From Multi-Class to Multi-Label: Revisiting Edge Dropping for Graph Neural Networks

Alexandru Andrei | Supervisors: Megha Khosla, Elena Congeduti | Responsible Professor: Megha Khosla
EEMCS, Delft University of Technology | CSE3000 Research Project | A.Andrei-3@student.tudelft.nl

1. Background: the multi-label gap

Graph neural networks (GNNs) label a node from its neighbours by repeatedly *averaging* features along edges (*message passing*) [1]. Stacking many layers makes all nodes look alike and accuracy falls (**over-smoothing**) [2,3]. The standard remedy is **edge dropping**: train on a random subset of edges each step (§3).

The gap. Edge dropping was validated *only* on **single-label** graphs, where an edge is cleanly good (same label) or bad (different). Real graphs are often **multi-label**: the same edge agrees on some labels and disagrees on others [6], and one shared embedding makes dropping it perturb *all* labels at once. Multi-label GNNs also tend to *underfit* [7], so a regulariser may remove signal the model still needs.

Label homophily h measures how often linked nodes share labels, averaged over all edges [6]:

$$h = \frac{1}{|E|} \sum_{(i,j)\in E} \frac{|Y_i \cap Y_j|}{|Y_i \cup Y_j|}.$$

It ranges from 0, when linked nodes share no labels, to 1, when they share all of them. **Single-label benchmarks all sit above** 0.7, so the low and mid range was never tested.

2. Research questions

How does the edge-dropping strategy interact with label homophily in multi-label node classification?

- **RQ1.** Does edge dropping help, and how does it scale with **drop rate** and **depth**?
- **RQ2.** How does **homophily** change it, and is DropEdge or TADropEdge preferable?
- **RQ3.** Do the synthetic trends **transfer to real** graphs?

3. Two edge-dropping strategies

DropEdge [4]: remove each edge with probability r, uniformly.

TADropEdge [5]: keep edges that bridge clusters, drop redundant ones. Each edge (i, j) gets a structural weight from the q lowest eigenvectors $\mathbf{V}$ of the graph Laplacian:

$$\omega_{ij} = \Sigma_{k<q}(\mathbf{V}_{ik} - \mathbf{V}_{jk})^2.$$

A high ω marks a bridge edge to keep, a low ω a redundant edge to drop. Three rules turn ω into a keep-probability. **Cutoff** thresholds at the median γ, **division** uses $1 - r\,\gamma/(\gamma + \omega)$, and **cdf** uses $(1 - r) + r\,F(\omega)$.

4. Experimental design

Synthetic control. Graphs from MLGNC [6]: labels + features via the hypersphere method [8], edges via Social Distance Attachment [9]. A decay α is binary-searched so each graph hits a *target* homophily, everything else fixed. We sweep one factor at a time:

- homophily $h \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$, labels $L \in \{4, 8, 16\}$
- depth $D \in \{2, 4, 6, 8\}$, drop rate $r \in \{0.1, 0.3, 0.5, 0.7\}$

× DropEdge + 3 TADropEdge rules × 10 seeds (each redraws the graph) gives **10 200 runs**; seeds vary the *graph*, so CIs support trend-level claims.

Real benchmarks spanning the h range:

Dataset	h	Nodes	Labels
DBLP	0.76 (high)	28 702	4
HumanGo	0.40 (med)	3 106	14
PCG	0.17 (low)	3 233	15

Table 1. Three real datasets.

Protocol. GCN, hidden 256, sigmoid + BCE; depth D swept; Adam, early stop on val F1; drop edges in *training only*, evaluate on the full graph. Score: **top-k F1-macro** with k the true label count [6]. We report the paired difference $\Delta\text{F1} = \text{F1}_{\text{method}} - \text{F1}_{\text{baseline}}$ per graph, where a negative value means the method trails the baseline.

Main finding

Across 10 200 synthetic and 3 real graphs, edge dropping shows *no reliable gain* over keeping all edges, and the average loss deepens the more you drop. The cause is underfitting (too little per-label signal), not the over-smoothing it was designed to fix.

5. Baseline regime: two problems

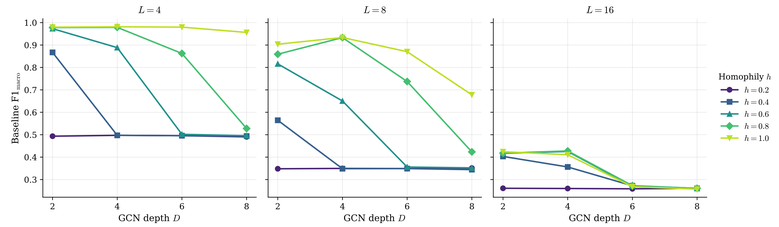

Figure 1. Baseline F1 vs. depth, per label count L and homophily h (10 seeds). Performance decays with depth in every cell except L=4, h=1.0; the depth at which it decays grows with homophily.

(i) Over-smoothing appears across almost the whole grid: F1 peaks at shallow depth and falls as it deepens, sooner at higher L, later at higher h. **(ii) Underfitting:** even the easiest graph (L=16, h=1.0) peaks at only F1 ≈ **0.42**, because the L heads share one embedding and each label sees about $1/L$ of the signal. The grid thus splits in two: where the baseline still holds real signal a regulariser could damage, and where it already sits at the random floor.

