UNDERSTANDING ROBUSTNESS TO NOISY SUPERVISION IN GRAPH NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph Neural Networks (GNNs) are vulnerable to noisy supervision, yet the role of architecture in mitigating this effect remains poorly understood. We study a modified version of Jumping Knowledge (JK) that preserves direct access to pre-propagation node representations by combining the initial and final representations at readout, while discarding intermediate propagation states. We theoretically show that this construction preserves information that message passing may make inaccessible to the prediction head, and characterize its advantage in terms of the prediction deficit of the propagated representation and the recoverability of this deficit from the pre-propagation representation. Across four GNN architectures, four datasets, three noise mechanisms, and three corruption levels, we find that JK improves accuracy under label noise by 4.3 points (around 7.0% relative) on average, with its benefit increasing with corruption in the regimes where information made inaccessible through propagation remains recoverable from the initial representation. Our analysis shows that label noise enlarges the prediction deficit across the corruption range considered, while recoverability remains comparatively stable, identifying recoverable prediction deficit as a key factor governing when representation preservation improves robustness to noisy supervision.

1 INTRODUCTION

Graph Neural Networks (GNNs) learn node representations by repeatedly propagating information through the graph (Scarselli et al., 2009; Wu et al., 2020). While this mechanism allows models to exploit relational structure, it also transforms the original node representation as information from neighboring nodes is incorporated (Yuan et al., 2023). Under noisy supervision, this creates an important question: *how much should the final predictor rely on representations produced after graph propagation?* This question is particularly relevant for graphs, where label noise can have effects beyond those observed in the *i.i.d.* setting. While a corrupted label directly affects only its associated example in the latter, in GNNs its influence on learning can interact with message passing, affecting representations beyond the corrupted node (Wang et al., 2024). At the same time, propagation provides valuable relational context that node features alone do not capture. Robustness to noisy supervision therefore cannot simply amount to reducing propagation; the relevant question is instead what information should remain accessible to the final predictor (Li et al., 2018).

Existing approaches to learning with noisy labels on graphs primarily address the problem through the training procedure, for example by identifying unreliable labels, correcting supervision, modifying the loss, or refining the graph structure (Dai et al., 2021; Li et al., 2021; Yuan et al., 2023; Chen et al., 2024). These methods are effective but operate outside the model: they change what the network is trained on, not how it represents the data. In this work, we take a different perspective and investigate whether robustness can emerge directly from the architecture. In particular, we study a simple principle: *preserving node representations before they are transformed by graph propagation*.

We investigate this principle through Jumping Knowledge (JK) (Xu et al., 2018b). Rather than using the complete sequence of intermediate representations, we consider a construction in which the predictor receives the initial, pre-propagation representation together with the final representation produced by the GNN. This gives the predictor access to node-level information that has not undergone message passing, alongside the contextual information learned through graph propagation (Luo et al., 2024). The intuition is that message passing may make information present before propaga-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

tion inaccessible to the final readout. Retaining a direct path from the encoded features to the prediction head preserves access to this information at essentially no cost.

We show theoretically that this construction provides a structural advantage whenever propagation collapses task-relevant distinctions. Retaining the initial representation preserves distinctions that may be collapsed by propagation and yields a predictor class that contains its propagated-only counterpart, independently of the particular message-passing operator. We further characterize the value of this additional pathway through two quantities: the prediction deficit, measuring the predictive signal inaccessible from the propagated representation, and its recoverability from the pre-propagation representation. We evaluate this mechanism on four graph architectures, four datasets, three noise mechanisms, and three corruption levels. JK improves accuracy by 4.3 points on average, a relative gain of 7.0%, although its benefit varies substantially across architectures and datasets. Our analysis shows that label noise enlarges the prediction deficit across the corruption range considered, while recoverability remains comparatively stable.

Our main contributions are the following:

- We adopt a minimal form of Jumping Knowledge that retains only the pre-propagation and final representations, and show theoretically that it preserves access to information that message passing may make inaccessible while yielding a predictor class that contains its propagated-only counterpart, independently of the propagation operator.
- We derive an exact decomposition of the optimal linear JK advantage into two factors: the prediction deficit of the propagated representation and the recoverability of this deficit from the pre-propagation representation. A large prediction deficit alone is insufficient; the missing predictive information must also remain recoverable from the preserved representation.
- Across four architectures, four datasets, three noise mechanisms, and three corruption levels, JK improves accuracy by 4.3 points (7.0% relative) on average. Our analysis shows that label noise enlarges the prediction deficit across the corruption range considered while recoverability remains comparatively stable, accounting for the increasing value of representation preservation in the regimes where JK provides the largest gains.

2 Background & Related Work

2.1 Noisy Labels and Jumping Knowledge

Learning with noisy labels. Learning with noisy labels has been widely studied through robust losses, sample reweighting, label correction, and explicit noise modeling (Yuan et al., 2023). In graph learning, the problem is further complicated by message passing, which couples node representations through the graph structure and can amplify the effect of unreliable supervision (Wang et al., 2024). Existing approaches therefore typically modify the learning procedure by estimating label reliability, correcting or reweighting supervision (Li et al., 2021; Chen et al., 2024; Wani et al., 2024; Cheung et al., 2018), or exploiting graph topology to identify unreliable samples and limit the effect of corrupted information (Dai et al., 2021; Wu et al., 2024). In contrast, we investigate whether robustness can emerge directly from architectural choices, without introducing an explicit noise-handling mechanism.

Jumping Knowledge. Jumping Knowledge (Xu et al., 2018b) combines representations from different propagation depths, allowing nodes to exploit information from neighborhoods of different ranges. H2GCN (Zhu et al., 2020) similarly separates ego- and neighbor-embeddings and combines intermediate representations at readout to cope with heterophily. Related architectures also preserve initial information during propagation: GCNII (Chen et al., 2020) introduces initial residual connections that repeatedly inject the original node representation, while APPNP (Gasteiger et al., 2018) combines propagated information with the initial representation through personalized PageRank. While these approaches address heterophily, depth, and oversmoothing, we study representation preservation specifically from the perspective of robustness to noisy supervision. We show that retaining the pre-propagation representation alongside the final propagated representation improves robustness across different propagation operators as label corruption increases.

2.2 GRAPH NEURAL NETWORKS

In this subsection, we review the four classes of GNNs that serve as the foundation for our analysis. Their propagation rules are presented below and illustrated in Fig. 2 in the Appendix.

Let $G = (V, E)$ be a graph with $n = |V|$ nodes and adjacency matrix $A \in \{0,1\}^{n\times n}$. We denote by $\mathcal{N}(u)$ the neighborhood of node u, and by $h_u^{(t)} \in \mathbb{R}^{f_t}$ its representation at layer t, where f_t is the corresponding feature dimension.

Graph Convolutional Network (GCN) (Kipf & Welling, 2017) aggregates neighboring representations using symmetric degree normalization. Let $\widetilde{A} = A + I_n$ be the adjacency matrix augmented with self-loops, where I_n is the $n \times n$ identity matrix, and let $\widetilde{D} = \mathrm{diag}(\widetilde{d}_1, \ldots, \widetilde{d}_n)$ be its degree matrix, with $\widetilde{d}_u = \sum_v \widetilde{A}_{uv}$. The normalized adjacency matrix is $\widehat{A} = \widetilde{D}^{-1/2}\widetilde{A}\widetilde{D}^{-1/2}$, yielding the per-node update

$$h_u^{(t+1)} = \sigma\left(\sum_{v\in\mathcal{N}(u)\cup\{u\}} \frac{1}{\sqrt{\widetilde{d}_u\,\widetilde{d}_v}} W^{(t)} h_v^{(t)}\right) \tag{1}$$

where $W^{(t)} \in \mathbb{R}^{f_{t+1}\times f_t}$ is a learnable weight matrix and σ is a nonlinear activation function. The aggregation weights are determined by the graph structure and node degrees.

