Community-Hop: Enhancing Node Classification through Community Preference

Ahmed Begga[1], Waqar Ali[2], Gabriel Niculescu[1], Francisco Escolano[1], Thilo Stadelmann[3,4], and Marcello Pelillo[2,3]

[1] University of Alicante, Alicante, Spain
[2] Ca' Foscari University of Venice, Venice, Italy
[3] ECLT European Centre for Living Technology, Venice, Italy
[4] ZHAW Zurich University of Applied Sciences, Winterthur, Switzerland

Abstract. In recent years, Graph Neural Networks (GNNs) have demonstrated significant influence on the analysis of graph structures by leveraging message-passing mechanisms to aggregate neighborhood information and perform various graph-related tasks from node classification to link prediction. Recently, GNNs have mostly been developed to deal with different types of graph structures, such as homophily (similar labels among connected nodes) and heterophily (dissimilar labels among connected nodes). However, existing methods lack the ability to combine node features and graph topology optimally to deal with heterophily. This paper proposes a Community-HOP-based GNN model for dealing with homophilic and heterophilic graph structures. Specifically, we incorporate valuable insights from the graph community structure to guide the feature aggregation process of the GNN layer to learn diverse graph properties and improve performance on node-level tasks. Extensive experiments on six node-level datasets under standard metrics demonstrate that the Community-HOP method surpasses existing baselines.

Keywords: Graph Neural Networks · Node Classification · Spectral Clustering.

1 Introduction

GNNs have proven to be powerful methods for analyzing graph-based data, finding use in diverse areas such as social network analysis and predicting molecular properties [9,18,8]. However, conventional GNN architectures often face challenges in capturing complex structural information and relationships between nodes at various distances, particularly in graphs with heterophilic properties (where connected nodes have dissimilar labels) [21]. Recent advancements in GNN design have addressed these limitations by incorporating higher-order neighborhood information. Models such as MixHop [1] and FSGNN [12] have demonstrated the effectiveness of aggregating features from nodes at different hop distances. These approaches allow for more flexible feature mixing and improved performance on various graph datasets.

While these methods have shown promise, they often treat all neighbors equally within each hop distance, potentially overlooking important structural information encoded in the graph's community structure. Community detection in graphs has long been a subject of study in network science [15,11], with spectral methods providing powerful tools for identifying clusters of densely connected nodes [3].

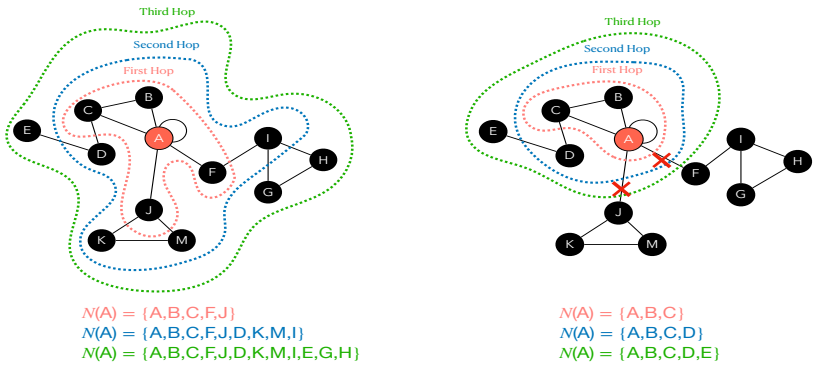

Fig. 1: Traditional Hop vs Community Hop. Evolution of the neighborhood of A, $N(\mathrm{A})$, in traditional hops (left) and in our approach, Community Hop (right).

This study develops a novel GNN layer that leverages graph community structure to guide the feature aggregation process. By combining spectral community detection techniques [11] with a modified transition matrix for inter-community hops, our approach aims to prioritize information flow within and between communities in a more meaningful way. This community-aware feature aggregation strategy allows the model to capture both local (by just combining community nodes) and global (by using the same GNN for all the clusters) graph structures more effectively.

