Restructuring Graph for Higher Homophily via Adaptive Spectral Clustering

Shouheng Li[1,3], Dongwoo Kim[2], Qing Wang[1]

[1]School of Computing, Australian National University

[2]CSE & GSAI, POSTECH

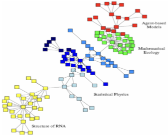
[3]Data61 CSIRO

February 19, 2024

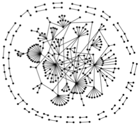
Outline

Graph Neural Networks

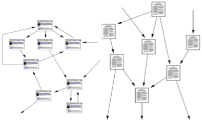
Social networks



Economic networks

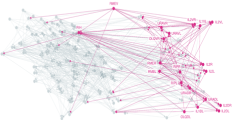


Communication networks



Information networks:
Web & citations

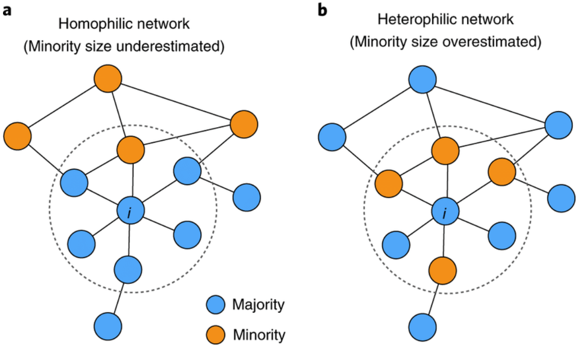
Internet



Networks of neurons

Graph neural networks have been introduced to solve prediction problems related to graph-structured datasets.

Homophilic GNNs for Node Classification



Most well studied spatial and spectral GNNs are homophilic GNNs, where node representations are learned using the information of local neighborhoods (1-hop to k-hop).

What happens if the target graph is heterophilic?

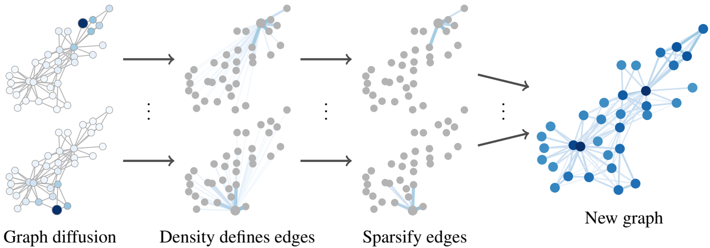
Limitation of Homophilic GNNs

	Heterophily	Homophily
GCN	37.14 ± 4.60	84.52 ± 0.54
GAT	33.11 ± 1.20	84.03 ± 0.97
GCN-Cheby	68.10 ± 1.75	84.92 ± 1.03
GraphSAGE	72.89 ± 2.42	85.06 ± 0.51
MixHop	58.93 ± 2.84	84.43 ± 0.94
MLP	$\mathbf{74.85} \pm 0.76$	71.72 ± 0.62

Well-studied GNNs perform worse than MLP with heterophilic graphs [Zhu et al., 2020].

Rewiring Graphs

Graph diffusion convolution (GDC) [Klicpera et al., 2019] has been proposed as a preprocessing step of GNNs.



GDC rewires the graph based on diffusion process.

Research Objective

Although GDC has been successfully employed to improve the node classification with homophilic graphs, it is unclear whether GDC improve the performance with heterophilic graphs.

In this work, we propose a new graph restructuring method that can be used as a preprocessing step for training well-studied GNNs on heterophilic graphs.

