Node2Graph: Diagnosing Task Unification in Graph Learning

Neelam Akula, Varun Shiralkar, and Baris Coskunuzer

Dept. Mathematical Sciences, U. Texas at Dallas, Richardson TX 75080, USA
{neelam.akula,varun.shiralkar,coskunuz}@utdallas.edu

Abstract. Graph foundation models aim to unify diverse tasks within a common framework, often by reformulating node and link prediction as subgraph classification. We present a systematic study of this node→subgraph conversion across homophilic and heterophilic benchmarks using both GNN and graph transformer backbones. Our analysis reveals that subgraph classification closely mirrors node classification in homophilic settings but suffers pronounced and growing accuracy losses in heterophilic contexts as the k-hop radius expands. To explain this behavior, we introduce a simple alignment score that quantifies neighborhood label support and reliably predicts the transfer gap. The results expose both the promise and the pitfalls of task unification, offering practical diagnostics and alignment-aware adaptations for more robust and generalizable graph foundation pipelines.

Keywords: Node Classification · Graph Classification · Task Transfer · GNN · Graph Transformers

1 Introduction

Foundation-style graph learning aims to train a single model that serves many tasks and domains. A popular path to this *task unification* casts diverse problems into one graph-classification interface: node classification is reformulated as classifying k-hop ego subgraphs around an anchor node, and link prediction as classifying enclosing subgraphs around a node pair. This design simplifies pipelines and supports large-scale pretraining with shared architectures [7]. For link prediction, enclosing-subgraph classification (SEAL) set a strong precedent [22], followed by scalable sampling and sketching variants [9,1]. Recent node methods similarly treat each node as an anchor-centered subgraph instance with flags and distance encodings, moving node classification into the subgraph classification interface [21]. A basic question remains: *when is the node→subgraph conversion faithful, and when does it quietly lose the signal that matters?*

We study this question through the lens of homophily versus heterophily, which governs how informative neighborhoods are for node labels [11,24,23]. We run a controlled comparison on standard homophilic and heterophilic benchmarks with representative GNNs (GCN, GraphSAGE, GAT), graph transformers (GPS, GT), and GPR-GNN, matching features, training budgets, and model sizes,

and preventing overlap leakage. Two findings summarize the story. **(i)** On homophilic datasets, subgraph classification closely tracks node classification across backbones, with only mild monotone drops as k increases, so task unification is largely *safe*. **(ii)** On heterophilic datasets, converting node classification to k-hop subgraph classification induces substantial and consistent accuracy losses that *grow with* k, regardless of the encoder.

To explain and predict when conversion works, we introduce a simple, task-aligned neighborhood diagnostic: a k-hop *graph label alignment* score that measures how supportive a node's k-hop neighborhood is for its label. This anchor-centric diagnostic correlates with dataset homophily and is inversely associated with the node→subgraph performance gap. High alignment predicts near parity; low alignment predicts larger, k-amplified losses. We also observe a modest second-order effect on some strongly heterophilous graphs where alignment, hence accuracy, can briefly improve at k=2 before deteriorating at k=3.

In this paper, we investigate the following **research questions**:

- **RQ1:** Under which homophily to heterophily regimes does the node→subgraph conversion preserve accuracy?
- **RQ2:** How does the conversion error evolve with neighborhood radius k, and can a local alignment score predict the resulting transfer gap?
- **RQ3:** Can lightweight alignment-aware variants narrow the gap while retaining a unified subgraph interface?

Beyond diagnosis, our results suggest practical guidance. Alignment-aware pooling that downweights outer shells, boundary normalization, and relation-aware messaging reduce the transfer gap on heterophily while preserving the subgraph interface. These ideas complement heterophily-aware architectures such as GPR-GNN [2] and other propagation variants [6].

Positioning and contributions. Prior work advances subgraph reformulations and analyzes when GNNs succeed under homophily and struggle under heterophily [22,24,23], with recent node-focused subgraph formulations reporting competitive accuracy [21]. Our goal is not to introduce yet another architecture, but to clarify *when* the popular k-hop subgraph reformulation is reliable and when it is brittle. We complement these threads by isolating the *conversion* itself and by offering a compact diagnostic that indicates when unification is reliable.

- **Systematic evaluation:** controlled comparison of node classification versus k-hop subgraph classification across homophilic and heterophilic benchmarks and GNN or transformer backbones, with compute-matched training and leakage-free splits.
- **Alignment diagnostic:** a simple neighborhood measure that tracks homophily and reliably predicts the node→subgraph transfer gap, which increases with k when alignment is low.
- **Actionable guidance:** alignment-aware variants (pooling, boundary normalization, relation-aware channels) that narrow the gap on heterophily while retaining a unified subgraph interface.

