CISPA
HELMHOLTZ CENTER FOR INFORMATION SECURITY

ICML
International Conference On Machine Learning

Fixed Aggregation Features Can Rival GNNs

Celia Rubio-Madrigal, Rebekka Burkholz

June 10, 2026 — Graph Signal Processing Workshop

Machine learning → Deep learning

- Train on training data $(X_{\text{train}}, y_{\text{train}})$, generalize on unseen test data $(X_{\text{test}} \longrightarrow \hat{y}_{\text{test}}?)$
- With lots of data + parameters, we reach **beyond the statistical regime**.

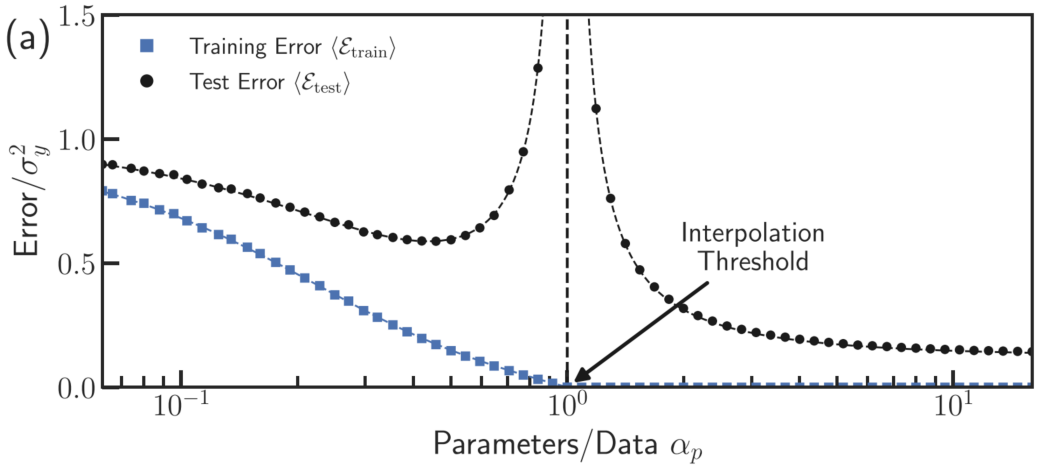


Figure from *"Memorizing without overfitting: Bias, variance, and interpolation in over-parameterized models"*

Questions for today

Does graph ML reap the benefits of deep learning?

Simplicity

When can we win with simple models?

Efficiency

Are we adding complexity unnecessarily?

Benchmarking

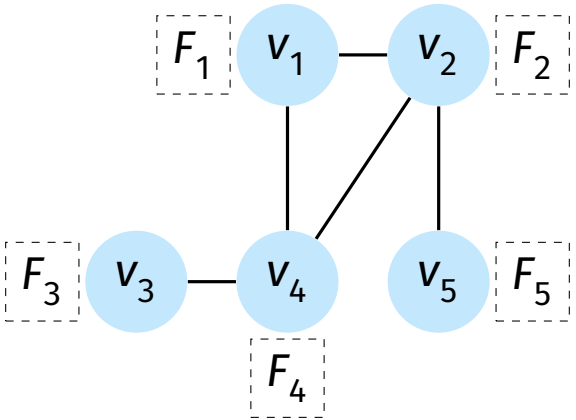
How well are we measuring progress?

Introduction: Graph Machine Learning

Graph data

Attributed graph

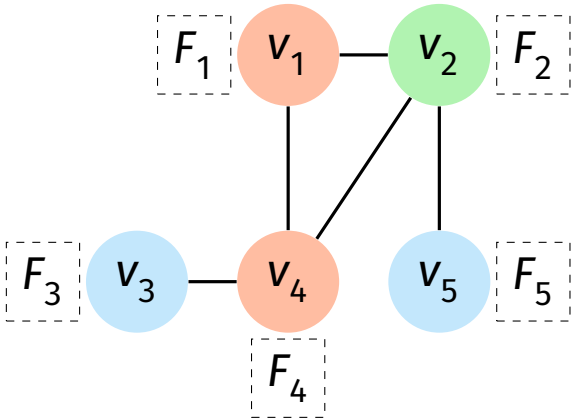
- A graph $G = (V, E)$ has nodes V and edges E
- Each node $v \in V$ carries additional information F_v



Task

Task types

- Graph: Global label Y_G
- Edge: Label y_e per $e \in E$
- Node: Label y_v per $v \in V$ (●●●)



