THE WEB CONFERENCE ACM

UNIVERSITY OF LIVERPOOL

Xi'an Jiaotong-Liverpool University 西交利物浦大學

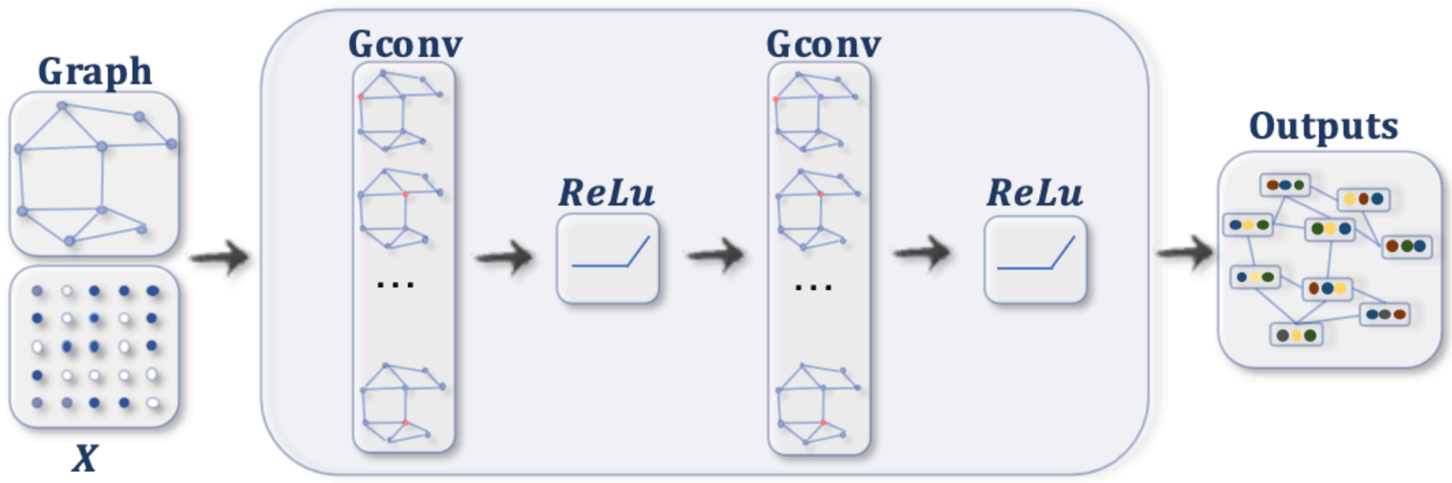
昆山杜克大学 DUKE KUNSHAN UNIVERSITY

Graph Neural Networks with Diverse Spectral Filtering

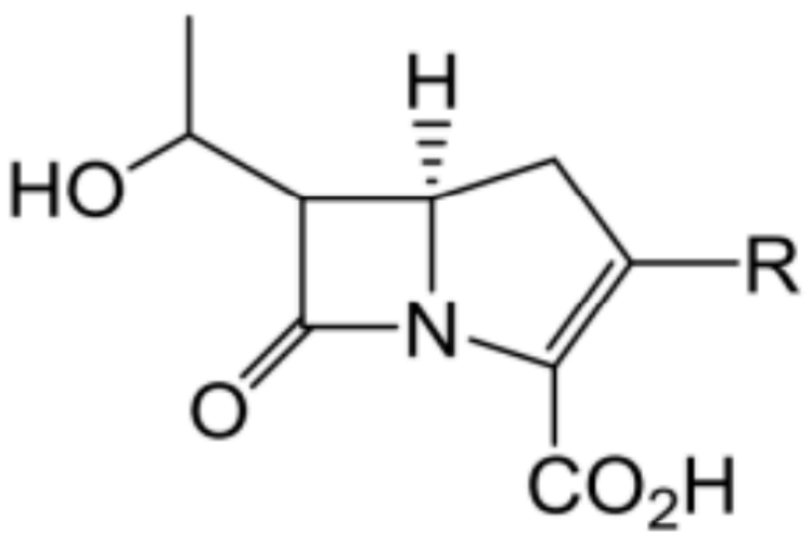



Jingwei Guo, Kaizhu Huang, Xinping Yi, Rui Zhang

Learning from Graph Structure Data

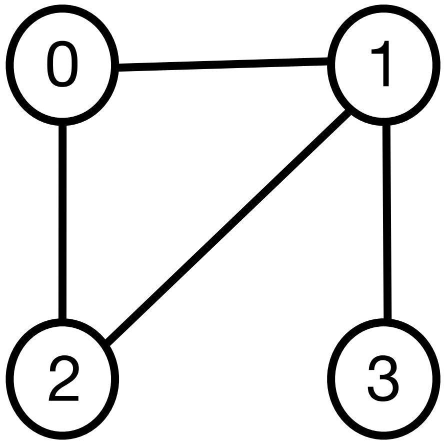


Graph Neural Networks collectively exploiting graph topology and node feature.



https://web.stanford.edu/class/cs224w/slides/01-intro.pdf
Wu et al. A Comprehensive Survey on Graph Neural Networks. In TNNLS, 2020.

Spectral GNNs: Graph Fourier Transform



$$\mathbf{A} = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \quad \mathbf{D} = \begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \mathbf{X} = \begin{pmatrix} 0 \\ 1 \\ 2 \\ 3 \end{pmatrix}$$

Normalized Graph Laplacian — Eigendecomposition

$$\hat{\mathbf{L}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$$

λ_i : The i-th eigenvalue.

→ Frequency

$\mathbf{U}_i$: The i-th eigenvector.