2 Related Work

Recent years have seen significant advancements in GNN architectures, particularly in addressing the challenges of heterophily and over-smoothing. These innovations have largely focused on modifying the feature aggregation process and leveraging higher-order neighborhood information. Several approaches have been proposed to tackle the heterophily problem, where connected nodes may have dissimilar features or labels. H2GCN [21] introduced ego- and neighbor-embedding separation, along with the exploration of higher-order neighborhood structures. GPR-GNN [5] minimizes over-smoothing by integrating the PageRank method with GNNs. Furthermore, GGCN [20] addresses heterophily and over-smoothing issues by utilizing degree corrections and signed messages.

Interestingly, studies have revealed that basic models like Multi-Layer Perceptrons (MLPs) and LINK [10] can occasionally surpass conventional GNN architectures when dealing with heterophilic datasets. This observation has led to the development of hybrid methods that merge node features with graph-based representations. A prominent example is LINKX [10], which integrates MLPs for node features with LINK regression, showing promising performance on heterophilic graphs.

Another line of research has focused on aggregating features from neighbors at different distances. MixHop [1] and FSGNN [12] utilize the transition matrix's powers to capture multi-hop neighborhood information. FSGNN uses a regularizer method, such as softmax and L2-Normalization in GNN's layers.

Recent work has also explored novel ways to address the over-smoothing problem in deeper GNN architectures. Ordered GNN [16] proposes an approach that aligns the hierarchy of a rooted-tree with ordered neurons in node embeddings, effectively preserving information from different neighborhood depths.

While these advancements have significantly improved GNN performance on various graph types, there remains room for innovation in leveraging graph structure more effectively, particularly in the context of community detection and inter-community information flow. Our proposed method builds on these insights by incorporating spectral clustering and community hops, offering a novel approach to enhance GNN performance across diverse node-level predictions.

3 Preliminaries

In this section, we define the mathematical notations used in this study. Let $G = (V, E)$ represent an undirected input graph, where V denotes the set of nodes and $E \subseteq V \times V$ represents the set of edges. We use the adjacency matrix $A \in \{0, 1\}^{n \times n}$ to capture the graph's topological structure, where $A_{ij} = 1$ if $(i, j) \in E$ and $A_{ij} = 0$ otherwise.

To account for self-loops, we modify the adjacency matrix to $\tilde{A} = A + I$, where I denotes the identity matrix. The features of each node are now represented by a matrix $F \in \mathbb{R}^{n \times k}$, where k indicates the dimension of the feature space.

Additionally, we utilize the diagonal degree matrix D for the graph G, where $D_{ii} = d_i$ denotes the degree of node i, calculated by $d_i = \sum_j A_{ij}$. The normalized transition matrix P is then defined as $P = D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$.

3.1 Spectral Clustering

Spectral clustering is a robust method that utilizes the spectral properties of graph Laplacians to achieve clustering [11]. This section covers the essential matrices and concepts that are fundamental to spectral clustering techniques.

A key component in spectral clustering is the graph Laplacian, which comes in two primary forms. The unnormalized graph Laplacian is given by $L = D - W$, where W represents the weighted adjacency matrix and D is the diagonal degree matrix, with $D_{ii} = \sum_j w_{ij}$.

Furthermore, the normalized graph Laplacian can be represented by the following formula: $\mathcal{L} = D^{-1/2}LD^{-1/2} = I - D^{-1/2}WD^{-1/2}$. Here, $\mathcal{L}$ provides a normalized version of the Laplacian that adjusts for the degree of nodes, facilitating more effective clustering.

These Laplacians have several important properties [6]:

1. They are symmetric and positive semi-definite.
2. The smallest eigenvalue is 0, with corresponding eigenvector $\mathbf{1}$ for L and $D^{1/2}\mathbf{1}$ for $\mathcal{L}$.
3. They have n non-negative, real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$.