Graph data: table + connections

Node features F

	GDP	#
v_1	x_{11}	x_{12}
v_2	x_{21}	x_{22}
v_3	x_{31}	x_{32}
v_4	x_{41}	x_{42}
v_5	x_{51}	x_{52}

one row per node

Adjacency A

$$\begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

who is connected to whom

Graph data: table + connections

Node features F

	GDP	#
v_1	x_{11}	x_{12}
v_2	x_{21}	x_{22}
v_3	x_{31}	x_{32}
v_4	x_{41}	x_{42}
v_5	x_{51}	x_{52}

one row per node

Adjacency A

$$\begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

who is connected to whom

Graph Machine Learning: ↑ **How to learn from this data?** ↑

Model: GNN

Graph Neural Network (GNN)

Learns representations by *iteratively* aggregating information from neighbors $N(\cdot)$

$$h_i^{(\ell+1)} = \text{UPDATE}^{(\ell)}\left(h_i^{(\ell)}, \text{AGGREGATE}^{(\ell)}\left(\left\{h_j^{(\ell)} : j \in N(i)\right\}\right)\right)$$

L-hop neighborhood

$h_i^{(L)}$ contains information from the L-hop neighborhood of node i.

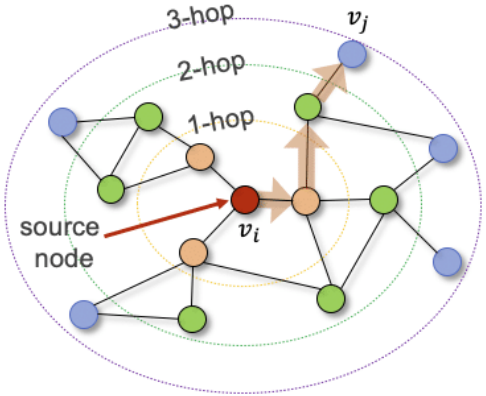


Figure from *"Understanding graph embedding methods and their applications"*

Model: Message passing

- Aggregation of neighbor features (A) → mean, sum, attention, etc.
- Combination via learnable parameters (C) → linear layer + ReLU

4-layer GCN: AC-AC-AC-AC

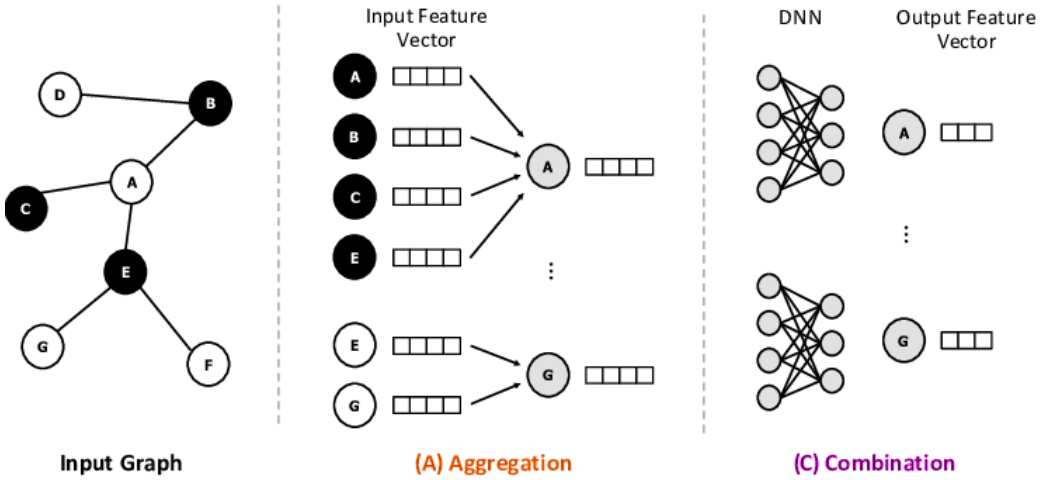


Figure from *"GROW: A Row-Stationary Sparse-Dense GEMM Accelerator for Memory-Efficient Graph Convolutional Neural Networks"*

Graph tabularization

Decoupling deep learning from the graph

Can we **not learn** the graph part of the process?

