Uncertainty-Aware Graph Structure Learning

Shen Han[†]
The State Key Laboratory of
Blockchain and Data Security,
Zhejiang University
Hangzhou, China
drhanshen@zju.edu.cn

Zhiyao Zhou[†]
The State Key Laboratory of
Blockchain and Data Security,
Zhejiang University
Hangzhou, China
zjucszzy@zju.edu.cn

Jiawei Chen[*†‡]
The State Key Laboratory of
Blockchain and Data Security,
Zhejiang University
Hangzhou, China
sleepyhunt@zju.edu.cn

Zhezheng Hao
Northwestern Polytechnical
University
Xi'an, China
haozhezheng@outlook.com

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

Gang Wang
Bangsun Technology
Hangzhou, China
wanggang@bsfit.com.cn

Yan Feng[†]
The State Key Laboratory of
Blockchain and Data Security,
Zhejiang University
Hangzhou, China
fengyan@zju.edu.cn

Chun Chen[†]
The State Key Laboratory of
Blockchain and Data Security,
Zhejiang University
Hangzhou, China
chenc@zju.edu.cn

Can Wang[‡]
The State Key Laboratory of
Blockchain and Data Security,
Zhejiang University
Hangzhou, China
wcan@zju.edu.cn

Abstract

Graph Neural Networks (GNNs) have become a prominent approach for learning from graph-structured data. However, their effectiveness can be significantly compromised when the graph structure is suboptimal. To address this issue, *Graph Structure Learning* (GSL) has emerged as a promising technique that refines node connections adaptively. Nevertheless, we identify two key limitations in existing GSL methods: 1) Most methods primarily focus on node similarity to construct relationships, while overlooking the quality of node information. Blindly connecting low-quality nodes and aggregating their ambiguous information can degrade the performance of other nodes. 2) The constructed graph structures are often constrained to be symmetric, which may limit the model's flexibility and effectiveness.

To overcome these limitations, we propose an **Uncertainty-aware Graph Structure Learning** (UnGSL) strategy. UnGSL estimates the uncertainty of node information and utilizes it to adjust the strength of directional connections, where the influence of nodes with high uncertainty is adaptively reduced. Importantly, UnGSL serves as a plug-in module that can be seamlessly integrated into existing GSL methods with minimal additional computational cost. In our experiments, we implement UnGSL into six representative GSL methods, demonstrating consistent performance improvements. The code is available at https://github.com/UnHans/UnGSL.

CCS Concepts

• **Computing methodologies** → **Neural networks**; • **Information systems** → Data mining.

Keywords

Graph Structure Learning; Graph Neural Network; Uncertainty

[*]Corresponding author.
[†]College of Computer Science and Technology, Zhejiang University.
[‡]Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security.

1 Introduction

Graph neural networks (GNNs) [17, 24, 44] have demonstrated remarkable performance in tackling graph-structured data. To date, GNNs have evolved with increasingly sophisticated model architectures [3, 5, 9, 11, 27, 39] to enhance their capabilities. However, these model-centric methods often neglect potential flaws in the underlying graph structure, which can lead to suboptimal performance. In practice, graph data frequently exhibit suboptimal characteristics, such as noisy connections and incomplete information, due to the inherent complexities and inconsistencies in data collection [37, 51].

To address these issues, Graph Structure Learning (GSL) [6, 13, 23, 30, 49], a data-centric approach, has garnered increasing attention. Beyond learning node representations with GNNs, GSL learns to
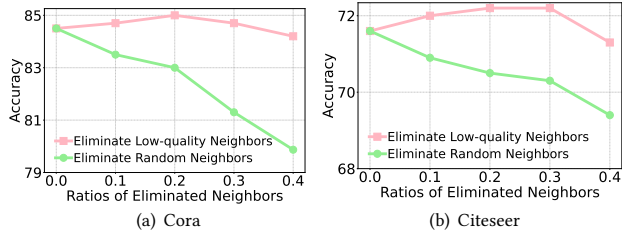
Figure 1: Performance varies with the ratios of eliminated neighbors on Cora and Citeseer datasets. Here we first constructed a graph based on GRCN [47]. We then eliminate a certain ratio of neighbors with the highest entropy for each node, and evaluate the performance of GCN on such a pruned graph. For comparison, we also report the performance under random elimination.

refine node connections and edge weights. This approach has been shown to effectively enhance the accuracy of GNNs on downstream tasks while improving their resilience to topological perturbations [26, 51]. Early work on GSL directly treated the graph structure (*i.e.,* the adjacency matrix) as learnable parameters. However, due to the large parameter space, these strategies often incur substantial computational overhead and are difficult to train effectively [14, 23]. More recently, research has shifted towards embedding-based GSL [6, 22, 28, 41], which constructs the adjacency matrix based on the similarity of node embeddings. Various similarity metrics, such as cosine similarity [22, 41] or neural networks [28], have been employed. These methods aim to increase graph homophily and typically achieve state-of-the-art performance, as nodes with similar features (or embeddings) are more likely to be connected.

Despite their success, we identify two key limitations in these embedding-based GSL methods:

- **These methods mainly rely on embedding similarity for graph construction while neglecting the quality of node information.** Given the critical role of edges in GNNs as conduits for information propagation, it is essential to evaluate the quality of the information being propagated. Aggregating unclear or ambiguous information from neighbor nodes can disrupt the embedding learning of the target node. Constructing connections based solely on node similarity, without assessing the quality of the node's information, may lead to suboptimal performance. To empirically validate this point, we conduct a simple experiment using a representative GSL method (GRCN [47]). As shown in Figure 1, removing a certain proportion of neighbors with the highest entropy results in a significant performance gain.
- **These methods often generate symmetric graph structures, which can hinder their effectiveness.** Current embedding-based methods tend to construct symmetric relationships between nodes, implying that both nodes exert an equal and bidirectional influence during the GNN learning process. This imposed symmetry can constrain the model's flexibility and capacity, particularly when the connected nodes differ in quality. For example, consider a scenario where a high-quality node is linked to a low-quality node (refer to Figure 2). While the high-quality node can provide valuable information that greatly benefits the low-quality node, the reverse influence from the low-quality node may have
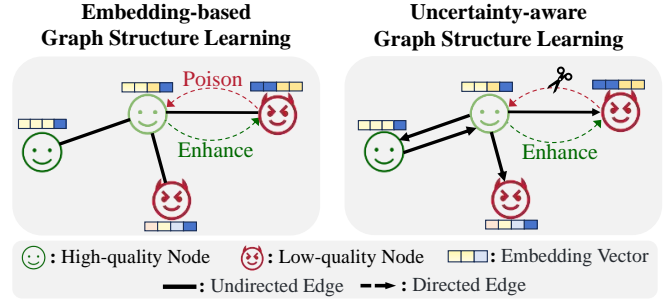

Figure 2: Illustration of how our UnGSL differs from existing embedding-based GSL methods. Existing GSL learns symmetric relationships, leading to a dilemma when managing connections between high-quality and low-quality nodes. In contrast, UnGSL learns asymmetric relationships, allowing low-quality nodes to benefit from high-quality nodes while mitigating the negative influence of low-quality nodes.

a negative impact on the high-quality node. Symmetric relationships fail to account for this disparity, leading to a dilemma in the learning process. This inspires us to explore asymmetric structure learning. By modeling directional relationships separately, we allow the low-quality node to benefit from the high-quality node's information while reducing the adverse influence in the opposite direction, thus protecting the high-quality node from negative effects. Although a few studies [30, 34, 35] have begun to explore asymmetric graph structure learning, they typically restrict the asymmetry to relations between labeled and unlabeled nodes, overlooking the richer relationships between the vast majority of unlabeled nodes.

To overcome these limitations, we propose an uncertainty-aware graph structure learning (UnGSL) method that considers nodes' information quality to learn an asymmetric graph structure. UnGSL directly utilizes the uncertainty (Shannon Entropy [1]) of the node in classification to indicate the node's information quality, and conducts theoretical analyses to demonstrate that blindly aggregating information from high-uncertainty nodes would lift the lower bound of uncertainty for the target node. Building on this, UnGSL leverages a learnable node-wise threshold to differentiate low-quality neighbors from high-quality ones, and adaptively reduces directional edge weights from those low-quality neighbors. Notably, our UnGSL is simple and can be easily incorporated into various embedding-based GSL methods, boosting their performance with minor extra computational overhead.

In summary, this work makes the following contributions:

- We highlight the necessity of modeling a node's uncertainty in graph structure learning, and theoretically demonstrate that the uncertainty of a node after GNN layer is positively correlated with those of its neighbors.
- We propose a simple yet novel uncertainty-aware graph structure learning strategy (UnGSL), which can be seamlessly integrated with various embedding-based GSL models to mitigate the directional impact of high-uncertainty nodes.
- We conduct extensive experiments to demonstrate that UnGSL can consistently boost existing embedding-based GSL models

across seven benchmark datasets, with an average performance increase of 2.07%.

