Learning from Graph: Mitigating Label Noise on Graph through Topological Feature Reconstruction

Zhonghao Wang
Zhejiang University
Hangzhou, China
wangzhonghao@zju.edu.cn

Yuanchen Bei
Zhejiang University
Hangzhou, China
yuanchenbei@zju.edu.cn

Sheng Zhou*
Zhejiang University
Hangzhou, China
zhousheng_zju@zju.edu.cn

Zhiyao Zhou
Zhejiang University
Hangzhou, China
zjucszzy@zju.edu.cn

Jiapei Fan
Alibaba Group
Hangzhou, China
jiapei.fjp@alibaba-inc.com

Hui Xue
Alibaba Group
Hangzhou, China
hui.xueh@alibaba-inc.com

Haishuai Wang
Zhejiang University
Hangzhou, China
haishuai.wang@zju.edu.cn

Jiajun Bu
Zhejiang University
Hangzhou, China
bjj@zju.edu.cn

Abstract

Graph Neural Networks (GNNs) have shown remarkable performance in modeling graph data. However, Labeling graph data typically relies on unreliable information, leading to noisy node labels. Existing approaches for GNNs under Label Noise (GLN) employ supervision signals beyond noisy labels for robust learning. While empirically effective, they tend to over-reliance on supervision signals built upon external assumptions, leading to restricted applicability. In this work, we shift the focus to exploring how to extract useful information and learn from the graph itself, thus achieving robust graph learning. We analyze the impact of graph label noise from an information theory perspective. We theoretically and empirically demonstrate that the graph itself contains reliable information for graph learning under label noise. Based on these insights, we propose the Topological Feature Reconstruction (TFR) method. Specifically, TFR leveraging the fact that the pattern of clean labels can more accurately reconstruct graph features through topology, while noisy labels cannot. TFR is a simple and theoretically guaranteed model for robust graph learning under label noise. We conduct extensive experiments across datasets with varying properties to demonstrate the robustness and broad applicability of our proposed TFR compared to state-of-the-art baselines. Experimental results demonstrate that TFR effectively mitigates graph label noise. Codes are available at https://github.com/eaglelab-zju/TFR.

*Sheng Zhou is the corresponding author.

CIKM '25, Seoul, Republic of Korea.

ACM ISBN 979-8-4007-2040-6/2025/11
https://doi.org/10.1145/3746252.3761185

CCS Concepts

• **Computing methodologies** → *Neural networks*; • **Information systems** → **Data cleaning**; *Social networks*; • **Theory of computation** → **Semi-supervised learning**.

Keywords

Graph Neural Networks; Label Noise; Graph reconstruction; Semi-supervised learning

ACM Reference Format:
Zhonghao Wang, Yuanchen Bei, Sheng Zhou, Zhiyao Zhou, Jiapei Fan, Hui Xue, Haishuai Wang, and Jiajun Bu. 2025. Learning from Graph: Mitigating Label Noise on Graph through Topological Feature Reconstruction. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM '25), November 10–14, 2025, Seoul, Republic of Korea.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3746252.3761185

1 Introduction

Many complex systems in the real world, such as biological interaction networks [15], social networks [17], knowledge graphs [5], and traffic networks [14], can be represented as graph data. Over the past decade, research has demonstrated the effectiveness of Graph Neural Networks (GNNs) in modeling graph data. Based on the message-passing mechanism, GNNs learn node representation in a semi-supervised manner by aggregating supervision signals from neighboring nodes and achieve impressive performance [16]. However, graph data are intricate and large-scale in practical applications, making the delicate labeling of each node both costly and impractical. As a result, labeling graph data, especially for node-level tasks, often relies on unreliable contributions [24], and this non-expert labeling can lead to node labels that are both *scarce* and *noisy* [7]. Previous works indicate that label noise can be extremely detrimental to graph learning [7, 24, 29, 43], for the disruptive impact of label noise can propagate along the graph topology [3] through the message-passing mechanism, causing significant damage to the model's performance on clean data [29].

To achieve robust graph learning under label noise, earlier attempts [20] have employed previously well-studied Learning with Label Noise (LLN) [10, 25, 28, 33, 34] strategies designed for independently and identically distributed (i.i.d.) data, on GNNs. However, due to the message-passing mechanism of GNNs and the label sparsity, non-i.i.d. nature of graph data, directly applying these LLN methods to GNNs on graph data does not achieve promising results against label noise [7, 29]. Recent studies on Graph Neural Networks under Label Noise (GLN) have focused on addressing the problem of graph label noise while considering the role of graph data. These works turn to identify appropriate external supervision signals from other observed graph data (e.g., topology and attributes) to guide the graph learning process explicitly [19, 30, 40] or implicitly [7, 24]. These supervision signals can be obtained from community labels [40], neighboring labels [9, 19, 30, 31, 39], and node representations [7, 24, 43] among neighbors or different graph views [18, 39]. By obtaining these external supervision signals from topology and features, these methods have achieved success in their respective domains.

Although these approaches have demonstrated empirical effectiveness in several scenarios, their design often relies on external assumptions. For instance, they may stipulate that accurate label prediction must align with community labels [40] or neighboring labels [9], or assume nodes with similar representations can provide reliable supervision signals [7, 24]. Thereby limiting their general applicability [29]. To tackle this issue, we shift the focus from external supervision signals and exploring how graph data itself can provide reliable information for robust graph learning. We start from an information theory perspective and explore the role of graph information under label noise. Through our theoretical analysis, we discover that a direct consequence of label noise is the reduction in Mutual Information (MI) between graph data and observed labels, while the MI between graph data and clean labels serves as an upper bound. Therefore, we further empirically verify this finding and propose that the graph itself can be a reliable source of information, naturally guiding to constraint of the model prediction to correspond with true labels, thus mitigating the influence of label noise on graph learning.

Based on this idea, we devise a high-level optimization objective that maximizes the MI between predicted labels and graph data. Since MI between two variables is notoriously difficult to estimate [1], especially for high-dimensional data like graphs, we further decompose this objective into a target that can be optimized by a decoder GNN and propose Topological Feature Reconstruction (TFR) to address this issue. TFR use is a simple, effective, computationally efficient, and theoretically guaranteed approach for mitigating graph label noise. The core idea behind TFR is that correct labels share more information and can more accurately reconstruct features through graph topology, whereas unreliable labels cannot reconstruct perfectly. Specifically, TFR uses graph topology information as implicit guidance to reconstruct graph features by predicted label patterns, and uses node features as explicit guidance to constrain the reconstructed node features to align with graph data. The only assumption of this idea is that graph labels, features, and topology share sufficient mutual information, making it widely applicable to various real-world scenarios. The primary contributions of this paper are as follows:

- We first conduct empirical and theoretical analyses to investigate the impact of label noise on graph learning from a mutual information perspective. Based on these findings, we identify the high-level objective of mutual information maximization for robust graph learning under label noise.
- To optimize this objective, we further decompose it and introduce a decoder, enabling its efficient optimization. Then, develop the Topological Feature Reconstruction (TFR) method, which serves as a simple, effective, computationally efficient, and theoretically guaranteed solution for robust graph learning under label noise.
- We conduct extensive experiments on benchmark datasets of varying scales, average degrees, and homophily to demonstrate the robustness and broad applicability of our proposed method compared to state-of-the-art baselines.

2 Related Works

2.1 Mutual Information Estimation

Mutual information (MI) measures the relationship between random variables and is notoriously difficult to estimate. MINE [1] addresses this by estimating the lower bound of MI using a classifier to distinguish samples based on the Donsker-Varadhan representation [8] of the KL-divergence. When adopting mutual information for representation learning, the info-max [2] principle advocates for maximizing mutual information between the input and output of deep neural networks. Following this principle, the core idea of Deep Info-max [12] is to maximize MI between the input and output of a neural network. Recently, there have also been works applying MI estimation to the field of graph representation learning. Building upon Deep Info-max, DGI [27] aims to maximize the mutual information between global graph representations and local node representations. DGI utilizes a noise-contrastive objective with a standard Binary Cross-Entropy (BCE) loss to distinguish between positive and negative examples. Unlike DGI, GMI [22] divides mutual information between graph data and representation into feature mutual information and topology-aware mutual information. By simultaneously maximizing these two parts of mutual information, GMI learns useful representations. However, these methods generally focus on aligning graph representation to feature and topology, and they are not specifically designed to mitigate graph label noise.

