Toward Robust Graph Semi-Supervised Learning Against Extreme Data Scarcity

Kaize Ding, Elnaz Nouri, Guoqing Zheng, Huan Liu, *Life Fellow, IEEE*, and Ryen White

***Abstract*— The success of graph neural networks (GNNs) in graph-based web mining highly relies on abundant human-annotated data, which is laborious to obtain in practice. When only a few labeled nodes are available, how to improve their robustness is key to achieving replicable and sustainable graph semi-supervised learning. Though self-training is powerful for semi-supervised learning, its application on graph-structured data may fail because 1) larger receptive fields are not leveraged to capture long-range node interactions, which exacerbates the difficulty of propagating feature-label patterns from labeled nodes to unlabeled nodes and 2) limited labeled data makes it challenging to learn well-separated decision boundaries for different node classes without explicitly capturing the underlying semantic structure. To address the challenges of capturing informative structural and semantic knowledge, we propose a new graph data augmentation framework, augmented graph self-training (AGST), which is built with two new (i.e., structural and semantic) augmentation modules on top of a decoupled GST backbone. In this work, we investigate whether this novel framework can learn a robust graph predictive model under the low-data context. We conduct comprehensive evaluations on semi-supervised node classification under different scenarios of limited labeled-node data. The experimental results demonstrate the unique contributions of the novel data augmentation framework for node classification with few labeled data.**

***Index Terms*— Data scarcity, graph neural networks (GNNs), robustness, self-training.**

I. Introduction

With the rapid development of the Worldwide Web, in recent years, we have witnessed the growth in our ability to generate and gather data on numerous online and offline platforms. Graphs, where entities are denoted as nodes and the relations connecting them are denoted as edges, have become a common language for modeling a plethora of structured and relational systems on the web, ranging from social networks [1] to knowledge graphs [2], to e-commerce user–item interaction graphs [3]. To ingest the valuable information encoded in graph-structured data, graph learning algorithms have been proposed in the research community and made a huge success in different domains. Recently, graph neural networks (GNNs), a generalized form of neural networks for graph-structured data, have become the prevailing paradigm due to their effectiveness and scalability [4], [5], [6].

As a typical graph-related learning task, node classification has received continuous endeavors in the research community [7]. Existing GNNs developed for node classification usually focus on the canonical *semi-supervised* setting where relatively abundant gold-labeled nodes are provided [7]. While this setting is often impractical since data labeling is extremely labor intensive, especially when considering the heterogeneity of graph-structured data [8], [9]. To overcome the data scarcity issue, self-training or pseudo-labeling [10] has been explored to combine with GNNs and proven to be effective for solving semi-supervised node classification with fewer labels [11], [12], [13].

Existing GST methods, however, simply combine the idea of self-training with GNNs, which can be ineffective in handling graph data with few labeled nodes, or in exploiting numerous unlabeled nodes due to two limitations.

1) *Structural Bottleneck:* Given a few labeled nodes, it is important for the GNN model to enable more propagation steps, so the feature patterns of labeled nodes can be better propagated to the long-distance unlabeled nodes. However, recent work pointed out the distortion of information flowing from distant nodes (i.e., over-squashing [14], [15]) as a factor limiting the efficiency of message-passing for tasks relying on long-range node interactions. In addition, real-world graphs often come with a certain level of structure noise (e.g., "noisy" and "missing" edges), which could be generated by either adversaries [16], [17] or the data collection process itself [18]. Such structure noise can easily interfere with the message-passing process and make it difficult to learn correct feature-label patterns with few labeled nodes. Hence, it is crucial to avoid over-squashing and reduce data noise in our endeavor to further improve the performance of GST with few labeled nodes.
2) *Semantic Bottleneck:* It is challenging to learn well-separated decision boundaries between different node classes when labeled training data is severely scarce and the semantic manifold is complex. Though GST methods attempt to alleviate the data scarcity problem by adding pseudo-labels, the nodes with pseudo-labels may introduce complex feature patterns and pseudo-labels can be unreliable, which causes the model performance to deteriorate. We ask if novel ideas can be explored to optimize the usage of pseudo-labels for capturing the

Manuscript received 26 February 2023; revised 9 July 2023 and 12 November 2023; accepted 22 December 2023. *(Corresponding author: Kaize Ding.)*

Kaize Ding is with the Department of Statistics and Data Science, Northwestern University, Evanston, IL 60208 USA (e-mail: kaize.ding@northwestern.edu).

Elnaz Nouri, Guoqing Zheng, and Ryen White are with Microsoft Research, Redmond, WA 98052 USA (e-mail: Elnaz.Nouri@microsoft.com; Guoqing.Zheng@microsoft.com; Ryen.White@microsoft.com).

Huan Liu is with the School of Computing and Augmented Intelligence, Arizona State University, Tempe, AZ 85281 USA (e-mail: huan.liu@asu.edu).

Digital Object Identifier 10.1109/TNNLS.2024.3351938

underlying semantic structure of the sparsely-labeled graph.

In this article, we propose an augmented graph self-training framework, namely AGST, for tackling semi-supervised node classification where few labeled nodes are available. We plan to address the limitations of conventional GST methods by proposing two original modules for *structural and semantic data augmentations*. Specifically, our framework employs a simple, decoupled GNN as the GST backbone, where the teacher model first performs high-order label propagation (LP) to generate pseudo-labels on unlabeled nodes based on personalized PageRank [19], and the student model conducts feature transformation (FT) by mapping the features of nodes to their gold/pseudo-labels. From the *structural data augmentation* perspective, our framework not only enables large receptive fields to capture long-range node interactions, but also avoids the over-squashing issue by decoupling the transformation and propagation steps in message passing. To further promote information propagation, in each GST iteration, a deterministic topology augmentation function is derived from the learned model and is utilized to refine the input topological structure for the next iteration. This way, we expect that AGST can capture richer (i.e., both local and global) and cleaner (i.e., less noisy) structure knowledge during the GST process. From the *semantic data augmentation* perspective, the pseudo-labels introduced by the teacher model can enrich the semantic knowledge of the training data when learning the student model. To optimize the semantic alignment between the few labeled nodes and generated pseudo-labeled nodes, we suggest explicitly capturing the semantic structures of the input graph by proposing a weakly supervised contrastive loss to encourage intraclass compactness and interclass separability in the latent feature space. As such, well-separated decision boundaries can be learned during the GST process even with few labeled nodes. The proposed AGST framework enables the two data augmentation modules to work seamlessly with the decoupled GST backbone and to quickly learn a robust graph predictive model even with few labeled nodes. To summarize, our key contributions are listed as follows:

1) *Problem*: We investigate the problem of semi-supervised node classification under the challenging low-data setting, which focuses on improving the replicability and sustainability of GNNs in practical scenarios.
2) *Algorithm*: We propose a principled GST framework, which differs from the existing efforts and improves the performance with scarce labeled data by augmenting data from both structural and semantic perspectives.
3) *Evaluation*: We conduct extensive experiments on various real-world datasets to evaluate the effectiveness of our approach. The experimental results demonstrate the unique contributions made by AGST to performance improvement over existing methods. Data and code are available at https://github.com/kaize0409/AGST.

II. Related Work

A. Graph Neural Networks

GNNs, a family of neural endeavors for learning latent node representations on a graph, have drawn much attention in the community of graph machine learning (GML) [4], [5], [20], [21]. In general, GNNs can be categorized into spectral [4], [20], [21], [22] and spatial approaches [5], [6], [23]. Originally inspired by graph spectral theory, spectral-based graph convolutional networks (GCNs) extend convolution operation in the spectral domain to graph-structured data. Among them, the model proposed by Kipf and Welling [4] has become the most prevailing one by using a linear filter. Later on, SGC [22] is proposed to further reduce the computational complexity by removing the nonlinearity of GCNs. As another line of work, spatial-based GNNs define graph convolutions based on a node's spatial relations [5], [6], [23]. For example, GAT [5] incorporates trainable attention weights to specify fine-grained weights on neighbors when aggregating neighborhood information of a node. In essence, although spectral-based and spatial-based GNNs start on a different basis, both of them share the same propagation rule, which is the message-passing scheme. Those methods model the homophily principle [24] and learn node representations by iteratively transforming, and propagating/aggregating node features within graph neighborhoods. When long-range node interactions are needed, the over-squashing issue can largely undermine the model performance if we directly increase the model depth. Thus, researchers try to solve this issue by proposing different techniques, such as self-attention mechanism [25], sampling or rewiring edges [6], [26], decoupling the FT and propagation steps [27], [28], [29], [30], [31], [32], and many others [11], [33]. In particular, decoupled GNNs [29], [31] have become a prevailing paradigm due to their simplicity and learning efficiency. For example, APPNP [27] propagates the neural predictions via personalized PageRank, which can preserve the node's local information while increasing the receptive fields. Liu et al. [28] propose to decouple the propagation and transformation steps and then utilize an adaptive adjustment mechanism to balance the information from the local and global neighborhoods of each node. However, the aforementioned models neglect the additional supervision signals from unlabeled data, which has become a bottleneck for pushing the performance boundary of GNNs. Though CGPN [34] leverages Poisson learning to propagate the labels to the entire graph, it cannot address the structure noise and explicitly capture the semantic structures of the input graph.

