ELU-GCN: EFFECTIVELY LABEL-UTILIZING GRAPH CONVOLUTIONAL NETWORK

Anonymous authors
Paper under double-blind review

ABSTRACT

The message-passing mechanism of graph convolutional networks (*i.e.,* GCNs) enables label information to be propagated to a broader range of neighbors, thereby increasing the utilization of labels. However, the label information is not always effectively utilized in the traditional GCN framework. To address this issue, we propose a new two-step framework called ELU-GCN. In the first stage, ELU-GCN conducts graph learning to learn a new graph structure (*i.e.,* ELU-graph), which enables GCNs to effectively utilize label information. In the second stage, we design a new graph contrastive learning on the GCN framework for representation learning by exploring the consistency and mutually exclusive information between the learned ELU graph and the original graph. Moreover, we theoretically demonstrate that the proposed method can ensure the generalization ability of GCNs. Extensive experiments validate the superiority of the proposed method.

1 INTRODUCTION

Graph Convolutional Networks (GCNs) (Kipf & Welling, 2017; Gasteiger et al., 2018; Huang et al., 2023a; Xu et al., 2018; Hamilton et al., 2017) have demonstrated remarkable capabilities, primarily due to their ability to propagate label information. This capability has driven their widespread applications in semi-supervised learning. To do this, GCN propagates the representations of unlabeled neighbors to labeled nodes by message passing mechanism, thereby enabling label information to supervise not only the labeled nodes but also their unlabeled neighbors (Ji et al., 2023; Dong et al., 2021). Consequently, the framework of optimizing label utilization in GCNs (LU-GCN) has become an increasingly prominent research topic (Wang et al., 2021; Yue et al., 2022; Yu et al., 2022).

Previous LU-GCN can be partitioned into three categories, *i.e.,* self-training methods, combination methods, and graph learning methods. self-training methods (Dong et al., 2021; Li et al., 2018; Sun et al., 2020; Ji et al., 2023) select unlabeled nodes with the highest classification probability by GCN as training data with pseudo-labels, and thus adding the number of labels to improve the GCN. Combination methods (Wang et al., 2021; Yue et al., 2022; Shi et al., 2021) regard the labels as the augment features so that labels can be used for both representation learning and classification tasks. The feature propagation mechanism allows GCNs to use labels to supervise the representation of both the node itself (*i.e.,* traditional label utilization) and its unlabeled neighbors (*i.e.,* neighboring label utilization). However, the two LU-GCN methods mentioned above primarily focus on optimizing traditional label utilization, neglecting the critical importance of neighboring label utilization in semi-supervised scenarios. Yet, due to noise in the original graph structure, GCNs often struggle to effectively utilize the neighboring labels. To address this issue, recent graph learning methods (Zheng et al., 2020; Luo et al., 2021; Liu et al., 2022) are designed to improve the relationship of every node and its neighbors by updating the graph structure, and thus may potentially improve the neighboring label utilization. For example, Bi et al. (Bi et al., 2022) adopt the own and neighbors' label similarity to rewire the graph, which can make features propagate on the same category nodes as possible.

Although existing graph learning methods have achieved promising performance, there are still some limitations that need to be addressed. First, previous methods have used heuristic approaches or downstream tasks to learn the graph structure, but they have not explored what kind of graph structures can make GCNs effectively utilize label information. As a result, the graph structures in their methods cannot guarantee that the GCN effectively utilizes the label information. Second,

existing graph learning methods fail to explore both the consistency information and the mutually exclusive information between the new graph and the original graph, where they have consistent information (*i.e.,* consistency (Xu et al., 2024)), which helps recognize the node effectively, and every graph contains unique and useful information different from another graph, *i.e.,* mutually exclusive information (Wang et al., 2017).

Based on the above observations, a possible solution to improving the effectiveness of GCNs is to define a graph structure that can maximize label utilization during the message-passing process and efficiently combine the original graph. To achieve this, two crucial challenges must be solved, *i.e.,* (i) it is difficult to evaluate whether a graph structure enables GCN to use labels effectively. (ii) it is necessary to mine the consistency and mutually exclusive information between the original graph and the new graph.

In this paper, to address the above issues, different from previous structure improvement methods, we investigate a new framework, *i.e.,* **E**ffectively **L**abel-**U**tilizing GCN (ELU-GCN for brevity), to conduct effective GCN. To achieve this, we first explore the influence of each class provided by labeled nodes on every unlabeled node. We then optimize the graph structure (*i.e.,* ELU-graph) by ensuring that the predictions of GCN align with the primary class information. This ensures that the GCN with the ELU-graph can effectively utilize the label information, thereby addressing **challenge (i)**. Moreover, we address **challenge (ii)** by designing contrasting constraints to bring the consistency information between two graph views (*i.e.,* the original graph and the ELU-graph) closer and push the mutually exclusive information further apart. Finally, we theoretically analyze that the proposed ELU-graph can not only ensure GCN to effectively utilizes labels, but also improve the generalization ability of the model. Compared with previous methods[1], our main contributions can be summarized as follows:

- To the best of our knowledge, we are the first attempt to study the limitation of GCNs that cannot effectively utilize labels in the graph framework. Moreover, we provide a quantitative framework to analyze which part of the nodes cannot effectively utilize the label information.
- We propose to adaptively construct the ELU-graph, which enables the GCN to utilize label information effectively. Furthermore, we design a contrastive loss to leverage the consistency and the mutually exclusive information between the ELU graph and the original graph.
- We theoretically prove that ELU-graph can ensure the generalization ability of GCN and we experimentally manifest the effectiveness of the proposed method across a variety of datasets, compared with numerous state-of-the-art methods.

2 METHOD

Notations. Given a graph $\mathcal{G} = (V, E, \mathbf{X}, \mathbf{Y})$, where V is the node set and E is the edge set. Original node representation is denoted by the feature matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$ where n is the number of nodes and d is the number of features for each node. The label matrix is denoted by $\mathbf{Y} \in \mathbb{R}^{n\times c}$ with a total of c classes. The first m points $\mathbf{x}_i (i \leq m)$ are labeled as $\mathbf{Y}_l$, and the remaining u points $\mathbf{x}_i$ $(m+1 \leq i \leq n)$ are unlabeled. The sparse matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ is the adjacency matrix of $\mathcal{G}$. Let $\mathbf{D} = \mathrm{diag}(d_1, d_2, \cdots, d_n)$ be the degree matrix, where $d_i = \sum_{j\in\mathcal{N}_i} a_{ij}$ is the degree of node i, the symmetric normalized adjacency matrix is represented as $\widehat{\mathbf{A}} = \widetilde{\mathbf{D}}^{-\frac{1}{2}}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-\frac{1}{2}}$ where $\widetilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, $\mathbf{I}$ is the identity matrix and $\widetilde{\mathbf{D}}$ is the degree matrix of $\widetilde{\mathbf{A}}$.

2.1 MOTIVATION

Given a classification function $f : \mathbf{X} \to \mathbb{R}^{n\times c}$, the cross entropy losses of Deep Neural Network (DNN) and GCN are formulated by:

$$\begin{aligned} \mathcal{L}_{\mathrm{DNN}} &= \mathrm{CE}(f_\theta(\mathbf{X}), \mathbf{Y}) = -\sum\nolimits_{i\in V_l, k\in C} y_{ik}(\log f_{ik}) \\ \mathcal{L}_{\mathrm{GCN}} &= \mathrm{CE}(\widehat{\mathbf{A}} f_\theta(\mathbf{X}), \mathbf{Y}) = -\sum\nolimits_{i\in V_l, k\in C} y_{ik}(\log \sum_{j\in\mathcal{N}_i} \widehat{a}_{ij} f_{jk}), \end{aligned} \tag{1}$$

[1]Related works are summarized in Appendix C.

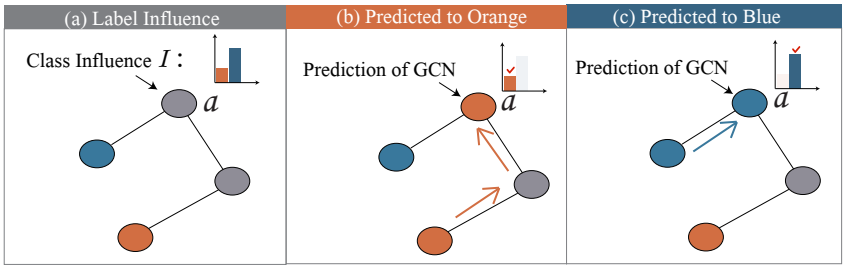


Figure 1: An illustration of effective label utilization. Sub-figure (a) wants to assign the label information to node a (gray node) by one unlabeled node (gray node) and two labeled nodes with different classes, *i.e.,* one blue node and one orange node. Moreover, the LPA algorithm is employed to obtain the probability of each labeled node to the node a, where the blue node has more influence (or higher probability) than the orange node based on the histogram in the upper right of the sub-figure (a). If the GCN predicts the node a as the orange color (as shown in sub-figure (b)), which is inconsistent with the class with most label information (*i.e.,* blue). It indicates that the label information provided by the message passing of the GCN does not help classify the node a, and may even hinder its correct classification. On the contrary, if GCN predicts the node a as the blue color, *i.e.,* sub-figure (c), it implies that the label information provided by the message passing of the GCN helps to classify the node a.

where θ is the parameters of the function f. In Eq. (1), the cross entropy loss of DNN is a one-to-one mapping between the feature space and the label space because every label y_i ($l = 1, ..., n$) is only used to supervise the representation learning of one node v_i. The mapping f efficiently captures the pattern and distribution of labeled nodes, but it overlooks unlabeled nodes so that the generalization ability of unlabeled nodes is limited. In contrast, the cross entropy loss of the GCN is a one-to-many mapping because its message-passing mechanism can propagate the information from labeled nodes to their neighbors including labeled nodes and unlabeled nodes. As a result, every label y_i is used to supervise the representation learning of both labeled nodes and unlabeled nodes, as shown in the second row of Eq. (1). Hence, unlabeled nodes in the GCN are able to use the label information of labeled nodes to improve the learning of their representations. Obviously, it is very important to guarantee that unlabeled nodes effectively utilize label information under the GCN framework. However, to the best of our knowledge, no research has focused on this issue. To address this issue, we first quantify the influence of every class on unlabeled nodes, and then make the class with the highest influence (*i.e.,* probability) on unlabeled nodes consistent with the prediction of the GCN to effectively utilize the label information.