2.2 Graph Learning under Label Noise

Graph Neural Networks (GNNs) have achieved notable success by employing the message-passing mechanism to aggregate information from neighboring nodes. However, this mechanism also makes GNNs susceptible to label noise [7, 24]. Recently, studies on Graph Neural Networks under Label Noise (GLN) have emerged to enhance robustness against label noise. These methods adopt various strategies, including Loss Regularization [9, 19, 40], Curriculum Learning [30, 31], Graph Structure Augmentation [7, 24, 43], Label Propagation [6], and Contrastive Learning [18, 39]. Most approaches aim to identify and utilize supervision signals other than contaminated labels from graph data to guide learning and prevent overfitting. For instance, CP [40] utilizes community labels as high-level supervision signals and regularizes GNNs' predictions

to align with them; Union-Net [19] samples and aggregates neighbor labels through random walks, utilizing these neighborhood labels as supervision signals. PI-GNN [9] assumes neighboring nodes tend to have similar labels and utilizes neighborhood labels as supervision signals. Graph Structure Augmentation methods, such as NRGNN [7], RTGNN [24], and RNCGLN [43], generally involve adding more edges to introduce additional neighborhood supervision, accompanied by pseudo-label strategies. While these methods have shown promise in addressing the graph label noise problem [29], they are mostly devised based on external assumptions on supervision signals (e.g., they may stipulate that accurate label prediction must align with community labels [40] or neighboring labels [9], or assume nodes with similar representations can provide reliable supervision signals [7, 24]), with few theoretical insights. To the best of our knowledge, no existing GLN works have investigated the graph label noise problem from the information theory perspective.

3 Revisiting Graph Learning under Label Noise

3.1 Notations and Problem Definition

Consider a graph denoted by $\mathcal{G} = \{\mathbf{X}, \mathbf{A}\}$ with N nodes and E edges, where $\mathbf{A} \in \mathbb{R}^{N\times N}$ is the adjacency matrix, and $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_N] \in \mathbb{R}^{N\times d}$ is the node feature matrix, where d is the feature dimension. The node features set is given by $\mathcal{X}$. Each node has a latent true label within c classes, denoted by the set $\mathcal{Y}^* = \{y_1^*, y_2^*, \cdots, y_N^*\}$, and the corresponding one-hot label matrix is $Y^* \in \mathbb{R}^{N\times c}$.

Let the set of all nodes be $\mathcal{V}$, our focus is on the semi-supervised node classification problem, where a small set of nodes $\mathcal{V}_L$ has assigned labels for the training procedure, represented by $\mathcal{Y}_L^* = \{y_1^*, y_2^*, \cdots, y_l^*\}$, where l is the number of labeled nodes. The remaining nodes are unlabeled, denoted as $\mathcal{V}_U = \mathcal{V} - \mathcal{V}_L$, and the corresponding label set is $\mathcal{Y}_U$. Given $\mathcal{G}$, the goal of node classification is to train a classifier $f_\theta : (\mathcal{G}) \rightarrow \hat{\mathbf{Y}}^{N\times c} = [\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2, \cdots, \hat{\mathbf{y}}_N]$ by minimizing $\mathcal{L}(f_\theta(\mathcal{G}), \mathrm{Y}_L^*)$, where c is the number of classes, and $\mathcal{L}$ is the loss function that measures the difference between the predicted labels and the ground truth labels. Typically, f_θ is a well-designed Graph Neural Network parameterized by θ. According to the Empirical Risk Minimization (ERM) principle, the well-trained classifier f_{θ^*} can generalize to unseen data $\mathcal{V}_U$.

However, in real-world scenarios, the accessible labels $\mathcal{Y}_L$ can be corrupted by label noise, reducing the generalization ability of GNNs. We denote the observed noisy labels as $\tilde{\mathcal{Y}}_L = \{\widetilde{y}_1, \widetilde{y}_2, \cdots \widetilde{y}_l\}$, with $\mathcal{Y}_L^*$ being their corresponding true labels. Typically, we consider three types of label noise, defined as follows:

Uniform noise [25] or symmetric noise assumes that the true label has a probability of $\epsilon \in [0, 1]$ to be uniformly flipped to another class. Formally, $\forall_{j\neq i}$ we have $\mathbb{P}(\widetilde{y} = j|y^* = i) = \frac{\epsilon}{c-1}$, where c represents the number of classes.

Pair noise [38] or pair flipping noise assumes that the true label can only be flipped to its corresponding pair class with a probability ϵ. Formally, we have $\mathbb{P}(\widetilde{y} = y_p|y^* = y_c) = \epsilon$ and $\forall_{j\neq y_p, y_c}\mathbb{P}(\widetilde{y} = j|y^* = y_c) = 0$, where y_p is the corresponding pair class of y_c.

Instance-dependent noise [32] or part-dependent label noise assumes that the label transition probability is related to specific part of feature, i.e. $\mathbb{P}(\widetilde{y} \neq y_c|y^* = y_c, x) = \epsilon$.

From above definitions, there exists $(1-\epsilon)\cdot l$ uncorrupted labels ($\tilde{Y} = Y^*$) and $\epsilon \cdot l$ corrupted labels ($\tilde{Y} \neq Y^*$) in $\mathcal{Y}_L$. In this study, we denote the noisy dataset as $\tilde{\mathcal{D}}_L = \{\mathcal{G}, \tilde{\mathcal{Y}}_L\} \sim \mathbb{P}(\tilde{\mathcal{D}})$ drawn from the noisy distribution, and the corresponding clean dataset $\mathcal{D}_L^* = \{\mathcal{G}, \mathcal{Y}_L^*\} \sim \mathbb{P}(\mathcal{D}^*)$ drawn from the original clean distribution. Our goal is to learn a GNN classifier $f_{\tilde{\theta}}$ on noisy data $\tilde{\mathcal{D}}_L$, but can generalize well on clean data distribution $\mathbb{P}(\mathcal{D}^*)$. We consider both the training and validation sets to be contaminated by label noise, i.e. $\tilde{\mathcal{Y}}_L = \{\tilde{\mathcal{Y}}_L^{train}, \tilde{\mathcal{Y}}_L^{val}\}$, using the clean label $\mathcal{Y}^{test} \in \mathcal{Y}_U^*$ to evaluate model performance.

3.2 An Information Theory Perspective on Graph Label Noise

In this subsection, we explore and explain the role of graph information in robust graph learning under label noise. Let $f_\theta(\mathcal{G}) = \hat{\mathrm{Y}}$ denotes the model prediction by GNN f_θ. Suppose f_θ is well designed and have sufficient expressive power. By minimizing the loss function $\mathcal{L}(f_\theta(\mathcal{G}), \tilde{\mathrm{Y}}_L)$, f_θ can accurately fit all training data.

$$\min_\theta \mathcal{L}(f_\theta(\mathcal{G}), \tilde{\mathrm{Y}}_L), \quad f_{\tilde{\theta}}(\mathcal{G}) = \hat{\mathrm{Y}}_{\tilde{\theta}} = \tilde{\mathrm{Y}}. \tag{1}$$

From the information theory perspective, when the training labels are contaminated by label noise, a direct consequence is the reduction in mutual information between the graph data and the training label:

Theorem 3.1. *Given graph $\mathcal{G}$, the latent true label and noisy label are denoted by Y^*, $\tilde{\mathrm{Y}}$, respectively. Let $\mathcal{I}(\cdot;\cdot)$ denote the mutual information calculation operation. Suppose the label transition probability is given by $\mathbb{P}(\tilde{\mathrm{Y}}|\mathrm{Y}^*, \mathcal{G})$, we have $\mathcal{I}(\tilde{\mathrm{Y}}; \mathcal{G}) \leq \mathcal{I}(\mathrm{Y}^*; \mathcal{G})$.*

Proof.