GraphSAGE (Hamilton et al., 2017) separates neighborhood aggregation from its combination with the central node representation. A permutation-invariant aggregation function first computes $m_u^{(t)} = \mathrm{AGG}^{(t)}(\{h_v^{(t)} : v \in \mathcal{N}(u)\})$, which summarizes the representations in the neighborhood of u. The resulting summary is then combined with the central node representation:

$$h_u^{(t+1)} = \sigma\left(W^{(t)}\left[h_u^{(t)} \,\|\, m_u^{(t)}\right]\right) \tag{2}$$

where $\|$ denotes concatenation and $W^{(t)}$ is a learnable weight matrix. Since the update is defined through local neighborhood aggregation rather than node-specific parameters, GraphSAGE supports inductive inference on previously unseen nodes.

Graph Attention Network (GAT) (Veličković et al., 2018) learns data-dependent weights for neighboring nodes through an attention mechanism. Node representations are first projected as $z_u^{(t)} = W^{(t)} h_u^{(t)}$. For each $v \in \mathcal{N}(u) \cup \{u\}$, an unnormalized attention score is computed as $e_{uv}^{(t)} = \mathrm{LeakyReLU}\left(a^{(t)\top}[z_u^{(t)} \| z_v^{(t)}]\right)$, where $a^{(t)}$ is a learnable attention vector. The scores are normalized over the neighborhood as $\alpha_{uv}^{(t)} = \exp(e_{uv}^{(t)}) / \sum_{w\in\mathcal{N}(u)\cup\{u\}} \exp(e_{uw}^{(t)})$, yielding the update

$$h_u^{(t+1)} = \sigma\left(\sum_{v\in\mathcal{N}(u)\cup\{u\}} \alpha_{uv}^{(t)} z_v^{(t)}\right) \tag{3}$$

GAT employs multiple attention heads, whose outputs are concatenated or averaged in the final layer.

Graph Isomorphism Network (GIN) (Xu et al., 2018a) uses sum aggregation followed by a multi-layer perceptron, a formulation designed to match the discriminative power of the 1-Weisfeiler–Lehman (1-WL) test. Its update is

$$h_u^{(t+1)} = \mathrm{MLP}^{(t)}\left(\left(1+\epsilon^{(t)}\right) h_u^{(t)} + \sum_{v\in\mathcal{N}(u)} h_v^{(t)}\right) \tag{4}$$

Where $\mathrm{MLP}^{(t)}$ is a learnable multi-layer perceptron and $\epsilon^{(t)}$ is a scalar that may be fixed or learned. The sum aggregator preserves information about the multiplicities of neighboring representations, while $\epsilon^{(t)}$ controls the contribution of the central node.

3 SIGNAL RECOVERABILITY UNDER NOISY LABELS

Retaining the pre-propagation representation enlarges what the prediction head can express, but not every such enlargement is useful. We characterize when the additional pathway matters by

quantifying the predictive signal inaccessible from the propagated representation and how much of this signal remains recoverable from the pre-propagation representation. Our analysis is agnostic to the propagation operator and to how the representations are learned. All proofs are in Appendix A.

Setup. Let $X \in \mathbb{R}^{n \times f_{\text{in}}}$ denote the input node features and let $H^{(0)} = E_\theta(X) \in \mathbb{R}^{n \times f}$ be their encoded representation, which we refer to as the *pre-propagation representation*. Starting from $H^{(0)}$, a T-layer graph network recursively computes

$$H^{(t+1)} = \mathcal{S}^{(t)}\left(H^{(t)}\right) + \sigma\left(\mathcal{P}^{(t)}\left(H^{(t)}; G\right)\right), \qquad t = 0, \ldots, T-1, \tag{5}$$

where $\mathcal{S}^{(t)}$ is an optional layer-wise skip branch, and $\mathcal{P}^{(t)}$ is a graph propagation operator. The formulation is agnostic to the choice of operator: $\mathcal{P}^{(t)}$ may instantiate a graph convolution, an attention-based aggregation, or a sum aggregator, while $\mathcal{S}^{(t)}$ may be absent, an identity map, or a learnable transformation. We write Φ for the composition of all T layers, so that

$$H^{(T)} = \Phi(H^{(0)}; G),$$

and refer to $H^{(T)}$ as the *propagated representation*. We denote by $h_u^{(0)}$ and $h_u^{(T)}$ the rows of $H^{(0)}$ and $H^{(T)}$ associated with node u, respectively. We study node classification where a fraction of the training labels is corrupted: some nodes are given a wrong class before training, and the model never learns which. The corruption affects both the labels used for training and those used for model selection, while the model is evaluated against clean labels.

The JK readout. We consider a variant of Jumping Knowledge (JK) (Xu et al., 2018b) that retains only the representations before and after propagation. Rather than exposing all intermediate depths, the JK prediction head receives

$$R_{\text{JK}} = \left[H^{(0)} \,\|\, H^{(T)}\right], \tag{6}$$

whereas the standard readout receives only $R_{\text{ST}} = H^{(T)}$. This is distinct from the layer-wise skip $\mathcal{S}^{(t)}$ in Eq. (5): the latter operates inside the propagation backbone, between consecutive layers, and does not directly expose $H^{(0)}$ to the prediction head. This distinction matters for the architectures we evaluate, since some preserve representations internally while others do not.

We take the prediction head to produce logits through an affine map. For node u,

$$z_u^{\text{JK}} = A_0 h_u^{(0)} + A_T h_u^{(T)} + b, \qquad z_u^{\text{ST}} = A_T h_u^{(T)} + b. \tag{7}$$

We denote by $\mathcal{H}_{\text{ST}}$ and $\mathcal{H}_{\text{JK}}$ the classes of node-level predictors realizable by the two readouts as the head parameters range over their domain, with the encoder and propagation backbone held fixed.

3.1 Structural Properties of JK

Write:

$$\Psi_{\text{ST}}(H^{(0)}) = \Phi(H^{(0)}; G) \qquad \text{and} \qquad \Psi_{\text{JK}}(H^{(0)}) = \left[H^{(0)} \,\|\, \Phi(H^{(0)}; G)\right]$$

for the two representation maps. The first property of the JK construction is that propagation cannot remove the prediction head's access to distinctions present before propagation.

Property 1. *For any propagation backbone Φ,*

$$H^{(0)} \neq \widetilde{H}^{(0)} \quad \Longrightarrow \quad \Psi_{\text{JK}}(H^{(0)}) \neq \Psi_{\text{JK}}(\widetilde{H}^{(0)}).$$

The property holds because the first block of the concatenation differs, independently of the behavior of Φ. It does not imply that $H^{(0)}$ is informative for the downstream task. Rather, it establishes that propagation cannot remove the prediction head's access to distinctions already present before it. The standard representation provides no analogous guarantee, since two distinct encodings may satisfy

$$\Phi(H^{(0)}; G) = \Phi(\widetilde{H}^{(0)}; G)$$

and therefore become indistinguishable to a readout that observes only $H^{(T)}$.