2 Related Work

2.1 Task unification via subgraph reformulation

A growing line of work seeks to homogenize graph prediction tasks by operating on induced subgraphs with a single graph classification backbone. In this view, node classification is recast as classifying k-hop ego graphs anchored at a target node, and link prediction as classifying small enclosing subgraphs around a node pair. The SEAL framework established this paradigm for link prediction by showing that classifying h-hop enclosing subgraphs is highly effective compared to full-graph encoders [22]. Subsequent work improved efficiency and scalability through principled sampling of enclosing subgraphs [9] and subgraph sketching that retains task signal while avoiding costly extraction [1]. In parallel, recent discussions on graph foundation models emphasize such task unification to enable shared architectures and large-scale pretraining [7].

For node tasks, recent approaches treat each node as an anchor-centered subgraph instance with anchor flags and distance encodings, effectively turning node classification into subgraph classification. SubGND exemplifies this direction, proposing a dedicated subgraph-based formulation and reporting competitive accuracy under this unified interface [20]. Existing work on subgraph-based learning thus focuses on designing powerful architectures and demonstrating that enclosing or ego subgraphs can match or improve standard node and link prediction baselines. Our contribution is complementary: rather than proposing a new model, we *isolate the node→subgraph conversion step* itself and ask when this task unification is faithful. We introduce an anchor-centric alignment metric that predicts the node→subgraph transfer gap and provides guidance on choosing k or avoiding the reformulation when alignment is low.

2.2 Homophily, heterophily, and alignment measures

Homophily, the tendency of connected nodes to share labels or attributes, has been a central explanatory factor for when GNNs succeed, whereas heterophily often degrades message passing by mixing conflicting signals across edges. Classic assortativity quantifies such mixing at the network level [11], and graph learning studies relate edge or node homophily to performance limits and design choices beyond vanilla message passing [24]. Recent surveys organize heterophily-aware architectures and emphasize that global homophily alone only partially predicts difficulty [23]. Complementing this, Platonov et al. argue that commonly used homophily metrics hinder cross-dataset comparability and advocate *adjusted homophily* as a better-behaved alternative, together with a label-informativeness view that distinguishes types of heterophily [13,14].

We adopt a local, anchor-centric perspective via a simple *alignment* measure: for each node, we compute the fraction of same-class nodes within its k-hop ego neighborhood and then average over nodes. Unlike global assortativity or dataset-level homophily indices that average over all edges or nodes, this neighborhood alignment is computed on the specific k-hop ego around each anchor and thus

reflects the exact evidence a k-hop subgraph classifier aggregates after reformulating node classification as graph classification. Empirically, lower alignment corresponds to larger node→subgraph transfer gaps, and this effect strengthens with k. We report both our alignment and the dataset's adjusted homophily to provide complementary views of difficulty.

2.3 Graph foundation pipelines and task homogenization

Graph foundation models (GFMs) aim to pretrain a single, versatile backbone that can be adapted across tasks and domains, echoing the trajectory of language and vision foundation models. Recent surveys describe GFMs as pipelines with three stages: large-scale pretraining (for example, masked or contrastive objectives), task adaptation or prompting, and lightweight finetuning or head swapping [8,10]. Within this paradigm, task homogenization is a recurring design choice where diverse tasks are mapped into a common interface so a single encoder can be reused. In practice, this often means reformulating node and link prediction as subgraph or graph classification, using anchor-centered ego graphs for node labels and enclosing subgraphs for links, so that pretraining and adaptation reuse the same classification stack [8,19].

Our study directly interrogates this homogenization step for node tasks. While foundation-style graph transformers and generalist backbones report broad transfer after pretraining on heterogeneous corpora [5], it remains unclear when the node→subgraph conversion itself preserves discriminative signal. We provide controlled evidence that accuracy parity largely holds in homophilous regimes but degrades in heterophily as the subgraph radius grows, and we introduce a simple alignment diagnostic that predicts this transfer gap. These findings complement GFM pipelines by clarifying when task homogenization is reliable and when adaptation should be alignment-aware or avoid subgraph reformulation altogether.

3 Node→Graph

We study when converting node classification into subgraph (graph) classification preserves accuracy, and when it fails. This section formalizes the task transfer, describes our subgraph construction and evaluation protocol, defines an alignment metric that predicts transfer success, and outlines small design variants that mitigate observed failures.

3.1 Problem Setup and Task Transfer

Let $G = (V, E, X, Y)$ be a graph with node features $X \in \mathbb{R}^{|V| \times F}$ and node labels $Y = \{y_v\}_{v \in V}$, $y_v \in \{1, \ldots, C\}$. The standard node classification problem seeks a predictor $f_{\text{node}} : (G, v) \mapsto \hat{y}_v$.