$$\begin{aligned}
\mathcal{I}(\tilde{\mathrm{Y}}; \mathcal{G}) &= \mathbb{E}_{\mathbb{P}(\tilde{\mathcal{D}})}\left[\log \frac{\mathbb{P}(\tilde{\mathrm{Y}}, \mathcal{G})}{\mathbb{P}(\tilde{\mathrm{Y}})\mathbb{P}(\mathcal{G})}\right] \\
&= \int_{\mathcal{G}}\int_{\tilde{\mathcal{Y}}} \mathbb{P}(\tilde{\mathrm{Y}}, \mathcal{G}) \log \frac{\mathbb{P}(\tilde{\mathrm{Y}}, \mathcal{G})}{\mathbb{P}(\tilde{\mathrm{Y}})\mathbb{P}(\mathcal{G})} \\
&= \int_{\mathcal{G}}\int_{\mathcal{Y}^*} \mathbb{P}(\tilde{\mathrm{Y}}, \mathcal{G}) \log \frac{\mathbb{P}(\tilde{\mathrm{Y}}|\mathrm{Y}^*, \mathcal{G})\mathbb{P}(\mathrm{Y}^*, \mathcal{G})}{\mathbb{P}(\tilde{\mathrm{Y}}|\mathrm{Y}^*, \mathcal{G})\mathbb{P}(\mathrm{Y}^*)\mathbb{P}(\mathcal{G})} \\
&= \int_{\mathcal{G}}\int_{\mathcal{Y}^*} \mathbb{P}(\tilde{\mathrm{Y}}, \mathcal{G}) \log \frac{\mathbb{P}(\mathrm{Y}^*, \mathcal{G})}{\mathbb{P}(\mathcal{G})\mathbb{P}(\mathrm{Y}^*)} \\
&= \int_{\mathcal{G}}\int_{\mathcal{Y}^*} \mathbb{P}(\tilde{\mathrm{Y}}|\mathrm{Y}^*, \mathcal{G})\mathbb{P}(\mathrm{Y}^*, \mathcal{G}) \log \frac{\mathbb{P}(\mathrm{Y}^*, \mathcal{G})}{\mathbb{P}(\mathcal{G})\mathbb{P}(\mathrm{Y}^*)} \\
&\leq \int_{\mathcal{Y}^*}\int_{\mathcal{G}} \mathbb{P}(\tilde{\mathrm{Y}}|\mathrm{Y}^*, \mathcal{G}) \int_{\mathcal{Y}^*}\int_{\mathcal{G}} \mathbb{P}(\mathrm{Y}^*, \mathcal{G}) \log \frac{\mathbb{P}(\mathrm{Y}^*, \mathcal{G})}{\mathbb{P}(\mathrm{Y}^*)\mathbb{P}(\mathcal{G})} \\
&= \mathbb{E}_{\mathbb{P}(\mathcal{D})}\left[\log \frac{\mathbb{P}(\mathrm{Y}^*, \mathcal{G})}{\mathbb{P}(\mathrm{Y}^*)\mathbb{P}(\mathcal{G})}\right] = \mathcal{I}(\mathrm{Y}^*; \mathcal{G})
\end{aligned} \tag{2}$$

Q.E.D. □

The above equality holds if and only if $\mathbb{P}(\tilde{\mathrm{Y}}|\mathrm{Y}^*, \mathcal{G}) = \mathbb{P}(\mathrm{Y}^*|\mathrm{Y}^*, \mathcal{G})$, indicating that the training label is clean. From Theorem 3.1, we observe that the mutual information $\mathcal{I}(\tilde{\mathrm{Y}}; \mathcal{G})$ can potentially be a good indicator for distinguishing between reliable and unreliable labels. But before applying this principle to real graph data, we aim to investigate three questions: 1) Does Eq 2 hold true in real graph

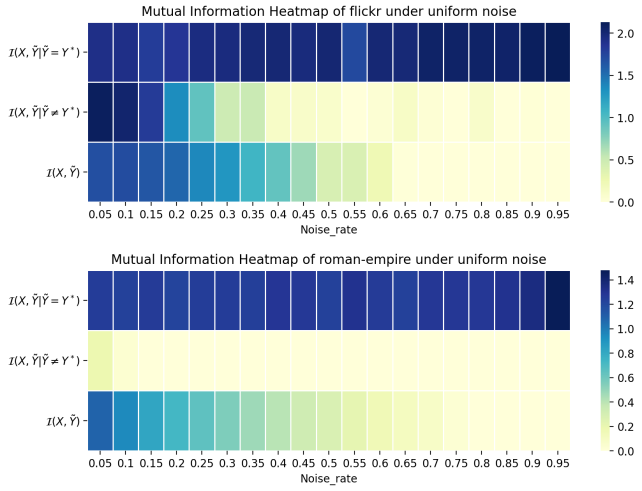

Figure 1: Estimated mutual information lower bound between labels and features of Flickr and Roman-Empire under different levels of uniform noise (10 Runs).

data? 2) Do uncorrupted labels share more mutual information with the graph than observed noisy labels? 3) Do corrupted labels share less mutual information with the graph than observed labels?

To investigate these questions, we conduct empirical studies using a previously developed MI estimation method [1]. Without loss of generality, we calculated the MI between node features and noisy labels. Specifically, we compute $\mathcal{I}(\mathbf{X},\tilde{\mathbf{Y}})$, $\mathcal{I}(\mathbf{X},\tilde{\mathbf{Y}}|\tilde{\mathbf{Y}}\neq\mathbf{Y}^*)$, and $\mathcal{I}(\mathbf{X},\tilde{\mathbf{Y}}|\tilde{\mathbf{Y}}=\mathbf{Y}^*)$. The experimental results presented in Figure 1 demonstrate that the lower bound of mutual information between the feature and observed label decreases as the noise rate of the label increases. Furthermore, we observe that uncontaminated labels $\{\tilde{\mathbf{Y}}|\tilde{\mathbf{Y}}\neq\mathbf{Y}^*\}$ maintain a high level of mutual information with the features, whereas contaminated labels $\{\tilde{\mathbf{Y}}|\tilde{\mathbf{Y}}=\mathbf{Y}^*\}$ exhibit a relatively low level of mutual information with the features. These results demonstrate that mutual information is a reliable indicator for identifying noisy labels and unreliable predictions.

Based on this finding, an intuitive solution for label noise-robust graph learning is to maximize the mutual information maximization objective defined in Eq. (2), which corresponds to the Info-max principle [2]. But before going to the next step, we consider label adjacency to be important information since neighboring labels generally follow specific distributions, like homophily or heterophily assumptions [6, 42]. Label adjacency information is also considered important in previous GLN studies [9]. Therefore, we introduce the graph topology information as a known condition, i.e., changing the objective to $\mathcal{I}\left((\hat{\mathbf{Y}},\mathbf{A});\mathcal{G}\right)$. To maximize the objective in Eq. (2), we can decompose the in the following form:

$$\max_{\theta}\mathcal{I}\left((\hat{\mathbf{Y}},\mathbf{A});\mathcal{G}\right)=\mathcal{H}(\mathcal{G})-\mathcal{H}(\mathcal{G}|\hat{\mathbf{Y}},\mathbf{A}),\tag{3}$$

where $\mathcal{H}(\mathcal{G})$ can be treated as a constant. Thus, the optimization objective is minimize $\mathcal{H}(\mathcal{G}|\hat{\mathbf{Y}})$. When optimizing this objective, we consider label adjacency to be important information since neighboring labels generally follow specific distributions, like homophily

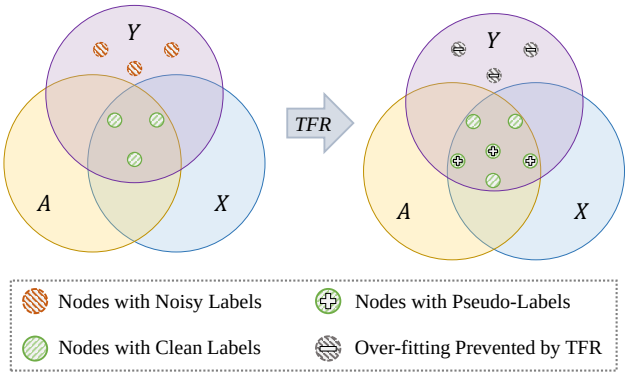

Figure 2: The basic idea of TFR. Clean labels share a higher MI with X and A. So TFR prevents over-fitting on noisy labels with low MI by Feature Reconstruction(FR), and selects high pseudo labels with high MI to enrich training data.

or heterophily assumptions [6, 42]. Label adjacency information is also considered important in previous GLN studies [9]. Therefore, we introduce the graph topology information as a known condition, i.e. $\mathcal{H}(\mathcal{G}|\mathbf{Y},\hat{\mathbf{A}})$. This objective can be further decomposed by:

$$\begin{aligned}&\mathcal{H}(\mathcal{G}|\hat{\mathbf{Y}},\mathbf{A})=\mathcal{H}(\mathbf{X},\mathbf{A}|\hat{\mathbf{Y}},\mathbf{A})=\mathcal{H}(\mathbf{X}|\hat{\mathbf{Y}},\mathbf{A})\\&=-\sum\mathbb{P}(\mathbf{X},\hat{\mathbf{Y}},\mathbf{A})\log(\mathbb{P}(\mathbf{X}|\hat{\mathbf{Y}},\mathbf{A}))\\&\geq-\sum\mathbb{P}(\mathbf{X})\log(\mathbb{P}(\mathbf{X}|\hat{\mathbf{Y}},\mathbf{A}))\\&=\text{CrossEntropy}(\mathbf{X},g_\phi(\hat{\mathbf{Y}},\mathbf{A})).\end{aligned}\tag{4}$$

Introducing graph topology information $\mathbf{A}$ not only helps to better capture the label adjacency information, but also simplifies the optimization objective since we don't have to model the joint distribution $\mathbb{P}(\mathbf{X},\mathbf{A})$.