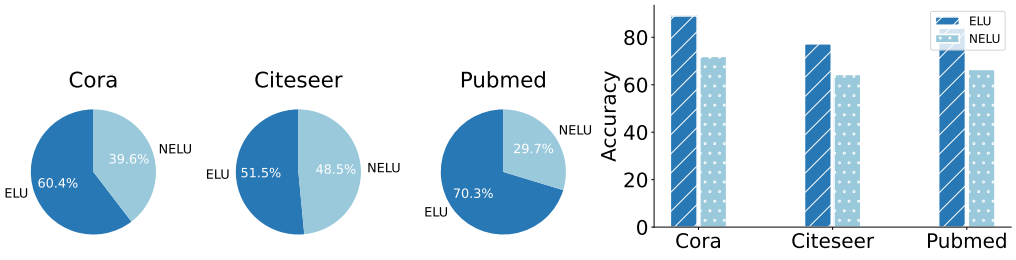
The recent study in (Xu et al., 2018) reveals that nodes follow the way of random walks to affect other nodes on the graph. Therefore, in this paper, we extend it to obtain the influence of every class of labeled nodes to the unlabeled node by Theorem 2.1, whose proof is provided in Appendix B.1.

Theorem 2.1. *Given an unlabeled node v_i ($i = 1, ..., n$), for an arbitrary category C_l ($l = 1, ..., c$), the influence of labeled nodes belong to C_l on the i-th node v_i is proportional to the probability that node v_i is classified as C_l by the Label Propagation Algorithm (LPA) in (Zhu, 2005), in the GCN framework.*

Based on Theorem 2.1, LPA can be utilized to calculate the probability of every class for unlabeled nodes in the GCN framework. The class with the highest probability is considered the most important for the unlabeled node, as it contributes the most label information. If the class that carries the most label information to a node in the GCN framework is the same as the GCN prediction, it indicates that the label information propagated by the message passing mechanism of the GCN positively influences the classification result for that node. In this way, this node is regarded as effectively utilizing the label information. We provide a case study to illustrate this in Figure 1 and give a formal definition as follows.

Definition 2.2. ***(Effective label-utilization)*** *The GCN effectively utilizes label information if the prediction of GCN is consistent with the output of LPA,* i.e.,

$$V_{\text{ELU}} = \{V | \text{LPA}(\mathcal{G}) = \text{GCN}(\mathcal{G})\}, \tag{2}$$



(a) Proportion (b) Accuracy

Figure 2: Visualization of both ELU nodes and NELU nodes in three real datasets, *i.e.,* Cora, Citerseer, and Pubmed. (a) every dataset contains NELU nodes and (b) the classification comparison between ELU nodes and NELU nodes, where ELU nodes have higher classification ability than NELU nodes.

where V_{ELU} and V_{NELU} (*i.e.,* $V_{\mathrm{NELU}} = \{V|\mathrm{LPA}(\mathcal{G}) \neq \mathrm{GCN}(\mathcal{G})\}$), respectively, represent the node set which effectively utilizes the label information and the node set which does not effectively utilizes the label information in the GCN framework.

In real applications, not all unlabeled nodes in GCN frameworks may effectively utilize the label information due to all kinds of reasons, including noise and the distribution of labeled nodes in the graph. Figure 2 shows that not all nodes effectively use label information in the GCN framework (*i.e.,* Figure 2 (a)) and the classification accuracy of V_{NELU} is lower than that of V_{ELU} in the same datasets (*i.e.,* Figure 2 (b)). Obviously, NELU nodes influence the effectiveness of the GCN. To address this issue, first, it is crucial to make unlabeled nodes effectively utilize label information. Since label information is propagated through the graph structure. As a result, the graph structure will be updated. Second, the original graph structures often contain noise to influence the message-passing mechanism. Hence, graph learning is obviously a feasible solution.

2.2 ELU Graph

Previous graph learning methods generally use either heuristic methods or downstream tasks to conduct graph learning, *i.e.,* updating the graph structure. For example, Pro-GNN (Jin et al., 2020) updates the graph structure through a heuristic approach to constrain the sparsity and smoothness of the graph. PTD-Net (Luo et al., 2021) updates the graph structure by the downstream task, such as the node classification task. However, heuristic methods rely on predefined rules, making it difficult for unlabeled nodes to fully access label-related global information. Downstream task methods focus too much on the performance of labeled nodes, neglecting the role of unlabeled nodes in the graph structure. Therefore, these efforts cannot ensure unlabeled nodes effectively utilize label information. To solve this issue, based on Definition 2.2, we investigate new graph learning methods that ensure unlabeled nodes effectively utilize label information.

Specifically, denoting the adjacency matrix $\mathbf{S}$ as the ELU graph can ensure the GCN effectively uses the label information, we use Theorem 2.1 to measure the influence of each class on every unlabeled node by the LPA:

$$\mathbf{Q} = \mathbf{SY}, \tag{3}$$

where the i-th row of $\mathbf{Q} \in \mathbb{R}^{n\times c}$ (*i.e.,* $\mathbf{Q}_{i,:}$) represents the influence of each class on node i. It is noteworthy that $\mathbf{S}$ in Eq. (3) can be the k-order of the graph structure. After that, the prediction of GCN with ELU graph can be written as follows (Yang et al., 2023):

$$\hat{\mathbf{Y}} = \mathbf{SH}, \quad s.t.\ \mathbf{H} = \mathrm{MLP}(\mathbf{X}), \tag{4}$$

where $\mathrm{MLP}(\cdot)$ denotes a Multi-Layer Perceptron. Note that the MLP is pre-trained. Therefore, based on Definition 2.2, the ELU graph (*i.e.,* $\mathbf{S}$) can be obtained by minimizing the following objective function:

$$\min \left\| \mathbf{Q} - \hat{\mathbf{Y}} \right\|_F^2 = \min_{\mathbf{S}} \|\mathbf{SY} - \mathbf{SH}\|_F^2 . \tag{5}$$

In Eq. (5), the prediction of GCN and the influence of each class are encouraged to be consistent for every node. This item can make all nodes satisfy $\mathrm{LPA}(\mathcal{G}) = \mathrm{GCN}(\mathcal{G})$ in Eq. (2), *i.e.,* this objective

function can ensure all nodes can effectively utilize label information by GCN. Therefore, we can obtain the $\mathbf{S}$ through the optimization algorithm by minimizing the Eq. (5). However, there are some problems with the above objective function. First, it is impracticable to solve the above problem directly, as it has a trivial solution: $s_{i,j} = 0, \forall i, \forall j$. Second, LPA generates the prediction for every labeled node to possibly revise the original labels, *i.e.,* the ground truth, adding noisy labels for representation learning. To overcome the above issues, We propose to iteratively update in two steps, *i.e.,* update labels by LPA and update the graph structure $\mathbf{S}$.

In the first step, we calculate the result of LPA $\mathbf{Q}^{(i)}$, *i.e.,* $\mathbf{Q}^{(i)} = \mathbf{S}^{(i-1)}\mathbf{Q}^{(i-1)}$, $(i = 1, \ldots, k)$, where $\mathbf{Q}^{(0)} = \mathbf{Y}$ and we initialize $\mathbf{S}^{(0)} = \mathbf{I_N}$. As a result, Eq.(5) is changed as follows:

$$\min_{\mathbf{S}} \left\| \mathbf{Q}^{(i)} - \mathbf{S}\mathbf{H} \right\|_F^2 + \beta \sum_{i,j=1} s_{i,j}^2, s.t.\ \ \mathbf{Q}_l^{(i)} = \mathbf{Y}_l, \tag{6}$$

where β is a non-negative parameter to trade off two terms, the second term can make the subsequent matrix inversion more stable. Eq. (6) holds the closed-form solution to address the first issue. The constraint term "s.t. $\mathbf{Q}_l^{(i)} = \mathbf{Y}_l$" term solves the second issue.

In the second step, we can obtain its closed-form solution, which is listed as follows and its details are in Appendix B.2:

$$\mathbf{S}^{(i)} = \mathbf{H}(\mathbf{Q}^{(i)})^T \left(\mathbf{H}\mathbf{H}^T + \beta \mathbf{I}_N\right)^{-1}, \tag{7}$$

where $\mathbf{I}_N \in \mathbb{R}^{n \times n}$ is the identity matrix.

Finally, we iteratively optimize Eq. (7) and $\mathbf{Q}^{(i)} = \mathbf{S}^{(i-1)}\mathbf{Q}^{(i-1)}$ to obtain the ELU graph $\mathbf{S}^*$.

However, the calculation of $\mathbf{S}^{(i)}$ in Eq. (7) is with the time complexity of $\mathcal{O}(n^3)$. In this paper, we use the Woodbury identity (Woodbury, 1950) to avoid calculating $\mathbf{S}^{(i)}$ during the iteration process by $\mathbf{Q}^{(i)} = \mathbf{S}^{(i-1)}\mathbf{Q}^{(i-1)}$, *i.e.,*

$$\mathbf{Q}^{(i)} = \mathbf{H}(\mathbf{Q}^{(i-1)})^T \left(\frac{1}{\beta}\mathbf{I}_N - \frac{1}{\beta^2}\mathbf{H}\left(\mathbf{I}_c + \frac{1}{\beta}\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\right)\mathbf{Q}^{(i-1)}, s.t.\ \ \mathbf{Q}_l^{(i)} = \mathbf{Y}_l, \tag{8}$$

where $\mathbf{I}_c \in \mathbb{R}^{c \times c}$ is the identity matrix and the specific derivation process is listed in the Appendix B.3. Based on the literature (Woodbury, 1950), we can obtain the time complexity of Eq. (8) is $\mathcal{O}(nc^2 + c^3)$, where $c^3 \ll n$. The details are provided in Appendix A.1.

Based on Eq. (8), we obtain $\mathbf{Q}^{(i)}$ $(i = 1, \ldots, k)$ from $\mathbf{Q}^{(i-1)}$. After obtaining $\mathbf{Q}^{(k)}$, we obtain the ELU graph $\mathbf{S}^*$ by calculating Eq. (7) only one time. To achieve efficiency, we employ the Woodbury identity to reduce the time complexity of calculating from cubic to quadratic, *i.e.,*

$$\mathbf{S}^* = \mathbf{H}(\frac{1}{\beta}(\mathbf{Q}^{(k)})^T - \frac{1}{\beta^2}(\mathbf{Q}^{(k)})^T\mathbf{H}\left(\mathbf{I}_c + \frac{1}{\beta}\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T). \tag{9}$$

The details of Eq. (9) are listed in Appendix A.2. The pseudocode of calculating Eq. (8) and $\mathbf{S}^*$ is presented in Algorithm 1. In the implementation, we make $\mathbf{S}^*$ sparse by assigning its element less than a threshold as zero, for achieving efficiency. We also use the pseudo labels of ELU nodes to expand the initial $\mathbf{Y}$, for avoiding the issue of limited labels in semi-supervised learning.

2.3 Graph Contrastive Learning

Given the ELU graph $\mathbf{S}^*$ and the original graph $\widehat{\mathbf{A}}$, previous graph learning methods often conduct a weighted fusion. For instance, SimP-GCN (Jin et al., 2021) employs a hyperparameter as a weight to fuse the node representation from the original graph with those from the feature similarity graph. However, only performing the weighted sum method may result in incorporating undesirable information from the original graph into the ELU graph. For example, the representation of a NELU node from the original graph might interfere with the learned representation of the corresponding node in the ELU graph. To solve this issue, in this paper, we propose a new contrastive learning paradigm to capture the consistency and mutually exclusive information between these two graphs.

