±0.30	OOM	OOM
DMP	80.41±1.48	71.08±1.21	76.29±2.44	71.90±1.84	82.37±1.86
Low-Rank GNNs	82.10±0.24	**73.91±1.03**	78.50±1.20	**83.90±0.90**	**91.80±1.04**

information is actually quite abundant for extracting node representation as for Low-Rank GNNs.

We also compare the results of proposed Low-Rank GNNs with GPR-GNN, FAGNN, H2GCN and DMP, which are all the GNNs designed for processing datasets with heterophily. It can be observed that Low-Rank GNNs achieves new state-of-the-art results on Squirrel, Actor and Texas, which are three heterophilic datasets. That is because the low-rank approximation remain the heterophilic information from the ego-networks instead of the impairing heterophilic information as processing by averaging operation.

These results suggest that by adopting the low-rank approximation, our proposed Low-Rank GNNs is more effective and universal than the previous models on processing datasets with both homophily and heterophily for node classification.

4.2.2 Visualization. To provide an intuitive interpretation, the t-SNE visualizations of node embeddings obtained by GCN,GAT and Low-Rank GNNs on four datasets are given in figure4. The clusters of embeddings of nodes from different classes are marked with various colors. The shapes of these clusters reflect the characteristics of the corresponding models. The clusters of embeddings of different classes processed by GCN are overlapped, which demonstrates that GCN tends to be under-fitting. The clusters of embeddings obtained by GAT are quite sharp, which indicates that labeled data plays a very essential rule for the embedding and tends to be overfitting, while the clusters of embedding obtained by Low-Rank GNNs are more regular and the nodes with the same label exhibit spatial clustering, which shows the discriminative power of Low-Rank GNNs.

4.3 Propagation v.s. Low-Rank Approximation

In this section, the comparison is conducted between existing propagation-based GNNs and the proposed Low-Rank GNNs to illustrate the superiority of low-rank approximation and the adequacy of shallow-layer information. Existing propagation-based GNNs are equivalent to weighted average of the neighbourhood nodes.The results on both homophilic and heterophilic datasets
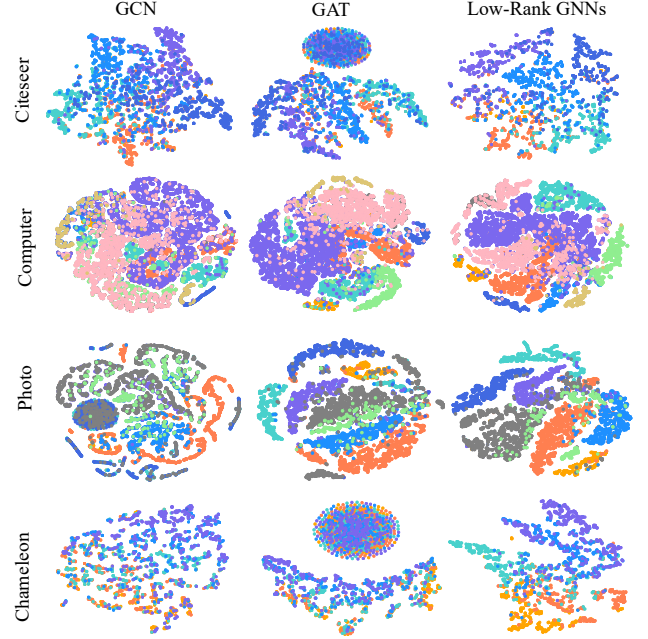
Figure 4: The visualization for node representations obtained by GCN, GAT and Low-Rank GNNs in a 2-D space. Node colors denote node labels.

are shown in Figure 7. Coauthor-CS (CS) and Coauthor-Physics (Physics) are two co-author networks based on the Microsoft Academic Graph from the KDD Cup 2016 challenge[32], which are two homophilic datasets. In this figure, Low-Rank GNNs (k=1) stands for processing 1-hop neighbors of each node by low-rank approximation, Low-Rank GNNs(k=2) stands for processing 2-hop neighbors of each node by low-rank approximation, AVG (k=1) stands for processing 1-hop neighbors of each node by averaging , and AVG (k=2) stands for processing 2-hop neighbors of each node by averaging.

It can be observed that the first-order information is actually quite abundant for extracting node representation as for Low-Rank GNNs. It is worth noting that the second-order information makes the performance much worse as for Low-Rank GNNs. The essential reason is that low-rank property often exists in local regions and the global low-rank approximation ignores the graph topology information.