The second property concerns the relation between the two predictor classes.

Property 2. *For a fixed encoder and propagation backbone, every predictor realizable by the standard readout is also realizable by the JK readout. Hence,*

$$\mathcal{H}_{\mathrm{ST}} \subseteq \mathcal{H}_{\mathrm{JK}}.$$

Indeed, setting $A_0 = 0$ in Eq. (7) recovers the standard logits exactly. The standard readout is therefore a special case of the JK readout.

3.2 The Value of Preserved Information

Property 2 establishes containment but not strictness. Containment alone does not imply an increase in effective representational capacity: if propagation preserves every distinction relevant to the prediction head, access to $H^{(0)}$ need not enlarge the set of realizable predictions. The following theorem gives a sufficient condition under which the enlargement is strict.

Theorem 1. *Fix the encoder and the backbone. Suppose there exist two nodes $u \neq w$ such that*

$$h_u^{(T)} = h_w^{(T)} \qquad \textit{and} \qquad h_u^{(0)} \neq h_w^{(0)}.$$

If the JK head family contains a predictor that separates the corresponding JK representations, then

$$\mathcal{H}_{\mathrm{ST}} \subsetneq \mathcal{H}_{\mathrm{JK}}.$$

No standard readout can separate u and w, whereas the JK readout can through their $H^{(0)}$ blocks.

The previous results do not explain why the value of the additional pathway may change with the label noise rate. To characterize this effect, we now quantify how much task-relevant signal is inaccessible from $H^{(T)}$ and how much of that signal remains recoverable from the complementary directions contributed by $H^{(0)}$.

Let $Y \in \mathbb{R}^{n \times C}$ denote the clean target matrix. In this subsection, $H^{(0)}$ and $H^{(T)}$ are understood to be representations learned by a model trained at label noise rate ρ and therefore depend implicitly on ρ. We specialize the prediction heads in Eq. (7) to optimal linear least-squares readouts and assume centered representations and targets, equivalently allowing an intercept.

Let P_T denote the orthogonal projector onto $\mathrm{col}(H^{(T)})$. The optimal least-squares risk achievable from the standard readout is

$$\mathcal{R}^*_{\mathrm{ST}}(\rho) = \min_W \frac{1}{n}\left\|Y - H^{(T)}W\right\|_F^2 = \frac{1}{n}\left\|(I - P_T)Y\right\|_F^2. \tag{8}$$

We call this quantity the *prediction deficit*, $D(\rho)$, which measures the amount of clean target signal that is not linearly accessible from the propagated representation.

The JK readout additionally exposes $H^{(0)}$ to the predictor. However, only directions in $H^{(0)}$ lying outside the column space of $H^{(T)}$ can enlarge the linear predictor space. We therefore define the complementary component

$$\widetilde{H}^{(0)} := (I - P_T)H^{(0)}, \tag{9}$$

and let Q denote the orthogonal projector onto $\mathrm{col}(\widetilde{H}^{(0)})$. By construction $\mathrm{col}(\widetilde{H}^{(0)}) \perp \mathrm{col}(H^{(T)})$:

$$\mathrm{col}\left(\left[H^{(0)} \,\|\, H^{(T)}\right]\right) = \mathrm{col}(H^{(T)}) \oplus \mathrm{col}(\widetilde{H}^{(0)}). \tag{10}$$

Proposition 1. *Let $\mathcal{R}^*_{\mathrm{JK}}(\rho)$ denote the optimal least-squares risk obtained from the JK representation $[H^{(0)} \,\|\, H^{(T)}]$. Then*

$$\mathcal{R}^*_{\mathrm{ST}}(\rho) - \mathcal{R}^*_{\mathrm{JK}}(\rho) = \frac{1}{n}\left\|Q(I - P_T)Y\right\|_F^2. \tag{11}$$

Proposition 1 shows that JK does not benefit merely from observing an additional representation. Its optimal linear advantage is exactly the portion of the residual target signal left unexplained by $H^{(T)}$ that lies in directions contributed by $H^{(0)}$ but absent from $H^{(T)}$.

To make this interaction explicit, define the *recoverability* of the prediction deficit as

$$\Gamma(\rho) := \frac{\left\| Q(I - P_T)Y \right\|_F^2}{\left\| (I - P_T)Y \right\|_F^2}, \qquad 0 \le \Gamma(\rho) \le 1, \tag{12}$$

whenever $D(\rho) > 0$. Defining the optimal linear JK advantage as $\Delta^*_{\mathrm{JK}}(\rho) := \mathcal{R}^*_{\mathrm{ST}}(\rho) - \mathcal{R}^*_{\mathrm{JK}}(\rho)$, we obtain the exact decomposition

$$\Delta^*_{\mathrm{JK}}(\rho) = D(\rho)\,\Gamma(\rho). \tag{13}$$

The two factors have distinct roles. The prediction deficit $D(\rho)$ quantifies how much clean target signal is inaccessible from the propagated representation, whereas the recoverability $\Gamma(\rho)$ quantifies what fraction of that deficit is linearly accessible through directions uniquely contributed by the retained pre-propagation representation. A large prediction deficit is therefore not sufficient for a large JK advantage: the inaccessible signal must also be recoverable from $H^{(0)}$.

Corollary 1. *Consider label noise rates $\rho_2 > \rho_1$ such that $D(\rho_i) > 0$ and $\Gamma(\rho_i) > 0$ for $i \in \{1, 2\}$.*

$$\Delta^*_{\mathrm{JK}}(\rho_2) > \Delta^*_{\mathrm{JK}}(\rho_1) \quad \Longleftrightarrow \quad \frac{D(\rho_2)}{D(\rho_1)} > \frac{\Gamma(\rho_1)}{\Gamma(\rho_2)}. \tag{14}$$

In particular, if $D(\rho_2) > D(\rho_1)$ and $\Gamma(\rho_2) \ge \Gamma(\rho_1) > 0$, then the optimal linear JK advantage strictly increases with the label noise rate.

Corollary 1 makes the role of label noise explicit. An increase in the prediction deficit does not by itself imply a larger JK advantage, since the residual target signal may simultaneously become less recoverable from $H^{(0)}$. Instead, the advantage grows when the relative increase in $D(\rho)$ exceeds any relative decrease in $\Gamma(\rho)$. In particular, if the prediction deficit increases while recoverability remains constant or increases, JK necessarily becomes more valuable as the noise rate grows.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate robustness to label noise in the node classification setting on four benchmarks with different levels of heterophily (see Table 4): `Roman Empire` (Platonov et al., 2023b), `Amazon-Ratings` (Platonov et al., 2023b), `BlogCatalog` and `Flickr` (Yang et al., 2023). Further details and summary statistics for all datasets are provided in Appendix B.

Baselines. We evaluate JK on 4 backbones: GCN (Kipf & Welling, 2017), GAT (Veličković et al., 2018), GraphSAGE (Hamilton et al., 2017), and GIN (Xu et al., 2018a). Where applicable, we follow the configurations of Luo et al. (2024), denoting the architectures as GCN*, GAT*, and SAGE*.