In the ELU graph $\mathbf{S}^*$, all nodes are theoretically ELU nodes. However, the original graph $\widehat{\mathbf{A}}$ includes ELU nodes and NELU nodes. Obviously, in representation learning, the representations of ELU

nodes in both $\mathbf{S}^*$ and $\widehat{\mathbf{A}}$ should be consistent for keeping common information related to the class, the representations of NELU nodes in $\widehat{\mathbf{A}}$ should be different from their representation in $\mathbf{S}^*$. To do this, we first propose to learn a projection head p_θ to map both the ELU graph representations and the original graph representations into the same latent space, *i.e.,* $\overline{\mathbf{P}} = p_\theta(\overline{\mathbf{H}})$ and $\widetilde{\mathbf{P}} = p_\theta(\widetilde{\mathbf{H}})$, where $\overline{\mathbf{H}}$ is the representation of the output layer of the GCN dominated by the original graph, and $\widetilde{\mathbf{H}}$ is the representation of the output layer of the GCN dominated by the ELU graph. We then design a contrastive loss as follows:

$$\begin{aligned}\mathcal{L}_{\text{con}} = &- \log \frac{\frac{1}{|V_{\text{ELU}}|}\sum_{i=0}^{V_{\text{ELU}}} \exp(d(\overline{\mathbf{P}}_i, \widetilde{\mathbf{P}}_i)/\tau)}{\frac{1}{|V_{\text{ELU}}|}\sum_{i=0}^{V_{\text{ELU}}} \exp(d(\overline{\mathbf{P}}_i, \widetilde{\mathbf{P}}_i)/\tau) + \frac{1}{|V_{\text{NELU}}|}\sum_{j=0}^{V_{\text{NELU}}} \exp(d(\overline{\mathbf{P}}_j, \widetilde{\mathbf{P}}_j)/\tau)} \\ &+ \gamma \log\left(\sum_{i,j=1}^{d} e^{\overline{\mathbf{P}}^T\overline{\mathbf{P}} + \widetilde{\mathbf{P}}^T\widetilde{\mathbf{P}}}\right)\end{aligned} \tag{10}$$

where $d(\cdot)$ is distance function, τ denotes the temperature parameter and γ is a hyper-parameter.

In Eq. (10), the first term encourages minimizing the distance between every ELU node in the ELU graph and its corresponding node in the original graph, while maximizing the distance between every NELU node in the original graph and its corresponding node in the ELU graph. The second term ensures that different dimensions of the representation matrices (*i.e.,* $\overline{\mathbf{P}}$ and $\widetilde{\mathbf{P}}$) are uniformly distributed over the latent space, thereby avoiding the issue of feature collapse. As a result, Eq. (10) is available to extract the consistency and mutually exclusive information between the representations dominated by the ELU graph and the original graph.

Finally, the final objective function of our proposed method is obtained by integrating the contrastive loss with the supervised loss (*i.e.,* cross entropy) as follows:

$$\mathcal{L} = CE((1-\eta)Softmax(\overline{\mathbf{H}}) + (\eta)Softmax(\widetilde{\mathbf{H}}), \mathbf{Y}) + \lambda\mathcal{L}_{con} \tag{11}$$

where $\eta, \lambda \in [0, 1]$ are hyper-parameters to fuse the predicted results of two views and two objective functions, respectively.

2.4 Theoretical Analysis

The ELU graph has been shown to effectively utilize the label information in Section 2.1. In this section, we theoretically analyze that the generalization ability of the GCN is related to the graph structure and the training labels by Theorem 2.3 (The proof can be found in Appendix B.4):

Theorem 2.3. *Given a graph $\mathcal{G}$ with its adjacency matrix $\mathbf{A}$, the label matrix in the training set $\mathbf{Y}$ and the label matrix of the ground truth $\mathbf{Y}_{\text{true}}$, for any unlabeled nodes, if a graph structure makes the labels in training set be consistent to the ground truth,* i.e., $\mathbf{Y}_{\text{true}} = \mathbf{AY}$*, then the upper bound of the generalization ability of the GCN is optimal.*

Based on Theorem 2.3, the graph structure $\mathbf{A}$ maximizes the generalization ability of the GCN if the following equation holds, *i.e.,* $\min_{\mathbf{A}} \|\mathbf{AY} - \mathbf{Y}_{\text{true}}\|_F^2$. Therefore, the graph structure can be used to measure if it is suitable for GCN. However, the true labels $\mathbf{Y}_{\text{true}}$ are fixed and unknown. Moreover, the original graph is also fixed so that it is difficult to achieve $\min_{\mathbf{A}} \|\mathbf{AY} - \mathbf{Y}_{\text{true}}\|_F^2$. Hence, the original graph should be updated. We then present the following theorem. The proof is listed in Appendix B.5.

Theorem 2.4. *The optimization Eq. (5) is equivalent to an approximate optimization of* $\min_{\mathbf{A}} \|\mathbf{AY} - \mathbf{Y}_{\text{true}}\|_F^2$.

Theorem 2.4 indicates that the ELU graph can ensure the generalization ability of the GCN.

3 Experiments

In this section, we conduct experiments on eleven public datasets to evaluate the proposed method (including citation networks, Amazon networks, social networks, and web page networks), compared to structure improvement methods[2]. Detailed settings are shown in Appendix F. Additional experimental results are shown in Appendix G.

[2]The code is released at https://anonymous.4open.science/r/ELU-GCN-8CAE

Table 1: Performance on node classification task. The highest results are highlighted in bold. "OOM" denotes out of memory.

Method	Cora	Citeseer	pubmed	Computers	Photo	Chameleon	squirrel
GCN	81.61±0.42	70.35±0.45	79.01±0.62	81.62±2.43	90.44±1.23	60.82±2.24	43.43±2.18
GAT	83.03±0.71	71.54±1.12	79.17±0.38	78.01±19.1	85.71±20.3	40.72±1.55	30.26±2.50
APPNP	83.33±0.62	71.80±0.84	80.10±0.21	82.12±3.13	88.63±3.73	56.36±1.53	46.53±2.18
GPRGNN	80.55±1.05	68.57±1.22	77.02±2.59	81.71±2.84	91.23±2.59	46.85±1.71	31.61±1.24
PCNet	82.81±0.50	69.92±0.70	80.01±0.88	81.82±2.31	89.63±2.41	59.74±1.43	48.53±1.12
GCN-LPA	83.13±0.51	72.60±0.80	78.64±1.32	83.54±1.41	90.13±1.53	50.26±1.38	42.78±2.36
N.S.-GCN	82.12±0.14	71.55±0.14	79.14±0.12	81.16±1.53	89.86±1.86	55.37±1.64	46.86±2.02
PTDNet-GCN	82.81±0.23	72.73±0.18	78.81±0.24	82.21±2.13	90.23±2.84	53.26±1.44	41.96±2.16
CoGSL	81.76±0.24	72.79±0.42	OOM	OOM	89.63±2.24	52.23±2.03	39.96±3.31
NodeFormer	80.28±0.82	71.31±0.98	78.21±1.43	80.35±2.75	89.37±2.03	34.71±4.12	38.54±1.51
GSR	83.08±0.48	72.10±0.25	78.09±0.53	81.63±1.35	90.02±1.32	62.28±1.63	50.53±1.93
BAGCN	83.70±0.21	72.96±0.75	78.54±0.72	79.63±2.52	**91.25±0.96**	52.63±1.78	42.36±1.53
ELU-GCN	**84.04±0.39**	**73.17±0.62**	**80.51±0.21**	**83.72±2.31**	90.80±1.33	**70.59±1.76**	**60.91±1.81**

3.1 Experimental Setup

3.1.1 Datasets

The used datasets include three benchmark citation datasets (Sen et al., 2008) (*i.e.,* Cora, Citeseer, and Pubmed), two co-purchase networks (Shchur et al., 2018) (*i.e.,* Computers and Photo), two web page networks (Pei et al.) (*i.e.,* Chameleon and Squirrel), which are heterophilic graph data), and four social network datasets (Traud et al., 2012) (*i.e.,* Caltech, UF, Hamilton, and Tulane).

3.1.2 Comparison Methods

The comparison methods include three traditional GNN methods, two advanced GNN methods, and seven structure improvement-based GCN methods. Traditional GNN methods include GCN (Kipf & Welling, 2017), GAT (Velickovic et al., 2018), and APPNP (Gasteiger et al., 2018). The advanced GNN methods include GPRGNN (Chien et al., 2021) and PCNet (Li et al., 2024). The structure improvement-based GCN methods include GCN-LPA (Wang & Leskovec, 2021), NeuralSparse-GCN (Zheng et al., 2020), PTDNet-GCN (Luo et al., 2021), CoGSL (Liu et al., 2022), NodeFormer (Wu et al., 2022), GSR (Zhao et al., 2023) and BAGCN (Zhang et al., 2024).

3.1.3 Evaluation Protocol

To evaluate the effectiveness of the proposed method, we follow the commonly used setting. Specifically, for the citation network (*i.e.,* Cora, Citeseer, and Pubmed), we use the public split recommended by (Kipf & Welling, 2017) with fixed 20 nodes per class for training, 500 nodes for validation, and 1000 nodes for testing. For Social networks (*i.e.,* Caltech, UF, Hamilton, and Tulane), we randomly generate different data splits with an average train/val/test split ratio of 60%/20%/20%. For the Webpage network (*i.e.,* Chameleon, Squirrel) and co-purchase networks (*i.e.,* Computers, Photo), we all use the public splits recommended in the original papers.

3.2 Rsults Analysis

3.2.1 Effectiveness Analysis

We first evaluate the effectiveness of the proposed method by reporting the results of node classification in Table 1 and Appendix G, respectively. Obviously, the proposed method obtains better performance on seven datasets than comparison methods.

First, compared with traditional GNN methods and advanced GNN methods. the proposed ELU-GCN outperforms them by large margins on most datasets. For example, the proposed ELU-GCN on average improves by 4.05 %, compared to GCN, and improves by 3.26 % compared to the best advanced GCN method (*i.e.,* PCNet), on all datasets. This demonstrates the superiority of graph

Table 2: Ablation study.

