IMPLICIT VS UNFOLDED GRAPH NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

It has been observed that graph neural networks (GNN) sometimes struggle to maintain a healthy balance between the efficient modeling long-range dependencies across nodes while avoiding unintended consequences such oversmoothed node representations or sensitivity to spurious edges. To address this issue (among other things), two separate strategies have recently been proposed, namely *implicit* and *unfolded* GNNs. The former treats node representations as the fixed points of a deep equilibrium model that can efficiently facilitate arbitrary implicit propagation across the graph with a fixed memory footprint. In contrast, the latter involves treating graph propagation as unfolded descent iterations as applied to some graph-regularized energy function. While motivated differently, in this paper we carefully quantify explicit situations where the solutions they produce are equivalent and others where their properties sharply diverge. This includes the analysis of convergence, representational capacity, and interpretability. In support of this analysis, we also provide empirical head-to-head comparisons across multiple synthetic and public real-world node classification benchmarks.

1 INTRODUCTION

Given graph data with node features, graph neural networks (GNNs) represent an effective way of exploiting relationships among these features to predict labeled quantities of interest, e.g., node classification (Wu et al., 2020; Zhou et al., 2018). In particular, each layer of a message-passing GNN is constructed by bundling a graph propagation step with an aggregation function such that information can be shared between neighboring nodes to an extent determined by network depth. (Kipf & Welling, 2017; Hamilton et al., 2017; Kearnes et al., 2016; Velickovic et al., 2018).

For sparsely-labeled graphs, or graphs with entity relations reflecting long-range dependencies, it is often desirable to propagate signals arbitrary distances across nodes, but without expensive memory requirements during training, oversmoothing effects (Oono & Suzuki, 2020; Li et al., 2018), or disproportionate sensitivity to bad/spurious edges (Zhu et al., 2020b;a; Zügner et al., 2019; Zügner & Günnemann, 2019). To address these issues, at least in part, two distinct strategies have recently been proposed. First, the framework of implicit deep learning (Bai et al., 2019; El Ghaoui et al., 2020) has be applied to producing supervised node embeddings that satisfy an equilibrium criteria instantiated through a graph-dependent fixed-point equation. The resulting so-called *implicit* GNN (IGNN) pipeline (Gu et al., 2020), and related antecedents (Dai et al., 2018; Gallicchio & Micheli, 2020), mimics the behavior of a GNN model with arbitrary depth for handling long-range dependencies, but is nonetheless trainable via a fixed memory budget and robust against oversmoothing effects.

Secondly, *unfolded* GNN (UGNN) architectures are formed via graph propagation layers patterned after the unfolded descent iterations of some graph-regularized energy function (Chen & Eldar, 2021; Liu et al., 2021; Ma et al., 2020; Pan et al., 2021; Zhang et al., 2020; Zhu et al., 2021). In this way, the node embeddings at each UGNN layer can be interpreted as increasingly refined approximations to an interpretable energy minimizer, which can be nested within a bi-level optimization framework (Wang et al., 2016) for supervised training akin to IGNN. More broadly, similar unfolding strategies have previously been adopted for designing a variety of new deep architectures or interpreting existing ones (Gregor & LeCun, 2010; Hershey et al., 2014; Sprechmann et al., 2015; He et al., 2017).

As such, IGNN and UGNN are closely related in the sense that their node embeddings are intentionally aligned with a meta-criterion: either a fixed-point for IGNN or an energy minimizer for UGNN, where in both cases a useful inductive bias is introduced to address similar desiderata. And yet despite these commonalities, there has thus far been no systematic examination of the meaningful similarities and differences. We take a first step in this direction via the following workflow. First, after introducing the basic setup and notation in Section 2, we overview the IGNN framework in Section 3, including its attractive reliance on memory-efficient implicit differentiation and existing convergence results. Next, Section 4 introduces a general-purpose UGNN framework that encompasses a variety of existing unfolded models as special cases, including models with graph attention, broad graph propagation operators, and nonlinear activation functions. Sections 5 and 6 provide comparative analysis of the relative strengths and weaknesses of IGNN and UGNN models with respect to convergence guarantees and model expressiveness. We conclude with empirical comparisons in Section 7. Overall, our contributions can be summarized as follows:

- Although the original conceptions are unique, we consider a sufficiently broad design space of IGNN and UGNN models such that practically-relevant, interpretable regions of exact overlap can be established, analyzed, and contrasted with key areas of nonconformity.
- Within this framework, we compare the relative ability of IGNNs and UGNNs to converge to optimal (or locally-optimal) solutions per various model-specific design criteria. Among other things, this analysis reveals that IGNNs enjoy an advantage in that (unlike UGNNs) their implicit layers do not unavoidably induce symmetric propagation weights. In contrast, we show that UGNNs are more flexible by accommodating a broader class of robust nonlinear graph propagation operators while still guaranteeing at least local convergence.
- We investigate the consequences of symmetric (layer-tied) UGNN propagation weights. In particular, we prove that with linear activations, a UGNN can reproduce any IGNN representation, and define sufficient conditions for equivalency in broader regimes. We also show that UGNN layers with symmetric, layer-tied weights can exactly mimic arbitrary graph convolutional network (GCN) models (Kipf & Welling, 2017) characterized by asymmetric, layer-specific weights without introducing any additional parameters or significant complexity. Collectively, these results suggest that the weight symmetry enforced by UGNN models may not be overly restrictive in practice.
- Empirically, we provide the comprehensive, head-to-head comparisons between equivalently-sized IGNN and UGNN models while evaluating computational complexity and memory costs. These results also serve to complement our analytical findings.

2 Basic Setup

Consider a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, with $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges. We define $L \in \mathbb{R}^{n \times n}$ as the Laplacian of $\mathcal{G}$, meaning $L = D - A = B^\top B$, where D and A are degree and adjacency matrices respectively, while $B \in \mathbb{R}^{m \times n}$ is an incidence matrix. We also let $\widetilde{A} = A + I$ (i.e., A with self loops) and denote $\widetilde{D}$ as the corresponding degree matrix.

Both IGNNs and UGNNs incorporate graph structure via optimized embeddings $Y^* \in \mathbb{R}^{n \times d}$ that are a function of some adjustable weights $\mathcal{W}$, i.e., $Y^* \equiv Y^*(\mathcal{W})$ (for simplicity of notation, we will frequently omit including this dependency on $\mathcal{W}$), where by design $\partial Y^*(\mathcal{W})/\partial \mathcal{W}$ is computable, either implicitly (IGNN) or explicitly (UGNN). We may then insert these embeddings within an application-specific meta-loss given by

$$\ell_\theta\left(\theta, \mathcal{W}\right) \triangleq \sum_{i=1}^{n'} \mathcal{D}\Big(g\left[\boldsymbol{y}_i^*(\mathcal{W}); \theta\right], \boldsymbol{t}_i\Big), \tag{1}$$

where $g : \mathbb{R}^d \to \mathbb{R}^c$ is some differentiable node-wise function with parameters θ and c-dimensional output tasked with predicting ground-truth node-wise targets $\boldsymbol{t}_i \in \mathbb{R}^c$. Additionally, $\boldsymbol{y}_i^*(\mathcal{W})$ is the i-th row of $Y^*(\mathcal{W})$, $n' < n$ is the number of labeled nodes (we assume w.l.o.g. that the first n' nodes are labeled), and $\mathcal{D}$ is a discriminator function, e.g., cross-entropy for classification, squared error for regression, etc. Given that $Y^*(\mathcal{W})$ is differentiable by construction, we can optimize $\ell_\theta\left(\theta, \mathcal{W}\right)$ via gradient descent to obtain our final predictive model.

At a conceptual level then, the only difference between IGNNs and UGNNs is in how the corresponding optimal embeddings $Y^*(\mathcal{W})$ are motivated and computed. We introduce the specifics of each option in the following two sections.

3 Overview of Implicit GNNs

IGNN models are predicated on the fixed-point update

$$Y^{(k+1)} = \sigma\left[PY^{(k)}W_p + f\left(X; W_x\right)\right], \tag{2}$$

where $Y^{(k)}$ are node embeddings after the k-th iteration, $\mathcal{W} = \{W_p, W_x\}$ is a set of two weight matrices, $P \in \mathbb{R}^{n\times n}$ is an arbitrary graph propagation operator (e.g., $P = A$), and σ is a nonlinear activation function. Additionally, $f : \mathbb{R}^{n\times d_0} \to \mathbb{R}^{n\times d}$ is a base predictor function that converts the initial d_0-dimensional node features $X \in \mathbb{R}^{n\times d_0}$ into d-dimensional candidate embeddings, e.g., $f(X; W_x) = XW_x$ or $f(X; W_x) = \text{MLP}[X; W_x]$.

