Towards Deep Attention in Graph Neural Networks: Problems and Remedies

Soo Yong Lee [1], Fanchen Bu [1], Jaemin Yoo [2], and Kijung Shin [1]

[1] KAIST, [2] CMU

[1]{syleetolow, boqvezen97, kijungs}@kaist.ac.kr, [2]{jaeminyoo}@cmu.edu

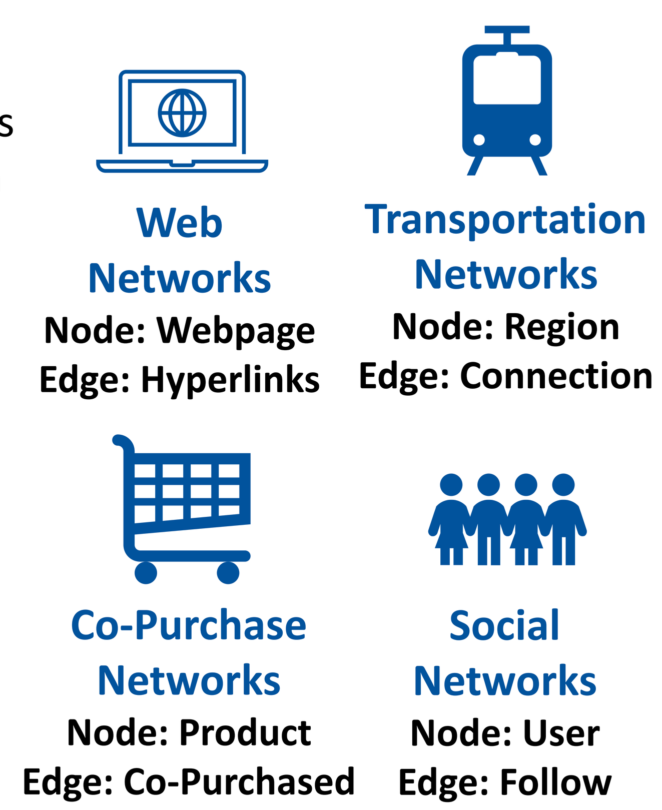
Code and Data: https://github.com/syleeheal/AERO-GNN

Summary

- **Limitations in Building Deep Graph Attention**
  - We identify **two problems** that limit existing attention-based GNNs from remaining expressive at deep layers
- **Proposed Solution**
  - To mitigate the problems, we propose a novel GNN architecture, **A**ttentive d**E**ep p**RO**pagation-**GNN** (**AERO-GNN**)
- **Theoretical and Empirical Results**
  - AERO-GNN provably mitigates the proposed problems
  - AERO-GNN outperforms baselines models in node classification task by learning the most adaptive and less smooth attention

Introduction

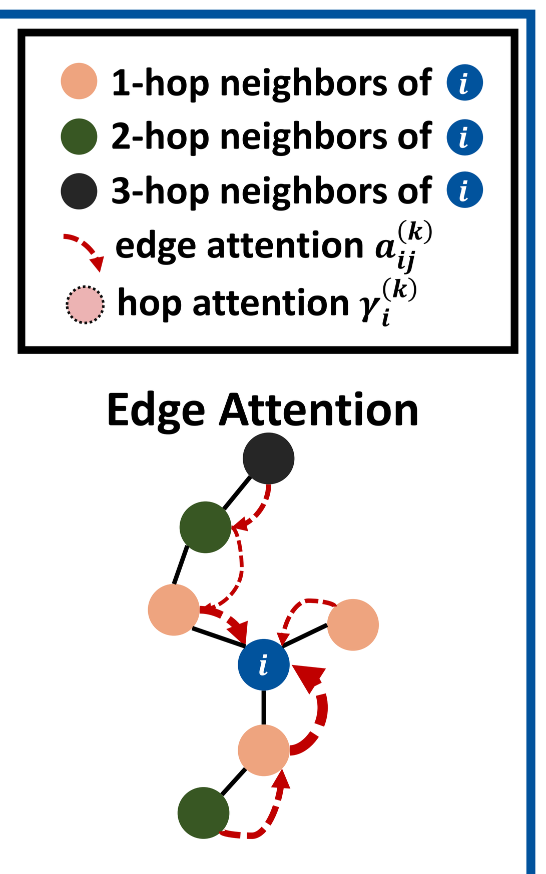
- **Graphs**
  - Relational data that consists of nodes and edges
  - Real-world networks can be expressed as graph
- **Graph Neural Networks (GNNs)**
  - Graph representation learning neural network
  - Can solve various downstream tasks on graphs
  - To enhance it expressiveness:
    - ☑ Graph Attention
    - ☑ Deeper GNNs
- **Research Question**
  - ***Can existing attention-based GNNs remain expressive over deep layers?***



