Aggregation Buffer:

Revisiting DropEdge with a New Parameter Block

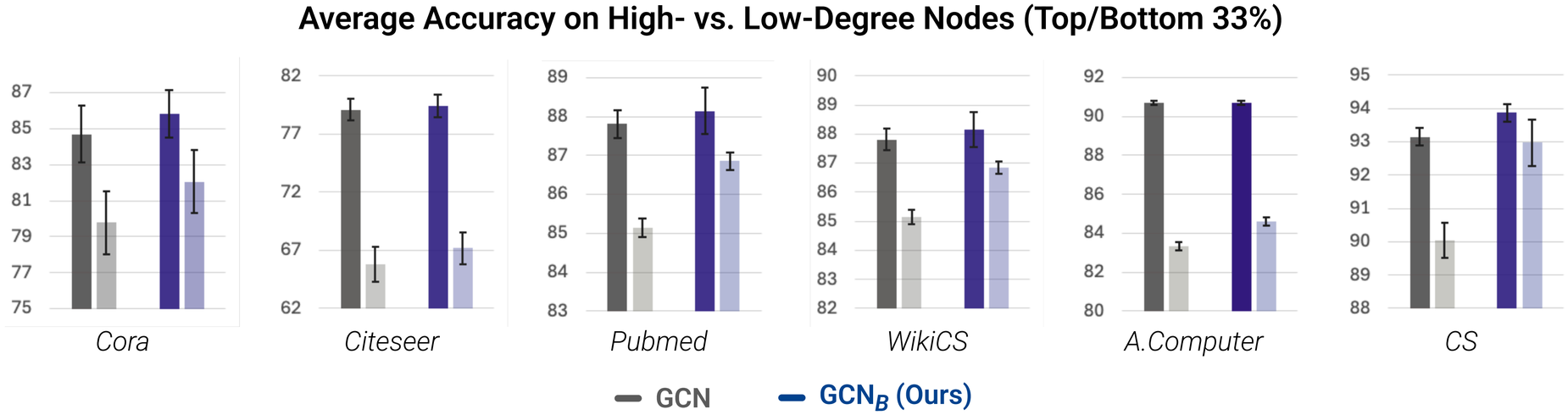
Dooho Lee [1] **Myeong Kong** [1] **Sagad Hamid** [1 2] **Cheonwoo Lee** [1] **Jaemin Yoo** [1]

[1]School of Electrical Engineering, KAIST, Daejeon, Republic of Korea
[2]Computer Science Department, University of Münster, Münster, Germany

Sub-optimalites of GNNs : Structural Inconsistencies

1) Degree Bias

GNNs perform ***worse on low-degree*** nodes than on high-degree nodes, especially in homophilous graphs.



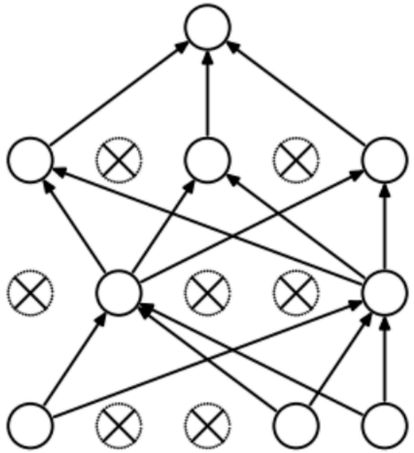
2) Structural Disparity (Mao et al., 2024)

GNNs exhibit poor accuracy on nodes whose neighbors have ***conflicting structural properties***, such as heterophilous neighbors in homophilous graphs, or vice versa.

We can frame several open problems in GNNs under the theme of “structural inconsistency”.

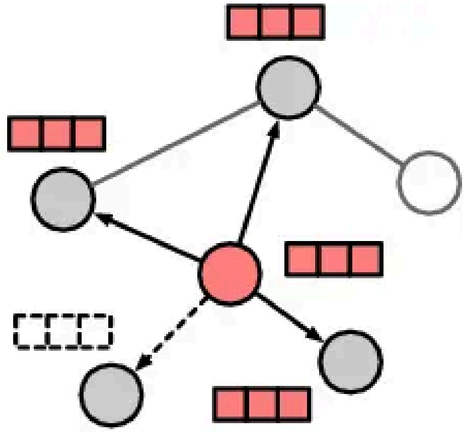
Random Dropping in ML

Common approaches to enhance robustness against input variations is training with ***random dropping***.



DropOut
(Srivastava et al., 2014)

CutOut
(DeVries, 2017)



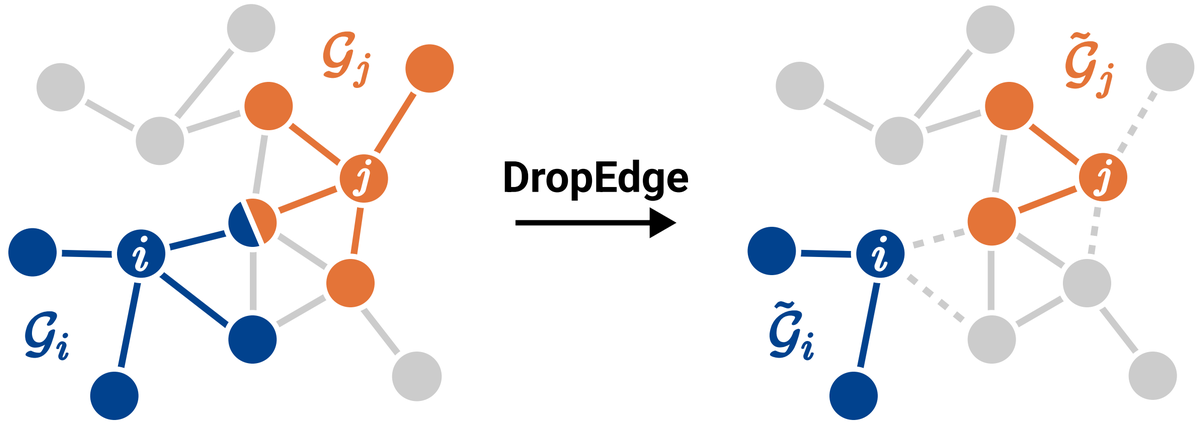
DropEdge
(Rong et al., 2019)

However, in the graph domain, the ***performance gain by DropEdge is limited in practice***, and DropEdge is often excluded from the standard hyperparameter search space of GNNs in benchmark studies.

