Theoretical Analyses of Hyperparameter Selection in Graph-Based Semi-Supervised Learning

Ally Yalei Du [1] **Eric Huang** [1] **Dravyansh Sharma** [1]

Abstract

Graph-based semi-supervised learning (SSL) is a powerful paradigm in machine learning for modeling and exploiting the underlying graph structure that captures the relationship between labeled and unlabeled data. A large number of classical as well as modern deep learning based algorithms have been proposed for this problem, often having tunable hyperparameters. Different values of hyperparameters define different node feature embedding in the underlying geometry and lead to different performances in terms of classification error. We initiate a formal study of hyperparameter tuning from parameterized algorithm families for this problem. We obtain novel $\Theta(\log n)$ pseudo-dimension upper bounds for hyperparameter selection in one of the classical label propagation based algorithm families, where n is the number of nodes, implying bounds on the amount of data needed for learning provably good parameters. We extend our study to hyperparameter selection in modern graph neural networks. We propose a novel tunable architecture that interpolates graph convolutional networks (GCN) and graph attention networks (GAT) in every layer, which we call GCAN. We then provide Rademacher complexity bounds for tuning the interpolation coefficient and study the influence of the interpolation coefficient on the node feature in the latent space. Finally, we empirically verify the effectiveness of GCAN on benchmark datasets.

1. Introduction

Semi-Supervised Learning (SSL) is a popular machine learning paradigm with significant theoretical interest (Zhou et al., 2003; Delalleau et al., 2005; Garg et al., 2020). In graph-based SSL, the graph nodes consist of labeled and unlabeled data points, and the graph edges denote feature similarity between the nodes. Classical algorithms focus on label-propagation based techniques, such as Zhou et al. (2003), Zhu et al. (2003), and many more. In recent years, graph neural networks (GNNs) have become increasingly popular in a wide range of application domains (Kipf & Welling, 2017; Veličković, Petar et al., 2018; Iscen et al., 2019). A large number of different architectures have been proposed, including graph convolution networks, graph attention networks, and so on (Dwivedi et al., 2023).

Hyperparameters, such as the weight for self-loop, play important roles in the performance of both classical methods and GNNs. Different values of the hyperparameter define different node feature embeddings in the underlying graph structure and therefore influence the performance of algorithms. Recent work (Donnat & Jeong, 2023) introduces and empirically studies parametric GCN-based families, but there are no existing theoretically principled studies on hyperparameter tuning. Another recent line of work (Balcan & Sharma, 2021; Sharma & Jones, 2023) considers the problem of learning hyperparameters through multiple problem instances drawn i.i.d. from a fixed distribution. However, their goal is to learn the best graph hyperparameter rather than the algorithm hyperparameter, and they do not consider deep learning structures like GNNs.

Moreover, focusing on the popular architecture Graph Attention Neural Network (GAT) and Graph Convolution Neural Network (GCN), we often observe that one architecture outperforms another on different problem instances (Dwivedi et al., 2023). However, there is no principled way of selecting the better of the two algorithms.

Contributions. In this paper, we take important initial steps to build the theoretical foundations of hyperparameter selection in graph-based semi-supervised learning, propose a novel GCAN architecture that outperforms both GAT and GCN, and empirically verifies our findings.

- We study hyperparameter tuning in a label propagation based SSL algorithm, the local and global consistency

[1]Carnegie Mellon University. Correspondence to: Dravyansh Sharma <dravyans@andrew.cmu.edu>.

Accepted as an extended abstract for the Geometry-grounded Representation Learning and Generative Modeling Workshop at the 41st International Conference on Machine Learning, ICML 2024, Vienna, Austria.

algorithms (Zhou et al., 2003). We prove new $\Theta(\log n)$ pseudo-dimension bound for the family, where n is the number of graph nodes. This result implies that it requires $m = O(\log n/\epsilon^2)$ problem instances to learn a ϵ-optimal hyperparameter.

- Next, we consider the modern graph neural networks (GNNs). We propose a novel architecture (GCAN) where a hyperparameter η interpolates two canonical GNN architectures: graph convolutional networks (GCNs) and graph attention networks (GATs). Theoretically, we bound the Rademacher complexity of tuning the interpolation coefficient and study the influence of η on the embedding through toy examples. Empirically, we test our novel GCAN interpolation method on 10 benchmark datasets to verify its empirical effectiveness.

Key Technical Insights. We address several challenges and introduce novel ideas in our proofs, setting our work apart from prior studies. To the best of our knowledge, this is the first work to study the pseudodimension guarantees of learning hyperparameters in label propagation-based algorithm families. While inspired by (Balcan & Sharma, 2021), our proof diverges significantly in its details. For the upper bound proof, we analyze changes in node classification predictions using a novel determinant evaluation and root counting argument, which may be of independent interest. For the lower bound proof, we construct instances based on connected components of graphs, resulting in instances with highly oscillating loss functions.

Additionally, we study GNN-based algorithm families and propose a novel architecture that interpolates between GCN and GAT. This architecture not only aims to select the better structure out of GCN and GAT but also has the potential to outperform both. We then provide Rademacher complexity bounds for our proposed GCAN architecture. Previous work (Garg et al., 2020) focuses on the Rademacher complexity of GNN-based algorithms for graph classification with a single problem instance and a fixed hyperparameter. In contrast, we consider node classification with multiple problem instances to learn the optimal hyperparameter. We bound our 0-1 loss using margin loss, reduce the Rademacher complexity of a sample set of graphs to that of the computation trees of single nodes, and bound the Rademacher complexity of each computation tree by studying the change in margin loss due to parameter variations and using a covering argument.

2. Preliminaries

Notations. Throughout this paper, $f(n) = O(g(n))$ denotes that there exists a constant $c > 0$ such that $|f(n)| \leq c|g(n)|$. $f(n) = \Omega(g(n))$ denotes that there exists a constant $c > 0$ such that $|f(n)| \geq c|g(n)|$. $f(n) = \Theta(g(n))$ denotes that $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$. The indicator function is indicated by $\mathbb{I}$, taking values in $\{0, 1\}$. In addition, we define the shorthand $[c] = \{1, 2, \ldots, c\}$. For a matrix W, we denote its Frobenius norm by $\|W\|_F$ and spectral norm by $\|W\|$. We also denote the Euclidean norm of a vector v by $\|v\|$.

Graph-based Semi-supervised Learning. We are given n data points, where some are labeled, denoted by $L \subseteq [n]$, and the rest unlabeled. We may have features associated with each data point, denoted by $z_i \in \mathbb{R}^d$ for $i \in [n]$. We can construct a graph G by placing (possibly weighed) edges $w(u, v)$ between pairs of data points u, v. The created graph G is denoted by $G = (V, E)$, where V represents the vertices and E represents the edges. We can calculate $W \in \mathbb{R}^{n\times n}$ as the adjacency matrix, i.e., $W_{ij} = w(i, j)$. We let $D \in \mathbb{R}^{n\times n}$ be the corresponding degree matrix.

We define input X as $X = (n, \{z_i\}_{i=1}^n, L, G)$, or $X = (n, L, G)$ (if no features). We denote the label matrix by $Y \in \{0, 1\}^{n\times c}$ where c is the number of classes. Throughout the paper, we assume $c = O_n(1)$, which matches most practical scenarios. Here, $Y_{ij} = 1$ if data point $i \in L$ has label j and $Y_{ij} = 0$ otherwise. The goal is to predict the labels of the unlabeled data.

An algorithm F in this setting may be considered as a function that takes in (X, Y) and outputs a predictor f that predicts a label in $[c]$ for each data. We denote $f(z_i)$ as our prediction on the i-th data.

Hyperparameter Selection. We consider several *parameterized families* of classification algorithms. Given a family of algorithms $\mathcal{F}_\rho$ parameterized by some hyperparameter ρ, and a set of m problem instances $\{(X^{(k)}, Y^{(k)})\}_{k=1}^m$ i.i.d. generated from the data distribution $\mathcal{D}$ of the input space $\mathcal{X}$ and the label space $\mathcal{Y}$, our goal is to select a $\hat{\rho}$ such that the corresponding prediction function $f_{\hat{\rho}}$ of algorithm $F_{\hat{\rho}}$ minimizes $\hat{\rho} = \operatorname{argmin}_\rho \frac{1}{mn}\sum_{k=1}^m \sum_{i=1}^n \ell_{0-1}(f_\rho(z_i^{(k)}), y_i^{(k)})$. Here, we denote $f_{\hat{\rho}}(z_i^{(k)})$ the predicted label of data point $z_i^{(k)}$ in the k-th problem instance. It is helpful to define the loss family induced by the algorithm family. Specifically, since each algorithm F_ρ maps a problem instance (X, Y) to a prediction function f_ρ, which further induces a loss $\frac{1}{n}\sum_{i=1}^n \ell_{0-1}(f_\rho(z_i), y_i)$, we can define H_ρ as the function that maps (X, Y) to the loss, and define $\mathcal{H}_\rho = \{H_{\rho'}\}_{\rho'}$ as the loss function family parameterized by ρ.

We will study the generalization ability of $f_{\hat{\rho}}$ given m problem instances, i.e., $\mathbb{E}_{(X,Y)\sim\mathcal{D}}\left[\frac{1}{n}\sum_{i=1}^n \ell_{0-1}\left(f_{\hat{\rho}}(z_i), y_i\right)\right] - \min_\rho \mathbb{E}_{(X,Y)\sim\mathcal{D}}\left[\frac{1}{n}\sum_{i=1}^n \ell_{0-1}\left(f_\rho(z_i), y_i\right)\right]$.

Note that our problem setting differs from prior theoretical works on graph-based semi-supervised learning. The classical setting considers a single algorithm (which may

come from a family of algorithms) and learning the model parameter from a single problem instance. We are considering *families of algorithms*, each parameterized by a single hyperparameter, and our goal is to learn the best hyperparameter from a set of *multiple problem instances* drawn i.i.d. from some underlying data distribution.

Complexity Measures and Generalization Bounds. We study the generalization ability of several representative families of algorithms. We aim to address the question of how many problem instances are required to learn a hyperparameter ρ such that a learning algorithm can perform near-optimally for instances drawn from a fixed problem distribution. We consider two learning-theoretic complexity measures for characterizing the learnability of algorithm families: the pseudo-dimension and the Rademacher complexity. Canonical theory gives generalization guarantees based on these measures (Appendix A).

Definition 2.1 (Pseudo-dimension (Pollard, 2012))**.** Let $\mathcal{H}$ be a set of real-valued functions from input space $\mathcal{X}$. We say that $C = (X^{(1)}, ..., X^{(m)}) \in \mathcal{X}^m$ is **pseudo-shattered** by $\mathcal{H}$ if there exists a vector $r = (r_1, ..., r_m) \in \mathbb{R}^m$ (called "witness") such that for all $b = (b_1, ..., b_m) \in \{\pm 1\}^m$ there exists $H_b \in \mathcal{H}$ such that $sign(H_b(X^{(k)}) - r_k) = b_k$. **Pseudo-dimension** of $\mathcal{H}$, denoted $\textsc{Pdim}(\mathcal{H})$, is the cardinality of the largest set pseudo-shattered by $\mathcal{H}$.

In Section 3, we will obtain *optimal* pseudo-dimension bounds for three canonical label-propagation algorithm families. Another classical complexity measure is the Rademacher complexity:

Definition 2.2 (Rademacher Complexity (Bartlett & Mendelson, 2002))**.** Given a space $\mathcal{X}$ and a distribution $\mathcal{D}$, let $S = \{X^{(1)}, \ldots, X^{(m)}\}$ be a set of examples drawn i.i.d. from $\mathcal{D}$. Let $\mathcal{H}$ be the class of functions $H : \mathcal{X} \to \mathbb{R}$. The **(empirical) Rademacher complexity** of $\mathcal{H}$ is

$$\hat{R}_m(\mathcal{H}) = \mathbb{E}_\sigma \left[\sup \left(\frac{1}{m} \sum_{k=1}^{m} \sigma_k H(X^{(k)}) \right) \right],$$

where each σ_k is i.i.d. sampled from $\{-1, 1\}$.