Meanwhile, it is obvious that the model with low-rank approximation obtains better results than averaging processing. That is because the neighbourhood is mixed with nodes from multiple categories, averaging between them may cause the loss of critical information in obtained node representation, which leads a wide difference between the Low-Rank GNNs and the existing GNNs which employ the averaging processing on heterophilic datasets. This attractive property also reveals that Low-Rank GNNs can handle the datasets with heterophiliy.

4.4 Robustness Analysis

In this section, we investigate the robustness of Low-Rank GNNs with randomly adding noisy edges and attributes, and compare the performance of Low-Rank GNNs with GCN and GAT on Cora, Citeseer and Pubmed.

Figure 5: Node classification performace on graphs with randomly adding noisy edges.

Figure 6: Node classification performace on graphs with randomly adding noisy attributes.
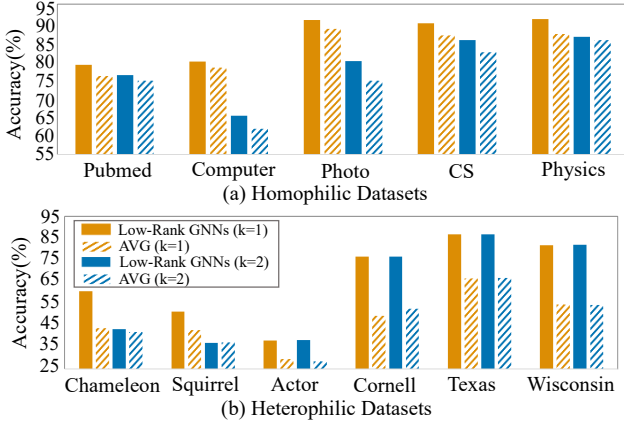

Figure 7: The comparison between existing propagation-based GNNs and Low-Rank GNNs on node classification task on both homophilic and heterophilic datasets.

As the noisy edges increase, the performance of GCN and GAT decrease significantly, which indicates the over-fitting issue. while the performance of Low-Rank GNNs are relatively stable, which shows the robustness of Low-Rank GNNs and can be attributed to the effect of low-rank approximation as shown in figure 5. Specifically, the proposed Low-Rank GNNs obtains node embedding without propagation and the given topology is only used to form the ego-network, while GCN needs to propagate the information based on topology and GAT focuses on correcting and refining the existing topology and then propagating, which indicates that they are more likely to be effected by corrupted topology information. Therefore, Low-Rank GNNs tends to be more robust to topology noises than the existing GNNs.

We also evaluate the robustness of Low-Rank GNNs with adding noisy attributes. Specifically, according to the given sampling rate, a certain number of attributes are randomly selected and changed from the original features. Figure 6 reports that the performance degradation of Low-Rank GNNs is slight and outperforms GCN and GAT with different levels of noise interference, which can be

attributed to the denoising in Low-rank GNNNs may take the high-order relationship in the ego-nework. Therefore, Low-Rank GNNs may be more robust to attribute noises as discussed in Section 3.4.

5 CONCLUSIONS

This paper has investigated some essential issues in most existing propagation-based graph neural networks, including causing over-smoothing and performance drop in networks with heterophily, irrelevance to model high-order relationship and fragility to topology and attribute noises. The propagation, as the key component of most GNNs, may be the essence of these serious issues. Therefore, the graph topology is only employed to construct the ego-networks instead of message propagation, which is sensitive to topology noises, and the propagation in the ego-network is replaced with a novel *local* operation, i.e., low-rank matrix approximation. Quantitative experimental analysis reveals: 1) the existence of low-rank characteristic in the node attributes from ego-networks and 2) the performance improvement by reducing its rank. The proposed Low-Rank GNNs, which perform low-rank attribute matrix approximation in ego-network, posses some characteristics, including robust to topology and attribute noises, handling networks with both homophily and heterophily, parameter-free and parallelizable with theoretically analysis and experimental evaluations. Low-Rank GNNs can be applied to task assignment learning tasks [45, 46].

REFERENCES

[1] Muhammet Balcilar, Guillaume Renton, Pierre Héroux, Benoit Gaüzère, Sébastien Adam, and Paul Honeine. 2021. Analyzing the Expressive Power of Graph Neural Networks in a Spectral Perspective. In *ICLR*.

[2] Deyu Bo, Xiao Wang, Chuan Shi, and Huawei Shen. 2021. Beyond Low-frequency Information in Graph Convolutional Networks. (2021), 3950–3957.