B. Node Classification With Few Labels

In real-world scenarios, labeled training samples are usually quite limited due to the intensive cost of data labeling. Albeit the great success of GNNs for graph-based semi-supervised learning, most of the existing efforts are designed as shallow models with a restricted receptive field, leading to their ineffectiveness in limited labeled data scenarios [28], [29], [35]. Under the extreme cases when very few labels are given, shallow GNNs cannot effectively propagate the training labels and characterize the global information of the input graph [11]. Many advanced deep GNNs [27], [28], [33], [36], [37] have shown their advantages in leveraging large receptive fields for propagating label signals. However, the main concern in semi-supervised node classification, that is, the shortage

of supervision information, has not been directly addressed. To counter this issue, self-training [38], also known as pseudo-labeling [10], where one imputes labels on unlabeled data based on a teacher model trained with limited labeled data, has been applied to improve GNNs to solve the problem of semi-supervised node classification. Among those methods, Li et al. [11] first combine GCNs and self-training to expand supervision signals. Furthermore, M3S [12] proposed multistage self-training and utilized the clustering method to eliminate the pseudo-labels that may be incorrect. Similar ideas can also be found in [39], [40], and [41]. However, existing methods still adopt shallow GNNs to build the teacher and student models, inherently restricting the effective propagation of label signals. Apart from the aforementioned methods, our AGST framework adopts a decoupled design, in which the teacher model is an LP module and the student model is an FT module. As such, our framework is capable of leveraging both large receptive fields and weak supervision signals, making the learned model more label-efficient. Additionally, we propose a weakly supervised contrastive loss and a graph topology augmentation function to further improve model performance during the self-training process.

C. Contrastive Learning

Contrastive learning methods have demonstrated promising outcomes in self-supervised representation learning. In general, contrastive learning performs data augmentation on input data and learns expressive representations by pulling together the augmented views of the same example while pushing away negative examples. For instance, MoCo [42] utilizes a memory queue to store consistent representations, while SimCLR [43] optimizes InfoNCE within mini-batches and incorporates effective training techniques like data augmentation. Recently, prototypical contrastive learning (PCL) [44] has been proposed, which uses cluster centroids as prototypes, and trains the network by pulling each instance closer to its assigned prototypes. However, these unsupervised contrastive learning approaches primarily focus on inducing transferable representations for downstream tasks rather than training with noisy labels. Although supervised contrastive learning [45] improves representations using human-annotated labels, it suffers from performance degradation in the presence of label noise [46], [47].

In the meantime, contrastive learning has also been successfully applied to graph-structured data to improve graph representation learning [48], [49], [50], [51], [52], [53], [54]. Different from the existing graph contrastive learning methods that mostly focus on unsupervised graph representation learning, the proposed weakly supervised graph contrastive learning algorithm in this work mitigates the label noise in the pseudo-node labels by aligning the semantic structure between labeled nodes and pseudo-labeled nodes. This weakly supervised learning loss helps improve the node representation learning when labeled nodes are extremely limited.

III. Proposed Approach

We start by introducing the notations used throughout this article. We let bold uppercase letters represent matrices and bold lowercase letters denote vectors. Let $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ denote an undirected graph with nodes $\mathcal{V}$ and edges $\mathcal{E}$. Let n denote the number of nodes and m the number of edges. The nodes in G are described by the attribute matrix $\mathbf{X} \in \mathbb{R}^{n\times f}$, where f denotes the number of features per node. The graph structure of G is described by the adjacency matrix $\mathbf{A} \in \{0,1\}^{n\times n}$, while $\tilde{\mathbf{A}}$ stands for the adjacency matrix for a graph with added self-loops. We let $\tilde{\mathbf{D}}$ be the diagonal degree matrix of $\tilde{\mathbf{A}}$ and $\mathbf{S} = \tilde{\mathbf{D}}^{-(1/2)}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-(1/2)}$ denote the symmetric normalized adjacency matrix with self-loops. The class (or label) matrix is represented by $\mathbf{Y} \in \mathbb{R}^{n\times c}$, where c denotes the number of classes.

Problem Definition: Given an input graph $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, where the node set $\mathcal{V}$ is composed by two disjoint node subsets $\mathcal{V}^L$ and $\mathcal{V}^U$. In this article, we focus on the semi-supervised node classification task under the limited labeled data setting. Specifically, suppose that the labels of the labeled training set $\mathcal{V}^L$ are given, where each node class in $\mathcal{V}^L$ only has a few labeled nodes (could be either balanced or imbalanced for different classes), the goal is to predict the labels of the unlabeled nodes in $\mathcal{V}^U$. Note that if each class has the same number of K-labeled nodes, the studied problem can be called few-shot semi-supervised node classification.

Architecture Overview: In this section, we propose an AGST framework to solve the problem of semi-supervised node classification with only a few labeled nodes. Compared to existing efforts, AGST can address the structural and semantic bottlenecks under the limited labeled data setting by virtue of two focal designs: 1) a new GST backbone with a graph topology augmentation function that can leverage long-range node interactions while alleviating the structure noise and 2) a weakly supervised contrastive loss that enhances the semantic structures of the input graph by aligning the semantic similarities between pseudo-labeled data and gold-labeled data. A detailed illustration of the proposed approach can be found in Fig. 1.

A. Augmenting Structural Knowledge in GST

For semi-supervised node classification, graph neural predictors commonly have a large variance and are easy to overfit when the labeled training data is extremely limited [11], [12]. Although previous GST methods partially alleviate this issue by expanding the labeled training set, they still suffer from the incapability of leveraging long-range node interactions and handling structure noise in nature: on the one hand, if the teacher and student model share the same shallow GNN architecture, the over-squashing issue will largely impede the effective propagation of feature-label patterns when multiple layers are deployed; on the other hand, the missing or noisy edges in the input graph may also distort the information flow. To better exploit the useful graph structural knowledge, we go beyond the existing GST architectures and develop a decoupled GST backbone integrated with a structural data augmentation module.

1) Teacher Model: In a self-training framework, the teacher model serves the role of generating pseudo-labels on unlabeled data to augment the limited training set. The teacher model in our decoupled GST framework is an LP module that

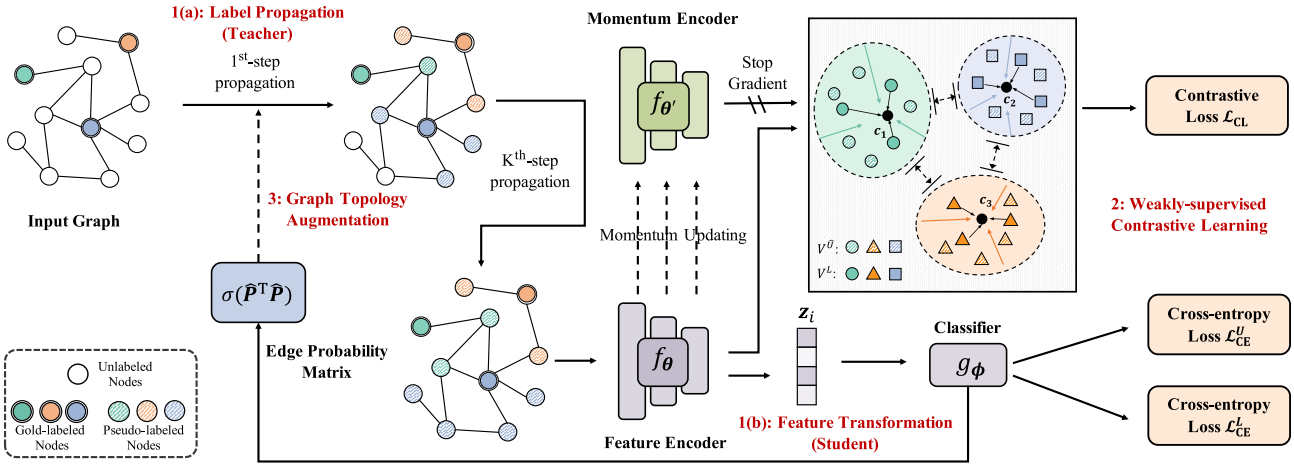

Fig. 1. Overview of the proposed framework. In each training iteration, AGST first performs high-order LP to generate pseudo-labels on unlabeled nodes, then conducts FT with the augmented training label set. In the meantime, a weakly supervised contrastive loss is used for optimizing the usage of pseudo-labels. Based on the computed edge probability matrix, the input graph structure will be augmented and fed to the next iteration. Figure best viewed in color.