To apply Rademacher complexity theory, we restrict to binary classification $c = 2$ and change the label space to $Y \in \{-1, 1\}^n$. For a predictor f, we also overload notation and let $f(z_i) \in [0, 1]$ be the probability of a single node z_i being classified as 1. In Section 4, we bound the Rademacher complexity of our proposed GCAN structure.

3. Label Propagation based Families and Generalization Guarantees

In this section, we consider one popular label propagation based algorithm family. In Appendix C, we also derive similar results for the smoothing-based algorithm family (Delalleau et al., 2005) and normalized adjacency matrix-based algorithm family. Label propagation families will output a soft-label score $F^* \in \mathbb{R}^{n \times c}$ where the (i, j)-th entry represents the score of class j for the i-th sample. The prediction for the i-th sample is just $\operatorname{argmax}_{j \in [c]} F^*_{ij}$.

3.1. The Algorithm Family

We consider the parametric family which we call the Local and Global Consistency Algorithm Family (Zhou et al., 2003), parameterized by $\alpha \in (0, 1)$. The optimal scoring matrix F^* is defined as

$$F^*_\alpha = (1 - \alpha)(I - \alpha S)^{-1} Y, \text{ where } S = D^{-1/2} W D^{-1/2}.$$

Here, S is the symmetrically normalized adjacency matrix. This F^*_α corresponds to minimizing the following objective function $\mathcal{Q}(F) = \frac{1}{2}\Big(\sum_{i,j=1}^{n} W_{ij} \Big\| \frac{1}{\sqrt{d_i}} F_i - \frac{1}{\sqrt{d_j}} F_j \Big\|^2 + \frac{1-\alpha}{\alpha} \sum_{i=1}^{n} \Big\| F_i - Y_i \Big\|^2 \Big)$, where Y is the $n \times c$ matrix whose rows of unlabeled nodes are all zeros. The first term of $Q(F)$ measures the local consistency, i.e., the prediction between nearby points should be similar. The second term measures the global consistency, i.e., large-scale patterns (manifolds) over the data. Therefore, the hyperparameter $\alpha \in (0, 1)$ induces a trade-off between the local and the global consistency. We define this family of algorithms as $\mathcal{F}_\alpha$, and the 0-1 losses as $\mathcal{H}_\alpha$.

3.2. Pseudo-dimension Guarantees

We study the generalization behavior of this algorithm family through pseudo-dimension. The following theorem indicates that it has pseudo-dimension $\Theta(\log n)$, where n is the number of data in each problem instance. This result indicates that it requires $m = O\left(\log n/\epsilon^2\right)$ problem instances to learn a ϵ-optimal hyperparameter for the local and global consistency family. The full proof is in Appendix C.

Theorem 3.1. *The pseudo-dimensions of the Local and Global Consistency algorithm family satisfy Pdim$(\mathcal{H}_\alpha) = \Theta(\log n)$, where n is the total number of labeled and unlabeled data points.*

4. GCAN Interpolation

In Section 4.1, we introduce a novel architecture, which we call GCAN, that interpolates two popular GNNS: the graph convolutional neural networks (GCN) and graph attention neural networks (GAT). We then obtain a Rademacher complexity bound for the GCAN algorithm family. A brief introduction to GAT and GCN is given in Appendix D.

In Section 4.2, we study how the interpolating hyperparameter influences the embedding of node features in the underlying graph structure.

In Section 4.3, we empirically study the effectiveness of GCAN architecture.

4.1. GCAN and its Generalization Guarantees

In practice, GCN and GAT outperform each other in different problem instances. To effectively choose the better algorithm, we introduce a family of algorithms that *interpolates* GCN and GAT, parameterized by η.

Formally, we introduce a hyperparameter $\eta \in [0,1]$ and design a novel update rule such that $\eta = 0$ corresponds to GCN and $\eta = 1$ corresponds to GAT:

$$h_i^\ell = \sigma\left(\sum_{j\in\mathcal{N}_i}\left(\eta\cdot e_{ij}^\ell + (1-\eta)\cdot\frac{1}{\sqrt{d_i d_j}}\right)U^\ell h_j^\ell\right)$$

where

$$e_{ij}^\ell = \frac{\exp(\hat{e}_{ij}^\ell)}{\sum_{j'\in\mathcal{N}_i}\exp(\hat{e}_{ij'}^\ell)},\ \ \hat{e}_{ij}^\ell = \sigma(V^\ell[U^\ell h_i^\ell, U^\ell h_j^\ell]).$$

Here e_{ij}^ℓ is the attention score of node j for node i. V^ℓ and U^ℓ are learnable parameters. $\sigma(\cdot)$ is the activation function, which we assume to be 1-Lipschitz. We denote the concatenation of $U^\ell h_i^\ell$ and $U^\ell h_j^\ell$ by $[U^\ell h_i^\ell, U^\ell h_j^\ell]$. At the first level, we initialize $h_i^0 = z_i$. We study the generalization ability of tuning η through Rademacher complexity.

Theorem 4.1. *Assume the parameter U^ℓ is shared over all layers, i.e. $U^\ell = U$ for all $\ell \in [L]$ (the assumption used in (Garg et al., 2020)), and the parameters are bounded: $\|U\|_F \le C_U$, $\|V^\ell\|_2 \le C_V$, $\|z_i\| \le C_z$, and $d_i \in [C_{dl}, C_{dh}]$. Denoting the branching factor by $r = \max_{i\in[n]}|\sum_{j\in[n]}\mathbb{I}[w_{ij} \neq 0]|$, we have that the Rademacher complexity of $\mathcal{H}_\eta^\gamma$ is bounded:*

$$\hat{R}_m(\mathcal{H}_\eta^\gamma) = O\left(\frac{d\sqrt{L\log\frac{rC_U}{C_{dl}+C_U} + \log\frac{mdC_z}{\gamma}}}{\sqrt{m}}\right).$$

The proof is inspired by (Garg et al., 2020), and involves reducing the problem to bounding the Rademacher complexity of tree networks. Then, we use the peeling technique and covering number argument to finish the proof. See Appendix E.1 for details.

4.2. Inherent Embedding Geometry

In this subsection, we study the influence of hyperparameter η on the node feature embedding of GCAN by two toy examples inspired by (Donnat & Jeong, 2023).

Toy Example 1: Structurally different neighborhood, similar feature Consider two nodes u, v, we assume there exists a feature vector $\bar{h}$ such that, at level ℓ,

$$\forall j \in \mathcal{N}(v)\cup\mathcal{N}(u)\cup\{u\}\cup\{v\},\ \ h_j^\ell = \bar{h} + \epsilon_j,$$

where ϵ_j is a vector where each index has a value close to 0 and we assume $\|\epsilon_j\| \le \|\epsilon\|$. We have the following inequality:

$$\|h_u^{\ell+1} - h_v^{\ell+1}\| \le C_U\|\epsilon\| + (1-\eta)C_U\|\bar{h}\|\left\|\sum_{j\in\mathcal{N}_u}\frac{1}{\sqrt{d_u d_j}} - \sum_{j\in\mathcal{N}_v}\frac{1}{\sqrt{d_v d_j}}\right\|$$

In this toy example, the feature embedding of the two nodes in the next layer is almost identical if we let η closer to 1. If we let η closer to 0, the feature embedding of the two nodes would reflect the underlying graph structure captured by the degree of the neighboring nodes.

Toy Example 2: Structurally equivalent neighborhood, different feature Consider two nodes u, v, we assume there exists a bijection $\pi : \mathcal{N}(v) \to \mathcal{N}(u)$ such that

$$\forall j \in \mathcal{N}(v),\ \ d_j = d_{\pi(j)},\ \ h_j^\ell - h_{\pi(j)}^\ell = \epsilon_j,$$

where we assume there exists ϵ such that $\|\epsilon_j\| \le \|\epsilon\|$ for all $j \in \mathcal{N}(v)$ We investigate the difference in the node embeddings of u and v in the next layer. We have the following inequality:

$$\|h_u^{\ell+1} - h_v^{\ell+1}\| \le (1-\eta)C_U\|\epsilon\|/C_{dl} + \eta\cdot\left\|\sum_{j\in\mathcal{N}_u}(e_{uj}^\ell U^\ell h_j^\ell - e_{v\pi(j)}^\ell U^\ell h_{\pi(j)}^\ell)\right\|$$

In this toy example, the feature of the two nodes in the next layer is almost identical if we let η closer to 0. When η is closer to 1, the difference in the feature embedding of the two nodes is defined by the difference in the feature value of the neighboring nodes.

4.3. Experiments

In this section, we empirically test our proposed GCAN interpolation methods on ten standard benchmark datasets. Our goal is to see whether tuning η gives better results than both GCN and GAT. The setup details of our experiment is described in Appendix G.

In Table 1 (also Figure 1 in the appendix), we show the mean accuracy across 30 runs of each η value and the 90% confidence interval associated with each experiment. The optimal η values are distinct for different datasets, and the best model is usually interpolated between GCN and GAT, showing that we can achieve an improvement on both baselines. The loss also does not change monotonically as η increases for many datasets. This suggests that one should learn the best η parameter for each specific dataset. Moreover, since our family of algorithm includes both GAT and GCN, the optimal η chosen by GCAN architecture is

Dataset	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	Rel. gain GCN	Rel. gain GAT
CIFAR10	0.7888 ± 0.0010	0.7908 ± 0.0008	0.7908 ± 0.0015	0.7907 ± 0.0012	0.7943 ± 0.0022	0.7918 ± 0.0018	0.7975 ± 0.0017	0.7971 ± 0.0023	0.7921 ± 0.0023	0.7986 ± 0.0028	$\mathbf{0.7984 \pm 0.0023}$	+4.55%	0%
WikiCS	0.9525 ± 0.0007	0.9516 ± 0.0006	0.9532 ± 0.0011	0.9545 ± 0.0008	0.9551 ± 0.0015	0.9545 ± 0.0012	0.9539 ± 0.0012	$\mathbf{0.9553 \pm 0.0012}$	0.9530 ± 0.0007	0.9536 ± 0.0009	0.9539 ± 0.0009	+5.89%	3.04%
Cora	0.6132 ± 0.0218	0.8703 ± 0.0251	0.8879 ± 0.0206	0.8396 ± 0.0307	0.8022 ± 0.0385	0.8615 ± 0.0402	$\mathbf{0.9011 \pm 0.0421}$	0.8088 ± 0.0362	0.8505 ± 0.0240	0.8549 ± 0.0389	0.8725 ± 0.0334	+74.43%	+22.43%
Citeseer	$\mathbf{0.7632 \pm 0.0052}$	0.6944 ± 0.0454	0.7602 ± 0.0566	0.7500 ± 0.0461	0.7339 ± 0.0520	0.7427 ± 0.0462	0.7588 ± 0.0504	0.7193 ± 0.0567	0.7661 ± 0.0482	0.7266 ± 0.0412	0.7471 ± 0.0444	0%	+6.80%
PubMed	0.9350 ± 0.0009	0.9306 ± 0.0006	0.9356 ± 0.0009	0.9281 ± 0.0007	$\mathbf{0.9356 \pm 0.0007}$	0.9319 ± 0.0009	0.9313 ± 0.0007	0.9288 ± 0.0009	0.9313 ± 0.0006	0.9338 ± 0.0010	0.9356 ± 0.0009	+0.92%	0%
CoauthorCS	0.9733 ± 0.0007	0.9733 ± 0.0008	$\mathbf{0.9765 \pm 0.0005}$	0.9744 ± 0.0005	0.9733 ± 0.0009	0.9690 ± 0.0007	0.9712 ± 0.0009	0.9722 ± 0.0005	0.9722 ± 0.0011	0.9722 ± 0.0007	0.9744 ± 0.0007	+11.99%	+8.20%
AmazonPhotos	0.9605 ± 0.0022	0.9617 ± 0.0007	0.9629 ± 0.0015	0.9599 ± 0.0013	0.9641 ± 0.0017	0.9574 ± 0.0018	0.9641 ± 0.0019	0.9592 ± 0.0133	$\mathbf{0.9653 \pm 0.0027}$	0.9635 ± 0.0031	0.9562 ± 0.0019	+12.15%	+20.78%
Actor	0.5982 ± 0.0016	0.5919 ± 0.0022	$\mathbf{0.6005 \pm 0.0039}$	0.5959 ± 0.0039	0.5965 ± 0.0038	0.5970 ± 0.0027	0.5976 ± 0.0037	0.5993 ± 0.0043	0.5930 ± 0.0041	0.5970 ± 0.0037	0.5953 ± 0.0031	+0.57%	+1.28%
Cornell	0.7341 ± 0.0097	0.7364 ± 0.0165	0.7364 ± 0.0073	0.7205 ± 0.0154	0.7523 ± 0.0109	0.7795 ± 0.0120	0.7568 ± 0.0188	0.7500 ± 0.0140	0.7477 ± 0.0138	0.7909 ± 0.0136	$\mathbf{0.8000 \pm 0.0423}$	+24.78%	0%
Wisconsin	0.8688 ± 0.0077	$\mathbf{0.8922 \pm 0.0035}$	0.8688 ± 0.0080	0.8906 ± 0.0049	0.8797 ± 0.0044	0.8578 ± 0.0120	0.8875 ± 0.0037	0.8781 ± 0.0082	0.8563 ± 0.0128	0.8750 ± 0.0121	0.8719 ± 0.0076	+17.83%	+15.85%