2 Preliminaries

In this section, we introduce basic notations and background on GNNs and GSL methods.

Consider the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A}, \mathbf{X})$, where $\mathcal{V}$ represents a set of n nodes $\{v_1, ..., v_n\}$ and $\mathcal{E}$ represents the set of edges. Let $\mathbf{A}$ be the initial adjacency matrix of the graph, where $\mathbf{A}_{ij} = 1$ if an edge exists between node v_i and v_j; otherwise, $\mathbf{A}_{ij} = 0$. The matrix $\mathbf{X} = [x_1, ..., x_n] \in \mathbb{R}^{n \times d}$ represents the node feature matrix, where each column x_i corresponds to the feature vector of node v_i. Let $\mathbf{D}$ denote the diagonal degree matrix defined as $\mathbf{D}_{ii} = 1 + \sum_j \mathbf{A}_{ij}$; and $\hat{\mathbf{A}}$ denotes the normalized adjacency matrix with self-loop, i.e., $\hat{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{A}+\mathbf{I})\mathbf{D}^{-\frac{1}{2}}$ for symmetric normalization or $\hat{\mathbf{A}} = \mathbf{D}^{-1}(\mathbf{A}+\mathbf{I})$ for row normalization.

2.1 Graph Neural Networks

Graph Neural Networks (GNNs) have become a prominent approach for learning from graph-structured data, where node representations are learned by iteratively aggregating and transforming information from neighboring nodes. In recent years, various designs for the aggregation and transformation processes have given rise to different GNN models[24, 38, 44]. Among these, the Graph Convolutional Network (GCN) [24] stands out as one of the most widely adopted and influential architectures. The operation at the l-th layer in GCN can be formulated as:

$$\mathbf{Z}^{(l)} = \sigma(\hat{\mathbf{A}}\mathbf{Z}^{(l-1)}\mathbf{W}^{(l)}), \tag{1}$$

where $\sigma(\cdot)$ denotes the activation function, $\mathbf{W}^{(l)}$ is a learnable parameter matrix used to transform the node features.

Given the critical role of the graph structure in GNNs, which determines the sources of information aggregation, ensuring the quality of graph structure is of paramount importance. Recent work demonstrates that suboptimal graph structures, even with the introduction of a small percentage of noisy edges or topological perturbations (e.g., 10%), can significantly degrade the performance of GNNs (e.g., 25%) [26, 51].

2.2 Graph Structure Learning

Graph Structure Learning (GSL) aims to enhance the accuracy and robustness of GNNs by learning a refined adjacency matrix $\mathbf{S}$, where $\mathbf{S}_{ij}$ denotes the edge weight between node v_i and node v_j. For convenience, we also use $\mathbf{S}_{ij}$ to indicate whether there exists a edge between node v_i and v_j. When $\mathbf{S}_{ij} > 0$, it indicates the presence of a directed edge from v_j to v_i, with the edge weight equal to $\mathbf{S}_{ij}$. Conversely, $\mathbf{S}_{ij} = 0$ indicates that no edge exists.

The quality of the learned adjacency matrix is often examined on the downstream tasks. Conventionally, $\mathbf{S}$ is fed into a GNN encoder to obtain node representations, which are used to evaluate performance on downstream tasks. The objective function of most GSL methods can be generally formulated as:

$$\mathcal{L} = \mathcal{L}_{Task}(\mathbf{Z}, \mathbf{Y}) + \lambda \mathcal{L}_{Reg}(\mathbf{Z}, \mathbf{S}), \tag{2}$$

where $\mathbf{Z} = \text{GNN}(\mathbf{S}, \mathbf{X})$ is the node embedding matrix, $\mathcal{L}_{Task}$ aims to utilize the labels $\mathbf{Y}$ as a supervised signal to optimize both the GNN

encoder and the adjacency matrix $\mathbf{S}$. The regularization term $\mathcal{L}_{Reg}$ are introduced by existing GSL methods to achieve diverse desirable properties in the learned adjacency matrix $\mathbf{S}$, such as sparsity [23], feature smoothness [35], and connectivity [47] in the graph structure. For example, PROGNN [23] employs l_1 norm penalization on the learned adjacency matrix to enhance the sparsity of the graph structure. λ is a trade-off hyperparameter.

Traditional GSL methods [14, 23] that treat each edge $\mathbf{S}_{ij}$ as a learnable parameter often suffer from significant computational overhead and are challenging to train efficiently. Recent research has focused on embedding-based GSL methods [6, 22, 28, 41] , which employ an auxiliary GNN encoder to extract node embeddings from the initial graph and estimate embedding similarities as edge weights:

$$\mathbf{E} = \text{GNN}(\mathbf{A}, \mathbf{X}), \tag{3}$$

$$\mathbf{S}_{ij} = \mathbf{S}_{ji} = \phi(\mathbf{E}_i, \mathbf{E}_j). \tag{4}$$

Here $\mathbf{E}$ is the node embedding matrix, which differs from $\mathbf{Z}$ as it is specifically used for constructing the graph structure. Note that existing works generally utilize different GNN architectures and input structure compared with those used to compute the embedding $\mathbf{Z}$ when constructing the embedding $\mathbf{E}$. $\phi(\cdot)$ is a metric function (i.e., cosine similarity) used to calculate the similarity between nodes.

Although the structure modeling paradigm in Equation (4) is widely adopted in GSL models, it suffers from two key limitations:

- **This paradigm relies on embedding similarity while neglecting the quality of node information.** Only semantic similarities between embeddings $\mathbf{E}_i$ and $\mathbf{E}_j$ are considered in this structure modeling paradigm, while the varying uncertainties of them, which reflect their information quality, are neglected. This may undermine the quality of embeddings of target nodes when aggregating inferior information from low-quality neighbors (as validated by the preliminary experiment in the Introduction) .
- **This paradigm constraining the graph to be symmetric, which potentially hinder effectiveness of GSL models.** The pair-wise similarity constrains the learned edge between v_i and v_j to be bidirectional, overlooking their unequal influence due to varying information quality. For example, if $\mathbf{E}_i$ contains higher-quality information than $\mathbf{E}_j$, the constructed edge $\mathbf{S}_{ji}$ can provide valuable information that greatly benefits the low-quality node v_j while the edge $\mathbf{S}_{ij}$ propagates inferior information to poison the embedding of v_i. By modeling directional relationships separately, we enable the low-quality node to benefit from the high-quality node's information while mitigating the negative impact in the reverse direction.

Although a few works [30, 34, 35] have been proposed to learn asymmetric graphs by generating directed edges $\mathbf{S}_{ij}$ from the labeled node v_j to unlabeled node v_i and constraining $\mathbf{S}_{ji} = 0$, which facilitates the propagation of label information and avoids introducing inconsistency to labeled nodes. However, these methods completely rely on annotated labels and fail to learn reasonable asymmetric connections between the vast majority of unlabeled nodes.

Given the flaws of existing methods, we argue for the necessity of incorporating node uncertainty into graph structure learning to
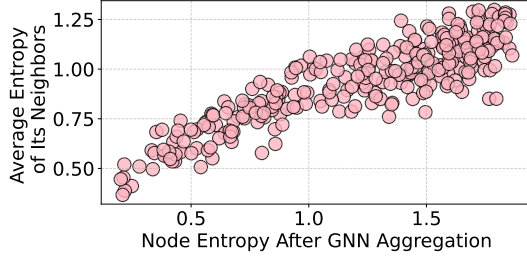
Figure 3: Visualization of node entropy after GNN aggregation (i.e., u_i) alongside the average entropy of its neighbors (i.e., $\sum_{v_j \in \mathcal{N}(v_i)} \hat{A}_{ij} u_j'$) on Cora dataset.

learn an optimal asymmetric structure. We propose the uncertainty-aware graph structure learning (UnGSL) method to enhance GSL models, which leverage learnable node-wise thresholds to identify high-uncertainty neighbors and adaptively reduce directional edge weights from those low-quality neighbors.

3 Methodology

In this section, we first conduct theoretical and empirical analyses to demonstrate the detrimental impact of neighbors with high uncertainty levels on GNN learning (Subsection 3.1). We then present the proposed uncertainty-aware graph structure learning method in detail (Subsection 3.2).

3.1 Analyses on the Impact of Neighbor Uncertainty

Aggregating unclear or ambiguous information can intuitively disrupt the learning process of target nodes, thereby negatively affecting the performance of Graph Neural Networks (GNNs). In this section, we aim to conduct both theoretical and empirical analyses to substantiate this claim. To begin, we introduce several formal concepts to facilitate these analyses.