Analysis of Graph Attention

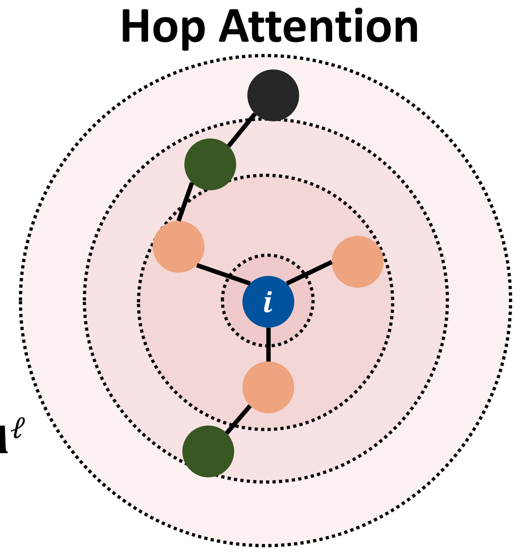
- **Graph Attention**
  - **Intuition**: Relational importance among node pairs
  - For GNNs, att. coefficient is the weight for feat. propagation
- **Edge-Attention GNNs**
  - **Intuition**: Learn neighbor importance *within* each hop
  - **Edge Attention Matrix**: $A^{(k)} = \left(a_{ij}^{(k)}\right) \in \mathbb{R}^{n \times n}$
    - ☑ $a_{ij}^{(k)}$ is an edge att. coef. btw. node i and j at layer k
  - **Feature Propagation**: Weighted for direct neighbors
  - **Propagation Function**: $H'^{(k)} = A^{(k)} H^{(k-1)}$
    - ☑ $H'^{(k)}$ is an agg. of node feat. $H^{(k-1)}$, based on $A^{(k)}$
  - **Models**: GATs, FAGCN, etc.

Edge Attention



- **Hop-Attention GNNs**
  - **Intuition**: Learn neighbor importance *of* each hop
  - **Hop Attention Matrix**: $\Gamma^{(k)} = diag\left(\gamma_i^{(k)}\right) \in \mathbb{R}^{n \times n}$
    - ☑ $\gamma_i^{(k)}$ is a hop att. coef. of node i for k-hop neighbors
  - **Feature Propagation**: Weighted for k-hop neighbors
  - **Propagation Function**: $Z^{(k)} = \sum_{\ell=0}^{k} \Gamma^{(\ell)} H^{(\ell)} = \sum_{\ell=0}^{k} \Gamma^{(\ell)} A^{\ell} H^{(0)}$
    - ☑ $Z^{(k)}$ is an agg. of node feat. $H^{(0)}$, based on $\Gamma^{(\ell)}$ and A^{ℓ}
  - **Models**: DAGNN, GPRGNN, etc.

Hop Attention



Proposed Model: AERO-GNN

- **Model Overview**

$$H^{(k)} = \begin{cases} \mathrm{MLP}(X), & \text{if } k = 0, \\ \mathcal{A}^{(k)} H^{(k-1)}, & \text{if } 1 \le k \le k_{max}, \end{cases}$$

$$Z^{(k)} = \sum_{l=0}^{k} \Gamma^{(l)} H^{(l)}, \forall 1 \le k \le k_{max},$$

$$Z^* = \sigma(Z^{(k_{max})}) W^*,$$

$X \in \mathbb{R}^{n \times d}$ is input feat.
$H \in \mathbb{R}^{n \times d}$ is hidden feat.
$Z \in \mathbb{R}^{n \times d}$ is layer-aggregated feat.
$Z^* \in \mathbb{R}^{n \times c}$ is the final prediction
$W^* \in \mathbb{R}^{d \times c}$ is a learnable weight
$A = (a_{ij}) \in \mathbb{R}^{n \times n}$ is edge att.
$\Gamma = diag(\gamma_i) \in \mathbb{R}^{n \times n}$ is hop att.
k is #(layers), n is #(nodes)
d is #(dimensions), c is #(classes)