We define the k hop task transfer operator T_k that maps a node instance (G, v) to an induced ego subgraph $S_k(v) = G\big[\mathcal{N}_k(v) \cup \{v\}\big]$ with anchor v and inherited

label y_v. The transferred task is graph classification on the subgraph distribution $\mathcal{D}_k = \{(S_k(v), y_v) : v \in V\}$, solved by a predictor $f_{\text{sub}} : S_k(v) \mapsto \hat{y}_v$. In all comparisons, f_{node} and f_{sub} share the same backbone family and hyperparameter grid. We evaluate $k \in \{1, 2, 3\}$.

Transfer gap. For a dataset $\mathcal{G}$ and backbone $\mathcal{B}$, let Acc_{node} be the node classification accuracy and $\text{Acc}_{\text{sub}}(k)$ be the subgraph classification accuracy at hop k. The transfer gap is $\Delta_k = \text{Acc}_{\text{node}} - \text{Acc}_{\text{sub}}(k)$.

We report Δ_k per dataset and backbone, and analyze its relation to alignment (Section 3.4).

3.2 Subgraph Construction and Controls

Features and anchors. Each $S_k(v)$ carries the original node features restricted to its nodes. We concatenate a one hot anchor flag to mark v in $S_k(v)$ and a radial encoding that records the integer hop distance to v.

Readout and pooling. We evaluate several readouts on subgraphs so that conclusions are not tied to a specific pooling: mean pooling, attention pooling, and sort pooling. Graph transformers use a learnable class token with global attention as readout.

Leakage free splits. For node classification, we use standard transductive masks. For subgraphs, we prevent label leakage by grouping all ego subgraphs of a test node into the test fold and disallowing any subgraph whose *anchor* lies in the test set from appearing in training. Test nodes may still appear as unlabeled neighbors inside training subgraphs, as in the usual transductive setting, but no subgraph with a test anchor is ever used to update the model. Grouping is performed per random split seed so that structure is preserved and labeled information about test anchors is omitted.

Compute matching. We match training budgets across node and subgraph tasks: identical optimizer, batch size, epochs, early stopping patience, hidden dimensions, and thus comparable parameter counts per backbone. We also report best tuned settings alongside the compute matched ones in our experiments.

3.3 Backbones and Training

We instantiate six representative backbones: GCN [4], GraphSAGE [3], GAT [18], GPS [15], a Graph Transformer (GT), and Generalized PageRank GNN (GPR-GNN) [2]. For node classification, the backbone operates on G. For subgraph classification, the same backbone operates on $S_k(v)$ and shares its hyperparameter grid with the node setting. Training uses cross entropy loss with class weights for imbalanced datasets and early stopping on a validation split that respects the grouping rule above. Optimization uses Adam with cosine learning rate decay, batch norm, and dropout.

3.4 Alignment Metric

We quantify how supportive the k hop ego subgraph is for the anchor label.

Definition 1. *For a fixed node $v \in V$ with k hop subgraph $S_k(v)$, we define the* k-hop graph label alignment $N2GLA_k(v)$ *to be*

$$\mathrm{N2GLA}_k(v) := \frac{|\{u \in S_k(v) : y_u = y_v\}|}{|S_k(v)|} = \frac{1 + |\{u \in \mathcal{N}_k(v) : y_u = y_v\}|}{1 + |\mathcal{N}_k(v)|}$$

and for a graph $G = (V, E)$ we define $\mathrm{N2GLA}_k$ *to be the average score:*

$$\mathrm{N2GLA}_k := \frac{1}{|V|} \sum_{v \in V} N2GLA_k(v).$$

This is an anchor centric neighborhood homophily score: it measures the fraction of nodes in the k hop ego that share the anchor's label, including the anchor itself. Unlike global assortativity or dataset level homophily indices that average over all edges or nodes, $\mathrm{N2GLA}_k$ is computed on the exact k hop egos that a subgraph classifier sees after the node→graph conversion.

Summary. Our pipeline isolates the conversion itself by holding features, training budgets, and architectures constant across node and subgraph views, enforcing leakage free evaluation, and using an interpretable alignment metric to explain when task transfer succeeds.

4 Experiments

We evaluate node→subgraph task transfer across a variety of datasets, encompassing the spectrum of homophily to gain further insight into the limitations of task transfer. We share our Python implementation at the link [1].

4.1 Experimental Setup

Datasets. We use three widely-used homophilic citation-network datasets: Cora, Citeseer, and PubMed [16]; and six heterophilic datasets: Texas, Cornell, Wisconsin, Actor [12], Roman-Empire and Chameleon [14]. Following the findings of [14], which identified train-test data leakage due to node and edge duplication in Chameleon*, we use the filtered variant, *Chameleon-filtered*, to ensure reliable evaluation. Dataset information can be found in Table 1.