Outline

Spectral Clustering

A Spectral Clustering (SC) algorithm involves the following four steps:

1. Compute eigenvalues and eigenvectors of Laplacian matrix
2. Pick L eigenvectors corresponding with the largest L eigenvalues
3. Represent a node with L components from the selected eigenvectors
4. Perform clustering (such as K-means) with the new node representations

Spectral Clustering

A Spectral Clustering (SC) algorithm involves the following four steps:

1. Compute eigenvalues and eigenvectors of Laplacian matrix
2. Pick L eigenvectors corresponding with the largest L eigenvalues
3. Represent a node with L components from the selected eigenvectors
4. Perform clustering (such as K-means) with the new node representations

Spectral Clustering

A Spectral Clustering (SC) algorithm involves the following four steps:

1. Compute eigenvalues and eigenvectors of Laplacian matrix
2. Pick L eigenvectors corresponding with the largest L eigenvalues
3. Represent a node with L components from the selected eigenvectors
4. Perform clustering (such as K-means) with the new node representations

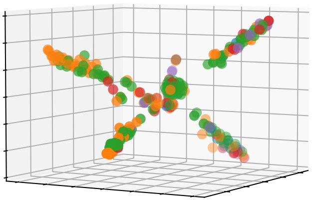
Spectral Clustering

A Spectral Clustering (SC) algorithm involves the following four steps:

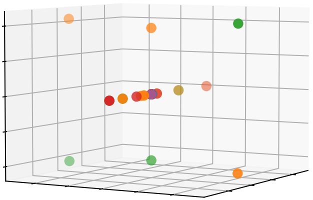
1. Compute eigenvalues and eigenvectors of Laplacian matrix
2. Pick L eigenvectors corresponding with the largest L eigenvalues
3. Represent a node with L components from the selected eigenvectors
4. Perform clustering (such as K-means) with the new node representations

Observation from Heterophilic Graphs (2)

Dataset: Wisconsin



(a) Leading 5 eigenvectors (t-sne)



(b) Selected 3 eigenvectors

Can we make SC adaptive such that the algorithm can automatically select appropriate eigenvectors with given label information?

Revisit Spectral Clustering

Node representation $\boldsymbol{f}_i$ used in spectral clustering can be obtained from low-pass filter on one-hot node signal [Tremblay et al., 2016].

$$\boldsymbol{f}_i = g_{\lambda_L}(\Lambda)\boldsymbol{U}^\top \delta_i,$$

where g_{λ_L} is the low-pass filter:

$$g_{\lambda_L}(\lambda) = \begin{cases} 1 & \text{if } \lambda \leq \lambda_L \\ 0 & \text{otherwise.} \end{cases}$$

that filter out components whose frequencies are greater than λ_L.

Can we use band-pass filter instead low-pass? However, eigen-decomposition is often expensive to perform.

Revisit Spectral Clustering

Node representation $\boldsymbol{f}_i$ used in spectral clustering can be obtained from low-pass filter on one-hot node signal [Tremblay et al., 2016].

$$\boldsymbol{f}_i = g_{\lambda_L}(\Lambda)\boldsymbol{U}^\top \delta_i,$$

where g_{λ_L} is the low-pass filter:

$$g_{\lambda_L}(\lambda) = \begin{cases} 1 & \text{if } \lambda \leq \lambda_L \\ 0 & \text{otherwise.} \end{cases}$$

that filter out components whose frequencies are greater than λ_L.

Can we use band-pass filter instead low-pass? However, eigen-decomposition is often expensive to perform.

Revisit Spectral Clustering

Node representation $\boldsymbol{f}_i$ used in spectral clustering can be obtained from low-pass filter on one-hot node signal [Tremblay et al., 2016].

$$\boldsymbol{f}_i = g_{\lambda_L}(\Lambda)\boldsymbol{U}^\top \delta_i,$$

where g_{λ_L} is the low-pass filter:

$$g_{\lambda_L}(\lambda) = \begin{cases} 1 & \text{if } \lambda \le \lambda_L \\ 0 & \text{otherwise.} \end{cases}$$

that filter out components whose frequencies are greater than λ_L.

Can we use band-pass filter instead low-pass? However, eigen-decomposition is often expensive to perform.

