Beyond Low-Pass Filters: Adaptive Feature Propagation on Graphs

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

[1]School of Computing, Australian National University
[2]Graduate School of Artifical Intelligence, Postech

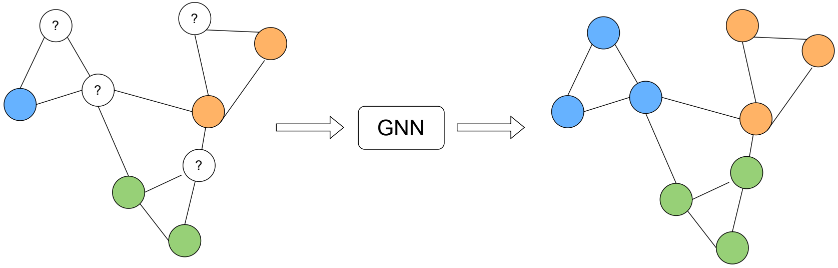
Australian National University

{shouheng.li; qing.wang}@anu.edu.au; dongwookim@postech.ac.kr

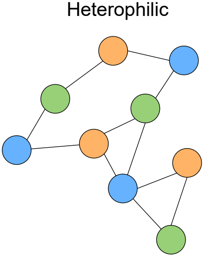
Agenda

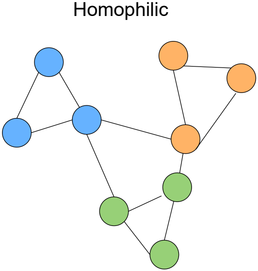
- Problem Setting
- Contribution
- Background
- Proposed Method
- Empirical Results

Problem Setting: Node Classification

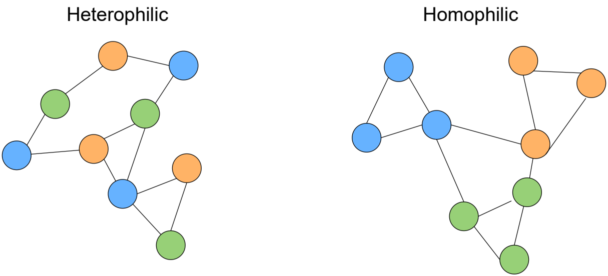


Problem Setting: Homophilic vs. Heterophlic Graphs





Problem Setting: Homophilic vs. Heterophlic Graphs

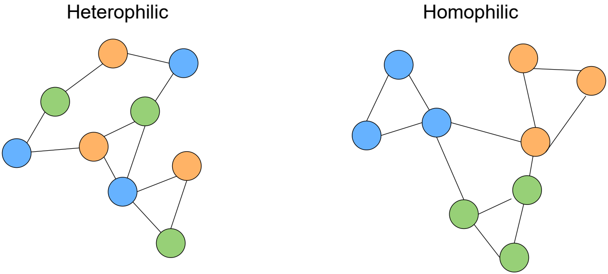


$$\beta = \frac{1}{N} \sum_{v \in V} \beta_v$$

$$\beta_v = \frac{|\{u \in \mathcal{N}_v | \ell(u) = \ell(v)\}|}{|\mathcal{N}_v|}$$

$\ell(v)$: label of node v.

Problem Setting: Homophilic vs. Heteophlic Graphs



- Existing GNNs work well on homophilic but not heterophilic graphs
- Outperformed by MLP using node features only

Proposed Method: Contributions

- A generalized GNN adapts well on graphs of different homophily levels
- A node attention mechanism using spectral filters learned from data
- Empirical analysis on importance of frequency components on homophilic and heterophilic graphs

Graph Neural Network (GNN)

GNN learns the embedding of node v by:

- AGGREGATE features from its neighbor nodes $u \in N_v$.

$$\boldsymbol{m}_v = \text{aggregate}(\{\boldsymbol{h}_u^{(l-1)} | u \in \mathcal{N}_v\})$$

$\boldsymbol{h}_u^{(l-1)}$ is the embedding of node u at the $(l-1)$th layer.
$\boldsymbol{m}_v$ is the aggregated feature from the neighbors.