Therefore, maximizing $\mathcal{I}((\hat{\mathbf{Y}}_\theta,\mathbf{A});\mathcal{G})$ can be achieved by learning another mapping function $\hat{\mathbf{X}}=g_\phi(\hat{\mathbf{Y}},\mathbf{A})$ parameterized by ϕ to reconstruct the feature distribution and minimize Eq. (4). By minimizing this reconstruction loss, the gradients will propagate in the following direction: $(\mathbf{X},\mathbf{A}\rightarrow g_\phi\rightarrow f_\theta)$, enforcing the prediction of f_θ to correspond more with features and less with noisy labels. In this way, $\mathcal{I}\left((\hat{\mathbf{Y}}_\theta,\mathbf{A});\mathcal{G}\right)$ will theoretically reach its upper bound $\mathcal{I}((\mathbf{Y}^*,\mathbf{A});\mathcal{G})$ so that the label prediction can share more information with the true label.

4 Methodology

Based on the above insights, we propose a simple yet effective framework for mitigating graph label noise called Topological Feature Reconstruction (TFR). TFR learns a robust representation under label noise by maximizing $\mathcal{I}\left((\mathbf{A},\hat{\mathbf{Y}}_\theta),\mathcal{G}\right)$, the basic idea of TFR is illustrated at Figure. (2). This framework employs a two-stage Graph Neural Network approach, comprising a backbone GNN and a decoder GNN. The backbone GNN maps the features and topology to the label space, making label predictions. In contrast, the decoder GNN maps label predictions to the feature space, passing supervision signals from the features to the backbone GNN while

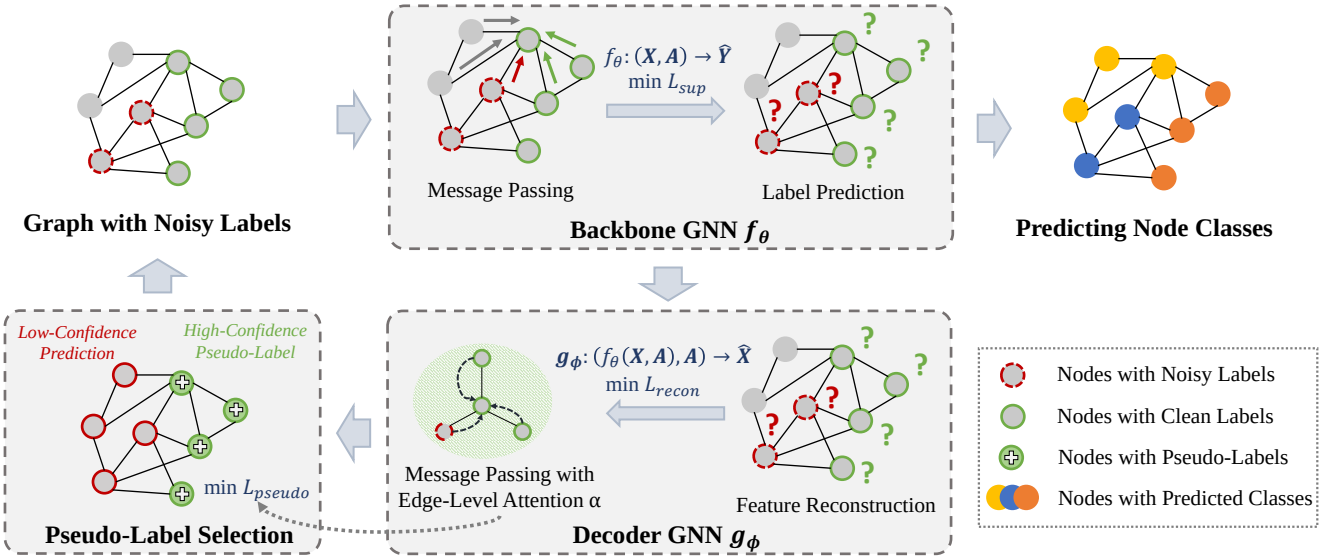

Figure 3: The overall workflow of Topological Feature Reconstruction.

maximizing the mutual information between label and feature. The overall workflow is depicted in Figure 3.

4.1 Topological Feature Reconstruction

In TFR, the function of backbone GNN f_θ is learning the mapping relationship from the input features and the graph topology to the output labels, which is a normal node representation learning procedure.

$$\hat{\mathbf{Y}}_\theta = f_\theta(\mathbf{X}, \mathbf{A}). \tag{5}$$

Here, f_θ can be generalized to any GNN backbone, and we empirically verify this generalization ability in section 5.

Theorem 3.1 reveal that maximizing $\mathcal{I}(\hat{\mathbf{Y}}_\theta; \mathcal{G})$ can make $\hat{\mathbf{Y}}_\theta$ theoretically share more information with $\mathbf{Y}^*$. According to Eq. (4), this objective can be achieved by minimizing $-\sum \mathbb{P}(\mathbf{X})\log(\mathbb{P}(\mathbf{X}|\mathbf{A}, \hat{\mathbf{Y}}))$. Therefore, we utilize a decoder GNN to model $\mathbb{P}(\mathbf{X}|\mathbf{A}, \hat{\mathbf{Y}})$. As an overview, the decoder GNN uses the output from the backbone GNN and graph topology to reconstruct the feature distribution.

$$\hat{\mathbf{X}} = g_\phi(\hat{\mathbf{Y}}, \mathbf{A}) = g_\phi(f_\theta(\mathbf{X}, \mathbf{A}), \mathbf{A}). \tag{6}$$

To capture edge-level label adjacency relationships, we compute edge-level instead of node-level representations. The decoder GNN employs an edge-level attention mechanism to simultaneously reconstruct node features and select confident label predictions. For edge-level attention computation, we adopt the same way as the Graph Attention Network (GAT) [26]. Specifically, for each edge (u, v), the attention score α_{uv} is computed as follows:

$$\alpha_{uv} = \frac{\exp(\sigma(W_{att}[\tau(W_{src}\hat{\mathbf{y}}_u), \tau(W_{dst}\hat{\mathbf{y}}_v)]))}{\sum_{t\in\mathcal{N}(u)} \exp(\sigma(W_{att}[\tau(W_{src}\hat{\mathbf{y}}_u), \tau(W_{dst}\hat{\mathbf{y}}_t)]))}. \tag{7}$$

Here, W_{att}, W_{src}, and W_{dst} are learnable parameters. W_{src} and W_{dst} project label distributions from label space to hidden space, and W_{att} is used to calculate the attention score. $\tau(\cdot)$ and $\sigma(\cdot)$ denote Sigmoid and ReLU activations, respectively. The aggregate edge-level representation $\mathbf{h}_v$ for each node v is computed based on the attention weights and label distribution. We use mean pooling to aggregate edge-level representations:

$$\mathrm{h}_v = \tau(\mathrm{Mean}(\{\alpha_{uv}W^h[W_{src}\hat{\mathbf{y}}_u, W_{dst}\hat{\mathbf{y}}_v] : u \in \mathcal{N}(v)\})), \tag{8}$$

where $\tau(\cdot)$ denotes ReLU activation, and W^h is a projection parameter that maps the label space to the hidden space. Our objective is to maximize $\mathcal{I}\left((\hat{\mathbf{Y}}_\theta, \mathbf{A}); \mathcal{G}\right)$, and excessive redundant parameters in θ_2 can result in over-fitting $\mathcal{I}((\phi, \mathbf{A}); \mathcal{G})$. To reduce the number of parameters in g_ϕ, we adopt the same projection parameters W_{src} and W_{dst} in the attention calculation process. Finally, the reconstructed feature distribution $\hat{\mathbf{x}}_v$ for each node v is obtained by:

$$\hat{\mathrm{x}}_v = \mathrm{softmax}(W^x\mathrm{h}_v), \tag{9}$$

where W^x is a projection parameter that maps the hidden space to the feature space.