Random Node Features

Let $\boldsymbol{R} = [\boldsymbol{r}_1|\boldsymbol{r}_2|...|\boldsymbol{r}_\eta] \in \mathbb{R}^{N\times\eta}$ be a random node feature matrix where $\boldsymbol{r}_i \sim \mathcal{N}(0, \eta^{-1}\boldsymbol{I}_N)$, and

$$\tilde{\boldsymbol{f}}_i = (\boldsymbol{U} g_{\lambda_L}(\Lambda)\boldsymbol{U}^\top \boldsymbol{R})^\top \delta_i.$$

Lemma ([Tremblay et al., 2016][1])

Let $\epsilon, \beta > 0$ be given. If η is larger than $\frac{4+2\beta}{\epsilon^2/2-\epsilon^3/3}\log N$, then with probability at least $1 - N^{-\beta}$, we have $\forall (i,j) \in [1, N]^2$,

$$(1-\epsilon)\|\boldsymbol{f}_i - \boldsymbol{f}_j\|^2 \leq \|\tilde{\boldsymbol{f}}_i - \tilde{\boldsymbol{f}}_j\|^2 \leq (1+\epsilon)\|\boldsymbol{f}_i - \boldsymbol{f}_j\|^2.$$

by Johnson-Lindenstrauss lemma [Dasgupta and Gupta, 2003].

[1] holds for any band-pass filter.

Random Node Features

Let $\boldsymbol{R} = [\boldsymbol{r}_1|\boldsymbol{r}_2|...|\boldsymbol{r}_\eta] \in \mathbb{R}^{N\times\eta}$ be a random node feature matrix where $\boldsymbol{r}_i \sim \mathcal{N}(0, \eta^{-1}\boldsymbol{I}_N)$, and

$$\tilde{\boldsymbol{f}}_i = (\boldsymbol{U} g_{\lambda_L}(\Lambda)\boldsymbol{U}^\top \boldsymbol{R})^\top \delta_i.$$

Lemma ([Tremblay et al., 2016][1])

Let $\epsilon, \beta > 0$ be given. If η is larger than $\frac{4+2\beta}{\epsilon^2/2-\epsilon^3/3}\log N$, then with probability at least $1 - N^{-\beta}$, we have $\forall(i,j) \in [1, N]^2$,

$$(1-\epsilon)\|\boldsymbol{f}_i - \boldsymbol{f}_j\|^2 \leq \|\tilde{\boldsymbol{f}}_i - \tilde{\boldsymbol{f}}_j\|^2 \leq (1+\epsilon)\|\boldsymbol{f}_i - \boldsymbol{f}_j\|^2.$$

by Johnson-Lindenstrauss lemma [Dasgupta and Gupta, 2003].

[1] holds for any band-pass filter.

Spectral Clustering with Any Band-Pass Filters

To apply spectral clustering with any band-pass filters, we approximate the band-pass rectangular filters using

$$\hat{g}_{s,a}(\lambda) = \frac{1}{s^{2m}}\left(\left(\frac{\lambda - a}{2+\hat{\epsilon}}\right)^{2m} + \frac{1}{s^{2m}}\right)^{-1}$$

- s: width of window
- $a \in [0,2]$: center of window

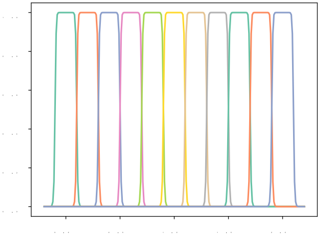


Figure: A set of slicers.

Filtered Node Signals

With band-pass filter $\hat{g}_{s,a}$, representation of node i is

$$\boldsymbol{z}_i^{(s,a)} = (\boldsymbol{U}\hat{g}_{s,a}(\Lambda)\boldsymbol{U}^\top \boldsymbol{R})^\top \delta_i$$

$$= \left(\frac{1}{s^{2m}} \underbrace{\left(\left(\frac{\boldsymbol{L} - a\boldsymbol{I}}{2+\hat{\epsilon}} \right)^{2m} + \frac{\boldsymbol{I}}{s^{2m}} \right)^{-1}}_{\text{expensive}} \boldsymbol{R} \right)^\top \delta_i$$

Lemma (Matrix inversion)

For all $\hat{\epsilon} > \frac{2s^{2m}}{s^{2m}-1} - 2$, the inverse of $\left(\frac{\boldsymbol{L}-a\boldsymbol{I}}{2+\hat{\epsilon}}\right)^{2m} + \frac{\boldsymbol{I}}{s^{2m}}$ can be expressed by a Neumann series with guaranteed convergence.