Graph Neural Network (GNN)

GNN learns the embedding of node v in two steps:

- AGGREGATE features from its neighbor nodes $u \in N_v$.

$$\boldsymbol{m}_v = \text{aggregate}(\{\boldsymbol{h}_u^{(l-1)} | u \in \mathcal{N}_v\})$$

$\boldsymbol{h}_u^{(l-1)}$ is the embedding of node u at the $(l-1)$th layer.
$\boldsymbol{m}_v$ is the aggregated feature from the neighbors

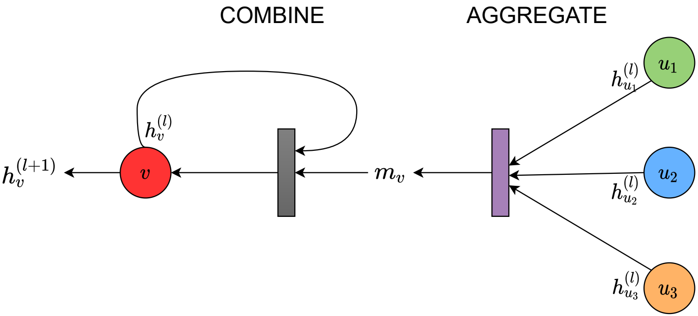
- COMBINE the aggregated features with its own.

$$\boldsymbol{h}_v^{(l)} = \text{combine}(\boldsymbol{h}_v^{(l-1)}, \boldsymbol{m}_v)$$

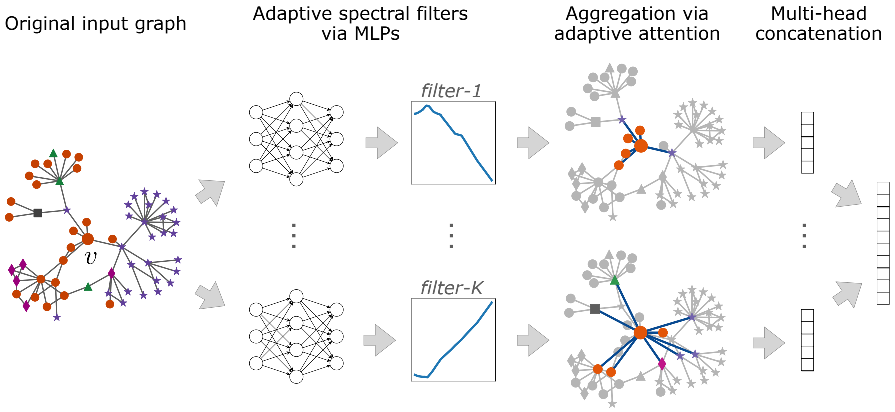
$\boldsymbol{h}_v^{(l)}$ is the embedding of node v at the lth layer.

Graph Neural Network (GNN)

GNN learns the embedding of node v by AGGREGATE and COMBINE:



Proposed Method: ASGAT



Proposed Method: Preliminary

- Let $\mathcal{G} = (V, E, A, \boldsymbol{x})$ be an undirected graph with N nodes
- V, E, and A are the node set, edge set, and adjacency matrix of $\mathcal{G}$
- $\boldsymbol{x}$ is the node feature matrix.

Proposed Method: Preliminary

- Let $\mathcal{G} = (V, E, A, \boldsymbol{x})$ be an undirected graph with N nodes
- V, E, and A are the node set, edge set, and adjacency matrix of $\mathcal{G}$
- $\boldsymbol{x}$ is the node feature matrix.
- $\Lambda =$ diag$(\lambda_1, ..., \lambda_N)$ is a diagonal matrix of eigenvales, g is the spectral filter.

$$\text{Graph Laplacian: } \boldsymbol{L} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2} = \boldsymbol{U}\Lambda\boldsymbol{U}^H$$

$$\text{Spectral Convolution: } g(\boldsymbol{L})\vec{x} = \boldsymbol{U}g(\Lambda)\hat{\vec{x}}.$$

Proposed Method: ASGAT

Compute spectral filtering response $\boldsymbol{\psi}_v$

$$\boldsymbol{\psi}_v = \boldsymbol{U}\text{diag}(\text{MLP}(\Lambda))\boldsymbol{U}^H\delta_v \tag{1}$$

Perform sparsification

$$\bar{\psi}_{vu} = \begin{cases} \psi_{vu} & \text{if } \psi_{vu} \in \text{topK}(\{\psi_{v0}, ..., \psi_{vN}\}, k) \\ -\infty & \text{otherwise} \end{cases}$$

Obtain attention weight $\boldsymbol{a}_v$

$$\boldsymbol{a}_v = \text{softmax}(\bar{\boldsymbol{\psi}}_v)$$

Equation 1 can be approximated using polynomial or rational functions without eigen-decomposition of L

Proposed Method: ASGAT

Compute node embedding: AGGREGATE and COMBINE

$$\boldsymbol{h}_v^{(l)} = \sigma\left(\sum_{u=1}^{N} a_{vu}\boldsymbol{h}_u^{(l-1)}\boldsymbol{W}^{(l)}\right),$$

where $\boldsymbol{W}^{(l)}$ is a weight matrix shared across all nodes at the lth layer and σ is ELU nonlinear activation.