4.2 Pseudo-Label Selection

In real-world scenarios, there are often insufficient reliable labels for training. This situation can be even worse when training labels are contaminated by label noise. To gain more supervision signals from the label space, one way is to utilize the pseudo-labeling strategy to enrich the training labels. The major challenge when using pseudo labels is identifying confident model predictions as pseudo labels. A common approach is calculating the confidence score based on the output of f_θ [7, 24, 39, 43]. However, in our work, we use g_ϕ to objectively determine confident pseudo labels.

From our empirical study and theoretical analysis in section 3.2, we know that unreliable predictions in $\hat{\mathbf{Y}}_\theta$ will result in lower mutual information $\mathcal{I}\left((\hat{\mathbf{Y}}_\theta, \mathbf{A}); \mathcal{G}\right)$. So we want to select pseudo labels that share more mutual information with graph data, as we illustrated in figure. (2). To maximize $\mathcal{I}\left((\hat{\mathbf{Y}}_\theta, \mathbf{A}); \mathcal{G}\right)$ in Eq. 3, the decoder GNN g_ϕ will automatically minimize the attention score

on those edges connected with nodes with unreliable predicted labels and exclude them from the feature reconstruction process. So a higher attention score indicates that the specific label prediction can be considered more reliable. Therefore, we utilize the attention score defined in Eq.(7) to select reliable pseudo-labels. Specifically, the label confidence s_v^{conf} for each node v in the current epoch is calculated as the average edge-level attention of all edges connected to v:

$$s_v^{conf} = \frac{1}{d_v} \sum_{u \in \mathcal{N}(v)} \alpha_{vu}. \tag{10}$$

The confidence values range from 0 to 1. During the training phase, the predictions of nodes with the top-k highest confidence are selected as pseudo labels:

$$\mathcal{T}_{sel} = \text{top}_k(S^{conf}). \tag{11}$$

In each epoch, $\mathcal{T}_{sel}$ changes dynamically. This allows pseudo-labels to be continuously adjusted during the learning process, thereby reducing the impact of potential errors in pseudo-label selection.

4.3 Objective Function

By leveraging feature reconstruction and pseudo labeling, we can gain useful supervision signals from features and enlarged confident pseudo labels. But since there exist unmodified labels in noisy label $\tilde{\mathbf{Y}}$, we still consider the observed label as a useful source of supervision signal, and minimize the cross-entropy between the backbone output and observed label. Therefore, to train and optimize TFR, we apply three loss functions collaboratively as the model's objective function.

Feature Reconstruction Loss: According to Eq. 3, to maximize the mutual information between the predicted label, graph topology, and features, we minimize the cross-entropy between the reconstructed feature and the original feature. The feature reconstruction loss is defined as follows:

$$\mathcal{L}_{recon} = -\sum_{v \in \mathcal{D}} \mathbf{x}_v \log(g_\phi(\mathbf{A}, \hat{\mathbf{Y}}_\theta))_v. \tag{12}$$

Pseudo-Label Loss: When there is a lack of sufficient reliable labels for training, pseudo-labels selected by Eq. 11 can be considered a reliable source of additional supervision. We minimize the cross-entropy between the backbone output and confident pseudo labels. The loss based on pseudo-labels is defined as follows:

$$\mathcal{L}_{pseudo} = -\sum_{v \in \mathcal{T}_{sel}} \hat{\mathbf{y}}_v \log(\hat{\mathbf{y}}_v). \tag{13}$$

Supervised Loss: Although there might be possible errors in the training labels, we still regard the training labels as a valuable source of supervision information. The supervised loss for the training labels is calculated as follows:

$$\mathcal{L}_{sup} = -\sum_{v \in \mathcal{D}_{train}} \tilde{\mathbf{y}}_v \log(\hat{\mathbf{y}}_v). \tag{14}$$

Therefore, the overall learning objective is given by:

$$\mathcal{L} = \gamma^t \cdot \mathcal{L}_{pseudo} + (1 - \gamma^t) \cdot \mathcal{L}_{sup} + \beta \cdot \mathcal{L}_{recon}. \tag{15}$$

Here, γ^t is a heuristic dynamic balance factor that set to the *training accuracy* of the current epoch t. A high training accuracy indicates over-fitting, leading to a higher γ and prompting the model to focus more on learning from pseudo-labels. Although simple, this design enables the model to learn from noisy labels during the early under-fitting stage and gradually fit high-confidence pseudo-labels thereafter. We further validate its effectiveness in Section 5.6. The hyper-parameter β is a trade-off weight, where a higher β results in greater robustness but lower expressive power, and vice versa.

4.4 Time Complexity Analysis

Let the hidden dimension be h. For each hidden layer of the backbone GNN f_θ, the time complexity is $O(E \cdot h + N \cdot h^2)$, where $O(E \cdot h)$ corresponds to message passing and $O(N \cdot h^2)$ to linear transformations. For the decoder GNN g_ϕ, the complexity of each component is: (i) Edge attention computation: $O(E \cdot c \cdot h)$, (ii) Aggregate edge-level representation: $O(E \cdot h)$, (iii) Feature reconstruction: $O(N \cdot h \cdot d)$. Thus, the overall time complexity of g_ϕ is $O(E \cdot c \cdot h + N \cdot h \cdot d)$. This analysis shows that the additional decoder introduces only a moderate computational overhead, as its operations are fundamentally similar to those of a standard GNN. Consequently, TFR maintains strong scalability in real-world applications. We further validate this in Section 5.4.

5 Experiments

In this section, we conduct comprehensive experiments on benchmark datasets to answer the following questions:

- **RQ1:** How does TFR perform compared to representative and state-of-the-art GLN models?
- **RQ2:** Can TFR be generalized to different GNN backbones?
- **RQ3:** What is the computational efficiency of TFR?
- **RQ4:** What is the effect of different components in TFR?
- **RQ5:** How do hyper-parameters impact the performance of TFR?

5.1 Experimental Settings

5.1.1 Datasets. We perform node classification experiments under label noise on five widely adopted benchmark datasets, including two citation networks, DBLP [21] and OGBN-Arxiv [13], two online social networks, BlogCatalog and Flickr [37], and one word network, Roman Empire [23]. The detailed statistics of each dataset are shown in Table 1. These datasets vary in scale (ranging from 5,196 to 169,343 nodes), average degrees (ranging from 2.90 to 66.11), and homophily (ranging from 0.05 to 0.83; we follow the homophily definition in [41]), allowing for a multidimensional evaluation of our proposed method and other baseline methods.

Data Split. For BlogCatalog, Flickr, Roman Empire, and DBLP, we adhere to the splitting strategy employed in the popular GLN Benchmark NoisyGL [29], which involves a random split with a specified percentage. For the large-scale datasets OGBN-Arxiv, we utilize the original fixed splits provided in OGB [13].

Label Noise Injection. For label noise injection, we follow the most commonly used experimental setting in NoisyGL [29]. Specifically, we first generate a label transition probability according to the label noise definition in Section 3.1. Then, for each clean label in the training and validation sets, we draw a noisy label from a categorical distribution according to its corresponding transition probability. These noisy labels are used for the training and model

Table 1: Overview of the Dataset Statistics.