Why hasn't DropEdge become a default in GNNs like other data augmentations in different domains?

Revisiting DropEdge : Objective Shift

For each node i, the edge removal operation in DropEdge can be interpreted as transforming the rooted subgraph $\mathcal{G}_i$, centered on node i , into a ***reduced rooted subgraph***, denoted as $\tilde{\mathcal{G}}_i$.



$$\mathcal{L}(\theta) = D_{\mathrm{KL}}(P(\boldsymbol{y}_i|\mathcal{G}_i)\|Q(\boldsymbol{y}_i|\mathcal{G}_i)) \longrightarrow \tilde{\mathcal{L}}(\theta) = D_{\mathrm{KL}}(P(\boldsymbol{y}_i|\mathcal{G}_i)\|Q(\boldsymbol{y}_i|\tilde{\mathcal{G}}_i))$$

** **P** - True Distribution , **Q** - Modeled Distribution by GNNs*

Revisiting DropEdge : Bias-robustness Tradeoff

The shifted objective $\tilde{\mathcal{L}}$ can be decomposed as follows:

$$\tilde{\mathcal{L}}(\theta) = D_{\mathrm{KL}}(P(\boldsymbol{y}_i|\mathcal{G}_i)\|Q(\boldsymbol{y}_i|\mathcal{G}_i)) + \mathbb{E}_P[\log Q(\boldsymbol{y}_i|\mathcal{G}_i) - \log Q(\boldsymbol{y}_i|\tilde{\mathcal{G}}_i)]$$

By assuming, $Q \approx P$

$$\tilde{\mathcal{L}}_{\mathrm{Q}}(\theta) = \underbrace{D_{\mathrm{KL}}(P(\boldsymbol{y}_i|\mathcal{G}_i)\|Q(\boldsymbol{y}_i|\mathcal{G}_i))}_{\textbf{\textit{Bias}}\ \textit{(Standard objective)}} + \underbrace{D_{\mathrm{KL}}(Q(\boldsymbol{y}_i|\mathcal{G}_i)\|Q(\boldsymbol{y}_i|\tilde{\mathcal{G}}_i))}_{\textbf{\textit{Robustness}}\ \textit{(Structural consistency)}}$$

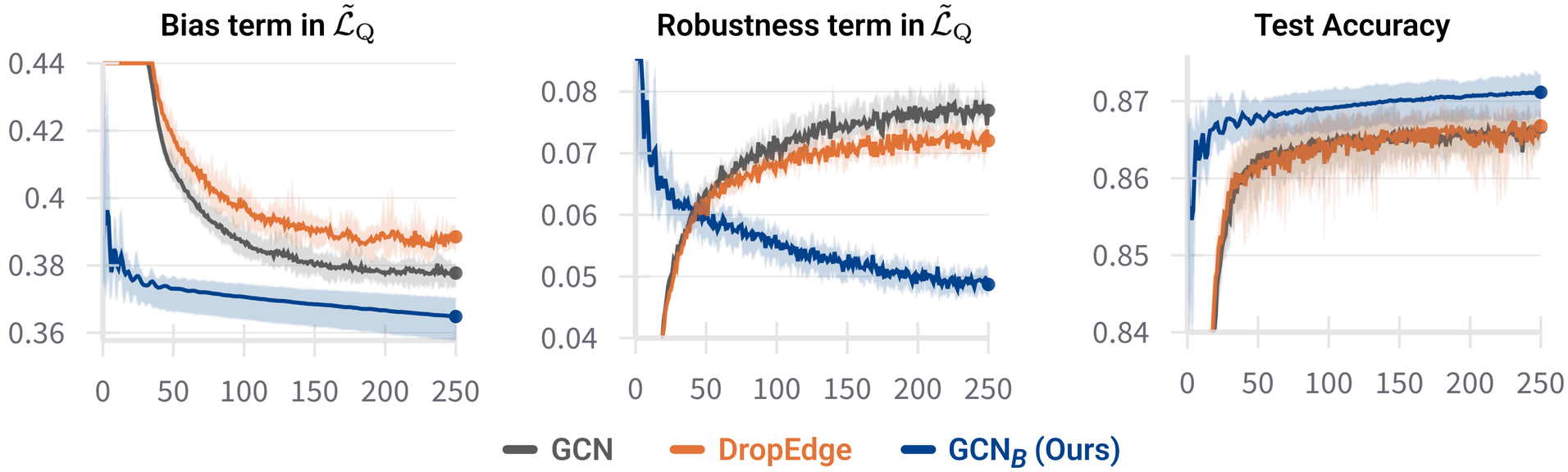
The second term works as a regularizer, promoting consistency across different reduced rooted subgraphs. Finding an optimal balance between ***bias and robustness*** is key to maximizing test performance.

Training with DropEdge is expected to improve robustness against structural inconsistency.

Unexpected Failure of DropEdge

In other domains, small perturbations of data do not significantly interfere with the primary learning objective .

**Accuracy and loss terms on test data during the training of a GCN at PubMed*



However, in GNNs trained with DropEdge, ***optimizing robustness immediately increases the bias term*** on test data, preventing sufficient robustness to be achieved.