Table 1. Results on the proposed GCAN interpolation. Each column corresponds to one η value. Each row corresponds to one dataset. Each entry shows the accuracy and the interval. We find in most cases, the optimal η is neither 0 (pure GCN) nor 1 (pure GAT). In the rightmost two columns, we also calculate the relative reduction in avergae loss of GCAN over the baseline algorithms GCN ($\eta = 0$) and GAT ($\eta = 1$). The relative improvement in the average loss of GCAN over baselines GAT and GCN (last two columns) is calculated as $(\text{Loss}_{\text{Baseline}} - \text{Loss}_{\text{GCAN}})/\text{Loss}_{\text{Baseline}}$

guaranteed to be no worse than both GAT and GCN. We emphasize this idea in the rightmost two columns of Table 1, where we show the relative reduction in the average loss of GCAN over baselines GAT and GCN, calculated as $(\text{Loss}_{\text{Baseline}} - \text{Loss}_{\text{GCAN}})/\text{Loss}_{\text{Baseline}}$.

5. Conclusion

We study hyperparameter selection in graph-based semi-supervised learning. We do this by leveraging access to multiple instances of data from a given domain. We provide formal guarantees on the number of data samples needed to learn the best algorithm for classical parameterized families as well as a novel family (GCAN) that interpolates convolution (GCN) and attention (GAT) in graph neural networks. We also study how the hyperparameter influences node features in the latent space of GCAN, and empirically test the effectiveness of GCAN through experiments.

References

Anthony, M. and Bartlett, P. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, USA, 1st edition, 2009.

Balcan, M.-F. and Sharma, D. Data driven semi-supervised learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Bartlett, P. L. and Mendelson, S. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

Bartlett, P. L., Foster, D. J., and Telgarsky, M. J. Spectrally-normalized margin bounds for neural networks. *Advances in Neural Information Processing Systems*, 30, 2017.

Delalleau, O., Bengio, Y., and Le Roux, N. Efficient non-parametric function induction in semi-supervised learning. In *International Workshop on Artificial Intelligence and Statistics*, pp. 96–103. PMLR, 2005.

Donnat, C. and Jeong, S. W. Studying the effect of GNN spatial convolutions on the embedding space's geometry. In Evans, R. J. and Shpitser, I. (eds.), *Uncertainty in Artificial Intelligence (UAI)*, volume 216 of *Proceedings of Machine Learning Research*, pp. 539–548. PMLR, 31 Jul–04 Aug 2023.

Dwivedi, V. P., Joshi, C. K., Luu, A. T., Laurent, T., Bengio, Y., and Bresson, X. Benchmarking graph neural networks. *Journal of Machine Learning Research*, 24 (43):1–48, 2023.

Garg, V., Jegelka, S., and Jaakkola, T. Generalization and representational limits of graph neural networks. In III,

H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3419–3430. PMLR, 13–18 Jul 2020.

Iscen, A., Tolias, G., Avrithis, Y., and Chum, O. Label propagation for deep semi-supervised learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5070–5079, 2019.

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

Mohri, M., Rostamizadeh, A., and Talwalkar, A. *Foundations of Machine Learning*. MIT Press, 2012.

Pollard, D. *Convergence of stochastic processes*. Springer Science & Business Media, 2012.

Sharma, D. and Jones, M. Efficiently learning the graph for semi-supervised learning. *Uncertainty in Artificial Intelligence (UAI)*, 2023.

Veličković, Petar, Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. Graph attention networks. *International Conference on Learning Representations (ICLR)*, 2018.

Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., and Weinberger, K. Simplifying graph convolutional networks. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 6861–6871. PMLR, 09–15 Jun 2019.

Zhou, D., Bousquet, O., Lal, T., Weston, J., and Schölkopf, B. Learning with local and global consistency. *Advances in Neural Information Processing Systems*, 16, 2003.

Zhu, X., Ghahramani, Z., and Lafferty, J. D. Semi-supervised learning using gaussian fields and harmonic functions. In *International conference on Machine learning (ICML)*, pp. 912–919, 2003.

A. Classical learning-theoretic results on generalization

We first state a classical generalization bound based on the pseudo-dimension.

Theorem A.1. *(Anthony & Bartlett, 2009) Suppose $\mathcal{H}$ is a class of real-valued functions with range in $[0,1]$ and finite Pdim($\mathcal{H}$). Then for any $\epsilon > 0$ and $\delta \in (0,1)$, for any distribution $\mathcal{D}$ and for any set $S = \{X^{(1)}, \ldots, X^{(m)}\}$ of $m = O\left(\frac{Pdim(\mathcal{H})}{\epsilon^2} + \log(\frac{1}{\delta})\right)$ samples from $\mathcal{D}$, with probability at least $1-\delta$, we have*

$$\left| \frac{1}{m} \sum_{k=1}^{m} H(X^{(k)}) - \mathbb{E}_{X \sim \mathcal{D}}[H(X)] \right| \leq \epsilon, \text{ for all } H \in \mathcal{H}.$$

Theorem A.2. *(Mohri et al., 2012) Suppose $\mathcal{H}$ is a class of real-valued functions with range in $[0,1]$. Then for any $\delta \in (0,1)$, any distribution $\mathcal{D}$, and any set $S = \{X^{(k)}\}_{k=1}^{m}$ of m samples from $\mathcal{D}$, with probability at least $1-\delta$, we have*

$$\left| \frac{1}{m} \sum_{k=1}^{m} H(X^{(k)}) - \mathbb{E}_{X \sim D}[H(X)] \right| = O\left(\hat{R}_m(\mathcal{H}) + \sqrt{\frac{1}{m} \log \frac{1}{\delta}} \right), \quad \text{for all} \quad H \in \mathcal{H}.$$

B. Additional Label Propagation Based Families

Smoothing-Based Algorithm Family ($\mathcal{F}_\lambda$) This second class of algorithm is parameterized by $\lambda \in (0, +\infty)$ (Delalleau et al., 2005). Let $\Delta \in \{0,1\}^{n \times n}$ be a diagonal matrix where elements are 1 only if the index is in the labeled set. The scoring matrix F_λ^* is

$$F_\lambda^* = (S + \lambda I_n \Delta_{i \in L})^{-1} \lambda Y, \quad \text{where} S = D - W.$$

The idea of $\mathcal{F}_\lambda$ is very similar to $\mathcal{F}_\alpha$. λ balances the relative importance of the known labels and the structure of the unlabeled points . Note that by Lemma C.4, $(S + \lambda I_n \Delta_{i \in L})$ is invertible as long as every connected component in G has at least one labeled node. This condition is reasonable because G is the only information we can use to relate the labeled and unlabeled nodes, and if there is a connected component with no label, then we have no information to label nodes in this component. Therefore, we will assume $(S + \lambda I_n \Delta_{i \in L})$ to be invertible in our analysis. Here we denote the set of 0-1 loss functions corresponding to $\mathcal{F}_\lambda$ as $\mathcal{H}_\lambda$.

Normalized Adjacency Matrix Based Family ($\mathcal{F}_\delta$) Here we consider a new algorithm family which we name *Normalized Adjacency Matrix Based Family*. This class of algorithm is parameterized by $\delta \in [0,1]$. The scoring matrix F_δ^* is

$$F_\delta^* = (I - c \cdot S)^{-1} Y, \quad \text{where} \quad S = D^{-\delta} W D^{\delta - 1}$$

is the normalized adjacency matrix.

This family of algorithms is motivated by $\mathcal{F}_\alpha$ and the family of spectral operators defined in (Donnat & Jeong, 2023). We may notice that the F_δ^* defined here is very similar to the F_α^* in $\mathcal{F}_\alpha$ when α is set to a constant c, whose default value considered in (Zhou et al., 2003) is 0.99. Here, instead of focusing on the trade-off between local and global consistency, we study the spatial convolutions S. With $\delta = 1$, we have the row-normalized adjacency matrix $S = D^{-1}W$. With $\delta = 0$, we have the column-normalized adjacency matrix $S = WD^{-1}$. Finally, with $\delta = 1/2$, we have the symmetrically normalized adjacency matrix that we used in $\mathcal{F}_\alpha$ and many other default implementations (Donnat & Jeong, 2023; Wu et al., 2019). We denote the set of 0-1 loss functions corresponding to $\mathcal{F}_\delta$ as $\mathcal{H}_\delta$.

B.1. Pseudodimension Guarantees

The pseudodimension of the above-mentioned two algorithm families are the same as the local and global consistency family mentioned in Section 3.

Theorem B.1. *The pseudo-dimension of the Smoothing-Based Algorithmic Family, $\mathcal{F}_\lambda$, is Pdim($\mathcal{H}_\lambda$) = $\Theta(\log n)$, where n is the total number of labeled and unlabeled data points.*

Theorem B.2. *The pseudo-dimension of the Normalized Adjacency Matrix Based Algorithmic Family, $\mathcal{F}_\delta$, is Pdim($\mathcal{H}_\delta$) = $\Theta(\log n)$, where n is the total number of labeled and unlabeled data points.*

C. Proofs in Section 3

We provide the proof for Theorem 3.1, Theorem B.1, and Theorem B.2 together, since they follow a similar template.

First, we give an overview of the proof idea below.

Upper Bound We investigate the function structure of each index in F^*. For the function classes $\mathcal{F}_\alpha$ and $\mathcal{F}_\lambda$, the following lemma is useful.

Lemma C.1. *Let $A, B \in \mathbb{R}^{n\times n}$, and $C(x) = (A + xB)^{-1}$ for some $x \in \mathbb{R}$. Each entry of $C(x)$ is a rational polynomial $P_{ij}(x)/Q(x)$ for $i, j \in [n]$ with each P_{ij} of degree at most $n-1$ and Q of degree at most n.*

This lemma reduces each index in the matrix of form $C(x) = (A + xB)^{-1}$ into a polynomial of parameter x with degree at most n. By definition, we can apply this lemma to F^*_α and F^*_λ and conclude that each index of these matrices is a degree-n polynomial of variable α and λ, respectively.

For the algorithm family $\mathcal{F}_\delta$, the following lemma is helpful:

Lemma C.2. *Let $S = D^{-x}WD^{x-1} \in \mathbb{R}^{n\times n}$, and $C(x) = (I - c \cdot S)^{-1}$ for some constant $c \in (0, 1)$ and variable $x \in [0, 1]$. For any $i, j \in [n]$, the i, j-the entry of $C(x)$ is an exponential $C(x)_{ij} = a_{ij} \exp(b_{ij}x)$ for some constants a_{ij}, b_{ij}.*

By definition of F^*_δ, this lemma indicates that each index of F^*_δ is a weighted sum of n exponentials of the hyperparameter δ.