enables long-range propagation of label signals for computing the pseudo-labels. This way the pseudo-labels preserve both local and global structure knowledge when further training the student model. Specifically, the objective of LP is to find a prediction matrix $\hat{Y} \in \mathbb{R}^{n\times c}$ that agrees with the label matrix $\mathbf{Y}$ while being smooth on the graph such that nearby vertices have similar soft labels

$$\hat{Y} = \arg\min_{\hat{Y}} \Big(\underbrace{\mathbf{Tr}(\hat{Y}^{\mathrm{T}}(\mathbf{I}-\mathbf{S})\hat{Y})}_{\text{smoothness constraint}} + \underbrace{\mu||\hat{Y}-\mathbf{Y}||_2^2}_{\text{fitting constraint}} \Big) \tag{1}$$

where μ is a positive parameter that balances the tradeoff between these two competing constraints. The smoothness term smooths each column of the prediction matrix along the graph structure, while the fitting term enforces the prediction matrix $\hat{Y}$ to agree with the label matrix $\mathbf{Y}$.

By solving the above-unconstrained optimization function, a closed-form solution can be computed as $\hat{Y} = (1-\alpha)(\mathbf{I}-\alpha\mathbf{S})^{-1}\mathbf{Y}$, where $\alpha = (1)/(\mu+1)$. As derived by Zhou et al. [55], the solution can be approximated via the following iteration:

$$\mathbf{Y}^{(t+1)} = \alpha\mathbf{S}\mathbf{Y}^{(t)} + (1-\alpha)\mathbf{Y}^{(0)} \tag{2}$$

where $\mathbf{Y}^{(0)} = \mathbf{Y}$, which converges to $\hat{Y}$ rapidly. Here $1-\alpha$ can be naturally connected with the teleport probability in Personalized PageRank [27]. With an appropriate α, the smoothed labels can avoid losing the focus on local neighborhood [11] even using infinitely many propagation steps.

2) Student Model: Since the LP-based teacher model supports long-range propagation without losing its focus on the local neighborhood, both local and global structure knowledge will be captured in the computed soft pseudo-labels. Next, we develop an FT module as the student model to distill the knowledge from the teacher model and meanwhile learn the feature knowledge by transforming the node features to class labels. The student model is composed of an encoder network $f_{\boldsymbol{\theta}}(\cdot)$ followed by a prediction network $g_{\boldsymbol{\phi}}(\cdot)$. For node v_i, the class prediction can be computed

$$\hat{\mathbf{p}}_i = g_{\boldsymbol{\phi}}(\mathbf{z}_i), \quad \mathbf{z}_i = f_{\boldsymbol{\theta}}(\mathbf{x}_i) \tag{3}$$

where the predicted label $\hat{\mathbf{p}}_i$ is computed based on the node features $\mathbf{x}_i$. Specifically, the encoder network $f_{\boldsymbol{\theta}}(\cdot)$ is built with a two-layer MLP, and the prediction network $g_{\boldsymbol{\phi}}(\cdot)$ is a feed-forward layer followed by Softmax, producing a vector of confidence scores.

To learn the student model, instead of using hard pseudo-labels as previous GST methods, here we consider the soft pseudo-labels generated by the LP-based teacher model as the ground truth and compute the standard cross-entropy loss on unlabeled nodes

$$\mathcal{L}_{\mathrm{CE}}^{U} = -\sum_{v_i\in\mathcal{V}^U}\sum_{c=1}^{C} \hat{y}_i^c \log \hat{p}_i^c. \tag{4}$$

In addition, we derive another cross-entropy loss for the nodes from the labeled training set

$$\mathcal{L}_{\mathrm{CE}}^{L} = -\sum_{v_i\in\mathcal{V}^L}\sum_{c=1}^{C} y_i^c \log \hat{p}_i^c. \tag{5}$$

By jointly optimizing the above losses, we can learn a simple yet effective student model for semi-supervised node classification with only a few labels.

3) Graph Topology Augmentation: Real-world graphs commonly come with a certain level of structure noise, which could be induced by either partial observation, graph preprocessing, or even adversarial attacks [16], [18], [56]. Since labeled nodes are extremely limited, the feature patterns of labeled nodes will be even harder to propagate to unlabeled nodes due to the imperfect graph structure. Considering that message-passing is a type of Laplacian Smoothing [27], representations of nodes belonging to different classes will become inseparable due to the existence of many unnecessary interclass edges. As such, we argue that it is helpful to eliminate potentially noisy edges and strengthen the connections between similar nodes for better preserving the graph structure knowledge and improving the effectiveness of message-passing.

To this end, after the student model converges in each self-training iteration, we in turn use it to refine the graph topology

by strengthening intraclass edges and reducing interclass connections. Specifically, given the predicted label matrix $\hat{\mathbf{P}}$, the edge probability matrix $\hat{\mathbf{A}}$ (symmetric) is computed by

$$\hat{\mathbf{A}} = \sigma(\hat{\mathbf{P}}^{\mathrm{T}}\hat{\mathbf{P}}), \quad \hat{\mathbf{P}} = g_{\boldsymbol{\phi}}(f_{\boldsymbol{\theta}}(\mathbf{X})) \tag{6}$$

where $\hat{\mathbf{A}}_{ij}$ denotes the probability that node v_i and v_j belong to the same class and σ is an element-wise sigmoid function.

Based on the Homophily principle [24] that assumes similar nodes are likely to be connected, thus we add/remove the edge e_{ij} in the original adjacency matrix $\mathbf{A}$ if the edge probability $\hat{\mathbf{A}}_{ij}$ is larger/less than a threshold. Specifically, we add top $\beta_a|\mathcal{E}|$ nonexist (intraclass) edges with highest edge probabilities and removes the $\beta_r|\mathcal{E}|$ existing (interclass) edges with lowest edge probabilities, where $\beta_a, \beta_r \in [0, 1]$.

4) Design Discussion: It is noteworthy that such a design has the following unique advantages: 1) unlike traditional GNNs, where neighborhood aggregation and FT are tightly coupled in each layer, we employ a decoupled approach by separating the transformation and propagation steps in message passing. Despite this innovation, it remains firmly rooted in the foundational principles of GNNs and adopts the decoupled GNN structure to harness its advantages. It not only enables long-range propagation of feature-label patterns by decoupling the transformation and propagation steps, but also improves the propagation process by using the learned student model to augment the input graph structure; 2) different from the standard self-training paradigm where teacher and student have the same architecture, our decoupled backbone uses an LP module as the teacher and an MLP as the student, which is parameter-less and evidently efficient; and 3) previous GST methods need to select unlabeled samples with high confidence as training targets. However, many of these selected predictions are incorrect due to the poor calibration of neural networks [57]. Our approach uses propagated soft pseudo-labels to circumvent the process of pseudo-label selection.

B. Augmenting Semantic Knowledge in GST

Despite the effectiveness of the above design, the generated pseudo-labels could introduce complex feature patterns and noisy training labels since the teacher model is trained with few labels, which exacerbates the difficulty of learning well-separated decision boundaries. Hence, how to enforce pseudo-labeled nodes to have aligned usage of gold-labeled ones is another important factor for improving the semantic knowledge of GST under the low-data regime.