For any vector $f \in \mathbb{R}^n$, we have:

$$f^T L f = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2; \quad f^T \mathcal{L} f = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}\left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}}\right)^2 \tag{1}$$

The multiplicity k of the eigenvalue 0 equals the number of connected components in the graph. The eigenspace of 0 is spanned by the indicator vectors of these components for L, and by $D^{1/2}$-scaled indicator vectors for $\mathcal{L}$ [11].

Spectral clustering is closely related to the Normalized Cut (NCut) problem [15]. Given a partition of V into k disjoint subsets $A_1, \ldots, A_k$, the NCut is defined as:

$$\mathrm{NCut}(A_1, \ldots, A_k) = \sum_{i=1}^{k} \frac{\mathrm{cut}(A_i, V \setminus A_i)}{\mathrm{vol}(A_i)} \tag{2}$$

where $\mathrm{cut}(A, B) = \sum_{i \in A, j \in B} w_{ij}$ and $\mathrm{vol}(A) = \sum_{i \in A} d_i$.

Minimizing NCut is NP-hard, but it can be relaxed to a tractable eigenvalue problem. This relaxation leads to the spectral clustering algorithm, which computes the first k eigenvectors $u_1, \ldots, u_k$ corresponding to the k smallest eigenvalues of $\mathcal{L}$ (or generalized eigenvectors of $Lu = \lambda Du$) [11,15].

These eigenvectors form a matrix $U \in \mathbb{R}^{n \times k}$, where each row represents a node's k-dimensional embedding. This embedding enhances cluster properties in the data [11], allowing for easier separation in the new representation.

The final step involves clustering these embeddings, typically using the k-means algorithm, to obtain the approximate solution to the NCut problem. This approach effectively captures important graph properties such as communities and structural characteristics through the spectrum of the Laplacian, providing a powerful tool for graph partitioning [6,11,15].

3.2 Graph Neural Networks

Graph Neural Networks (GNNs) have emerged as a significant technique for handling data that is structured as graphs. These models adapt the concepts of convolutional neural networks to the non-Euclidean nature of graph data. The

core idea behind GNNs is to iteratively update node representations by collecting and processing information from neighboring nodes. A typical GNN layer can be formulated as:

$$H^{(i+1)} = \sigma(AH^{(i)}W^{(i)}), \tag{3}$$

where $H^{(i+1)}$ denotes the updated node features matrix at layer $i+1$ after applying the layer transformation,$H^{(i)}$ represents the node features matrix before the transformation, $W^{(i)}$ is a matrix of learnable parameters and σ is a non-linear activation function. In the literature [9], it is common to use the normalized adjacency matrix, which is denoted as $\bar{A} = D^{-\frac{1}{2}}\hat{A}D^{-\frac{1}{2}}$.

4 Methodology

Our methodology addresses the limitations of existing GNN approaches by combining spectral graph theory with flexible multi-hop neighborhood aggregation. Figure 1 illustrates the difference between traditional hops and our community-based approach, which mitigates oversmoothing by emphasizing communal connections [4].

The foundation of our approach leverages spectral graph clustering to uncover global community structure. We begin with the eigendecomposition of the normalized Laplacian $\mathcal{L}$ [6]:

$$\mathcal{L} = U\Lambda U^T, \tag{4}$$

where U is the matrix of eigenvectors and Λ is the diagonal matrix of eigenvalues. The spectral properties of $\mathcal{L}$ are intimately connected to the graph's structure, with eigenvalues in the interval $[0, 2]$ [6].

We focus on the spectral gap, defined as $\gamma = \lambda_2 - \lambda_1$, where $\lambda_1 = 0$ and λ_2 is the smallest non-zero eigenvalue. This gap is related to the graph's connectivity and mixing time [17]. Specifically, the Cheeger constant $h(G)$, which measures the "bottleneckedness" of the graph, is bounded by the spectral gap through the Cheeger inequality:

$$\frac{\lambda_2}{2} \leq h(G) \leq \sqrt{2\lambda_2}, \tag{5}$$

This relationship, known as the Lovász bound [11,2], provides crucial insights into the graph's community structure. A small Cheeger constant indicates the presence of well-defined communities, while a large constant suggests a more uniformly connected graph [7].