For F^* being a prediction matrix of any of the above three algorithmic family. Recall that the prediction on each node $i \in [n]$ is $\hat{y}_i = \text{argmax}_{j\in[c]}([F^*]_{ij})$, so the prediction on a node can change only when $\text{sign}([F^*]_{ij} - [F^*]_{ik})$ changes for some $j, k \in [c]$. For the families $\mathcal{F}_\alpha$ and $\mathcal{F}_\lambda$, $[F^*]_{ij} - [F^*]_{ik}$ is a rational polynomial $(P_{ij}(\alpha) - P_{ik}(\alpha))/Q(\alpha)$ where $(P_{ij}(\alpha) - P_{ik}(\alpha))$ and $Q(\alpha)$ are degree of at most n (we can simply replace α with λ for $\mathcal{F}_\lambda$). Therefore, its sign can only change at most $O(n)$ times. For the family $\mathcal{F}_\delta$, we refer to the following lemma and conclude that the sign of $F^*_{ij} - F^*_{ik}$ can only change at most $O(n)$ times as well.

Lemma C.3. *Let $a_1, \ldots, a_n \in \mathbb{R}$ be not all zero, $b_1, \ldots, b_n \in \mathbb{R}$, and $f(x) = \sum_{i=1}^n a_i e^{b_i x}$. The number of roots of f is at most $n-1$.*

Therefore, for all three families, the prediction on a single node can change at most $\binom{c}{2}O(n) \in O(nc^2)$ times as the hyperparameter is varied. For n nodes, this implies we have at most $O(mn^2c^2)$ distinct values of the loss function over the m problem instances. The pseudo-dimension m satisfies $2^m \le O(mn^2c^2)$, which implies $\text{Pdim}(\mathcal{H}_\alpha) = \text{Pdim}(\mathcal{H}_\lambda) = \text{Pdim}(\mathcal{H}_\delta) = O(\log n)$.

Lower Bound Our proof relies on a collection of parameter thresholds and well-designed labeling instances that are shattered by the thresholds. Here we present the proof idea of pseudo-dimension lower bound of the family $\mathcal{F}_\alpha$. The pseuodo-dimension lower bound for families $\mathcal{F}_\lambda$ and $\mathcal{F}_\delta$ depends on a similar construction.

We first describe a hard instance of 4 nodes, using binary labels a and b. We have two points labeled a (namely a_1, a_2), and one point labeled b (namely b_1) connected with both a_1 and a_2 with edge weight 1. We also have an unlabeled point u connected to b_1 with edge weight $x \ge 0$. That is, the affinity matrix and initial labels are

$$W = \begin{bmatrix} 0 & 1 & 1 & x \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ x & 0 & 0 & 0 \end{bmatrix}, Y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}.$$

With this construction, the prediction on node u changes and only change when $\alpha = \frac{(x+2)^{1/2}}{2}$. Let $x = 4\beta^2 - 2 \ge 0$, then $\hat{y}_4 = 0$ when $\alpha < \beta$ and $\hat{y}_4 = 1$ when $\alpha \ge \beta$.

Now we can create a large graph of n nodes, consisting of $n/4$ connected components as described above. We assume 4 divides n for simplicity. Given a sequence of α's such that $0 < \alpha_0 < 1/\sqrt{2} \le \alpha_1 < \alpha_2 < ... < \alpha_{n/4} < 1$, we can create the i-th connected component with $x = 4\alpha_i^2 - 2$. Now the predicted label of the unlabeled node in the i-th connected

component is 0 when $\alpha < \alpha_i$ and 1 when $\alpha \geq \alpha_i$. By alternatively labeling these unlabeled nodes, the 0-1 loss of this problem instance fluctuates as α increases.

Finally, by precisely choosing the subsequences so that the oscillations align with the bit flips in the binary digit sequence, we can construct m instances that satisfy the 2^m shattering constraints.

C.1. Proof of Invertibility

Lemma C.4. *Let $G = (V, E)$ be a graph with n vertices and m weighted edges. Let W be its adjacency matrix, $D = diag(W\vec{1}_n)$, $L = D - W$, $\Delta \in \{0,1\}^{n\times n}$ be a diagonal matrix where elements are 1 only if the index is in the labeled set, and $\lambda > 0$. Then, $(L + \lambda\Delta)$ is invertible if, for each connected component in G, there is at least one labeled node.*

Proof. Let $x \in \mathbb{R}^n \backslash \{0\}$, then

$$x^\top (L + \lambda\Delta)x = x^\top B^\top Bx + \lambda x^\top \Delta x \geq 0,$$

where $B \in \mathbb{R}^{m\times n}$ is incidence matrix defined by

$$B_{e,v} = \begin{cases} -\sqrt{W_e}, & \text{if } v \text{ is the initial vertex of edge } e \\ \sqrt{W_e}, & \text{if } v \text{ is the end vertex of edge } e \\ 0, & \text{otherwise.} \end{cases}$$

This shows that $(L + \lambda\Delta)$ is positive semi-definite, so it is invertible only when it is positive definite, or equivalently, $x^\top (L + \lambda\Delta)x \neq 0$.

Consider when $x^\top (L + \lambda\Delta)x = 0$, then $x^\top B^\top Bx = 0$ and $\lambda x^\top \Delta x = 0$. For $x^\top B^\top Bx = (Bx)^\top (Bx) = 0$, this means

$$[Bx]_e = \sqrt{W_e}(x_{e_2} - x_{e_1}) = 0, \quad \text{for all edges } e,$$

where e_1, e_2 are the initial and end vertices of edge e. This implies $x^\top B^\top Bx = 0$ only when $x_i = x_j$ for any i, j that are connected. Let $g_1, ..., g_k \subseteq V$ represent the sets of vertices in each connected component of G, and $(a_j)_{i\in[k]}$ be a sequence of real numbers that are not all zero. For each $j \in [k]$, we assume $x_i = a_j$ for all $i \in g_j$.

Under these assumptions on x, $x \neq \mathbf{0}$ and $x^\top B^\top Bx = 0$ always holds. We now consider when $\lambda x^\top \Delta x = 0$:

$$\lambda x^\top \Delta x = \lambda \sum_{i=1}^{n} \Delta_{ii} x_i^2 = \lambda \sum_{j=1}^{k} a_j^2 \left(\sum_{i\in g_j} \Delta_{ii} \right) = \lambda \sum_{j\in[k], a_j \neq 0} a_j^2 \left(\sum_{i\in g_j} \mathbf{1}_{\text{label}}(i) \right) = 0.$$

Since $\lambda > 0$ and $\mathbf{x} \neq 0$, the equation only holds when $\sum_{i\in g_j} \mathbf{1}_{\text{label}}(i) = 0$ for some g_j, i.e. there exists some connected component such that none of its nodes are labeled.

We conclude that the matrix $(L + \lambda\Delta)$ is positive definite and thus invertible if there exists at least one labeled node in each connected component of G. □

C.2. Proof of Lemma C.1

Proof. Using the adjugate matrix, we have

$$C(x) = \frac{1}{\det(A + xB)} \text{adj}(A + xB).$$

The determinant of $A + xB$ can be written as

$$\det(A + xB) = \sum_{\sigma\in S_n} \left(\text{sgn}(\sigma) \prod_{i=1}^{n} [A + xB]_{i\sigma_i} \right),$$

where S_n represents the symmetric group and $\text{sgn}(\sigma) \in \{\pm 1\}$ is the signature of permutation σ. Thus $\det(A + xB)$ is a polynomial of x with a degree at most n. The adjugate of $A + xB$ is

$$\text{adj}(A + xB) = C^\top,$$

where C is the cofactor matrix of $A + xB$. By definition, each entry of C is $C_{ij} = (-1)^{i+j} k_{ij}$ where k_{ij} is the determinant of the $(n-1) \times (n-1)$ matrix that results from deleting i-th row and j-th column of $A + xB$. This implies that each entry of C (and thus $\text{adj}(A + xB)$) is a polynomial of degree at most $n - 1$. Letting $Q(x) = \det(A + xB)$ and $P_{ij}(x) = [\text{adj}(A + xB)]_{ij}$ concludes our proof. □

C.3. Proof of Lemma C.2

Proof. The ij-th element of $I - c \cdot S$ is

$$[I - c \cdot S]_{ij} = \begin{cases} -c \cdot d_i^{-\delta} W_{ij} d_j^{\delta - 1} = -(d_i^{-1} d_j)^{\delta} (c \cdot W_{ij} d_j^{-1}) & \text{, if } i \neq j \\ 1 = (d_i^{-1} d_i)^{\delta} & \text{, otherwise.} \end{cases}$$

Using adjugate matrix, we have

$$(I - c \cdot S)^{-1} = \frac{1}{\det(I - c \cdot S)} \text{adj}(I - c \cdot S).$$

Note that the determinant of any $k \times k$ matrix A can be written as

$$\det(A) = \sum_{\sigma \in S_k} \left(\text{sgn}(\sigma) \prod_{i=1}^{k} [A]_{i\sigma_i} \right),$$

where S_k represents the symmetric group and $\text{sgn}(\sigma) \in \{\pm 1\}$ is the signature of permutation σ.

Now consider $\text{adj}(I - c \cdot S)$. Let M_{ij} be the $(n-1) \times (n-1)$ matrix resulting from deleting i-th row and j-th column from $[I - c \cdot S]$. Then,

$$[\text{adj}(I - c \cdot S)]_{ij} = (-1)^{i+j} \det(M_{ji}) = \sum_{\sigma \in S_{n-1}} \left(\text{sgn}(\sigma) \prod_{k=1}^{n-1} [M_{ji}]_{k\sigma_k} \right) = \sum_{\sigma \in S_{n-1}} \left(a_\sigma \exp(\delta \ln b_\sigma) \right),$$

for some constants a_σ, b_σ that satisfies

$$b_\sigma = \Big(\prod_{k \in [n] \setminus \{j\}} d_k^{-1} \Big) \Big(\prod_{k \in [n] \setminus \{i\}} d_k \Big) = d_i^{-1} d_j.$$

We can then rewrite $[\text{adj}(I - c \cdot S)]_{ij}$ as

$$[\text{adj}(I - c \cdot S)]_{ij} = \sum_{\sigma \in S_{n-1}} (a_\sigma \exp(\delta \ln(d_i^{-1} d_j))) = a_{ij} \exp(\delta \ln(d_i^{-1} d_j)),$$

where $a_{ij} = \sum_{\sigma \in S_{n-1}} a_\sigma$. □

C.4. Proof of Lemma C.3

Proof. We prove by induction on n. If $n = 1$, then $f(x) = ae^{bx}$ and $a \neq 0$, so $f(x)$ has $0 = n - 1$ root. Now assume that the statement holds for some $n = m$ and consider when $n = m + 1$. That is, we have

$$f(x) = \sum_{i=1}^{m+1} a_i e^{b_i x}.$$

Assume for the sake of contradiction that f has $n = m + 1$ roots. Define

$$g(x) = \frac{f(x)}{e^{b_{m+1} x}} = \sum_{i=1}^{m} a_i e^{(b_i - b_{m+1}) x} + a_{m+1},$$

then g also has $m + 1$ roots. Since g is continuous,

$$g'(x) = \sum_{i=1}^{m} (b_i - b_{m+1}) a_i e^{(b_i - b_{m+1}) x}$$

must have m roots. However, using our induction hypothesis, it should have at most $m-1$ roots. This means our assumption is incorrect, i.e. f must have at most $m = n-1$ roots.

We conclude that f must have at most $n-1$ roots. □

C.5. Proof of Theorem 3.1

Upper Bound. Proof is given in Section 3.