Filtered Node Signals

With band-pass filter $\hat{g}_{s,a}$, representation of node i is

$$\boldsymbol{z}_i^{(s,a)} = (\boldsymbol{U}\hat{g}_{s,a}(\Lambda)\boldsymbol{U}^\top \boldsymbol{R})^\top \delta_i$$
$$= \left(\frac{1}{s^{2m}} \underbrace{\left(\left(\frac{\boldsymbol{L} - a\boldsymbol{I}}{2+\hat{\epsilon}} \right)^{2m} + \frac{\boldsymbol{I}}{s^{2m}} \right)^{-1}}_{\text{expensive}} \boldsymbol{R} \right)^\top \delta_i$$

Lemma (Matrix inversion)

For all $\hat{\epsilon} > \frac{2s^{2m}}{s^{2m}-1} - 2$, the inverse of $\left(\frac{\boldsymbol{L}-a\boldsymbol{I}}{2+\hat{\epsilon}}\right)^{2m} + \frac{\boldsymbol{I}}{s^{2m}}$ can be expressed by a Neumann series with guaranteed convergence.

Filtered Node Signals

With band-pass filter $\hat{g}_{s,a}$, representation of node i is

$$\mathbf{z}_i^{(s,a)} = (\mathbf{U}\hat{g}_{s,a}(\Lambda)\mathbf{U}^\top \mathbf{R})^\top \delta_i$$

$$= \left(\frac{1}{s^{2m}} \underbrace{\left(\left(\frac{\mathbf{L} - a\mathbf{I}}{2+\hat{\epsilon}} \right)^{2m} + \frac{\mathbf{I}}{s^{2m}} \right)^{-1}}_{\text{expensive}} \mathbf{R} \right)^\top \delta_i$$

Lemma (Matrix inversion)

For all $\hat{\epsilon} > \frac{2s^{2m}}{s^{2m}-1} - 2$, the inverse of $\left(\frac{\mathbf{L}-a\mathbf{I}}{2+\hat{\epsilon}}\right)^{2m} + \frac{\mathbf{I}}{s^{2m}}$ can be expressed by a Neumann series with guaranteed convergence.

Node Representations

Representation of node i with multiple band-pass filters is

$$\boldsymbol{z}_i = \left[\boldsymbol{z}_i^{(s_1,a_1)\top} \boldsymbol{z}_i^{(s_2,a_2)\top} \cdots \boldsymbol{z}_i^{(s_L,a_L)\top}\right]^\top .$$

To incorporate node features into the learning framework, we simply concatenate the node features into the representation

$$\boldsymbol{z}_i = \left[\boldsymbol{z}_i^{(s_1,a_1)\top} \boldsymbol{z}_i^{(s_2,a_2)\top} \cdots \boldsymbol{z}_i^{(s_L,a_L)\top} \boldsymbol{x}_i^\top\right]^\top ,$$

where $\boldsymbol{x}_i$ is the input feature of node i.

Node Representations

Representation of node i with multiple band-pass filters is

$$\boldsymbol{z}_i = \left[\boldsymbol{z}_i^{(s_1,a_1)\top} \boldsymbol{z}_i^{(s_2,a_2)\top} \cdots \boldsymbol{z}_i^{(s_L,a_L)\top}\right]^\top.$$

To incorporate node features into the learning framework, we simply concatenate the node features into the representation

$$\boldsymbol{z}_i = \left[\boldsymbol{z}_i^{(s_1,a_1)\top} \boldsymbol{z}_i^{(s_2,a_2)\top} \cdots \boldsymbol{z}_i^{(s_L,a_L)\top} \boldsymbol{x}_i^\top\right]^\top,$$

where $\boldsymbol{x}_i$ is the input feature of node i.