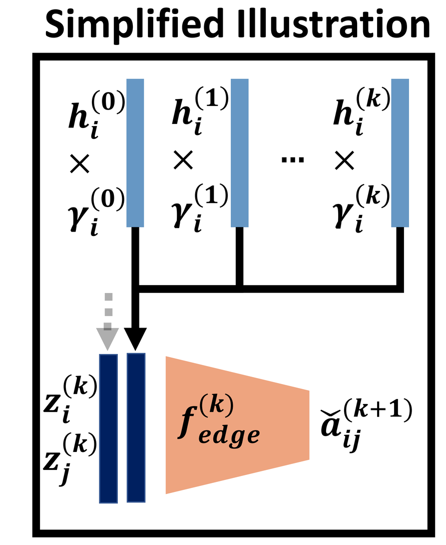
- **AERO-GNN** first transforms input feat. X with an MLP, then propagates the feat. $H^{(0)}$ with edge att. $A^{(k)}$ and hop att. $\Gamma^{(k)}$ to learn the final node representation $Z^{(k)}$.

- **Computing Edge Attention**
  - **Edge Attention Function**

$$\tilde{\alpha}_{ij}^{(k)} = \mathrm{softplus}((W_{edge}^{(k)})^{\top} \sigma(\tilde{Z}_i^{(k-1)} \| \tilde{Z}_j^{(k-1)}))$$

  - **Input**: $Z^{(k)}$ enables att. function to consider node feat. $H^{(\ell)}$ from all previous layers
  - **Non-linearity**: MLP layer to compute att. coef.
  - **Softplus**: Positively map att. coef.
  - **Normalization**: Symmetric normalization used

Edge Attention: Simplified Illustration

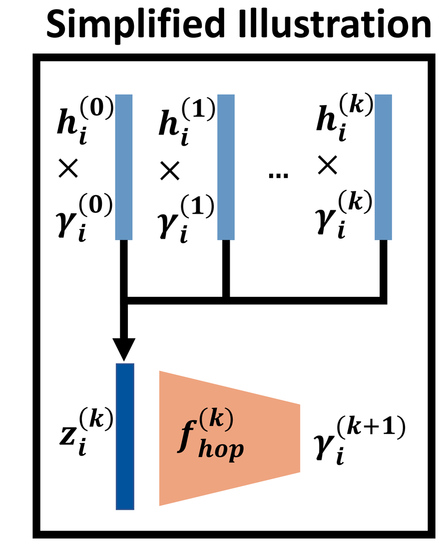


- **Computing Hop Attention**
  - **Hop Attention Function**

$$\gamma_i^{(k)} = (W_{hop}^{(k)})^{\top} \sigma(H_i^{(k)} \| \tilde{Z}_i^{(k-1)}) + b_{hop}^{(k)}$$

  - **Input**: $Z^{(k)}$ enables att. function to consider node feat. $H^{(\ell)}$ from all previous layers
  - **Non-linearity**: MLP layer to compute att. coef.
  - **Negative Att.**: Allows negative attention coefficient
  - **Node-Adaptive**: Each node may have different hop att.

Hop Attention: Simplified Illustration



Theoretical Results

- **Theoretical Limitations to Deep Graph Attention**
  - Problem 1: Vulnerability to Over-Smoothing
    - ☑ (Informal) The att. coef. become identical for over-smoothed node feat.
  - Problem 2: Smooth Cumulative Attention $T^{(k)}$
    - ☑ (Informal) Cumul. att. vectors $T_i^{(k)}$ become identical for nodes i at deep layer
    - ☑ **Cumulative Attention Matrix** $T^{(k)}$
      - ► **Intuition**: Expresses att. at k^{th} layer, considering both edge and hop att.
      - ► $T^{(k)} = \Gamma^{(k)} \prod_{\ell=k}^{1} A^{(\ell)} \in \mathbb{R}^{n \times n}$
- **Properties of Attention Functions**
  - Att. function of AERO-GNN is the most flexible, with many desirable properties
  - ***The properties allows only AERO-GNN to provably mitigate the proposed problems of deep graph att.***