Method	Cora	Citeseer	pubmed	Computers	Photo	Chameleon	squirrel
GCN	81.61±0.42	70.35±0.45	79.01±0.62	81.62±2.43	90.44±1.23	60.82±2.24	43.43±2.18
+ELU graph	83.49±0.55	72.02±0.36	80.25±0.79	82.56±1.23	90.52±1.33	65.12±1.43	54.12±1.32
+$\mathcal{L}_{con}$	**84.04**±0.39	**73.17**±0.62	**80.51**±0.21	**83.72**±2.31	**90.80**±1.33	**70.59**±1.76	**60.91**±1.81

structure learning methods, as the label information cannot be effectively utilized for many nodes in the original graph. Second, compared to the improvement methods, the proposed ELU-GCN achieves the best results, followed by GSR, GCN-LPA, CoGSL, PTDNet-GCN, NeuralSparse-GCN, and NodeFormer. For example, our method on average improves by 2.21% compared to the best comparison method GSR on all seven datasets. This can be attributed to the fact that the proposed ELU-GCN, which can obtain a graph structure (*i.e.,* the ELU graph) that is more suitable for the GCN model to effectively utilize the label information and efficiently mine the consistency and mutually exclusive information between the original graph and the newly obtained graph. In addition, the Webpage networks (i.e., Chameleon and Squirrel) are heterophilic graphs. As mentioned in the theoretical analysis section, the original graph is difficult to guarantee the generalization ability of GCN, especially for heterophilic graphs. Experimental results show that the proposed ELU-GCN outperforms the GCN using the original heterophilic graph by an average of 9.5%, confirming the results of our theoretical analysis. Consequently, the effectiveness of the proposed method is verified in node classification tasks.

We further evaluate the effectiveness of the proposed method on social network datasets and report the results of node classification in Appendix G.1. We can observe that the proposed method also achieves competitive results on the social network datasets compared to other baselines. For example, the proposed method outperforms the best baseline (*i.e.,* GSR), on almost all datasets.

3.2.2 Ablation Study

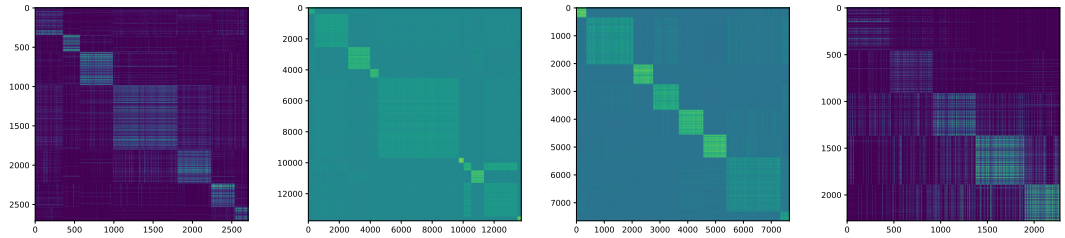
The proposed ELU-GCN framework investigates the ELU graph to enable the GCN to utilize label information effectively. Additionally, a contrastive loss function (*i.e.,* , $\mathcal{L}_{con}$) is introduced to efficiently minimize consistency and mutually exclusive information between the original graph and the ELU graph. To verify the effectiveness of each component of the proposed method and the results are reported in Table 2.

According to Table 2, we can draw the following conclusions. First, our proposed method achieves the best performance when each component is present, indicating that each is essential. This demonstrates the importance of both learning the ELU graph and extracting information from the original graph, as they not only enable GCN to effectively utilize labels but also retain important information in the original graph. Second, the ELU graph component provided the biggest improvement. For example, the ELU graph improves performance by an average of 2.9% compared to not considering it, and the $\mathcal{L}_{con}$ term improves performance by an average of 1.3% compared to not considering it. This illustrates the importance of enabling nodes to effectively utilize the label information.

3.2.3 Visualization

To verify the effectiveness of the learned ELU graph, we visualize the adjacency matrix of the ELU graph in the heatmap on the Cora, Computers, Photo, and Chameleon datasets and report the results in Figure 3.

Specifically, the rows and columns of heatmaps are reordered by node labels. In the heatmaps, the lighter a pixel, the larger the value of the ELU graph matrix weight. From Figure 3, we observe that the heatmaps exhibit a clear block diagonal structure, with each block corresponding to a category. This indicates that the obtained ELU graph tends to increase the weight connections between nodes of the same category and avoid noisy connections from different classes. As a result, the training labels will be transferred to nodes of the same category under the GCN framework with a high probability, thereby reducing intra-class variance and increasing inter-class distance. Especially on the Chameleon dataset, where the original graph tends to connect nodes with different labels with a high probability (*i.e.,* heterophily). Fortunately, our method can still obtain a graph structure where

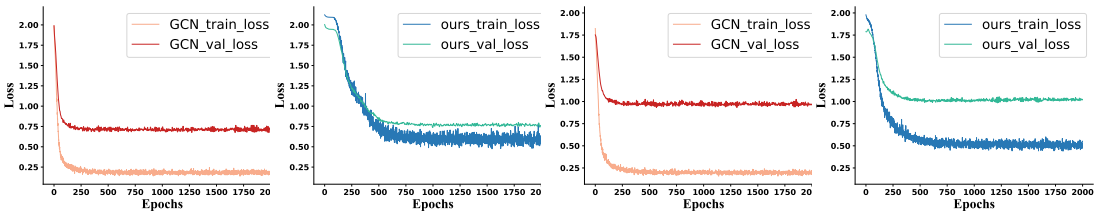


(a) Cora (b) Computers (c) Photo (d) Chameleon

Figure 3: Visualization of the adjacency matrix of the ELU graph on Cora, Computers, Photo, and Chameleon datasets.

nodes are connected with the same category, as shown by the experimental results, demonstrating the universality of our method.

3.2.4 Generalization Gap Analysis



(a) G.G. of GCN on Cora (b) G.G. of ours on Cora (c) G.G. of GCN on Citeseer (d) G.G. of ours on Citeseer

Figure 4: Visualization of the generalization gap (*i.e.,* G.G) of our model (*i.e.,* ELU-GCN) and GCN on Cora and Citeseer datasets.

As Theorem 2.4 mentioned the proposed ELU-GCN can ensure the generalization ability but GCN using the original graph cannot (*i.e.,* traditional GCN). Therefore, to verify the generalization ability, we introduce the generalization gap (Keskar et al., 2016), which is the gap between the training loss and validating loss. A small gap between the two losses indicates a model with good generalization. We visualize the generalization gap of ELU-GCN and GCN on Cora and Citeseer datasets, the results are shown in Figure 4.

Specifically, the proposed ELU-GCN shows a small generalization gap, compared to GCN. For example, the proposed method's generalization gap on the Cora and Citeseer datasets is approximately 63.6% and 26.7% lower than that of GCN, respectively. This is consistent with the observation in Theorem 2.4 and further verifies the effectiveness of the proposed ELU-GCN.

4 Conclusion

In this paper, we study the label utilization of GCN and reveal that a considerable number of unlabeled nodes cannot effectively utilize label information in the GCN framework. Furthermore, we propose a standard for determining which unlabeled nodes can effectively utilize label information in the GCN framework. To make more nodes to effectively utilize label information. We propose an effective label-utilizing graph convolutional network framework. To do this, we optimize the graph structure by constraining every node effectively using label information. Moreover, we design a novel contrastive loss to minimize consistency or mutually exclusive information between the original graph and the ELU graph. Our theoretical analysis demonstrates that ELU-GCN provides superior generalization capabilities compared to conventional GCNs. Extensive experimental results further validate that our method consistently outperforms state-of-the-art methods.

REFERENCES

Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. In *International Conference on Learning Representations(ICLR)*, 2021.

Wendong Bi, Lun Du, Qiang Fu, Yanlin Wang, Shi Han, and Dongmei Zhang. Make heterophily graphs better fit gnn: A graph rewiring approach. *arXiv preprint arXiv:2209.08264*, 2022.

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

Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized pagerank graph neural network. In *International Conference on Learning Representations, (ICLR)*. OpenReview.net, 2021.

Samuel I Daitch, Jonathan A Kelner, and Daniel A Spielman. Fitting a graph to vector data. In *Proceedings of the 26th annual international conference on machine learning (ICML)*, pp. 201–208, 2009.

David L Davies and Donald W Bouldin. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence(TPAMI)*, (2):224–227, 1979.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems(NeurIPS)*, volume 29, pp. 3837–3845, 2016.

Hande Dong, Jiawei Chen, Fuli Feng, Xiangnan He, Shuxian Bi, Zhaolin Ding, and Peng Cui. On the equivalence of decoupled graph convolution network and label propagation. In *Proceedings of the Web Conference(WWW)*, pp. 3651–3662, 2021.

Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *International Conference on Learning Representations, (ICLR)*, 2018.

Jhony H Giraldo, Konstantinos Skianis, Thierry Bouwmans, and Fragkiskos D Malliaros. On the trade-off between over-smoothing and over-squashing in deep graph neural networks. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management(CIKM)*, pp. 566–576, 2023.

Shengbo Gong, Jiajun Zhou, Chenxuan Xie, and Qi Xuan. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM)*, pp. 3908–3912, 2023.

William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in neural information processing systems(Neurips)*, pp. 1025–1035, 2017.

Jincheng Huang, Lun Du, Xu Chen, Qiang Fu, Shi Han, and Dongmei Zhang. Robust mid-pass filtering graph convolutional networks. In *Proceedings of the Web Conference(WWW)*, pp. 328–338, 2023a.

Jincheng Huang, Ping Li, Rui Huang, Na Chen, and Acong Zhang. Revisiting the role of heterophily in graph representation learning: An edge classification perspective. *ACM Transactions on Knowledge Discovery from Data(TKDD)*, 18:13:1–13:17, 2023b.

Jincheng Huang, Jialie Shen, Xiaoshuang Shi, and Xiaofeng Zhu. On which nodes does GCN fail? enhancing GCN from the node perspective. In *Forty-first International Conference on Machine Learning (ICML)*, 2024.

Feng Ji, See Hian Lee, Hanyang Meng, Kai Zhao, Jielong Yang, and Wee Peng Tay. Leveraging label non-uniformity for node classification in graph neural networks. In *International Conference on Machine Learning(ICML)*, pp. 14869–14885, 2023.

Bo Jiang, Ziyan Zhang, Doudou Lin, Jin Tang, and Bin Luo. Semi-supervised learning with graph learning-convolutional networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11313–11320, 2019.

Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, and Jiliang Tang. Graph structure learning for robust graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining (SIGKDD)*, pp. 66–74, 2020.

Wei Jin, Tyler Derr, Yiqi Wang, Yao Ma, Zitao Liu, and Jiliang Tang. Node similarity preserving graph convolutional networks. In *Proceedings of the 14th ACM international conference on web search and data mining (WSDM)*, pp. 148–156, 2021.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations(ICLR)*, 2016.

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

Bingheng Li, Erlin Pan, and Zhao Kang. Pc-conv: Unifying homophily and heterophily with two-fold filtering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 13437–13445, 2024.

Mingjie Li, Xiaojun Guo, Yifei Wang, Yisen Wang, and Zhouchen Lin. G^2 cn: Graph gaussian convolution networks with concentrated graph filters. In *International Conference on Machine Learning(ICML)*, pp. 12782–12796. PMLR, 2022.

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

Meng Liu, Hongyang Gao, and Shuiwang Ji. Towards deeper graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining(SIGKDD)*, pp. 338–348, 2020.

Nian Liu, Xiao Wang, Lingfei Wu, Yu Chen, Xiaojie Guo, and Chuan Shi. Compact graph structure learning via mutual information compression. In *Proceedings of the ACM Web Conference (WWW))*, pp. 1601–1610, 2022.