$\Longrightarrow$ It is theoretically viable, and

$\Longrightarrow$ It is well-performing for **the vast majority** of datasets

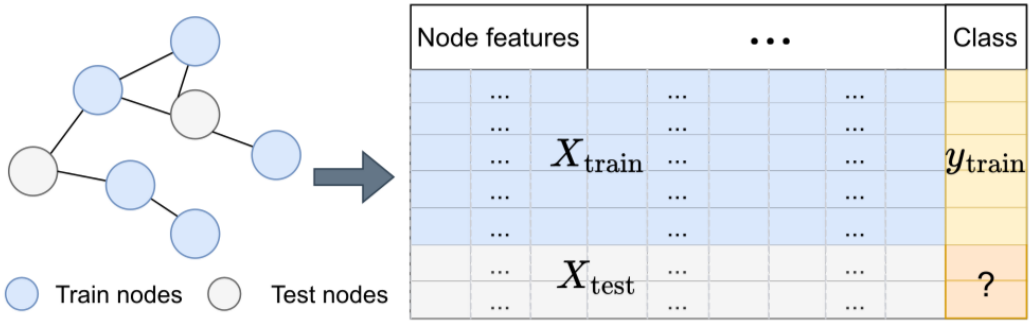


Figure from *"Can TabPFN Compete with GNNs for Node Classification via Graph Tabularization?"*

Related work: simpler GNNs

GNNs can empirically compete with Graph Transformers

- Benchmark GCN, GAT, SAGE.
- Proper **hyperparameter tuning** & optimization techniques.

Proved to compete or outperform more complex graph architectures, like state-of-the-art **graph transformers** and heterophily-aware models.

"Classic GNNs are strong baselines: Reassessing GNNs for node classification." (Luo et al., 2024)

Related work: non-trained GNNs

Efficiency has inspired pre-computing the aggregation

- Aggregate across **multiple hops** without learning.
- Sometimes, a more complex attention mechanism is trained afterwards.
- **Performance loss** is assumed in exchange of scalability.

"Simplifying Graph Convolutional Networks" (Wu et al., 2019)

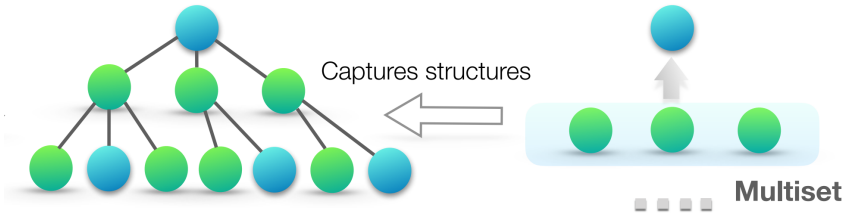
"SIGN: Scalable Inception Graph Neural Networks" (Frasca et al., 2020)
"Graph Attention Multi-Layer Perceptron" (Zhang et al., 2022)
"Less is More: Hop-Wise Graph Attention for Scalable and Generalizable Learning on Circuits" (Deng et al., 2024)

Expressive capacity of GNNs

Related work: expressiveness

GNN expressiveness is over multiset functions

- Characterize GNN power via multiset functions over neighborhoods.
- Derive GIN, an architecture that is maximally expressive over them.



"How Powerful are Graph Neural Networks?" (Xu et al., 2018)

What does it take to preserve information?

Color code: Aggregation Combination Neighborhood function

Kolmogorov-Arnold theorem (Schmidt-Hieber et al., 2021)

For dimension $d \geq 2$, there exists a monotone function $f : [0,1] \to C$ (the Cantor set) such that any function $g : [0,1]^d \to \mathbb{R}$ can be decomposed as

$$g(x_1, \dots, x_d) = \varphi\left(3 \sum_{p=1}^{d} 3^{-p} f(x_p)\right)$$

for some univariate function $\varphi : C \to \mathbb{R}$.

Moreover, if g is continuous, then also φ is continuous.

Aggregation is fixed for every neighborhood function!

- We can aggregate neighborhood information as a preprocessing step.

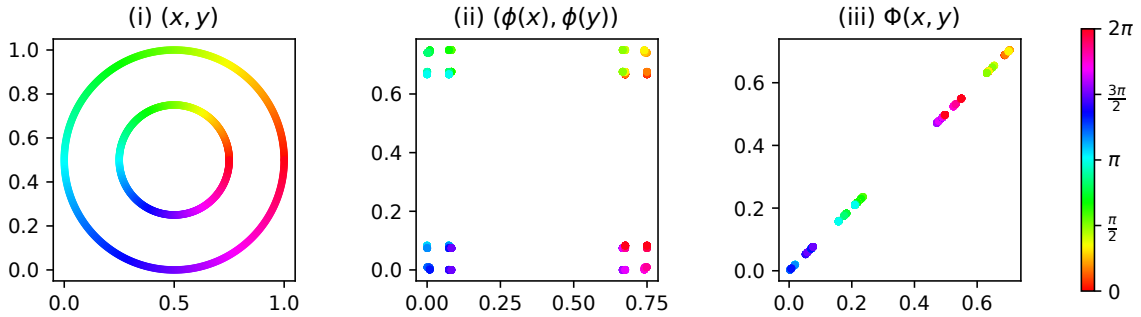
e.g., 4-hop: AAAA-CCCC, as long as A is an **injective** aggregation.