To this end, we propose a *weakly supervised contrastive loss* that mitigates pseudo-label noise and enhances semantic structure learning. Specifically, a contrastive loss [43] encourages the similarity function to assign large values to the positive pairs and small values to the negative pairs. With similarity measured by dot product, a form of a contrastive loss function, called InfoNCE [42], [58] has been widely used in self-supervised learning

$$\mathcal{L}_{\text{InfoNCE}} = \sum_{i=1}^{n} -\log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}'_i/\tau)}{\sum_{j=0}^{r} \exp(\mathbf{z}_i \cdot \mathbf{z}'_j/\tau)} \tag{7}$$

where $\mathbf{z}'_i$ are positive embedding for $\mathbf{z}_i$, and $\mathbf{z}'_j$ includes one positive embedding and r negative embeddings for other instances. Here, τ is a temperature hyperparameter.

Different from the unsupervised contrastive loss [58] which only preserves the local smoothness around each instance, our goal is to improve the utility of pseudo-labeled nodes to learn better semantic structures of the input graph. To achieve this, we first compute the similarity distribution between pseudo-labeled samples and different class prototypes in the feature space

$$s_i^j = \frac{\exp(\mathbf{z}_i \cdot \mathbf{c}_j/\tau)}{\sum_{c=1}^{C} \exp(\mathbf{z}_i \cdot \mathbf{c}_c/\tau)}, \quad \mathbf{c}_c = \frac{1}{|\mathcal{V}_c^L|} \sum_{v_i \in \mathcal{V}_c^L} \mathbf{z}_i \tag{8}$$

where $\mathcal{V}_c^L$ denotes the labeled node set of class c and $\mathbf{c}_c$ is the corresponding class prototype computed as the average of the labeled examples in class c.

Due to the existence of incorrect pseudo-labels, there could be an inconsistency between the latent feature space and the pseudo-label space. Here, we apply the following rule to obtain a filtered set of pseudo-labeled nodes $\mathcal{V}^{\widetilde{U}}$:

$$s_i^{\hat{y}_i} > 1/C, \quad \hat{y}_i = \arg\max_j \hat{y}_i^j \tag{9}$$

where $\hat{y}_i$ is the hard pseudo-label of node v_i with the maximum confidence score. For each pseudo-labeled node v_i, we consider it trustworthy if its embedding similarity (to its pseudo-labeled prototype) $s_i^{\hat{y}_i}$ is higher than uniform probability, this way we largely reduce the risk of noisy training on hard pseudo-labels.

With the calibrated pseudo-labeled node set, next, we try to enhance the intraclass compactness and interclass separability of learned node representations. Specifically, our contrastive loss encourages each node to cluster around its corresponding class prototypes, which can be formulated as follows:

$$\mathcal{L}_{\text{CL}} = \sum_{v_i \in \mathcal{V}^{\widetilde{U}}} -\log \frac{\exp(\mathbf{z}_i \cdot \mathbf{c}_{\hat{y}_i}/\tau)}{\sum_{c=1}^{C} \exp(\mathbf{z}_i \cdot \mathbf{c}_c/\tau)} \tag{10}$$

where $\mathbf{c}_{\hat{y}_i}$ denotes the corresponding prototype of node v_i. For any pseudo-labeled node v_i, its embedding, $\mathbf{z}_i$, is treated as the anchor, the embedding of its corresponding class prototype $\mathbf{c}_{\hat{y}_i}$ forms the positive sample, and the embeddings of other class prototypes are naturally regarded as negative samples. This loss will be optimized to reduce the variance of pseudo-labeled nodes that share the same semantics while pushing away instances from different classes.

For the sake of stable training, we follow the idea of MoCo [42] and use the node representations learned from a *momentum encoder* parameterized by $f_{\boldsymbol{\theta}'}(\mathbf{x}_i)$ to compute the momentum prototype of each class. Note that the momentum encoder has the same architecture as the encoder network, and its parameters are the moving-average of the encoder's parameters. Formally

$$\boldsymbol{\theta}'_t = m \cdot \boldsymbol{\theta}'_{t-1} + (1 - m) \cdot \boldsymbol{\theta}_t \tag{11}$$

where m, $\boldsymbol{\theta}$, and $\boldsymbol{\theta}'$ are momentum, encoder parameters, and the momentum encoder parameters, respectively.

Algorithm 1 Learning Algorithm of AGST

Input: The input graph $G = (\mathcal{V}, \mathcal{E})$ with labeled node set $\mathcal{V}^L$ and unlabeled node set $\mathcal{V}^U$, self-training iterations I

Output: The well-trained student model

Initialize the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$
for $i = 1, 2, \ldots, I$ **do**
  ▷ *Label Propagation (Teacher)*
  Generate soft pseudo-labels on unlabeled nodes by Eq. (2);
  ▷ *Feature Transformation (Student)*
  while *not converge* **do**
    Compute the classification loss according to Eq. (4), Eq. (5);
    Compute the contrastive loss according to Eq. (10);
    Update the student model's parameters by optimizing the joint loss in Eq. (12);
  ▷ *Graph Topology Augmentation*
  Compute the edge probability matrix using Eq. (6);
  Augment the input graph by adding/removing edges
return The student model

C. Model Training

Given the above focal designs, we train the AGST framework in an iterative learning fashion. In each self-training iteration, the teacher model first generates pseudo-labels and we then optimize the student model until converges. Afterward, the graph structure will be refined by the topology augmentation function and fed to the next iteration.

To train the student model end-to-end, we jointly optimize the classification losses and the weakly-supervised contrastive loss. The full training objective is defined as follows:

$$\mathcal{L} = \mathcal{L}_{\mathrm{CE}}^{L} + \lambda_1 \mathcal{L}_{\mathrm{CE}}^{U} + \lambda_2 \mathcal{L}_{\mathrm{CL}} \tag{12}$$

where λ_1 and λ_2 are balancing parameters. According to our preliminary experiments, simply setting the parameters λ_1 and λ_2 as 1 and 0.1 can offer stable and strong performance in practice. By minimizing the training objective, the rich unlabeled data and the scarce, yet valuable labeled data work collaboratively to provide additional supervision signals for learning the discriminative prediction model. The detailed learning process of AGST is presented in Algorithm 1. Note that in each iteration, we refine the graph topology based on the original graph structure instead of the previously refined graph, since informative edges might be accidentally removed at the early stage of the training procedure. To reduce computational complexity, we only consider the edge between node v_i and v_j as candidates only when they have the same hard labels when adding edges. Furthermore, by refining the graph structure, it is essential to repeat training both the teacher and student models, which connects naturally to the iteration loops in conventional self-training.

TABLE I
SUMMARY STATISTICS OF THE EVALUATION DATASETS

Dataset	# Nodes	# Edges	# Features	# Classes
Cora	2,708	5,278	1,433	7
CiteSeer	3,327	4,552	3,703	6
PubMed	19,717	44,324	500	3
Coauthor-CS	18,333	81,894	6,805	15
Coauthor-Physics	34,493	247,962	8,415	5
Amazon-Photo	7,487	119,043	745	8

IV. EXPERIMENTS

In this section, we start by introducing the setup of our experiments. Then, we conduct experiments on benchmark datasets to show the effectiveness of the proposed framework.

A. Experimental Setup

1) Evaluation Datasets: We adopt six graph benchmark datasets to demonstrate the effectiveness of the proposed approach for semi-supervised node classification. Specifically, *Cora* [59], *CiteSeer* [59], and *PubMed* [60] are three most widely used citation networks. *Coauthor-CS* [61] and *Coauthor-Physics* [61] are two co-authorship graphs based on the Microsoft Academic Graph. *Amazon-Photo* [61] is an Amazon product co-purchase networks. The detailed statistics of the datasets are summarized in Table I.

To provide a robust and fair comparison between different models on each dataset, we evaluate two low-data settings with different data splitting protocols as follows.

1) *Balanced Training Setting:* Similar to the setting in [28] and [61], for each dataset, we sample a few (i.e., K-shot) labeled nodes per class as the training set, 30 nodes per class as the validation set, and the rest as the test set. We conduct 100 runs for random training/validation/test splits to ensure a fair comparison.
2) *Imbalanced Training Setting:* In this setting, we strictly follow the setup in [11] and [12] and randomly split the data into one small sample subset for training, and the test sample subset with 1000 samples. Following this line of work, we report the mean accuracy of ten runs without validation to make a fair comparison.