The homophilic datasets are citation networks where nodes represent scientific publications and edges indicate citation links. Labels correspond to research topics. The heterophilic benchmarks span Wikipedia networks, bipartite graphs, and synthetic datasets with low feature-label correlation. Texas, Cornell, and Wisconsin are webpage datasets where nodes are web pages and edges are

[1] https://anonymous.4open.science/r/node2graph-D112

hyperlinks between them. Actor is a co-occurrence network, induced from the film-director-actor-writer network introduced in [17], where nodes correspond to an actor and edges denote co-occurence on the same Wikipedia page. Roman-Empire is a synthetic graph with controlled structural and label homophily properties. Chameleon is a Wikipedia page graph characterized by strong heterophily where nodes are articles and edges represent mutual links.

Table 1. Benchmark datasets for node classification

Dataset	Nodes	Edges	Classes	Features	Avg. Degree	Homophiliy
Cora	2,708	10,556	7	1,433	7.80	0.83
Citeseer	3,327	9,104	6	3,703	5.48	0.72
PubMed	19,717	88,648	3	500	9.00	0.79
Texas	183	325	5	1,703	3.56	0.10
Cornell	183	298	5	1,703	3.26	0.39
Wisconsin	251	515	5	1,703	4.10	0.15
Actor	7,600	30,019	5	932	7.90	0.20
Roman	22,622	32,927	18	300	5.82	0.05
Chameleon∗	890	8,854	5	2,325	19.90	0.24

Model Setup and Metrics. For all datasets we split nodes into a 60%, 20%, 20% split for training, validation, and testing and report accuracy averages over 5 seeded runs. While other metrics, in particular F1 score, may be more relevant for some class-imbalanced datasets, we focus solely on accuracy to evaluate the performance gap which occurs in task transfer.

Model Hyperparameters. All backbones share the same training setup. We use adjustable batch-sizes depending on the dataset with 3 hidden layers. For attention-based models (GAT/GPS/GT), we use 4 heads; non-attention backbones ignore this parameter. A learning rate of 0.001 is employed with the Adam optimization method capped at a maximum of 200 epochs with early stopping based on validation performance.

The results of the experiment are in Table 2.

4.2 Results: Task transfer across homophily regimes

Setup recap. Our evaluation isolates the effect of the conversion itself by (i) holding features, budgets, and model sizes constant between NC and GC, and (ii) enforcing leakage–free subgraph splits via anchor grouping Section 3.2 which we found critical on small heterophilous graphs.

RQ1 — When does node→subgraph preserve accuracy? On homophilic graphs (Cora, Citeseer, PubMed), 1-hop GC closely tracks NC for all six backbones, including the propagation-heavy GPR-GNN. Gaps at k=1 are small and typically within a few points, then grow mildly and monotonically as k increases

Table 2. Accuracy of node classification (NC) compared with subgraph classification (GC) on k-hop ego graphs across GNN and GT backbones. Node homophily ratios are shown below the dataset names.