Model	Edge & Hop Att.	Use Z as Input	Use MLP to Score Att.	Negative Att.	Node-Adaptive	Resistance to Over-Smoothing	Non-Smooth Cumul. Att.
GATv2	X	X	O	X	X	X	X
FAGCN	X	O	O	O	X	△	X
GPRGNN	X	X	X	O	X	X	X
DAGNN	X	X	X	X	O	X	X
AERO-GNN	O	O	O	O	O	O	O

Experimental Results

Task: Node Classification

Datasets: Citation Graphs, Co-Purchase Graphs, Webpage Graphs

Evaluation: **Measure:** Accuracy; **#(Trials):** 100; **Perf.:** Mean ± StDev

Baselines: Simple GNNs, Deep GNNs, Edge-Att. GNNs, Hop-Att. GNNs

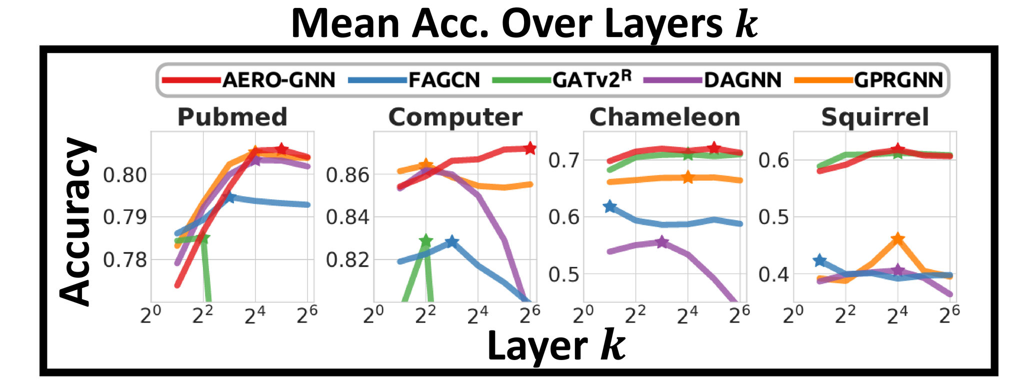
- **Node Classification Performance**
  - AERO-GNN significantly outperforms the baselines