2) Compared Methods: In our experiments, we compare the proposed approach AGST with both classic and state-of-the-art methods on the semi-supervised node classification task. In addition to the traditional semi-supervised learning method LP, other baseline methods can be generally categorized into three classes: 1) *Vanilla GNNs* that only allows shallow message passing, including GCN [4], GAT [5], and SGC [22]; 2) *Deep GNNs* that can better propagate messages from limited labeled data, including GLP [62], IGCN [62], CGPN [34], and NAGphormer [25]; and 3) *Self-training GNNs* that adopt the teacher–student architecture to leverage pseudo-labels during training, including PTA [29], ST-GCNs [11] (and its variants), and M3S [12].

3) Implementation Details: We implement the proposed AGST in PyTorch with a 12 GB Titan Xp GPU. Specifically, we use a two-layer MLP with 64 hidden units for the FT module. The self-training iteration of AGST is set

TABLE II

TEST ACCURACY OF SEMI-SUPERVISED NODE CLASSIFICATION WITH FEW LABELS (IMBALANCED TRAINING SETTING): MEAN ACCURACY (%) WITH 95% CONFIDENCE INTERVAL. RESULTS OF BASELINE METHODS ARE BORROWED FROM M3S [12]

Dataset	Label Rate	LP	GCN	Co-train	Self-train	Union	InterSection	M3S	**AGST** (teacher)	**AGST** (student)
Cora	0.5%	57.6	50.6	53.9	56.8	55.3	50.6	61.5	60.2 ± 1.02	$\mathbf{70.2 \pm 0.93}$
	1.0%	61.0	58.4	57.0	60.4	60.0	60.4	67.2	63.6 ± 0.87	$\mathbf{75.8 \pm 0.80}$
	2.0%	63.5	70.0	69.7	71.7	71.7	70.0	75.6	68.0 ± 0.48	$\mathbf{78.3 \pm 0.51}$
	3.0%	64.3	75.7	74.8	76.8	77.0	74.6	77.8	70.2 ± 0.40	$\mathbf{80.1 \pm 0.36}$
CiteSeer	0.5%	37.7	44.8	42.0	51.4	48.5	51.3	56.1	46.8 ± 0.91	$\mathbf{63.3 \pm 0.88}$
	1.0%	41.6	54.7	50.0	57.1	52.6	61.1	62.1	54.2 ± 0.82	$\mathbf{71.4 \pm 0.80}$
	2.0%	41.9	61.2	58.3	64.1	61.8	63.0	66.4	59.5 ± 0.52	$\mathbf{72.0 \pm 0.43}$
	3.0%	44.4	67.0	64.7	67.8	66.4	69.5	70.3	61.6 ± 0.39	$\mathbf{72.8 \pm 0.31}$
PubMed	0.03%	58.3	51.1	55.5	56.3	57.2	55.0	59.5	59.3 ± 1.25	$\mathbf{69.6 \pm 1.03}$
	0.05%	61.3	58.0	61.6	63.6	64.3	58.2	64.4	63.3 ± 1.05	$\mathbf{73.5 \pm 0.98}$
	0.1%	63.8	67.5	67.8	70.0	70.0	67.0	70.6	65.7 ± 0.93	$\mathbf{78.9 \pm 0.89}$

to 3 for all the datasets. We set the propagation steps $K = 10$ by default. We optimize the model with the Adam optimizer and grid search for the edge addition/removal rate in $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8., 0.9, 1.0\}$. The optimal values are selected when the model achieves the best performance for the validation set. The early stopping criterion uses a patience of $p = 100$ and an (unreachably high) maximum of $n = 10\,000$ epochs. The patience is reset whenever the accuracy increases or the loss decreases on the validation set.

B. Main Results

In our experiments, we evaluate the proposed framework AGST and all the baseline methods on semi-supervised node classification tasks in low-data settings, which aims to predict the missing node labels with only a few labeled nodes.

1) Balanced Training Setting: We first compare the proposed framework AGST with baseline methods under the canonical few-shot semi-supervised setting. In this setting, each node class is provided with a few labeled samples. We run each model with three, five, and ten labeled nodes per class, referred to as three-shot, five-shot, and ten-shot configurations. The resulting average test accuracy under such a balanced training setting is reported in Table II. According to the reported results, we can make the following in-depth observations and analysis.

1) Overall, AGST consistently outperforms all the baseline methods across each dataset, as evidenced by paired t-tests with a significance level of $p < 0.05$. For example, AGST improves the best-performing baseline model (i.e., PTA) on Cora and obtains a 4.01% improvement in five-shot evaluation. This observation further proves that the design of AGST is effective for tackling the node classification problem when only a few labels per class are given.
2) Both deep GNNs and GST methods can achieve better performance over shallow GNNs such as GCN and GAT when training data is extremely scarce. While compared to the deep GNNs, existing GST methods cannot achieve better performance in most cases, which verifies our claim that their shallow backbones largely restrict the effective propagation of label signals. In contrast, our framework, AGST, adopts a decoupled backbone that inherits the advantages of both deep GNN models and GST methods, which is more data-efficient.
3) Though LP only relies on structure information, it can perform competitively with shallow GNNs on some datasets when training labels are extremely limited, such as Cora and PubMed. However, we also noticed that LP became the worst-performing method on datasets like CiteSeer and Coauthor CS. The main reason behind this is that noisy graph structure could easily lead to incorrect propagation of label signals, which verifies the rationality and necessity of refining the graph topology in GST.

2) Imbalanced Training Setting: Furthermore, we follow the imbalanced training setting used in [11] and [12] and conduct an additional series of experiments with varying label rates for each model. Specifically, we use label rates of 0.5%, 1%, 2%, and 3% for Cora and CiteSeer, and 0.03%, 0.05%, and 0.1% for PubMed. In this setup, the number of training nodes for each class is proportional to the total number of nodes belonging to a specific class within the dataset. Compared to the balanced training setting, this evaluation setting is more challenging since the training labels for each class could vary a lot (i.e., the ratio between the largest and the smallest classes in Cora is approximately 4:1). We report the average accuracy on three datasets in Table III. For a fair comparison, the results of baseline methods are borrowed from the previous work [12].

1) Similar to the balanced training setting, GCN that only uses limited receptive fields cannot achieve satisfactory results when labeled data is scarce and imbalanced. The integration of pseudo-labels into the learning process, as demonstrated by methods such as Co-train, Self-train, Union, and Intersection, proves effective in enhancing the performance of GCN when only limited labels are available.
2) However, the performance of those baselines based on pseudo-labeling varies a lot under different datasets. This highlights the practical challenge of effectively choosing informative pseudo-labels and mitigating the pseudo-label noise. Though M3S partially addresses this by using the clustering methods, it still largely falls behind our approach due to the inability to leverage large receptive fields and handle structure noise.

TABLE III
TEST ACCURACY OF SEMI-SUPERVISED NODE CLASSIFICATION WITH FEW LABELS (BALANCED TRAINING SETTING): MEAN ACCURACY (%) WITH 95% CONFIDENCE INTERVAL