Empirical Results: Accuracy

	Homophily ⟸⟹ Heterophily							
	CORA	PUBMED	CITESEER	CHAMELEON	SQUIRREL	WISCONSIN	CORNELL	TEXAS
β	0.83	0.79	0.71	0.25	0.22	0.16	0.11	0.06
#Nodes	2,708	19,717	3,327	2,277	5,201	251	183	183
#Edges	5,429	44,338	4,732	36,101	217,073	515	298	325
#Features	1,433	500	3,703	2,325	2,089	1,703	1,703	1,703
#Classes	7	3	6	5	5	5	5	5
GCN	87.4 ± 0.2	87.8 ± 0.2	78.5 ± 0.5	$59.8 \pm 2.6^{\ddagger}$	$36.9 \pm 1.3^{\ddagger}$	64.1 ± 6.3	59.2 ± 3.2	64.1 ± 4.9
ChevNet	88.2 ± 0.2	89.3 ± 0.3	79.4 ± 0.4	66.0 ± 2.3	39.6 ± 3.0	82.5 ± 2.8	76.5 ± 9.4	79.7 ± 5.0
ARMANet	85.2 ± 2.5	86.3 ± 5.7	76.7 ± 0.5	62.1 ± 3.6	47.8 ± 3.5	78.4 ± 4.6	74.9 ± 2.9	82.2 ± 5.1
GAT	87.6 ± 0.3	83.0 ± 0.1	77.7 ± 0.3	$54.7 \pm 2.0^{\ddagger}$	$30.6 \pm 2.1^{\ddagger}$	62.0 ± 5.2	58.9 ± 3.3	60.0 ± 5.7
SGC	87.2 ± 0.3	81.1 ± 0.3	78.8 ± 0.4	33.7 ± 3.5	46.9 ± 1.7	51.8 ± 5.9	58.1 ± 4.6	58.9 ± 6.1
GraphSAGE	86.3 ± 0.6	89.2 ± 0.5	77.4 ± 0.5	51.1 ± 0.5	$41.6 \pm 0.7^{\ddagger}$	77.6 ± 4.6	67.3 ± 6.9	82.7 ± 4.8
APPNP	**88.4** ± 0.3	86.0 ± 0.3	77.6 ± 0.6	45.3 ± 1.6	31.0 ± 1.6	81.2 ± 2.5	70.3 ± 9.3	79.5 ± 4.6
Geom-GCN	86.3 ± 0.3	89.1 ± 0.1	**81.4** ± 0.3	$60.9^{\dagger}$	$38.1^{\dagger}$	$64.1^{\dagger}$	$60.8^{\dagger}$	$67.6^{\dagger}$
H_2GCN	88.3 ± 0.3	89.1 ± 0.4	78.4 ± 0.5	59.4 ± 2.0	37.9 ± 2.0	86.5 ± 4.4	82.2 ± 6.0	82.7 ± 5.7
MLP	72.1 ± 1.3	88.6 ± 0.2	74.9 ± 1.8	45.7 ± 2.7	28.1 ± 2.0	82.7 ± 4.5	81.4 ± 6.3	79.2 ± 6.1
Vanilla ASGAT	–	–	–	–	–	**86.9** ± 4.2	**84.6** ± 5.8	82.2 ± 3.2
ASGAT-Cheb	87.5 ± 0.5	**89.9** ± 0.9	79.3 ± 0.6	**66.5** ± 2.8	**55.8** ± 3.2	86.3 ± 3.7	82.7 ± 8.3	**85.1** ± 5.7
ASGAT-ARMA	87.4 ± 1.1	88.3 ± 1.0	79.2 ± 1.4	65.8 ± 2.2	51.4 ± 3.2	84.7 ± 4.4	83.2 ± 5.5	79.5 ± 7.7

Empirical Results: Accuracy (Short)