Dataset	Chameleon	Squirrel	Actor	Texas	Cornell	Wisconsin	Computer	Photo	Wiki-CS	Pubmed	Citeseer	Cora	A.R.
Homophily	0.04	0.03	0.01	0.00	0.02	0.05	0.70	0.77	0.57	0.66	0.63	0.77	
GCN	67.97 ± 2.5	53.33 ± 1.3	30.57 ± 0.7	65.65 ± 4.8	58.41 ± 3.3	62.02 ± 5.9	81.27 ± 1.4	90.24 ± 1.3	79.08 ± 0.5	79.54 ± 0.4	72.50 ± 0.5	83.15 ± 0.5	9.1
APPNP	53.04 ± 2.2	40.37 ± 1.5	35.49 ± 1.0	79.89 ± 4.2	80.16 ± 5.9	84.24 ± 4.6	81.27 ± 1.4	91.12 ± 1.2	79.05 ± 0.5	79.90 ± 0.3	73.06 ± 0.3	83.60 ± 1.3	7.8
GCN-II	60.38 ± 1.9	48.76 ± 2.4	35.77 ± 1.0	78.59 ± 6.6	78.84 ± 6.6	83.20 ± 4.7	84.24 ± 1.2	91.81 ± 0.9	79.28 ± 0.6	80.14 ± 0.6	**73.20 ± 0.8**	**85.33 ± 0.5**	5.5
A-DGN	69.63 ± 2.0	57.77 ± 1.9	36.41 ± 1.0	82.22 ± 4.8	**83.14 ± 6.7**	**85.84 ± 4.0**	83.70 ± 1.5	90.53 ± 1.3	79.11 ± 0.6	78.68 ± 0.6	70.16 ± 0.9	79.84 ± 0.9	6.4
GAT	68.01 ± 2.5	54.49 ± 1.7	30.36 ± 0.9	60.46 ± 6.2	58.22 ± 3.7	63.59 ± 6.1	84.46 ± 1.3	89.88 ± 1.1	79.44 ± 0.5	78.94 ± 0.4	71.89 ± 0.6	83.78 ± 0.5	8.5
GATv2	69.06 ± 2.2	57.67 ± 2.4	30.27 ± 0.8	60.32 ± 7.0	58.35 ± 3.8	61.94 ± 4.7	84.19 ± 1.2	89.87 ± 1.2	79.64 ± 0.5	79.12 ± 0.3	71.15 ± 1.2	83.88 ± 0.6	8.9
GATv2^R	**70.88 ± 1.9**	**61.23 ± 1.5**	33.73 ± 0.9	60.68 ± 6.6	57.32 ± 4.5	60.61 ± 5.1	81.73 ± 2.2	88.71 ± 1.7	79.75 ± 0.6	78.28 ± 0.4	71.00 ± 0.8	82.42 ± 0.6	9.3
GT	69.34 ± 1.2	55.04 ± 1.9	36.29 ± 1.0	**84.08 ± 5.6**	80.00 ± 4.9	**84.80 ± 4.3**	84.38 ± 1.3	91.28 ± 1.1	**79.93 ± 0.5**	79.04 ± 0.5	70.16 ± 0.8	82.09 ± 0.7	5.6
FAGCN	60.98 ± 2.3	42.20 ± 1.8	35.67 ± 0.9	77.00 ± 7.7	78.32 ± 6.3	82.41 ± 3.8	82.79 ± 2.7	91.99 ± 1.0	79.27 ± 0.6	79.19 ± 0.4	71.55 ± 0.8	83.88 ± 0.5	7.5
DMP	63.79 ± 4.1	34.19 ± 7.6	28.30 ± 2.7	66.08 ± 7.0	56.41 ± 5.5	62.73 ± 4.5	70.58 ± 11.3	82.63 ± 4.1	56.41 ± 7.8	70.07 ± 4.1	59.12 ± 4.4	75.05 ± 3.8	12.8
MixHop	60.30 ± 2.1	41.05 ± 2.0	**36.48 ± 1.2**	77.73 ± 7.3	75.95 ± 5.7	82.12 ± 4.5	79.52 ± 2.1	89.45 ± 1.5	78.59 ± 0.7	80.10 ± 0.4	71.42 ± 0.9	81.61 ± 0.8	9.3
GPRGNN	66.92 ± 1.7	46.32 ± 1.5	35.58 ± 0.9	81.51 ± 6.6	76.86 ± 7.1	84.06 ± 5.2	85.82 ± 0.9	**92.41 ± 0.7**	79.67 ± 0.5	80.28 ± 0.4	71.59 ± 0.8	84.20 ± 0.5	**5.2**
DAGNN	54.99 ± 2.0	40.03 ± 1.4	33.69 ± 1.0	61.35 ± 6.1	63.89 ± 7.0	62.27 ± 4.2	**85.83 ± 0.8**	92.30 ± 0.7	79.31 ± 0.6	**80.44 ± 0.5**	73.16 ± 0.6	**84.43 ± 0.5**	7.2
AERO-GNN	**71.58 ± 2.4**	**61.76 ± 2.4**	**36.57 ± 1.1**	**84.35 ± 5.2**	**81.24 ± 6.8**	**84.80 ± 3.3**	**86.69 ± 1.4**	**92.50 ± 0.7**	**79.95 ± 0.5**	**80.59 ± 0.5**	**73.20 ± 0.6**	83.90 ± 0.5	**1.4**

• In each column, ■ indicates ranking the first, and ■ indicates ranking the second. A.R. denotes average ranking.