We select the k leading eigenvectors corresponding to the smallest non-zero eigenvalues, where k is a hyperparameter. The choice of k can be guided by examining subsequent spectral gaps $(\lambda_{i+1} - \lambda_i)$, with a large gap suggesting a natural number of clusters [11].

To identify communities, we apply k-means clustering to the rows of the truncated eigenvector matrix U_k. This spectral embedding tends to separate nodes into more linearly distinguishable clusters than in the original graph space.

We then introduce an edge-pruning mechanism to emphasize intra-community connections:

$$\mathcal{A}_{ij} = A_{ij} \cdot [C(i) = C(j)]\,, \tag{6}$$

where $C(i)$ denotes the cluster assignment of node i. This pruning creates a block-diagonal structure in $\mathcal{A}$, aligning with the theoretical expectation of an ideal community structure in the spectral clustering framework. Following MixHop [1] and FSGNN [12], we will use the transition matrix to perform hops but this time with $\mathcal{A}$. Now the transition matrix can be defined as $\mathcal{P} = D^{-\frac{1}{2}}\mathcal{A}D^{-\frac{1}{2}}$

Building on this community-aware structure, we incorporate a multi-hop aggregation scheme with attention-like learnable parameters, inspired by recent GNN advancements [1,12]. The feature update rule for the $(i+1)$-th layer is:

$$H^{(i+1)} = \left[\alpha_1 A H^{(i)} W_1^{(i)} \| \alpha_2 \mathcal{P}^1 H^{(i)} W_2^{(i)} \| \alpha_3 \mathcal{P}^2 H^{(i)} W_3^{(i)} \| \cdots \| \alpha_{j+1} \mathcal{P}^j H^{(i)} W_{j+1}^{(i)}\right], \tag{7}$$

where j is the number of hops, $H^{(i)} \in \mathbb{R}^{n \times d_i}$ is the node feature matrix at the i-th layer, with n nodes and d_i features. $\mathcal{P} \in \mathbb{R}^{n \times n}$ is our community-aware transition matrix, and $W_{j+1}^{(i)} \in \mathbb{R}^{d_i \times d_{out}}$ are learnable weight matrices for each hop distance j at layer i, and $\|$ denotes column-wise concatenation.

We introduce learnable attention-like parameters α_{j+1} for each hop embedding and the original node features and adjacency, allowing the model to weigh the importance of different neighborhood scales adaptively. Importantly, these attention parameters are constrained to sum to 1: $\sum_{k=1}^{j+1} \alpha_k = 1, \quad \alpha_k \geq 0 \quad \forall k \in \{0, 1, \ldots, m\}$:

This constraint ensures that the attention mechanism is a proper weighting system across different hop distances.

This multi-hop aggregation allows the model to simultaneously capture and weigh information from various neighborhood scales, as we illustrate in Figure 2. For instance, if $j = 2$, the model considers the initial adjacency ($\alpha_1 A H^{(i)} W_1^{(i)}$), its immediate neighbors ($\alpha_2 \mathcal{P}^1 H^{(i)} W_1^{(i)}$), and its 2-hop neighbors ($\alpha_3 \mathcal{P}^2 H^{(i)} W_2^{(i)}$) in each layer The use of different weight matrices $W_{j+1}^{(i)}$ and attention parameters α_{j+1} for each hop distance and the initial adjacency, enables the model to learn the relative importance of information from different scales while maintaining a balanced aggregation.

This approach generalizes the power iteration method often used in spectral clustering, allowing the capture of higher-order relationships in the graph while maintaining the ability to differentiate between local and global structural information. By learning to assign different importance to various neighborhood scales and preserving the original feature information, our model effectively captures complex patterns of node similarity and dissimilarity, adapting to both homophilic and heterophilic graph structures.