However, in practice, robustness is not effectively optimized, even when training with DropEdge.

Reason of the Failure : Core Limitations of GNNs

Robustness term in $\tilde{\mathcal{L}}_{\mathrm{Q}}$ can be optimized only when a GNN is able to produce ***similar outputs from different inputs***—particularly for varying adjacency matrices, such as $\boldsymbol{A}$ and $\tilde{\boldsymbol{A}}$.

Definition 3.3 (Discrepancy bound)**.** Let $\boldsymbol{H}_1^{(l)}$ and $\boldsymbol{H}_2^{(l)}$ be the outputs of the l-th layer of a network f given different inputs $\boldsymbol{H}_1^{(l-1)}$ and $\boldsymbol{H}_2^{(l-1)}$. The discrepancy bound of f at the l-th layer is a constant C, such that

difference in **outputs** $\|\boldsymbol{H}_1^{(l)} - \boldsymbol{H}_2^{(l)}\|_2 \leq C\|\boldsymbol{H}_1^{(l-1)} - \boldsymbol{H}_2^{(l-1)}\|_2,$ difference in **inputs**

where C is independent of the specific inputs.

Theorem 3.9. *Under the same conditions as Theorem 3.8, the discrepancy of a GCN at layer l is bounded as*

$$\|\boldsymbol{H}_1^{(l)} - \boldsymbol{H}_2^{(l)}\|_2 \leq C_1\|\boldsymbol{H}_1^{(l-1)} - \boldsymbol{H}_2^{(l-1)}\|_2 + C_2,$$ **uncontrollable** term

where $C_1 = L_\sigma\|\boldsymbol{W}^{(l)}\|_2$, $C_2 = C_1|V|\|\hat{\boldsymbol{A}}_1 - \hat{\boldsymbol{A}}_2\|_2$, *and* $\hat{\boldsymbol{A}}$ *is the normalized adjacency matrices of* $\boldsymbol{A}$.

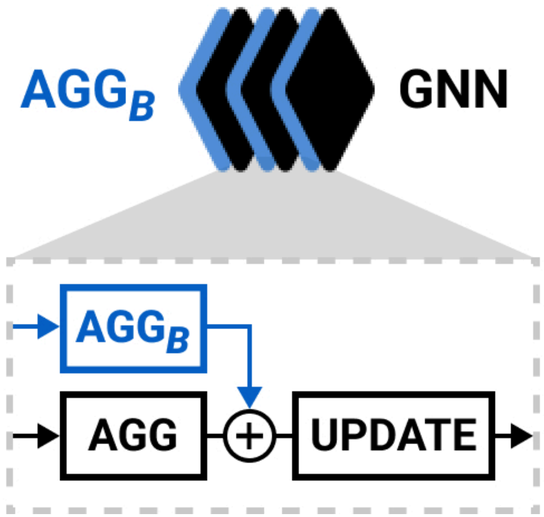
Robustness remains unoptimizable due to GNNs' aggregation operation, not DropEdge.

Achieving Edge-robustness

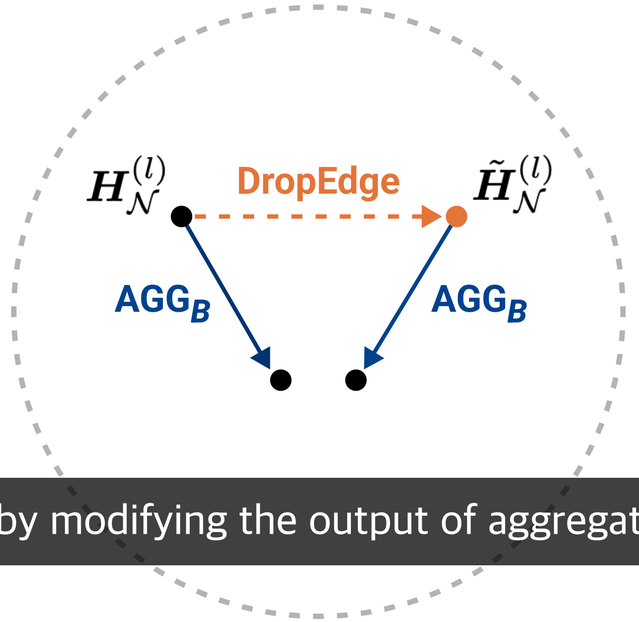
"fix right after where discrepancy arises"

"optimize robustness separately as a **post-processing**"

Propose an ***aggregation buffer*** (AGG_B), a new parameter block added to each layer of a ***frozen trained GNNs***, aims to resolve discrepancies caused by the aggregation operation.



$$\boldsymbol{H}_{\mathcal{N}}^{(l)} = \underbrace{\mathrm{AGG}^{(l)}(\boldsymbol{H}^{(l-1)}, \boldsymbol{A})}_{\textit{Original Aggregation}} + \underbrace{\mathrm{AGG}_B^{(l)}(\boldsymbol{H}^{(0:l-1)}, \boldsymbol{A})}_{\textit{Aggregation Buffer}}$$



Aggregation Buffer resolves discrepancy by modifying the output of aggregation.