Lower Bound. We first construct the small connected component of 4 nodes:

Lemma C.5. *Given $x \in [1/\sqrt{2}, 1)$, there exists a labeling instance (G, L) with 4 nodes, such that the predicted label of the unlabeled points changes only at $\alpha = x$ as α varies in $(0, 1)$.*

Proof. We use binary labeling a and b. We have two points labeled a (namely a_1, a_2), and one point labeled b (namely b_1) connected with both a_1 and a_2 with edge weight 1. We also have an unlabeled point u connected to b_1 with edge weight $x \geq 0$. That is, the affinity matrix and initial labels are

$$W = \begin{bmatrix} 0 & 1 & 1 & x \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ x & 0 & 0 & 0 \end{bmatrix}, Y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}.$$

Recall that the score matrix is

$$F^* = (1-\alpha)(I - \alpha S)^{-1} Y.$$

We now calculate:

$$D^{-1/2} = \begin{bmatrix} (x+2)^{-1/2} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & x^{-1/2} \end{bmatrix},$$

$$S = D^{-1/2} W D^{-1/2} = \begin{bmatrix} 0 & (x+2)^{-1/2} & (x+2)^{-1/2} & x^{1/2}(x+2)^{-1/2} \\ (x+2)^{-1/2} & 0 & 0 & 0 \\ (x+2)^{-1/2} & 0 & 0 & 0 \\ x^{1/2}(x+2)^{-1/2} & 0 & 0 & 0 \end{bmatrix},$$

$$\begin{aligned} (I - \alpha S)^{-1} =& \frac{1}{\det(I - \alpha S)} \text{adj}(I - \alpha S) \\ =& \frac{1}{1-\alpha^2} \begin{bmatrix} 1 & \frac{\alpha}{(x+2)^{1/2}} & \frac{\alpha}{(x+2)^{1/2}} & \frac{\alpha x^{1/2}}{(x+2)^{1/2}} \\ \frac{\alpha}{(x+2)^{1/2}} & 1 - \frac{\alpha^2(x+1)x}{(x+2)} & \frac{\alpha^2}{x+2} & \frac{\alpha^2 x^{1/2}}{(x+2)} \\ \frac{\alpha}{(x+2)^{1/2}} & \frac{\alpha^2}{x+2} & 1 - \frac{\alpha^2(x+1)x}{(x+2)} & \frac{\alpha^2 x^{1/2}}{(x+2)} \\ \frac{\alpha x^{1/2}}{(x+2)^{1/2}} & \frac{\alpha^2 x^{1/2}}{(x+2)} & \frac{\alpha^2 x^{1/2}}{(x+2)} & 1 - \frac{2\alpha^2}{x+2} \end{bmatrix}. \end{aligned}$$

Recall that the prediction on the unlabeled point is $\hat{y}_4 = \text{argmax} F^*_4$, so we calculate

$$\begin{aligned} \hat{y}_4 = \text{sign}(F^*_{4,2} - F^*_{4,1}) =& \text{sign}\left(\frac{\alpha x^{1/2}(2\alpha - (x+2)^{1/2})}{(1+\alpha)(x+2)} \right) \\ =& \text{sign}\left(x^{1/2}(2\alpha - (x+2)^{1/2}) \right). \qquad \text{(since } \alpha \in (0,1) \text{ and } x \geq 0) \end{aligned}$$

Solving the equation $x^{1/2}(2\alpha - (x+2)^{1/2}) = 0$, we know that the prediction changes and only change when $\alpha = \frac{(x+2)^{1/2}}{2}$. Let $x = 4x^2 - 2 \geq 0$, then $\hat{y}_4 = 0$ when $\alpha < x$ and $\hat{y}_4 = 1$ when $\alpha \geq x$, which completes our proof. □

Lemma C.6. *Given integer $n > 1$ and a sequence of α's such that $0 < \alpha_0 < 1/\sqrt{2} \leq \alpha_1 < \alpha_2 < ... < \alpha_n < 1$, there exists a real-valued witness $w > 0$ and a problem instance of partially labeled $4n$ points, such that for $0 \leq i \leq n/2 - 1$, $l < w$ for $\alpha \in (\alpha_{2i}, \alpha_{2i+1})$, and $l > w$ for $\alpha \in (\alpha_{2i+1}, \alpha_{2i+2})$.*

Proof. We create n connected components using the previous lemma, with $x_i = \alpha_i$. Let the unlabeled point in the ith component be u_i, then as α increases from α_{i-1} to α_i, the predicted label of u_i changes from a to b. If the sequence u_i is alternately labeled with u_1 labeled a, then the loss increases and decreases alternately as all the labels turn to b when α increases to α_n. Specifically, as α increases to α_1, the point u_1 has predicted label changes from a to b. Since its true label is a and the predicted labels of other u_i's remain unchanged, our loss slightly increases to l_{max}. Then, as α increases to α_2, the point u_2 gets correctly labeled as b and all other nodes unchanged, which slightly decreases our loss back to l_{min}. The loss thus fluctuates between l_{min} and l_{max}. We therefore set the witness w as something in between.

$$w = \frac{l_{min} + l_{max}}{2}.$$

□

We now finish the lower bound proof for Theorem 3.1.

Proof. Arbitrarily choose $n' = n/4$ (assumed to be a power of 2 for convenient representation) real numbers $1/\sqrt{2} \leq \alpha_{[000..1]} < \alpha_{[000...10]} < ... < \alpha_{[111...11]} < 1$. The indices are increasing binary numbers of length $m = \log n'$. We create m labeling instances that can be shattered by these α values. For the i-th instance $(X^{(i)}, Y^{(i)})$, we apply the previous lemma with a subset of the α_b sequence that corresponds to the i-th bit flip in b, where $b \in \{0,1\}^m$. For example, $(X^{(1)}, Y^{(1)})$ is constructed using $r_{[100..0]}$, and $(X^{(2)}, Y^{(2)})$ is constructed using $r_{[010..0]}, r_{[100.0]}$ and $r_{[110..0]}$. The lemma gives us both the instances and the sequence of witnesses w_i.

This construction ensures $\text{sign}(l_{\alpha_b} - w_i) = b_i$ for all $b \in \{0,1\}^m$. Thus the pseudo-dimension is at least $\log n' = \log n - \log 4 = \Omega(\log n)$ □

C.6. Proof of Theorem B.1

Upper Bound. The closed-form solution F^* is given by

$$F^* = (S + \lambda I_n \Delta_{i \in L})^{-1} \lambda Y.$$

By Lemma C.1, each coefficient $[F^*]_{ij}$ is a rational polynomial in λ of the form $P_{ij}(\lambda)/Q(\lambda)$ where P_{ij} and Q are polynomials of degree n and n respectively. Note that the prediction for each node $i \in [n]$ is $\hat{y}_i = \text{argmax}_{j \in c} f_{ij}$ and thus the prediction on any node in the graph can only change when $sign(f_{ij} - f_{ik})$ changes for some $j, k \in [c]$. Note that $f_{ij} - f_{ik}$ is also a rational polynomial $(P_{ij}(\lambda) - P_{ik}(\lambda))/Q(\lambda)$ where both the numerator and denominator are polynomials in λ of degree n, meaning the sign can change at most $O(n)$ times. As we vary λ, we have that the prediction on a single node can change at most $\binom{c}{2} O(n) \in O(nc^2)$. Across the m problem instances and the n total nodes, we have at most $O(n^2 c^2 m)$ distinct values of our loss function. The pseudo-dimension m thus satisfies $2^m \leq O(n^2 c^2 m)$, or $m = O(\log n)$

Lower Bound. We construct the small connected component of 4 nodes as follows:

Lemma C.7. *Given $\lambda' \in (1, \infty)$, there exists a labeling instance (X, Y) with 4 nodes, such that the predicted label of the unlabeled points changes only at $\lambda = \lambda'$ as λ varies in $(0, \infty)$.*

Proof. We use binary labeling a and b. We have two points labeled a (namely a_1, a_2), and one point labeled b (namely b_1). We also have an unlabeled point u connected to b_1 with edge weight $x \geq 0$ and connected with both a_1 and a_2 with edge weight 1. That is, the weight matrix and initial labels are

$$W = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & x \\ 0 & 0 & x & 0 \end{bmatrix}, Y = \begin{bmatrix} -1 \\ -1 \\ 0 \\ 1 \end{bmatrix}.$$

The closed form solution is

$$F^* = (S + \lambda I_n \Delta_{i\in L})^{-1} \lambda Y$$

where $S = \text{diag}(W\vec{1}_n) - W$. We now calculate:

$$S = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & -1 & 0 \\ -1 & -1 & x+2 & -x \\ 0 & 0 & -x & x \end{bmatrix}$$

$$S + \lambda I_n \Delta_{i\in L} = \begin{bmatrix} 1+\lambda & 0 & -1 & 0 \\ 0 & 1+\lambda & -1 & 0 \\ -1 & -1 & x+2 & -x \\ 0 & 0 & -x & x+\lambda \end{bmatrix}$$

Recall that the prediction on the unlabeled point is $\hat{y}_3 = \text{sign}([F*]_{32} - [F^*]_{31})$, so we calculate

$$\begin{aligned} \hat{y}_3 = \text{sign}(F*]_{32} - [F^*]_{31}) =& \text{sign}\left(-2\lambda\left(\frac{\lambda + x}{\lambda^2 x + 2\lambda^2 + 3\lambda x}\right) + \lambda\left(\frac{\lambda x + x}{\lambda^2 x + 2\lambda^2 + 3\lambda x}\right)\right) \\ =& \text{sign}\left(-2\lambda(\lambda + x) + \lambda(\lambda x + x)\right) && \text{(since } \lambda > 0 \text{ and } x \geq 0\text{)} \\ =& \text{sign}\left(-2(\lambda + x) + (\lambda x + x)\right) && \text{(since } \lambda > 0\text{)} \\ =& \text{sign}\left(-2\lambda - x + \lambda x\right) \end{aligned}$$

Solving the equation $-2\lambda - x + \lambda x = 0$, we know that the prediction changes and only change when $\lambda = \frac{x}{x-2}$. Let $x = \frac{2\lambda}{\lambda - 1} \geq 0$, then $\hat{y}_3 = -1$ when $\lambda < \lambda'$ and $\hat{y}_3 = 1$ when $\lambda \geq \lambda'$, which completes our proof. □

The remaining proof is exactly the same as Lemma C.6 and Theorem 3.1, by simply replacing notation α with λ.

C.7. Proof of Theorem B.2

Upper Bound. Using Lemma C.2, we know that each entry of F^* is

$$F^*_{ij}(\delta) = \frac{1}{\det(I - c \cdot S)} \sum_{k=1}^{n} [\text{adj}(I - c \cdot S)]_{ik} Y_{kj} = \frac{1}{\det(I - c \cdot S)} \sum_{k=1}^{n} (a_{ik} Y_{kj}) \exp(\delta \ln(d_i^{-1} d_k)).$$

Recall that the prediction on a node is made by $\hat{y}_i = \text{argmax}(F^*_i)$, so the prediction changes only when

$$\begin{aligned} F^*_{ic_1} - F^*_{ic_2} &= \frac{1}{\det(I - c \cdot S)} \left(\sum_{k=1}^{n} (a_{ik} Y_{kc_1}) \exp(\delta \ln(d_i^{-1} d_k)) - \sum_{k=1}^{n} (a_{ik} Y_{kc_2}) \exp(\delta \ln(d_i^{-1} d_k)) \right) \\ &= \frac{1}{\det(I - c \cdot S)} \left(\sum_{k=1}^{n} (a_{ik} (Y_{kc_1} - Y_{kc_2})) \exp(\delta \ln(d_i^{-1} d_k)) \right) \\ &= 0. \end{aligned}$$

By Lemma C.3, $F^*_{ic_1} - F^*_{ic_2}$ has at most $n - 1$ roots, so the prediction on node i can change at most $n - 1$ times. As δ vary, the prediction can change at most $\binom{c}{2} O(n) \in O(nc^2)$ times. For n nodes and m problem instances, this implies that we have at most $O(mn^2c^2)$ distinct values of loss. The pseudo-dimension m then satisfies $2^m \leq O(mn^2c^2)$, or $m = O(\log nc)$.

Lower Bound We construct the small connected component as follows:

Lemma C.8. *Consider when $c \geq 1/2$. Given $x \in [\log(2c)/\log(2), 1)$, there exists a labeling instance (G, L) with 4 nodes, such that the predicted label of the unlabeled points changes only at $\delta = x$ as δ varies in $(0, 1)$.*

Proof. We use binary labeling a and b. We have two points labeled a (namely a_1, a_2), and one point labeled b (namely b_1) connected with both a_1 and a_2 with edge weight 1. We also have an unlabeled point u connected to b_1 with edge weight

$x \geq 0$. That is, the affinity matrix and initial labels are

$$W = \begin{bmatrix} 0 & 1 & 1 & x \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ x & 0 & 0 & 0 \end{bmatrix}, Y = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}.$$

Recall that the score matrix is

$$F^* = (I - c \cdot S)^{-1} Y,$$

where $S = D^{-\delta} W D^{\delta-1}$ and D is diagonal with $D_{ii} = \sum_i W_{ij}$. We now calculate:

$$S = D^{-\delta} W D^{\delta-1} = \begin{bmatrix} 0 & (x+2)^{-\delta} & (x+2)^{-\delta} & x^{\delta}(x+2)^{-\delta} \\ (x+2)^{-\delta} & 0 & 0 & 0 \\ (x+2)^{-\delta} & 0 & 0 & 0 \\ x^{\delta}(x+2)^{-\delta} & 0 & 0 & 0 \end{bmatrix},$$

$$\begin{aligned} \det(I - c \cdot S) &= \det \begin{bmatrix} 1 & -c(x+2)^{-\delta} & -c(x+2)^{-\delta} & -cx^{\delta}(x+2)^{-\delta} \\ -c(x+2)^{-\delta} & 1 & 0 & 0 \\ -c(x+2)^{-\delta} & 0 & 1 & 0 \\ -cx^{\delta}(x+2)^{-\delta} & 0 & 0 & 1 \end{bmatrix} \\ &= 1 - c^2 \neq 0, \end{aligned}$$

so $(I - c \cdot S)$ is invertible on our instance.