Songtao Liu, Jinghui Chen, Tianfan Fu, Lu Lin, Marinka Zitnik, and Dinghao Wu. Graph adversarial diffusion convolution. In *Forty-first International Conference on Machine Learning (ICML)*, 2024.

Dongsheng Luo, Wei Cheng, Wenchao Yu, Bo Zong, Jingchao Ni, Haifeng Chen, and Xiang Zhang. Learning to drop: Robust graph neural network via topological denoising. In *Proceedings of the 14th ACM international conference on web search and data mining(WSDM)*, pp. 779–787, 2021.

Parth Natekar and Manik Sharma. Representation based complexity measures for predicting generalization in deep learning. *arXiv preprint arXiv:2012.02775*, 2020.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. *Advances in neural information processing systems(NeurIPS)*, 30: 5947–5956, 2017.

Khang Nguyen, Nong Minh Hieu, Vinh Duc Nguyen, Nhat Ho, Stanley Osher, and Tan Minh Nguyen. Revisiting over-smoothing and over-squashing using ollivier-ricci curvature. In *International Conference on Machine Learning(ICML)*, volume 202, pp. 25956–25979, 2023.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. In *8th International Conference on Learning Representations (ICLR), year = 2020,.*

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

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*, 2018.

Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjing Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1548–1554. ijcai.org, 2021.

Ke Sun, Zhouchen Lin, and Zhanxing Zhu. Multi-stage self-supervised learning for graph convolutional networks on graphs with few labeled nodes. In *Proceedings of the AAAI conference on artificial intelligence (AAAI)*, volume 34, pp. 5892–5899, 2020.

Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. In *International Conference on Learning Representations(ICLR)*, 2022.

Amanda L Traud, Peter J Mucha, and Mason A Porter. Social structure of facebook networks. *Physica A: Statistical Mechanics and its Applications*, 391(16):4165–4180, 2012.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations, (ICLR)*. OpenReview.net, 2018.

Fei Wang and Changshui Zhang. Label propagation through linear neighborhoods. In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, pp. 985–992, 2006.

Hongwei Wang and Jure Leskovec. Combining graph convolutional neural networks and label propagation. *ACM Transactions on Information Systems (TOIS)*, 40(4):1–27, 2021.

Xiaobo Wang, Xiaojie Guo, Zhen Lei, Changqing Zhang, and Stan Z Li. Exclusivity-consistency regularized multi-view subspace clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 923–931, 2017.

Yangkun Wang, Jiarui Jin, Weinan Zhang, Yong Yu, Zheng Zhang, and David Wipf. Bag of tricks for node classification with graph neural networks. *arXiv preprint arXiv:2103.13355*, 2021.

Max A Woodbury. *Inverting modified matrices*. Department of Statistics, Princeton University, 1950.

Qitian Wu, Wentao Zhao, Zenan Li, David P Wipf, and Junchi Yan. Nodeformer: A scalable graph structure learning transformer for node classification. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:27387–27401, 2022.

Zhihao Wu, Zhao Zhang, and Jicong Fan. Graph convolutional kernel machine versus graph convolutional networks. *Advances in neural information processing systems (NeurIPS)*, 36, 2024.

Jie Xu, Yazhou Ren, Xiaolong Wang, Lei Feng, Zheng Zhang, Gang Niu, and Xiaofeng Zhu. Investigating and mitigating the side effects of noisy views for self-supervised clustering algorithms in practical multi-view scenarios. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22957–22966, 2024.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning(ICML)*, pp. 5453–5462. PMLR, 2018.

Cheng Yang, Chengdong Yang, Chuan Shi, Yawen Li, Zhiqiang Zhang, and Jun Zhou. Calibrating graph neural networks from a data-centric perspective. In *Proceedings of the ACM on Web Conference (WWW)*, pp. 745–755, 2024.

Chenxiao Yang, Qitian Wu, Jiahua Wang, and Junchi Yan. Graph neural networks are inherently good generalizers: Insights by bridging gnns and mlps. In *International Conference on Learning Representations(ICLR)*, 2023.

Le Yu, Leilei Sun, Bowen Du, Tongyu Zhu, and Weifeng Lv. Label-enhanced graph neural network for semi-supervised node classification. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

Han Yue, Chunhui Zhang, Chuxu Zhang, and Hongfu Liu. Label-invariant augmentation for semi-supervised graph classification. In *Advances in Neural Information Processing Systems (Neurips)*, volume 35, pp. 29350–29361, 2022.

Acong Zhang, Jincheng Huang, Ping Li, and Kai Zhang. Building shortcuts between distant nodes with biaffine mapping for graph convolutional networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 18(6):1–21, 2024.

Jianan Zhao, Qianlong Wen, Mingxuan Ju, Chuxu Zhang, and Yanfang Ye. Self-supervised graph structure refinement for graph neural networks. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM)*, pp. 159–167, 2023.

Cheng Zheng, Bo Zong, Wei Cheng, Dongjin Song, Jingchao Ni, Wenchao Yu, Haifeng Chen, and Wei Wang. Robust graph representation learning via neural sparsification. In *International Conference on Machine Learning*, pp. 11458–11468. PMLR, 2020.

Xiaojin Zhu. *Semi-supervised learning with graphs*. Carnegie Mellon University, 2005.

A Complexity

A.1 Complexity of Eq. 8

As mentioned above, by changing the order of matrix multiplication, the time complexity can be reduced, the Eq. 8 is as follows:

$$\begin{aligned}\mathbf{Q}^{(i)} &= \mathbf{H}(\mathbf{Q}^{(i-1)})^T \left(\frac{1}{\beta}\mathbf{I}_N - \frac{1}{\beta^2}\mathbf{H}\left(\mathbf{I}_c + \frac{1}{\beta}\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\right)\mathbf{Q}^{(i-1)} \\ &= \mathbf{H}(\mathbf{Q}^{(i-1)})^T \left(\frac{1}{\beta}\mathbf{Y} - \frac{1}{\beta^2}\mathbf{H}\left(\mathbf{I}_c + \frac{1}{\beta}\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{Q}^{(i-1)}\right).\end{aligned} \tag{12}$$

We first let $\mathbf{B} = \frac{1}{\beta^2}\mathbf{H}\left(\mathbf{I}_c + \frac{1}{\beta}\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{Q}^{(i-1)}$ and compute it from right to left. Specifically, the matrix inversion operation on a $c \times c$ matrix is $\mathcal{O}(c^3)$. Therefore, the overall time complexity of $\mathbf{S} \in \mathbb{R}^{n\times c}$ is $\mathcal{O}(nc^2 + c^3)$, where $c \ll n$. Then we can compute $\mathbf{H}(\mathbf{Q}^{(i-1)})^T\mathbf{B}$, likewise, we calculate it from right to left, this can reduce the time complexity from $\mathcal{O}(n^2c)$ to $\mathcal{O}(nc^2)$. Therefore the overall time complexity of calculating Eq. 8 is $\mathcal{O}(nc^2 + c^3)$. This significantly improves the model efficiency.

A.2 Complexity of Eq. 9

Calculating $\mathbf{S}^*$ by eq.(7) will result in $\mathcal{O}(n^3)$ computational cost, which leads to significant memory overhead on large datasets. Thus, we first use the Woodbury identity matrix transformation by Appendix B.3, then the Eq. 7 can be transformed as:

$$\mathbf{S}^* = \mathbf{H}(\mathbf{Q}^{(i)})^T\left(\mathbf{H}\mathbf{H}^T + \beta\mathbf{I}_N\right)^{-1} = \mathbf{H}(\mathbf{Q}^{(i)})^T\left(\frac{1}{\beta}\mathbf{I} - \frac{1}{\beta^2}\mathbf{H}\left(\mathbf{I}_c + \frac{1}{\beta}\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\right). \tag{13}$$

Then, we can transform the calculation order to reduce memory and time overhead as follows:

$$\begin{aligned}\mathbf{S}^* &= \mathbf{H}(\mathbf{Q}^{(i)})^T\left(\frac{1}{\beta}\mathbf{I} - \frac{1}{\beta^2}\mathbf{H}\left(\mathbf{I}_c + \frac{1}{\beta}\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\right) \\ &= \mathbf{H}(\frac{1}{\beta}(\mathbf{Q}^{(i)})^T - \frac{1}{\beta^2}(\mathbf{Q}^{(i)})^T\mathbf{H}\left(\mathbf{I}_c + \frac{1}{\beta}\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T)\end{aligned} \tag{14}$$

We first let $\mathbf{P} = \frac{1}{\beta^2}(\mathbf{Q}^{(i)})^T\mathbf{H}\left(\mathbf{I}_c + \frac{1}{\beta}\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T$ and calculate $(\mathbf{Q}^{(i)})^T\mathbf{H}$, wich time complexity is $\mathcal{O}(nc^2)$, then we can get a $c \times c$ matrix $(\mathbf{Q}^{(i)})^T\mathbf{H}$, the time complexity of $\left(\mathbf{I}_c + \frac{1}{\beta}\mathbf{H}^T\mathbf{H}\right)^{-1}$ is $\mathcal{O}(c^3)$, thus the overall complexity of $\mathbf{P}$ is $\mathcal{O}(nc^2 + c^3)$. Finally, the complexity of $\mathbf{HP}$ is $\mathcal{O}(n^2c)$, since c is the number of classes, it have $c \ll n$. Therefore, the complexity grows quadratically with the number of samples *i.e.,* $\mathcal{O}(n^2)$.

B Theoretical Proof

B.1 Proof for Theorem 2.1

Proof. To prove Theorem 2.1, we first introduce a lemma to describe the influence of a node on the other node:

Lemma B.1. *(Xu et al., 2018) Assume that the activation function of GCN is* ReLU. *Let $P_k^{a\to b}$ be a path $[v^{(k)}, v^{(k-1)}, \cdots, v^{(0)}]$ of length k from node v_a to node v_b, where $v^{(k)} = v_a, v^{(0)} = v_b$, and $v^{(i-1)} \in \mathcal{N}_{v^{(i)}}$ for $i = k, \cdots, 1$. Then we have the influence of node v_a on v_b is:*

$$I\left(v_b, v_a; k\right) = \sum_{P_k^{b\to a}} \prod_{i=k}^{1} \tilde{a}_{v^{(i-1)},v^{(i)}}, \tag{15}$$

where $\tilde{a}_{v^{(i-1)},v^{(i)}}$ is the weight of the edge $(v^{(i)}, v^{(i-1)})$.