Essential Conditions of AGG_B

C1: Edge-Awareness. When the adjacency matrix $\boldsymbol{A}$ is perturbed to $\tilde{\boldsymbol{A}}$, AGG_B should produce distinct outputs to compensate for structural changes:

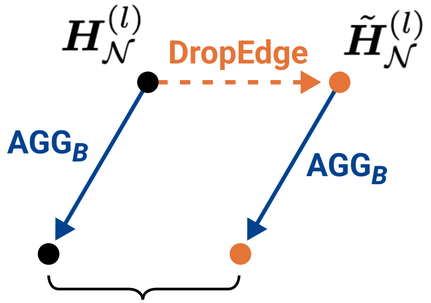
$$\mathrm{AGG}_B^{(l)}(\boldsymbol{A}) \neq \mathrm{AGG}_B^{(l)}(\tilde{\boldsymbol{A}}).$$

C2: Stability. For any perturbed adjacency matrix $\tilde{\boldsymbol{A}} \subset \boldsymbol{A}$ created by random edge dropping, AGG_B should produce outputs with a smaller deviation from the original output when given $\boldsymbol{A}$, compared to when given $\tilde{\boldsymbol{A}}$:

$$\|\mathrm{AGG}_B^{(l)}(\boldsymbol{A})\|_{\mathrm{F}} < \|\mathrm{AGG}_B^{(l)}(\tilde{\boldsymbol{A}})\|_{\mathrm{F}}.$$

When C1 is not satisfied

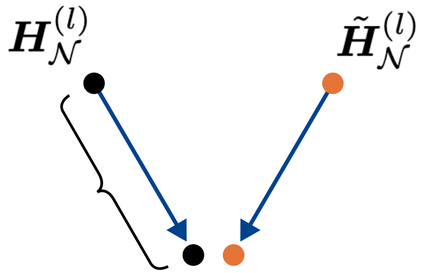
Ex) Residual Connection, JK-net



Discrepancy from adjacency changes remains unresolved

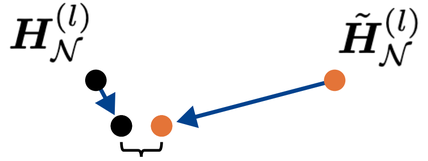
When C2 is not satisfied

Ex) Aggregations in GNNs



Trained knowledge is unnecessarily altered

When both C1, C2 satisfied



Reduce discrepancy while preserving learned knowledge

Our Solution. $(\boldsymbol{D}+\boldsymbol{I})^{-1}\boldsymbol{H}^{(0:l-1)}\boldsymbol{W}^{(l)}$

Train AGG_B with DropEdge

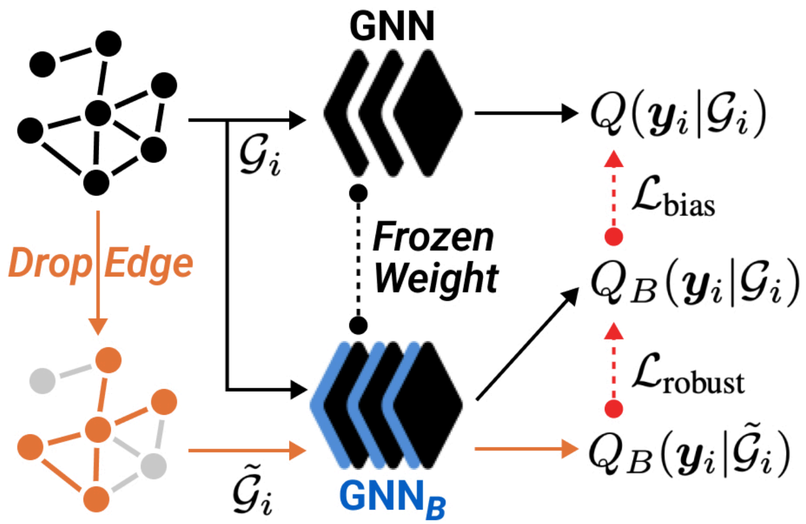
We train the AGG_B to minimize an objective function, ***robustness-controlled loss***, which has a few adjustments from the objective induced from the DropEdge.

$$\tilde{\mathcal{L}}_{\mathrm{Q}}(\theta) = D_{\mathrm{KL}}(P(\boldsymbol{y}_i|\mathcal{G}_i)\|Q(\boldsymbol{y}_i|\mathcal{G}_i)) + D_{\mathrm{KL}}(Q(\boldsymbol{y}_i|\mathcal{G}_i)\|Q(\boldsymbol{y}_i|\tilde{\mathcal{G}}_i))$$

By assuming, $Q \approx P$ and reformulate, $Q \rightarrow Q_B$

$$\mathcal{L}_{\mathrm{RC}}(\theta_B) = \frac{1}{|V_{\mathrm{trn}}|}\sum_{i\in V_{\mathrm{trn}}} D_{\mathrm{KL}}(Q(\boldsymbol{y}_i|\mathcal{G}_i)\|Q_B(\boldsymbol{y}_i|\mathcal{G}_i)) + \lambda\cdot\frac{1}{|V|}\sum_{i\in V} D_{\mathrm{KL}}(Q_B(\boldsymbol{y}_i|\mathcal{G}_i)\|Q_B(\boldsymbol{y}_i|\tilde{\mathcal{G}}_i))$$
$$= \mathcal{L}_{\mathrm{bias}}(\theta_B) + \lambda\cdot\mathcal{L}_{\mathrm{robust}}(\theta_B)$$

* λ is balancing hyper-parameter



Aggregation buffer is trained with DropEdge and effectively optimizes edge-robustness.

Experiments Setting

We use a two-layer GCN and report results averaged over ten runs with different splits. Hyper-parameters are selected via grid search based on validation accuracy from the first five runs.