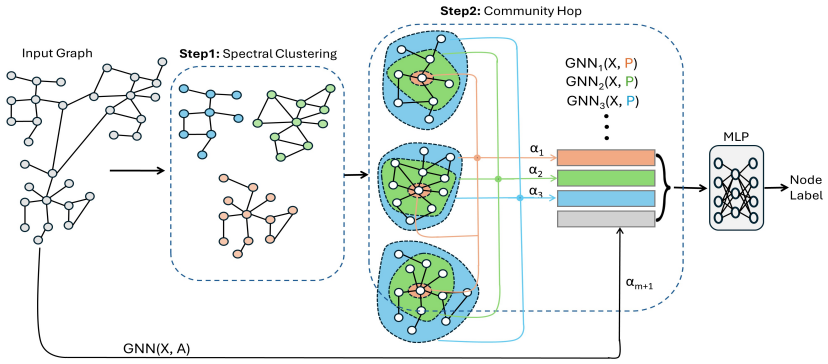

Fig. 2: Illustration of the spectral clustering and GNN propagation process. The input graph undergoes spectral clustering to identify communities (Step 1). Then, a community-aware multi-hop aggregation is performed (Step 2), where information is propagated within communities. The obtained node representations are concatenated and then passed through a MLP for the final prediction of node labels.

4.1 Computational Complexity

Our approach's computational complexity is divided into two primary processes: preprocessing and processing.

The preprocessing phase involves calculating the k leading eigenvectors of the normalized Laplacian matrix. For a graph with n nodes and m edges, this computation has a worst-case time complexity of $O(n^3)$ and a space complexity of $O(n^2)$. However, performance can be enhanced by employing optimized algorithms tailored for sparse graphs.

During the processing phase, k-means clustering and GNN propagation are performed. This step has a time complexity is $O(nk^2 + LHmF)$, where k denotes the number of clusters and the dimension of spectral embedding, L is the number of GNN layers, H represents the number of hops, and F is the number of features. The space complexity for the processing phase is $O(n(k + F) + m)$.

Despite the preprocessing step being computationally intensive, especially for larger graphs, it provides a comprehensive basis for identifying global community structures. This trade-off between computational cost and structural insight allows our approach to effectively capture both global and local patterns in the graph, enabling robust performance on both homophilic and heterophilic graph structures.

5 Experiments and discusions

This section evaluates the proposed method's performance on six node classification benchmarks. The experimental results reveal that the Community-HOP

	Texas	Wisconsin	Cornell	Citeseer	Pubmed	Cora
Hom level	0.11	0.21	0.30	0.74	0.80	0.81
# Nodes	183	251	183	3,327	19,717	2,708
# Edges	295	466	280	4,676	44,324	5,278
# Classes	5	5	5	7	3	6
MLP	80.81 ± 4.75	85.29 ± 6.40	81.89 ± 6.40	74.02 ± 1.90	75.69 ± 2.00	87.16 ± 0.37
GCN [9]	55.14 ± 5.16	51.76 ± 3.06	60.54 ± 5.30	76.50 ± 1.36	88.42 ± 0.50	86.98 ± 1.27
GAT [18]	52.16 ± 6.63	49.41 ± 4.09	61.89 ± 5.05	76.55 ± 1.23	87.30 ± 1.10	86.33 ± 0.48
GraphSAGE [8]	82.43 ± 6.14	81.18 ± 5.56	75.95 ± 5.01	76.04 ± 1.30	88.45 ± 0.50	86.90 ± 1.04
H2GCN [21]	84.86 ± 7.23	87.65 ± 4.89	82.70 ± 5.28	77.11 ± 1.57	89.49 ± 0.38	87.87 ± 1.20
Geom-GCN [13]	66.76 ± 2.72	64.51 ± 3.66	60.54 ± 3.67	78.02 ± 1.15	89.95 ± 0.47	85.35 ± 1.57
LINKX [10]	74.60 ± 8.37	75.49 ± 5.72	77.84 ± 5.81	73.19 ± 0.99	87.86 ± 0.77	84.64 ± 1.13
GGCN [20]	84.86 ± 4.55	86.86 ± 3.29	85.68 ± 6.63	77.14 ± 1.45	89.15 ± 0.37	87.95 ± 1.05
CGNN [19]	71.35 ± 4.05	74.31 ± 7.26	66.22 ± 7.69	76.91 ± 1.81	87.70 ± 0.49	87.10 ± 1.35
MixHop [1]	77.84 ± 7.73	75.88 ± 4.90	73.51 ± 6.34	76.26 ± 1.33	85.31 ± 0.61	87.61 ± 0.85
FSGNN [12]	87.30 ± 5.29	87.84 ± 3.37	85.13 ± 6.07	77.40 ± 1.90	77.40 ± 1.93	87.93 ± 1.00
GPRGNN [5]	78.38 ± 4.36	82.94 ± 4.21	80.27 ± 8.11	77.13 ± 1.67	87.54 ± 0.38	87.95 ± 1.18
Community-HOP	89.46 ± 5.72	89.01 ± 3.84	82.70 ± 3.00	78.30 ± 2.13	89.50 ± 0.47	88.22 ± 1.29