Recall that the prediction on the unlabeled point is $\hat{y}_4 = \text{argmax} F_4^*$, so we calculate

$$\hat{y}_4 = \text{sign}(F^*_{4,2} - F^*_{4,1}) = \text{sign}\left(\frac{c \cdot x^{1-\delta}(2c - (x+2)^{\delta})}{(1-c^2)(x+2)}\right) = \text{sign}\left(2c - (x+2)^{\delta}\right). \qquad \text{(since } c \in (0,1), \text{ and } x \geq 0)$$

Solving the equation $2c - (x+2)^{\delta} = 0$, we know that the prediction changes and only change when $\delta = \frac{\ln(2c)}{\ln(x+2)}$. Since $x \leq \ln(2c)/\ln(2) \leq 1$, we can let $x = (2c)^{1/x} - 2 \geq 0$, then $\hat{y}_4 = 0$ when $\alpha < x$ and $\hat{y}_4 = 1$ when $\alpha \geq x$, which completes our proof. □

D. Introduction of GCN and GAT

Graph Convolutional Neural Networks (GCNs) The fundamental idea behind GCNs is to repeatedly apply the convolution operator on graphs (Kipf & Welling, 2017). Define $h_i^0 = z_i$ as the input feature of the i-th node and let h_i^{ℓ} be the feature of the ℓ-th layer of the i-th node. We have the following update rule for the features of h_i^{ℓ}

$$h_i^{\ell} = \sigma\left(\sum_{j \in \mathcal{N}_i} \frac{1}{\sqrt{d_i d_j}} U^{\ell-1} h_j^{\ell-1}\right)$$

where d_i represents the degree of vertex i, U^{ℓ} represents the learnable weights in our model, $\mathcal{N}_i$ represents the neighbors of vertex i, and $\sigma(\cdot)$ is the activation function.

Graph Attention Neural Networks (GATs) GAT is a more recent architecture that leverages the self-attention mechanisms to capture the importance of neighboring nodes to generate the features of the next layer (Veličković, Petar et al., 2018). One of the advantages of GAT is its ability to capture long-range dependencies within the graph while giving more weight to influential nodes. This makes GAT particularly effective for tasks involving irregular graph structures and tasks where global context is essential.

Different from GCN, GAT uses the update rule for each layer

$$h_i^{\ell} = \sigma(\sum_{j \in \mathcal{N}_i} e_{ij}^{\ell-1} U^{\ell-1} h_j^{\ell-1}),$$

where

$$e_{ij}^{\ell} = \frac{\exp(\hat{e}_{ij}^{\ell})}{\sum_{j' \in \mathcal{N}_i} \exp(\hat{e}_{ij'}^{\ell})}, \;\; \hat{e}_{ij}^{\ell} = \sigma(V^{\ell}[U^{\ell} h_i^{\ell}, U^{\ell} h_j^{\ell}]). \tag{1}$$

Here e_{ij}^{ℓ} is the attention score of node j for node i and V^{ℓ} and U^{ℓ} are learnable parameters.

E. Proofs in Section 4

First, we show that, to show that an algorithm family F_ρ is strong for new problem instances, it suffices to bound the empirical Rademacher complexity $\hat{R}_m(\mathcal{H}_\rho^\gamma)$.

We define the margin loss as $\tau(f(z_i), y_i) = (2f(z_i) - 1)y_i$. Then, $\tau(f(z_i), y_i) < 0$ if and only if there is a classification error. The margin loss with a loss hyperparameter $\gamma > 0$ is defined as

$$\ell_\gamma(f(z_i), y_i) = \mathbf{1}[a_i > 0] + (1 + a_i/\gamma)\mathbf{1}\left[a_i \in [-\gamma, 0]\right]$$

where $a_i = -\tau(f(z_i), y_i)$.

Now we define $H_\rho^\gamma(X) = \frac{1}{n}\sum_{i=1}^n \ell_\gamma\left(f_\rho(z_i), y_i\right)$ to be the margin loss of the entire graph when using a parameterized algorithm F_ρ. Based on this definition, we have an induced loss function family $\mathcal{H}_\rho^\gamma$. Now given m instances, for any $\gamma > 0$, we can obtain an upper bound for all $H_\rho^\gamma \in \mathcal{H}_\rho^\gamma$:

$$\begin{aligned}
&\mathbb{E}_{(X,Y)\sim\mathcal{D}}\left[\frac{1}{n}\sum_{i=1}^n \ell_{0-1}\left(f_{\hat{\rho}}(z_i), y_i\right)\right] \\
=&\frac{1}{m}\sum_{i=1}^m H_\rho^\gamma(X^{(k)}) + O\left(\hat{R}_m(\mathcal{H}_\rho^\gamma) + \sqrt{\frac{\log(1/\delta)}{m}}\right). && \text{(by Theorem A.2)}
\end{aligned}$$

Therefore, suppose we find a $\hat{\rho}$ whose empirical margin loss $\frac{1}{m}\sum_{i=1}^m H_{\hat{\rho}}^\gamma(X^{(k)})$ is small, and if we can show $\hat{R}_m(\mathcal{H}_\rho^\gamma)$ is small, then $F_{\hat{\rho}}$ is a strong algorithm for the new problem instances.

We provide additional proof details from Section 4 below.

E.1. Proof of Theorem 4.1

Lemma E.1. *For any $z, z', \in \mathbb{R}^{d\times r}$ and $b, b' \in \mathbb{R}^{r\times t}$ such that $\|z\|_F \le C_z, \|z'\|_F \le C_z, \|b\|_F \le C_b, \|b'\|_F \le C_b$, we have*

$$\|zb - z'b'\|_F \le C_z\|b - b'\|_F + C_b\|z - z'\|_F.$$

The result also holds when z, b, z', b' are vectors or real numbers. The corresponding norms are $\|\cdot\|$ and $|\cdot|$.

Also, by recursively using the inequality above, we may have that for any $z_1, \ldots, z_n$ and $z_1', \ldots, z_n'$ such that $\|z_i\| \le C_i, \|z_i'\| \le C_i$,

$$\|z_1 z_2 \ldots z_n - z_1' z_2' \ldots z_n'\| \le \sum_{i=1}^n \left(\|z_i - z_i'\| \prod_{j\in[n], j\neq i} C_j\right).$$

Here, for simplicity of notation, we used $\|\cdot\|$ to denote the type of norm that corresponds to the dimension of the z_i's.

Proof.

$$\begin{aligned}
\|ab - a'b'\|_F &= \|ab - a'b' + ab' - ab'\|_F \\
&\le \|ab - ab'\|_F + \|ab' - a'b'\|_F && \text{(by triangle inequality)} \\
&\le \|a\|_F\|b - b'\|_F + \|b'\|_F\|a - a'\|_F && \text{(by Cauchy-Schwarz inequality)} \\
&\le C_z\|b - b'\|_F + C_b\|a - a'\|_F
\end{aligned}$$

□

Lemma E.2. *The l_2 norm of different embedding vectors at level L, h_i^L, produced by $(\alpha, U, V), (\alpha', U', V')$ after they process the tree all the way from the leaf level to the root can be bounded as*

$$\begin{aligned}\Delta_{i,L} \leq & C_U(\max_{j\in\mathcal{N}_i}\|h_j^{L-1}\|)|\eta-\eta'| + rC_U(\max_{j\in\mathcal{N}_i}\|h_j^{L-1}\|) + (\max_{j\in\mathcal{N}_i}\|h_j^{L-1}\|)\|U-U'\| \\ & + C_U(\max_{j\in\mathcal{N}_i}\|h_j^{L-1} - h_j^{\prime(L-1)}\|) + \frac{2rC_U}{C_{dl}}\|h_i^{L-1} - h_i^{\prime(L-1)}\| \\ & + \frac{2r}{C_{dl}}\|h_i^{L-1}\|\|U-U'\| + \frac{2rC_U}{C_{dl}}|\eta-\eta'|\end{aligned}$$

Proof.

$$\begin{aligned}\Delta_{i,L} =& \|h_i^L(\eta, U, V) - h_i^L(\eta', U', V')\| \\ =& \|\sigma\left(\sum_{j\in\mathcal{N}_i}\left(\eta\cdot e_{ij}^{L-1} + (1-\eta)\cdot\frac{1}{\sqrt{d_i d_j}}\right)Uh_j^{L-1}\right) \\ & - \sigma\left(\sum_{j\in\mathcal{N}_i}\left(\eta'\cdot e_{ij}^{\prime(L-1)} + (1-\eta')\cdot\frac{1}{\sqrt{d_i d_j}}\right)U'h_j^{\prime(L-1)}\right)\| \\ \leq& \|\sum_{j\in\mathcal{N}_i}\left((\eta\cdot e_{ij}^{L-1}Uh_j^{L-1}) - (\eta'\cdot e_{ij}^{\prime(L-1)}U'h_j^{\prime(L-1)})\right) \\ & + \sum_{j\in\mathcal{N}_i}\left((1-\eta)\cdot\frac{1}{\sqrt{d_i d_j}}Uh_j^{L-1} - (1-\eta')\cdot\frac{1}{\sqrt{d_i d_j}}U'h_j^{\prime(L-1)}\right)\| && \text{(since } \sigma \text{ is 1-Lipschitz)} \\ \leq& \sum_{j\in\mathcal{N}_i}\|(\eta\cdot e_{ij}^{L-1}Uh_j^{L-1}) - (\eta'\cdot e_{ij}^{\prime(L-1)}U'h_j^{\prime(L-1)})\| \\ & + \sum_{j\in\mathcal{N}_i}\|(1-\eta)\cdot\frac{1}{\sqrt{d_i d_j}}Uh_i^{L-1} - (1-\eta')\cdot\frac{1}{\sqrt{d_i d_j}}U'h_i^{\prime(L-1)}\| && \text{(by triangle inequality)}\end{aligned}$$

Using Lemma E.1, we can bound each term in the first summation as

$$\begin{aligned}&\|(\eta\cdot e_{ij}^{L-1}Uh_j^{L-1}) - (\eta'\cdot e_{ij}^{\prime(L-1)}U'h_j^{\prime(L-1)})\| \\ \leq& C_U\bar{e}_{ij}^{L-1}\bar{h}_j^{L-1}\cdot|\eta-\eta'| + C_U\bar{h}_j^{L-1}\cdot|e_{ij}^{L-1} - e_{ij}^{\prime(L-1)}| \\ & + \bar{e}_{ij}^{L-1}\bar{h}_j^{L-1}\|U-U'\| + C_U\bar{e}_{ij}^{L-1}\|h_j^{L-1} - h_j^{\prime(L-1)}\|\end{aligned}$$

Here, $\bar{h}_j^{L-1}$ is an upper bound on $\|h_j^{L-1}\|$ and $\|h_j^{\prime(L-1)}\|$, and $\bar{e}_{ij}^{L-1}$ is an upper bound on $|e_{ij}^{L-1}|$ and $|e_{ij}^{\prime(L-1)}|$.