→ Frequency Component

$$\tilde{\mathbf{X}} = \mathbf{U}^T\mathbf{X}$$

Graph Domain ⇄ Frequency Domain

$$\mathbf{X} = \mathbf{U}\tilde{\mathbf{X}}$$

Spectral GNNs: Graph Spectral Filter

Define a filter function $g : [0,2] \rightarrow \mathbb{R}$ in the frequency domain:

Fourier transform of graph signal → Filtering operation with frequency response → Inverse graph Fourier transform

$$\mathbf{S} = \mathbf{U}^T\mathbf{X} \qquad \mathbf{S}^{'}_{[i,:]} = g(\lambda_i)\mathbf{S}_{[i,:]} \qquad \mathbf{Z} = \mathbf{U}\mathbf{S}^{'}$$

$$\mathbf{Z} = g(\hat{\mathbf{L}}) = \mathbf{U}g(\mathbf{\Lambda})\mathbf{U}^T X$$

$$\mathbf{U}\boxed{\mathbf{S}^{'}} \qquad \mathbf{U}\boxed{\mathbf{S}}$$

New Coefficients   Old Coefficients

Spectral GNNs: Polynomial Approximation

Polynomial approximation to the filter function g:

$$g(\lambda) = \sum_{k=0}^{K} \omega_k \lambda^k = \sum_{k=0}^{K} \alpha_k P_k(\lambda)$$

GCN: fixed filter with $g(\lambda) = 2 - \lambda$

BernNet: learnable α_k with Bernstein polynomial basis $P_k(\lambda) = B_{k,K}(\lambda)$

Theoretical expressive power in learning arbitrary filter function.

Kipf & Welling. Semi-supervised Classification with Graph Convolutional Networks. In ICLR 2017.
He et al. BernNet: Learning Arbitrary Graph Spectral Filters via Bernstein Approximation. In NeurIPS 2021.

Motivations: Homogenous Spectral Filtering

- Local modeling nature within K-hop neighborhood

$$\mathbf{Z} = \sum_{k=0}^{K} \alpha_k P_k(\hat{\mathbf{L}})\mathbf{X} \ \underline{\text{while practically}} \ \boxed{K \to \infty \Rightarrow \alpha_k \to 0}$$

- All nodes share the identical transforming coefficient

$$\mathbf{Z} = \sum_{k=0}^{K} \alpha_k P_k(\hat{\mathbf{L}})\mathbf{X} = \boxed{\sum_{n=1}^{N} S_n' \cdot \mathbf{U}_n}$$

where S_n' is a scalar in case of one-channel $\mathbf{X}$ as an example

Existing spectral filtering scheme implicitly assumes the homogenous distributions between different graph parts.

Chien et al. Adaptive Universal Generalized Pagerank Graph Neural Network. In ICLR 2021.
Wang & Zhang. How Powerful are Spectral Graph Neural Networks. In ICML 2022.

Motivations: Heterogeneous Mixing Pattern

Definition 1 (Local Label Homophily). We define the Local Label Homophily as a measure of the local homophily level surrounding each node v_i:

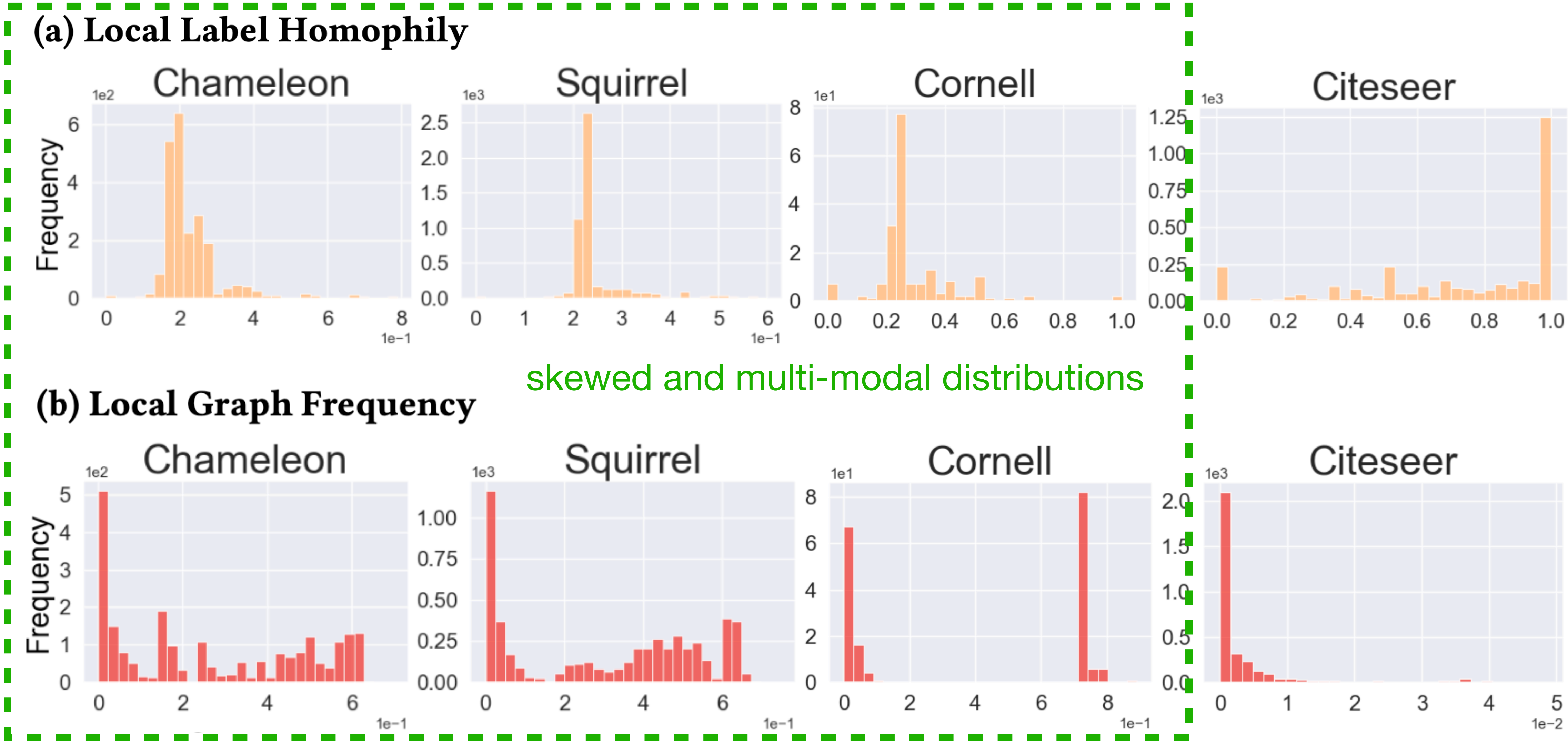
$$h_i = \frac{|\{(v_p, v_q) | \mathbf{y}_p = \mathbf{y}_q \wedge (v_p, v_q) \in \mathcal{E}_{i,k}\}|}{|\mathcal{E}_{i,k}|}$$