		Homophilic			Heterophilic					
Backbone	**Task**	**Cora** 0.83	**Citeseer** 0.72	**PubMed** 0.79	**Texas** 0.10	**Cornell** 0.39	**Wisconsin** 0.15	**Actor** 0.20	**Roman** 0.05	**Chameleon*** 0.24
GCN	NC	88.18±0.98	76.04±1.18	87.94±0.44	48.95±3.94	47.89±4.53	49.41±6.13	28.47±1.13	51.60±0.46	38.20±3.55
	1-hop GC	84.57±0.70	71.92±1.35	86.76±0.31	31.05±10.84	36.32±3.87	39.22±3.28	23.37±0.45	60.00±0.71	37.30±1.61
	2-hop GC	82.87±1.07	70.75±0.99	81.89±0.38	35.26±4.88	27.89±4.88	48.24±10.49	22.76±0.73	26.55±1.18	40.11±2.09
	3-hop GC	81.40±2.08	69.85±0.61	80.93±0.61	30.00±4.59	27.37±6.78	39.61±5.02	20.80±1.03	11.81±0.41	37.19±1.86
GSAGE	NC	87.33±1.43	75.05±0.88	89.53±0.30	81.05±5.37	82.63±4.59	85.88±4.37	35.05±1.44	78.08±0.24	39.89±2.73
	1-hop GC	85.12±1.25	71.23±1.20	87.58±0.45	42.11±6.86	35.26±12.85	41.57±4.54	27.30±0.69	68.27±0.73	38.99±2.88
	2-hop GC	82.54±1.59	70.51±0.42	81.54±0.72	36.32±6.32	25.79±1.97	37.65±8.36	25.08±0.18	37.21±0.18	37.98±3.84
	3-hop GC	81.95±2.18	69.22±0.65	81.20±0.58	35.79±7.37	33.68±9.47	41.96±10.27	23.95±1.23	14.01±0.64	36.18±2.81
GAT	NC	87.40±0.45	74.44±0.60	86.80±0.57	49.47±8.39	52.11±5.86	51.76±6.15	28.83±1.68	47.00±0.21	40.22±2.03
	1-hop GC	85.45±0.74	72.43±1.05	88.10±0.36	35.79±15.49	35.26±6.36	32.16±4.90	21.89±0.49	44.44±0.63	34.83±4.75
	2-hop GC	82.87±1.49	71.80±0.79	82.22±0.44	25.79±13.87	22.11±3.57	33.33±6.44	19.99±0.74	19.11±0.56	36.74±5.59
	3-hop GC	80.52±1.49	69.49±0.93	81.48±0.52	22.63±12.74	20.53±3.07	34.90±8.72	18.71±1.57	11.88±0.21	37.30±4.99
GPS	NC	85.05±1.27	75.86±0.86	88.52±0.24	80.00±7.55	81.58±1.66	87.84±2.29	35.63±0.46	76.08±0.32	41.12±2.39
	1-hop GC	85.01±0.28	71.44±0.54	87.75±0.35	42.11±6.86	60.00±7.88	52.16±6.97	27.20±1.14	74.88±1.34	37.19±1.64
	2-hop GC	82.43±1.18	70.75±0.63	82.65±0.54	36.32±6.09	58.95±2.68	53.73±2.00	25.75±0.36	46.75±1.25	36.74±3.03
	3-hop GC	80.22±1.57	69.61±0.69	81.85±0.47	37.89±4.88	58.95±1.29	50.98±6.20	26.22±1.29	13.30±0.64	35.96±4.19
GT	NC	85.78±1.31	72.88±0.87	89.02±0.58	71.05±11.65	65.26±7.88	73.33±8.00	36.57±0.81	74.91±0.74	42.47±6.79
	1-hop GC	84.97±0.76	72.55±0.71	88.63±0.42	39.47±8.49	37.37±16.84	33.33±6.68	28.87±1.15	67.00±0.73	38.54±3.58
	2-hop GC	82.91±1.51	71.26±0.54	82.38±0.43	29.47±14.26	28.95±16.89	29.02±11.05	26.28±1.47	30.15±0.83	37.30±2.37
	3-hop GC	80.48±1.65	70.30±1.06	81.80±0.92	24.74±15.67	31.05±17.80	31.37±6.32	25.70±1.21	13.13±0.35	35.84±2.05
GPR-GNN	NC	87.07±1.39	75.05±1.76	90.01±0.33	55.26±4.40	57.89±6.45	63.14±8.72	34.29±0.79	58.42±0.68	38.65±4.18
	1-hop GC	86.26±1.35	73.26±1.25	87.87±0.33	49.47±4.53	37.89±7.91	41.18±11.83	29.93±1.55	45.11±0.70	36.85±3.75
	2-hop GC	83.54±2.44	72.97±0.63	82.76±0.35	48.95±3.94	39.47±13.42	39.22±9.53	27.22±0.47	20.86±0.84	35.28±2.23
	3-hop GC	81.66±3.18	71.74±0.79	81.36±0.49	48.42±7.55	38.42±8.90	44.71±8.36	25.99±0.94	13.51±0.59	34.04±1.97

Table 3. Transfer gap Δ_k for each backbone across homophilic and heterophilic datasets.

		Homophilic			Heterophilic					
Backbone	**Gap**	**Cora**	**Citeseer**	**PubMed**	**Texas**	**Cornell**	**Wisconsin**	**Actor**	**Roman**	**Chameleon***
GCN	Δ_1	3.61	4.12	1.18	17.90	11.57	10.19	5.10	-8.40	0.90
	Δ_2	5.31	5.29	6.05	13.69	20.00	1.17	5.71	25.05	-1.91
	Δ_3	6.78	6.19	7.01	18.95	20.52	9.80	7.67	39.79	1.01
GSAGE	Δ_1	2.21	3.82	1.95	38.94	47.37	44.31	7.75	9.81	0.90
	Δ_2	4.79	4.54	7.99	44.73	56.84	48.23	9.97	40.87	1.91
	Δ_3	5.38	5.83	8.33	45.26	48.95	43.92	11.10	64.07	3.71
GAT	Δ_1	1.95	2.01	-1.30	13.68	16.85	19.60	6.94	2.56	5.39
	Δ_2	4.53	2.64	4.58	23.68	30.00	18.43	8.84	27.89	3.48
	Δ_3	6.88	4.95	5.32	26.84	31.58	16.86	10.12	35.12	2.92
GPS	Δ_1	0.04	4.42	0.77	37.89	21.58	35.68	8.43	1.20	40.38
	Δ_2	2.62	5.11	5.87	43.68	22.63	34.11	9.88	29.33	4.38
	Δ_3	4.83	6.25	6.67	42.11	22.63	36.86	9.41	62.78	5.16
GT	Δ_1	0.81	0.33	0.39	31.58	27.89	40.00	7.70	7.91	3.93
	Δ_2	2.87	1.62	6.64	41.58	36.31	44.31	10.29	44.76	5.17
	Δ_3	5.30	2.58	7.22	46.31	34.21	41.96	10.87	61.78	6.63
GPR-GNN	Δ_1	1.72	2.94	0.59	27.99	25.05	29.95	7.18	2.61	3.01
	Δ_2	4.02	3.84	6.22	33.47	33.15	29.25	8.93	33.58	2.60
	Δ_3	5.83	5.16	6.91	35.89	31.57	29.88	9.83	52.70	3.88