Learnable Spectral Clustering

We use a triple loss to learn distances between nodes:

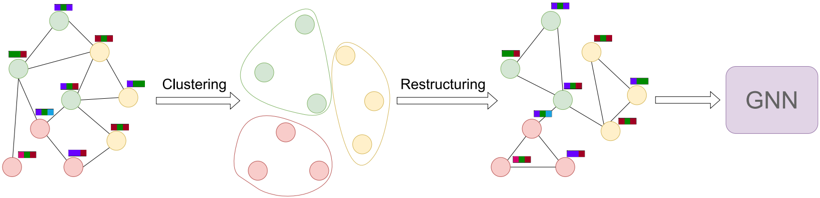
$$\mathcal{L}(\theta) = \sum_{i,p,n} \left[\underbrace{||h_\theta(\boldsymbol{z}_i) - h_\theta(\boldsymbol{z}_p)||^2}_{\text{positive pair}} - \underbrace{||h_\theta(\boldsymbol{z}_i) - h_\theta(\boldsymbol{z}_n)||^2}_{\text{negative pair}} + \epsilon \right]_+ ,$$

where p and n are indexes of positive and negative nodes, ϵ is a margin, and h_θ is a learnable function parameterized by θ.

h_θ adaptively utilizes graph spectrum to minimize the loss.

Restructuring Graphs

We use the learned distance to restructure the graph.



Starting from zero edges, we iteratively add edges from the lowest distances until homophily is maximized.

Outline

Homophily Metrics

There are two widely used homophily metrics:

- Edge homophily (y_u: label of node u)

$$h_{\text{edge}} = \frac{|(u,v) \in E : y_u = y_v|}{|E|}$$

$\Rightarrow$ Fraction of edges connecting the same class.

- Node homophily

$$h_{\text{node}} = \frac{1}{N} \sum_{u \in V} \frac{|v \in \mathcal{N}_u : y_u = y_v|}{|\mathcal{N}_u|}$$

$\Rightarrow$ Average fraction of neighbors sharing the same class.

Homophily Metrics

There are two widely used homophily metrics:

- Edge homophily (y_u: label of node u)

$$h_{\text{edge}} = \frac{|(u, v) \in E : y_u = y_v|}{|E|}$$

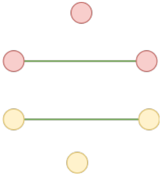
$\Rightarrow$ Fraction of edges connecting the same class.

- Node homophily

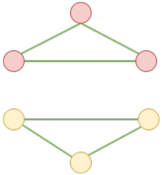
$$h_{\text{node}} = \frac{1}{N} \sum_{u \in V} \frac{|v \in \mathcal{N}_u : y_u = y_v|}{|\mathcal{N}_u|}$$

$\Rightarrow$ Average fraction of neighbors sharing the same class.

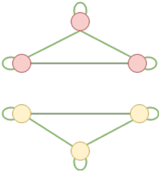
Limitation of Node and Edge Homophily (1)



(a) $h_{\text{edge}} = 1$
$h_{\text{node}} = 1$



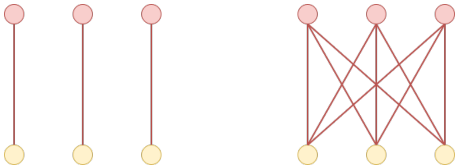
(b) $h_{\text{edge}} = 1$
$h_{\text{node}} = 1$



(c) $h_{\text{edge}} = 1$
$h_{\text{node}} = 1$

Figure: Node and edge homophily metrics do not account density.

Limitation of Node and Edge Homophily (2)



(a) $h_{\text{edge}} = 0$
$h_{\text{node}} = 0$

(b) $h_{\text{edge}} = 0$
$h_{\text{node}} = 0$

Figure: Node and edge homophily metrics do not account density.

Requirements

We design a new metric satisfying following properties:

- A complete set of intra-class with zero inter-class has a score of 1.
- A complete set of inter-class with zero intra-class has a score of 0.
- An Erdos-Renyi random graph $G(n, p)$ has an expected score of 0.5.
- A disconnected graph and a complete graph have a score of 0.5.
- For graphs with the same intra- and inter-class edge ratios, the denser graph has a relatively higher score (before scaling).