Semi-supervised Node Classification Task. For convenience, we refer to recent analytical work on GNNs [31] and focus our theoretical analysis on the semi-supervised node classification task, which is the most common and widely studied scenario. Nevertheless, at the end of this section, we will also discuss how our method can be adapted to the unsupervised learning scenario. Following the definitions in [31], we consider a K-class classification problem and employ a linear classification model. Formally, the classification logits can be expressed as:

$$\mathbf{O} = \mathbf{D}^{-1}(\mathbf{A} + \mathbf{I})\mathbf{X}\mathbf{W}, \tag{5}$$

where $\mathbf{W} \in \mathbb{R}^{d \times K}$ is the linear classification matrix. Assume the logit in matrix $\mathbf{O}$ are bounded by the scalar 1, *i.e.*, $\max |\mathbf{O}_{ij}| < 1$. From a local perspective for node v_i, its probabilities can be formulated as :

$$p_i = \frac{\mathbf{O}_i + \mathbf{1}_K}{\sum_{j=1}^{K}(\mathbf{O}_{ij} + 1)}, \tag{6}$$

where $\mathbf{1}_K$ is a K-dimensional all-ones vector. Here we simply omit the nonlinear exponential function in the softmax, given that our primary focus is on the impact of aggregation on node uncertainty and the exponential function mainly serves to generate positive values. This simplification has been adopted in prior work [48, 50], and it can also be considered a first-order Taylor approximation.

Uncertainty Estimation via Entropy. Entropy measures the information uncertainty within a probability distribution [1]. For node classification tasks, the entropy of the classification probabilities reflects the classifier's certainty in assigning the node representation to a specific class. A high-entropy node indicates that its representation carries significant uncertainty, making it challenging for the classifier to reach a confident decision. Aggregating information from such nodes can poison the target node's representation, hindering the generation of accurate predictions. Given probabilities p_i of node v_i, its entropy is defined as:

$$u_i = -\sum_{k=1}^{K} p_{ik} \log(p_{ik}). \tag{7}$$

We further discuss the uncertainty metric for unsupervised learning scenario at the end of this section.

Formally, to demonstrate the impact of the uncertainty of neighbors along GNNs, we have the following proposition, with detailed proof provided in Appendix A:

PROPOSITION 1. *Define the logits of the initial node feature matrix as $\mathbf{O}' = \mathbf{X}\mathbf{W}$. For a given node v_i, let u_i denote its entropy after GNN aggregation. $\forall v_j \in \mathcal{N}(v_i)$, let u_j' denote the entropy of its initial features. Then the entropy of v_i and the entropy of $v_j \in \mathcal{N}(v_i)$ satisfy the following inequality:*

$$u_i \geq \sum_{v_j \in \mathcal{N}(v_i)} \eta_j u_j', \tag{8}$$

where

$$\eta_j = \frac{\sum_{k=1}^{K} \hat{\mathbf{A}}_{ij}(O_{jk}' + 1)}{\sum_{v_j \in \mathcal{N}(v_i)} \sum_{k=1}^{K} \hat{\mathbf{A}}_{ij}(O_{jk}' + 1)}. \tag{9}$$

For coefficient η_j, we have $0 < \eta_j < 1$ and $\sum_{v_j \in \mathcal{N}(v_i)} \eta_j = 1$.

Discussion. Proposition 1 establishes the bounded relationship between the uncertainty of a targe node after aggregation and the uncertainty of its neighbors. It suggests blindly aggregating information from high-uncertainty nodes would lift the lower bound of the uncertainty for the target node. Removing or mitigating the impact of these high-uncertainty nodes, may significantly decrease the uncertainty of the target node, leading to performance gains.

Empirical Analyses. We conduct simple experiments to empirically demonstrate that aggregating neighbors with higher uncertainty would result in a high-uncertainty node. Specifically, we train the model with 1-layer GCN and linear classifier on given datasets. Then we visualize the entropy of a node after aggregation (*i.e.*, u_i) alongside the average entropy of its neighbors (*i.e.*, $\sum_{v_j \in \mathcal{N}(v_i)} \hat{\mathbf{A}}_{ij} u_j'$). As shown in Figure 3, we observe a strong positive correlation between the entropy of a node after aggregation and the average entropy of its neighbors, which substantiates our proposition (results on more datasets are provided in Appendix D.1).

The above analyses clearly illustrates the impact of node uncertainty in aggregation process in GNNs. Blindly connecting and aggregating information from nodes with high uncertainty may undermine the performance of the nodes themselves. It is therefore important to consider node uncertainty in graph structure learning to learn a reasonable asymmetric structure. Specifically, one can prevent a node falling into a high-uncertainty region by weakening

its connections to neighbors with high uncertainty. Meanwhile, it can receive more stable information via strengthened connections with low-uncertainty nodes.

3.2 Uncertainty-aware Graph Structure Learning

Given the importance of considering uncertainty in graph structure learning, we propose the simple yet novel uncertainty-aware graph structure learning (UnGSL) method that leverages learnable node-wise thresholds to distinguish low-quality neighbors from high-quality ones and adaptively refines edges based on their uncertainty levels. Beyond similarity, we introduce uncertainty to guide the learning of the refined adjacency matrix. Specifically, the uncertainty estimated through entropy is transformed into confidence scores. Afterwards, UnGSL applies learnable node-wise thresholds to split neighbors with different levels of confidence into two groups (*i.e.*, high-confidence and low confidence) for each node. Finally, it amplifies edge weights from confident neighbors while reducing edge weights from uncertain ones. In summary, we refine the generation of adjacent matrix of existing GSL from Equation (4) into Equation (10):

$$\hat{S}_{ij} = S_{ij} \cdot \psi(e^{-u_j} - \varepsilon_i), \tag{10}$$

Here u_j is entropy of node v_j estimated during pretraining stage, ε_i is the learnable node-wise threshold of node v_i, $\psi(\cdot)$ is an activate function, S is the generated adjacency matrix of the GSL model.

Equation (10) first normalizes node uncertainty into confidence within the interval $(0, 1)$, and then leverages node-wise learnable thresholds ε to distinguish low-confidence neighbors (*i.e.*, $e^{-u_j} < \varepsilon_i$) from high-confidence ones (*i.e.*, $e^{-u_j} \geq \varepsilon_i$) and adaptively refines the corresponding edges using the following activation function:

$$\psi(x) = \begin{cases} \tau \cdot s(x), & x \geq 0, \\ \beta, & x < 0, \end{cases} \tag{11}$$

where $s(\cdot)$ is the sigmoid function, τ is a hyperparameter amplifies the edge weights from high-confidence neighbors, β is a hyperparameter that controls the reduction of edge weights from low-confidence neighbors.

For the training of UnGSL, we employ the same loss function (*i.e.*, Equation (2)) and the same training procedure as the original GSL model, except we modify the generation of the adjacency matrix from Equation (4) to Equation (10), *i.e.*, using $\hat{S}$ to replace the generated adjacency matrix S in existing GSL methods. Therefore, UnGSL can be seamlessly integrated into existing GSL methods with minimal additional computational cost.

Implementation Details. We begin by selecting a specific GSL method that UnGSL aims to enhance , pretraining it to obtain the classifier and estimating uncertainty using entropy (Equation (7)). Next, the chosen GSL method is re-trained, during which Equation (10) is employed to generate the adjacency matrix. Finally, the learned graph structure is leveraged for downstream applications.

3.3 Discussion

Based on the above formulation, we next discuss several key advantages of UnGSL:

Uncertainty-aware. UnGSL considers the information quality of nodes during the structure modeling process, facilitating the

learning of an optimal graph that mitigates the negative impact of high-uncertainty neighbors in GNN learning.

Model-agnostic. UnGSL refines the learned adjacency matrix S by utilizing uncertainty-aware weights. In essence, UnGSL just replaces the calculation of the adjacency matrix from Equation (4) to Equation (10) for GSL methods. Therefore, UnGSL can be seamlessly integrated into existing GSL methods to further enhance their ability to learn graphs, improving the performance of GNNs on downstream tasks. Comprehensive experiments supporting this can be found in Subsection 4.2 and Subsection 4.4.

Asymmetric Graph Structure. UnGSL proposes learning an asymmetric graph, where the edge weights between nodes differ based on their uncertainty levels. Specifically, UnGSL weakens edges from uncertain nodes to confident nodes, mitigating the impact of inferior information. Meanwhile it enhances the edge in the opposite direction to improve the representations of uncertain nodes.

Notably, several methods that construct directed edges from labeled nodes to unlabeled nodes [30, 35] can be viewed as specific cases of UnGSL, as the embeddings of labeled nodes are directly optimized during training and therefore more likely to exhibit lower uncertainty. In contrast, UnGSL models the asymmetric relationships among the predominantly unlabeled nodes, leading to superior performance. We further empirically validate this in Subsection 4.2.