Here, h_i directly computes the edge homophily ratio [50] on the subgraph made up of the k-hop neighbors, and $\mathcal{E}_{i,k} = \{(v_p, v_q) | v_p, v_q \in \mathcal{N}_{i,k} \wedge (v_p, v_q) \in \mathcal{E}\}$ denotes its edge set.

Definition 2 (Local Graph Frequency). The Local Graph Frequency is defined by measuring the local smoothness level of the decomposed Laplacian eigenbases, and for each node v_i we have:

$$\lambda_{n,i} = \sum_{(v_p, v_q) \in \mathcal{E}_{i,k}} \left(\frac{1}{\sqrt{\deg_p}} \mathbf{u}_{n,p} - \frac{1}{\sqrt{\deg_q}} \mathbf{u}_{n,q}\right)^2$$

where $\lambda_{n,i}$ denotes the frequency or smoothness level of each Laplacian eigenbasis $\mathbf{u}_n$ upon the subgraph induced by the k-hop neighbors. Since all summed elements in Eq. 1 are positive and $\mathcal{E}_{i,k} \subseteq \mathcal{E}$, we can always have a $\xi_i \in (0, 1)$ such that $\lambda_{n,i} = \xi_i \lambda_n$.



Our Solution: Diverse Spectral Filtering

Homogenous spectral filtering:

$$\mathbf{Z} = \sum_{n=1}^{N} \underbrace{S_n^{'}}_{} \overset{\text{Dot product}}{\cdot} \mathbf{U}_n \qquad \overset{\text{Scalar coefficient}}{S_n^{'}} = \sum_{k=0}^{K} \alpha_k P_k(\lambda_n) \mathbf{U}_n^T \mathbf{X}$$

Diverse spectral filtering:

$$\mathbf{Z} = \sum_{n=1}^{N} \mathbf{S}_n^{'} \overset{\text{Hadamard product}}{\odot} \mathbf{U}_n \qquad \overset{\text{The } i\text{-th element of vector coefficients}}{\mathbf{S}_n^{'}(i)} = \sum_{k=0}^{K} \alpha_k P_k(\lambda_{n,i}) \mathbf{U}_n^T \mathbf{X}$$

Our Solution: Diverse Spectral Filtering

<u>Substitution</u> using $\lambda_{n,i} = \xi_i \lambda_{n,i}$ s.t. $0 < \xi_i < 1$

Proposition 1. Suppose a K-order polynomial function $f : [0,2] \to \mathbb{R}$ with polynomial basis $P_k(\cdot)$ and coefficients $\{\alpha_k\}_{k=0}^{K}$ in real number. For any pair of variables $x, \hat{x} \in [0,2]$ satisfying $x = \xi\hat{x}$ where ξ is a constant real number, we always have a function $g : [0,2] \to \mathbb{R}$ with the same polynomial basis but a different set of coefficients $\{\beta_k\}_{k=0}^{K}$ such that $f(x) = g(\hat{x})$.

It allows $\mathbf{S}'_n(i) = \sum_{k=0}^{K} \boxed{\alpha_k P_k(\lambda_{n,i})} \mathbf{U}_n^T \mathbf{X} = \sum_{k=0}^{K} \boxed{\beta_{k,i} P_k(\lambda_i)} \mathbf{U}_n^T \mathbf{X}$

<u>Diverse spectral filtering:</u>

$$\mathbf{Z} = \sum_{k=0}^{K} \begin{pmatrix} \beta_{k,1} & 0 & \cdots & 0 \\ 0 & \beta_{k,2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \beta_{k,N} \end{pmatrix} P_k(\hat{\mathbf{L}})\mathbf{X}$$