Now assume that σ is a differentiable component-wise non-expansive (CONE) mapping as specified in (Gu et al., 2020). This condition stipulates that σ applies the same function individually to each element of its input, and that this function satisfies $\|x - y\| \geq \|\sigma(x) - \sigma(y)\|$ for all $\{x, y\} \in \mathbb{R}^2$ (with some abuse of notation, we will overload the definition of σ when the meaning is clear from context). Furthermore, we assume that the weights W_p are such that $\lambda_{\text{pf}}(|P \otimes W_p|) < 1$, where $|\cdot|$ denotes the element-wise absolute value and λ_{pf} refers to the Peron-Frobenius eigenvalue.[1]

Under these conditions, it has been shown in (Gu et al., 2020) that $\lim_{k\to\infty} Y^{(k)} = Y^*$, where Y^* satisfies the fixed-point equation $Y^* = \sigma[PY^*W_p + f(X; W_x)]$. Therefore, as an IGNN forward pass we can iterate (2) K times, where K is sufficiently large, to compute node embeddings $Y^{(K)} \approx Y^*$ for use within the meta-loss from (1). For the IGNN backward pass, implicit differentiation methods (Bai et al., 2019; El Ghaoui et al., 2020), carefully tailored for handling graph-based models (Gu et al., 2020), can but used to compute gradients of Y^* with respect to W_p and W_x. Critically, this implicit technique does *not* require storing each intermediate representation $\{Y^{(k)}\}_{k=1}^{K}$ from the forward pass, and hence is quite memory efficient even if K is arbitrarily large. As K can be viewed as quantifying the extent of signal propagation across the graph, IGNNs can naturally capture long-range dependencies with a K-independent memory budget unlike more traditional techniques.

4 A General-Purpose Unfolded GNN Framework

In this section we will first introduce a general-purpose energy function that, when reduced with simplifying assumptions, decomposes into various interpretable special cases that have been proposed in the literature, both as the basis for new UGNN models as well as motivation for canonical GNN architectures. Later, will describe the generic proximal gradient descent iterations designed to optimize this energy. By construction, these iterations unfold in one-to-one correspondence with the UGNN model layers of each respective architecture under consideration.

4.1 Flexible Energy Function Design

Unlike IGNN models, the starting point of UGNNs is an energy function. In order to accommodate a broad variety of existing graph-regularized objectives and ultimately GNN architectures as special cases, we propose the rather general form

$$\ell_Y(Y; \mathcal{W}, f, \rho, \widetilde{B}, \phi) \triangleq \|Y - f(X; W_x)\|^2_{W_f} + \sum_{i=1}^{m} \rho\left(\left[\widetilde{B}YW_pY^\top\widetilde{B}^\top\right]_{ii}\right) + \sum_{i=1}^{n} \phi(\boldsymbol{y}_i; W_\phi), \tag{3}$$

where $\mathcal{W} = \{W_x, W_f, W_p, W_\phi\}$ is a set of four weight matrices, $\rho : \mathbb{R} \to \mathbb{R}$ is a differentiable function, and $\widetilde{B} \in \mathbb{R}^{m\times n}$ is a function of A that can be viewed as a generalized incidence matrix. Furthermore, $\phi : \mathbb{R}^d \to \mathbb{R}$ is an arbitrary function (possibly non-smooth) of the node embeddings that is bounded from below, and the weighted norm $\|\cdot\|_W$ is defined such that $\|X\|^2_W = \text{tr}[X^\top WX]$. The above loss is composed of three terms: (i) A quadratic penalty on deviations of the embeddings Y from the node-wise base model $f(X; W_x)$, (ii) a (possibly) non-convex graph-aware penalty that favors smoothness across edges (see illustrative special cases below), and (iii) an arbitrary function of each embedding that is independent of both the graph structure A and the original node features X.

As a representative special case, if $\widetilde{B} = B$, then the second term from (3) satisfies

$$\sum_{i=1}^{m} \rho\left(\left[\widetilde{B}YW_pY^\top\widetilde{B}^\top\right]_{ii}\right) = \sum_{\{i,j\}\in\mathcal{E}} \rho\left(\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2_{W_p}\right), \tag{4}$$

which penalizes deviations between the embeddings of two nodes sharing an edge. In particular, if ρ is chosen to be concave and non-decreasing on $[0, \infty)$, the resulting regularization favors robustness

[1]The Peron-Frobenius (PF) eigenvalue is a real, non-negative eigenvalue that exists for all square, non-negative matrices and produces the largest modulus.

to bad edges, and could be useful for handling heterophily graphs or adversarial attacks (Yang et al., 2021). And if in addition to the assumption from (4), ϕ is set to zero, ρ is an identity mapping, $W_f = I$, and $W_p = \lambda I$ with $\lambda > 0$, then (3) collapses to the more familiar loss

$$\ell_Y(Y; W_x, f) \triangleq \|Y - f(X; W_x)\|_{\mathcal{F}}^2 + \lambda \text{tr}\left[Y^\top L Y\right], \tag{5}$$

as originally proposed in (Zhou et al., 2004) to smooth base predictions from $f(X; W_x)$ using a quadratic, graph-aware regularization factor tr $\left[Y^\top L Y\right] = \sum_{\{i,j\}\in\mathcal{E}} \left\|\boldsymbol{y}_i - \boldsymbol{y}_j\right\|^2$. This construction has also formed the basis of a wide variety of work linking different GNN architectures (Ma et al., 2020; Pan et al., 2021; Zhang et al., 2020; Zhu et al., 2021); more details to follow in Section 4.2.

Similarly, if we allow for broader choices of $\widetilde{B} \neq B$ with $\widetilde{L} \triangleq \widetilde{B}^\top \widetilde{B} = \pi(A)$ for some function $\pi : \mathbb{S}^n \to \mathbb{S}^n$ over the space of n-dimensional PSD matrices $\mathbb{S}^n$, then (5) can be generalized by swapping L for this $\widetilde{L}$ as proposed in (Ioannidis et al., 2018). Candidates for $\widetilde{L}$ include normalized graph Laplacians (Von Luxburg, 2007) and various diffusion kernels (Klicpera et al., 2019b) designed according to application-specific requirements, e.g., graph signal denoising.

4.2 Descent Iterations That Form UGNN Layers

We now derive descent iterations for the general loss from the previous section (and later more interpretable special cases) that will be mapped to UGNN layers. For this purpose, let $U^{(k)}$ denote a single gradient descent step along (3), excluding the possibly non-differentiable ϕ-dependent term, and evaluated at some candidate point $Y^{(k)}$. We may compute such an abridged gradient step as

$$U^{(k)} = Y^{(k)} - \alpha\left[\widetilde{B}^\top \Gamma^{(k)} \widetilde{B} Y^{(k)}\left(W_p + W_p^\top\right) + Y^{(k)}\left(W_f + W_f^\top\right) - f(X; W)\right], \tag{6}$$

where α is the step size and $\Gamma^{(k)}$ is a diagonal matrix with i-th diagonal element given by

$$\gamma_i^{(k)} = \left.\frac{\partial \rho(z)}{\partial z}\right|_{z = \text{diag}\left[\widetilde{B} Y^{(k)} W_p \left(Y^{(k)}\right)^\top \widetilde{B}^\top\right]_i}. \tag{7}$$

We then have the following:

Lemma 4.1 *If ρ has Lipschitz continuous gradients, then the proximal gradient update*

$$Y^{(k+1)} = \text{prox}_\phi\left(U^{(k)}\right) \triangleq \arg\min_Y \left[\frac{1}{2\alpha}\|U^{(k)} - Y\|_{\mathcal{F}}^2 + \sum_i \phi(\boldsymbol{y}_i; W_\phi)\right], \tag{8}$$

is guaranteed to satisfy $\ell_Y(Y^{(k+1)}; \mathcal{W}, f, \rho, \widetilde{B}, \phi) \leq \ell_Y(Y^{(k)}; \mathcal{W}, f, \rho, \widetilde{B}, \phi)$ for any $\alpha \in (0, 1/\mathcal{L}]$, where $\mathcal{L}$ is the Lipschitz constant for gradients of (3) w.r.t. Y, excluding the non-smooth ϕ term.