Node Classification Datasets (12)

	Cora	Citeseer	PubMed	Wiki-CS	Photo	Computer	CS	Physics	Arxiv	Actor	Squirrel	Chameleon
# nodes	2,708	3,327	19,717	11,701	7,650	13,752	18,333	34,493	169,343	7,600	2,334	890
# edges	10,556	9,228	88,651	431,726	238,162	491,722	163,788	495,924	1,166,243	33,391	93,996	18,598
# features	1,433	3,703	500	300	745	767	6,805	8,415	128	932	2,089	2,325
# classes	7	6	3	10	8	10	15	5	40	5	5	5
Homophily Ratio	0.8100	0.7355	0.8024	0.6543	0.8272	0.7772	0.8081	0.9314	0.6542	0.2167	0.2072	0.2361

Baselines (7)

MLP, GCN, DropEdge(2019), DropNode(2020), DropMessage(2023), TUNEUP(2023), GraphPatcher(2024)

Random Dropping Methods in Graph ***Degree Bias Methods***

Table 1. Accuracy (%) of all models for test nodes grouped by degree. Head nodes refer to the top 33% of nodes by degree, while tail nodes refer to the bottom 33%. **Bold** values indicate the best performance, and underlined values indicate the second-best performance. Standard deviations are shown as subscripts. Our GCN_B achieves at least the second-best in 31 out of 36 settings.

Method	Cora	Citeseer	PubMed	Wiki-CS	A.Photo	A.Computer	CS	Physics	Arxiv	Actor	Squirrel	Chameleon
OVERALL PERFORMANCE												
MLP	$64.86_{\pm 1.21}$	$65.55_{\pm 0.76}$	$84.62_{\pm 0.28}$	$75.98_{\pm 0.51}$	$85.97_{\pm 0.81}$	$80.81_{\pm 0.40}$	$\mathbf{93.55_{\pm 0.18}}$	$95.09_{\pm 0.12}$	$56.41_{\pm 0.14}$	$\mathbf{34.86_{\pm 0.97}}$	$32.55_{\pm 1.51}$	$32.10_{\pm 3.10}$
GCN	$83.44_{\pm 1.44}$	$72.45_{\pm 0.80}$	$86.48_{\pm 0.17}$	$80.26_{\pm 0.34}$	$92.21_{\pm 1.36}$	$88.24_{\pm 0.63}$	$91.85_{\pm 0.29}$	$95.18_{\pm 0.17}$	$71.80_{\pm 0.10}$	$30.16_{\pm 0.73}$	$41.67_{\pm 2.42}$	$40.19_{\pm 4.29}$
DropEdge	$83.27_{\pm 1.55}$	$72.29_{\pm 0.60}$	$86.47_{\pm 0.21}$	$80.22_{\pm 0.55}$	$92.14_{\pm 1.42}$	$88.08_{\pm 1.08}$	$91.91_{\pm 0.16}$	$95.13_{\pm 0.16}$	$71.73_{\pm 0.21}$	$29.86_{\pm 0.82}$	$38.40_{\pm 2.57}$	$\underline{40.51_{\pm 3.38}}$
DropNode	$\underline{83.65_{\pm 1.83}}$	$72.20_{\pm 0.67}$	$86.55_{\pm 0.18}$	$80.11_{\pm 0.61}$	$91.89_{\pm 1.21}$	$88.17_{\pm 0.40}$	$91.93_{\pm 0.28}$	$95.11_{\pm 0.16}$	$71.72_{\pm 0.16}$	$29.07_{\pm 0.93}$	$38.01_{\pm 2.00}$	$39.74_{\pm 2.79}$
DropMessage	$83.45_{\pm 1.56}$	$72.44_{\pm 0.76}$	$\underline{86.56_{\pm 0.16}}$	$80.30_{\pm 0.37}$	$92.13_{\pm 1.56}$	$\underline{88.52_{\pm 0.44}}$	$92.08_{\pm 0.21}$	$95.14_{\pm 0.18}$	$71.93_{\pm 0.20}$	$29.62_{\pm 1.05}$	$38.75_{\pm 3.34}$	$40.48_{\pm 3.07}$