Graph Filter Weights

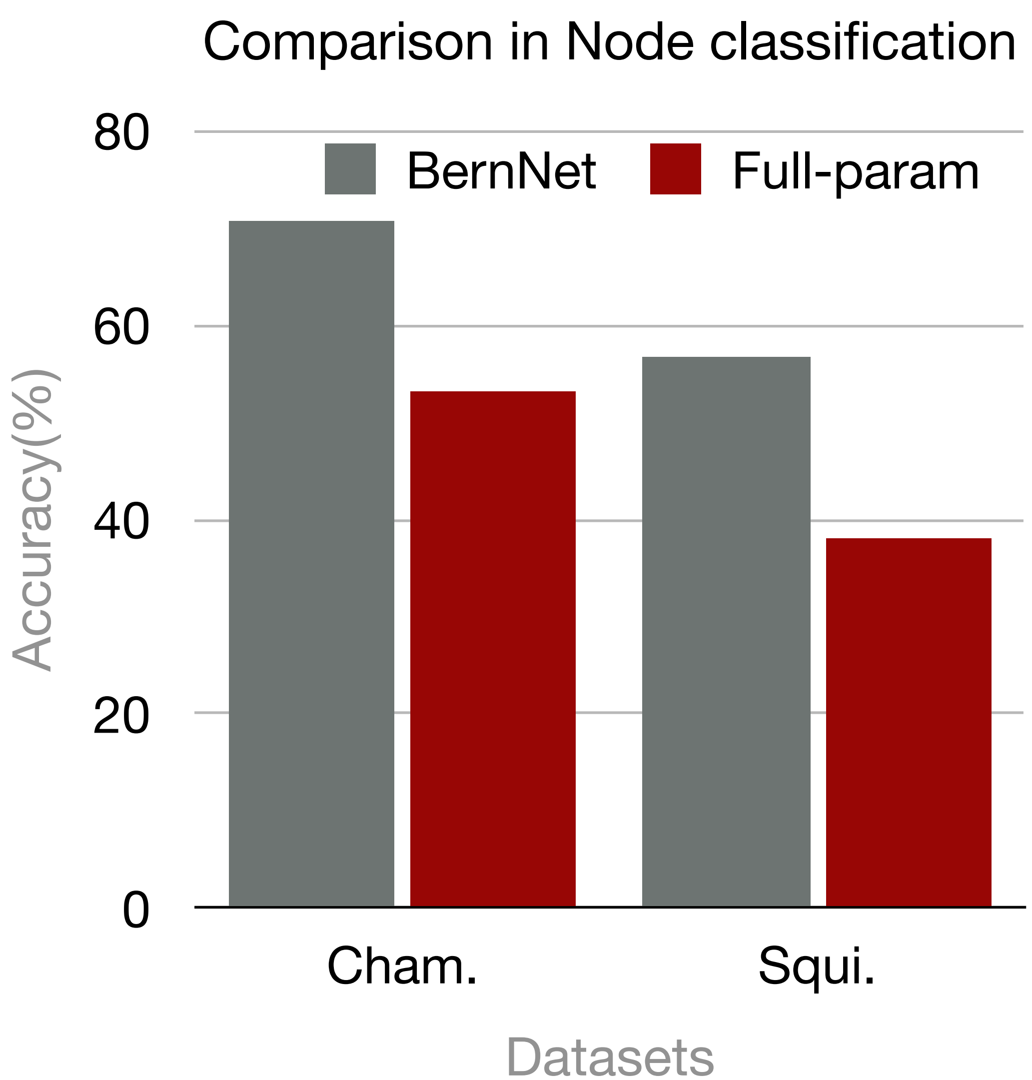
$\alpha_k \to$ diag$(\beta_{k,1}, \beta_{k,2}, \cdots, \beta_{k,N})$

Node-specific Filter Weights

Challenge: Complexity & Noise Overfitting

Issue 1: Large number of Learnable filter weights ($\propto$ # nodes)

Issue 2: Overfitting to local noises in training



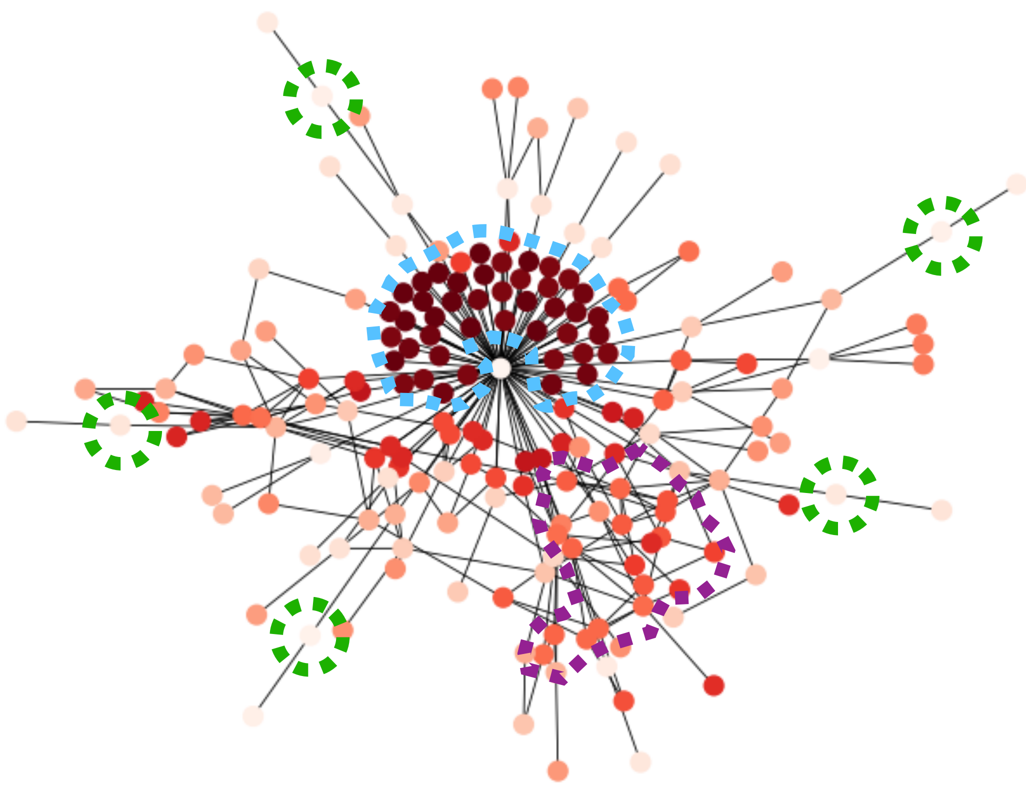
Learning with full-parametrization leads a clear accuracy drop.

"A reason design should be built upon a shared global model whilst locally adapted to each node with awareness of its graph position."

Challenge: Complexity & Noise Overfitting

Observ. 1: Nearby nodes display similar local structures due to their overlapped neighborhoods.

Observ. 2: Distant nodes may still posses alike local context because of the global graph characteristics.