Efficiency. UnGSL contains only n learnable parameters and refines the existing edges of the given graph without generating new ones. The additional operations introduced by UnGSL have a complexity of $O(n + m)$, where n and m denote the number of nodes and edges in the graph. Therefore, UnGSL imposes minimal computational costs on the base GSL models. We provide empirical evidence to demonstrate the efficiency of UnGSL in Subsection 4.6.

Adaptive to Unsupervised Scenarios. Despite the mainstream focus of GSL research on supervised learning, our approach can be generalized to unsupervised GSL scenarios. The main challenge lies in determining an appropriate uncertainty metric for unsupervised GSL method. We suggest employing the self-supervised structure learning loss (*i.e.*, node contrastive learning loss [29]) as a proxy for uncertainty, which can be formulated as:

$$u_i = \frac{1}{2}(l(z_i, \tilde{z}_i) + l(\tilde{z}_i, z_i), \tag{12}$$

where

$$l(z_i, \tilde{z}_i) = -\log \frac{e^{\text{sim}(z_i, \tilde{z}_i)/t}}{\sum_{k=1}^{n} e^{\text{sim}(z_i, \tilde{z}_k)/t}}. \tag{13}$$

Here $\text{sim}(\cdot)$ is the cosine similarity function. z_i and $\tilde{z}_i$ are the embeddings of node v_i learned by GNN from target graph and augmented graph, respectively. $\sum_{k=1}^{n} e^{\text{sim}(z_i, \tilde{z}_k)/t}$ is cumulative similarity between z_i and embeddings of different nodes in the augmented graph. t is the temperature parameter. The GCL loss can measure the invariance of node representations to feature or structure perturbations, where this invariance can be interpreted as the uncertainty of nodes with respect to their original features and structure.

4 Experiments

In this section, we conduct experiments to evaluate the effectiveness of the proposed UnGSL strategy. Our experiments aim to answer the following research questions:

Table 1: Node classification accuracy±std comparsion(%). Each experiment is repeated 10 times with different random seeds. "OOM" denotes out of memory. The top-performing results are marked in bold.

Model	Cora	Citeseer	Pubmed	BlogCatalog	Roman-empire	Flickr	Ogbn-arxiv
GRCN	84.70±0.31	72.49±0.77	78.94±0.16	76.17±0.23	44.29±0.28	59.55±0.31	OOM
GRCN+UnGSL	**85.84±0.51**	**73.88±0.55**	**79.59±0.48**	**76.78±0.11**	**52.54±0.31**	**63.85±0.22**	OOM
PROGNN	80.39±0.41	67.94±0.52	OOM	76.17±0.22	OOM	61.74±0.24	OOM
PROGNN+UnGSL	**81.86±0.55**	**69.66±0.27**	OOM	**76.82±0.19**	OOM	**61.92±0.43**	OOM
PROSE	81.1±0.45	72.3±0.37	83.3±0.71	75.31±0.17	55.61±0.34	59.77±1.13	71.22±0.31
PROSE+UnGSL	**81.90±0.31**	**73.10 ±0.32**	**83.86±0.30**	**75.77±0.36**	**56.17±0.31**	**64.72±0.97**	**71.46±0.12**
IDGL	84.50±0.5	72.49±0.67	82.83±0.33	89.66±0.28	46.67±0.56	85.77±0.16	70.45 ± 0.36
IDGL+UnGSL	**84.90±0.42**	**73.74±1.01**	**83.33±0.32**	**92.13±0.18**	**47.05±0.59**	**86.42±0.35**	**71.02±0.12**
SLAPS	72.89±1.02	70.05±0.83	70.96±0.99	91.62±0.39	65.35±0.45	83.89±0.70	OOM
SLAPS+UnGSL	**74.28±0.95**	**72.08±0.94**	**72.25±1.48**	**91.89±0.41**	**66.20±0.33**	**84.92±0.68**	OOM
SUBLIME	83.30±1.04	72.36±0.68	80.41±0.69	95.20±0.21	63.48±0.53	88.68±0.23	71.37±0.13
SUBLIME+UnGSL	**84.24±0.91**	**74.34±0.73**	**80.84±0.92**	**96.18±0.38**	**65.45±0.32**	**89.23±0.20**	**71.82±0.15**

Table 2: Performance comparison between the CUR decomposition and UnGSL modules.

Model	Cora	Citeseer	Roman-empire
GRCN+CUR	84.89±0.22	73.35±0.46	44.04±0.28
GRCN+UnGSL	**85.84±0.51**	**73.88±0.55**	**52.54±0.31**
IDGL+CUR	84.73±0.23	72.93±0.85	OOM
IDGL+UnGSL	**84.90±0.42**	**73.74±1.01**	**47.05±0.59**

- **RQ1:** Does the integration of UnGSL into existing GSL methods lead to performance improvements?
- **RQ2:** What is the impact of key configurations (*e.g.,* node-wise thresholds, asymmetric graphs, hyperparameters β and τ) on UnGSL performance?
- **RQ3:** Can UnGSL enhance the robustness of GNNs against structural noise, feature noise and lable noise?
- **RQ4:** How well does UnGSL generalize across different GNN backbones?
- **RQ5:** Does UnGSL introduce significant additional computational overhead?

4.1 Experiment Settings

4.1.1 Datasets. To comprehensively evaluate UnGSL's performance on node classification, we follow previous works [47, 51] and select 7 commonly used datasets, including four homophilous citation datasets [20, 33] (Cora, Citeseer, Pubmed and Ogbn-arxiv) and three heterophilous datasets (Blogcatalog [21] , Roman-Empire [32] and Flickr [21]). The chosen datasets cover a wide range of homophily levels and graph sizes, allowing us to demonstrate UnGSL's effectiveness under various conditions. For a fair comparison, we strictly follow the data split settings used in the newly proposed benchmark for GSL [51]. Detailed statistics of these datasets are provided in Appendix B.

4.1.2 Baselines. To demonstrate UnGSL's generalizability across different GSL models, we select 6 state-of-the-art GSL algorithms corresponding to the chosen datasets as baselines, including supervised models (GRCN [42], PROGNN [23], IDGL [6], PROSE [41],

SLAPS [13]), and a self-supervised model (SUBLIME [29]). Detailed information of these baselines is provided in Appendix C. For all models, we report the average performance and standard deviations of 10 runs with different random seeds.

4.1.3 Configuration. With regard to hyperparameter settings, only the hyperparameter β and the learning rate of the threshold ε require fine-tuning. The threshold ε is consistently initialized as a random number in the range [0, 1] across various datasets. The hyperparameter τ is simply fixed as a constant value of 2. The hyperparameter β can be quickly tuned using parameter search tools like Bayesian optimization within the range of [0.001,1]. We optimized the UnGSL using Adam optimizer, with the learning rate selected from range [0.0001, 0.01]. For detailed hyperparameter settings please refer to Appendix D.2.

For all baselines, we strictly adhere to their original settings for hyperparameter tuning to ensure that they attain best performance. All GSL methods are evaluated based on the performance of GNNs on downstream tasks when using the learned structure. We also consider cross-architecture scenarios in Subsection 4.5, where GSL training and downstream tasks use different GNN architectures.

4.2 Main Results (RQ1)

4.2.1 Comparison to Vanilla GSL Models. Table 1 presents the experimental results of the UnGSL module applied to various GSL models. We can observe that: 1) UnGSL significantly improves the node classification accuracy for all GSL models across all datasets with an average increase of 2.07%, achieving new state-of-the-art performance in the GSL literature. The improvement brought by UnGSL is particularly pronounced on GRCN [47], achieving an average improvement of 5.12%. These results empirically demonstrate UnGSL's effectiveness in further denoising the learned graph structure from the perspective of uncertainty, resulting in more accurate predictions. 2) For the self-supervised GSL model SUBLIME [29], UnGSL continues to achieve higher accuracy by utilizing contrastive loss in Equation (12) as a proxy of uncertainty estimation to guide structure refinement, resulting in an average improvement of 1.40%.

Table 3: Ablation study on the UnGSL when integrating with GRCN model [47].

Method	Cora	Citeseer	Roman-empire
Fixed ε	85.23±0.15	73.2±0.96	44.72±0.14
Symmetrize $\hat{S}$	85.03±0.40	73.74±0.49	52.28±0.025
UnGSL	**85.84±0.51**	**73.88±0.55**	**52.54±0.31**
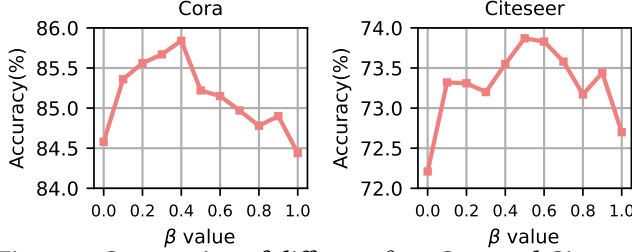
Figure 4: Comparsion of different β on Cora and Citeseer datasets.