TUNEUP	$83.59_{\pm 1.26}$	$\underline{73.00_{\pm 0.78}}$	$86.43_{\pm 0.36}$	$80.56_{\pm 0.47}$	$92.11_{\pm 1.37}$	$88.14_{\pm 0.95}$	$90.89_{\pm 0.45}$	$94.51_{\pm 0.25}$	$71.81_{\pm 0.15}$	$28.95_{\pm 1.48}$	$41.49_{\pm 2.65}$	$40.24_{\pm 4.24}$
GraphPatcher	$83.57_{\pm 1.38}$	$72.22_{\pm 0.73}$	$86.21_{\pm 0.23}$	$\underline{80.64_{\pm 0.51}}$	$\mathbf{92.89_{\pm 0.57}}$	$88.49_{\pm 0.71}$	$91.74_{\pm 0.25}$	$\underline{95.25_{\pm 0.24}}$	$\underline{72.06_{\pm 0.06}}$	$28.07_{\pm 0.67}$	$\underline{41.89_{\pm 2.49}}$	$40.35_{\pm 4.11}$
GCN_B(Ours)	$\mathbf{84.84_{\pm 1.39}}$	$\mathbf{73.32_{\pm 0.85}}$	$\mathbf{87.56_{\pm 0.27}}$	$\mathbf{80.75_{\pm 0.42}}$	$\underline{92.44_{\pm 1.42}}$	$\mathbf{88.76_{\pm 0.65}}$	$\underline{93.54_{\pm 0.37}}$	$\mathbf{95.79_{\pm 0.17}}$	$\mathbf{72.43_{\pm 0.16}}$	$\underline{30.56_{\pm 0.84}}$	$\mathbf{42.39_{\pm 2.19}}$	$\mathbf{40.96_{\pm 4.83}}$
ACCURACY ON HEAD NODES (HIGH-DEGREE)												
MLP	$65.86_{\pm 1.56}$	$70.99_{\pm 1.33}$	$84.70_{\pm 0.32}$	$80.06_{\pm 0.83}$	$88.58_{\pm 1.12}$	$86.09_{\pm 0.68}$	$\mathbf{94.08_{\pm 0.24}}$	$97.50_{\pm 0.14}$	$63.93_{\pm 0.17}$	$\mathbf{34.27_{\pm 1.42}}$	$25.80_{\pm 3.72}$	$29.74_{\pm 3.68}$
GCN	$84.70_{\pm 1.60}$	$79.10_{\pm 0.97}$	$87.81_{\pm 0.36}$	$85.13_{\pm 0.56}$	$\underline{94.85_{\pm 2.01}}$	$90.72_{\pm 0.75}$	$93.15_{\pm 0.26}$	$\underline{97.64_{\pm 0.12}}$	$80.81_{\pm 0.10}$	$27.63_{\pm 1.39}$	$35.12_{\pm 3.80}$	$36.51_{\pm 6.92}$
DropEdge	$84.74_{\pm 2.01}$	$78.92_{\pm 0.78}$	$87.77_{\pm 0.38}$	$84.99_{\pm 0.30}$	$94.50_{\pm 1.75}$	$90.10_{\pm 1.27}$	$93.14_{\pm 0.13}$	$97.61_{\pm 0.11}$	$80.67_{\pm 0.26}$	$27.51_{\pm 2.35}$	$33.64_{\pm 4.98}$	$37.58_{\pm 6.54}$
DropNode	$84.82_{\pm 2.47}$	$79.01_{\pm 1.34}$	$87.80_{\pm 0.34}$	$85.02_{\pm 0.55}$	$91.89_{\pm 1.21}$	$90.53_{\pm 0.58}$	$93.19_{\pm 0.23}$	$97.59_{\pm 0.11}$	$80.73_{\pm 0.25}$	$26.62_{\pm 1.15}$	$32.33_{\pm 5.09}$	$35.92_{\pm 6.81}$
DropMessage	$84.86_{\pm 1.60}$	$79.33_{\pm 1.10}$	$\underline{87.84_{\pm 0.45}}$	$84.96_{\pm 0.42}$	$94.62_{\pm 2.24}$	$\underline{91.01_{\pm 0.75}}$	$93.28_{\pm 0.29}$	$97.57_{\pm 0.11}$	$80.77_{\pm 0.25}$	$27.55_{\pm 1.70}$	$30.42_{\pm 4.14}$	$38.85_{\pm 7.47}$
TUNEUP	$84.58_{\pm 1.46}$	$79.43_{\pm 0.83}$	$87.78_{\pm 0.54}$	$85.35_{\pm 0.51}$	$94.73_{\pm 1.95}$	$90.62_{\pm 1.12}$	$92.12_{\pm 0.40}$	$97.26_{\pm 0.15}$	$80.74_{\pm 0.18}$	$26.56_{\pm 1.43}$	$34.85_{\pm 3.81}$	$35.82_{\pm 5.38}$