Cornell (webpage network): similar color means akin local structure

- Observ. 1 → ■ ■
- Observ. 2 → ■ in •–•–•

DSF: Positional-aware Filter Weights & LGWD

Position-aware Filter Weights

$$\arg\min_{\mathbf{P}} \mathscr{L}_P = \|\mathbf{P}^{(0)} - \mathbf{P}\|_2^2 + \kappa_1 tr(\mathbf{P}^T \hat{\mathbf{L}} \mathbf{P}) + \kappa_2 \|\mathbf{P}^T \mathbf{P} - \mathbf{I}\|_2^2$$

$\mathbf{P} \in \mathbb{R}^{N \times d}$ is a matrix of node positional embeddings

Iterative gradient method with stepwise b:

$$\mathbf{P}^{(k)} = \mathbf{P}^{(k-1)} - b \cdot \frac{\partial \mathscr{L}_p}{\partial \mathbf{P}} \bigg|_{\mathbf{P}=\mathbf{P}^{(k-1)}}$$

$$= \eta_1 \mathbf{P}^{(0)} + (1 - \eta_1) \left((1 + \eta_2) \hat{\mathbf{A}} - \eta_2 \left(\mathbf{P}^{(k-1)} \mathbf{P}^{(k-1)^T} \right) \right) \mathbf{P}^{(k-1)}$$

where $b = 1/(2 + 2\kappa_1 - 4\kappa_2)$, $\eta_2 = 2\kappa_2/(\kappa_1 - 2\kappa_2)$, $\eta_1 = 1/(1 + \kappa_1 - 2\kappa_2)$.

DSF: Positional-aware Filter Weights & LGWD

Position-aware Filter Weights

$$\mathbf{P}^{(k)} = \eta_1 \mathbf{P}^{(0)} + (1-\eta_1)\left(\left(1+\eta_2\right)\hat{\mathbf{A}} - \eta_2 \sigma\left(\mathbf{P}^{(k-1)}\mathbf{W}\mathbf{P}^{(k-1)^T}\right)\right)\mathbf{P}^{(k-1)}$$

$$\mathbf{P}^{(k)} \leftarrow \tanh(\mathbf{P}^{(k)})$$

$\beta_{k,i} = \sigma_p(\mathbf{W}^{(k)}\mathbf{P}_i^{(k)} + \mathbf{b}^{(k)})$ for each node v_i $\qquad k = 1,2,\cdots,K$

Local and Global Weight Decomposition (LGWD)

$$\beta_{k,i} \leftarrow \gamma_i \cdot \theta_{k,i}$$

invariant graph properties

diverse node contexts

DSF: Overall Algorithm

Original Design → DSF-x-I:

$$\arg\min_{\mathbf{P}} \mathscr{L}_P = \|\mathbf{P}^{(0)} - \mathbf{P}\|_2^2 + \kappa_1 tr(\mathbf{P}^T \hat{\mathbf{L}} \mathbf{P}) + \kappa_2 \boxed{\|\mathbf{P}^T\mathbf{P} - \mathbf{I}\|_2^2}$$

Complexity Overhead with $\mathcal{O}(N^2)$

$$\mathbf{P}^{(k)} = \eta_1 \mathbf{P}^{(0)} + (1-\eta_1)\left(\left(1+\eta_2\right)\hat{\mathbf{A}} - \eta_2 \boxed{\sigma\left(\mathbf{P}^{(k-1)}\mathbf{W}\mathbf{P}^{(k-1)^T}\right)}\right)\mathbf{P}^{(k-1)}$$

Orthogonal Regularization → DSF-x-R:

$$\eta_2 = 0 \quad \& \quad \mathscr{L}_{Orth} = \|\hat{\mathbf{P}}^{(K)}\hat{\mathbf{P}}^{(K)} - \mathbf{I}\|_2^2 \qquad \hat{\mathbf{P}}^{(K)} \xleftarrow{\text{normalization}} \mathbf{P}^{(K)}$$

$$\mathscr{L} = \mathscr{L}_{task} + \lambda_{orth}\mathscr{L}_{orth}$$

$x \in \{\text{GPR}^{[1]}, \text{Bern}^{[2]}, \text{Jacobi}^{[3]}\}$

[1] Chien et al. Adaptive Universal Generalized Pagerank Graph Neural Network. In ICLR 2021.
[2] He et al. BernNet: Learning Arbitrary Graph Spectral Filters via Bernstein Approximation. In NeurIPS 2021.
[3] Wang & Zhang. How Powerful are Spectral Graph Neural Networks. In ICML 2022.

DSF: Overall Algorithm

Original Design → DSF-x-I:

Table 1: Average running time per epoch (ms)/average total running time (s). Although DSF-GPR-I is less efficient on large networks, DSF-GPR-R, (our major model) can reduce it by more than 75% on average (though reasonably slower than GPR-GNN).

Datasets	Small-scale	Large-scale	Average
GPR-GNN	1.10/2.24	0.98/5.01	1.08/2.74
DSF-GPR-I	5.96/12.19	40.34/131.77	12.21/33.93
DSF-GPR-R	2.49/6.29	3.02/14.48	2.59/7.78

$$\mathcal{L} = \mathcal{L}_{task} + \lambda_{orth}\mathcal{L}_{orth}$$

$x \in \{\text{GPR}^{[1]}, \text{Bern}^{[2]}, \text{Jacobi}^{[3]}\}$

[1] Chien et al. Adaptive Universal Generalized Pagerank Graph Neural Network. In ICLR 2021.
[2] He et al. BernNet: Learning Arbitrary Graph Spectral Filters via Bernstein Approximation. In NeurIPS 2021.
[3] Wang & Zhang. How Powerful are Spectral Graph Neural Networks. In ICML 2022.

DSF effectively improves SOTAs spectral GNNs.