The total influence is to sum over all lengths path. From Lemma B.1, we can easily obtain the influence of all labeled nodes with label y_1 on v_a is

$$I\left(\{v_b : y_v = y_1\}, v_a\right) = \sum_{v_b:y_b=y_1} \sum_{j=1}^{k} \sum_{P_j^{b\to a}} \prod_{i=j}^{1} \tilde{a}_{v^{(i-1)},v^{(i)}}. \tag{16}$$

For LPA, is a random walk algorithm starting from the label node, we denote the classified probability of node v_a in the y_1 dimension (*i.e.,* y_1 category) as $y_a[y_1]$. It is clear that

$$y_a\left[y_1\right] = \frac{y_a\left[y_1\right]'}{\sum_{y_i \in y} y_a\left[y_i\right]} \quad s.t., \quad y_a\left[y_1\right]' = \sum_{v_b:y_b=y_1} \sum_{j=1}^{k} \sum_{P_j^{b\to a}} \prod_{i=j}^{1} \tilde{a}_{v^{(i-1)},v^{(i)}}. \tag{17}$$

Thus, we can get $y_a\left[y_1\right] \propto I\left(\{v_b : y_v = y_1\}, v_a\right)$.

□

B.2 Closed-Form Solution

Given the objective function in Eq. 6, we let

$$\begin{aligned} \mathcal{L} &= \left\|\mathbf{Q}^{(i)} - \mathbf{SH}\right\|_F^2 + \beta \sum_{i,j=1} s_{i,j}^2 \\ &= Tr((\mathbf{Q}^{(i)} - \mathbf{SH})^T(\mathbf{Q}^{(i)} - \mathbf{SH})) + 2\beta\mathbf{S} \end{aligned} \tag{18}$$

where $Tr(\cdot)$ indicates the trace of matrix. Then we have

$$\frac{\partial \mathcal{L}}{\partial \mathbf{S}} = -2(\mathbf{Q}^{(i)})^T\mathbf{H} + 2\mathbf{SHH}^T + 2\beta\mathbf{S} \tag{19}$$

Let Eq. 19 equal to 0, we can obtain the closed-form solution $\mathbf{S}^{(i)}$ *i.e.,*

$$\mathbf{S}^{(i)} = \mathbf{H}(\mathbf{Q}^{(i)})^T\left(\mathbf{HH}^T + \beta\mathbf{I}_N\right)^{-1}. \tag{20}$$

B.3 THE WOODBURY IDENTITY

Given four matrices *i.e.,* $\mathbf{A} \in \mathbb{R}^{n\times n}$, $\mathbf{U} \in \mathbb{R}^{n\times k}$, $\mathbf{B} \in \mathbb{R}^{k\times k}$, $\mathbf{V} \in \mathbb{R}^{k\times n}$. We adopt a variation commonly used by the Woodbury identity (Woodbury, 1950) is as follows:

$$(\mathbf{A}+\mathbf{UBV})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}\left(\mathbf{B}^{-1}+\mathbf{VA}^{-1}\mathbf{U}\right)^{-1}\mathbf{VA}^{-1} \tag{21}$$

Without loss of generality, the matrix $\mathbf{A}$ and $\mathbf{B}$ can be replaced with the identity matrix, therefore, we further have

$$(\mathbf{I}+\mathbf{UV})^{-1} = \mathbf{I} - \mathbf{U}(\mathbf{I}+\mathbf{VU})^{-1}\mathbf{V} \tag{22}$$

We can replace the matrices $\mathbf{U}, \mathbf{V}$ with the matrix $\mathbf{H}$ in Eq. 22, thus, we have:

$$\left(\mathbf{HH}^T+\beta\mathbf{I}_N\right)^{-1} = \frac{1}{\beta}\mathbf{I} - \frac{1}{\beta^2}\mathbf{H}\left(\mathbf{I}_c+\frac{1}{\beta}\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T. \tag{23}$$

Therefore, based on Eq. 23, we can transform $\mathbf{Q}^{(i)} = \mathbf{S}^{(i-1)}\mathbf{Q}^{(i-1)}$ as:

$$\begin{aligned}\mathbf{Q}^{(i)} &= \mathbf{S}^{(\mathbf{i-1})}\mathbf{Q}^{(i-1)} \\ &= \mathbf{H}(\mathbf{Q}^{(i-1)})^T\left(\frac{1}{\beta}\mathbf{I}_N - \frac{1}{\beta^2}\mathbf{H}\left(\mathbf{I}_c+\frac{1}{\beta}\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\right)\mathbf{Q}^{(i-1)}.\end{aligned} \tag{24}$$

B.4 PROOF FOR THEOREM 2.3

Theorem B.2. *Given a graph $\mathcal{G}$ with adjacency matrix $\mathbf{A}$, training set node label $\mathbf{Y}$ and ground truth label $\mathbf{Y}_{\text{true}}$. For any unknown-label nodes, if $\mathbf{Y}_{\text{true}} = LPA(\mathbf{A}, \mathbf{Y})$, then the upper bound of the GCN's generalization ability reaches optimal on graph $\mathcal{G}$.*

Proof. To prove the Theorem 2.3, We first introduce the Complexity Measure to help us understand the generalization ability of GCN. It is the current mainstream method to measure the generalization ability of the model (Neyshabur et al., 2017), which describes the **a lower complexity measure means a better generalization ability**. We follow (Natekar & Sharma, 2020) to adopt Consistency of Representations as our Complexity Measure, which is designed based on the Davies-Bouldin Index (Davies & Bouldin, 1979). Formally, for a given dataset and a given layer of a model, the Davies-Bouldin Index can be written as follows:

$$S_a = \left(\frac{1}{n_a}\sum_{\tau}^{n_a}\left|O_a^{(i)} - \mu_{O_a}\right|^p\right)^{1/p} \quad \text{for } a = 1\cdots k \tag{25}$$

$$M_{a,b} = \left\|\mu_{O_a} - \mu_{O_b}\right\|_p \quad \text{for } a, b = 1\cdots k, \tag{26}$$

where a, b are two different classes, $O_a^{(i)}$ is the GCN smoothed feature of node i belonging to class a, μ_{O_a} is the cluster centroid of the representations of class a, here we set $p = 2$, thus S_a measures the intra-class distance of class a and $M_{a,b}$ is a measure of inter-class distance between class a and b. Then, we can define complexity measure based on the Davies-Bouldin Index as follows:

$$C = \frac{1}{k}\sum_{i=0}^{k-1}\max_{a\neq b}\frac{S_a+S_b}{M_{a,b}}. \tag{27}$$

We define P_0 as the probability that a node's neighbor belongs to the '0-th' class, and I_0 as the probability that the node itself belongs to the '0-th' class. Thus, we can calculate the cluster centroid after GCN smoothed features:

$$\begin{aligned}\mu_{O_0} = \mathbb{E}[O_0^i] &= \mathbb{E}[\mathbf{W}\sum_{j\in\mathcal{N}_i}\frac{1}{d_i}\mathbf{X}^j] \\ &= \mathbf{W}(I_0P_0\mu_{X_0} + I_0(1-P_0)\mu_{X_1}),\end{aligned} \tag{28}$$

where $\mathbf{X}^j$ is the 'j-th' node feature and μ_{X_i} is the cluster centroid of the node features of class i. Likewise, we have:

$$\mu_{O_1} = \mathbf{W}(I_1 P_1 \mu_{X_1} + I_1(1-P_1)\mu_{X_0}). \tag{29}$$

Then, the $M_{0,1}$ can be computed by:

$$\begin{aligned} M_{0,1} &= \|\mu_{O_a} - \mu_{O_b}\| \\ &= \|\mathbf{W}(I_0 P_0 \mu_{X_0} + I_0(1-P_0)\mu_{X_1} - (I_1 P_1 \mu_{X_1} + I_1(1-P_1)\mu_{X_0}))\| \\ &= \|\mathbf{W}(I_0 P_0 \mu_{X_0} + I_0 \mu_{X_1} - I_0 P_0 \mu_{X_1} - I_1 P_1 \mu_{X_1} - I_1 \mu_{X_0} + I_1 P_1 \mu_{X_0})\| \\ &= (I_0 P_0 + I_1 P_1)\,\|\mathbf{W}(\mu_{X_0} - \mu_{X_1})\| + \|I_0 \mu_{X_1} - I_1 \mu_{X_0}\| \\ &\leq (I_0 P_0 + I_1 P_1)\,\|\mathbf{W}(\mu_{X_0} - \mu_{X_1})\| + \|\mu_{X_1}\| + \|\mu_{X_0}\|. \end{aligned} \tag{30}$$

Then S_0^2 is calculated by:

$$\begin{aligned} S_0^2 &= \mathbb{E}\left[\left\|O_0^{(i)} - \mu_{O_0}\right\|^2\right] = \mathbb{E}\left[< O_0^{(i)} - \mu_{O_0}, O_0^{(i)} - \mu_{O_0} >\right] \\ &= \mathbb{E}[(I_0 P_0)(I_0 P_0 (X_0 - \mu_{X_0})^T \mathbf{W}^T \mathbf{W}(X_0 - \mu_{X_0}))] \\ &+ \mathbb{E}[I_0(1-P_0) I_0(1-P_0)(X_1 - \mu_{X_1})^T \mathbf{W}^T \mathbf{W}(X_1 - \mu_{X_1}))] \\ &= I_0^2 P_0^2 \mathbb{E}[\|\mathcal{W}(X_0 - \mu_{X_0})\|] + I_0^2 (1-P_0)^2 \mathbb{E}[\|\mathcal{W}(X_1 - \mu_{X_1})\|]. \end{aligned} \tag{31}$$

Similarly, we have:

$$\begin{aligned} S_1^2 &= \mathbb{E}\left[\left\|O_1^{(i)} - \mu_{O_1}\right\|^2\right] = \mathbb{E}\left[< O_1^{(i)} - \mu_{O_1}, O_1^{(i)} - \mu_{O_1} >\right] \\ &= \mathbb{E}[(I_1 P_1)(I_1 P_1 (X_1 - \mu_{X_1})^T \mathbf{W}^T \mathbf{W}(X_1 - \mu_{X_1}))] \\ &+ \mathbb{E}[I_1(1-P_1) I_1(1-P_1)(X_0 - \mu_{X_0})^T \mathbf{W}^T \mathbf{W}(X_0 - \mu_{X_0}))] \\ &= I_1^2 P_1^2 \mathbb{E}[\|\mathcal{W}(X_1 - \mu_{X_1})\|] + I_1^2 (1-P_1)^2 \mathbb{E}[\|\mathcal{W}(X_0 - \mu_{X_0})\|], \end{aligned} \tag{32}$$

where $< \cdot, \cdot >$ is inner production. For simplicity, let $\sigma_0^2 = \mathbb{E}[\|\mathcal{W}(X_0 - \mu_{X_0})\|]$ and $\sigma_1^2 = \mathbb{E}[\|\mathcal{W}(X_1 - \mu_{X_1})\|]$, then the above equation can then be simplified to:

$$S_0^2 = (I_0 P_0)^2 \sigma_0^2 + (I_0(1-P_0))^2 \sigma_1^2 \geq I_0^2 \frac{\sigma_0^2 \sigma_1^2}{\sigma_0^2 + \sigma_1^2}. \tag{33}$$

Similarly, we have:

$$S_1^2 = (I_1 P_1)^2 \sigma_1^2 + (I_1(1-P_1))^2 \sigma_1^2 \geq I_1^2 \frac{\sigma_0^2 \sigma_1^2}{\sigma_0^2 + \sigma_1^2}. \tag{34}$$