Bounding each term in the second summation, we have

$$\begin{aligned}&\|(1-\eta)\cdot\frac{1}{\sqrt{d_i d_j}}Uh_i^{L-1} - (1-\eta')\cdot\frac{1}{\sqrt{d_i d_j}}U'h_i^{\prime(L-1)}\| \\ \leq& \|\frac{1}{\sqrt{d_i d_j}}Uh_i^{L-1} - \frac{1}{\sqrt{d_i d_j}}U'h_i^{\prime(L-1)}\| + \|\eta\cdot\frac{1}{\sqrt{d_i d_j}}Uh_i^{L-1} - \eta'\cdot\frac{1}{\sqrt{d_i d_j}}U'h_i^{\prime(L-1)}\| && \text{(by triangle inequality)} \\ \leq& \frac{1}{C_{dl}}\|Uh_i^{L-1} - U'h_i^{\prime(L-1)}\| + \frac{1}{C_{dl}}\|\eta\cdot Uh_i^{L-1} - \eta'\cdot U'h_i^{\prime(L-1)}\| \\ \leq& \frac{1}{C_{dl}}\left(C_U\|h_i^{L-1} - h_i^{\prime(L-1)}\| + \bar{h}_i^{L-1}\|U-U'\|\right) \\ & + \frac{1}{C_{dl}}\left(C_U\|h_i^{L-1} - h_i^{\prime(L-1)}\| + \bar{h}_i^{L-1}\|U-U'\| + C_U\bar{h}_i^{L-1}|\eta-\eta'|\right) && \text{(using Lemma E.1)} \\ =& \frac{1}{C_{dl}}\left(2C_U\|h_i^{L-1} - h_i^{\prime(L-1)}\| + 2\bar{h}_i^{L-1}\|U-U'\| + C_U\bar{h}_i^{L-1}|\eta-\eta'|\right).\end{aligned}$$

Combining the above results, we have

$$\begin{aligned}
\Delta_i^L \leq & \sum_{j\in\mathcal{N}_i} \Big(C_U \bar{e}_{ij}^{L-1}\bar{h}_j^{L-1}\cdot|\eta-\eta'| + C_U\bar{h}_j^{L-1}\cdot|e_{ij}^{L-1}-e_{ij}^{\prime(L-1)}| \\
& + \bar{e}_{ij}^{L-1}\bar{h}_j^{L-1}\|U-U'\| + C_U\bar{e}_{ij}^{L-1}\|h_j^{L-1}-h_j^{\prime(L-1)}\| \\
& + \frac{1}{C_{dl}}\big(2C_U\|h_i^{L-1}-h_i^{\prime(L-1)}\| + 2\bar{h}_i^{L-1}\|U-U'\| + C_U\bar{h}_i^{L-1}|\eta-\eta'|\big)\Big) \\
\leq & C_U(\max_{j\in\mathcal{N}_i}\bar{h}_j^{L-1})|\eta-\eta'| + rC_U(\max_{j\in\mathcal{N}_i}\bar{h}_j^{L-1}) + (\max_{j\in\mathcal{N}_i}\bar{h}_j^{L-1})\|U-U'\| \\
& + C_U(\max_{j\in\mathcal{N}_i}\|h_j^{L-1}-h_j^{\prime(L-1)}\|) + \frac{2rC_U}{C_{dl}}\|h_i^{L-1}-h_i^{\prime(L-1)}\| \\
& + \frac{2r}{C_{dl}}\bar{h}_i^{L-1}\|U-U'\| + \frac{2rC_U}{C_{dl}}|\eta-\eta'| \qquad \text{(since } e_{ij}^\ell\leq 1, \textstyle\sum_{j\in\mathcal{N}_i} e_{ij}^\ell = 1\text{, and the branching factor is } r\text{)}
\end{aligned}$$

It remains for us to derive $\bar{h}_j^{L-1}$ for all j. □

Lemma E.3. *We can upper bound the norm of node feature embedding at level $\ell+1$ by*

$$\|h_i^\ell\| \leq r^\ell C_U^{\ell+1}C_z\max(1,\frac{1}{C_{dl}})^\ell.$$

Proof.

$$\begin{aligned}
\|h_i^{\ell+1}\| &= \|\sigma\left(\sum_{j\in\mathcal{N}_i}(\eta\cdot e_{ij}^\ell + (1-\eta)\cdot\frac{1}{\sqrt{d_id_j}})Uh_j^\ell\right)\| \\
&\leq \|\sum_{j\in\mathcal{N}_i}(\eta\cdot e_{ij}^\ell + (1-\eta)\cdot\frac{1}{\sqrt{d_id_j}})Uh_j^\ell\| && \text{(since } \|\sigma(x)\|\leq\|x\|\text{)} \\
&\leq \sum_{j\in\mathcal{N}_i}\left|\eta\cdot e_{ij}^\ell + (1-\eta)\cdot\frac{1}{\sqrt{d_id_j}}\right|\|U\|\|h_j^\ell\| && \text{(by triangle inequality and Cauchy-Schwarz inequality)} \\
&\leq C_U\sum_{j\in\mathcal{N}_i}\left|\eta\cdot e_{ij}^\ell + (1-\eta)\cdot\frac{1}{\sqrt{d_id_j}}\right|\|h_j^\ell\| \\
&\leq rC_U\max(1,\frac{1}{C_{dl}})(\max_{j\in\mathcal{N}_i}\|h_j^{\ell-1}\|)
\end{aligned}$$

Recursively bounding the terms, we have

$$\|h_i^\ell\| \leq r^\ell C_U^\ell\max(1,\frac{1}{C_{dl}})^\ell\max_{j\in[n]}\|h_j^0\| \leq r^\ell C_U^{\ell+1}C_z\max(1,\frac{1}{C_{dl}})^\ell.$$

□

Lemma E.4. *The change in margin loss due to the change in parameter values after L layers satisfies*

$$\Lambda_i \leq \frac{2}{k}\left(k_1 + k_2|\eta-\eta'| + k_3\|U-U'\|\right)\frac{k_4^L-k_4}{k_4-1} + k_4C_z\|U-U'\|,$$

where

$$\begin{aligned}
k_1 &= r^LC_U^{L+1}C_z\max(1,\frac{1}{C_{dl}})^{L-1} \\
k_2 &= r^{L-1}C_U^{L+1}C_z\max(1,\frac{1}{C_{dl}})^{L-1} + \frac{2rC_U}{C_{dl}} \\
k_3 &= \left(1+\frac{2r}{C_{dl}}\right)r^{L-1}C_U^LC_z\max(1,\frac{1}{C_{dl}})^{L-1} \\
k_4 &= C_U + \frac{2rC_U}{C_{dl}}.
\end{aligned}$$

Proof. Using the previous two lemmas, we know

$$\begin{aligned}
&\|h_i^L(\eta,U,V) - h_i^L(\eta',U',V')\| \\
\leq& C_U(\max_{j\in\mathcal{N}_i}\bar{h}_j^{L-1})|\eta-\eta'| + rC_U(\max_{j\in\mathcal{N}_i}\bar{h}_j^{L-1}) + (\max_{j\in\mathcal{N}_i}\bar{h}_j^{L-1})\|U-U'\| \\
&+ C_U(\max_{j\in\mathcal{N}_i}\|h_j^{L-1} - h_j^{\prime(L-1)}\|) + \frac{2rC_U}{C_{dl}}\|h_i^{L-1} - h_i^{\prime(L-1)}\| + \frac{2r}{C_{dl}}\bar{h}_i^{L-1}\|U-U'\| + \frac{2rC_U}{C_{dl}}|\eta-\eta'| \\
\leq& k_1 + k_2|\eta-\eta'| + k_3\|U-U'\| + k_4(\max_{j\in[n]}\|h_j^{L-1} - h_j^{\prime(L-1)}\|) \\
=& (k_1 + k_2|\eta-\eta'| + k_3\|U-U'\|)\,\frac{k_4^L - k_4}{k_4 - 1} + k_4(\max_{j\in[n]}\|h_j^0 - h_j^{\prime 0}\|) \\
\leq& (k_1 + k_2|\eta-\eta'| + k_3\|U-U'\|)\,\frac{k_4^L - k_4}{k_4 - 1} + k_4C_z\|U-U'\|
\end{aligned}$$

where

$$\begin{aligned}
k_1 &= r^L C_U^{L+1} C_z \max(1, \frac{1}{C_{dl}})^{L-1} \\
k_2 &= r^{L-1} C_U^{L+1} C_z \max(1, \frac{1}{C_{dl}})^{L-1} + \frac{2rC_U}{C_{dl}} \\
k_3 &= \left(1 + \frac{2r}{C_{dl}}\right) r^{L-1} C_U^L C_z \max(1, \frac{1}{C_{dl}})^{L-1} \\
k_4 &= C_U + \frac{2rC_U}{C_{dl}}.
\end{aligned}$$

The change in margin loss for each node after L layers is then

$$\begin{aligned}
\Lambda_i &= |g_\gamma(-\tau(f_{\eta,U,V}(x_i), y_i)) - g_\gamma(-\tau(f_{\eta',U',V'}(x_i), y_i))| \\
&\leq \frac{1}{\gamma}\,|\tau(f_{\eta,U,V}(x_i), y_i)) - \tau(f_{\eta',U',V'}(x_i), y_i))| && \text{(since } g_\gamma \text{ is } 1/\gamma\text{-Lipschitz)} \\
&= \frac{1}{\gamma}\,|(2f_{\beta,\theta}(x_i) - 1)y_i - (2f_{\beta',\theta'}(x_i) - 1)y_i)| \\
&\leq \frac{2}{\gamma}\,|y_i|\,|f_{\eta,U,V}(x_i) - f_{\eta',U',V'}(x_i)| && \text{(by Cauchy-Schwarz inequality)} \\
&\leq \frac{2}{\gamma}\left|\sigma(h_i^L(\eta,U,V)[0]) - \sigma(h_i^L(\eta',U',V')[0])\right| && \text{(since } y_i \in \{-1,1\}\text{)} \\
&\leq \frac{2}{\gamma}\left|h_i^L(\eta,U,V)[0] - h_i^L(\eta',U',V')[0]\right| && \text{(since } \sigma \text{is 1-Lipschitz)} \\
&\leq \frac{2}{\gamma}\,(k_1 + k_2|\eta-\eta'| + k_3\|U-U'\|)\,\frac{k_4^L - k_4}{k_4 - 1} + k_4C_z\|U-U'\|.
\end{aligned}$$

□

Lemma E.5. *The change in margin loss Λ_i for each node can be bounded by ϵ, using a covering of size P, where P depends on ϵ, with*

$$\log P \leq (d^2+1)\log\left(\frac{8\max\{A, BC_U\sqrt{d}\}}{\epsilon}\right).$$

Proof. We let $A = \frac{2k_2(k_4^L - k_4)}{k(k_4-1)}$ and $B = \frac{2k_3(k_4^L-k_4)+\gamma(k_4^2-k_4)C_z}{\gamma(k_4-1)}$ for simplicity of notation. Note that we have $\Lambda_i \leq A|\eta-\eta'| + B\|U-U'\|$.

We begin by noting that we can find a covering $\mathcal{C}(\eta, \frac{\epsilon}{2A}, |\cdot|)$ of size

$$\mathcal{N}(\eta, \frac{\epsilon}{2A}, |\cdot|) \leq 1 + \frac{4A}{\epsilon}.$$

We can also find a covering $\mathcal{C}(U, \frac{\epsilon}{2B}, \|\cdot\|_F)$ with size

$$\mathcal{N}(U, \frac{\epsilon}{2B}, \|\cdot\|_F) \leq \left(1 + \frac{4BC_U\sqrt{d}}{\epsilon}\right)^{d^2}.$$

For any specified ϵ, we can ensure that Λ_i is at most ϵ with a covering number of

$$\begin{aligned} P \leq & \mathcal{N}(\eta, \frac{\epsilon}{2A}, |\cdot|) \cdot \mathcal{N}(U, \frac{\epsilon}{2B}, \|\cdot\|_F) \\ & \leq \left(1 + \frac{4A}{\epsilon}\right)\left(1 + \frac{4BC_U\sqrt{d}}{\epsilon}\right)^{d^2} \leq (1 + \frac{4\max\{A, BC_U\sqrt{d}\}}{\epsilon})^{d^2+1} \end{aligned}$$

Moreover, when $\epsilon \leq 4\max\{A, BC_U\sqrt{d}\}$, we have

$$\log P \leq (d^2+1)\log\left(\frac{8\max\{A, BC_U\sqrt{d}\}}{\epsilon}\right).$$

□

Now we can finish our proof for Theorem 4.1.