to 2 and 3, with the same qualitative pattern for GNNs and graph transformers. This near-parity regime is evident in the main accuracy table (Table 2) and its gap summary (Tables 3 and 5). The consistency across GCN, GraphSAGE, GAT, GPS, GT, and GPR-GNN indicates that, in assortative settings, the bottleneck is the conversion interface rather than the encoder class.

Table 4. Node homophily ratios and k-hop graph label alignment scores (N2GLA$_k$) across datasets. The last column reports Pearson correlations between homophily and N2GLA$_k$.

Dataset	Cora	Citeseer	PubMed	Texas	Cornell	Wisconsin	Actor	Roman	Chameleon*	Corr.
Node Hom.	0.83	0.72	0.79	0.10	0.39	0.15	0.20	0.05	0.24	
1-Hop	0.8708	0.8199	0.8647	0.3908	0.4268	0.4224	0.4124	0.3045	0.3503	0.97
2-Hop	0.7750	0.7579	0.7608	0.5352	0.4076	0.4341	0.2401	0.1931	0.2956	0.87
3-Hop	0.6626	0.7115	0.6871	0.3536	0.2888	0.3265	0.2157	0.1468	0.2445	0.92

Table 5. Average transfer gap across all backbones. The last column reports Pearson correlations between average performance gap and N2GLA$_k$ scores found in Table 4.

Dataset	Cora	Citeseer	PubMed	Texas	Cornell	Wisconsin	Actor	Roman	Chameleon*	Corr.
Δ_1	1.571	2.731	1.288	24.296	24.210	28.623	6.713	7.198	2.808	-0.522
Δ_2	3.941	3.546	6.396	28.945	30.700	28.361	8.626	34.243	3.370	-0.471
Δ_3	5.763	4.851	7.200	31.051	29.560	27.971	9.578	51.408	4.006	-0.605

RQ2 — How does error evolve with radius k, and what predicts it? On heterophilic datasets (Texas, Cornell, Wisconsin, Actor, Roman-Empire) the conversion induces large accuracy losses that *widen* with k for most backbones, and the variance across seeds increases, reflecting instability of pooled neighborhood evidence. Any sporadic k=1 gains (for example, GCN on Roman-Empire) vanish or reverse by k=2 and k=3 (Tables 2 and 3). A simple neighborhood diagnostic explains this behavior: our k-hop graph-label alignment, N2GLA$_k$, is high at k=1 on homophilic graphs and declines moderately with radius, while it starts low on heterophilic graphs and often deteriorates further by k=3 with occasional second-order bumps at k=2 (for example, Texas) (Table 4). As a result, average transfer gaps are inversely related to N2GLA$_k$ for all radii (Table 5, Figure 1); backbone-specific scatter plots reproduce the same inverse trend and k-ordering, confirming that the effect stems from the conversion rather than a particular encoder. The chief outlier is Chameleon, which has very high average degree, so the ego subgraph already contains much of the node's 1-hop information and the measured gap stays small despite decreasing alignment.

Quantitative summary. Across heterophilous graphs, the mean gap at k=1 is already large and typically expands at k=2 and k=3 (see $\Delta_1, \Delta_2, \Delta_3$ in Table 3); by contrast, homophilous graphs show small Δ_1 that grow only modestly with radius. Pearson correlations between homophily and N2GLA$_k$ are strong for all radii (last column of Table 4), and the correlation between average gap and alignment is negative for each k (last column of Table 5).