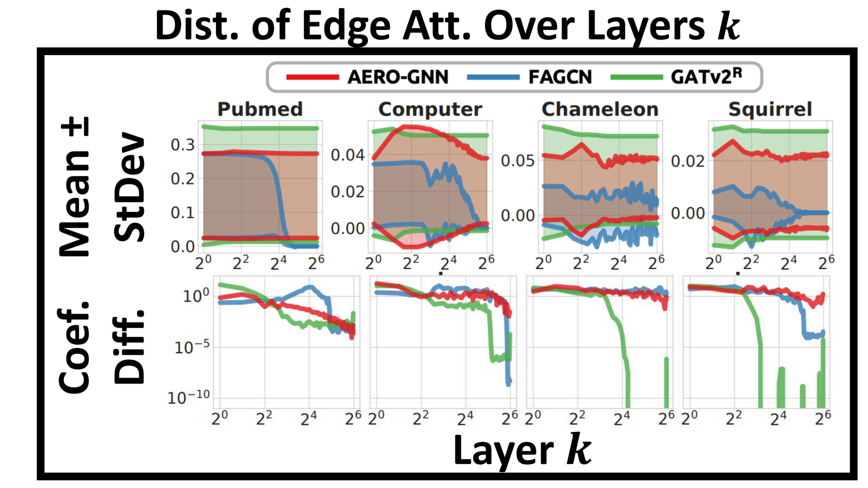
- **Performance Over Layers**
  - AERO-GNN's perf. Increases over layer k (**RED line trend**)
  - AERO-GNN achieves higher best perf. (**Higher ★**)

Mean Acc. Over Layers k

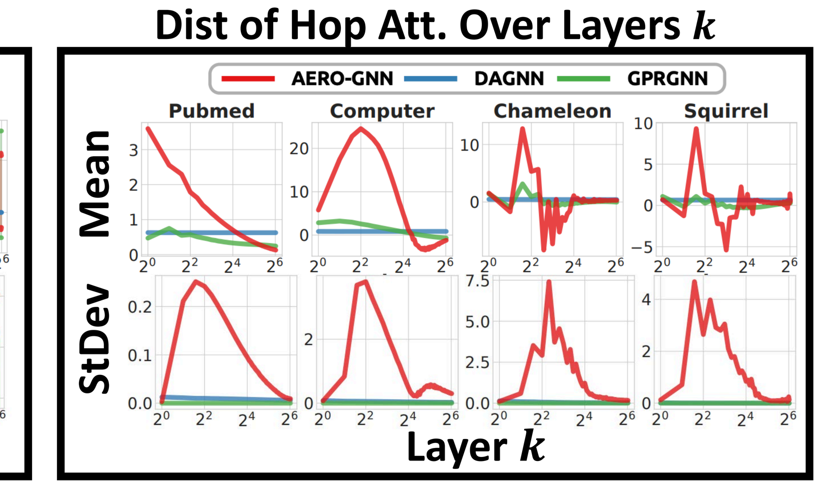


- **Post-Hoc Analysis of Att.**
  - AERO-GNN (**RED**) learns the most adaptive att. coef., when stacked deep layers

Dist. of Edge Att. Over Layers k

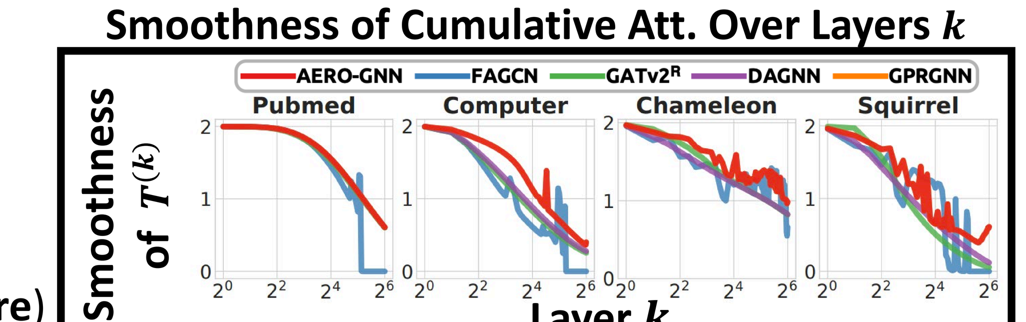


Dist of Hop Att. Over Layers k



- AERO-GNN's (**RED**) cumul. att. $T^{(k)}$ is the least smooth (**high smoothness score**), and often un-smoothes (**increasing smoothness score**)

Smoothness of Cumulative Att. Over Layers k



Discussion: Implications

Attention-Based GNNs

A larger focus has been placed on designing **a more expressive layer**

- *with new designs*
- *with new loss terms*
- *with more feat.*

Deep GNNs

Making deeper GNNs have been an important **setback to GNN research**

- *over-smoothing*
- *over-squashing*
- *over-correlation*

We Bridged the Two

The two are complementary

- *deep graph attention*
- *higher node cls. perf.*