[3] Stephen P. Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. 2011. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Found. Trends Mach. Learn.* 3, 1 (2011), 1–122. https://doi.org/10.1561/2200000016

[4] Stephen P. Boyd and Lieven Vandenberghe. 2014. *Convex Optimization.* Cambridge University Press. https://doi.org/10.1017/CBO9780511804441

[5] Jian-Feng Cai, Emmanuel J. Candès, and Zuowei Shen. 2010. A Singular Value Thresholding Algorithm for Matrix Completion. *SIAM J. Optim.* 20, 4 (2010), 1956–1982. https://doi.org/10.1137/080738970

[6] Emmanuel J. Candès, Xiaodong Li, Yi Ma, and John Wright. 2011. Robust principal component analysis? *J. ACM* 58, 3 (2011), 11:1–11:37. https://doi.org/10.1145/1970392.1970395

[7] Xiaochun Cao, Liang Yang, and Xiaojie Guo. 2016. Total Variation Regularized RPCA for Irregularly Moving Object Detection Under Dynamic Background. *IEEE Trans. Cybern.* 46, 4 (2016), 1014–1027. https://doi.org/10.1109/TCYB.2015.2419737

[8] Zongsheng Cao, Qianqian Xu, Zhiyong Yang, Xiaochun Cao, and Qingming Huang. 2021. Dual Quaternion Knowledge Graph Embeddings. In *AAAI*. 6894–6902. https://ojs.aaai.org/index.php/AAAI/article/view/16850

[9] Zongsheng Cao, Qianqian Xu, Zhiyong Yang, Xiaochun Cao, and Qingming Huang. 2021. Dual Quaternion Knowledge Graph Embeddings. In *AAAI Conference on Artificial Intelligence*, Vol. 35. 6894–6902.

[10] Jie Chen, Tengfei Ma, and Cao Xiao. 2018. FastGCN: Fast Learning with Graph Convolutional Networks via Importance Sampling. In *ICLR*. https://openreview.net/forum?id=rytstxWAW

[11] Jingjing Chen, Liangming Pan, Zhipeng Wei, Xiang Wang, Chong-Wah Ngo, and Tat-Seng Chua. 2020. Zero-Shot Ingredient Recognition by Multi-Relational Graph Convolutional Network. In *AAAI*. 10542–10550. https://ojs.aaai.org/index.php/AAAI/article/view/6626

[12] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. 2020. Simple and Deep Graph Convolutional Networks. In *ICML*. 1725–1735.

[13] Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. 2021. Adaptive Universal Generalized PageRank Graph Neural Network. In *ICLR*.

[14] Fan RK Chung and Fan Chung Graham. 1997. *Spectral graph theory*. Number 92. American Mathematical Soc.

[15] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *NIPS*. 3837–3845.

[16] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural Message Passing for Quantum Chemistry. In *ICML*. 1263–1272.

[17] Andrew B. Goldberg, Xiaojin Zhu, Ben Recht, Jun-Ming Xu, and Robert D. Nowak. 2010. Transduction with Matrix Completion: Three Birds with One Stone. In *NIPS*. 757–765. https://proceedings.neurips.cc/paper/2010/hash/06409663226af2f3114485aa4e0a23b4-Abstract.html

[18] Shuhang Gu, Lei Zhang, Wangmeng Zuo, and Xiangchu Feng. 2014. Weighted Nuclear Norm Minimization with Application to Image Denoising. In *CVPR*. 2862–2869. https://doi.org/10.1109/CVPR.2014.366

[19] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *NIPS*. 1024–1034.

[20] Qianxiu Hao, Qianqian Xu, Zhiyong Yang, and Qingming Huang. 2021. Learning Meta-path-aware Embeddings for Recommender Systems. In *ACM International Conference on Multimedia*. 3909–3917.

[21] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.

[22] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. 2019. Predict then Propagate: Graph Neural Networks meet Personalized PageRank. In *ICLR*.

[23] Ron Levie, Federico Monti, Xavier Bresson, and Michael M. Bronstein. 2019. CayleyNets: Graph Convolutional Neural Networks With Complex Rational Spectral Filters. *IEEE Trans. Signal Process.* 67, 1 (2019), 97–109. https://doi.org/10.1109/TSP.2018.2879624

[24] Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper Insights Into Graph Convolutional Networks for Semi-Supervised Learning. In *AAAI*. 3538–3545.

[25] Zhaopeng Li, Qianqian Xu, Yangbangyan Jiang, Xiaochun Cao, and Qingming Huang. 2020. Quaternion-Based Knowledge Graph Network for Recommendation. In *MM*. 880–888. https://doi.org/10.1145/3394171.3413992

[26] Guangcan Liu, Zhouchen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma. 2013. Robust Recovery of Subspace Structures by Low-Rank Representation. *IEEE Trans. Pattern Anal. Mach. Intell.* 35, 1 (2013), 171–184. https://doi.org/10.1109/TPAMI.2012.88

[27] Galileo Namata, Ben London, Lise Getoor, and Bert Huang. 2012. Query-driven active surveying for collective classification. In *International Workshop on Mining and Learning with Graphs*.