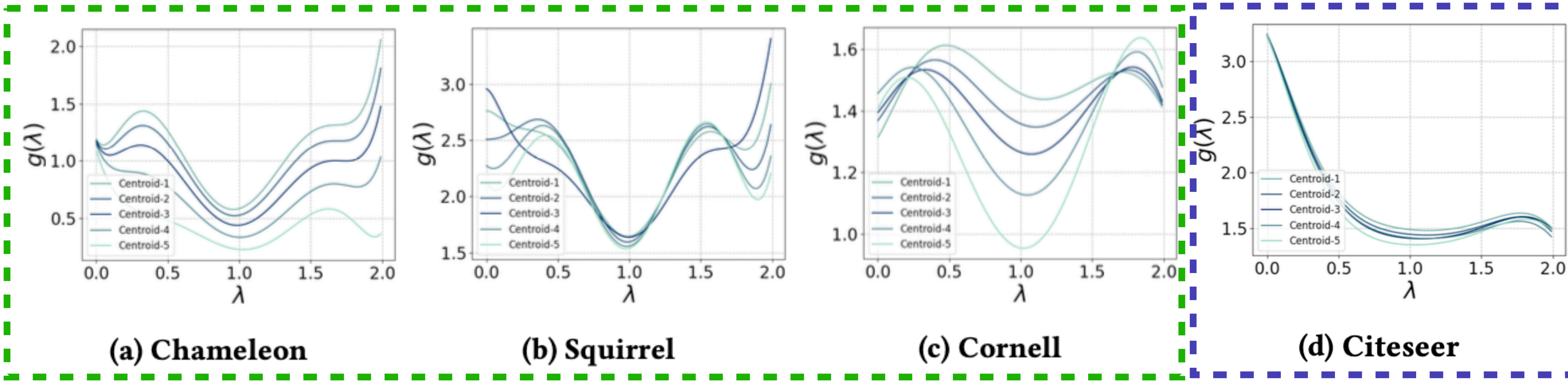
Table 2: Node classification accuracies (%) ± 95% confidence interval over 100 runs. The row of PA-GNN [49]* lists the relative improvements of PA-GNN upon GPR-GNN based on the results obtained from its paper, where – denotes values not provided. Our Improv. gives the best relative improvements between our DSF variants over their common underlying model.

Datasets	Heterophilic Graphs						Homophilic Graphs				
	Chameleon	Squirrel	Wisconsin	Cornell	Texas	Twitch-DE	Cora	Citeseer	Pubmed	Computers	Photo
GCN [24]	67.22±0.43	54.21±0.41	59.45±0.72	52.76±1.17	61.66±0.71	73.94±0.15	88.13±0.25	77.00±0.27	89.07±0.11	91.06±0.12	93.99±0.12
GAT [40]	67.72±0.41	52.26±0.58	57.94±0.89	50.20±0.93	55.37±1.10	73.00±0.15	88.47±0.22	77.23±0.27	88.30±0.11	91.69±0.11	94.55±0.11
ChebNet [11]	64.85±0.44	48.14±0.33	80.93±0.72	77.98±1.00	75.83±1.20	73.73±0.14	87.64±0.21	76.93±0.24	89.91±0.11	91.65±0.12	95.27±0.07
APPNP [15]	53.66±0.33	36.08±0.36	81.23±0.64	81.29±0.78	79.42±1.05	72.65±0.11	88.70±0.21	77.77±0.24	89.93±0.09	91.62±0.10	94.92±0.09
GNN-LF [51]	54.29±0.36	36.87±0.33	59.85±0.60	62.90±0.98	61.88±0.95	73.03±0.13	88.90±0.25	77.35±0.29	88.89±0.10	91.12±0.11	95.13±0.08
GNN-HF [51]	55.22±0.42	35.45±0.30	68.17±0.72	72.98±1.02	66.66±1.34	71.92±0.13	89.01±0.19	77.74±0.23	89.53±0.10	90.73±0.10	95.26±0.09
FAGCN [6]	68.38±0.51	50.08±0.60	82.11±0.85	79.00±0.93	81.00±0.95	74.15±0.13	88.82±0.20	77.65±0.29	90.13±0.11	91.90±0.11	95.25±0.10
GPR-GNN [9]	69.01±0.50	55.39±0.33	82.72±0.85	80.81±0.78	81.66±1.02	74.07±0.18	89.03±0.20	77.63±0.28	90.10±0.44	92.34±0.13	95.34±0.09
DSF-GPR-I	71.18±0.52	57.08±0.29	**87.64**±0.79	84.76±0.90	85.44±1.05	74.58±0.16	**89.64**±0.20	78.03±0.26	90.26±0.08	92.49±0.12	95.64±0.07
DSF-GPR-R	**71.64**±0.55	**58.44**±0.30	87.43±0.74	**84.93**±0.90	**85.56**±0.93	**74.81**±0.14	89.63±0.17	**78.22**±0.29	**90.51**±0.07	**92.80**±0.12	**95.73**±0.08
Our Improv.	2.63%	3.05%	4.92%	4.12%	3.9%	0.74%	0.61%	0.59%	0.41%	0.46%	0.39%
PA-GNN [49]*	0.66%	1.28%	–	–	–	–	-0.09%	-0.74%	-0.03%	1.03%	0.02%
BernNet [20]	70.59±0.42	56.63±0.32	85.00±0.94	82.10±0.95	82.20±0.98	74.45±0.15	88.72±0.23	77.52±0.29	90.21±0.46	92.57±0.10	95.42±0.08
DSF-Bern-I	72.95±0.53	59.45±0.32	**88.23**±0.81	**85.07**±0.93	**84.59**±1.07	74.96±0.15	89.05±0.22	**78.32**±0.27	90.40±0.10	92.76±0.10	95.73±0.07
DSF-Bern-R	**73.60**±0.53	**59.99**±0.30	88.02±0.91	84.29±0.93	84.42±1.00	**75.00**±0.15	**89.10**±0.22	78.27±0.26	**90.52**±0.10	**92.84**±0.10	**95.79**±0.06
Our Improv.	3.01%	3.36%	3.23%	2.97%	2.39%	0.55%	0.38%	0.80%	0.31%	0.27%	0.37%
JacobiConv [42]	73.71±0.42	57.22±0.24	83.21±0.68	82.34±0.88	82.42±0.90	74.34±0.12	89.24±0.19	77.81±0.29	89.50±0.47	92.26±0.10	95.62±0.06
DSF-Jacobi-I	74.88±0.39	58.26±0.26	85.34±0.74	**84.54**±0.81	83.68±1.12	74.65±0.13	89.54±0.19	78.18±0.26	89.78±0.09	92.38±0.11	**95.76**±0.07
DSF-Jacobi-R	**75.00**±0.38	**59.23**±0.27	**86.13**±0.70	84.39±0.88	**84.46**±0.81	**74.75**±0.15	**89.66**±0.19	**78.23**±0.25	**90.07**±0.10	**92.44**±0.11	95.75±0.08
Our Improv.	1.29%	2.01%	2.92%	2.20%	2.04%	0.41%	0.42%	0.42%	0.41%	0.18%	0.14%