Noise mechanisms. We corrupt the training and validation labels of every architecture under three mechanisms, each at three corruption rates (0.15, 0.30 and 0.45):

- *Uniform*: replaces the true label with another random label, independently of the input.
- *Random*: introduces structured label corruption by preferentially flipping labels between specific pairs of classes, rather than uniformly across all incorrect classes.
- *Instance-dependent* (Instance): makes the probability of corruption depend on the individual node, so that some samples are more likely to receive an incorrect label than others.

Since validation labels are also corrupted, early stopping relies on noisy labels, whereas test accuracy is always computed against clean labels (Chen et al., 2021). Each configuration is run 3 times, and we report mean accuracy and standard deviation. Experiments are run on a machine with two NVIDIA RTX 4090 GPUs. Additional experimental details are in Appendix B. The code used to reproduce our experiments is provided in the supplementary material.

Table 1: Effect of JK on Accuracy under different label noise types and rates, averaged over three seeds. Best and second-best results are shown in **bold** and underlined.

Dataset	Noise Type	Noise Rate	GAT*		GCN*		GIN		SAGE*	
			JK	No JK	JK	No JK	JK	No JK	JK	No JK
BlogCatalog	Uniform	0.15	**0.944** ± 0.004	0.848 ± 0.016	0.940 ± 0.006	0.866 ± 0.015	0.919 ± 0.003	0.717 ± 0.006	0.903 ± 0.012	0.902 ± 0.014
		0.30	**0.903** ± 0.013	0.710 ± 0.038	0.852 ± 0.040	0.711 ± 0.013	0.829 ± 0.011	0.751 ± 0.048	0.795 ± 0.003	0.800 ± 0.011
		0.45	**0.859** ± 0.019	0.699 ± 0.011	0.681 ± 0.056	0.601 ± 0.040	0.704 ± 0.020	0.646 ± 0.011	0.642 ± 0.053	0.656 ± 0.036
	Random	0.15	**0.944** ± 0.006	0.839 ± 0.008	0.930 ± 0.010	0.829 ± 0.023	0.905 ± 0.007	0.762 ± 0.062	0.905 ± 0.017	0.901 ± 0.018
		0.30	**0.896** ± 0.015	0.718 ± 0.024	0.862 ± 0.043	0.698 ± 0.029	0.828 ± 0.011	0.666 ± 0.085	0.793 ± 0.034	0.783 ± 0.022
		0.45	**0.834** ± 0.048	0.651 ± 0.020	0.727 ± 0.032	0.559 ± 0.021	0.699 ± 0.016	0.610 ± 0.044	0.639 ± 0.035	0.645 ± 0.040
	Instance	0.15	**0.935** ± 0.010	0.827 ± 0.025	0.909 ± 0.037	0.849 ± 0.007	0.891 ± 0.015	0.763 ± 0.027	0.882 ± 0.026	0.883 ± 0.023
		0.30	**0.885** ± 0.009	0.671 ± 0.041	0.837 ± 0.047	0.711 ± 0.077	0.821 ± 0.024	0.688 ± 0.034	0.787 ± 0.033	0.788 ± 0.015
		0.45	**0.747** ± 0.124	0.667 ± 0.023	0.633 ± 0.016	0.551 ± 0.046	0.628 ± 0.041	0.680 ± 0.068	0.623 ± 0.021	0.633 ± 0.006
Roman Empire	Uniform	0.15	0.819 ± 0.002	0.799 ± 0.003	**0.824** ± 0.002	0.799 ± 0.003	0.799 ± 0.003	0.766 ± 0.002	0.818 ± 0.003	0.820 ± 0.000
		0.30	0.784 ± 0.004	0.749 ± 0.004	**0.792** ± 0.003	0.764 ± 0.001	0.772 ± 0.001	0.768 ± 0.006	0.779 ± 0.005	0.782 ± 0.003
		0.45	0.730 ± 0.006	0.690 ± 0.006	0.753 ± 0.004	0.709 ± 0.007	0.741 ± 0.002	0.724 ± 0.002	0.748 ± 0.009	**0.756** ± 0.004
	Random	0.15	**0.823** ± 0.002	0.815 ± 0.004	0.822 ± 0.003	0.813 ± 0.002	0.805 ± 0.006	0.795 ± 0.005	0.813 ± 0.003	0.821 ± 0.004
		0.30	0.789 ± 0.004	0.769 ± 0.005	0.791 ± 0.003	0.778 ± 0.003	0.771 ± 0.004	0.767 ± 0.006	0.786 ± 0.002	**0.792** ± 0.000
		0.45	0.744 ± 0.003	0.725 ± 0.005	0.749 ± 0.009	0.722 ± 0.008	0.737 ± 0.007	0.734 ± 0.009	0.739 ± 0.003	**0.751** ± 0.002
	Instance	0.15	0.812 ± 0.010	0.807 ± 0.006	**0.819** ± 0.002	0.808 ± 0.003	0.795 ± 0.001	0.757 ± 0.003	0.811 ± 0.002	0.817 ± 0.001
		0.30	0.776 ± 0.009	0.757 ± 0.006	0.781 ± 0.005	0.768 ± 0.009	0.764 ± 0.005	0.756 ± 0.006	0.778 ± 0.006	**0.782** ± 0.002
		0.45	0.709 ± 0.018	0.693 ± 0.015	0.719 ± 0.015	0.690 ± 0.010	**0.724** ± 0.011	0.714 ± 0.010	0.723 ± 0.013	0.724 ± 0.005
Amazon-Ratings	Uniform	0.15	**0.480** ± 0.003	0.469 ± 0.001	0.465 ± 0.001	0.461 ± 0.004	0.444 ± 0.019	0.427 ± 0.018	0.478 ± 0.007	0.478 ± 0.004
		0.30	0.439 ± 0.009	**0.441** ± 0.011	0.434 ± 0.009	0.428 ± 0.010	0.411 ± 0.018	0.402 ± 0.010	0.437 ± 0.017	0.436 ± 0.010
		0.45	0.413 ± 0.008	0.412 ± 0.007	0.397 ± 0.012	0.401 ± 0.006	0.363 ± 0.009	0.358 ± 0.016	**0.417** ± 0.001	0.407 ± 0.006
	Random	0.15	0.475 ± 0.003	0.478 ± 0.007	0.462 ± 0.006	0.460 ± 0.012	0.439 ± 0.008	0.436 ± 0.007	**0.490** ± 0.001	0.487 ± 0.002
		0.30	0.434 ± 0.009	**0.436** ± 0.004	0.427 ± 0.006	0.432 ± 0.009	0.402 ± 0.022	0.407 ± 0.009	0.433 ± 0.010	0.432 ± 0.006
		0.45	**0.410** ± 0.008	0.404 ± 0.003	0.408 ± 0.005	0.408 ± 0.014	0.363 ± 0.017	0.350 ± 0.021	0.406 ± 0.005	0.403 ± 0.006
	Instance	0.15	0.473 ± 0.003	0.478 ± 0.002	0.464 ± 0.005	0.462 ± 0.003	0.432 ± 0.012	0.440 ± 0.017	**0.486** ± 0.001	0.482 ± 0.002
		0.30	0.436 ± 0.004	0.432 ± 0.005	0.438 ± 0.003	**0.445** ± 0.008	0.393 ± 0.022	0.389 ± 0.028	0.441 ± 0.010	0.435 ± 0.011
		0.45	**0.415** ± 0.005	0.407 ± 0.010	0.413 ± 0.009	0.401 ± 0.009	0.367 ± 0.004	0.365 ± 0.003	0.413 ± 0.007	0.398 ± 0.009
Flickr	Uniform	0.15	0.834 ± 0.020	0.806 ± 0.018	0.809 ± 0.021	0.747 ± 0.005	**0.854** ± 0.016	0.831 ± 0.010	0.848 ± 0.012	0.843 ± 0.017
		0.30	0.747 ± 0.032	0.660 ± 0.023	0.680 ± 0.018	0.563 ± 0.008	**0.785** ± 0.025	0.708 ± 0.020	0.734 ± 0.006	0.731 ± 0.008
		0.45	0.627 ± 0.039	0.501 ± 0.025	0.495 ± 0.022	0.337 ± 0.008	**0.705** ± 0.036	0.575 ± 0.008	0.606 ± 0.007	0.624 ± 0.014
	Random	0.15	**0.854** ± 0.013	0.799 ± 0.001	0.805 ± 0.013	0.756 ± 0.011	0.843 ± 0.009	0.825 ± 0.013	0.849 ± 0.003	0.845 ± 0.011
		0.30	0.743 ± 0.033	0.662 ± 0.021	0.678 ± 0.030	0.511 ± 0.092	**0.786** ± 0.009	0.714 ± 0.011	0.736 ± 0.022	0.744 ± 0.021
		0.45	0.637 ± 0.008	0.499 ± 0.017	0.501 ± 0.025	0.382 ± 0.032	**0.705** ± 0.019	0.577 ± 0.009	0.623 ± 0.021	0.615 ± 0.010
	Instance	0.15	0.828 ± 0.016	0.792 ± 0.020	0.788 ± 0.014	0.724 ± 0.026	**0.847** ± 0.011	0.814 ± 0.008	0.845 ± 0.011	0.843 ± 0.008
		0.30	0.726 ± 0.023	0.665 ± 0.010	0.683 ± 0.014	0.541 ± 0.051	**0.790** ± 0.015	0.701 ± 0.010	0.745 ± 0.006	0.740 ± 0.005
		0.45	0.638 ± 0.025	0.533 ± 0.036	0.522 ± 0.009	0.373 ± 0.010	**0.685** ± 0.011	0.565 ± 0.002	0.609 ± 0.008	0.595 ± 0.013