The function $\text{prox}_\phi : \mathbb{R}^d \to \mathbb{R}^d$ is known as the *proximal operator*[2] associated with ϕ (separably extended across all n nodes in the graph), and the proof follows from basic properties of gradient descent (Bertsekas, 1999) and proximal methods (Combettes & Pesquet, 2011); see Appendix.

Generally speaking, the mapping $Y^{(k)} \mapsto U^{(k)}$ from (6) can be viewed as a (possibly nonlinear) graph filter, while prox_ϕ from (8) serves as an activation function applied to the embedding of each node. Collectively then, $Y^{(k+1)} = \text{prox}_\phi\left(U^{(k)}\right)$ provides a flexible template for UGNN layers that naturally reduces to common GNN architectures per various design choices. And analogous to IGNN, we can execute K UGNN steps to approximate some Y^*, which could be a fixed-point, stationary point, or global optimum; more on this in Sections 4.3 and 5 below.

For example, consider the selection $\sum_i \phi(\boldsymbol{y}_i; W_\phi) = \sum_{i,j} \mathcal{I}_\infty[y_{ij} < 0]$, where y_{ij} is the (i,j)-th element of Y and $\mathcal{I}_\infty$ is an indicator function that assigns an infinite penalty to any $y_{ij} < 0$. The proximal operator then becomes $\text{prox}_\phi(U) = \text{ReLU}(U)$, i.e., a ReLU function that element-wise sets negative values to zero. If we add this ϕ-dependent term to (5) (as a special case of (3)), the resulting update becomes

$$Y^{(k+1)} = \text{prox}_\phi\left(U^{(k)}\right) = \text{ReLU}\left(Y^{(k)} - \alpha\left[(\lambda L + I) Y^{(k)} - f(X; W)\right]\right). \tag{9}$$

[2]If (8) happens to have multiple solutions, then the proximal operator selects one of them so as to remain a proper deterministic function of its argument, which is relevant for forming later connections with IGNN.

And based on observations from (Ma et al., 2020; Pan et al., 2021; Zhang et al., 2020; Zhu et al., 2021), when we initialize with $Y^{(0)} = f(X; W) = XW$ and apply simple reparameterizations, the first iteration reduces to the canonical GCN layer $Y^{(1)} = \text{ReLU}\left[\widetilde{D}^{-1/2}\widetilde{A}\widetilde{D}^{-1/2}XW\right]$ from (Kipf & Welling, 2017). Subsequent iterations are no longer equivalent, although the inherent differences (e.g., the skip connections from the input layer) are useful for avoiding oversmoothing effects that can at times hamper deep GCN models (Oono & Suzuki, 2020; Li et al., 2018; Rong et al., 2020).

Additionally, the widely-used APPNP architecture (Klicpera et al., 2019a) is also a special case of (9) when the ReLU operator is removed (i.e., ϕ is zero), $\alpha = \frac{1}{1+\lambda}$, and L is changed to the symmetric normalized Laplacian (Von Luxburg, 2007). And as a final representative example, if we adopt a nonlinear choice for ρ, then L in (9) will be replaced by $L^{(k)} \triangleq B^\top \Gamma^{(k)} B$, where $\Gamma^{(k)}$ rescales graph edges. As discussed in (Yang et al., 2021), this instantiates a form of graph attention, whereby the attention weights produced by a concave, non-decreasing ρ add robustness to spurious edges.

4.3 UGNN Fixed Points and Connections with IGNN

While UGNN layers are formed from the descent steps of a graph-regularized energy, for a more direct comparison with IGNN, it is illuminating to consider the situation where the update $Y^{(k+1)} = \text{prox}_\phi\left(U^{(k)}\right)$ is iterated with k becoming sufficiently large. In this case, we may ideally reach a fixed point Y^* that satisfies

$$\begin{aligned} Y^* &= \text{prox}_\phi\left(Y^* - \alpha\left[\widetilde{B}^\top\Gamma^*\widetilde{B}Y^*\left(W_p + W_p^\top\right) + Y^*\left(W_f + W_f^\top\right) - f(X;W)\right]\right) \\ &= \text{prox}_\phi\left(Y^*\left[I - \alpha W_f^s - \alpha W_p^s\right] + \alpha\left[I - \widetilde{B}^\top\Gamma^*\widetilde{B}\right]Y^* W_p^s + \alpha f(X;W)\right), \end{aligned} \tag{10}$$

where $W_p^s \triangleq W_p + W_p^\top$ and $W_f^s \triangleq W_f + W_f^\top$ are symmetric weight matrices, and Γ^* is defined analogously to (7). It now follows that if $W_f^s = \frac{1}{\alpha}I - W_p^s$, $\alpha = 1$ and $\Gamma^* = I$ (i.e., ρ is an identity mapping), and we define the propagation operator as $P = I - \widetilde{B}^\top\widetilde{B} \equiv I - \widetilde{L}$, then (10) reduces to

$$Y^* = \text{prox}_\phi\left[PY^*W_p^s + f(X;W)\right]. \tag{11}$$

This expression is exactly equivalent to the IGNN fixed point from Section 3 when we set σ to the proximal operator of ϕ and we restrict W_p to be symmetric. For direct head-to-head comparisons then, it is instructive to consider what CONE activation functions σ can actually be expressed as the proximal operator of some penalty ϕ. In this regard we have the following:

Lemma 4.2 *A continuous CONE function $\sigma : \mathbb{R} \to \mathbb{R}$ can be expressed as the proximal operator of some function ϕ iff σ is also non-decreasing.*

The proof follows from results in (Gribonval & Nikolova, 2020). This result implies that, at least with respect to allowances for decreasing activation functions σ, IGNN is more flexible than UGNN.

However, the relative flexibility of IGNN vs UGNN fixed points becomes much more nuanced once we take into account convergence considerations. For example, the proximal gradient iterations from Section 4.2 do *not* require that prox_ϕ is non-expansive or CONE to guarantee descent, although it thus far remains ambiguous how this relates to obtainable fixed points of UGNN. To this end, in the next section we will evaluate when such fixed points exist, what properties they have, convergence conditions for reaching them, and importantly for present purposes, how UGNN fixed points relate to their IGNN counterparts.

5 Convergence Comparisons

In this section we first detail situations whereby convergence to a unique fixed point, that also may at times correspond with a unique UGNN global minimum, can be established. In these situations IGNN facilitates stronger guarantees in terms of broader choices for W_p and σ. Later, we examine alternative scenarios whereby UGNN has a distinct advantage with respect to convergence guarantees to potentially local solutions.

5.1 Convergence to Unique Global Solutions

To facilitate the most direct, head-to-head comparison between IGNN and UGNN, we consider a restricted form of the UGNN energy from (3). In particular, let

$$\ell_Y(Y;\mathcal{W},f,\phi) \triangleq \|Y - f(X;W_x)\|^2_{W_f} + \text{tr}\left[Y^\top \widetilde{L} Y W_p\right] + \sum_{i=1}^{n} \phi(\boldsymbol{y}_i), \tag{12}$$

which is obtainable from (3) when ρ is an identity mapping and ϕ is assumed to have no trainable parameters. We also define $\Sigma \triangleq I \otimes W_f^s + \widetilde{L} \otimes W_p^s$, with minimum and maximum eigenvalues given by $\lambda_{\text{min}}(\Sigma)$ and $\lambda_{\text{max}}(\Sigma)$ respectively.