6. No gain at any drop rate or depth (RQ1)

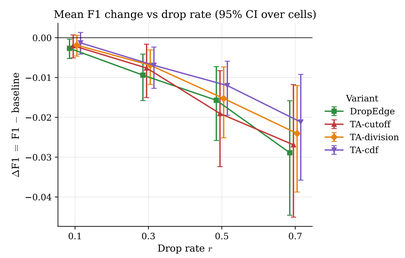

Figure 2. ΔF1 vs. rate, pooled (95% CI). Every variant below zero; harm grows monotonically with r.

D	DE	TA
2	−0.012	−0.009
4	−0.019	−0.021
6	−0.015	−0.009
8	−0.011	−0.009

Table 2. ΔF1 by depth.

All four curves stay below zero and deepen with the rate (DropEdge −0.003 at r=0.1 to −0.029 at r=0.7). The harm is *flat across depth*, so the "deeper GCNs benefit more" claim for single-label DropEdge [4] does not appear: where the baseline has collapsed, dropping edges only reaches the same floor.

7. Homophily sets which method is milder (RQ2)

h	DE	TA	TA−DE
0.2	0.000	0.000	0.000
0.4	−0.012	−0.014	−0.002
0.6	−0.011	−0.022	**−0.011**
0.8	−0.025	−0.019	**+0.006**
1.0	−0.023	−0.006	**+0.017**

Table 3. ΔF1 by h.

Both methods hurt above the floor, but the **TA−DE gap flips sign** with homophily. TADropEdge keeps the structurally important edges:

- **High h:** those edges also carry labels, so TA is milder.
- **Mid h:** structure and labels disagree, so TA is worse.

Per-cell gaps are smaller than the seed-to-seed variation, so we read the *steady trend* across h rather than any single cell.

8. The pattern holds on real graphs (RQ3)

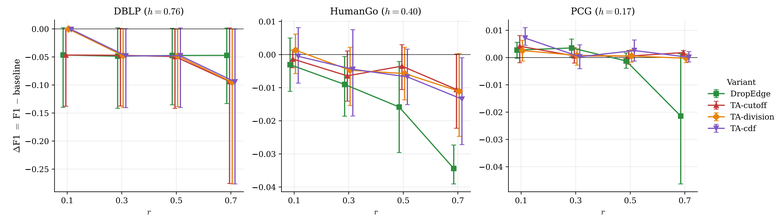

Figure 3. ΔF1 vs. drop rate on real data, pooled over depth (note DBLP's scale and wide CIs). Harm is largest for high-h DBLP and near zero for low-h PCG, the same ordering as the synthetic grid.

DBLP (−0.047 mean): most harmed, driven by the deeper GCNs; the drop to −0.094 at r=0.7 is a single unstable D=6 run, not a trend. **HumanGo**: TADropEdge gentler (−0.006 vs −0.016). **PCG** (+0.002, within noise): the one non-negative case, a genuinely multi-label, low-h graph where TADropEdge's premise should hold best.

9. A confound: the two methods drop unequally

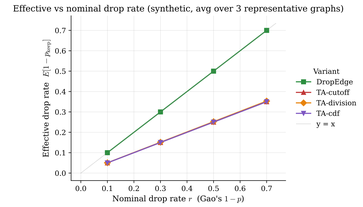

Figure 4. Realised vs. nominal rate.

At the same nominal r, TADropEdge removes only ≈ $r/2$ (it protects bridge edges), so part of its mildness is just dropping less. But this cannot explain the sign flip: if it could, TA would always beat DropEdge, yet at h=0.6 it loses by 0.011. The flip is about *which* edges are kept, not how many.

The three TADropEdge rules nearly coincide (mean ΔF1 of −0.014, −0.012, and −0.011 for cutoff, division, and cdf), so we pool over them.

10. Conclusions & future work

- Edge dropping does not help multi-label GNNs here; the binding constraint looks like **underfitting**, not over-smoothing.
- Homophily predicts *how much* is lost and *which* method is milder.
- **Next:** per-label capacity / frequency-weighted loss; other backbones (GraphSAGE [10], GAT [11]).

Limitations. One GCN backbone, three real datasets with three seeds each (wide CIs), and PCG confounds low homophily with a high label count.

References

[1] Kipf & Welling. GCNs. *ICLR* 2017. [2] Li et al. Deeper Insights into GCNs. *AAAI* 2018. [3] Oono & Suzuki. GNNs Lose Expressive Power. *ICLR* 2020. [4] Rong et al. DropEdge. *ICLR* 2020. [5] Gao et al. TADropEdge. *arXiv:2106.02892* 2021. [6] Zhao et al. Multi-label Node Classification. *TMLR* 2023. [7] Zhao & Khosla. GNN-MultiFix. *arXiv:2411.14094* 2024. [8] Tomas et al. Hypersphere ML Data. *HAIS* 2014. [9] Boguñá et al. Social Distance Attachment. *PRE* 2004. [10] Hamilton et al. GraphSAGE. *NeurIPS* 2017. [11] Veličković et al. GAT. *ICLR* 2018.