Dataset	# Nodes	# Edges	# Feat.	# Classes	# Homophily	Avg. # degree	Split (#Train/#Valid/#Test)
BlogCatalog	5,196	343,486	8,189	6	0.40	66.11	(5%/ 15%/ 80%)
Flickr	7575	239738	12047	9	0.24	63.30	(5%/ 15%/ 80%)
DBLP	17,716	105,734	1,639	4	0.83	5.97	(1%/ 15%/ 84%)
Roman-Empire	22,662	32,927	300	18	0.05	2.90	(5%/ 15%/ 80%)
OGBN-Arxiv	169,343	583,121	128	40	0.66	6.89	(90,941/ 29,799/ 48,603)

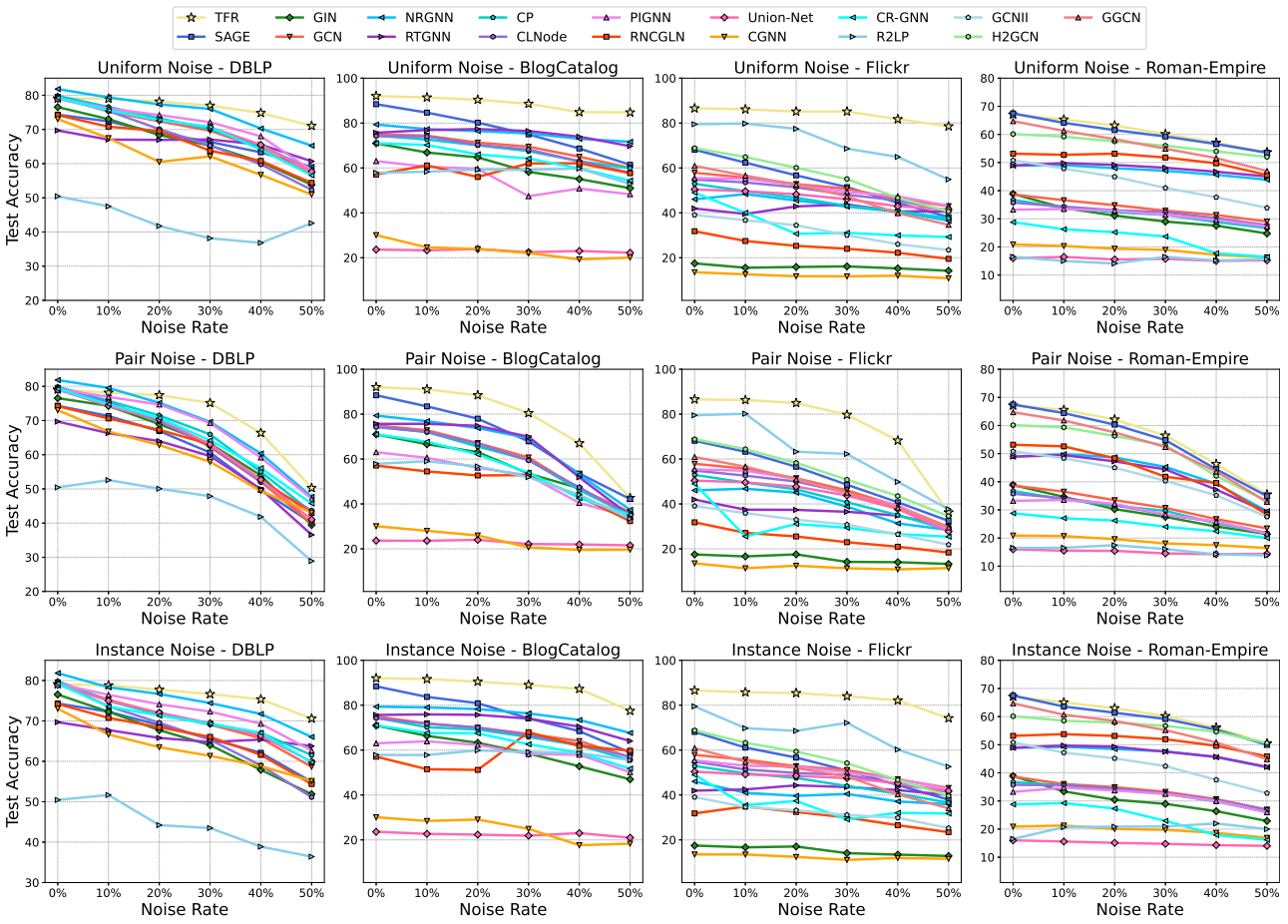

Figure 4: Experiment results on Blogcatalog, DBLP, Flickr, and Roman-Empire under different types of label noise. (10 runs)

selection procedures, while the original clean labels are used to test the model's generalization ability on clean data.

5.1.2 Baselines. We conduct a comparative analysis of our methods against ten state-of-the-art baselines. The comparison includes three Loss Regularization methods: CP [40], Union-Net [19], and PIGNN [9]; one Curriculum Learning method: CLNode [30]; three Graph Structure Learning methods: NRGNN [7], RTGNN [24], and RNCGLN [43]; one Label Propagation method: R2LP [6]; and two Contrastive Learning methods: CGNN [39] and CRGNN [18]. We employ three widely used Graph Neural Network (GNN) models as backbones and baselines, specifically GCN [16], GraphSAGE [11], and GIN [35]. For two highly heterophily datasets, Flickr and Roman Empire, we additionally include three heterophily GNNs, i.e. GCNII [4], H2GCN [42], GGCN [36].

5.1.3 Implementation Settings. To ensure a fair comparison, we adhere to the method configurations outlined in NoisyGL [29], with hyperparameters fine-tuned carefully to achieve optimal performance. For three heterophily GNNs, we tune their hyperparameters using Neural Network Intelligence (NNI). In our proposed TFR approach, we explore the search space of the trade-off hyperparameter β, defined as {0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100}, and the pseudo label threshold parameter K, defined as {0.01, 0.02, 0.03,

Table 2: Generalization experiments on GCN, GIN, and GraphSAGE, under 30% uniform noise. (10 runs)

Dataset	DBLP	BlogCatalog	OGBN-Arxiv
GraphSAGE	66.33 ± 2.69	74.95 ± 1.92	63.11 ± 0.28
Ours+SAGE	**75.48 ± 0.90**	**88.60 ± 3.98**	**64.04 ± 0.23**
Improvement	+13.79%	+18.21%	+1.47%
GIN	65.29 ± 4.22	58.24 ± 4.44	49.45 ± 2.53
Ours+GIN	**73.70 ± 6.51**	**80.65 ± 9.63**	**68.19 ± 0.41**
Improvement	+12.88%	+38.47%	+37.89%
GCN	69.63 ± 2.69	69.48 ± 2.69	62.59 ± 0.24
Ours+GCN	**76.97 ± 1.58**	**80.20 ± 7.82**	**64.58 ± 0.14**
Improvement	+10.11%	+15.42%	+3.17%

0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.2, 0.3, 0.4, 0.5}. For each model, we use validation set accuracy as the early stopping criterion to maximize the model's generalization capability.

5.2 Main Results (RQ1)

The datasets utilized in these experiments can be primarily categorized into two groups: the first group comprises graph datasets with relatively high homophily, i.e., Blogcatalog and DBLP, and the second group includes highly heterophily graph datasets from Flickr and Roman-Empire. For each dataset, we evaluate the performance of all candidate methods under 0% − 50% uniform, pair, and instance-dependent noise, respectively. On the DBLP dataset, TFR uses GCN as backbone; On the other three datasets, TFR uses GraphSAGE as backbone. Main results are illustrated in figures 4, and we report the mean accuracy of 10 runs. From these results, we have the following observations:

TFR consistently outperforms other baseline methods in most conditions. The curve tendency and comparison demonstrate TFR's capability in mitigating label noise on both homophily and heterophily graphs. This verifies the effectiveness of our TFR in noise graph learning. Further, similar to all other baseline methods, we can also observe that the improvement of TFR on the backbone GNN under 50% pair noise is not very significant. One possible explanation is that when the pair noise rate exceeds 50%, labels tend to be swapped with other labels, resulting in minimal useful information regarding the original label. In such scenarios, re-labeling the data is usually a more effective approach.

Impact of different noise types. For uniform noise, as noise levels rise, the degradation in test accuracy is generally gradual for TFR compared to baselines. This highlights the capability of TFR to handle random label corruption. Furthermore, TFR also achieves remarkable performance under instance-dependent label noise, indicating its strong potential to deal with complex, real-world label noise scenarios. The effect of pair noise is more pronounced, with accuracy dropping faster than that of uniform noise and instance-dependent noise for all methods. Even though TFR still maintains a performance advantage over baselines.

5.3 Generalization Study (RQ2)

To evaluate the effectiveness of the proposed TFR in enhancing the robustness of various backbone GNNs under label noise, we conduct generalization experiments on three backbone GNNs, i.e., GraphSAGE [11], GIN [35], and GCN [16], under 30% uniform noise, both with and without TFR, across the DBLP, BlogCatalog, and OGBN-Arxiv datasets. We report the mean test accuracy and corresponding standard deviation over 10 runs. The results are presented in Table 2. From these results, we can observe that TFR can substantially improve the robustness of different backbone GNNs to label noise in most cases. For instance, TFR improves the test accuracy of GCN, GIN, and GraphSAGE on BlogCatalog by 15.42%, 38.47%, and 18.21%, respectively. This provides strong evidence that TFR can effectively prevent overfitting on noisy labels on various GNN backbones.