	Homophily ⟸					⟹ Heterophily		
	CORA	PUBMED	CITESEER	CHAMELEON	SQUIRREL	WISCONSIN	CORNELL	TEXAS
GCN	87.4 ± 0.2	87.8 ± 0.2	78.5 ± 0.5	$59.8 \pm 2.6^{\ddagger}$	$36.9 \pm 1.3^{\ddagger}$	64.1 ± 6.3	59.2 ± 3.2	64.1 ± 4.9
ChevNet	88.2 ± 0.2	89.3 ± 0.3	79.4 ± 0.4	66.0 ± 2.3	39.6 ± 3.0	82.5 ± 2.8	76.5 ± 9.4	79.7 ± 5.0
ARMANet	85.2 ± 2.5	86.3 ± 5.7	76.7 ± 0.5	62.1 ± 3.6	47.8 ± 3.5	78.4 ± 4.6	74.9 ± 2.9	82.2 ± 5.1
GAT	87.6 ± 0.3	83.0 ± 0.1	77.7 ± 0.3	$54.7 \pm 2.0^{\ddagger}$	$30.6 \pm 2.1^{\ddagger}$	62.0 ± 5.2	58.9 ± 3.3	60.0 ± 5.7
SGC	87.2 ± 0.3	81.1 ± 0.3	78.8 ± 0.4	33.7 ± 3.5	46.9 ± 1.7	51.8 ± 5.9	58.1 ± 4.6	58.9 ± 6.1
GraphSAGE	86.3 ± 0.6	89.2 ± 0.5	77.4 ± 0.5	51.1 ± 0.5	$41.6 \pm 0.7^{\ddagger}$	77.6 ± 4.6	67.3 ± 6.9	82.7 ± 4.8
APPNP	**88.4** ± 0.3	86.0 ± 0.3	77.6 ± 0.6	45.3 ± 1.6	31.0 ± 1.6	81.2 ± 2.5	70.3 ± 9.3	79.5 ± 4.6
Geom-GCN	86.3 ± 0.3	89.1 ± 0.1	**81.4** ± 0.3	$60.9^{\dagger}$	$38.1^{\dagger}$	$64.1^{\dagger}$	$60.8^{\dagger}$	$67.6^{\dagger}$
H_2GCN	88.3 ± 0.3	89.1 ± 0.4	78.4 ± 0.5	59.4 ± 2.0	37.9 ± 2.0	86.5 ± 4.4	82.2 ± 6.0	82.7 ± 5.7
MLP	72.1 ± 1.3	88.6 ± 0.2	74.9 ± 1.8	45.7 ± 2.7	28.1 ± 2.0	82.7 ± 4.5	81.4 ± 6.3	79.2 ± 6.1
Vanilla ASGAT	–	–	–	–	–	**86.9** ± 4.2	**84.6** ± 5.8	82.2 ± 3.2
ASGAT-Cheb	87.5 ± 0.5	**89.9** ± 0.9	79.3 ± 0.6	**66.5** ± 2.8	**55.8** ± 3.2	86.3 ± 3.7	82.7 ± 8.3	**85.1** ± 5.7
ASGAT-ARMA	87.4 ± 1.1	88.3 ± 1.0	79.2 ± 1.4	65.8 ± 2.2	51.4 ± 3.2	84.7 ± 4.4	83.2 ± 5.5	79.5 ± 7.7

Empirical Results: Heterophily

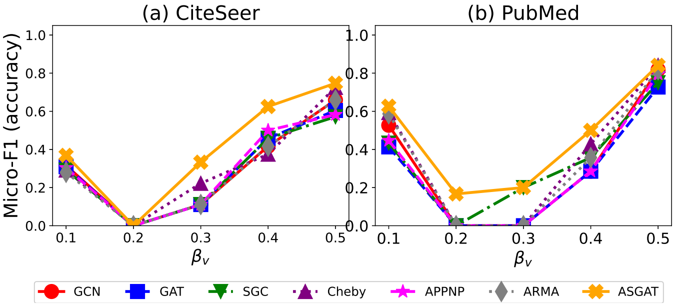


Figure: Classification accuracy on heterophilic nodes ($\beta_v \leq 0.5$)

Summary

- Existing GNNs perform sub-optimally in node classification on heterophilic graphs
- The proposed ASGAT uses attentions derived from learnable spectral filters to achieve adaptive feature aggregation on graphs of different homophily levels
- ASGAT outperforms benchmarks on hetereophilic graphs while performs comparably on homophilic ones, demonstrated adaptiveness