Table 2: Relative accuracy gain from Jumping Knowledge, computed as (JK − No JK) / No JK. Each entry averages the three noise types of Table 1.

Dataset	Noise Rate	GAT*	GCN*	GIN	SAGE*
BlogCatalog	0.15	+12.3%	+9.3%	+21.2%	+0.1%
	0.30	+28.0%	+20.3%	+18.0%	+0.2%
	0.45	+21.0%	+19.4%	+5.3%	−1.5%
	Average	+20.4%	+16.3%	+14.9%	−0.4%
Roman Empire	0.15	+1.4%	+1.9%	+3.5%	−0.7%
	0.30	+3.3%	+2.3%	+0.7%	−0.6%
	0.45	+3.6%	+4.7%	+1.4%	−0.9%
	Average	+2.7%	+3.0%	+1.9%	−0.7%
Amazon-Ratings	0.15	+0.2%	+0.6%	+1.0%	+0.5%
	0.30	+0.0%	−0.4%	+0.7%	+0.6%
	0.45	+1.2%	+0.7%	+1.9%	+2.3%
	Average	+0.5%	+0.3%	+1.2%	+1.1%
Flickr	0.15	+5.0%	+7.9%	+3.0%	+0.4%
	0.30	+11.5%	+26.6%	+11.2%	+0.0%
	0.45	+24.2%	+39.3%	+22.0%	+0.3%
	Average	+13.6%	+24.6%	+12.1%	+0.2%

4.2 RESULTS

Table 1 reports the effect of JK across architectures, datasets, noise mechanisms, and corruption levels, while Table 2 summarizes the corresponding relative gains over the default architectures. Over-

Table 3: Representation-level analysis of the information contributed by $H^{(0)}$ alongside the propagated representation $H^{(T)}$.

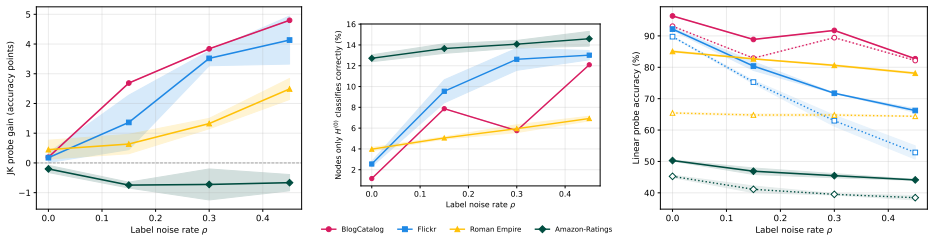
(a) Probe gain and node-level complementarity (GCN*, uniform noise).

	Dataset	$\rho = 0$	$\rho = 0.15$	$\rho = 0.30$	$\rho = 0.45$
G_{probe}	BlogCatalog	+0.19	+2.69	+3.84	+4.80
	Roman Empire	+0.45	+0.64	+1.32	+2.49
	Amazon-Ratings	−0.20	−0.75	−0.72	−0.66
	Flickr	+0.18	+1.36	+3.52	+4.13
$H^{(0)}$ only	BlogCatalog	1.2	7.9	5.8	12.1
	Roman Empire	4.0	5.1	5.9	6.9
	Amazon-Ratings	12.7	13.7	14.1	14.6
	Flickr	2.6	9.5	12.6	13.0

(b) Prediction deficit D, recoverability Γ and optimal linear advantage $\Delta^*_{\text{JK}} = D\Gamma$ across corruption rates.

Dataset	ρ	Γ	D	Δ^*_{JK}
BlogCatalog	0	0.971	0.110	0.106
	0.15	0.976	0.132	0.129
	0.30	0.974	0.117	0.114
	0.45	0.979	0.159	0.155
Roman Empire	0	0.302	0.350	0.105
	0.15	0.307	0.379	0.116
	0.30	0.308	0.396	0.122
	0.45	0.313	0.415	0.130
Amazon-Ratings	0	0.104	0.556	0.058
	0.15	0.102	0.577	0.059
	0.30	0.105	0.591	0.062
	0.45	0.103	0.600	0.062
Flickr	0	0.589	0.206	0.121
	0.15	0.548	0.255	0.140
	0.30	0.555	0.307	0.170
	0.45	0.552	0.355	0.196

Figure 1: Representation-level analysis of JK under uniform label noise for GCN*. (a) The probe gain $G_{\text{probe}}(\rho)$ is small under clean supervision and increases with the corruption rate on three of the four benchmarks. (b) The percentage of nodes classified correctly only by the $H^{(0)}$ probe is larger at high corruption than under clean supervision, although the trajectory is not always monotonic. (c) The $H^{(T)}$ probe (solid) remains more accurate than the $H^{(0)}$ probe (dotted) throughout.



(a) JK gain emerges with corruption.

(b) Node-level complementarity.