Method	Cora			CiteSeer			PubMed		
	3-shot	5-shot	10-shot	3-shot	5-shot	10-shot	3-shot	5-shot	10-shot
LP	52.76 ± 0.92	58.72 ± 0.79	64.03 ± 0.65	34.87 ± 0.93	37.58 ± 0.81	41.74 ± 0.50	59.58 ± 0.98	62.32 ± 0.94	67.02 ± 0.75
GCN	56.31 ± 0.81	64.18 ± 0.66	72.87 ± 0.53	47.59 ± 0.90	54.27 ± 0.81	62.26 ± 0.57	59.24 ± 0.81	66.40 ± 0.85	72.37 ± 0.74
GAT	63.39 ± 0.98	69.93 ± 0.84	76.44 ± 0.35	51.62 ± 0.97	58.67 ± 0.81	65.13 ± 0.51	64.72 ± 0.91	68.32 ± 0.90	73.85 ± 0.60
SGC	55.94 ± 0.97	59.77 ± 0.97	67.76 ± 0.91	52.60 ± 0.92	58.94 ± 0.85	64.92 ± 0.54	58.74 ± 0.92	64.72 ± 0.91	69.02 ± 0.83
GLP	65.99 ± 0.94	72.31 ± 0.89	77.56 ± 0.43	50.46 ± 0.96	59.09 ± 0.88	66.06 ± 0.38	66.31 ± 0.95	72.59 ± 0.73	75.82 ± 0.58
IGCN	66.91 ± 0.91	72.78 ± 0.85	78.27 ± 0.31	50.99 ± 0.97	59.53 ± 0.89	66.51 ± 0.39	66.23 ± 0.97	71.96 ± 0.85	75.97 ± 0.50
CGPN	71.88 ± 2.52	71.83 ± 3.14	74.85 ± 1.54	$\mathbf{62.54 \pm 3.56}$	62.20 ± 1.63	63.76 ± 1.09	68.21 ± 3.89	71.21 ± 2.90	75.44 ± 2.53
NAGphormer	63.82 ± 0.87	70.28 ± 0.41	74.88 ± 0.97	48.50 ± 0.98	53.22 ± 0.72	59.26 ± 0.58	67.07 ± 0.53	68.13 ± 0.38	73.49 ± 0.23
PTA	69.21 ± 0.99	73.98 ± 0.73	78.69 ± 0.39	54.18 ± 0.94	61.13 ± 0.86	66.69 ± 0.48	67.69 ± 0.92	72.28 ± 0.82	76.47 ± 0.51
ST-GCNs	65.85 ± 0.94	71.16 ± 0.87	76.54 ± 0.49	49.85 ± 0.95	61.39 ± 0.91	68.58 ± 0.36	65.99 ± 0.93	70.26 ± 0.98	74.10 ± 0.63
M3S	64.01 ± 0.71	69.26 ± 0.75	77.20 ± 0.41	50.31 ± 0.88	59.72 ± 0.82	65.99 ± 0.41	66.01 ± 0.90	72.38 ± 0.85	75.31 ± 0.49
AGST	$\mathbf{73.20 \pm 0.79}$	$\mathbf{76.95 \pm 0.72}$	$\mathbf{80.89 \pm 0.42}$	60.31 ± 0.85	$\mathbf{66.58 \pm 0.80}$	$\mathbf{72.30 \pm 0.39}$	$\mathbf{71.16 \pm 0.93}$	$\mathbf{74.87 \pm 0.89}$	$\mathbf{77.84 \pm 0.41}$

Method	Coauthor-CS			Coauthor-Physics			Amazon-Photo		
	3-shot	5-shot	10-shot	3-shot	5-shot	10-shot	3-shot	5-shot	10-shot
LP	57.77 ± 0.77	62.09 ± 0.60	66.18 ± 0.36	73.46 ± 0.93	76.94 ± 0.61	80.55 ± 0.41	69.24 ± 0.92	73.43 ± 0.72	77.78 ± 0.61
GCN	77.17 ± 0.79	84.09 ± 0.59	89.01 ± 0.98	82.49 ± 0.88	87.50 ± 0.69	90.78 ± 0.38	69.54 ± 0.99	74.42 ± 0.97	80.30 ± 0.78
GAT	79.66 ± 0.75	85.11 ± 0.49	89.34 ± 0.19	86.07 ± 1.16	89.35 ± 0.48	91.64 ± 0.48	70.47 ± 1.19	77.89 ± 1.05	82.39 ± 1.11
SGC	84.93 ± 0.57	88.11 ± 0.35	90.13 ± 0.99	87.55 ± 0.64	87.68 ± 0.39	91.38 ± 0.31	75.05 ± 0.88	78.73 ± 0.69	84.14 ± 0.45
GLP	84.58 ± 0.61	87.36 ± 0.61	91.59 ± 0.15	89.34 ± 0.99	91.52 ± 0.32	93.02 ± 0.20	75.11 ± 1.19	81.99 ± 0.97	85.33 ± 0.38
IGCN	84.26 ± 0.47	86.45 ± 0.33	90.82 ± 0.13	89.82 ± 0.57	91.33 ± 0.29	92.78 ± 0.21	75.36 ± 0.98	82.10 ± 0.89	85.50 ± 0.32
CGPN	88.96 ± 3.37	89.14 ± 3.27	90.37 ± 2.14	90.06 ± 3.48	91.76 ± 2.33	92.56 ± 2.22	83.57 ± 3.24	84.74 ± 2.63	87.78 ± 2.44
NAGphormer	85.61 ± 0.38	89.08 ± 0.85	90.55 ± 0.49	86.26 ± 0.55	89.91 ± 0.82	92.54 ± 0.24	74.79 ± 0.35	83.58 ± 0.33	87.85 ± 0.19
PTA	86.56 ± 0.46	89.43 ± 0.31	90.72 ± 0.18	88.62 ± 0.60	90.36 ± 0.53	92.15 ± 0.32	77.43 ± 0.89	82.63 ± 0.76	85.51 ± 0.74
ST-GCNs	88.34 ± 0.46	89.68 ± 0.45	91.39 ± 0.14	87.61 ± 0.69	90.23 ± 0.39	91.75 ± 0.21	73.86 ± 1.53	81.93 ± 1.09	85.54 ± 0.67
M3S	84.11 ± 0.46	86.96 ± 0.41	91.08 ± 0.11	89.12 ± 0.55	91.27 ± 0.31	92.93 ± 0.25	74.96 ± 0.97	81.88 ± 0.93	85.42 ± 0.37
AGST	$\mathbf{90.29 \pm 0.33}$	$\mathbf{91.31 \pm 0.37}$	$\mathbf{92.76 \pm 0.12}$	$\mathbf{92.86 \pm 0.62}$	$\mathbf{93.04 \pm 0.47}$	$\mathbf{94.37 \pm 0.24}$	$\mathbf{85.08 \pm 0.89}$	$\mathbf{86.53 \pm 0.92}$	$\mathbf{89.27 \pm 0.62}$

3) Compared to the original LP, the teacher model in the proposed AGST framework achieves better performance, even though they both use the same LP algorithm. It demonstrates that the graph structure can be well refined by our graph topology augmentation function. Based on both labeled and unlabeled data, the student model further pushes forward the performance by performing FT and weakly supervised contrastive learning.

3) Standard (20-Shot) Training Setting: Next, we examine the performance of AGST under the standard semi-supervised node classification tasks. Specifically, we randomly sample 20 labeled nodes for each class (i.e., 20-shot) as the training set and test the performance of different methods. According to the average performance reported in Table IV, we can observe that: 1) the performance gain of GST methods over the vanilla GNNs decreases since the standard training setting has more labeled data; 2) data-efficient GNN such as CGPN cannot perform well under the standard semi-supervised learning setting; and 3) though AGST is mainly proposed for few-shot semi-supervised learning, it still achieves the best performance for the standard semi-supervised node classification task, illustrating the superiority of our approach.

TABLE IV
TEST ACCURACY ON STANDARD (20-SHOT) NODE CLASSIFICATION: MEAN ACCURACY (%) WITH 95% CONFIDENCE INTERVAL

Method	Cora	CiteSeer	PubMed	Coauthor-CS
LP	67.04 ± 0.41	45.29 ± 0.34	69.78 ± 0.54	72.24 ± 0.24
GCN	77.85 ± 0.33	65.95 ± 0.42	76.33 ± 0.47	90.92 ± 0.11
GAT	76.85 ± 0.34	65.12 ± 0.72	73.20 ± 0.49	90.39 ± 0.98
SGC	71.19 ± 0.29	69.20 ± 0.37	72.13 ± 0.66	91.03 ± 0.21
GLP	79.33 ± 0.27	68.94 ± 0.28	78.49 ± 0.39	82.53 ± 0.29
IGCN	80.11 ± 0.31	67.89 ± 0.29	78.64 ± 0.39	83.50 ± 0.23
CGPN	74.12 ± 1.54	67.34 ± 1.07	75.81 ± 1.26	89.92 ± 1.10
NAGphormer	78.76 ± 0.11	63.03 ± 0.12	76.65 ± 0.23	91.59 ± 0.19
PTA	81.54 ± 0.35	69.84 ± 0.25	78.66 ± 0.44	92.51 ± 0.15
ST-GCNs	79.75 ± 0.24	70.26 ± 0.23	78.12 ± 0.30	91.61 ± 0.11
M3S	78.11 ± 0.39	70.42 ± 0.29	77.98 ± 0.29	91.90 ± 0.18
AGST	$\mathbf{82.57 \pm 0.22}$	$\mathbf{71.52 \pm 0.11}$	$\mathbf{79.92 \pm 0.27}$	$\mathbf{93.27 \pm 0.13}$

C. Ablation Study

In this section, we further conduct ablation studies to demonstrate the contribution of each component in AGST and justify our architectural design choice. Here, *AGST-base* denotes the decoupled GST backbone of our framework. Meanwhile, we include other two variants by removing each of the other two key designs in the proposed framework. Specifically, *w/o contrast* represents the variant of AGST that excludes the weakly supervised contrastive loss, and *w/o augment* is the variant without the graph topology augmentation function. Compared to the complete framework AGST, *w/o contrast* loses part of the semantic knowledge and *w/o augment* loses part of the structural knowledge.