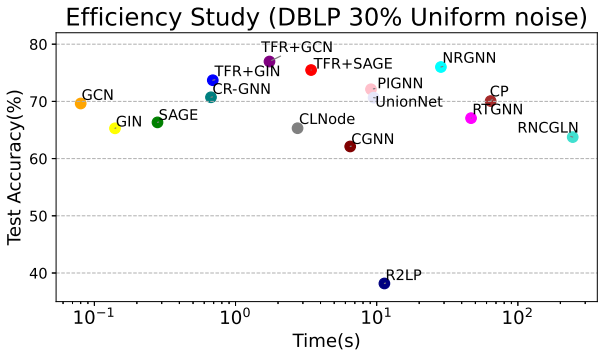

Figure 5: Time consumption and test accuracy of different methods on DBLP, 30% uniform noise. (10 runs)

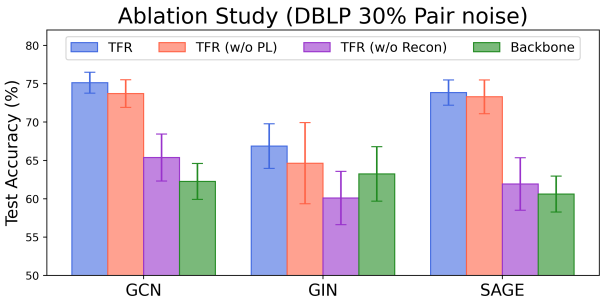

Figure 6: Ablation study on DBLP, 30% pair noise. (10 runs)

5.4 Efficiency Study (RQ3)

In real-world applications, computational efficiency is a crucial consideration. Generally, there is a trade-off between computational efficiency and model performance. Our primary concern is whether our method has achieved a balance between the two. For all methods, we report the mean test accuracy and processing time (time consumption until the highest valid accuracy) on DBLP under 30% uniform noise across 10 runs, using an NVIDIA L20 GPU with 48GB memory. The results are shown in Figure 5. These results indicate that while TFR consumes more time than backbone GNNs, it achieves a reasonable trade-off between performance and computational efficiency. In contrast, some other methods require significantly more time to converge, even necessitating thousands of times more processing time than GNNs.

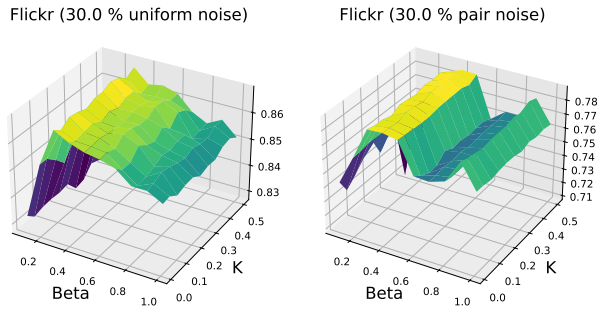

Figure 7: Hyper-parameter sensitivity analysis of β and k on Flickr under 30% uniform noise (Left) and 30% pair noise (Right), 10 runs.

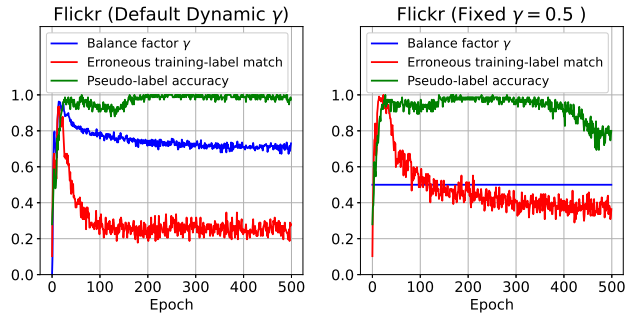

Figure 8: Analysis of balance factor γ on Flickr under 30% uniform noise: Comparison between the proposed dynamic γ (Left), and a fixed $\gamma = 0.5$ (Right).

5.5 Ablation Study (RQ4)

Our TFR framework consists of two components: feature reconstruction and pseudo-labeling. We empirically verify the effectiveness of the two components of TFR through an ablation study. Specifically, we do an ablation study on Pseudo Labeling by directly removing $\mathcal{L}_{pseudo}$. Since the confidence score for pseudo-labeling is calculated based on feature reconstruction, we implemented gradient decoupling between backbone and decoder GNN to do an ablation study on Feature Reconstruction, i.e., cut off the gradient flow $g_\phi \rightarrow f_\theta$. Given that the pseudo-labeling is primarily designed to mitigate label scarcity, we use only 1% of nodes in the DBLP dataset for training in this experiment. We report the mean test accuracy and corresponding standard deviation over 10 runs. As shown in Figure 6, combining $\mathcal{L}_{recon}$ and $\mathcal{L}_{pseudo}$ achieves optimal robustness – this confirms the synergistic effect of feature reconstruction and pseudo labeling. And using $\mathcal{L}_{recon}$ alone improves backbone robustness, while using $\mathcal{L}_{pseudo}$ alone shows limited benefits. These results validate the crucial role of the feature-reconstruction component in our method, as well as demonstrating the role of pseudo-labeling to provide reliable supervision signals. We also observed that removing $\mathcal{L}_{recon}$ leads to a greater performance drop. According to Eq. (10), this may be because pseudo-label selection fundamentally depends on minimizing $\mathcal{L}_{recon}$. Removing $\mathcal{L}_{recon}$ would affect the attention score computation, thus affecting the reliability of the confidence score.

5.6 Hyperparameter Analysis (RQ5)

How does the two key hyper-parameters β and K affect the performance of TFR ? From the definition, a higher β increases robustness to label noise but decreases alignment with the original labels. K determines the confidence score threshold for considering a pseudo label as a reliable supervision signal. We conduct sensitivity analysis on Flickr under 30% uniform noise and pair noise, and report the mean test accuracy of 10 runs. As shown in Figure 7, we observe that (i) β is sensitive to different types of label noise, as trends change significantly with different noise types; (ii) A low β results in a significant performance decline, as the mutual information between labels and features is not fully maximized, reducing the model's ability to resist label noise. Besides, since the feature reconstruction model is not well trained, the confidence score calculated from the attention score α becomes unreliable, further rendering the selected pseudo-labels unreliable; (iii) TFR is generally not sensitive to variations of K.

How does the dynamic balance factor γ evolve during training, and how does it affect model performance? To answer these questions, we track three metrics: (a) **Balance factor** γ. (b) **Erroneous training label match rate**: the fraction of backbone predictions that match corrupted (wrong) training labels—indicating overfitting. (c) **Pseudo label accuracy**: the accuracy of high-confidence pseudo-labels compared to clean labels. Results are shown in Figure 8 (Left), We observe: (i) In early training, the model quickly over-fits to noisy labels, while γ rises sharply to suppress noisy supervision. (ii) Around epoch 15, γ peaks as pseudo-label accuracy improves, and overfitting to incorrect labels drops significantly. (iii) In later stages, all metrics stabilize. We also evaluate a fixed $\gamma = 0.5$ (Figure 8, Right) for comparison. This setting fails to suppress overfitting and causes a sharp decline in pseudo-label accuracy in later epochs. These results indicate that γ can effectively balance supervision from training and pseudo-labels, thereby helping mitigate over-fitting to noisy labels.

6 Conclusion

In this paper, we analyze label noise in graphs from an information-theoretic perspective and propose a simple yet effective GLN method called Topological Feature Reconstruction (TFR). TFR leverages the insight that reliable labels share higher mutual information with the graph data, thereby modeling robustness through topological feature reconstruction and high-confidence pseudo-labels. Extensive experiments on datasets of varying properties demonstrate that TFR outperforms current state-of-the-art baselines. **Limitation and Future work**: We also recognize that the application scenarios of TFR are limited, as we do not specifically account for structure noise and feature noise in graph adversarial attack settings. Future work includes exploring robust graph learning strategies that are simultaneously resilient to label, feature, and structure noise.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 62476245).

A GenAI Usage Disclosure

During the preparation of this paper, GenAI were used solely for light editing on words, such as grammar corrections and minor stylistic refinements. No GenAI were employed to generate core content. All contributions of this paper are original and trustworthy.

References

[1] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and R Devon Hjelm. 2018. Mine: mutual information neural estimation. arXiv preprint arXiv:1801.04062 (2018).