4.2.2 Comparison to the Label-oriented Directed GSL Module. Here we compare the performance of UnGSL with CUR decomposition [30], another GSL module proposed recently to learn asymmetric graph structure by constructing directed edges from labeled nodes to unlabeled nodes. Table 2 shows the experiment results of UnGSL and CUR decomposition. We can observe that: 1) UnGSL consistently outperforms CUR decomposition on supervised GSL methods. 2) UnGSL demonstrates better scalability with large-scale graphs compared to CUR decomposition. For example, IDGL+UnGSL is applicable to the Roman-Empire dataset and further enhances accuracy, while IDGL+CUR encounters an out-of-memory problem.

4.3 Ablation Studies (RQ2)

4.3.1 Effects of Adaptive Threshold ε. To validate the effectiveness of the learnable threshold, we consider a variant where we fixed the ε in UnGSL during training phase. Specifically, for each node, we select a fixed proportion of the most uncertain neighbors, as high-uncertainty neighbors. This proportion is consistent across all nodes. Next, we reweight edges by Equation (10). As shown in Table 3, the learnable ε outperforms the fixed ε across 3 datasets. The results indicate that the learnable threshold effectively differentiates high-uncertainty neighbors from low-uncertainty neighbors.

4.3.2 Superiority of Asymmetric Graph. To demonstrate the superiority of the asymmetric edges constructed by UnGSL, we symmetrize the graph refined by UnGSL during training phase, generating an symmetric graph structure. As shown in Table 3, symmetrizing graph $\hat{S}$ of UnGSL degrades its performance on node classification. The underlying reason is that the symmetrization operation may perturb the graph by weakening edges from low-uncertainty neighbors and strengthening edge weights from high-uncertainty neighbors, which violates the core mechanism of UnGSL.

4.3.3 Analysis on Hyperparameter β. To explore the role of β in UnGSL's activation function $\psi(\cdot)$, we assign different values to β from the interval $[0, 1]$ and evaluate the corresponding performance on the GRCN model [47]. As shown in Figure 4, we can observe that: (1) UnGSL enhances the accuracy of the GSL model by reducing edge weights with low-uncertainty neighbors using an appropriate β. (2) UnGSL with positive β consistently outperform

Table 4: Comparsion of different τ on Cora and Citeseer datasets.

Method	$\tau = 1$	$\tau = 2$	$\tau = 3$
Cora	85.12±0.47	85.84±0.51	84.87±0.55
Citeseer	72.77±0.90	73.88±0.55	73.09±0.42

Table 5: Robustness analysis with random structural noise injection on Cora dataset.

Methods	Edge Deletion Level					Edge Addition Level				
	0%	20%	40%	60%	80%	0%	20%	40%	60%	80%
GRCN	84.70	83.00	79.87	78.47	75.17	84.70	78.03	75.13	73.20	69.43
GRCN+UnGSL	85.84	84.80	81.83	80.20	76.32	85.84	79.17	76.27	74.42	70.60
Improve(%)	1.34%	2.16%	2.45%	2.21%	1.53%	1.34%	1.46%	1.65%	1.66%	1.69%
IDGL	84.50	81.60	80.31	76.57	72.18	84.50	78.59	77.24	74.43	73.11
IDGL+UnGSL	84.90	82.88	81.41	77.83	73.24	84.90	80.02	78.77	76.25	75.31
Improve(%)	0.47%	1.57%	1.38%	1.65%	1.47%	0.47%	1.82%	1.98%	2.45%	3.01%

UnGSL with $\beta = 0$. The results indicate that high-uncertainty neighbors still contain valuable information that enhances the node's representation, and removing connections from high-uncertainty nodes blindly may lead to information loss. Results for additional GSL models are provided in Appendix D.3.

4.3.4 Analysis on Hyperparameter τ. We conduct additional experiments to evaluate the model's performance with different values of τ on Cora and Citeseer datasets using the base GSL model GRCN [47]. The results are presented in Table 4. As can be seen, as τ increases, the performance initially improves and then declines. Hyperparameter τ controls influence of high-confidence neighbors. While fine-tuning τ could potentially enhance model performance, we find that simply setting $\tau = 2$ is sufficient to achieve good results.

4.4 Robustness Analysis (RQ3)

To evaluate the robustness of UnGSL under structural noise, we compare the accuracy of the base GSL model and GSL+UnGSL under different noise levels and calculate the relative improvement achieved by GSL+UnGSL. Specifically, we randomly add or remove edges from the original dataset. We conduct experiments on the Cora dataset, and the results are presented in Table 5. We can observe that: 1) UnGSL consistently enhances the performance of GSL models across different noise levels. 2) With increasing levels of noise, UnGSL achieves greater relative improvements. For example, in the random edge addition scenario, IDGL+UnGSL achieves a 3.01% improvement under 80% edge perturbation, which is significantly higher than the 0.47% improvement observed at the 0% noise level. In summary, the results suggest that UnGSL enhances robustness against structural noise. We also validate UnGSL's robustness against feature noise and label noise, and the results are presented in Appendix D.4.

4.5 Generalizability on GNN Models (RQ4)

We further consider the scenario where GSL training and downstream tasks use different GNN backbones. We evaluate the generalizability of the learned structures generated by the GSL+UnGSL on several other GNN models, including SGC [44], APPNP [15], GAT [38], and JKNet [46]. The results are presented in Table 6.

Table 6: Generalizability of UnGSL with different backbones on Cora dataset. The value in bold signifies the top-performing result.

Methods	SGC	APPNP	GAT	JKNet
GRCN	84.40±0.00	84.00±0.15	81.45±1.04	83.73±1.33
GRCN+UnGSL	**84.80±0.10**	**84.40±0.61**	**82.45±1.04**	**84.53±1.38**
IDGL	78.00±0.00	82.13±0.32	79.87±1.01	76.23±1.19
IDGL+UnGSL	**78.20±0.12**	**82.50±0.61**	**80.23±0.49**	**77.10±0.50**

Table 7: Convergence time and GPU memory consumption of different models on Ogbn-arxiv dataset.

Model	Time(s)	Memory(MB)
IDGL	105	14,566
IDGL+UnGSL	113	15,338
SUBLIME	807	14,586
SUBLIME+UnGSL	886	14,678

We observe that the graphs produced by GSL+UnGSL improve the prediction of various GNN models compared to those generated by vanilla GSL. Overall, UnGSL demonstrates its generalizability in enhancing performance across different GNN architectures. Results for additional datasets are provided in Appendix D.5.

4.6 Efficiency Analysis (RQ5)

We analyze the time and memory efficiency of UnGSL on the Ogbn-Arxiv dataset, a large-scale graph with 169,343 nodes and 1,157,799 edges. To assess time efficiency, we evaluate the algorithms by measuring the time taken to converge, *i.e.,* to reach optimal performance on the validation set. As shown in Table 7, UnGSL slightly increases convergence time (8.71% on average) and training memory (2.97% on average) compared to the base GSL models. The results demonstrate UnGSL's efficiency when applied to GSL models.

5 Related Work

5.1 Graph Neural Networks

Graph neural networks (GNNs) are powerful models for learning node representations from graph data. Existing GNNs can be categorized as spectral GNNs and spatial GNNs. Spectral GNNs leverage eigenvectors and eigenvalues within the graph Laplacian matrix to design graph signal filters in the spectral domain [2, 8, 18, 25]. Spatial GNNs [10, 17, 24, 38, 44, 45] simplified spectral graph filter [8] using first-order approximation, which aggregates features from neighboring nodes in the spatial graph to generate node embeddings. However, existing GNNs assume that the input graph structure is sufficiently clean for learning, whereas real-world graphs are often noisy and incomplete, which limits GNN performance on downstream tasks [4, 16]. In this paper, we propose uncertainty-aware graph structure learning, which effectively denoises the graph structure to alleviate the above limitation.