Theorem 5.1 *If prox_ϕ is non-expansive and $\lambda_{min}(\Sigma) > 0$, then (12) has a unique global minimum. Additionally, starting from any initialization $Y^{(0)}$, the iterations $Y^{(k+1)} = prox_\phi\left[U^{(k)}\right]$ applied to (12) with $\alpha \in (0, 1/\lambda_{max}(\Sigma))$ are guaranteed to converge to this global minimum as $k \to \infty$.*

Corollary 5.1.1 *For any non-decreasing CONE function σ and symmetric $W_p \equiv W_p^s$ satisfying $\|W_p^s\|_2 < \|P\|_2^{-1}$, there will exist a function ϕ such that the fixed point of (2) is the unique global minimum of (12) when W_f is chosen such that $W_f^s = I - W_p^s$ and $P = I - \widetilde{L}$.*

Lemma 5.2 *For any non-expansive σ (not necessarily element-wise) and W_p (not necessarily symmetric) satisfying $\|W_p\|_2 < \|P\|_2^{-1}$, iterating (2) will converge to a unique fixed point.*[3]

Hence from Theorem 5.1 and Corollary 5.1.1, if we are willing to accept symmetric graph propagation weights W_p^s (and a non-decreasing σ), UGNN enjoys the same convergence guarantee as IGNN, but with the added benefit of an interpretable underlying energy function. In contrast, when W_p is *not* symmetric as in Lemma 5.2, we can only guarantee a unique IGNN fixed point, but we are no longer able to establish an association with a specific UGNN energy. In fact, it does not seem likely that any familiar/interpretable functional form can even possibly exist to underlie such a fixed point when W_p is not symmetric. This is largely because of the following simple result:

Lemma 5.3 *There does not exist any second-order smooth function $h : \mathbb{R}^{n\times d} \to \mathbb{R}$ such that $\partial h(Y;W)/\partial Y = YW$ for all (asymmetric) matrices $W \in \mathbb{R}^{d\times d}$ with $d > 1$ and $n \geq 1$.*

And by continuity arguments, a similar result will apply to many non-smooth functions. Consequently, with asymmetric weights there is no obvious way to associate fixed points with stationary points as we have done for UGNN.

5.2 Broader Convergence Regimes

As we move to alternative energy functions with nonlinear dependencies on the graph, e.g., ρ not equal to identity, meaningful (albeit possibly weaker) convergence properties for UGNN can still be established. In particular, we consider convergence to local minima, or more generally, stationary points of the underlying objective. However, because (3) may be non-convex and non-smooth, we must precisely define an applicable notion of stationarity.

While various possibilities exist, for present purposes we define Y^* as a stationary point of (3) if $\mathbf{0} \in \partial_{\mathcal{F}}\left[\ell_Y(Y^*;\mathcal{W},f,\rho,\widetilde{B},\phi)\right]$, where $\partial_{\mathcal{F}}$ denotes the Fréchet subdifferential. The latter generalizes the standard subdifferential as defined for convex functions, to non-convex cases. More formally, the Fréchet subdifferential (Li et al., 2020) of some function $h(Y)$ is defined as the set

$$\partial_{\mathcal{F}}[h(Y)] = \left\{S : h(Y) \geq h(Z) + \text{tr}\left[S^\top (Y - Z)\right] + o\left(\|Y - Z\|_{\mathcal{F}}\right)\ \ \forall Y\right\}, \tag{13}$$

which is equivalent to the gradient when h is differentiable and the regular subdifferential when h is convex. Based on this definition, we have the following:

Theorem 5.4 *Assume that W_p and W_f are PSD, ϕ is continuous with a non-empty Fenchel subdifferential for all Y*[4] *and ρ is a concave, non-decreasing function with Lipschitz-continuous gradients. Additionally, starting from any initialization $Y^{(0)}$, let $\left\{Y^{(k)}\right\}_{k=0}^{\infty}$ denote a sequence generated by $Y^{(k+1)} = \text{prox}_\phi\left(U^{(k)}\right)$ with step size parameter $\alpha \in (0, 1/\mathcal{L}]$. Then all accumulation*

[3] An analogous result has been shown in (Gu et al., 2020), but restricted to CONE mappings (not arbitrary non-expansive mappings) and with a dependency on the less familiar Peron-Frobenius eigenvalue; see Section 3.

[4] For minor technical reasons, we also assume ϕ is such that $\lim_{\|Y\|\to\infty} \ell_Y(Y;\mathcal{W},f,\rho,\widetilde{B},\phi) = \infty$.

points of $\{Y^{(k)}\}_{k=0}^{\infty}$ *are stationary points of (3). Furthermore,* $\lim_{k\to\infty} \ell_Y(Y^{(k)}; \mathcal{W}, f, \rho, \widetilde{B}, \phi) = \ell_Y(Y^*; \mathcal{W}, f, \rho, \widetilde{B}, \phi)$ *for some stationary point* Y^**.*

The proof is based on Zangwill's global convergence theorem (Luenberger, 1984) and additional results from (Sriperumbudur & Lanckriet, 2009); see Appendix for further details.

Corollary 5.4.1 *If in addition to the conditions from Theorem 5.4,* ϕ *is non-expansive and* ρ *is chosen such that the composite function* $\rho\left[(\cdot)^2\right]$ *is convex, then* $\lim_{k\to\infty} \ell_Y(Y^{(k)}; \mathcal{W}, f, \rho, \widetilde{B}, \phi) = \ell_Y(Y^*; \mathcal{W}, f, \rho, \widetilde{B}, \phi)$*, where* Y^* *is a global minimizer of (3).*

These results both apply to situations where the k-dependent UGNN graph propagation operator $P^{(k)} \triangleq I - \widetilde{B}^\top \Gamma^{(k)} \widetilde{B}$ is nonlinear by virtue of the $Y^{(k)}$-dependency of $\Gamma^{(k)}$, i.e., $P^{(k)} Y^{(k)} W_p^s \neq PY^{(k)} W_p^s$ for any fixed P. In contrast, it is unknown (and difficult to determine) if general nonlinear alternatives to PYW_p in (2) will converge.

5.3 Recap of Relative IGNN and UGNN Flexibility

In terms of model expressiveness, the advantage of the IGNN framework is two-fold: (i) it allows for asymmetric weights W_p while still providing a strong convergence guarantee, and (ii) it allows for decreasing activation functions. However, the latter is likely much less relevant in practice, as most deep models adopt some form of non-decreasing activation anyway, e.g., ReLU, etc.

In contrast, UGNN models are more flexible than IGNN in their accommodation of: (i) nonlinear graph propagation through the graph attention mechanism described in Section 4.2, and (ii) expansive proximal operators. While the latter may seem like a mere technicality, expansive proximal operators actually undergird a variety of popular sparsity shrinkage penalties, which have been proposed for integration with GNN models (Zheng et al., 2021). For example, the ℓ_0 norm and related non-convex regularization factors (Chen et al., 2017; Fan & Li, 2001) are expansive and can be useful for favoring parsimonious node embeddings. And overall, with only mild assumptions, UGNN at least guarantees cost function descent across a broad class of models per Lemma 4.1, with even convergence to stationary points possible in relevant situations from Theorem 5.4 that IGNN cannot emulate.

6 How Limiting Are Symmetric (Layer-Tied) Propagation Weights?

Previously we observed that the primary advantage IGNN has over UGNN, at least in terms of model expressiveness, is that IGNN places no restriction that the propagation weight matrix W_p need be symmetric, although both models ultimately involve an architecture with layer-tied weights unlike typical message-passing models. e.g. GCN. To better understand the implications of this distinction, we will now explore the expressiveness of GNN models with symmetric, layer-tied weights.

6.1 Fixed-Point Equivalency with Symmetric and Asymmetric Weights

In this section we examine situations whereby UGNN models are able to reproduce, up to some inconsequential transform, the fixed points of an IGNN, even when the latter includes an asymmetric propagation weight matrix W_p. However, because it is challenging to analyze general situations with arbitrary activation functions, we first consider the case where σ is an identity mapping; we also assume that $f(X; W_x) = XW_x$ as adopted in (Gu et al., 2020). We then have the following:

Theorem 6.1 *For any* $W_p \in \mathbb{R}^{d\times d}$*,* $W_x \in \mathbb{R}^{d_0 \times d}$*,* $X \in \mathbb{R}^{n\times d_0}$*, and* P *that admit a unique IGNN fixed point* $Y^* = PY^*W_p + XW_x$*, there exists a* $Y' \in \mathbb{R}^{n\times d'}$*,* $\tilde{W}_x \in \mathbb{R}^{d_0\times d'}$*, right-invertible transform* $T \in \mathbb{R}^{d\times d'}$*, and symmetric* $W_p^s \in \mathbb{R}^{d'\times d'}$*, such that*

$$Y'T = PY'TW_p^s + X\tilde{W}_x, \quad \textit{with} \ \|Y' - Y^*\| < \epsilon, \ \forall \epsilon > 0. \tag{14}$$

This result implies that an UGNN model with $\phi = 0$ can produce node-wise embeddings $Y = Y'T$ capable of matching any IGNN fixed-point with arbitrary precision up to some transform T. And given that T can be absorbed into the meta-loss output layer g from (1) (which is often a linear layer anyway), the distinction is negligible.