[2] Anthony J Bell and Terrence J Sejnowski. 1995. An information-maximization approach to blind separation and blind deconvolution. Neural computation 7, 6 (1995), 1129–1159.

[3] WANG Botao, Jia Li, Yang Liu, Jiashun Cheng, Yu Rong, Wenjia Wang, and Fugee Tsung. 2023. Deep insights into noisy pseudo labeling on graph data. In Thirty-seventh Conference on Neural Information Processing Systems.

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

[5] Xiaojun Chen, Shengbin Jia, and Yang Xiang. 2020. A review: Knowledge reasoning over knowledge graph. Expert systems with applications 141 (2020), 112948.

[6] Yao Cheng, Caihua Shan, Yifei Shen, Xiang Li, Siqiang Luo, and Dongsheng Li. 2024. Resurrecting Label Propagation for Graphs with Heterophily and Label Noise. In Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (Barcelona, Spain) (KDD '24). Association for Computing Machinery, New York, NY, USA, 433–444. doi:10.1145/3637528.3671774

[7] Enyan Dai, Charu Aggarwal, and Suhang Wang. 2021. 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. 227–236.

[8] Monroe D Donsker and SR Srinivasa Varadhan. 1983. Asymptotic evaluation of certain Markov process expectations for large time. IV. Communications on pure and applied mathematics 36, 2 (1983), 183–212.

[9] Xuefeng Du, Tian Bian, Yu Rong, Bo Han, Tongliang Liu, Tingyang Xu, Wenbing Huang, Yixuan Li, and Junzhou Huang. 2021. Noise-robust graph learning by estimating and leveraging pairwise interactions. arXiv preprint arXiv:2106.07451 (2021).

[10] Jacob Goldberger and Ehud Ben-Reuven. 2022. Training deep neural-networks using a noise adaptation layer. In International conference on learning representations.

[11] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. Advances in neural information processing systems 30 (2017).

[12] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. 2018. Learning deep representations by mutual information estimation and maximization. arXiv preprint arXiv:1808.06670 (2018).

[13] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open graph benchmark: Datasets for machine learning on graphs. Advances in neural information processing systems 33 (2020), 22118–22133.

[14] Weiwei Jiang and Jiayun Luo. 2022. Graph neural network for traffic forecasting: A survey. Expert Systems with Applications 207 (2022), 117921.

[15] Jeremy Kawahara, Colin J Brown, Steven P Miller, Brian G Booth, Vann Chau, Ruth E Grunau, Jill G Zwicker, and Ghassan Hamarneh. 2017. BrainNetCNN: Convolutional neural networks for brain networks; towards predicting neurodevelopment. NeuroImage 146 (2017), 1038–1049.

[16] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016).

[17] Jure Leskovec and Julian Mcauley. 2012. Learning to discover social circles in ego networks. Advances in neural information processing systems 25 (2012).

[18] Xianxian Li, Qiyu Li, Haodong Qian, Jinyan Wang, et al. 2024. Contrastive learning of graphs under label noise. Neural Networks 172 (2024), 106113.

[19] Yayong Li, Jie Yin, and Ling Chen. 2021. Unified robust training for graph neural networks against label noise. In Pacific-Asia Conference on Knowledge Discovery and Data Mining. Springer, 528–540.

[20] Hoang NT, Choong Jun Jin, and Tsuyoshi Murata. 2019. Learning graph neural networks with noisy labels. arXiv preprint arXiv:1905.01591 (2019).

[21] Shirui Pan, Jia Wu, Xingquan Zhu, Chengqi Zhang, and Yang Wang. 2016. Tri-party deep network representation. In International Joint Conference on Artificial Intelligence 2016. Association for the Advancement of Artificial Intelligence (AAAI), 1895–1901.

[22] Zhen Peng, Wenbing Huang, Minnan Luo, Qinghua Zheng, Yu Rong, Tingyang Xu, and Junzhou Huang. 2020. Graph representation learning via graphical mutual information maximization. In Proceedings of The Web Conference 2020. 259–270.

[23] Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. 2023. A critical look at the evaluation of GNNs under heterophily: Are we really making progress? arXiv preprint arXiv:2302.11640 (2023).

[24] Siyi Qian, Haochao Ying, Renjun Hu, Jingbo Zhou, Jintai Chen, Danny Z Chen, and Jian Wu. 2023. Robust training of graph neural networks via noise governance. In Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining. 607–615.

[25] Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. 2022. Learning from noisy labels with deep neural networks: A survey. IEEE transactions on neural networks and learning systems (2022).

[26] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. arXiv preprint arXiv:1710.10903 (2017).

[27] Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2018. Deep graph infomax. arXiv preprint arXiv:1809.10341 (2018).

[28] Haoyu Wang, Zhuo Huang, Zhiwei Lin, and Tongliang Liu. 2024. Noisegpt: Label noise detection and rectification through probability curvature. Advances in Neural Information Processing Systems 37 (2024), 120159–120183.

[29] Zhonghao Wang, Danyu Sun, Sheng Zhou, Haobo Wang, Jiapei Fan, Longtao Huang, and Jiajun Bu. 2024. NoisyGL: A Comprehensive Benchmark for Graph Neural Networks under Label Noise. In Advances in Neural Information Processing Systems, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (Eds.), Vol. 37. Curran Associates, Inc., 38142–38170. https://proceedings.neurips.cc/paper_files/paper/2024/file/436ffa18e7e17be336fd884f8ebb5748-Paper-Datasets_and_Benchmarks_Track.pdf

[30] Xiaowen Wei, Xiuwen Gong, Yibing Zhan, Bo Du, Yong Luo, and Wenbin Hu. 2023. Clnode: Curriculum learning for node classification. In Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining. 670–678.

[31] Yuhao Wu, Jiangchao Yao, Xiaobo Xia, Jun Yu, Ruxin Wang, Bo Han, and Tongliang Liu. 2024. Mitigating label noise on graph via topological sample selection. arXiv preprint arXiv:2403.01942 (2024).

[32] Xiaobo Xia, Tongliang Liu, Bo Han, Nannan Wang, Mingming Gong, Haifeng Liu, Gang Niu, Dacheng Tao, and Masashi Sugiyama. 2020. Part-dependent label noise: Towards instance-dependent label noise. Advances in Neural Information Processing Systems 33 (2020), 7597–7610.

[33] Xiaobo Xia, Tongliang Liu, Nannan Wang, Bo Han, Chen Gong, Gang Niu, and Masashi Sugiyama. 2019. Are anchor points really indispensable in label-noise learning? Advances in neural information processing systems 32 (2019).

[34] Gezheng Xu, Li Yi, Pengcheng Xu, Jiaqi Li, Ruizhi Pu, Changjian Shui, Charles Ling, A Ian McLeod, and Boyu Wang. 2025. Unraveling the mysteries of label noise in source-free domain adaptation: Theory and practice. IEEE Transactions on Pattern Analysis and Machine Intelligence (2025).

[35] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How powerful are graph neural networks? arXiv preprint arXiv:1810.00826 (2018).

[36] Yujun Yan, Milad Hashemi, Kevin Swersky, Yaoqing Yang, and Danai Koutra. 2022. Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks. In 2022 IEEE International Conference on Data Mining (ICDM). IEEE, 1287–1292.

[37] Renchi Yang, Jieming Shi, Xiaokui Xiao, Yin Yang, Juncheng Liu, and Sourav S Bhowmick. 2020. Scaling attributed network embedding to massive graphs. Proceedings of the VLDB Endowment 14, 1 (2020), 37–49.

[38] Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. 2019. How does disagreement help generalization against label corruption?. In International conference on machine learning. PMLR, 7164–7173.

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

[40] Mengmei Zhang, Linmei Hu, Chuan Shi, and Xiao Wang. 2020. Adversarial label-flipping attack and defense for graph neural networks. In 2020 IEEE International Conference on Data Mining (ICDM). IEEE, 791–800.

[41] Zhou Zhiyao, Sheng Zhou, Bochao Mao, Xuanyi Zhou, Jiawei Chen, Qiaoyu Tan, Daochen Zha, Yan Feng, Chun Chen, and Can Wang. 2024. Opengsl: A comprehensive benchmark for graph structure learning. Advances in Neural Information Processing Systems 36 (2024).

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

[43] Yonghua Zhu, Lei Feng, Zhenyun Deng, Yang Chen, Robert Amor, and Michael Witbrock. 2024. Robust Node Classification on Graph Data with Graph and Label Noise. In Proceedings of the AAAI Conference on Artificial Intelligence. 17220–17227.