Proof. Using Lemma A.5 from (Bartlett et al., 2017), we obtain that

$$\hat{R}_{\mathcal{T}}(\mathcal{H}^{\gamma}_{(\eta,U,V))}) \leq \inf_{\alpha>0}\left(\frac{4\alpha}{\sqrt{m}} + \frac{12}{m}\int_{\alpha}^{\sqrt{m}} \sqrt{\log \mathcal{N}(\mathcal{H}^{\gamma}_{(\eta,U,V))}, \epsilon, \|\cdot\|)}d\epsilon\right).$$

Using the previous lemmas, we have

$$\begin{aligned} \int_{\alpha}^{\sqrt{m}} \sqrt{\log \mathcal{N}(\mathcal{H}^{\gamma}_{(\eta,U,V))}, \epsilon, \|\cdot\|)}d\epsilon &= \int_{\alpha}^{\sqrt{m}} \sqrt{\log P}d\epsilon \\ &\leq \int_{\alpha}^{\sqrt{m}} \sqrt{(d^2+1)\log\left(\frac{8\max\{A, BC_U\sqrt{d}\}}{\epsilon}\right)}d\epsilon \\ &\leq \sqrt{m}\sqrt{(d^2+1)\log\left(\frac{8\max\{A, BC_U\sqrt{d}\}}{\alpha}\right)} \end{aligned}$$

Plugging in $\alpha = \sqrt{\frac{1}{m}}$, we have

$$\hat{R}_{\mathcal{T}}(\mathcal{H}^{\gamma}_{(\eta,U,V))}) \leq \frac{4}{m} + \frac{12\sqrt{(d^2+1)\log\left(8\sqrt{m}\max\{A, BC_U\sqrt{d}\}\right)}}{\sqrt{m}}.$$

□

F. Proofs in Section 4.2

First, we can bound the difference in the feature of the two nodes as follows:

$$\begin{aligned} & \|h_u^{\ell+1} - h_v^{\ell+1}\| \\ =& \|\sigma\left(\sum_{j\in\mathcal{N}_u}\left(\eta\cdot e_{uj}^{\ell} + (1-\eta)\cdot\frac{1}{\sqrt{d_u d_j}}\right)U^{\ell}h_j^{\ell}\right) - \sigma\left(\sum_{j\in\mathcal{N}_v}\left(\eta\cdot e_{vj}^{\ell} + (1-\eta)\cdot\frac{1}{\sqrt{d_v d_j}}\right)U^{\ell}h_j^{\ell}\right)\| \\ \leq& \|\left(\sum_{j\in\mathcal{N}_u}\left(\eta\cdot e_{uj}^{\ell} + (1-\eta)\cdot\frac{1}{\sqrt{d_u d_j}}\right)U^{\ell}h_j^{\ell}\right) - \left(\sum_{j\in\mathcal{N}_v}\left(\eta\cdot e_{vj}^{\ell} + (1-\eta)\cdot\frac{1}{\sqrt{d_v d_j}}\right)U^{\ell}h_j^{\ell}\right)\|. \ (\sigma \text{ is 1-Lipschitz}) \end{aligned}$$

F.1. Proof for Toy Example 1

$$
\begin{aligned}
&\|h_u^{\ell+1} - h_v^{\ell+1}\| \\
\leq &\| \left(\sum_{j\in\mathcal{N}_u} \left(\eta \cdot e_{uj}^\ell + (1-\eta)\cdot \frac{1}{\sqrt{d_u d_j}} \right) U^\ell h_j^\ell \right) - \left(\sum_{j\in\mathcal{N}_v} \left(\eta \cdot e_{vj}^\ell + (1-\eta)\cdot \frac{1}{\sqrt{d_v d_j}} \right) U^\ell h_j^\ell \right) \| \\
\leq &\| \sum_{j\in\mathcal{N}_u} \left(\eta \cdot e_{uj}^\ell + (1-\eta)\cdot \frac{1}{\sqrt{d_u d_j}} \right) - \sum_{j\in\mathcal{N}_v} \left(\eta \cdot e_{vj}^\ell + (1-\eta)\cdot \frac{1}{\sqrt{d_v d_j}} \right) \| \|U^\ell \bar{h}\| + C_U\|\epsilon\| \\
&\qquad\qquad \text{(Triangle Inequality and Cauchy-Schwartz Inequality)} \\
\leq &C_U\|\bar{h}\|\eta \cdot \| \sum_{j\in\mathcal{N}_u} e_{uj}^\ell - \sum_{j\in\mathcal{N}_v} e_{vj}^\ell \| + C_U\|\bar{h}\|(1-\eta)\cdot \| \sum_{j\in\mathcal{N}_u} \frac{1}{\sqrt{d_u d_j}} - \sum_{j\in\mathcal{N}_v} \frac{1}{\sqrt{d_v d_j}} \| + C_U\|\epsilon\| \\
= &C_U\|\bar{h}\|(1-\eta)\cdot \| \sum_{j\in\mathcal{N}_u} \frac{1}{\sqrt{d_u d_j}} - \sum_{j\in\mathcal{N}_v} \frac{1}{\sqrt{d_v d_j}} \| + C_U\|\epsilon\| \qquad \text{(because } \textstyle\sum_{j\in\mathcal{N}_u} e_{uj}^\ell = \sum_{j\in\mathcal{N}_v} e_{vj}^\ell = 1\text{)}
\end{aligned}
$$

F.2. Proof for Toy Example 2

$$
\begin{aligned}
&\|h_u^{\ell+1} - h_v^{\ell+1}\| \\
\leq &\| \left(\sum_{j\in\mathcal{N}_u} \left(\eta \cdot e_{uj}^\ell + (1-\eta)\cdot \frac{1}{\sqrt{d_u d_j}} \right) U^\ell h_j^\ell \right) - \left(\sum_{j\in\mathcal{N}_v} \left(\eta \cdot e_{vj}^\ell + (1-\eta)\cdot \frac{1}{\sqrt{d_v d_j}} \right) U^\ell h_j^\ell \right) \| \\
\leq &\| \sum_{j\in\mathcal{N}_u} \left(\eta \cdot e_{uj}^\ell + (1-\eta)\cdot \frac{1}{\sqrt{d_u d_j}} \right) U^\ell h_j^\ell - \left(\eta \cdot e_{v\pi(j)}^\ell + (1-\eta)\cdot \frac{1}{\sqrt{d_v d_{\pi(j)}}} \right) U^\ell h_{\pi(j)}^\ell \| \quad \text{(By assumption on } \pi\text{)} \\
\leq &\eta \cdot \| \sum_{j\in\mathcal{N}_u} (e_{uj}^\ell U^\ell h_j^\ell - e_{v\pi(j)}^\ell U^\ell h_{\pi(j)}^\ell) \| + (1-\eta)\cdot \| \sum_{j\in\mathcal{N}_u} (\frac{1}{\sqrt{d_u d_j}} U^\ell h_j^\ell - \frac{1}{\sqrt{d_v d_{\pi(j)}}} U^\ell h_{\pi(j)}^\ell) \| \\
&\qquad\qquad \text{(Triangle Inequality)}
\end{aligned}
$$

We can further simplify the second term as

$$
\| \sum_{j\in\mathcal{N}_u} (\frac{1}{\sqrt{d_u d_j}} U^\ell h_j^\ell - \frac{1}{\sqrt{d_v d_{\pi(j)}}} U^\ell h_{\pi(j)}^\ell) \| = \| \sum_{j\in\mathcal{N}_u} \frac{1}{\sqrt{d_u d_j}} U^\ell (h_j^\ell - h_{\pi(j)}^\ell) \| \leq \frac{C_U\|\epsilon\|}{C_{dl}}
$$

G. Additional Experiment Details

G.1. Experiment Setup for GCAN

We apply dropout with a probability of 0.4 for all learnable parameters, apply 1 head of the specialized attention layer (with new update rule), and then an out attention layer. The activation we choose is eLU activation (following prior work (Veličković, Petar et al., 2018)), with 8 hidden units, and 3 attention heads. We start training with an initial learning rate of 7×10^{-5} and a weight decay of 5×10^{-4}.

These GCAN interpolation experiments are all run with only 20% of the dataset being labeled datapoints, and the remaining 80% representing the unlabeled datapoints that we test our classification accuracy on. Table 2 notes the exact setup of each dataset, and the overall training time of each experiment.

In Figure 1, we plot the accuracy of GCAN on different η values, visualizing the result of Table 1.

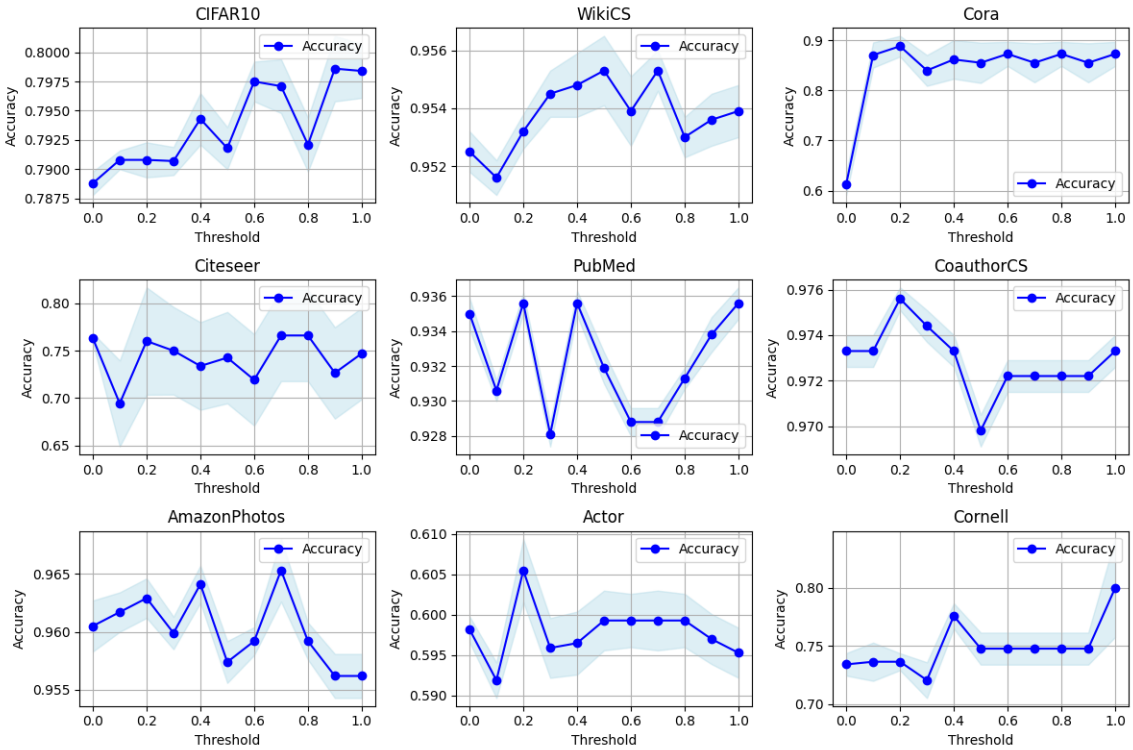

Figure 1. Accuracies and confidence intervals of the proposed GCAN interpolation method on different datasets. "Threshold" on the x-axis refers to the η hyperparameter in the GCAN interpolation method.

Dataset	Num of training nodes	Learning rate	Epoch	Num of exp	Training time(sec)	Dim of hidden layers	Num of Attention Heads
CiFAR10	400	7e-3	1000	30	13.5354	1	3
WikiCS	192	7e-3	1000	30	6.4742	1	3
Cora	170	7e-3	1000	30	7.4527	1	3
Citeseer	400	7e-3	1000	30	6.4957	1	3
Pubmed	400	7e-3	1000	30	13.1791	1	3
CoAuthor CS	400	0.01	1000	30	6.8015	1	3
Amazon Photos	411	0.01	400	30	11.0201	1	3
Actor	438	0.01	1000	30	14.7753	1	3
Cornell	10	0.01	1000	30	6.9423	1	3
Wisconsin	16	0.01	1000	30	6.9271	1	3

Table 2. Experiment setup