RQ3 — Can alignment-aware choices mitigate the gap while keeping a unified interface? Simple, encoder-agnostic readouts can help in low-alignment regimes. Pooling that downweights outer shells, boundary-aware normalization, and relation-aware channels tend to reduce the heterophilous gap in our experiments without affecting homophilous cases, while preserving the unified subgraph interface. This suggests a practical recipe: compute N2GLA$_k$,

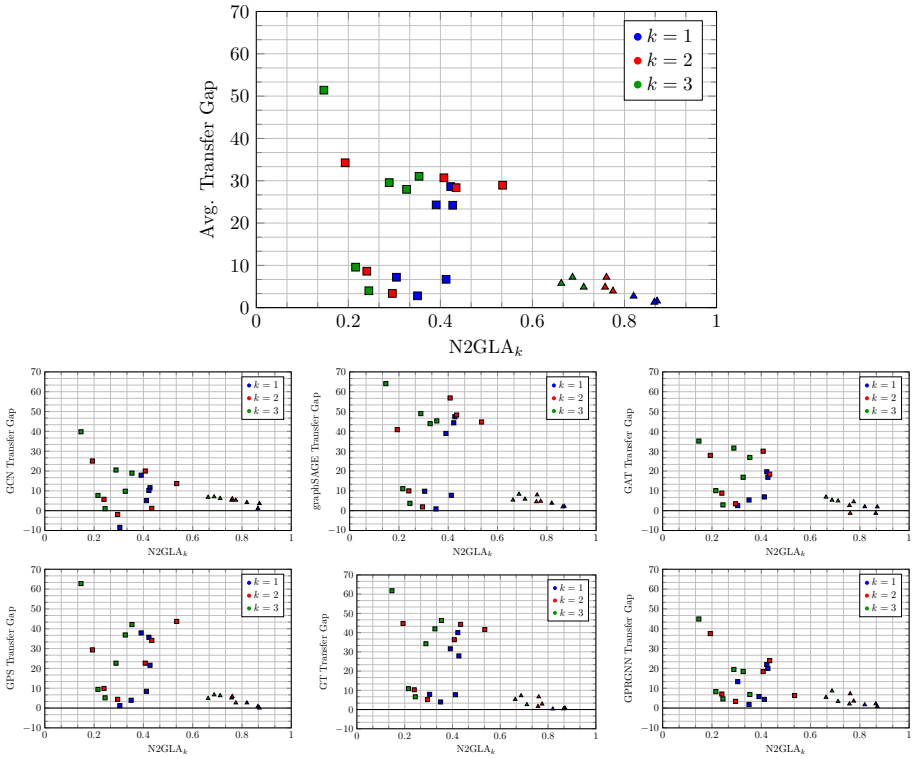

Fig. 1. Transfer gap versus N2GLA$_k$. For each dataset and radius, we show an average over backbones (top) and separate plots per backbone (bottom). Homophilic datasets are plotted as triangles and heterophilic datasets as squares. On heterophilic graphs, increasing k moves points to the right (lower alignment) and up (larger gap).

pick the smallest k that maintains adequate alignment, and if alignment remains low, prefer alignment-aware pooling or avoid naïve reformulation.

Takeaway and practical recipe. Alignment is the governing variable: homophilic graphs yield high N2GLA$_1$ and near parity between NC and GC, while heterophilic graphs exhibit low alignment and large, k-amplified gaps. This backbone-invariant pattern, visible in both the main tables and backbone-wise plots, provides a concrete diagnostic for safe task unification. In practice, we recommend the following decision rule for node→subgraph reformulation:

(i) compute N2GLA$_k$ for small radii k on the training graph;

(ii) if N2GLA$_1$ is high, use the smallest k (often k=1),and the unified subgraph interface is safe;

(iii) if N2GLA$_k$ remains low for all small k, avoid naïve task unification or use alignment-aware pooling and boundary normalization rather than simply increasing k.

Limitations and scope. We focus on single-graph node classification on widely used benchmarks and small radii $k \in \{1, 2, 3\}$, which already span a broad range of homophily levels and degrees. Our experiments target the *supervised adaptation* stage of graph foundation pipelines: we keep the encoder architecture fixed and study how the node→subgraph reformulation behaves under standard training, rather than varying pretraining objectives or large multi-graph corpora. The alignment metric we propose is label-based, which makes it directly usable in common semi-supervised settings; extending similar ideas to fully unlabeled pretraining or to denser graphs is a natural direction for future work.

5 Conclusion

In this paper, we examined the promise and limits of task unification that recasts node classification as k-hop subgraph classification. Across homophilic and heterophilic benchmarks and diverse GNN and graph transformer backbones, we found near parity between node and subgraph models in homophily, but substantial accuracy loss in heterophily that increases with the radius k. We introduced a simple, task aligned neighborhood metric, $\mathrm{N2GLA}_k$, which strongly correlates with dataset homophily and reliably predicts the node to subgraph transfer gap: high alignment implies safe transfer, while low alignment signals degradation, especially at larger k. These results yield practical guidance for graph foundation pipelines: prefer small radius egos or alignment aware pooling in assortative domains, and avoid naive subgraph reformulation in strongly heterophilous settings unless paired with corrective designs such as boundary normalization or relation aware channels. Looking ahead, we will develop alignment conditioned pretraining and adapters to preserve node task signal under a unified subgraph interface, learn adaptive k selection during training, provide formal guarantees and robustness analyses for $\mathrm{N2GLA}_k$, and extend the study to link prediction, multirelational and temporal graphs, and scalable alignment aware subgraph sketching.