(c) $H^{(T)}$ (solid) vs. $H^{(0)}$ dotted.

all, JK improves relative accuracy by 7.0% on average. This aggregate result, however, conceals variation across backbones: GCN*, GAT*, and GIN gain +11.0%, +9.3%, and +7.5% on average, respectively, whereas SAGE* gains only +0.1%. This split is consistent with differences in how each architecture preserves node-specific information during propagation. Although GCN*, GAT*, and GIN include self-information in their updates, none directly exposes the pre-propagation representation $H^{(0)}$ to the final readout. JK provides this direct pathway explicitly. SAGE* differs in that it keeps the central node representation and the aggregated neighborhood as separate inputs to the layer transformation, which may already preserve part of the node-specific information that JK would otherwise contribute. This is consistent with the small additional benefit of exposing $H^{(0)}$ at the final readout.

Theorem 1 provides a structural interpretation: the additional pathway can enlarge the predictor class when $H^{(0)}$ retains task-relevant distinctions that are inaccessible from $H^{(T)}$. The four backbones differ in how node-specific information survives propagation. SAGE* keeps the central node representation as a separate block of its layer input (Eq. (2)), allowing self and neighborhood information to be transformed separately at every depth. The small JK gains observed for SAGE* are consistent with this distinction: its internal self pathway may already preserve much of the node-specific information that JK would otherwise contribute at readout.

4.2.1 Analysis of the Impact of Jumping Knowledge in the Noisy Setting

To better understand the impact of constructing the architecture with the Jumping Knowledge mechanism, we examine how label corruption changes the predictive information retained in $H^{(0)}$ relative to $H^{(T)}$. We freeze the representations learned by GCN* under uniform noise, fit linear probes on $H^{(0)}$, on $H^{(T)}$, and on their concatenation, and evaluate all probes on the clean test labels.

Propagation preserves a strong representation. We first assess the predictive capability of $H^{(0)}$ and $H^{(T)}$ under noise. Figure 1c shows that the $H^{(T)}$ probe remains more accurate than the $H^{(0)}$ probe at every corruption rate and on every dataset, with a gap of 0.6–19.6 accuracy points. Therefore, any benefit from retaining $H^{(0)}$ does not come from overtaking $H^{(T)}$. Whether it instead carries information that $H^{(T)}$ does not is the question we address next.

Corruption increases node-level complementarity. We analyze whether $H^{(0)}$ preserves useful information precisely where $H^{(T)}$ fails. For each node, we measure whether it is correctly classified by the $H^{(0)}$ probe but not by the $H^{(T)}$ probe. As shown in Fig. 1b, this fraction is larger at $\rho = 0.45$ than under clean supervision on all datasets, increasing from 2.6% to 13.0% on Flickr, 1.2% to 12.1% on BlogCatalog and 12.7% to 14.6% on Amazon-Ratings. Thus, increasing corruption leaves more nodes for which useful predictive information is available in $H^{(0)}$ but not in $H^{(T)}$. This is the node-level complementarity that retaining both representations can potentially exploit.

The probe gain increases under corruption. We test whether the complementary information identified above can improve a joint predictor. We define the representation-level probe gain as

$$G_{\text{probe}}(\rho) := \text{Acc}\left([H^{(0)} \| H^{(T)}]\right) - \text{Acc}\left(H^{(T)}\right), \tag{15}$$

It measures the benefit of exposing both representations to the same readout. Since they are frozen, we isolate the information available in their representation from end-to-end training effects.

As shown by Table 3a and Fig. 1a, this benefit is negligible in the absence of label noise, but grows as the noise increases, reaching $+4.80$ points on BlogCatalog and $+4.13$ on Flickr. This indicates that, as supervision becomes noisier, $H^{(0)}$ increasingly provides information that complements rather than replaces $H^{(T)}$. Amazon-Ratings is the important counterexample: despite having many nodes correctly classified only from $H^{(0)}$, its probe gain remains negative. Node-level complementarity is not sufficient, since the additional information must be exploitable by a single readout.

Recoverability explains when JK is useful. Table 3b shows that a large prediction deficit alone does not imply a large JK gain, instantiating the case anticipated by Corollary 1. At every corruption rate, $D(\rho)$ is largest on Amazon-Ratings, where the empirical gain is negligible, and smallest on BlogCatalog, where the probe gain under corruption is largest. The key difference is recoverability: $\Gamma(\rho)$ stays around 0.97 on BlogCatalog and 0.55 on Flickr, but falls to 0.31 on Roman Empire and 0.10 on Amazon-Ratings, an ordering that matches that of the observed probe gains at every corrupted noise level ($\rho > 0$). JK therefore helps when the information missing from $H^{(T)}$ remains accessible through $H^{(0)}$, not simply when propagation loses more information. The two factors also respond differently to corruption. Between $\rho = 0$ and $\rho = 0.45$, $D(\rho)$ increases on every dataset, by 8–72% (non-monotonically on BlogCatalog), whereas $\Gamma(\rho)$ varies by at most 0.041 (Flickr) and by at most 0.011 elsewhere: label noise enlarges the prediction deficit without substantially reorienting it relative to the subspace contributed by $H^{(0)}$. Consequently, the growth condition of Corollary 1 holds between $\rho = 0$ and $\rho = 0.45$ on all four benchmarks, and Δ^*_{JK} increases accordingly.

5 Conclusion

In this work, we studied Jumping Knowledge as a representation-preservation mechanism for robustness to noisy supervision in GNNs. Retaining the pre-propagation representation $H^{(0)}$ alongside $H^{(T)}$ preserves access to information that message passing may make inaccessible, and its optimal linear value factorizes into prediction deficit and recoverability. Label noise enlarges the prediction deficit while leaving recoverability stable. JK helps when the lost signal is recoverable from $H^{(0)}$.

Future work. Promising directions include adaptive preservation mechanisms and extensions to deeper networks, larger graphs, and other noise models.

AI USE STATEMENT

In this work, we used generative AI tools to interpret results and to assist in writing proofs.

We did not use generative AI tools to generate synthetic datasets, develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, assist with translation, clean or reformat datasets, or support qualitative and thematic data analysis.

We additionally used generative AI tools to create or edit software code, suggest a structure for the paper, propose a title or keywords, discover research topics or identify gaps, brainstorm, search for information, and edit the paper to improve readability.

We have reviewed all AI-assisted work. We checked LLM-generated research ideas for potential plagiarism through a manual literature survey, and LLM-generated code was verified and tested for correctness.

ETHICS STATEMENT

This work presents a methodological study on learning with noisy labels in the context of graph neural networks. Our research is purely computational and does not involve human subjects, personal data, or sensitive information. The datasets used in our experiments are publicly available benchmarks commonly adopted in the graph learning community. We do not foresee any direct negative societal impact arising from this work. No conflicts of interest or external sponsorship influenced the design or outcomes of this study.

REPRODUCIBILITY STATEMENT

To ensure reproducibility, we release the full source code of our method and experiments as part of the Supplementary Material. Detailed descriptions of the experimental setup, including hyperparameter configurations, dataset preprocessing steps, and evaluation protocols, are provided in Section 4.1 and Appendix B.

REFERENCES

Ling-Hao Chen, Yuanshuo Zhang, Taohua Huang, Liangcai Su, Zeyi Lin, Xi Xiao, Xiaobo Xia, and Tongliang Liu. Erase: Error-resilient representation learning on graphs for label noise tolerance. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pp. 270–280, 2024.

Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *International conference on machine learning*, pp. 1725–1735. PMLR, 2020.

Pengfei Chen, Junjie Ye, Guangyong Chen, Jingwei Zhao, and Pheng-Ann Heng. Robustness of accuracy metric and its inspirations in learning with noisy labels. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 11451–11461, 2021.