Proceeding further, if we allow for nonlinear activation functions σ, we may then consider a more general, parameterized family of penalty functions $\phi(\boldsymbol{y}; W_\phi)$ such that the resulting proximal operator increases our chances of finding a UGNN fixed point that aligns with IGNN. However, some care must be taken to control UGNN model capacity to reduce the possibility of trivial, degenerate alignments. To this end, we consider proximal operators in the set

$$\mathcal{S}_\sigma = \{\text{prox}_\phi : \boldsymbol{y} \mapsto G\sigma(C\boldsymbol{y}) | \text{with } \{G, C,\} \text{ chosen such that } \text{prox}_\phi \text{ is proximal operator.}\}, \quad (15)$$

where the matrices G and C can be aggregated into W_ϕ. We then derive sufficient conditions under which UGNN has optimal solutions equivalent to IGNN fixed points (it remains an open question if a necessary condition can be similarly derived). See Appendix for details.

6.2 UGNN Capacity to Match Canonical GCN Architectures

The previous section considered the alignment of fixed points, which are obtainable after executing potentially infinite graph propagation steps, from models with and without symmetric propagation weights. Somewhat differently, this section investigates analogous issues in the context of the possible embeddings obtainable after k steps of graph propagation. Specifically, we evaluate the expressiveness of a canonical GCN model with arbitrary, layer-independent weights versus a UGNN model structured so as to effectively maintain an equivalent capacity (up to a right-invertible linear transformation as discussed above).

Theorem 6.2 *Let $Y_{\text{GCN}}^{(k+1)} = \sigma\left(PY_{\text{GCN}}^{(k)}W_p^{(k)} + \beta Y_{\text{GCN}}^{(k)}\right)$ denote the k-th layer of a GCN, where $W_p^{(k)}$ is a layer-dependent weight matrix, $\sigma = \text{prox}_\phi$ for some function ϕ, and $\beta \in \{0, 1\}$ determines whether or not a residual connection is included. Then with $\alpha = 1$, ρ an identity mapping, and $f(X; W_x)$ set to zero, there will always exist a right-invertible T, initialization $Y^{(0)}$, and symmetric weights W_p^s and W_f^s such that the k-th iteration step computed via (8) satisfies $Y^{(k+1)}T = Y_{\text{GCN}}^{(k+1)}$.*

Given the close association between GCNs and certain UGNN models, Theorem 6.2 suggests that, at least in principle, symmetric layer-tied weights may not be prohibitively restrictive in the finite layer regime. And as can be observed from the proof, this equivalency result is possible using a matching parameter budget instantiated through an expanded hidden dimension but constrained via block-sparse weight matrices W_p^s and W_f^s. Of course in practice we cannot always guarantee that reliance on symmetric weights will not have unintended consequences that may in certain circumstances adversely impact performance.

7 Experiments

As prior work has already showcased the value of IGNN and UGNN models across various graph prediction tasks, in this section we narrow our attention to complementary experiments designed to elucidate some of the particular issues raised by our analysis in previous sections. To this end, we begin by exploring the interplay between the type of graph propagation weights (symmetric vs asymmetric), the graph propagation path length (finite as with UGNN vs approximately infinite as with IGNN), and model expressiveness. For this purpose we design the following quasi-synthetic experiment: First, we train separate IGNN and UGNN models on the ogbn-arxiv dataset, where the architectures are equivalent with the exception of W_p and the number of propagation steps (see Appendix for all network and training details). Additionally, because UGNN requires symmetric propagation weights, a matching parameter count can be achieved by simply expanding the UGNN hidden dimension. Once trained, we then generate predictions from each model and treat them as new, ground-truth labels. We next train four additional models separately on both sets of synthetic labels: (i) An IGNN with architecture equivalent to the original used for generating labels, (ii) an analogous UGNN, (iii) an IGNN with symmetric weights, and (iv) a UGNN with asymmetric weights (equivalent to IGNN with finite propagation steps). For all models the number of parameters is a small fraction of the number of labels to mitigate overfitting issues.

Results are presented in Table 1, where the columns indicate the label generating models, and the rows denote the recovery models. As expected, the highest recovery accuracy occurs when the generating and recovery models are matched such that perfect recovery is theoretically possible by design. We also observe that UGNN with asymmetric weights (denoted "UGNN/as") performs significantly worse recovering IGNN data, indicating that truncated propagation steps can reduce performance when the true model involves long-range propagation. Similarly, IGNN with symmetric weights (IGNN/s) struggles somewhat to recover IGNN labels, indicating that symmetric weights may not always be able to exactly mimic asymmetric ones across all practical settings, which is not surprising. IGNN/s is however reasonably effective at recovering UGNN data given that the later involves symmetric weights and finite propagation steps. In general though, the fact that the performance of all models varies within a range of a few percentage points suggests a significant overlap of model expressiveness as might be expected.

Next, in terms of time and memory consumption, we compare UGNN and IGNN on the Amazon Co-purchase benchmark, which has been advocated in (Gu et al., 2020) as a suitable data source for testing IGNN. Results are shown in Figures 1 and 2 based on executing 100 steps of training and evaluation on a single Tesla T4 GPU; see Appendix for further details. Clearly, IGNN maintains a huge advantage in terms of memory consumption, while UGNN has a faster runtime provided the number of propagation steps is not too large.

$\mathscr{G}$ / $\mathscr{R}$	IGNN	UGNN
IGNN	95.7 ± 0.9	91.0 ± 1.5
UGNN	89.8 ± 1.9	94.5 ± 0.8
IGNN/s	90.2	93.2
UGNN/as	91.6 ± 1.2	92.3 ± 0.7

Table 1: Accuracy recovering synthetic labels

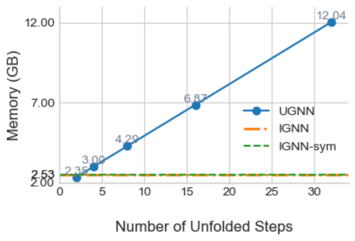

Figure 1: Memory cost

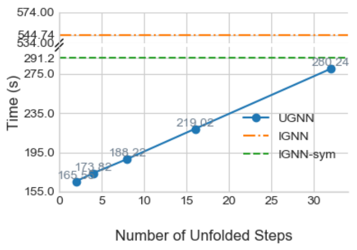

Figure 2: Running time

Finally, we compare the node classification accuracy of UGNN and IGNN models across a number of standard benchmarks in Table 2; see Appendix for dataset details and training protocols, etc. For these results, we do not enforce an equivalent architecture, but instead tune the models for the best accuracy. We also consider a variant of each model whereby the propagation matrix, either W_p or W_p^s, is set to identity and rescaled to ensure IGNN convergence (denoted "w/I" in the table). Overall, UGNN with $W_p^s = I$ performs the best, indicating that in practice symmetric weights may not be a hindrance. Furthermore, on the heterophily datasets Wisconsin, Texas, Actor, and Cornell (Pei et al., 2019) the nonlinear graph propagation capabilities of UGNN can be especially advantageous as has been previously discussed in (Yang et al., 2021). Note however that these commonly-used benchmarks (even Amazon Co-Purchase) may be somewhat inadequate for demonstrating the full effectiveness of truly long-range graph propagation as facilitated by IGNN, and determining the prevalence of such data in real-world graphs is a valuable direction for future work.

	Cora	Citeseer	Pubmed	Arxiv	Wisconsin	Texas	Actor	Cornell	Amazon
IGNN	80.9 ± 1.0	69.6 ± 1.0	78.7 ± 0.7	71.1 ± 0.2	83.1 ± 5.6	79.2 ± 6.0	35.5 ± 2.9	76.0 ± 6.2	84.4 ± 0.4
IGNN w/ I	**83.3 ± 0.7**	71.2 ± 0.8	79.4 ± 0.6	68.0 ± 1.2	63.4 ± 9.2	66.7 ± 8.7	33.2 ± 3.4	64.9 ± 4.7	87.8 ± 0.0
UGNN	81.6 ± 0.5	70.0 ± 0.9	78.9 ± 0.4	68.6 ± 0.2	81.2 ± 5.1	78.7 ± 7.0	36.4 ± 1.0	81.1 ± 7.3	80.0 ± 0.9
UGNN w/I	**83.3 ± 0.3**	**74.1 ± 0.5**	**80.7 ± 0.5**	**72.9 ± 0.2**	**86.7 ± 4.2**	**84.6 ± 3.8**	**37.4 ± 1.5**	**86.8 ± 5.1**	**89.1 ± 0.0**

Table 2: Accuracy results on node classification benchmarks.