Requirements

We design a new metric satisfying following properties:

- A complete set of intra-class with zero inter-class has a score of 1.
- A complete set of inter-class with zero intra-class has a score of 0.
- An Erdos-Renyi random graph $G(n, p)$ has an expected score of 0.5.
- A disconnected graph and a complete graph have a score of 0.5.
- For graphs with the same intra- and inter-class edge ratios, the denser graph has a relatively higher score (before scaling).

Requirements

We design a new metric satisfying following properties:

- A complete set of intra-class with zero inter-class has a score of 1.
- A complete set of inter-class with zero intra-class has a score of 0.
- An Erdos-Renyi random graph $G(n, p)$ has an expected score of 0.5.
- A disconnected graph and a complete graph have a score of 0.5.
- For graphs with the same intra- and inter-class edge ratios, the denser graph has a relatively higher score (before scaling).

Requirements

We design a new metric satisfying following properties:

- A complete set of intra-class with zero inter-class has a score of 1.
- A complete set of inter-class with zero intra-class has a score of 0.
- An Erdos-Renyi random graph $G(n, p)$ has an expected score of 0.5.
- A disconnected graph and a complete graph have a score of 0.5.
- For graphs with the same intra- and inter-class edge ratios, the denser graph has a relatively higher score (before scaling).

Density-Aware Homophily Metric

We propose a new homophily metric to account the density of graph:

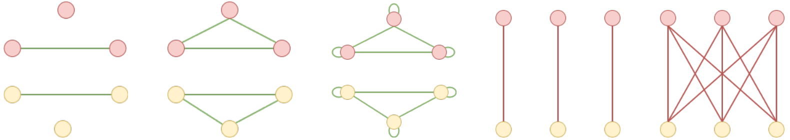
$$h_{\text{den}} = \frac{1 + \min\{d_k - \bar{d}_k\}_{k=0}^{K-1}}{2},$$

where

$$d_k = \frac{2\left|(u,v) \in E : k_u = k_v = k\right|}{|Y_k|(|Y_k| + 1)}, \quad \bar{d}_k = \max\{d_{kj} : j = 0, \ldots, K-1; j \neq k\}.$$

- d_{kj} is the inter-class edge density of label j and k
- $|Y_k|$ is the number of nodes with label k.

Result: New Homophily Metric



(a) $h_{\text{edge}} = 1$
$h_{\text{node}} = 1$
$h_{\text{den}} = 0.54$

(b) $h_{\text{edge}} = 1$
$h_{\text{node}} = 1$
$h_{\text{den}} = 0.63$

(c) $h_{\text{edge}} = 1$
$h_{\text{node}} = 1$
$h_{\text{den}} = 1$

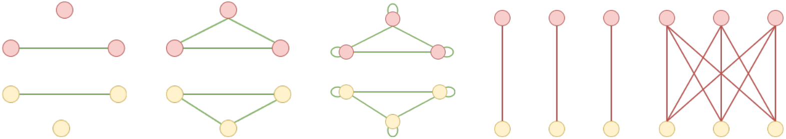
(d) $h_{\text{edge}} = 0$
$h_{\text{node}} = 0$
$h_{\text{den}} = 0.42$

(e) $h_{\text{edge}} = 0$
$h_{\text{node}} = 0$
$h_{\text{den}} = 0$

Lemma

$\forall K > 1$, $\mathbb{E}[h_{den}] = 0.5$ *for the Erdos-Renyi random graph* $G(n, p)$.