Acknowledgments. This work was partially supported by National Science Foundation under grants DMS-2220613 and DMS-2229417. The authors acknowledge the Texas Advanced Computing Center (TACC) at UT Austin for providing computational resources that have contributed to the research results reported within this paper.

References

1. Chamberlain, B.P., Shirobokov, S., Rossi, E., Frasca, F., Markovich, T., Hammerla, N., Bronstein, M.M., Hansmire, M.: Graph neural networks for link prediction with subgraph sketching. ICLR (2023)
2. Chien, E., et al.: Adaptive universal generalized pagerank graph neural network. ICLR (2021)
3. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. NeurIPS **30** (2017)

4. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: ICLR (2017)
5. Lachi, D., Azabou, M., Arora, V., Dyer, E.: Graphfm: A scalable framework for multi-graph pretraining. arXiv preprint arXiv:2407.11907 (2024)
6. Lim, D., Hohne, F., Li, X., Huang, S.L., Gupta, V., Bhalerao, O., Lim, S.N.: Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. NeurIPS **34**, 20887–20902 (2021)
7. Liu, J., Li, X., Liu, Y., Liu, M., Yu, P.S.: Graph foundation models: Concepts, opportunities and challenges. arXiv preprint arXiv:2310.11829 (2023)
8. Liu, J., Yang, C., Lu, Z., Chen, J., Li, Y., Zhang, M., Bai, T., Fang, Y., Sun, L., Yu, P.S., et al.: Graph foundation models: Concepts, opportunities and challenges. IEEE Transactions on Pattern Analysis and Machine Intelligence (2025)
9. Louis, P., Babaei, V., Rossi, R.A., Rao, A., Ahmed, N.K.: Sampling enclosing subgraphs for link prediction. In: Proceedings of the 31st ACM International Conference on Information and Knowledge Management. pp. 1998–2007 (2022)
10. Mao, H., et al.: Position: Graph foundation models are already here. In: ICML (2024)
11. Newman, M.E.J.: Mixing patterns in networks. Physical Review E **67**(2), 026126 (2003)
12. Pei, H., Wei, B., Chang, K.C.C., Lei, Y., Yang, B.: Geom-gcn: Geometric graph convolutional networks. arXiv preprint arXiv:2002.05287 (2020)
13. Platonov, O., Kuznedelev, D., Babenko, A., Prokhorenkova, L.: Characterizing graph datasets for node classification: Homophily-heterophily dichotomy and beyond. NeurIPS **36**, 523–548 (2023)
14. Platonov, O., Kuznedelev, D., Diskin, M., Babenko, A., Prokhorenkova, L.: A critical look at the evaluation of gnns under heterophily: Are we really making progress? In: ICLR (2023)
15. Rampášek, L., Galkin, M., Dwivedi, V.P., Luu, A.T., Wolf, G., Beaini, D.: Recipe for a general, powerful, scalable graph transformer. NeurIPS **35**, 14501–14515 (2022)
16. Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., Eliassi-Rad, T.: Collective classification in network data. AI Magazine **29**(3), 93–93 (2008)
17. Tang, J., Sun, J., Wang, C., Yang, Z.: Social influence analysis in large-scale networks. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 807–816 (2009)
18. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. ICLR (2018)
19. Wang, Z., Liu, Z., Ma, T., Li, J., Zhang, Z., Fu, X., Li, Y., Yuan, Z., Song, W., Ma, Y., et al.: Graph foundation models: A comprehensive survey. arXiv preprint arXiv:2505.15116 (2025)
20. Zeng, Q., Lin, X., Gao, J., Yu, Y.: Using subgraph gnns for node classification: an overlooked potential approach. arXiv preprint arXiv:2503.06614 (2025)
21. Zeng, Q., et al.: Using subgraph gnns for node classification: an efficient reformulation. arXiv:2503.06614 (2025)
22. Zhang, M., Chen, Y.: Link prediction based on graph neural networks. In: NeurIPS (2018)
23. Zheng, X., Wang, Y., Liu, Y., Li, M., Zhang, M., Jin, D., Yu, P.S., Pan, S.: Graph neural networks for graphs with heterophily: A survey. arXiv preprint arXiv:2202.07082 (2022), v3 updated Feb 2024
24. Zhu, J., Ma, J., Mei, Q., Tang, J., et al.: Beyond homophily in graph neural networks: Current limitations and effective designs. In: NeurIPS (2020)