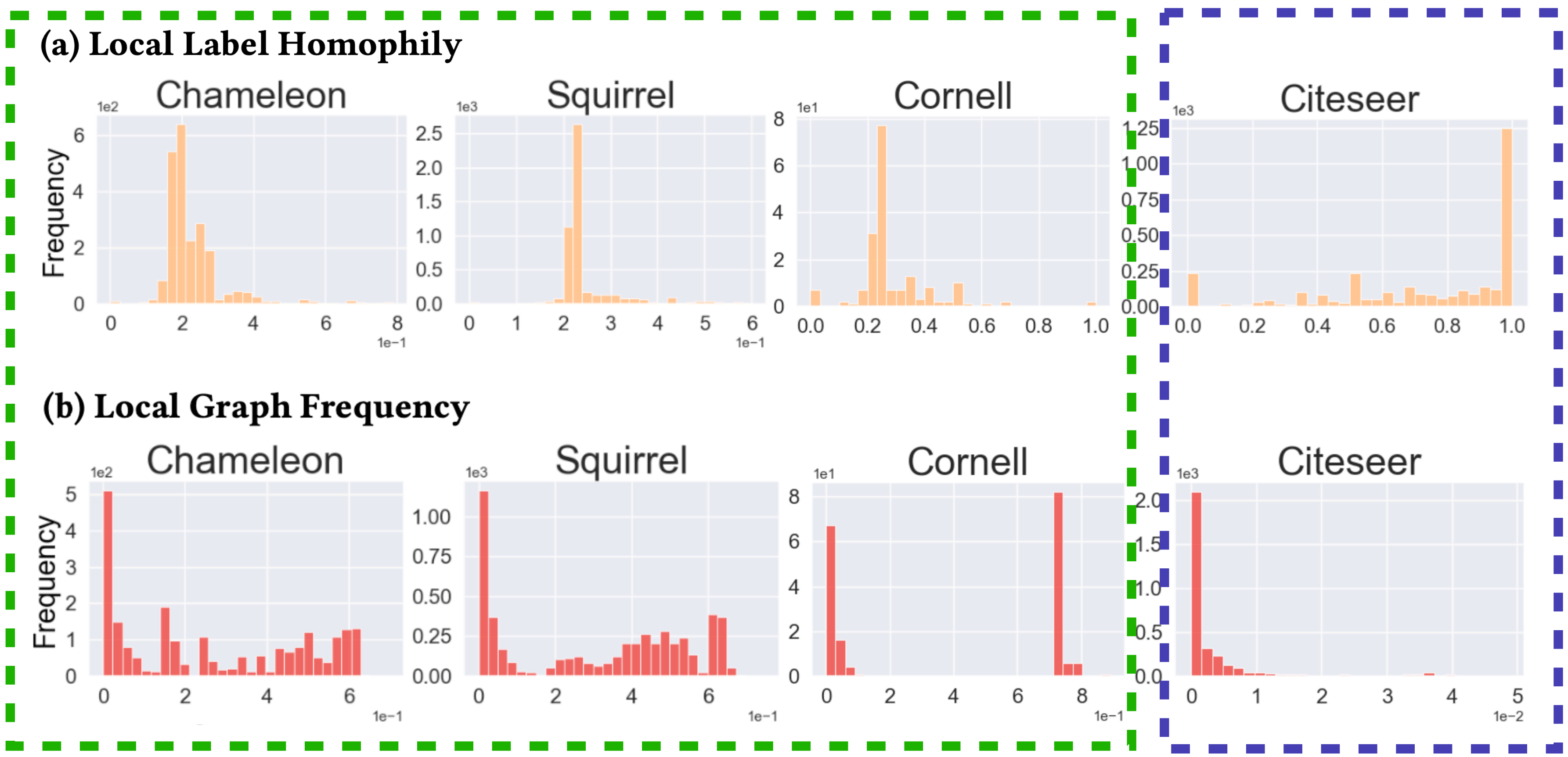
Het. graphs: tend to showcase heterogeneous graph pattern

Hom. graphs: mostly exhibit homogeneous graph pattern

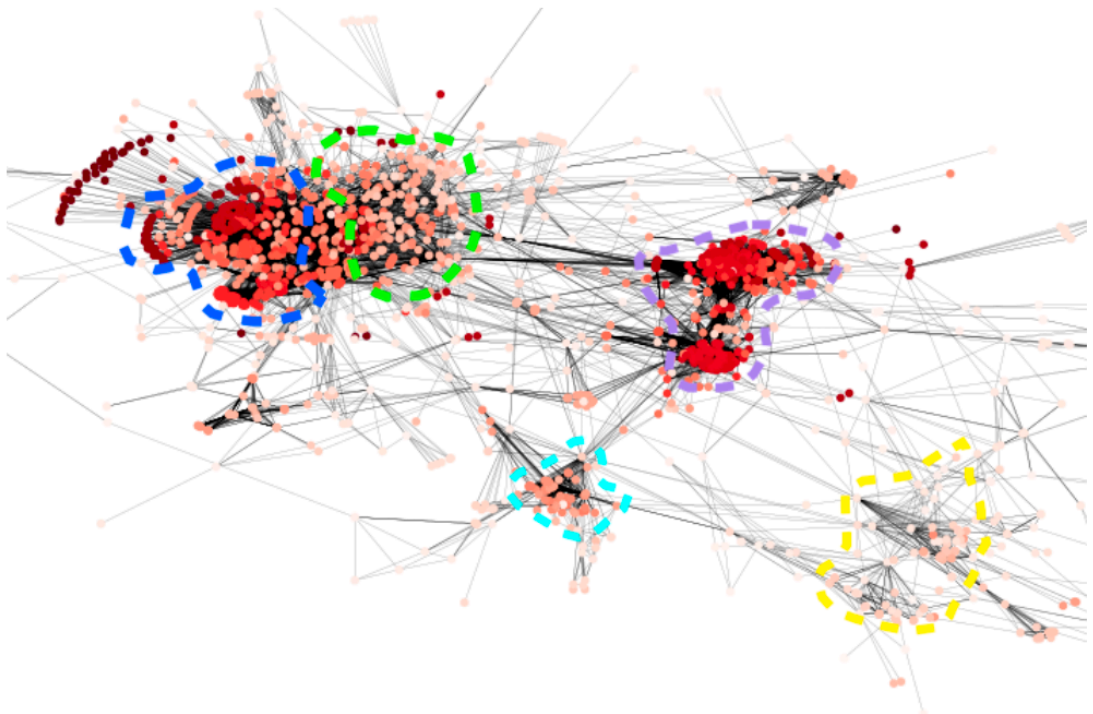
DSF learns interpretable diverse filters