Table 1: Node-classification accuracies. Top three models are highlighted: **First**, **Second**, **Third**.

method achieved the highest performance on four out of six datasets compared to baselines. This section describes the datasets and experimental settings, followed by a comprehensive comparison of the results and a detailed analysis.

To evaluate the efficacy of the Community-HOP, we selected six small to medium real-world node classification benchmark datasets: Cora, Cornell, PubMed, Texas, and Wisconsin [14]. A statistical summary of these datasets, including the edge homophily ratio (HOM LEVEL) [21], offers insight into the dataset's heterophily. A higher HOM LEVEL indicates greater heterophily, posing a challenge for vanilla GNN models, which typically perform worse under these conditions.

For the node classification experiments, we utilized the dataset splits provided by [14]. Each split includes 48% of the data for training, 32% for validation, and 20% for testing. The performance metrics are reported as the average accuracy with standard deviation across 10 different splits. All models were trained for a total of 3000 epochs using the Adam optimizer and cross-entropy loss function. To optimize the Community-HOP method, hyperparameter tuning was carried out, focusing on parameters such as learning rate, dropout rate, number of clusters, number of hops, and hidden dimensions. A grid search strategy was used to examine different combinations of these hyperparameters. Detailed information on the hyperparameter settings for each dataset is available at the following link.

Our experimental findings show that Community-HOP markedly exceeds the performance of existing methods in accuracy across four diverse datasets, showcasing its effectiveness and adaptability in node classification tasks with varying degrees of heterophily. The analysis of edge homophily ratios (HOM LEVEL) emphasizes the difficulties encountered with higher heterophily levels

and highlights the improvements offered by Community-HOP over conventional GNNs and other state-of-the-art techniques.

However, our method does not achieve superior performance on the Cornell and PubMed datasets. In the case of Cornell, this limitation can be attributed to difficulties in accurately computing spectral clusters, exacerbated by significant gaps in the dataset's homophily structure. For PubMed, the high volume of nodes and edges impedes our ability to effectively capture the underlying community structure. Consequently, inadequate clustering results in suboptimal performance for community-specific hops.

Our method focuses on community nodes by executing multiple hops within the community and shows promising results, particularly in homophilic environments where neighbors share the same label. This characteristic is advantageous as it aligns with the assumption that nodes within such environments exhibit high intra-community homophily. Notably, our Community-HOP also demonstrates effective performance in heterophilic contexts, suggesting that it can adeptly manage heterophily within communities. This adaptability contributes to its overall improved classification performance across various datasets, showing its potential for broader applicability in diverse graph-based tasks.