Aggregation is fixed for every neighborhood function!

- We can aggregate neighborhood information as a preprocessing step.

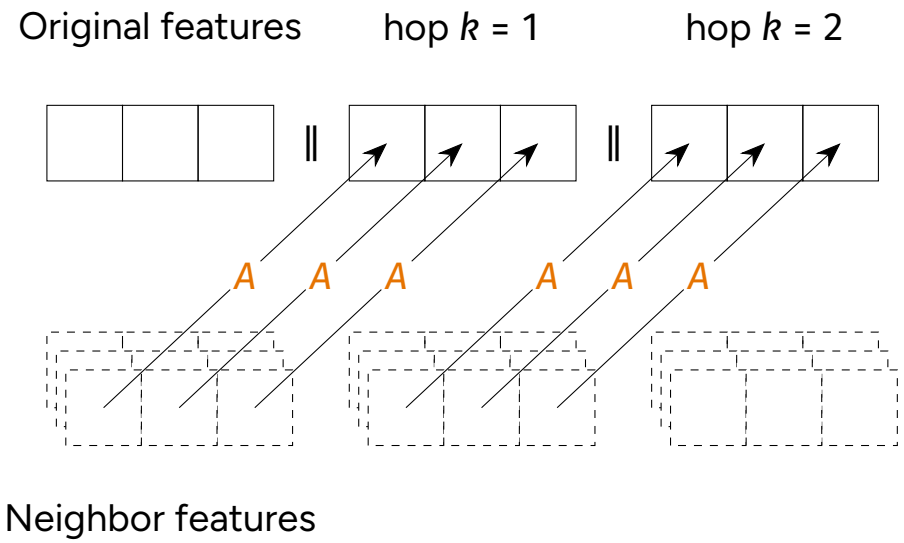
e.g., 4-hop: AAAA-CCCC, as long as A is an **injective** aggregation.

- In practice, the K-A aggregation is not well-behaved.
- We often rely on theorems that do not translate to practice!



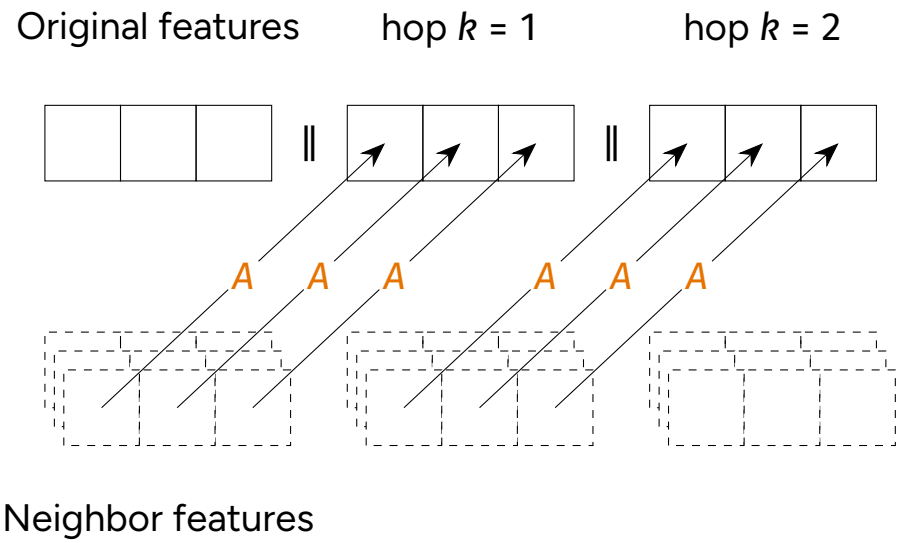
Our approach: Fixed Aggregation Features (FAF)

- Choice of aggregation(s): $A \in$ [mean, sum, max, std] → **concatenate all**
- Choice of tabular model: MLP with input size $F \cdot (1 + 4K)$ with K number of hops



Our approach: Fixed Aggregation Features (FAF)