Recall:

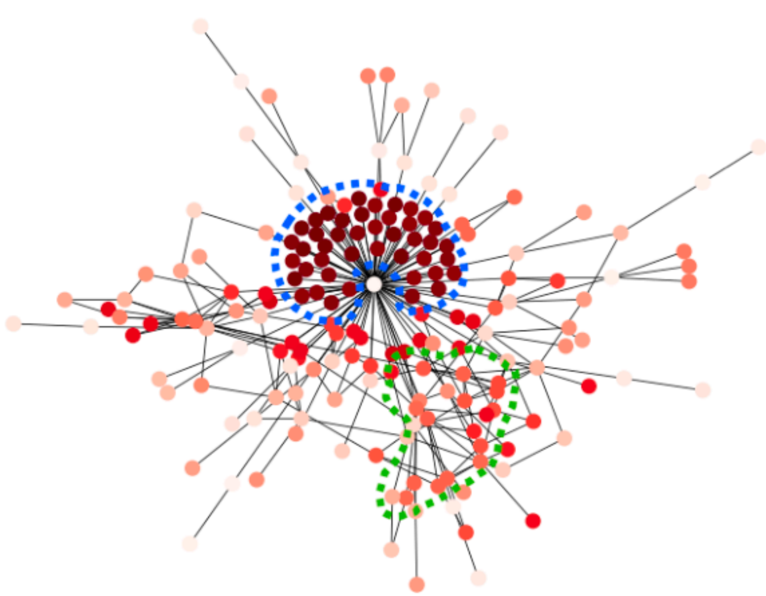


DSF learns interpretable diverse filters

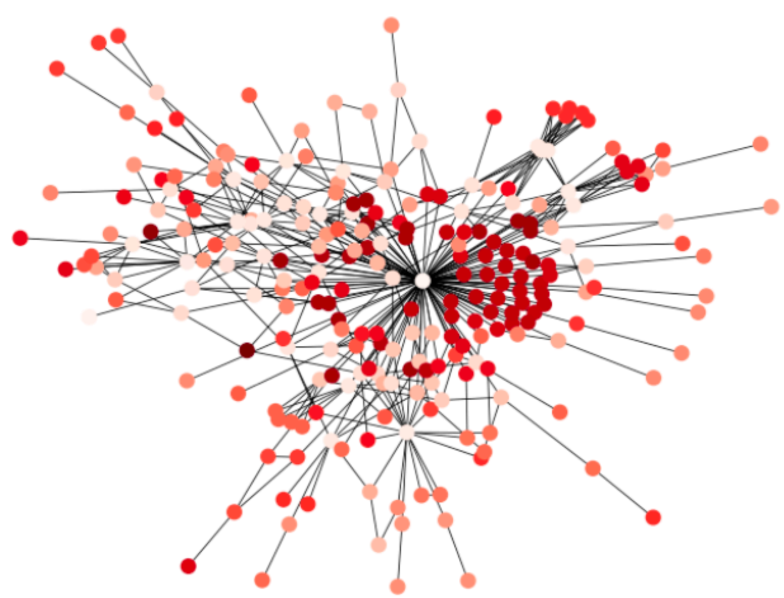


Chameleon

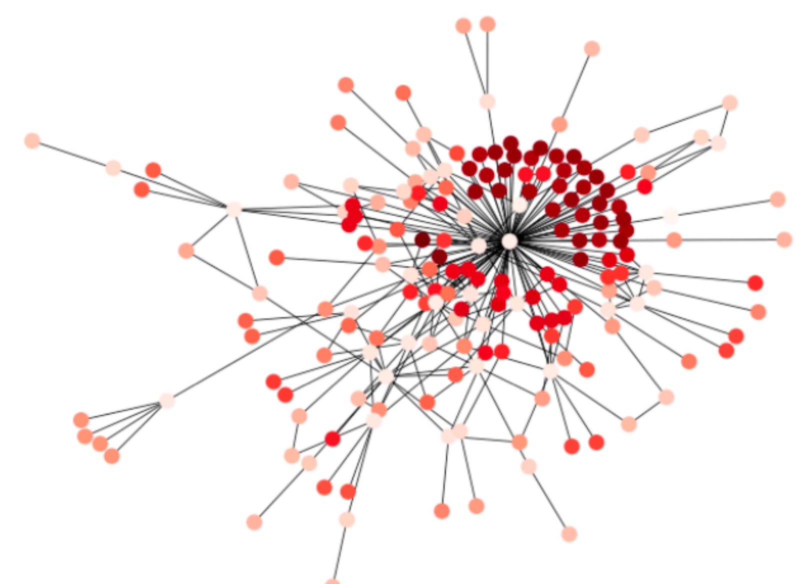
Squirrel



Cornell



Wisconsin



Texas

Thank you!

Contact: Jingwei.Guo@Liverpool.ac.uk

Jingwei Guo

Kaizhu Huang

Xinping Yi

Rui Zhang