Then the complexity measure can be represented as:

$$C = \frac{\sqrt{S_0^2 + S_1^2 + 2 S_0 \cdot S_1}}{M_{0,1}} \geq \frac{2\sigma_0 \sigma_1 (I_0 + I_1)^2}{\sqrt{\sigma_0^2 + \sigma_1^2} \cdot ((I_0 P_0 + I_1 P_1)\,\|\mathbf{W}(\mu_{X_0} - \mu_{X_1})\| + \|\mu_{X_1}\| + \|\mu_{X_0}\|)}. \tag{35}$$

Thus, we obtain a lower bound of complexity measure. Also this is the upper bound of the generalization ability. Notice that σ_0 and σ_1 could not be zero, otherwise, the classification problem is meaningless. We observe the above equation for nodes with unknown labels and analysis the relationship between the distribution of label I_0, I_1 and the lower bound of complexity measure, we find that the probability of their own label (*i.e.,* I_0 or I_1) and the probability of their neighbors' labels (*i.e.,* P_0 or P_1) affect the upper bound on their generalization ability. Since $I_0 + I_1 = 1$, we analyze term $(I_0 P_0 + I_1 P_1)$,

$$(I_0 P_0 + I_1 P_1) = \frac{1}{n} \sum_i^n I_{0,i} P_{0,i} + I_{1,i} P_{1,i} \tag{36}$$

where $I_{0,i} \in \{0, 1\}$ is the binary probability that the 'i-th' node label belongs to class 0 where $I_{1,i} = 1 - I_{0,i}$ and $P_{0,i}$ is the probability that the 'i-th' node whose neighbor belongs to class 0. In order to minimize the lower bound of complexity measure, *i.e.,* to maximize the upper bound of generalization ability, it is necessary to maximize $(I_0P_0 + I_1P_1)$ here. Obviously, the maximum $(I_0P_0 + I_1P_1)$ is obtained at $I_{0,i} = argmax(P_{1,i}P_{0,i})$.

Let's look at the Label Propagation Algorithm(LPA). For nodes with unknown labels,

$$\widehat{y}_i = \frac{1}{d_i} \sum_{j \in \mathcal{N}_i} y_j. \tag{37}$$

Then the probability that the LPA predicts that the 'i-th' node belongs to class 0 can be obtained:

$$\widehat{I}_{0,i} = argmax(\frac{1}{d_i} \sum_{j \in \mathcal{N}_i} y_i == 1, \frac{1}{d_i} \sum_{j \in \mathcal{N}_i} y_i == 0) = argmax(P_{1,i}P_{0,i}). \tag{38}$$

Similarly, the probability of predicting the 'i-th' node to belong to class 1 is:

$$\widehat{I}_{1,i} = argmax(\frac{1}{d_i} \sum_{j \in \mathcal{N}_i} y_i == 0, \frac{1}{d_i} \sum_{j \in \mathcal{N}_i} y_i == 1) = argmax(P_{0,i}P_{1,i}). \tag{39}$$

Thus, the upper bound on the generalization ability is maximized when the labels of the unknown label set are distributed as LPA-generated labels.

□

B.5 Proof for Theorem 2.4

Theorem B.3. *The ELU graph can ensure the generalization ability of the GCN, potentially bringing it closer to optimal performance.*

Proof. Recall our objective function (*i.e.,* Eq. (5)) $\min_{\mathbf{S}} \|\mathbf{SY} - \mathbf{SH}\|_F^2$, and we first pre-training a GCN (*i.e.,* $\mathbf{SH}$, where $\mathbf{H} = MLP(X)$ is trained in advance) to predict labels for all nodes (*i.e.,* $\widehat{\mathbf{Y}}$), thus our objective function can be rewritten as $\min_{\mathbf{S}} \left\|\mathbf{SY} - \widehat{\mathbf{Y}}\right\|_F^2$, which align with the form of $\min_{\mathbf{A}} \|\mathbf{AY} - \mathbf{Y}_{\text{true}}\|_F^2$ and $\widehat{\mathbf{Y}}$ is often used to estimate $\mathbf{Y}_{\text{true}}$ (Yang et al., 2024; Gong et al., 2023). Therefore, the ELU graph (*i.e.,* $\mathbf{S}$) can ensure the GCN's generalization ability to a certain extent. Moreover, a better adjacency matrix $\mathbf{S}$ can further improve the GCN's predictions (*i.e.,* $\widehat{\mathbf{Y}}$), making $\widehat{\mathbf{Y}}$ increasingly closer to ground truth (*i.e.,* $\mathbf{Y}_{\text{true}}$). Ultimately, we can obtain a graph structure to ensure the GCN's generalization ability is closer to optimal performance. □

C Related Works

This section briefly reviews the topics related to this work, including graph convolutional networks and graph structure learning.

C.1 Graph Convolutional Networks

Graph convolutional networks (GCNs) are the most popular and commonly used model in the field of graph deep learning. Early work attempted to apply the successful convolutional neural network to graph structures. For example, CheybNet (Defferrard et al., 2016) first propose that transform the graph signal from the spatial domain to the spectral domain through discrete Fourier transform, and then use polynomials to fit the filter shape (*i.e.,* convolution). CheybNet laid the foundation for the development of spectral domain graph neural networks. The popular GCN was proposed by Kipf et al. (Kipf & Welling, 2017), which is a simplified version of ChebyNet and has demonstrated strong efficiency and effectiveness, thereby promoting the development of the graph deep learning field.

Based on the traditional GCN, many advanced GCNs have been proposed. For example, numerous works are focused on increasing the number of GCN layers. APPNP (Gasteiger et al., 2018) combines

personalized random walks to expand the range of neighbors aggregated by GCN and reduce training time. JK-Net (Xu et al., 2018) integrates information from each GCN layer to enable better structure-aware representation. Additionally, there are many advanced GCNs have been proposed to increase the number of GCN layers (Chen et al., 2020; Liu et al., 2020; Wu et al., 2024). Recently, Li et al. (Li et al., 2022) developed a new framework i.e., concentration analysis, proposing a linear feature smooth method with flexible concentration properties. Huang et al. (Huang et al., 2024) found that GCN would fail on some nodes, which are often far away from the label nodes and have few neighbors, so they designed a powerful GCN model for these nodes. Liu et al. (Liu et al., 2024) proposed the graph adversarial diffusion convolution that can make GCN more robust. The core of current GCN methods is feature propagation, which allows label information to supervise the features of more nodes. However, to the best of our knowledge, no work has explored whether the label information effectively influences the features of the neighboring nodes within the GCN framework.

C.2 Graph Structure Learning

Before the recent rise of Graph Neural Networks, graph structure learning had already been extensively explored from various perspectives within the field of traditional machine learning.

Graph structure learning is an important technology in the graph field. It can improve the graph structure and infer new relationships between samples, thereby promoting the development of graph representation learning or other fields. Existing Graph structure learning methods can be classified into three categories, *i.e.,* traditional unsupervised graph structure learning methods, supervised graph structure learning methods, and graph rewiring methods. Traditional unsupervised graph structure learning methods aim to directly learn a graph structure from a set of data points in an unsupervised manner. Early works (Wang & Zhang, 2006; Daitch et al., 2009) exploit the neighborhood information of each data point for graph construction by assuming that each data point can be optimally reconstructed using a linear combination of its neighbors (*i.e.,* $\min_A \|\mathbf{AX}-\mathbf{X}\|_F^2$). Similarly, (Daitch et al., 2009) introduce the weight (*i.e.,* $\min \sum_i \left\|\mathbf{D}_{i,i}\mathbf{X}_i - \sum_j \mathbf{A}_{i,j}\mathbf{X}_j\right\|^2$). Smoothness Jiang et al. (2019) is another widely adopted assumption on natural graph signals, the smoothness of the graph signals is usually measured by the Dirichlet energy (*i.e.,* $\min_{\mathbf{A}} \frac{1}{2}\sum_{i,j}\mathbf{A}_{i,j}\|\mathbf{X}_i-\mathbf{X}_j\|^2 = \min_{\mathbf{L}} \operatorname{tr}\left(\mathbf{X}^\top\mathbf{L}\mathbf{X}\right)$). Until now, there have been a lot of works based on the above objective function to learn graph structure. Supervised graph structure learning methods aim to use the downstream task to supervise the structure learning, which can learn a suitable structure for the downstream task. For example, NeuralSparse (Zheng et al., 2020) and PTDNet (Luo et al., 2021) directly use the adjacency matrix of the graph as a parameter and update the adjacency matrix through the downstream task. SA-SGC (Huang et al., 2023b) learns a binary classifier by distinguishing the edges connecting nodes with the same label and the edges connecting nodes with different labels in the training set, thereby deleting the edges between nodes belonging to different categories in the test set. BAGCN (Zhang et al., 2024) uses metric learning to obtain new graph structures and learns suitable metric spaces through downstream tasks. The goal of graph rewiring methods is to prevent the over-squashing (Alon & Yahav, 2021) problem. For example, FA (Alon & Yahav, 2021) proposed to use a fully connected graph as the last layer of GCN to overcome over-squashing. SDRF (Topping et al., 2022), SJLR (Giraldo et al., 2023), and BORF (Nguyen et al., 2023) aim to enhance the curvature of the neighborhood by rewiring connecting edges with small curvature. They increase local connectivity in the graph topology indirectly expanding the influence range of labels. However, the graph structure obtained by the current graph structure learning methods can not guarantee that the GNN model can effectively utilize the supervisory information transmitted in the graph.

D Model Detail

D.1 Pretaining MLP

As mentioned in the method section *MLP has been trained in advance*. Specifically, we employ the two-layer MLP and crosse-entropy to pre-train the MLP:

$$\mathcal{L}_{mlp}: \min_{\Theta_1,\Theta_2} CE(\mathbf{X}\boldsymbol{\Theta}^{(1)}\boldsymbol{\Theta}^{(2)}, \mathbf{Y}) \tag{40}$$

where $\Theta^{(1)}$ and $\Theta^{(2)}$ are learnable parameters. After the above objective function converges by gradient descent algorithm, we can get $\mathbf{H}$ as follows:

$$\mathbf{H} = \mathbf{X}\boldsymbol{\Theta}^{(\mathbf{1})}\boldsymbol{\Theta}^{(\mathbf{2})}. \tag{41}$$

D.2 Details of Sparse $\mathbf{S}^*$ and Initialize $\mathbf{Y}$

Sparse $\mathbf{S}^*$. $\mathbf{S}^*$ is a fully-connected adjacency matrix. It will bring computationally expensive overhead in message passing, especially for large-scale graph datasets. To mitigate this, we set the elements with small absolute values to 0; Specifically, $\forall i, j$ where $|\mathbf{S}^*_{i,j}| < \eta$, we set $|\mathbf{S}^*_{i,j}| = 0$, while elements with $|\mathbf{S}^*_{i,j}| > \eta$ remain unchanged, where η is a non-negative parameter that we usually set to correspond to the top 10 percent of element values. The graph described by $\mathbf{S}^*$ is referred to as the effectively label-utilizing graph (ELU-graph) in this paper.