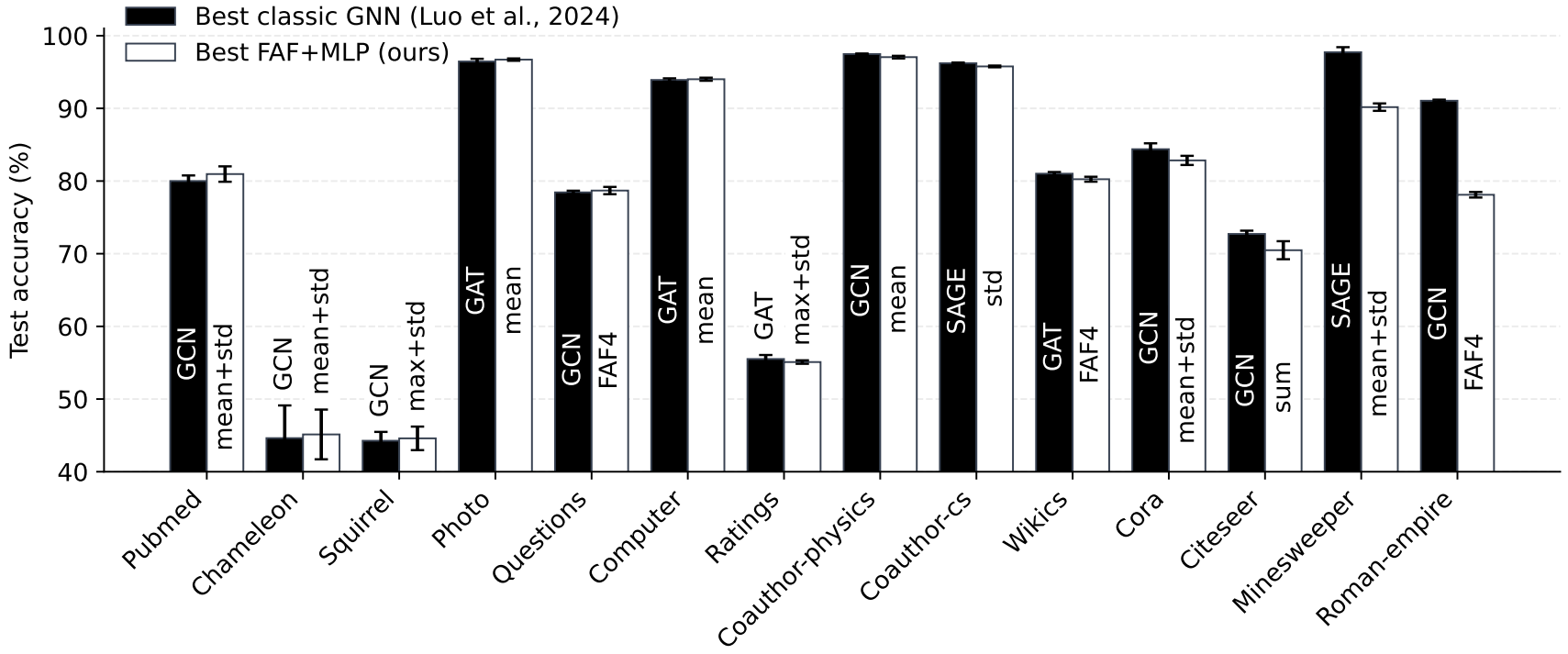
- Choice of aggregation(s): $A \in$ [mean, sum, max, std] → **concatenate all**
- Choice of tabular model: MLP with input size $F \cdot (1 + 4K)$ with K number of hops



Tuning well the MLP predictor is necessary!

Experiments over node classification benchmarks

Results on FAF vs. classic GNNs



Observations

1. If GNNs are necessary, they should clearly beat FAFs.
   $\Longrightarrow$ We are also competitive with complex baselines, including GTs.

Observations

1. If GNNs are necessary, they should clearly beat FAFs.
   $\Longrightarrow$ We are also competitive with complex baselines, including GTs.

2. Two failure points of FAF: Minesweeper and Roman-Empire.
   $\Longrightarrow$ Their GNNs require learnable residual blocks and **long ranges** (>10 hops).