Gene Cheung, Weng-Tai Su, Yu Mao, and Chia-Wen Lin. Robust semisupervised graph classifier learning with negative edge weights. *IEEE Transactions on Signal and Information Processing over Networks*, 4(4):712–726, 2018.

Enyan Dai, Charu Aggarwal, and Suhang Wang. Nrgnn: Learning a label noise resistant graph neural network on sparsely and noisily labeled graphs. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pp. 227–236, 2021.

Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. *arXiv preprint arXiv:1810.05997*, 2018.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30:1–11, 2017.

Thomas. N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, 2017.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Yayong Li, Jie Yin, and Ling Chen. Unified robust training for graph neural networks against label noise. In *Pacific-Asia conference on knowledge discovery and data mining*, pp. 528–540. Springer, 2021.

Yuankai Luo, Lei Shi, and Xiao-Ming Wu. Classic gnns are strong baselines: Reassessing gnns for node classification. *Advances in Neural Information Processing Systems*, 37:97650–97669, 2024.

Oleg Platonov, Denis Kuznedelev, Artem Babenko, and Liudmila Prokhorenkova. Characterizing graph datasets for node classification: Homophily-heterophily dichotomy and beyond. *Advances in Neural Information Processing Systems*, 36:523–548, 2023a.

Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of gnns under heterophily: Are we really making progress? In *The Eleventh International Conference on Learning Representations*, 2023b.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

Zhonghao Wang, Danyu Sun, Sheng Zhou, Haobo Wang, Jiapei Fan, Longtao Huang, and Jiajun Bu. Noisygl: A comprehensive benchmark for graph neural networks under label noise. *Advances in neural information processing systems*, 37:38142–38170, 2024.

Farooq Ahmad Wani, Maria Sofia Bucarelli, and Fabrizio Silvestri. Learning with noisy labels through learnable weighting and centroid similarity. In *2024 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–9. IEEE, 2024.

Yuhao Wu, Jiangchao Yao, Xiaobo Xia, Jun Yu, Ruxin Wang, Bo Han, and Tongliang Liu. Mitigating label noise on graphs via topological sample selection. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53944–53972. PMLR, 21–27 Jul 2024. URL `https://proceedings.mlr.press/v235/wu24ae.html`.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, 2020.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018a.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5453–5462, 2018b.

Renchi Yang, Jieming Shi, Xiaokui Xiao, Yin Yang, Sourav S Bhowmick, and Juncheng Liu. Pane: scalable and effective attributed network embedding. *The VLDB Journal*, 32(6):1237–1262, 2023.

Jingyang Yuan, Xiao Luo, Yifang Qin, Yusheng Zhao, Wei Ju, and Ming Zhang. Learning on graphs under label noise. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2023.

Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. *Advances in neural information processing systems*, 33:7793–7804, 2020.

A THEORETICAL RESULTS

In this section, we provide the complete statements and proofs of the theoretical results presented in the main paper.

Property 1. *For any propagation backbone Φ:*

$$H^{(0)} \neq \widetilde{H}^{(0)} \quad \Longrightarrow \quad \Psi_{\mathrm{JK}}(H^{(0)}) \neq \Psi_{\mathrm{JK}}(\widetilde{H}^{(0)}).$$

Proof. By definition:

$$\Psi_{\mathrm{JK}}(H^{(0)}) = [H^{(0)} \| \Phi(H^{(0)}; G)] \qquad \Psi_{\mathrm{JK}}(\widetilde{H}^{(0)}) = [\widetilde{H}^{(0)} \| \Phi(\widetilde{H}^{(0)}; G)].$$

Since $H^{(0)} \neq \widetilde{H}^{(0)}$, the first blocks of the two concatenated representations differ. Therefore:

$$\Psi_{\mathrm{JK}}(H^{(0)}) \neq \Psi_{\mathrm{JK}}(\widetilde{H}^{(0)}),$$

independently of the behavior of Φ. □

Property 2. *For a fixed encoder, propagation backbone, and prediction-head family, every predictor realizable by the standard architecture is also realizable by the JK architecture. Hence:*

$$\mathcal{H}_{\mathrm{ST}} \subseteq \mathcal{H}_{\mathrm{JK}}.$$

Proof. Consider any predictor in $\mathcal{H}_{\mathrm{ST}}$. Its first affine transformation has the form:

$$a_v^{\mathrm{ST}} = A_T h_v^{(T)} + b.$$

The JK architecture can reproduce the same transformation by setting $A_0 = 0$. Then:

$$a_v^{\mathrm{JK}} = 0 \cdot h_v^{(0)} + A_T h_v^{(T)} + b = a_v^{\mathrm{ST}}.$$

Since the remainder of the prediction-head family is unchanged, every predictor realizable by the standard architecture is also realizable by the JK architecture. Therefore:

$$\mathcal{H}_{\mathrm{ST}} \subseteq \mathcal{H}_{\mathrm{JK}}.$$

□

Theorem 1. *Fix the encoder and propagation backbone. Suppose there exist two nodes $u \neq w$ such that*

$$h_u^{(T)} = h_w^{(T)} \qquad \textit{and} \qquad h_u^{(0)} \neq h_w^{(0)}.$$

If the JK head family contains a predictor that separates the corresponding JK representations, then

$$\mathcal{H}_{\mathrm{ST}} \subsetneq \mathcal{H}_{\mathrm{JK}}.$$

Proof. By Property 2, $\mathcal{H}_{\mathrm{ST}} \subseteq \mathcal{H}_{\mathrm{JK}}$; it remains to show that the inclusion is strict. Let $u \neq w$ satisfy $h_u^{(T)} = h_w^{(T)}$ and $h_u^{(0)} \neq h_w^{(0)}$. Every predictor $g \in \mathcal{H}_{\mathrm{ST}}$ depends on node v only through $h_v^{(T)}$, hence $g(u) = g(w)$. The JK representations $[h_u^{(0)} \| h_u^{(T)}]$ and $[h_w^{(0)} \| h_w^{(T)}]$ differ in their first block and, by assumption, the JK head family contains a predictor $g^\star$ with $g^\star(u) \neq g^\star(w)$. Thus $g^\star \notin \mathcal{H}_{\mathrm{ST}}$, and $\mathcal{H}_{\mathrm{ST}} \subsetneq \mathcal{H}_{\mathrm{JK}}$. □

Proposition 1. *Let $\mathcal{R}^*_{\mathrm{JK}}(\rho)$ denote the optimal least-squares risk obtained from the JK representation $[H^{(0)} \,\|\, H^{(T)}]$. Then*

$$\mathcal{R}^*_{\mathrm{ST}}(\rho) - \mathcal{R}^*_{\mathrm{JK}}(\rho) = \frac{1}{n} \Big\| Q(I - P_T) Y \Big\|_F^2.$$

Proof. Let $r = (I - P_T)Y$ be the residual target signal after projection onto $\mathrm{col}(H^{(T)})$. By Eq. (10), $\mathrm{col}(\widetilde{H}^{(0)}) \perp \mathrm{col}(H^{(T)})$, so the orthogonal projector onto $\mathrm{col}([H^{(0)} \| H^{(T)}])$ is $P_T + Q$, with $QP_T = 0$. Hence

$$\mathcal{R}^*_{\mathrm{JK}}(\rho) = \frac{1}{n} \big\| (I - P_T - Q) Y \big\|_F^2 = \frac{1}{n} \big\| (I - Q) r \big\|_F^2.$$