5.2 Graph Structure Learning

Graph Structure Learning (GSL) aims to learn an optimal graph that improves the accuracy and robustness of Graph Neural Networks (GNNs) on downstream tasks. Early GSL methods [14, 23] directly treat the target adjacency matrix as learnable parameters, incurring

substantial computational overhead and optimization challenge. Mainstream GSL methods [6, 22, 41, 47] learn edge weights based on node-pair embedding similarities, employing various metrics such as cosine similarity [22, 41], inner product [47] or neural networks [28]. These methods aim to increase graph homophily and typically achieve state-of-the-art performance, as these metrics can capture nodes with similar semantics. However, these embedding-based GSL methods suffer from two limitations: First, they construct edges solely based on embedding similarities while neglecting node uncertainty, which may introduce inferior information that poison the target node's embedding. Although some works [12, 28] incorporate predictive uncertainty in structure learning, they only use it for cross-view structure fusion without distinguishing nodes of varying uncertainty levels within a graph. Second, embedding-based methods impose bidirectional edges between nodes, disregarding their unequal influence due to varying levels of uncertainty. Recently, several methods propose constructing directed edges from labeled to unlabeled nodes to facilitate the propagation of supervision signals [30, 34, 35]. However, these methods fail to learn reasonable asymmetric connections between the vast majority of unlabeled nodes. Different from the above works, we propose an uncertainty-aware structure learning (UnGSL) strategy that learns node-wise thresholds to differentiate low-uncertainty from high-uncertainty neighbors and adaptively refine the corresponding edges.

5.3 Uncertainty in GNNs

GNNs inevitably present uncertainty towards their predictions, leading to unstable and erroneous prediction results [40]. In recent years, uncertainty in GNNs has been widely researched to adapt various tasks, including out-of-distribution (OOD) detection [36, 48, 50], trustworthy GNN learning [19, 43], and GNN modeling [37]. However, these uncertainty-based GNNs assume that the input graph structure is clean and primarily focus on incorporating uncertainty into the model architecture. In contrast, our proposed uncertainty-aware graph structure learning aims to refine the edge based on node uncertainty, assisting in improving graph quality.

6 Conclusion

In this paper, we first conduct theoretical and empirical analyses to demonstrate the detrimental impact of neighbors with high uncertainty on GNN learning. Building on this, we propose the UnGSL strategy, a lightweight plug-in module that integrates seamlessly with state-of-the-art GSL models and boosts performance with minimal extra computational overhead. UnGSL learns node-wise thresholds to differentiate between low-uncertainty and high-uncertainty neighbors, and adaptively refines the graph based on each node's uncertainty level. Experiments demonstrate that UnGSL consistently enhances the performance and robustness of GSL models. In the future, we plan to explore more effective uncertainty metrics to accurately identify uncertain nodes in graph structure learning.

References

[1] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. 2021. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion* 76 (2021), 243–297.

[2] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2013. Spectral networks and locally connected networks on graphs. In *2nd International Conference on Learning Representations, ICLR 2014*. Curran Associates Inc., Red Hook; NY; United States.

[3] Hao Chen, Yuanchen Bei, Qijie Shen, Yue Xu, Sheng Zhou, Wenbing Huang, Feiran Huang, Senzhang Wang, and Xiao Huang. 2024. Macro graph neural networks for online billion-scale recommender systems. In *Proceedings of the ACM on Web Conference 2024*. Association for Computing Machinery, New York, NY, USA, 3598–3608.

[4] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2023. Bias and debias in recommender system: A survey and future directions. *ACM Transactions on Information Systems* 41, 3 (2023), 1–39.

[5] Sirui Chen, Jiawei Chen, Sheng Zhou, Bohao Wang, Shen Han, Chanfei Su, Yuqing Yuan, and Can Wang. 2024. SIGformer: Sign-aware Graph Transformer for Recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, New York, NY, USA, 1274–1284.

[6] Yu Chen, Lingfei Wu, and Mohammed Zaki. 2020. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. *Advances in Neural Information Processing Systems* 33 (2020), 19314–19326.

[7] Imre Csiszár, Paul C Shields, et al. 2004. Information theory and statistics: A tutorial. *Foundations and Trends® in Communications and Information Theory* 1, 4 (2004), 417–528.

[8] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in Neural Information Processing Systems* 29 (2016).

[9] Chenhui Deng, Zichao Yue, and Zhiru Zhang. 2024. Polynormer: Polynomial-Expressive Graph Transformer in Linear Time. In *The Twelfth International Conference on Learning Representations*. Curran Associates Inc., Red Hook; NY; United States. https://openreview.net/forum?id=hmv1LpNfXa

[10] Mucong Ding, Tahseen Rabbani, Bang An, Evan Wang, and Furong Huang. 2022. Sketch-GNN: Scalable graph neural networks with sublinear training complexity. *Advances in Neural Information Processing Systems* 35 (2022), 2930–2943.

[11] Hande Dong, Jiawei Chen, Fuli Feng, Xiangnan He, Shuxian Bi, Zhaolin Ding, and Peng Cui. 2021. On the equivalence of decoupled graph convolution network and label propagation. In *Proceedings of the Web Conference 2021*. Association for Computing Machinery, New York,NY,United States, 3651–3662.

[12] Liang Duan, Xiang Chen, Wenjie Liu, Daliang Liu, Kun Yue, and Angsheng Li. 2024. Structural Entropy Based Graph Structure Learning for Node Classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. AAAI Press, Washington DC,USA, 8372–8379.

[13] Bahare Fatemi, Layla El Asri, and Seyed Mehran Kazemi. 2021. Slaps: Self-supervision improves structure learning for graph neural networks. *Advances in Neural Information Processing Systems* 34 (2021), 22667–22681.

[14] Luca Franceschi, Mathias Niepert, Massimiliano Pontil, and Xiao He. 2019. Learning discrete structures for graph neural networks. In *International Conference on Machine Learning*. PMLR, Curran Associates Inc., Long Beach, California, USA, 1972–1982.

[15] Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. 2019. Combining Neural Networks with Personalized PageRank for Classification on Graphs. In *International Conference on Learning Representations*. Curran Associates Inc., Red Hook; NY; United States. https://openreview.net/forum?id=H1gL-2A9Ym

[16] Simon Geisler, Tobias Schmidt, Hakan Şirin, Daniel Zügner, Aleksandar Bojchevski, and Stephan Günnemann. 2021. Robustness of graph neural networks at scale. *Advances in Neural Information Processing Systems* 34 (2021), 7637–7649.

[17] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*. PMLR, JMLR.org, Sydney NSW Australia, 1263–1272.

[18] Mingguo He, Zhewei Wei, and Ji-Rong Wen. 2022. Convolutional neural networks on graphs with chebyshev approximation, revisited. *Advances in Neural Information Processing Systems* 35 (2022), 7264–7276.

[19] Hans Hao-Hsun Hsu, Yuesong Shen, Christian Tomani, and Daniel Cremers. 2022. What makes graph neural networks miscalibrated? *Advances in Neural Information Processing Systems* 35 (2022), 13775–13786.

[20] 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. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.). Curran Associates Inc., Red Hook; NY; United States. https://proceedings.neurips.cc/paper/2020/hash/fb60d411a5c5b72b2e7d3527cfc84fd0-Abstract.html

[21] Xiao Huang, Jundong Li, and Xia Hu. 2017. Label informed attributed network embedding. In *Proceedings of the tenth ACM International Conference on Web Search and Data Mining*. Association for Computing Machinery, New York, NY, USA, 731–739.

[22] Yeonjun In, Kanghoon Yoon, Kibum Kim, Kijung Shin, and Chanyoung Park. 2024. Self-Guided Robust Graph Structure Refinement. In *Proceedings of the ACM on Web Conference 2024*. Association for Computing Machinery, New York, NY, USA, 697–708.

[23] Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, and Jiliang Tang. 2020. Graph structure learning for robust graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data mining*. Association for Computing Machinery, New York, NY, USA, 66–74.

[24] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*. Curran Associates, Inc., Red Hook; NY; United States. https://openreview.net/forum?id=SJU4ayYgl

[25] Qimai Li, Xiaotong Zhang, Han Liu, Quanyu Dai, and Xiao-Ming Wu. 2021. Dimensionwise separable 2-D graph convolution for unsupervised and semi-supervised learning on graphs. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. Association for Computing Machinery, New York, NY, USA, 953–963.

[26] Zhixun Li, Liang Wang, Xin Sun, Yifan Luo, Yanqiao Zhu, Dingshuo Chen, Yingtao Luo, Xiangxin Zhou, Qiang Liu, Shu Wu, Liang Wang, and Jeffrey Xu Yu. 2023. GSLB: The Graph Structure Learning Benchmark. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. Curran Associates Inc., Red Hook; NY; United States. https://openreview.net/forum?id=xT3i5GS3zU

[27] Juncheng Liu, Bryan Hooi, Kenji Kawaguchi, Yiwei Wang, Chaosheng Dong, and Xiaokui Xiao. 2024. Scalable and Effective Implicit Graph Neural Networks on Large Graphs. In *The Twelfth International Conference on Learning Representations*. Curran Associates, Inc., Red Hook; NY; United States. https://openreview.net/forum?id=QcMdPYBwTu

[28] Nian Liu, Xiao Wang, Lingfei Wu, Yu Chen, Xiaojie Guo, and Chuan Shi. 2022. Compact graph structure learning via mutual information compression. In *Proceedings of the ACM Web Conference 2022*. Association for Computing Machinery, New York, NY, USA, 1601–1610.