Result: New Homophily Metric



(a) $h_{\text{edge}} = 1$ $h_{\text{node}} = 1$ $h_{\text{den}} = 0.54$

(b) $h_{\text{edge}} = 1$ $h_{\text{node}} = 1$ $h_{\text{den}} = 0.63$

(c) $h_{\text{edge}} = 1$ $h_{\text{node}} = 1$ $h_{\text{den}} = 1$

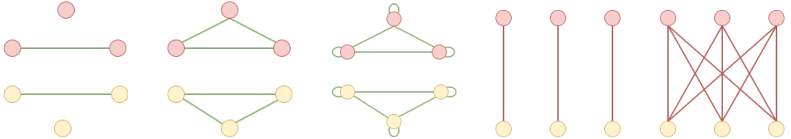
(d) $h_{\text{edge}} = 0$ $h_{\text{node}} = 0$ $h_{\text{den}} = 0.42$

(e) $h_{\text{edge}} = 0$ $h_{\text{node}} = 0$ $h_{\text{den}} = 0$

Lemma

$\forall K > 1$, $\mathbb{E}[h_{den}] = 0.5$ *for the Erdos-Renyi random graph* $G(n, p)$*.*

Result: New Homophily Metric



(a) $h_{\text{edge}} = 1$ $h_{\text{node}} = 1$ $h_{\text{den}} = 0.54$

(b) $h_{\text{edge}} = 1$ $h_{\text{node}} = 1$ $h_{\text{den}} = 0.63$

(c) $h_{\text{edge}} = 1$ $h_{\text{node}} = 1$ $h_{\text{den}} = 1$

(d) $h_{\text{edge}} = 0$ $h_{\text{node}} = 0$ $h_{\text{den}} = 0.42$

(e) $h_{\text{edge}} = 0$ $h_{\text{node}} = 0$ $h_{\text{den}} = 0$

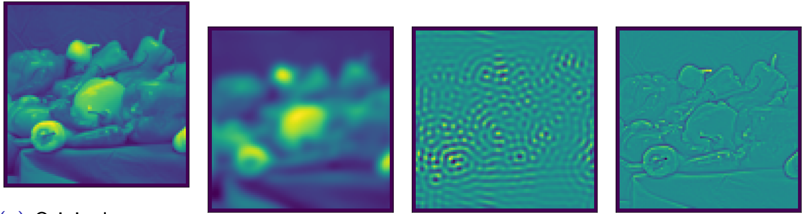
Lemma

$\forall K > 1$, $\mathbb{E}[h_{den}] = 0.5$ *for the Erdos-Renyi random graph* $G(n, p)$.

Outline

1. Motivation
   - Rewiring Graphs
2. Semi-Supervised Spectral Clustering
   - Learnable spectral clustering
   - Graph restructuring
3. Homophily Metric
4. Experiments

Expressive Power in Spectral Perspective



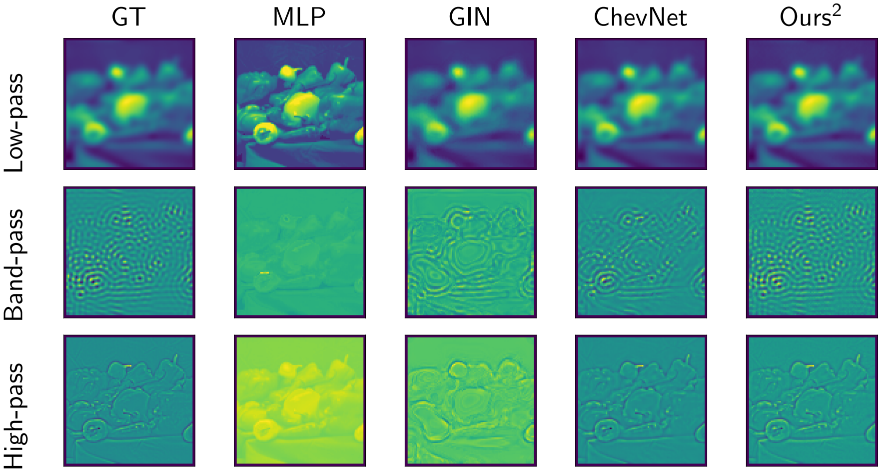
(a) Original [Balcilar et al., 2021] (b) Low-pass (c) Band-pass (d) High-pass

On grid structured graph, we analyze the ability of the learnable spectral clustering to learn specific frequency patterns by training f_θ via

$$\mathcal{L}(\theta) = \sum_{ij} (f_\theta(\boldsymbol{z}_{ij}) - \boldsymbol{y}_{ij})^2,$$

where $\boldsymbol{y}_{ij}$ is the filtered value of pixel ij.