Since Q is an orthogonal projector, $\|r\|_F^2 = \|Qr\|_F^2 + \|(I - Q)r\|_F^2$. Subtracting from $\mathcal{R}^*_{\mathrm{ST}}(\rho) = \frac{1}{n}\|r\|_F^2$ yields Eq. (11). □

Corollary 1. *Consider two label noise rates $\rho_2 > \rho_1$ such that $D(\rho_i) > 0$ and $\Gamma(\rho_i) > 0$ for $i \in \{1, 2\}$. Then*

$$\Delta^*_{\mathrm{JK}}(\rho_2) > \Delta^*_{\mathrm{JK}}(\rho_1) \quad \Longleftrightarrow \quad \frac{D(\rho_2)}{D(\rho_1)} > \frac{\Gamma(\rho_1)}{\Gamma(\rho_2)}.$$

In particular, if $D(\rho_2) > D(\rho_1)$ and $\Gamma(\rho_2) \geq \Gamma(\rho_1) > 0$, then the optimal linear JK advantage strictly increases with the label noise rate.

Proof. By Eq. (13),

$$\Delta^*_{\mathrm{JK}}(\rho_i) = D(\rho_i)\Gamma(\rho_i), \qquad i \in \{1, 2\}.$$

Since $D(\rho_i), \Gamma(\rho_i) > 0$,

$$\Delta^*_{\mathrm{JK}}(\rho_2) > \Delta^*_{\mathrm{JK}}(\rho_1) \iff \frac{D(\rho_2)}{D(\rho_1)} > \frac{\Gamma(\rho_1)}{\Gamma(\rho_2)}.$$

The final statement follows directly from $D(\rho_2) > D(\rho_1)$ and $\Gamma(\rho_2) \geq \Gamma(\rho_1) > 0$. □

B Additional Information on the Experimental Setup

Table 4: Summary statistics for the datasets, reporting the number of nodes, edges, features, classes and two types of homophily.

Dataset	**#nodes**	**#edges**	**#features**	**#classes**	**Edge Homophily**	**Adjusted Homophily**
BlogCatalog	5,196	343,486	8,189	6	0.40	0.27
Roman-Empire	22,662	32,927	300	18	0.05	-0.05
Amazon-Ratings	24,492	93,050	300	5	0.38	0.14
Flickr	7,575	479,476	12,047	9	0.24	0.09

Table 5: Model configurations for GCN*, GAT*, GIN and SAGE* across datasets.

Dataset	**Model**	**Learned Residual**	**Hidden dim**	**#layers**	**drop_in**	**dropout**	**epochs**	**patience**
BlogCatalog	GCN*	True	256	6	0.4	0.6	3500	500
	GAT*	True	256	6	0.4	0.6	3500	500
	GIN	True	256	4	–	0.9	2500	2500
	SAGE*	True	256	6	0.4	0.6	3500	500
Roman Empire	GCN*	True	512	10	0.6	0.5	3500	1000
	GAT*	True	512	10	0.6	0.5	3500	1000
	GIN	False	256	9	–	0.3	2500	2500
	SAGE*	False	256	9	0.2	0.3	2500	2500
Amazon-Ratings	GCN*	True	512	4	0.2	0.5	3500	500
	GAT*	True	512	4	0.2	0.5	3500	500
	GIN	True	512	9	–	0.5	2500	2500
	SAGE*	True	512	9	0.2	0.5	2500	2500
Flickr	GCN*	True	256	4	0.4	0.6	3500	500
	GAT*	True	256	4	0.4	0.6	3500	500
	GIN	True	256	4	–	0.975	2500	2500
	SAGE*	True	256	4	0.4	0.6	3500	500

Table 6: Inference computational cost (in GFLOPs) for each model across all datasets (↓ is better). Best and second-best results are shown in **bold** and underlined.

Dataset	**GCN***	**GAT***	**SAGE***	**GIN**
BlogCatalog	**27.958**	<u>29.546</u>	32.037	30.681
Roman Empire	281.005	282.131	<u>66.173</u>	**60.388**
Amazon-Ratings	**149.054**	<u>150.246</u>	277.537	251.866
Flickr	**53.733**	<u>55.212</u>	57.699	55.738

B.1 Dataset Description

Table 4 summarizes the main characteristics of the datasets used in our experiments. The four benchmarks differ substantially in graph size, feature dimensionality, and number of classes, providing diverse settings for evaluating robustness to noisy supervision. We also measure edge homophily and adjusted homophily (Platonov et al., 2023a). Edge homophily measures the fraction of edges in a graph that connect nodes sharing the same class label. Given a graph $G = (V, E)$ with node labels y_v for each $v \in V$, it is defined as:

$$h_{\text{edge}} = \frac{|\{(u, v) \in E : y_u = y_v\}|}{|E|}$$

The value lies in $[0, 1]$, with $h_{\text{edge}} \to 1$ indicating homophily and $h_{\text{edge}} \to 0$ indicating heterophily. However, h_{edge} is sensitive to class imbalance: a dominant class can inflate the score even under random wiring.

To correct for this bias, adjusted homophily subtracts the expected homophily under a degree-preserving random graph and normalizes:

$$h_{\text{adj}} = \frac{h_{\text{edge}} - \sum_{k=1}^{C} \bar{p}(k)^2}{1 - \sum_{k=1}^{C} \bar{p}(k)^2} \qquad \bar{p}(k) = \frac{\sum_{v\,:\,y_v=k} d(v)}{2|E|}$$

Where C is the number of classes, $\bar{p}(k)$ is the empirical degree-weighted class distribution and $d(v)$ the degree of node v.

Intuitively, $h_{\text{adj}} > 0$ indicates homophily beyond what chance alone would produce, $h_{\text{adj}} = 0$ matches random expectation, and $h_{\text{adj}} < 0$ signals heterophily. The larger the magnitude, the stronger the effect. For data splitting we use the standard PyTorch Geometric[1] splits where available, and otherwise adopt a $80/10/10$ train/validation/test partition.

B.2 Experimental Setup

Table 5 lists the architecture and training configuration of the four backbones on each dataset. For all datasets, hidden representations are normalized with batch normalization, optimization uses a learning rate of 0.001 with weight decay 0.0. To ensure a fair comparison and prevent trivial baseline failures under heavy label corruption, hyperparameters were tuned per backbone. In particular, for GIN on BlogCatalog and Flickr, the base (No JK) model required adjusting dropout and disabling layer-wise learned residuals to stabilize convergence, as the default configuration led to severe degradation in the absence of the pre-propagation pathway. Label noise is injected into the training and validation sets only; the test set is kept clean, so that reported accuracy measures generalization to uncorrupted labels (Chen et al., 2021).

B.3 Inference Cost Analysis

Table 6 reports the inference cost of each backbone across the four datasets. GCN* and GAT* have the lowest computational cost on Amazon-Ratings, BlogCatalog, and Flickr, with only small differences between them. Roman Empire exhibits a different pattern, where GIN and SAGE* require substantially fewer FLOPs than GCN* and GAT*. Overall, no architecture is uniformly the most efficient across datasets, highlighting that inference cost depends on both the propagation mechanism and the characteristics of the underlying graph.

[1] https://pytorch-geometric.readthedocs.io/en/latest/

Figure 2: Overview of the graph propagation architectures considered in this work.