1) Overall Performance

- Aggregation Buffer is the ***only method that consistently and significantly*** improves GCN performance across all datasets—achieving the best accuracy on 9 and second-best on 3 datasets.
- While random dropping methods fail to reliably outperform base GCN, our method—despite also using DropEdge—succeeds, reinforcing our claim that the limitation is due to the GNNs' inductive bias.

2) Addressing Degree Bias

- Aggregation Buffer effectively reduces degree bias, achieving substantial gains on low-degree nodes—ranking at least second-best on tail nodes in 10 datasets and head nodes in 9.
- Degree-bias methods often struggle in heterophilous graphs due to its reversed bias trends. Our method still improves in these cases, highlighting ***edge-robustness is a broader and more reliable approach.***

DropNode	83.65±1.83	72.20±0.67	86.55±0.18	80.11±0.61	91.89±1.21	88.17±0.40	91.93±0.28	95.11±0.16	71.72±0.16	29.07±0.93	38.01±2.00	39.74±2.79
DropMessage	83.45±1.56	72.44±0.76	86.56±0.16	80.30±0.37	92.13±1.56	88.52±0.44	92.08±0.21	95.14±0.18	71.93±0.20	29.62±1.05	38.75±3.34	40.48±3.07
TUNEUP	83.59±1.26	73.00±0.78	86.43±0.36	80.56±0.47	92.11±1.37	88.14±0.95	90.89±0.45	94.51±0.25	71.81±0.15	28.95±1.48	41.49±2.65	40.24±4.24
GraphPatcher	83.57±1.38	72.22±0.73	86.21±0.23	80.64±0.51	**92.89±0.57**	88.49±0.71	91.74±0.25	95.25±0.24	72.06±0.06	28.07±0.67	41.89±2.49	40.35±4.11
GCN$_B$(Ours)	**84.84±1.39**	**73.32±0.85**	**87.56±0.27**	**80.75±0.42**	92.44±1.42	**88.76±0.65**	93.54±0.37	**95.79±0.17**	**72.43±0.16**	30.56±0.84	**42.39±2.19**	**40.96±4.83**
ACCURACY ON HEAD NODES (HIGH-DEGREE)												
MLP	65.86±1.56	70.99±1.33	84.70±0.32	80.06±0.83	88.58±1.12	86.09±0.68	**94.08±0.24**	97.50±0.14	63.93±0.17	**34.27±1.42**	25.80±3.72	29.74±3.68
GCN	84.70±1.60	79.10±0.97	87.81±0.36	85.13±0.56	94.85±2.01	90.72±0.75	93.15±0.26	97.64±0.12	80.81±0.10	27.63±1.39	35.12±3.80	36.51±6.92
DropEdge	84.74±2.01	78.92±0.78	87.77±0.38	84.99±0.30	94.50±1.75	90.10±1.27	93.14±0.13	97.61±0.11	80.67±0.26	27.51±2.35	33.64±4.98	37.58±6.54
DropNode	84.82±2.47	79.01±1.34	87.80±0.34	85.02±0.55	91.89±1.21	90.53±0.58	93.19±0.23	97.59±0.11	80.73±0.25	26.62±1.15	32.33±5.09	35.92±6.81
DropMessage	84.86±1.60	79.33±1.10	87.84±0.45	84.96±0.42	94.62±2.24	91.01±0.75	93.28±0.29	97.57±0.11	80.77±0.25	27.55±1.70	30.42±4.14	**38.85±7.47**
TUNEUP	84.58±1.46	**79.43±0.83**	87.78±0.54	**85.35±0.51**	94.73±1.95	90.62±1.12	92.12±0.40	97.26±0.15	80.74±0.18	26.56±1.43	34.85±3.81	35.82±5.38
GraphPatcher	85.21±1.56	79.00±0.66	87.66±0.47	85.22±0.65	**95.28±0.61**	**91.51±0.69**	93.25±0.42	97.46±0.20	**80.89±0.06**	26.85±1.38	**35.72±4.41**	36.40±4.99
GCN$_B$(Ours)	**85.82±1.31**	79.41±0.99	**88.14±0.60**	85.04±0.56	94.84±2.05	90.70±0.80	93.87±0.26	**97.70±0.11**	80.85±0.13	27.65±1.48	35.38±4.28	37.68±7.39
ACCURACY ON TAIL NODES (LOW-DEGREE)												
MLP	63.20±1.36	60.27±1.42	84.30±0.43	73.02±1.02	81.91±0.90	75.51±0.73	92.96±0.28	92.76±0.21	49.71±0.19	**34.47±1.34**	35.59±3.33	28.94±5.09
GCN	79.79±1.75	65.77±1.49	85.14±0.25	77.83±0.58	87.98±0.88	83.35±0.92	90.04±0.53	92.74±0.33	62.76±0.21	32.33±2.79	45.85±4.69	37.17±6.51
DropEdge	79.61±1.56	65.54±1.32	85.21±0.34	77.99±0.55	88.13±1.01	83.65±1.13	90.09±0.32	92.66±0.36	62.65±0.33	31.94±1.91	43.20±3.17	34.91±5.93
DropNode	80.19±1.63	65.50±1.28	85.33±0.24	77.62±0.67	87.69±1.01	83.23±0.54	90.12±0.54	92.67±0.34	62.69±0.17	30.77±1.51	42.76±2.09	34.33±5.88
DropMessage	79.71±1.86	65.75±1.42	85.31±0.30	77.90±0.56	88.07±1.03	83.61±0.52	90.35±0.32	92.72±0.38	63.20±0.18	30.73±2.05	44.44±6.24	34.66±6.55
TUNEUP	80.40±1.77	66.35±1.66	85.12±0.28	78.13±0.80	87.87±0.97	83.45±0.86	88.98±0.59	91.64±0.32	62.89±0.19	31.09±3.29	45.51±4.66	37.50±6.91
GraphPatcher	81.13±1.91	65.39±1.17	84.98±0.24	78.88±0.99	**89.28±0.66**	83.24±1.02	89.48±0.49	93.03±0.39	63.56±0.13	29.22±1.71	46.24±3.85	**38.29±6.88**
GCN$_B$(Ours)	**82.05±1.75**	**67.17±1.37**	**86.85±0.22**	**79.25±0.58**	88.53±1.09	**84.61±0.98**	**92.97±0.69**	**94.07±0.27**	**64.40±0.20**	32.25±1.99	**47.06±4.13**	37.35±6.99

3) Addressing Structural Disparity

- Consistent with recent findings (Mao et al., 2024), our experiments show that MLPs generally outperform GNNs on heterophilous nodes, while GNNs perform better on homophilous nodes.
- Aggregation buffer achieves the ***highest GNN performance on heterophilous nodes in 9 datasets.***