8 Conclusions

This work has closely examined the relationship between IGNNs and UGNNs, shedding light on their similarities and differences. In this regard, representative take-home messages are as follows: (i) IGNN has a clear advantage when long-range propagation is required; however, standard node classification benchmarks are not generally adequate for fully showcasing this advantage (e.g., as evidenced by Table 2 results). (ii) UGNN symmetric propagation weights do not appear to be a significant hindrance, while UGNN nonlinear propagation operators can be advantageous relative to IGNN when unreliable edges are present. (iii) IGNN has a major advantage in memory costs, while UGNN sometimes has a practical advantage in complexity and accuracy (although with more refined benchmarks the latter may not always be true). (iv) Both models benefit from the inductive biases of their respective design criteria, which occasionally overlap but retain important areas of distinction.

REFERENCES

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. Deep equilibrium models. *arXiv preprint arXiv:1909.01377*, 2019.

Dimitri Bertsekas. *Nonlinear Programming*. Athena Scientific, 2nd edition, 1999.

Siheng Chen and Yonina C Eldar. Graph signal denoising via unrolling networks. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5290–5294, 2021.

Yichen Chen, Dongdong Ge, Mengdi Wang, Zizhuo Wang, Yinyu Ye, and Hao Yin. Strong NP-hardness for sparse optimization with concave penalty functions. In *International Confernece on Machine Learning*, 2017.

Patrick L Combettes and Jean-Christophe Pesquet. Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering*, pp. 185–212. Springer, 2011.

Hanjun Dai, Zornitsa Kozareva, Bo Dai, Alex Smola, and Le Song. Learning steady-states of iterative algorithms over graphs. In *International Conference on Machine Learning*, pp. 1106–1114, 2018.

Laurent El Ghaoui, Fangda Gu, Bertrand Travacca, Armin Askari, and Alicia Tsai. Implicit deep learning. *arXiv preprint arXiv:1908.06315*, 2020.

Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *JASTA*, 96(456):1348–1360, 2001.

Claudio Gallicchio and Alessio Micheli. Fast and deep graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 3898–3905, 2020.

Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. In *International Conference on Machine Learning*, 2010.

Rémi Gribonval and Mila Nikolova. A characterization of proximity operators. *Journal of Mathematical Imaging and Vision*, 62(6):773–789, 2020.

Fangda Gu, Heng Chang, Wenwu Zhu, Somayeh Sojoudi, and Laurent El Ghaoui. Implicit graph neural networks. In *Advances in Neural Information Processing Systems*, 2020.

William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 1025–1035, 2017.

Hao He, Bo Xin, Satoshi Ikehata, and David Wipf. From bayesian sparsity to gated recurrent nets. In *Advances in Neural Information Processing Systems*, 2017.

John Hershey, Jonathan Le Roux, and Felix Weninger. Deep unfolding: Model-based inspiration of novel deep architectures. *arXiv preprint arXiv:1409.2574*, 2014.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. 2020.

Vassilis N Ioannidis, Meng Ma, Athanasios N Nikolakopoulos, Georgios B Giannakis, and Daniel Romero. Kernel-based inference of functions on graphs. In D. Comminiello and J. Principe (eds.), *Adaptive Learning Methods for Nonlinear System Modeling*. Elsevier, 2018.

Steven M. Kearnes, Kevin McCloskey, Marc Berndl, Vijay S. Pande, and Patrick Riley. Molecular graph convolutions: moving beyond fingerprints. *J. Comput. Aided Mol. Des.*, 30(8):595–608, 2016.

D. Kinderlehrer and G. Stampacchia. *An Introduction to Variational Inequalities and Their Applications*. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104), 1980. ISBN 9780898719451.

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

Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *International Conference on Learning Representations*, 2019a.

Johannes Klicpera, Stefan Weißenberger, and Stephan Günnemann. Diffusion improves graph learning. In *Advances in Neural Information Processing Systems*, 2019b.

Jiajin Li, Anthony Man-Cho So, and Wing-Kin Ma. Understanding notions of stationarity in nonsmooth optimization: A guided tour of various constructions of subdifferential for nonsmooth functions. *IEEE Signal Processing Magazine*, 37(5):18–31, 2020.

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

Xiaorui Liu, Wei Jin, Yao Ma, Yaxin Li, Hua Liu, Yiqi Wang, Ming Yan, and Jiliang Tang. Elastic graph neural networks. In *International Conference on Machine Learning*, 2021.

D.G. Luenberger. *Linear and Nonlinear Programming*. Addison–Wesley, Reading, Massachusetts, second edition, 1984.

Yao Ma, Xiaorui Liu, Tong Zhao, Yozen Liu, Jiliang Tang, and Neil Shah. A unified view on graph neural networks as graph signal denoising. *arXiv preprint arXiv:2010.01777*, 2020.

Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *8th International Conference on Learning Representations, ICLR*, 2020.

Xuran Pan, Shiji Song, and Gao Huang. A unified framework for convolution-based graph neural networks, 2021. URL https://openreview.net/forum?id=zUMD--Fb9Bt.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. In *International Conference on Learning Representations*, 2019.

Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. In *8th International Conference on Learning Representations, ICLR*, 2020.

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

Pablo Sprechmann, Alex Bronstein, and Guillermo Sapiro. Learning efficient sparse and low rank models. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 37(9), 2015.

Bharath Sriperumbudur and Gert Lanckriet. On the convergence of the concave-convex procedure. In *Advances in Neural Information Processing Systems*, 2009.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR*, 2018.

Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

Zhangyang Wang, Qing Ling, and Thomas Huang. Learning deep ℓ_0 encoders. In *AAAI Conference on Artificial Intelligence*, volume 30, 2016.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

Yongyi Yang, Tang Liu, Yangkun Wang, Jinjing Zhou, Quan Gan, Zhewei Wei, Zheng Zhang, Zengfeng Huang, and David Wipf. Graph neural networks inspired by classical iterative algorithms. In *International Conference on Machine Learning*, 2021.

Hongwei Zhang, Tijin Yan, Zenjun Xie, Yuanqing Xia, and Yuan Zhang. Revisiting graph convolutional network on semi-supervised node classification from an optimization perspective. *CoRR*, abs/2009.11469, 2020.

Xuebin Zheng, Bingxin Zhou, Junbin Gao, Yu Guang Wang, Pietro Lio, Ming Li, and Guido Montúfar. How framelets enhance graph neural networks. In *International Conference on Machine Learning*, 2021.

Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. *Advances in Neural Information Processing Systems*, 2004.

Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *arXiv preprint arXiv:1812.08434*, 2018.

Jiong Zhu, Ryan A Rossi, Anup Rao, Tung Mai, Nedim Lipka, Nesreen K Ahmed, and Danai Koutra. Graph neural networks with heterophily. *arXiv preprint arXiv:2009.13566*, 2020a.

Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. In *Advances in Neural Information Processing Systems, NeurIPS*, 2020b.

Meiqi Zhu, Xiao Wang, Chuan Shi, Houye Ji, and Peng Cui. Interpreting and unifying graph neural networks with an optimization framework. *arXiv preprint arXiv:2101.11859*, 2021.

Daniel Zügner and Stephan Günnemann. Adversarial attacks on graph neural networks via meta learning. In *7th International Conference on Learning Representations, ICLR*, 2019.

Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. Adversarial attacks on neural networks for graph data. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI*, 2019.

A PROOFS

A.1 NOTATIONS

For simplicity, in this section we denote $\ell_Y(Y; \mathcal{W}, f, \rho, \widetilde{B}, \phi)$ in (3) by $\ell_Y(Y)$, denote $\phi(\boldsymbol{y}_i; W_\phi)$ by $\phi(\boldsymbol{y}_i)$, and denote $f(X;W)$ by $f(X)$. And the smooth part of ℓ_Y is denoted by $\ell_Y^s(Y) = \ell_Y(Y) - \sum_i \phi(\boldsymbol{y}_i)$. And we possibly ignore the parameter of the function (e.g. ℓ_Y).

We denote the vectorize of a matrix M by $\text{vec}(M)$ and the kronecker product of two matrices M_1 and M_2 denoted by $M_1 \otimes M_2$.

For Fenchel subdifferential respect to Y, we denote it by ∂_Y, ignoring the subscript $_\mathcal{F}$ used in main paper.