[29] Yixin Liu, Yu Zheng, Daokun Zhang, Hongxu Chen, Hao Peng, and Shirui Pan. 2022. Towards unsupervised deep graph structure learning. In *Proceedings of the ACM Web Conference 2022*. Association for Computing Machinery, New York, NY, USA, 1392–1403.

[30] Jianglin Lu, Yi Xu, Huan Wang, Yue Bai, and Yun Fu. 2024. Latent graph inference with limited supervision. *Advances in Neural Information Processing Systems* 36 (2024).

[31] Yao Ma, Xiaorui Liu, Neil Shah, and Jiliang Tang. 2022. Is Homophily a Necessity for Graph Neural Networks?. In *International Conference on Learning Representations*. Curran Associates, Inc., Austria. https://openreview.net/forum?id=ucASPPD9GKN

[32] 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?. In *The Eleventh International Conference on Learning Representations*. Curran Associates, Inc., Kigali, Rwanda.

[33] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI magazine* 29, 3 (2008), 93–93.

[34] Zixing Song, Yifei Zhang, and Irwin King. 2022. Towards an optimal asymmetric graph structure for robust semi-supervised node classification. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Dining*. Association for Computing Machinery, New York, NY, USA, 1656–1665.

[35] Zixing Song, Yifei Zhang, and Irwin King. 2024. Optimal block-wise asymmetric graph construction for graph-based semi-supervised learning. *Advances in Neural Information Processing Systems* 36 (2024).

[36] Maximilian Stadler, Bertrand Charpentier, Simon Geisler, Daniel Zügner, and Stephan Günnemann. 2021. Graph posterior network: Bayesian predictive uncertainty for node classification. *Advances in Neural Information Processing Systems* 34 (2021), 18033–18048.

[37] Daeho Um, Jiwoong Park, Seulki Park, and Jin young Choi. 2023. Confidence-Based Feature Imputation for Graphs with Partially Known Features. In *The Eleventh International Conference on Learning Representations*. Curran Associates, Inc., Red Hook; NY; United States. https://openreview.net/forum?id=YPKBIILy-Kt

[38] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations*. Curran Associates, Inc., Red Hook; NY; United States. https://openreview.net/forum?id=rJXMpikCZ

[39] Bohao Wang, Jiawei Chen, Changdong Li, Sheng Zhou, Qihao Shi, Yang Gao, Yan Feng, Chun Chen, and Can Wang. 2024. Distributionally Robust Graph-based Recommendation System. In *Proceedings of the ACM on Web Conference 2024*.

Association for Computing Machinery, New York, NY, USA, 3777–3788.

[40] Fangxin Wang, Yuqing Liu, Kay Liu, Yibo Wang, Sourav Medya, and Philip S. Yu. 2024. Uncertainty in Graph Neural Networks: A Survey. *Transactions on Machine Learning Research* (2024). https://openreview.net/forum?id=0e1Kn76HM1

[41] Huizhao Wang, Yao Fu, Tao Yu, Linghui Hu, Weihao Jiang, and Shiliang Pu. 2023. Prose: Graph structure learning via progressive strategy. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, New York, NY, USA, 2337–2348.

[42] Ruijia Wang, Shuai Mou, Xiao Wang, Wanpeng Xiao, Qi Ju, Chuan Shi, and Xing Xie. 2021. Graph structure estimation neural networks. In *Proceedings of the Web Conference 2021*. Association for Computing Machinery, New York, NY, USA, 342–353.

[43] Xiao Wang, Hongrui Liu, Chuan Shi, and Cheng Yang. 2021. Be confident! towards trustworthy graph neural networks via confidence calibration. *Advances in Neural Information Processing Systems* 34 (2021), 23768–23779.

[44] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *International Conference on Machine Learning*. PMLR, Curran Associates, Inc., Long Beach, California, USA, 6861–6871.

[45] Junkang Wu, Wentao Shi, Xuezhi Cao, Jiawei Chen, Wenqiang Lei, Fuzheng Zhang, Wei Wu, and Xiangnan He. 2021. DisenKGAT: knowledge graph embedding with disentangled graph attention network. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. Association for Computing Machinery, New York,NY,United States, 2140–2149.

[46] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning*. PMLR, Curran Associates, Inc., Stockholm, Sweden, 5453–5462.

[47] Donghan Yu, Ruohong Zhang, Zhengbao Jiang, Yuexin Wu, and Yiming Yang. 2021. Graph-revised convolutional network. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part III*. Springer, Springer-Verlag, Berlin, Heidelberg, 378–393.

[48] Linlin Yu, Yifei Lou, and Feng Chen. 2023. Uncertainty-aware Graph-based Hyperspectral Image Classification. In *The Twelfth International Conference on Learning Representations*. OpenReview.net, Vienna, Austria.

[49] Zhen Zhang, Jiajun Bu, Martin Ester, Jianfeng Zhang, Zhao Li, Chengwei Yao, Huifen Dai, Zhi Yu, and Can Wang. 2021. Hierarchical multi-view graph pooling with structure learning. *IEEE Transactions on Knowledge and Data Engineering* 35, 1 (2021), 545–559.

[50] Xujiang Zhao, Feng Chen, Shu Hu, and Jin-Hee Cho. 2020. Uncertainty aware semi-supervised learning on graph data. *Advances in Neural Information Processing Systems* 33 (2020), 12827–12836.

[51] Zhiyao Zhou, Sheng Zhou, Bochao Mao, Xuanyi Zhou, Jiawei Chen, Qiaoyu Tan, Daochen Zha, Yan Feng, Chun Chen, and Can Wang. 2023. OpenGSL: A Comprehensive Benchmark for Graph Structure Learning. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. Curran Associates Inc., Red Hook, NY, USA. https://openreview.net/forum?id=yXLyhKvK4D

A Proof of Proposition 1

To prove Proposition 1, we first introduce the log-sum inequality [7] below.

LEMMA 1 (LOG-SUM INEQUALITY). *Let $a_1, ..., a_n$ and $b_1, ..., b_n$ be non-negative numbers. Denote the sum of all a_is by a and the sum of all b_is by b. Then log-sum inequality states that*

$$\sum_{i=1}^{n} a_i \log \frac{a_i}{b_i} \geq a \log \frac{a}{b}, \tag{14}$$

Proof. $\forall v_i \in \mathcal{V}$, by define the logit $o_i = (\sum_{v_j \in \mathcal{N}(v_i)} \hat{A}_{ij} x_j) W$ and logit $o'_i = x_i W$, where $\sum_{v_j \in \mathcal{N}(v_i)} \hat{A}_{ij} = 1$, we have:

$$o_i = \sum_{v_j \in \mathcal{N}(v_i)} \hat{A}_{ij} o'_j, \tag{15}$$

then for predictive probability vector p_i of o_i, $\forall v_j \in \mathcal{N}(v_i)$, let p'_j denote the classification probabilities of its initial features. We have:

$$\begin{aligned}
p_i &= \frac{o_i + 1_K}{\sum_{k=1}^{K}(o_{ik} + 1)} \\
&= \frac{\sum_{v_j \in \mathcal{N}(v_i)} \hat{A}_{ij}(o'_j + 1_K)}{\sum_{k=1}^{K}(\sum_{v_j \in \mathcal{N}(v_i)} \hat{A}_{ij} o'_{jk} + 1)} \\
&= \frac{\sum_{v_j \in \mathcal{N}(v_i)} p'_j(\hat{A}_{ij} \sum_{k=1}^{K}(o'_{jk} + 1))}{\sum_{v_j \in \mathcal{N}(v_i)} \hat{A}_{ij} \sum_{k=1}^{K}(o'_{jk} + 1)}, \\
&= \sum_{v_j \in \mathcal{N}(v_i)} \eta_j p'_j, \tag{16}
\end{aligned}$$

where, $\sum_{v_j \in \mathcal{N}(v_i)} \eta_j = 1$. Now we focus the l-th element in the probability vector:

$$\begin{aligned}
p_{il} &= \sum_{v_j \in \mathcal{N}(v_i)} \eta_j p'_{jl} \\
\implies p_{il} \log p_{il} &= (\sum_{v_j \in \mathcal{N}(v_i)} \eta_j p'_{jl}) \log(\sum_{v_j \in \mathcal{N}(v_i)} \eta_j p'_{jl}) \\
&= (\sum_{v_j \in \mathcal{N}(v_i)} \eta_j p'_{jl}) \log(\frac{\sum_{v_j \in \mathcal{N}(v_i)} \eta_j p'_{jl}}{\sum_{v_j \in \mathcal{N}(v_i)} \eta_j}) \\
&\leq \sum_{v_j \in \mathcal{N}(v_i)} \eta_j p'_{jl} \log p'_{jl} \\
\implies -\sum_{l=1}^{K} p_{il} \log p_{il} &\geq -\sum_{v_j \in \mathcal{N}(v_i)} \eta_j \sum_{l=1}^{K} p'_{jl} \log p'_{jl} \\
&= \sum_{v_j \in \mathcal{N}(v_i)} \eta_j u'_j, \tag{17}
\end{aligned}$$

which completes the proof.