We report the accuracy results of each variant (balanced training) on two datasets Cora, CiteSeer in Fig. 2. It is apparent that the classification accuracy will decrease when any one of the focal components is removed, which reveals that both the weakly supervised contrastive loss and the graph topology augmentation function make essential contributions to boosting the model performance. Meanwhile, compared to the conventional GNNs in Table III, the backbone of AGST, that is, *AGST-base* can achieve better classification performance. Meanwhile, by comparing *w/o augment* with *AGST-base*, we can see that our contrastive loss brings further

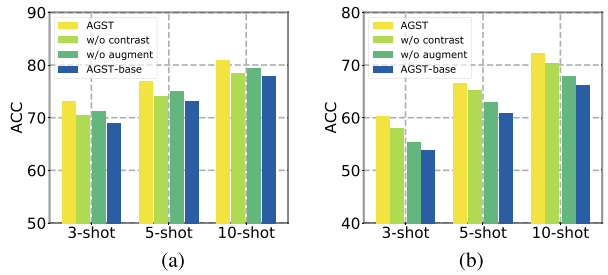

Fig. 2. Ablation results for different model variants. (a) Cora. (b) CiteSeer.

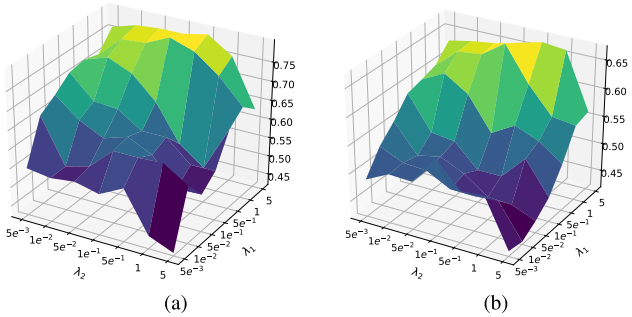

Fig. 3. Parameter analysis (λ_1, λ_2) results on (a) Cora and (b) Citeseer (5-shot).

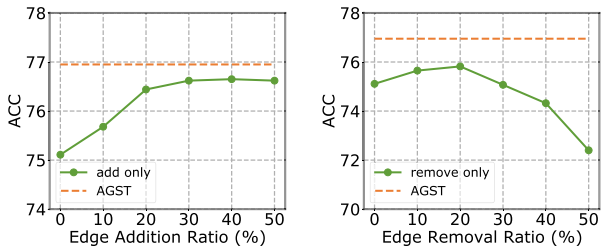

Fig. 4. Parameter analysis (β_a, β_r) results on Cora (5-shot).

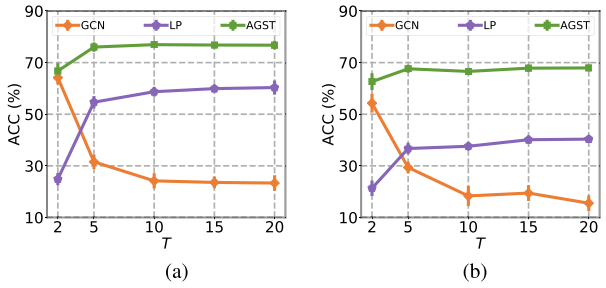

Fig. 5. Parameter analysis for propagation steps **T**. (a) Cora. (b) CiteSeer.

improvements. Notably, the importance of the topology augmentation function varies on different datasets, it usually has a larger contribution on datasets with noisy graph structures, such as CiteSeer.

D. Parameter Analysis

1) Loss Balancing Parameters: We explore the sensitivity of the model performance in terms of hyperparameters λ_1 and λ_2, which are in the final objective function of AGST. λ_1 controls the contribution of the cross-entropy loss on pseudo-labels (i.e., $\mathcal{L}_{\text{CE}}^{U}$) and λ_2 controls the contribution of weakly supervised contrastive loss (i.e., $\mathcal{L}_{\text{CL}}$). Specifically, we vary the values of λ_1 and λ_2 as $\{0.005, 0.01, 0.05, 0.1, 0.5, 1, 5\}$ on the Cora and Citeseer datasets and report the results of AGST in Fig. 3. As we can see from the figure, the performance of AGST goes up when we increase the value of λ_1 and reach the peak when $\lambda_1 = 1$. For λ_2, the best-performing value is 0.1. The performance will decrease if the value is either too large or too small. We have aligned observations with other datasets.

2) Topology Augmentation: We further examine the impact of two hyperparameters in our graph topology augmentation module, that is, the edge addition ratio β_a and the edge removal ratio β_r for refining the graph structure. Fig. 4 shows the performance change (five-shot) on the Cora dataset by varying the value of each parameter, with a constant interval of 0.1. We also add the performance of AGST as a reference. We observe that our graph topology augmentation function improves performance by either adding or removing edges within an appropriate range to mitigate the structure noise. For instance, the accuracy is first boosted by adding edges, then it reaches the peak until $\sim 40\%$ and becomes stable throughout the range. Similarly, edge removal improves performance until $\sim 20\%$, then the accuracy decreases quickly. One explanation is that a higher threshold may result in the accidental removal of possibly useful intraclass edges.

3) Propagation Steps: To demonstrate the effects of using different propagation steps, we compare our approach with two baselines (i.e., LP and GCN) under the five-shot setting with varying numbers of T. As shown in Fig. 5, we can clearly see that *GCN* encounters performance degradation if we largely increase the number of propagation steps. Though LP is a naive baseline, its performance increases by using larger propagation steps. Our framework AGST adopts a decoupled backbone where LP serves as the teacher model, thus it can address the oversquashing issue and leverage large receptive fields. AGST achieves stable performance when the propagation step $T > 5$.

V. Conclusion

Addressing the challenge of limited labeled nodes is crucial for achieving robust and sustainable graph semi-supervised learning. In this article, we investigate the problem of semi-supervised node classification when only a few labeled nodes are available and present a novel solution—an AGST framework. Specifically, this framework incorporates two unique graph data augmentation modules to capture both structural and semantic information for graph self-training to improve the model performance using a few labeled nodes. We perform experiments across diverse benchmark datasets in various low-data settings. The empirical results demonstrate the effectiveness of our proposed framework versus the baseline methods, particularly in scenarios involving classification with few labeled nodes. As part of future work, we aim to explore learnable graph augmentation modules that can adapt to nonhomophilous graphs [63], [64] and to develop a more powerful GST backbone model.

REFERENCES

[1] R. Zafarani, M. A. Abbasi, and H. Liu, *Social Media Mining: An Introduction*. Cambridge, U.K.: Cambridge Univ. Press, 2014.

[2] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 12, pp. 2724–2743, Dec. 2017.

[3] J. McAuley, R. Pandey, and J. Leskovec, "Inferring networks of substitutable and complementary products," in *Proc. KDD*, 2015, pp. 785–794.

[4] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. NeurIPS*, 2017.

[5] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proc. ICLR*, 2018.

[6] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. NeurIPS*, 2017.

[7] Z. Song, X. Yang, Z. Xu, and I. King, "Graph-based semi-supervised learning: A comprehensive review," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 11, pp. 8174–8194, Nov. 2023.

[8] K. Ding, J. Wang, J. Li, K. Shu, C. Liu, and H. Liu, "Graph prototypical networks for few-shot learning on attributed networks," in *Proc. CIKM*, 2020, pp. 295–304.