6 Conclusion and Future Work

In this study, we introduced a Community-HOP-based GNN model designed to address the challenge of heterophily in graphs. Central to our approach is the Community-Hop method, which leverages community structural information to refine the feature aggregation process within the GNN layers. This technique enhances the relevance of information flow both within and across communities by integrating spectral community detection with an adapted transition matrix for inter-community hops. However, a significant limitation of our approach is its computational cost, particularly for large-scale graphs and the determination of an optimal parameter k. Future work will focus on addressing these challenges by advancing spectral clustering techniques and developing methods for automatic optimization of k.

Acknowledgement

The authors acknowledge the support from the Spanish Government under project PID2022-142516OB-I00.

References

1. Abu-El-Haija, S., Perozzi, B., Kapoor, A., Alipourfard, N., Lerman, K., Harutyunyan, H., Ver Steeg, G., Galstyan, A.: Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In: international conference on machine learning (2019)

2. Arnaiz-Rodríguez, A., Begga, A., Escolano, F., Oliver, N.: Diffwire: Inductive graph rewiring via the lovász bound. In: Learning on Graphs Conference, LoG 2022, 9-12 December 2022, Virtual Event. Proceedings of Machine Learning Research (2022)
3. Bianchi, F.M., Grattarola, D., Alippi, C.: Mincut pooling in graph neural networks. CoRR **abs/1907.00481** (2019)
4. Cai, C., Wang, Y.: A note on over-smoothing for graph neural networks. CoRR **abs/2006.13318** (2020)
5. Chien, E., Peng, J., Li, P., Milenkovic, O.: Adaptive universal generalized pagerank graph neural network. In: International Conference on Learning Representations (2021)
6. Chung, F.R.: Spectral Graph Theory. American Mathematical Society (1997)
7. Deng, S., Ling, S., Strohmer, T.: Strong consistency, graph laplacians, and the stochastic block model. J. Mach. Learn. Res. **22**, 117:1–117:44 (2021)
8. Hamilton, W.L., Ying, R., Leskovec, J.: Inductive representation learning on large graphs. In: Proceedings of the 31st International Conference on Neural Information Processing Systems (2017)
9. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (ICLR) (2017)
10. Lim, D., Hohne, F.M., Li, X., Huang, S.L., Gupta, V., Bhalerao, O.P., Lim, S.N.: Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. In: Advances in Neural Information Processing Systems (2021)
11. von Luxburg, U.: A tutorial on spectral clustering. CoRR **abs/0711.0189** (2007)
12. Maurya, S.K., Liu, X., Murata, T.: Improving graph neural networks with simple architecture design (2021)
13. Pei, H., Wei, B., Chang, K.C., Lei, Y., Yang, B.: Geom-gcn: Geometric graph convolutional networks. CoRR **abs/2002.05287** (2020)
14. Pei, H., Wei, B., Chang, K.C.C., Lei, Y., Yang, B.: Geom-gcn: Geometric graph convolutional networks (2020)
15. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8), 888–905 (2000)
16. Song, Y., Zhou, C., Wang, X., Lin, Z.: Ordered GNN: Ordering message passing to deal with heterophily and over-smoothing. In: The Eleventh International Conference on Learning Representations (2023)
17. Topping, J., Giovanni, F.D., Chamberlain, B.P., Dong, X., Bronstein, M.M.: Understanding over-squashing and bottlenecks on graphs via curvature. In: International Conference on Learning Representations (2022)
18. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph Attention Networks. International Conference on Learning Representations (2018)
19. Yamamoto, T.: Crystal graph neural networks for data mining in materials science (2019)
20. Yan, Y., Hashemi, M., Swersky, K., Yang, Y., Koutra, D.: Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks. CoRR **abs/2102.06462** (2021)
21. Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L., Koutra, D.: Beyond homophily in graph neural networks: Current limitations and effective designs. In: Proceedings of the 34th International Conference on Neural Information Processing Systems (2020)