Method	Cora	Citeseer	PubMed	Wiki-CS	Photo	Computer	CS	Physics	Arxiv	Actor	Squirrel	Chameleon
ACCURACY ON HOMOPHILOUS NODES												
MLP	71.68±1.77	76.37±1.19	89.90±0.58	86.30±0.58	83.63±1.41	86.01±0.59	96.96±0.25	98.02±0.07	74.69±0.14	36.88±1.52	35.84±2.73	33.50±5.96
GCN	92.69±1.53	87.96±1.25	95.99±0.22	94.05±0.65	96.45±3.76	94.47±0.53	99.25±0.15	99.32±0.15	95.43±0.08	**39.47±1.62**	48.71±3.57	47.15±5.79
DropEdge	92.38±1.88	88.06±0.90	96.12±0.36	94.44±0.42	96.35±3.84	94.72±0.43	99.28±0.13	99.32±0.12	95.68±0.15	38.30±1.08	41.25±4.34	42.39±4.22
DropNode	92.81±1.63	87.84±0.97	96.17±0.32	93.99±0.60	96.48±3.71	94.29±0.30	99.31±0.17	99.29±0.13	95.62±0.13	38.00±0.79	40.73±5.13	42.67±5.93
DropMessage	92.51±1.67	88.06±1.02	**96.18±0.23**	94.49±0.34	96.35±3.67	94.62±0.47	**99.37±0.15**	99.32±0.13	**95.79±0.06**	38.02±1.64	45.22±4.60	41.25±4.98
TUNEUP	93.17±1.60	88.35±1.12	96.04±0.30	94.05±0.65	96.30±3.76	94.57±0.55	99.14±0.14	99.26±0.13	95.67±0.11	37.94±2.57	48.58±3.79	**47.89±5.89**
GraphPatcher	93.23±1.24	87.38±1.09	96.04±0.28	94.08±0.53	**98.18±0.19**	94.65±0.64	98.33±0.30	**99.44±0.07**	95.72±0.09	34.76±1.18	48.71±2.89	44.83±5.22
GCN$_B$(Ours)	**94.13±1.39**	**88.78±1.52**	95.83±0.28	**94.65±0.57**	96.54±3.76	**95.30±0.49**	98.64±0.56	99.33±0.17	95.66±0.13	38.26±1.90	**49.52±3.40**	47.01±5.60
ACCURACY ON HETEROPHILOUS NODES												
MLP	50.93±0.98	**44.88±1.82**	**73.22±0.71**	**57.90±0.93**	81.71±1.34	66.84±0.69	**85.96±0.29**	**89.08±0.37**	**34.53±0.18**	**31.66±2.79**	32.56±4.13	29.53±4.83
GCN	64.18±2.49	41.96±1.24	67.34±0.47	51.89±1.08	81.74±0.75	71.42±1.25	76.81±0.68	86.60±0.37	32.51±0.28	19.13±1.55	42.19±5.54	33.74±7.61
DropEdge	64.09±2.68	41.78±1.27	67.12±0.52	50.97±1.49	81.50±0.69	71.06±1.95	76.92±0.35	86.47±0.40	31.70±0.52	19.29±1.72	41.59±6.04	37.01±5.02
DropNode	64.60±3.58	41.59±1.08	67.24±0.51	51.66±1.21	80.67±0.97	71.38±1.21	76.93±0.65	86.46±0.39	31.91±0.57	18.93±1.02	41.54±4.52	36.78±5.27
DropMessage	64.39±2.77	41.84±0.84	67.23±0.39	51.48±0.98	81.65±0.82	71.87±0.98	77.27±0.51	86.47±0.44	32.29±0.46	19.49±1.18	40.79±4.68	**37.90±7.63**
TUNEUP	63.59±2.36	42.74±1.07	67.09±0.89	52.50±0.72	81.58±0.87	71.03±2.17	74.16±1.11	84.67±0.65	31.68±0.23	18.24±0.92	42.13±5.24	33.00±6.69
GraphPatcher	64.17±2.22	44.47±0.89	66.41±0.34	53.03±0.86	81.46±1.68	71.56±1.96	78.67±0.53	86.87±0.64	33.38±0.14	18.49±1.33	42.41±5.16	34.45±7.90
GCN$_B$(Ours)	**65.54±2.34**	43.24±1.05	70.77±0.71	52.44±1.27	**82.29±1.05**	**72.02±1.25**	82.75±0.63	88.43±0.35	34.02±0.34	19.96±1.49	**42.42±4.97**	35.03±7.53

Aggregation buffer effectively optimizes edge-robustness improved structural generalization.

Generalization to other GNN architectures

Our method consistently delivers significant performance ***improvements across all four widely-used GNN architectures***, demonstrating its broad applicability and effectiveness.

Table 3. Accuracy of different GNN models before and after the integration with AGG_B. AGG_B achieves consistent and significant performance improvements across various architectures.

	Pubmed	CS	Arxiv	Chameleon
SAGE	$87.07_{\pm 0.24}$	$92.44_{\pm 0.60}$	$70.92_{\pm 0.16}$	$37.34_{\pm 3.56}$
SAGE_B	$\mathbf{88.09}_{\pm \mathbf{0.28}}$	$\mathbf{93.36}_{\pm \mathbf{0.47}}$	$\mathbf{71.16}_{\pm \mathbf{0.14}}$	$\mathbf{37.85}_{\pm \mathbf{3.80}}$
GAT	$85.64_{\pm 0.24}$	$90.50_{\pm 0.28}$	$71.86_{\pm 0.14}$	$38.54_{\pm 2.70}$
GAT_B	$\mathbf{87.47}_{\pm \mathbf{0.37}}$	$\mathbf{93.09}_{\pm \mathbf{0.60}}$	$\mathbf{72.26}_{\pm \mathbf{0.14}}$	$\mathbf{39.08}_{\pm \mathbf{2.84}}$
SGC	$84.01_{\pm 0.76}$	$90.89_{\pm 0.45}$	$69.15_{\pm 0.05}$	$38.24_{\pm 3.00}$
SGC_B	$\mathbf{84.77}_{\pm \mathbf{1.02}}$	$\mathbf{91.90}_{\pm \mathbf{0.43}}$	$\mathbf{69.55}_{\pm \mathbf{0.04}}$	$\mathbf{38.91}_{\pm \mathbf{3.08}}$
GIN	$85.42_{\pm 0.20}$	$87.88_{\pm 0.51}$	$63.94_{\pm 0.53}$	$39.84_{\pm 2.69}$
GIN_B	$\mathbf{87.18}_{\pm \mathbf{0.17}}$	$\mathbf{88.58}_{\pm \mathbf{1.00}}$	$\mathbf{65.66}_{\pm \mathbf{0.75}}$	$\mathbf{41.72}_{\pm \mathbf{2.41}}$

Aggregation Buffer

$$\boldsymbol{H}_{\mathcal{N}}^{(l)} = \underbrace{\mathrm{AGG}^{(l)}(\boldsymbol{H}^{(l-1)}, \boldsymbol{A})}_{\textbf{Aggregation of any GNNs}} + \mathrm{AGG}_B^{(l)}(\boldsymbol{H}^{(0:l-1)}, \boldsymbol{A})$$

Aggregation buffer is compatible with most GNN architectures due to its modular design.

Conclusion

1. We identify a ***key limitation of DropEdge***: it fails to fully optimize robustness during training.
2. Theoretical analysis reveals this stems from discrepancies that arise during GNN aggregation.
3. To address this, we propose ***Aggregation Buffer*** (AGG_B)—a post-hoc parameter block that refines the aggregation output at each GNN layer to improve edge-robustness.
4. We evaluated on 12 node classification benchmarks and 5 different architectures. It showed significant and ***consistent gains, especially under structural inconsistencies*** such as degree bias and structural disparity.

Future Work

Enabling end-to-end AGG_B training with a GNN backbone for joint optimization of bias and robustness.

Full Paper

Project Page