[9] K. Ding, C. Zhang, J. Tang, N. Chawla, and H. Liu, "Toward graph minimally-supervised learning," in *Proc. KDD*, 2022, pp. 4782–4783.

[10] D.-H. Lee et al., "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Proc. ICML Workshop*, 2013, pp. 1–6.

[11] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proc. AAAI*, 2018.

[12] K. Sun, Z. Lin, and Z. Zhu, "Multi-stage self-supervised learning for graph convolutional networks on graphs with few labeled nodes," in *Proc. AAAI*, 2020.

[13] K. Ding, Z. Xu, H. Tong, and H. Liu, "Data augmentation for deep graph learning: A survey," *ACM SIGKDD Explor. Newslett.*, vol. 24, no. 2, pp. 61–77, Nov. 2022.

[14] U. Alon and E. Yahav, "On the bottleneck of graph neural networks and its practical implications," in *Proc. ICLR*, 2021.

[15] J. Topping, F. Di Giovanni, B. P. Chamberlain, X. Dong, and M. M. Bronstein, "Understanding over-squashing and bottlenecks on graphs via curvature," in *Proc. ICLR*, 2022.

[16] Y. Zhu et al., "A survey on graph structure learning: Progress and opportunities," 2021, *arXiv:2103.03036*.

[17] H. Song, M. Kim, D. Park, Y. Shin, and J.-G. Lee, "Learning from noisy labels with deep neural networks: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 11, pp. 8135–8153, Nov. 2023.

[18] T. Zhao, Y. Liu, L. Neves, O. Woodford, M. Jiang, and N. Shah, "Data augmentation for graph neural networks," in *Proc. AAAI*, 2021.

[19] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," Tech. Rep., 1998.

[20] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2013, *arXiv:1312.6203*.

[21] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. NeurIPS*, 2016.

[22] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *Proc. ICML*, 2019.

[23] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *Proc. ICLR*, 2019.

[24] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annu. Rev. Sociol.*, vol. 27, no. 1, pp. 415–444, 2001.

[25] J. Chen, K. Gao, G. Li, and K. He, "NAGphormer: A tokenized graph transformer for node classification in large graphs," in *Proc. ICLR*, 2023.

[26] F. Monti, K. Otness, and M. M. Bronstein, "MOTIFNet: A MOTIF-based graph convolutional network for directed graphs," in *Proc. IEEE Data Sci. Workshop (DSW)*, Jun. 2018, pp. 225–228.

[27] J. Klicpera, A. Bojchevski, and S. Günnemann, "Predict then propagate: Graph neural networks meet personalized pagerank," in *Proc. ICLR*, 2019.

[28] M. Liu, H. Gao, and S. Ji, "Towards deeper graph neural networks," in *Proc. KDD*, 2020, pp. 338–348.

[29] H. Dong et al., "On the equivalence of decoupled graph convolution network and label propagation," in *Proc. Web Conf.*, 2021, pp. 3651–3662.

[30] T. Xie, B. Wang, and C.-C. J. Kuo, "GraphHop: An enhanced label propagation method for node classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 11, pp. 9287–9301, Nov. 2023.

[31] K. Ding, J. Wang, J. Caverlee, and H. Liu, "Meta propagation networks for graph few-shot semi-supervised learning," in *Proc. AAAI*, 2022.

[32] Y. Liu, K. Ding, J. Wang, V. Lee, H. Liu, and S. Pan, "Learning strong graph neural networks with weak information," in *Proc. KDD*, 2023.

[33] K. Xu, C. Li, Y. Tian, T. Sonobe, K. Kawarabayashi, and S. Jegelka, "Representation learning on graphs with jumping knowledge networks," in *Proc. ICML*, 2018.

[34] S. Wan, Y. Zhan, L. Liu, B. Yu, S. Pan, and C. Gong, "Contrastive graph Poisson networks: Semi-supervised learning with extremely limited labels," in *Proc. NeurIPS*, 2021.

[35] W. Lin, Z. Gao, and B. Li, "Shoestring: Graph-based semi-supervised classification with severely limited labeled data," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 4173–4181.

[36] E. Chien, J. Peng, P. Li, and O. Milenkovic, "Adaptive universal generalized pagerank graph neural network," in *Proc. ICLR*, 2021.

[37] C. Yang, J. Liu, and C. Shi, "Extract the knowledge of graph neural networks and go beyond it: An effective knowledge distillation framework," in *Proc. Web Conf.*, 2021, pp. 1227–1237.

[38] D. Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," in *Proc. ACL*, 1995.

[39] Z. Zhou, J. Shi, S. Zhang, Z. Huang, and Q. Li, "Effective stabilized self-training on few-labeled graph data," 2019, *arXiv:1910.02684*.

[40] E. Dai, C. Aggarwal, and S. Wang, "NRGNN: Learning a label noise resistant graph neural network on sparsely and noisily labeled graphs," in *Proc. KDD*, 2021, pp. 227–236.

[41] H. Liu, B. Hu, X. Wang, C. Shi, Z. Zhang, and J. Zhou, "Confidence may cheat: Self-training on graph neural networks under distribution shift," in *Proc. ACM Web Conf.*, 2022, pp. 1248–1258.

[42] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 9726–9735.

[43] M. Chen, Z. Wei, Z. Huang, B. Ding, and Y. Li, "Simple and deep graph convolutional networks," in *Proc. ICML*, 2020.

[44] J. Li, P. Zhou, C. Xiong, and S. Hoi, "Prototypical contrastive learning of unsupervised representations," in *Proc. ICLR*, 2021.

[45] P. Khosla et al., "Supervised contrastive learning," in *Proc. NeurIPS*, 2020.

[46] S. Li, X. Xia, S. Ge, and T. Liu, "Selective-supervised contrastive learning with noisy labels," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 316–325.

[47] J. Li, C. Xiong, and S. C. H. Hoi, "Learning from noisy data with robust representation learning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 9465–9474.

[48] K. Hassani and A. H. Khasahmadi, "Contrastive multi-view representation learning on graphs," in *Proc. ICML*, 2020.

[49] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen, "Graph contrastive learning with augmentations," in *Proc. NeurIPS*, 2020.

[50] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Graph contrastive learning with adaptive augmentation," in *Proc. Web Conf.*, 2021, pp. 2069–2080.

[51] J. Qiu et al., "GCC: Graph contrastive coding for graph neural network pre-training," in *Proc. SIGKDD*, 2020, pp. 1150–1160.

[52] S. Lin et al., "Prototypical graph contrastive learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 2, pp. 2747–2758, Feb. 2024.

[53] M. Gong, H. Zhou, A. K. Qin, W. Liu, and Z. Zhao, "Self-paced co-training of graph neural networks for semi-supervised node classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 11, pp. 9234–9247, Nov. 2023.

[54] K. Ding, Y. Wang, Y. Yang, and H. Liu, "Eliciting structural and semantic global knowledge in unsupervised graph contrastive learning," in *Proc. AAAI*, 2023.

[55] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Proc. NeurIPS*, 2004.

[56] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, and X. Sun, "Measuring and relieving the over-smoothing problem for graph neural networks from the topological view," in *Proc. AAAI*, 2020.

[57] M. N. Rizve, K. Duarte, Y. S. Rawat, and M. Shah, "In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning," in *Proc. ICLR*, 2021.

[58] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2018, *arXiv:1807.03748*.

[59] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, p. 93, 2008.

[60] G. Namata, B. London, L. Getoor, B. Huang, and U. Edu, "Query-driven active surveying for collective classification," in *Proc. 10th Int. Workshop Mining Learn. Graphs*, 2012, pp. 1–8.

[61] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann, "Pitfalls of graph neural network evaluation," 2018, *arXiv:1811.05868*.

[62] Q. Li, X.-M. Wu, H. Liu, X. Zhang, and Z. Guan, "Label efficient semi-supervised learning via graph filtering," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9574–9583.

[63] J. Zhu, Y. Yan, L. Zhao, M. Heimann, L. Akoglu, and D. Koutra, "Beyond homophily in graph neural networks: Current limitations and effective designs," in *Proc. NeurIPS*, 2020.

[64] J. Palowitch, A. Tsitsulin, B. Mayer, and B. Perozzi, "GraphWorld: Fake graphs bring real insights for GNNs," in *Proc. KDD*, 2022, pp. 3691–3701.