B Datasets

Table 8 shows the statistics of 7 datasets.

C Baselines

we introduce all GSL models used in the experiment in this section.

- GRCN [47] uses a GCN to extract topological features and compute the similarity between nodes as the edge weights of the learned graph.

- ProGNN [23] treats the graph as a learnable adjacency matrix and optimizes the sparsity, low-rankness, and feature smoothness of the graph structure.
- PROSE [41] identifies influential nodes using PageRank scores and reconstructs the graph structure by connecting these influential nodes.
- IDGL [6] iteratively learns the graph structure and node embeddings, and introduces a node-anchor message-passing paradigm to scale IDGL to large graphs.
- SLAPS [13] proposes learning a denoising autoencoder to filter noisy edges in the graph structure.
- CUR [30] is a model-agnostic structure learning module, which proposes constructing unidirectional edges from unlabeled nodes to labeled nodes via CUR decomposition to facilitate the propagation of supervision signals to unlabeled nodes.
- SUBLIME [29] is an unsupervised GSL model that employs contrastive learning between the learned graph and an augmented graph to enhance the robustness of the graph structure.

D Additional Experimental Results

D.1 Visualization of Node Entropy and Average Neighbor Entropy

We also conduct experiments on more datasets with different homophily characteristics. Figure 5 presents the node entropy after GNN aggregation, along with the average entropy of its neighbors, on the Citeseer and Flickr datasets. We observe a strong positive correlation between the entropy of nodes and that of their neighbors across different datasets.

(a) Citeseer dataset
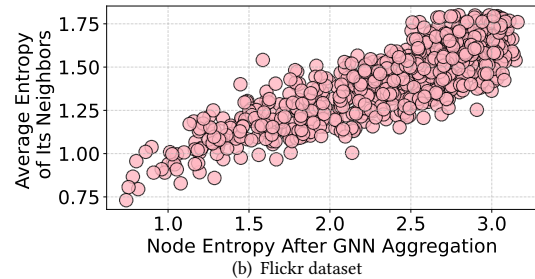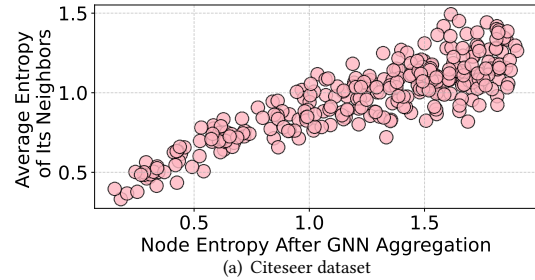

(b) Flickr dataset

Figure 5: Visualization of node entropy after GNN aggregation (i.e., u_i) alongside the average entropy of its neighbors (i.e., $\sum_{v_j \in \mathcal{N}(v_i)} \hat{A}_{ij} u'_j$) on Citeseer and Flickr datasets.

Table 8: Detailed statistics of node classification datasets.

Dataset	#Nodes	#Edges	#Feat.	#Avg.degree	#Homophily
Cora	2,708	5,278	1,433	3.9	0.81
Citeseer	3,327	4,552	3,703	2.7	0.74
Pubmed	19,717	44,324	500	4.5	0.80
BlogCatalog	5,196	171,743	8,189	66.1	0.40
Roman-Empire	22,662	32,927	300	2.9	0.05
Flickr	7,575	239,738	12,047	63.3	0.24
Ogbn-arxiv	169,343	1,157,799	767	13.67	0.65

Table 9: Hyperparameter settings of learning rate, β and initial value in UnGSL.

	Setting	GRCN	PROGNN	PROSE	IDGL	SLAPS	SUBLIME
Cora	lr	0.0005	0.0001	0.01	0.001	0.0001	0.001
	β	0.40	0.82	0.01	0.95	0.75	0.98
Citeseer	lr	0.006	0.03	0.001	0.0005	0.03	0.03
	β	0.56	0.57	0.75	0.78	0.19	0.89
Pubmed	lr	0.09	-	0.01	0.001	0.02	0.004
	β	0.35	-	0.22	0.16	0.65	0.46
BlogCatalog	lr	0.0056	0.0002	0.06	0.01	0.007	0.0001
	β	0.65	0.87	0.36	0.90	0.18	0.47
Roman-Empire	lr	0.0008	-	0.001	0.002	0.02	0.007
	β	0.001	-	0.001	0.82	0.83	0.54
Flickr	lr	0.008	0.0001	0.03	0.002	0.036	0.0001
	β	0.08	0.96	0.71	0.70	0.23	0.82
Ogbn-arxiv	lr	-	-	0.001	0.003	-	0.024
	β	-	-	0.2	0.59	-	0.48

Table 10: Robustness analysis with random noise injection on Cora dataset. We evaluate the performance of UnGSL under feature noise and label noise.

	Feature Noise Level					Label Noise Level				
Methods	0%	20%	40%	60%	80%	0%	20%	40%	60%	80%
GRCN	84.70	82.47	81.60	79.40	72.40	84.70	81.33	73.32	65.21	58.13
GRCN+UnGSL	85.84	83.70	82.80	80.62	73.70	85.84	83.43	74.52	67.94	62.17
Improve(%)	1.34%	1.49%	1.47%	1.53%	1.80%	1.34%	2.58%	1.63%	4.18%	6.95%
IDGL	84.50	82.10	79.25	75.23	65.16	84.50	77.60	70.80	54.81	43.65
IDGL+UnGSL	84.90	83.29	80.52	77.43	66.50	84.90	79.35	72.03	56.13	44.51
Improve(%)	0.47%	1.45%	1.60%	2.92%	2.04%	0.47%	2.26%	1.74%	2.41%	1.97%

Table 11: Generalizability of UnGSL with different backbones on Citeseer dataset. The value in bold signifies the top-performing result.

Methods	SGC	APPNP	GAT	JKNet
GRCN	72.00±0.00	72.73±0.96	70.77±1.06	71.83±0.72
GRCN+UnGSL	**72.37±0.06**	**73.50±0.30**	**70.88±1.70**	**72.32±1.75**
IDGL	71.90±0.00	72.20±0.75	63.83±0.97	71.00±1.15
IDGL+UnGSL	**73.40±0.00**	**73.37±0.61**	**67.00±0.69**	**72.33±0.9**

D.2 Hyperparameter Settings

Table 9 presents the hyperparameter settings of learning rate and β in UnGSL.

D.3 Additional Results on the Hyperparameter β

We assign different values to β from the interval [0,1] and evaluate the corresponding performance on the IDGL model [6]. The results are presented in Figure 6.
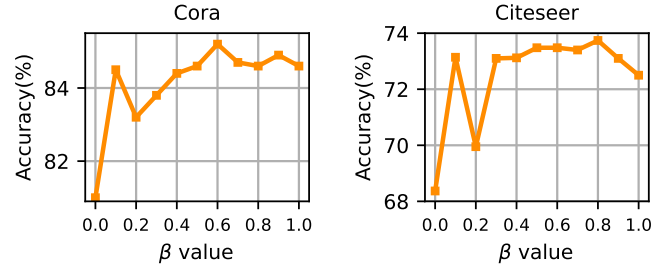
Figure 6: Comparsion of different β on Cora and Citeseer datasets.

D.4 Additional Results for Robustness Analysis

We evaluate the robustness of UnGSL against label noise and feature noise on the Cora dataset. The results are shown in Table 10. For feature noise, we randomly mask a proportion of node features by replacing their values with zeros. This approach allows us to investigate the performance of UnGSL when node features are subjected to varying degrees of damage. For label noise, we randomly alter node labels. The ratio of modifications varies from 0 to 0.8 to simulate different levels of noise. We can observe that: 1) UnGSL consistently enhances the performance of GSL models across different noise levels. 2) With increasing levels of noise, UnGSL achieves greater relative improvements. For example, with 80% wrong labels, GRCN+UnGSL achieves a 6.95% improvement, which is significantly higher than the improvement under 0% wrong labels scenario (1.34%). In summary, the results suggest that UnGSL enhances robustness to label noise and feature noise.

D.5 Additional Results on Generalizability of UnGSL

Table 11 illustrates the generalizability of UnGSL using various backbones on the Citeseer dataset.