A.2 PROOF OF LEMMA 4.1

For proof of Lemma 4.1, we shall first state a property of Lipschitz continuous function:

Fact A.1 ((Bertsekas, 1999)) *for any $\ell(\boldsymbol{y})$ whose is $\mathcal{L}$-Lipschitz, it satisfies*

$$\forall \boldsymbol{y}, \boldsymbol{z}, \ell(\boldsymbol{y}) \leq \ell(\boldsymbol{z}) + \nabla \ell(\boldsymbol{z})^\top (\boldsymbol{y} - \boldsymbol{z}) + \frac{\mathcal{L}}{2}\|\boldsymbol{y} - \boldsymbol{z}\|_2^2 \tag{16}$$

From Fact A.1, we have that

$$\begin{aligned}
\forall \boldsymbol{z}, \boldsymbol{y}, \ell(\boldsymbol{y}) &\leq \ell(\boldsymbol{z}) + \nabla \ell(\boldsymbol{z})^\top (\boldsymbol{y} - \boldsymbol{z}) + \frac{\mathcal{L}}{2}\|\boldsymbol{y} - \boldsymbol{z}\|_2^2 && (17)\\
&= \ell(\boldsymbol{z}) + \nabla \ell(\boldsymbol{z})^\top \boldsymbol{y} - \nabla \ell(\boldsymbol{z})^\top \boldsymbol{z} + \frac{\mathcal{L}}{2}\|\boldsymbol{y}\|_2^2 + \frac{\mathcal{L}}{2}\|\boldsymbol{z}\|_2^2 - 2\frac{\mathcal{L}}{2}\boldsymbol{y}^\top \boldsymbol{z} && (18)\\
&= \ell(\boldsymbol{z}) - \frac{1}{2\mathcal{L}}\|\nabla \ell(\boldsymbol{z})\|_2^2 + \frac{\mathcal{L}}{2}\left[\frac{1}{\mathcal{L}^2}\|\nabla \ell(\boldsymbol{z})\|_2^2 + \|\boldsymbol{y}\|_2^2 + \|\boldsymbol{z}\|_2^2 + \frac{2}{\mathcal{L}}\nabla \ell(\boldsymbol{z})^\top \boldsymbol{y} - \frac{2}{\mathcal{L}}\nabla \ell(\boldsymbol{z})^\top \boldsymbol{z} - 2\boldsymbol{y}^\top \boldsymbol{z}\right] && (19)\\
&= \ell(\boldsymbol{z}) - \frac{1}{2\mathcal{L}}\|\nabla \ell(\boldsymbol{z})\|_2^2 + \frac{\mathcal{L}}{2}\left\|\boldsymbol{y} - \left[\boldsymbol{z} - \frac{1}{\mathcal{L}}\nabla \ell(\boldsymbol{z})\right]\right\|_2^2. && (20)
\end{aligned}$$

Taking $\ell(\boldsymbol{y}) = \ell_Y^s(\boldsymbol{y})$, $y = \text{vec}(Y)$ and $z = \text{vec}\left(Y^{(k)}\right)$, and let

$$\beta_Y\left(Y^{(k)}\right) = \ell_Y^s\left(Y^{(k)}\right) - \frac{\alpha}{2}\left\|\nabla \ell_Y^s\left(Y^{(k)}\right)\right\|_{\mathcal{F}}^2, \tag{21}$$

we can define an upper-bound of ℓ_Y:

$$\ell_Y^{(\text{upp})}(Y) = \frac{1}{2\alpha}\left\|Y - \left[Y^{(k)} - \alpha \nabla \ell\left(Y^{(k)}\right)\right]\right\|_{\mathcal{F}}^2 + \sum_i \phi(\boldsymbol{y}_i) + \beta_Y\left(Y^{(k)}\right), \tag{22}$$

and adopting (20) we have that:

$$\ell_Y(Y) = \ell_Y^s(Y) + \sum_i \phi(\boldsymbol{y}_i) \leq \ell_Y^{(\text{upp})}. \tag{23}$$

(Notice since $\alpha \leq \frac{1}{\mathcal{L}}$, ℓ_Y is also $\frac{1}{\alpha}$-Lipschitz). It's not difficult to check $\ell_Y^{(\text{upp})}(Y) = \ell_Y(Y)$ when $Y = Y^{(k)}$. Also, since adding terms which is not related to Y does not affect the $\arg\min$ result in (8), we have that

$$\begin{aligned}
Y^{(k+1)} &= \arg\min_Y \frac{1}{2\alpha}\|Y - U^{(k)}\|_{\mathcal{F}}^2 + \sum_i \phi(\boldsymbol{y}_i) && (24)\\
&= \arg\min_Y \frac{1}{2\alpha}\|Y - U^{(k)}\|_{\mathcal{F}}^2 + \sum_i \phi(\boldsymbol{y}_i) + \beta_Y\left(Y^{(k)}\right) && (25)\\
&= \arg\min_Y \ell_Y^{(\text{upp})}(Y). && (26)
\end{aligned}$$

Then we have

$$\ell_Y\left(Y^{(k+1)}\right) \leq \ell_Y^{(\text{upp})}(Y^{(k+1)}) \leq \ell_Y^{(\text{upp})}\left(Y^{(k)}\right) = \ell_Y\left(Y^{(k)}\right) \tag{27}$$

A.3 Proof of Lemma 4.2

This lemma is a straight-forward corollary of Theorem 1 in (Gribonval & Nikolova, 2020). For clarity, we excerpt the theorem here:

Fact A.2 (Theorem 1 in (Gribonval & Nikolova, 2020)) *Let $\mathcal{H}$ be some Hilbert space (e.g. $\mathbb{R}^k$) and $\mathcal{Y} \subset \mathcal{H}$ be non-empty. A function $\sigma : \mathcal{Y} \to \mathcal{H}$ is the proximal operator of a function $\phi : \mathcal{H} \to \mathbb{R} \cup \{+\infty\}$ if and only if there exists a convex l.s.c. (lower semi-continuous) function $\psi : \mathcal{H} \to \mathbb{R} \cup \{+\infty\}$ such that for each $\boldsymbol{y} \in \mathcal{Y}, \sigma(\boldsymbol{y}) \in \partial\psi(\boldsymbol{y})$.*

Since our σ is component-wise, we only need to consider $\mathcal{Y} = \mathcal{H} = \mathbb{R}$. Furthermore, since $\sigma(\boldsymbol{x})$ is continuous, its indefinite integration exists. Let $\psi(\boldsymbol{x}) = \int \sigma(\boldsymbol{x})\text{d}\boldsymbol{x}$, which must be convex since its derivative is non-decreasing. Then according to Fact A.2, there exists a function ϕ whose proximal operator is σ. Conversely, Also if there exists ϕ such that $\sigma = \text{prox}_\phi$, from Fact A.2 it is subdifferential of some convex function ψ, then it is non-decreasing.

A.4 Proof of Lemma 5.3

We only need a counterexample to prove this. Suppose $n = 1$ and $d = 2$, consider $W = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$, $Y = [a \quad b]$, then $YW = \begin{bmatrix} 0 \\ a \end{bmatrix}$.

If there exists any second order smooth function h such that $\frac{\partial h}{\partial a} = 0$ and $\frac{\partial h}{\partial b} = a$, we have that

$$\frac{\partial^2 h}{\partial a \partial b} = 0 \neq \frac{\partial^2 h}{\partial b \partial a} = 1,$$

which contradicts the second order smoothness assumption of h.

A.5 Proof of Theorem 5.1

First, we will show that ℓ_Y^s is strongly convex under conditions given. We have

$$\begin{aligned} \operatorname{vec}\left(\frac{\partial \ell_Y^s}{\partial Y}\right) &= \operatorname{vec}\left[YW_f^s - f(X;W) + \tilde{L}YW_p^s\right] && (28) \\ &= \left(W_f^s \otimes I + W_p^s \otimes \tilde{L}\right)\operatorname{vec}(Y) + \operatorname{vec}\left[f(X;W)\right] && (29) \\ &= \Sigma \operatorname{vec}(Y) + \operatorname{vec}\left[f(X;W)\right] && (30) \end{aligned}$$

and therefore the Hessian of ℓ_Y^s is

$$\frac{\partial^2 \ell_Y^s}{\partial \operatorname{vec}(Y)^2} = \Sigma. \quad (31)$$

We also know that when $\lambda_{\text{min}}(\Sigma) > 0$, ℓ_Y^s is strongly convex respect to Y, thus has a unique global minimum.