Observations

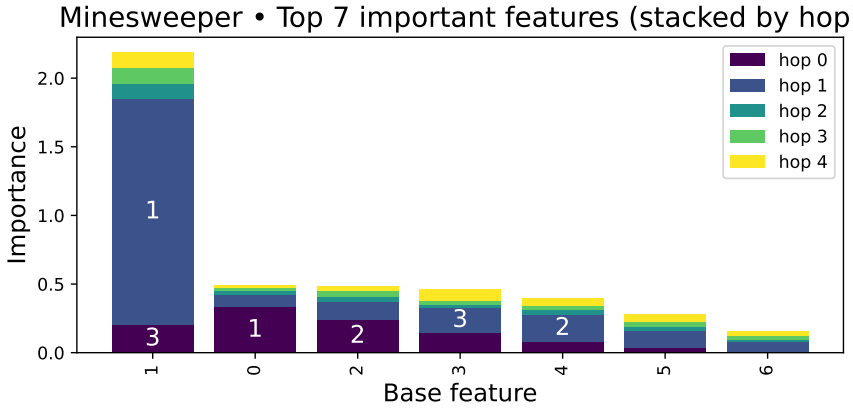
1. If GNNs are necessary, they should clearly beat FAFs.
   $\Longrightarrow$ We are also competitive with complex baselines, including GTs.

2. Two failure points of FAF: Minesweeper and Roman-Empire.
   $\Longrightarrow$ Their GNNs require learnable residual blocks and **long ranges** (>10 hops).

3. Extra benefits of using tabular models.
   $\Longrightarrow$ Interpretability, efficiency, augmenting capabilities, and TFMs.

Interpretability

- We use Shapley Additive exPlanations (SHAP)'s GradientExplainer.
- For Minesweeper we can see the model uses a heuristic.



Efficiency

Pros:

1. No backpropagation through graph convolutions.
2. No repeated aggregation at every forward pass (only 1 time).

Contra: 1-layer width increases → feature selection or compression helps.

Augmentations

We can concatenate anything as features.

- Multiple aggregators and hops.
- Structural statistics: degree, centrality, and other network-science metrics.
- Pre-processing graph rewiring (modified adjacency matrices):

Dataset	computer	photo	chameleon	citeseer	cora	pubmed	wikics
FAF_{mean}	93.16 ± 0.04	**96.06 ± 0.10**	47.99 ± 2.02	66.92 ± 0.87	83.28 ± 0.30	81.16 ± 0.97	81.58 ± 0.46
REW_{mean}	93.25 ± 0.08	95.90 ± 0.04	43.94 ± 2.39	66.92 ± 0.78	82.36 ± 0.17	80.80 ± 0.42	82.44 ± 0.56
SP_{mean}	93.20 ± 0.11	95.97 ± 0.10	43.71 ± 1.71	67.32 ± 0.99	81.80 ± 0.24	80.84 ± 0.62	82.38 ± 0.46
FAF_{mean}+REW_{mean}	**93.33 ± 0.12**	96.03 ± 0.04	**48.26 ± 1.66**	67.48 ± 0.36	83.20 ± 0.20	80.84 ± 0.52	**82.46 ± 0.48**
FAF_{mean}+SP_{mean}	93.31 ± 0.10	**96.06 ± 0.04**	47.99 ± 1.83	**67.88 ± 0.33**	**83.52 ± 0.36**	**81.24 ± 0.43**	82.43 ± 0.51

Graph-Tabular Foundation Models

Converting node classification into tables to leverage TFMs (TabPFN, TabICL, LimiX...)

- Eremeev et al. (2025): node features, structural embeddings, 1-hop aggregation
- Choi et al. (2025): node features, structural embeddings, last-hop aggregation
- Hayler et al. (2025): node features, structural embeddings, labels

We ask: How to perform graph tabularization? How viable is it?

Conclusions

Conclusions from FAFs

Many graph-learning benchmarks may not require learned message passing.

- We give a tangible way of assessing their usefulness.

Conclusions from FAFs

Many graph-learning benchmarks may not require learned message passing.

- We give a tangible way of assessing their usefulness.

FAFs can turn graph ML into tabular ML $\longrightarrow$ use them as baselines.

- Requiring graph-awareness is not enough. Require learned aggregation!

Conclusions from FAFs

Many graph-learning benchmarks may not require learned message passing.

- We give a tangible way of assessing their usefulness.

FAFs can turn graph ML into tabular ML $\longrightarrow$ use them as baselines.

- Requiring graph-awareness is not enough. Require learned aggregation!

Sometimes theoretical complexity does not explain performance boosts.

- It is always productive to try to find out why.

Thank you!

Celia Rubio-Madrigal

Rebekka Burkholz

Fixed Aggregation Features Can Rival GNNs, ICML 2026

https://arxiv.org/abs/2601.19449