[28] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. 2020. Geom-GCN: Geometric Graph Convolutional Networks. In *ICLR*.

[29] Benedek Rozemberczki, Carl Allen, and Rik Sarkar. 2019. Multi-scale Attributed Node Embedding. *arXiv:1909.13021* (2019).

[30] Victor Garcia Satorras and Joan Bruna Estrach. 2018. Few-Shot Learning with Graph Neural Networks. In *ICLR*. https://openreview.net/forum?id=BJj6qGbRW

[31] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI magazine* 29, 3 (2008), 93–93.

[32] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. 2019. Pitfalls of Graph Neural Network Evaluation. arXiv:1811.05868 [cs.LG]

[33] Jie Tang, Jimeng Sun, Chi Wang, and Zi Yang. 2009. Social influence analysis in large-scale networks. In *SIGKDD*. ACM, 807–816.

[34] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.

[35] Ruijia Wang, Shuai Mou, Xiao Wang, Wanpeng Xiao, Qi Ju, Chuan Shi, and Xing Xie. 2021. Graph Structure Estimation Neural Networks. In *WWW*. 342–353. https://doi.org/10.1145/3442381.3449952

[36] Felix Wu, Amauri H. Souza Jr., Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Q. Weinberger. 2019. Simplifying Graph Convolutional Networks. In *ICML*. 6861–6871.

[37] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. 2021. A Comprehensive Survey on Graph Neural Networks. *TNNLS* 32, 1 (2021), 4–24. https://doi.org/10.1109/TNNLS.2020.2978386

[38] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks?. In *ICLR*.

[39] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation Learning on Graphs with Jumping Knowledge Networks. In *ICML*. 5449–5458.

[40] Allen Y. Yang, Zihan Zhou, A. G. Balasubramanian, S. Shankar Sastry, and Yi Ma. 2013. Fast l1-Minimization Algorithms for Robust Face Recognition. *IEEE Trans. Image Process.* 22, 8 (2013), 3234–3246. https://doi.org/10.1109/TIP.2013.2262292

[41] Liang Yang, Lina Kang, Qiuliang Zhang, Mengzhe Li, Bingxin Niu, Dongxiao He, Zhen Wang, Chuan Wang, Xiaochun Cao, and Yuanfang Guo. 2022. OPEN: Orthogonal Propagation with Ego-Network Modeling. In *NeurIPS*. https://openreview.net/forum?id=G25uStbmC7

[42] Liang Yang, Zesheng Kang, Xiaochun Cao, Di Jin, Bo Yang, and Yuanfang Guo. 2019. Topology Optimization based Graph Convolutional Network. In *IJCAI*. 4054–4061. https://doi.org/10.24963/ijcai.2019/563

[43] Liang Yang, Mengzhe Li, Liyang Liu, Bingxin Niu, Chuan Wang, Xiaochun Cao, and Yuanfang Guo. 2021. Diverse Message Passing for Attribute with Heterophily. In *NeurIPS*. 4751–4763. https://proceedings.neurips.cc/paper/2021/hash/253614bbac999b38b5b60cae531c4969-Abstract.html

[44] Liang Yang, Chuan Wang, Junhua Gu, Xiaochun Cao, and Bingxin Niu. 2021. Why Do Attributes Propagate in Graph Convolutional Neural Networks?. In *AAAI*. 4590–4598.

[45] Zhiyong Yang, Qianqian Xu, Xiaochun Cao, and Qingming Huang. 2020. Task-Feature Collaborative Learning with Application to Personalized Attribute Prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43, 11 (2020), 4094–4110.

[46] Zhiyong Yang, Qianqian Xu, Yangbangyan Jiang, Xiaochun Cao, and Qingming Huang. 2019. Generalized Block-Diagonal Structure Pursuit: Learning Soft Latent Task Assignment against Negative Transfer. *Advances in Neural Information Processing Systems* 32 (2019).

[47] Muhan Zhang and Yixin Chen. 2018. Link Prediction Based on Graph Neural Networks. In *NeurIPS*. 5171–5181.

[48] Huan Zhao, Quanming Yao, James T. Kwok, and Dik Lun Lee. 2017. Collaborative Filtering with Social Local Models. In *ICDM*. 645–654. https://doi.org/10.1109/ICDM.2017.74

[49] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI Open* 1 (2020), 57–81. https://doi.org/10.1016/j.aiopen.2021.01.001

[50] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. 2020. Beyond Homophily in Graph Neural Networks: Current Limitations and Effective Designs. In *NeurIPS*.