Expressive Power in Spectral Perspective: Training Results



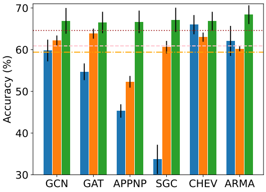
[2]uses 20× less parameters than ChevNet

Node Classification

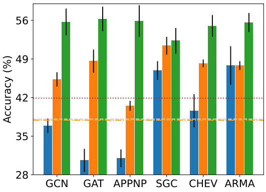
- Datasets - six heterophilic graphs
  - Texas, Cornell, Wisconsin, Actor, Chameleon, Squirrel
- Models
  - GCN, SGC, ChevNet, ARMANet, GAT, APPNP
- Report the performance before and after restructuring on test set
- The graphs are restructured to have edges that give the highest h_{den}.

Node Classification Results (1)

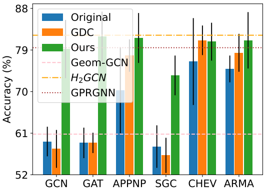
Classification accuracy before and after restructuring graphs.



(a) Chameleon



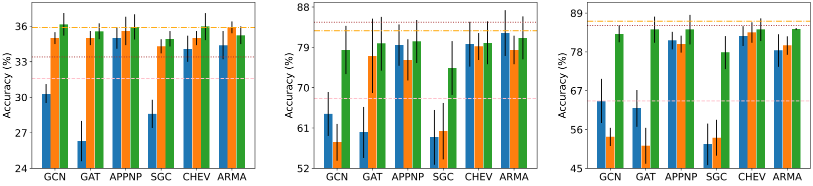
(b) Squirrel



(c) Cornell

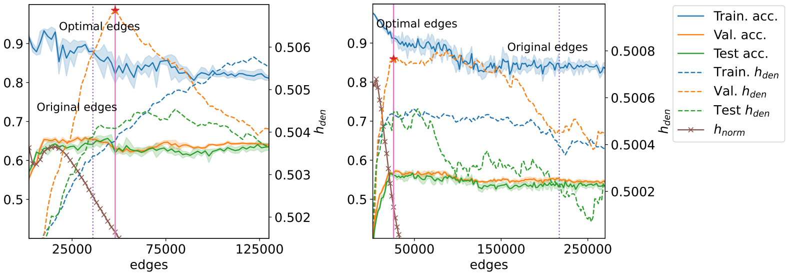
Node Classification Results (2)

Classification accuracy before and after restructuring graphs.



(a) Actor (b) Texas (c) Wisconsin

Result Analysis



(a) CHAMELEON (b) SQUIRREL

Figure: Homophily and accuracy of GCN as per edges numbers. The optimal number of edges are chosen based on h_{den} on validation set.

Conclusion

- We propose a learnable spectral clustering algorithm that can adaptively select proper eigenvectors such that the resulting distance between nodes matches the label distribution of nodes.
- We propose a new metric to measure the homophilic level of a graph.
- Well-developed GNN models enjoy the restructured graphs.

References I

Balcilar, M., Renton, G., Héroux, P., Gaüzère, B., Adam, S., and Honeine, P. (2021).
Analyzing the expressive power of graph neural networks in a spectral perspective.
In *9th International Conference on Learning Representations, ICLR*.

Dasgupta, S. and Gupta, A. (2003).
An elementary proof of a theorem of johnson and lindenstrauss.
Random Struct. Algorithms, 22(1):60–65.

Klicpera, J., Weißenberger, S., and Günnemann, S. (2019).
Diffusion improves graph learning.
In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 13333–13345.

References II

Tremblay, N., Puy, G., Borgnat, P., Gribonval, R., and Vandergheynst, P. (2016).
Accelerated spectral clustering using graph filtering of random signals.
In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L., and Koutra, D. (2020).
Beyond homophily in graph neural networks: Current limitations and effective designs.
In *Advances in Neural Information Processing Systems 33: NeurIPS*.