Initialize $\mathbf{Y}$. Since the number of initial label information is very limited in a semi-supervised scenario, having too many rows of all zeros in $\mathbf{Y}$ can cause the algorithm to be unstable. Thus, we propose a label initialization strategy to expand the initial labels with high quality. Specifically, since ELU nodes can effectively utilize the label information and demonstrate high accuracy as shown in Figure 2 (b), we use the pseudo labels of ELU nodes to expand the initial $\mathbf{Y}$.

E Pseudo Code

Algorithm 1 Pseudo code of calculating $\mathbf{S}^*$.

Input: Feature matrix $\mathbf{X}$, label matrix $\mathbf{Y}$, normalized adjacency matrix $\widehat{\mathbf{A}}$, and index of ELU nodes V_{ELU};
Output: ELU graph $\mathbf{S}^*$;

```
H = MLP(X);
Expand initial labels by pseudo labels of ELU nodes;
for i ← 1, 2, ··· , k do
    Calculate Q^(i) by Eq. (8);
    Q_l^(i) = Y_l in Eq. (8);
end for
Calculate S* Eq. (9);
return S*.
```

F Experiments Details

F.1 Datasets

Table 3: The statistics of the datasets

Datasets	Nodes	Edges	Train/Valid/Test Nodes	Features	Classes
Cora	2,708	5,429	140/500/1000	1,433	7
Citeseer	3,327	4,732	120/500/1,000	3,703	6
Pubmed	19,717	44,338	60/500/1,000	500	3
Amazon Computers	13,381	245,778	200/300/12,881	767	10
Amazon Photo	7,487	119,043	160/240/7,084	745	8
Chameleon	2,277	36,101	1,093/729/455	2,325	5
Squirrel	5,201	217,073	2,496/1,665/1,040	2,089	5
Caltech	13,882	763,868	8,240/2,776/2,776	6	6
UF	35,123	2,931,320	21,074/7,024/7,024	6	6
Hamilton	2,314	192,788	1,388/463/463	6	6
Tulane	7,752	567,836	4,652/1,550/1,550	6	6

The used datasets include three benchmark citation datasets (Sen et al., 2008) (*i.e.,* Cora, Citeseer, Pubmed), two co-purchase networks (Shchur et al., 2018) (*i.e.,* Computers, Photo), two web page networks (Pei et al.) (*i.e.,* Chameleon and Squirrel, note that these two datasets are heterophilic graph data), and four social network datasets (Traud et al., 2012) (*i.e.,* Caltech, UF, Hamilton, and Tulane). Table 3 summarizes the data statistics. We list the details of the datasets as follows.

- **Citation networks** include Cora, Citeseer, and Pubmed. They are composed of papers as nodes and their relationships such as citation relationships, and common authoring. Node feature is a one-hot vector that indicates whether a word is present in that paper. Words with a frequency of less than 10 are removed.
- **Co-purchase networks** include Photo and Computers, containing 7,487 and 13,752 products, respectively. Edges in each dataset indicate that two products are frequently bought together. The feature of each product is bag-of-words encoded product reviews. Products are categorized into several classes by the product category.
- **Webpage networks** include Squirrel and Chameleon, which are two subgraphs of web pages in Wikipedia. Our task is to classify nodes into five categories based on their average amounts of monthly traffic.
- **Social networks** include Caltech, UF, Hamilton, and Tulane, each graph describes the social relationship in a university. Each graph has categorical node attributes with practical meaning (e.g., gender, major, class year.). Moreover, nodes in each dataset belong to six different classes (a student/teacher status flag).

G Additional Experiments

G.1 Node Classification on Social Networks

Model	Caltech	UF	Hamilton	Tulane
GCN	88.47±1.91	83.94±0.61	92.26±0.35	87.93±0.97
GAT	81.17±2.15	81.68±0.59	91.43±1.25	84.45±1.45
APPNP	90.76±2.38	83.07±0.54	93.29±0.47	88.52±0.44
GCN-LPA	89.12±2.11	83.78±0.69	92.56±0.87	88.32±1.02
GSR	90.23±2.41	84.01±0.63	92.45±0.84	88.75±1.01
Ours	**91.93**±**0.69**	**85.62**±**0.53**	**93.65**±**0.78**	**89.30**±**0.77**

We further evaluate the effectiveness of the proposed method on the social network datasets by reporting the results of node classification. Obviously, our method achieves the best effectiveness on node classification tasks.

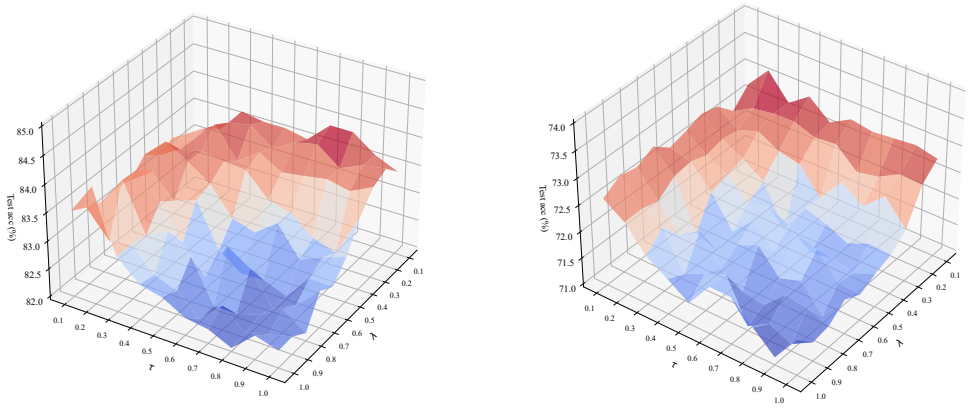
Specifically, the proposed method achieves competitive results on the social network datasets compared to other baselines. For example, the proposed method on average improves by 1.27%, compared to the best baseline (i.e., GSR), on almost all datasets. This demonstrates the universality of our method, which can achieve excellent results in most datasets.

G.2 Parameter Analysis

In the proposed method, we employ the non-negative parameters (*i.e.,* λ) to achieve a trade-off between the supervised loss and the consistency loss, and τ to achieve the temperature control. To investigate the impact of λ and τ with different settings, we conduct the node classification on the Cora and Citeseer datasets by varying the value of λ in the range of [0.1, 1.0] and τ in the range of [0.1, 1.0]. Note that the smaller the τ, the closer the model brings the positive samples and the further apart the negative samples. The results are reported in Figure 5.

From Figure 5, we have the following observations: First, the proposed method achieves significant performance when the parameter λ or τ is in the range of [0.1, 0.2]. if λ values are too large (e.g., >0.2) or too small (e.g., =0, the results shown in the Table 2), the performance degrades. This indicates that the proposed contrastive loss is necessary for the model. For τ, setting it to 1 is equivalent to

not using the temperature coefficient. The lower the temperature coefficient, the stronger the effect, indicating that τ is essential for the proposed method. Note that τ cannot be set to 0 because it is the denominator.



(a) Cora (b) Citeseer

Figure 5: The classification performance of the proposed method at different parameter settings (*i.e.,* τ, λ) on the Cora and Citeseer datasets.

G.3 Analysis the Improve on V_{NELU}

To better examine the effectiveness of the proposed ELU-GCN on V_{NELU}, we further evaluate the model's improvement over GCN on V_{NELU} on Cora, Citeseer, and Pubmed datasets. The results are shown in Figure 6.

Specifically, the proposed ELU-GCN shows a particularly significant improvement on V_{NELU} across the three datasets. For example, our method on average improves by 3.7 % on V_{NELU} and 2.1% on all test nodes compared to GCN on these three datasets. This can be attributed to the fact that the proposed ELU-GCN provides the ELU graph that can make V_{NELU} utilize the label information more effectively under the GCN framework, and this also indicates that the main improvement of the proposed ELU-GCN is on V_{NELU}.

H Running Time V.S. Accuracy

The biggest limitation of graph structure learning methods is the need to query in $\mathcal{O}(n^2)$ space when learning graph structures. Although we have previously analyzed that the complexity of the proposed algorithm in graph construction is $\mathcal{O}(nc^3)$ ($c^3 \ll n$), plus the final graph structure is $\mathcal{O}(n^2)$ (only one calculation is required), we further test the overall actual running time of the proposed ELU-GCN and compared with the commonly used baseline (*i.e.,* GCN and GAT). The results are in Figure 7.

From Figure 7, we have the observations as follows. First, the overall running time of the proposed method is slightly inferior to GCN, but significantly ahead of GAT. Second, the proposed method achieves the best classification performance. Combining the above two points, the proposed method achieves the optimal trade-off between running time and model performance.

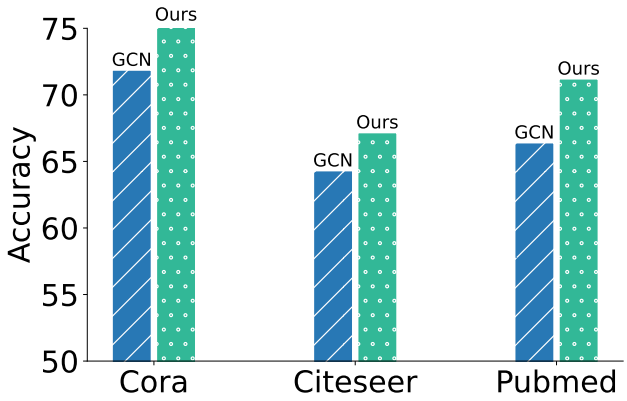


Figure 6: The accuracy of ELU-GCN and GCN of V_{NELU} on Cora, Citeseer, and Pubmed datasets.

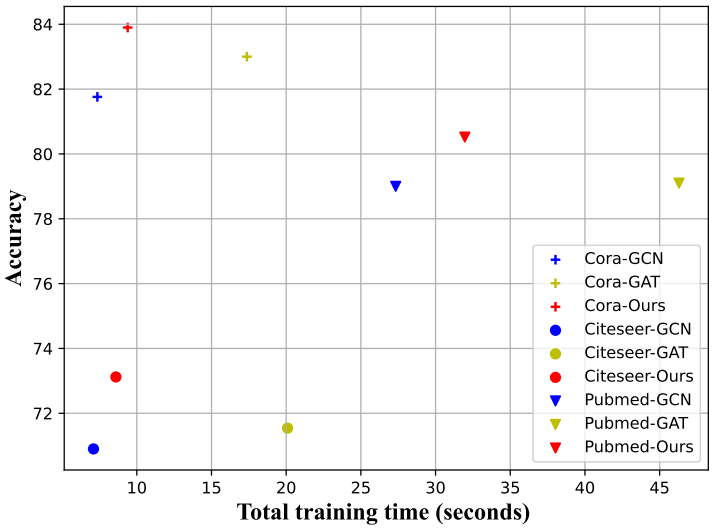


Figure 7: Running time V.S. accuracy.