As for the case where non-smooth penalty term ϕ included, we first introduce another theorem from (Gribonval & Nikolova, 2020).

Fact A.3 (Proposition 1 in (Gribonval & Nikolova, 2020)) *Let $\mathcal{H}$ be some Hilbert space (e.g. $\mathbb{R}^k$), a function $\sigma : \mathcal{H} \to \mathcal{H}$ defined everywhere is the proximity operator of a proper convex l.s.c function $\phi : \to \mathbb{R} \cup \{+\infty\}$ if and only if the following conditions hold jointly:*

1. *there exists a (convex l.s.c) function ψ such that for each $\boldsymbol{y} \in \mathcal{H}, \sigma(\boldsymbol{y}) \in \partial\psi(\boldsymbol{y})$;*
2. *σ is non-expansive.*

We already assumed σ be a proximal operator, thus through Fact A.2 we know condition 1 is already satisfied, and since we further assumed non-expansive σ, condition 2 is also satisfied. Therefore from Fact A.3 we conclude that ϕ is convex. And since ℓ_Y is the summation of a strongly convex part ℓ_Y^s and a convex part $\sum_{i=1}^n \phi(\boldsymbol{y}_i)$, it is strongly convex and have unique global minimum.

Furthermore, the $\lambda_{\text{max}}(\Sigma)$ is the Lipschitz constant of ℓ_Y^s, and from theorem 5.4 we know when $\alpha \leq 1/\lambda_{\text{max}}(\Sigma)$ the algorithm converges to the stationary point, which is the global optimum.

Proof of Corollary 5.1.1 Taking $W_f^s = I - W_p^s$ and $P = I - \tilde{L}$, we have that

$$\Sigma = I - P \otimes W_p^s. \quad (32)$$

By the spectral properties of Kronecker product, we have

$$\lambda_{\text{min}}(\Sigma) = 1 - \left\|W_p^s\right\|_2 \|P\|_2 > 0, \quad (33)$$

and shifting the order of the inequality gives $\|W_p^s\|_2 < \|P\|_2^{-1}$.

Proof of Lemma 5.2 We first convert (2) to vector form:

$$\operatorname{vec}\left(Y^{(k+1)}\right) = \sigma\left[(W_p \otimes P)\operatorname{vec}\left(Y^{(k)}\right) + \operatorname{vec}(f)\right], \quad (34)$$

where $f = f(X; W_x)$, ignoring the parameters for simplicity. Let $M = W_p \otimes P$, from the spectral properties of knronecker product, we have

$$\|M\|_2 = \|W_p\|_2 \|P\|_2 < 1. \tag{35}$$

By definition of matrix norm we have

$$\forall \boldsymbol{x}_1, \boldsymbol{x}_2, \|M(\boldsymbol{x}_1 - \boldsymbol{x}_2)\|_2 \leq \|M\|_2 \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2, \tag{36}$$

which means M (as a linear transformation) is a contraction mapping on Euclidean space. We also assumed σ is contraction mapping, thus from Banach's fixed point theorem(Kinderlehrer & Stampacchia, 1980) we conclude that (34), as well as (2), has a unique fixed point.

A.6 Proof of Theorem 5.4 and Corollary 5.4.1

The proof is followed by Zangwill's convergence theorem, we state it here:

Fact A.4 (Zangwill's Convergence Theorem (Luenberger, 1984)) *Let $A : \mathcal{X} \to 2^{\mathcal{X}}$ be a set-valued function, $\mathcal{G} \subset \mathcal{X}$ be a solution set (which can be any set we are interested in), for any sequence $\{\boldsymbol{x}_k\}_{k=1}^{\infty}$ such that $\boldsymbol{x}_{k+1} \in A(\boldsymbol{x}_k)$, if following conditions hold jointly:*

1. *$\{\boldsymbol{x}_k | 1 \leq k \leq \infty\}$ is contained in some compact subset of $\mathcal{X}$.*
2. *there exists a continuous function ℓ such that*
   - *(a) if $\boldsymbol{x} \notin \mathcal{G}$, then $\forall \boldsymbol{y} \in A(\boldsymbol{x}), \ell(\boldsymbol{y}) < \ell(\boldsymbol{x})$.*
   - *(b) if $\boldsymbol{x} \in \mathcal{G}$, then $\forall \boldsymbol{y} \in A(\boldsymbol{x}), \ell(\boldsymbol{y}) \leq \ell(\boldsymbol{x})$.*
3. *the mapping A is closed at all point of $\mathcal{X} \setminus \mathcal{G}$ i.e. $\{(\boldsymbol{x}, \boldsymbol{y}) | \boldsymbol{x} \in clos\,(X \setminus \mathcal{G}), \boldsymbol{y} \in A(\boldsymbol{x})\}$ is a closed set, where* clos *means set closure.*

then the limit of any convergent subsequence of $\{\boldsymbol{x}_k\}_{k=1}^{\infty}$ is a solution i.e. inside $\mathcal{G}$.

To prove Theorem 5.4 we define the solution set as the set of generalized fixed points of ℓ_Y:

$$\mathcal{G} = \left\{ Z \middle| Z \in \arg\min_{Y} \frac{1}{2\alpha} \|Y - (Z - \alpha \nabla \ell_Y^s(Z))\|_{\mathcal{F}}^2 + \phi(Y) \right\} \tag{37}$$

Then the proof is composed of three steps:

1. showing $\ell_Y(Y)$ is a descent function;
2. showing the process is closed and $Y^{(k)}$s are bounded;
3. showing that the solution set defined in (37) is the set of all stationary points.

The proof for step 1 is already finished in the proof of Lemma 4.2. Notice that by definition of the solution set, the strict descent condition on $\mathcal{X} - \mathcal{G}$ is automatically satisfied since if $Y^{(k)} \notin \mathcal{G}$ then

$$\ell_Y\left(Y^{(k)}\right) > \min_Y \left[\frac{1}{2\alpha} \left\|Y - U^{(k)}\right\|_{\mathcal{F}}^2 + \phi(Y)\right] = \ell_Y\left(Y^{(k+1)}\right). \tag{38}$$

To further prove the convergence using Fact A.4, we also need to illustrate that the process is closed and the iterations $Y^{(k)}$ are limited in a bounded set. The former is a straight-forward deduction of Lemma 6 in (Sriperumbudur & Lanckriet, 2009), and the latter one is easily shown as follows. Consider the set

$$S = \left\{Y | \ell_Y(Y) \leq \ell_Y\left(Y^{(0)}\right)\right\}. \tag{39}$$

By Lemma 4.2, $Y^{(k)} \in S$ and by the assumption that $\ell_Y(Y) \to \infty$ when $\|Y\| \to \infty$, we know S is bounded.

So far, by Fact A.4, we have proved any accumulation point is in $\mathcal{G}$, but we defined $\mathcal{G}$ in a way that it is the set of fixed points of the algorithm. We then connect fixed points with stationarity as promised in the Theorem 5.4 i.e. Step 3.

Lemma A.1 (Step 3)

$$Z \in \arg\min_{Y} \frac{1}{2\alpha} \|Y - (Z - \alpha\nabla \ell_Y^s(Z))\|_{\mathcal{F}}^2 + \phi(Y) \iff 0 \in \partial_Z \ell_Y(Z) \tag{40}$$

Proof

$$Z \in \arg\min_{Y} \frac{1}{2\alpha} \|Y - (Z - \alpha\nabla \ell_Y^s(Z))\|_{\mathcal{F}}^2 + \phi(Y) \tag{41}$$

$$\iff 0 \in \partial_Y \left[\frac{1}{2\alpha} \|Y - (Z - \alpha\nabla \ell_Y^s(Z))\|_{\mathcal{F}}^2 + \phi(Y)\right]\Bigg|_{Y=Z} \tag{42}$$

$$\iff 0 \in \nabla \ell_Y^s(Z) + \partial_Z \phi(Z) \tag{43}$$

$$\iff 0 \in \partial_Z(\ell_Y^s(Z) + \phi(Z)) = \partial_Z \ell_Y(Z) \tag{44}$$

■

Here we used the summation property and local minimality property of Fenchel subdifferential, further illustration of those properties can be found in (Li et al., 2020).

For the last part of the theorem, it follows directly from the continuity of ℓ_Y. Additionally, for Corollary 5.4.1, following the same deduction of Theorem 5.1, ℓ_Y will be strongly convex so there will only be one stationary point which is the global optimum.

A.7 Proof of Theorem 6.1

First consider a case which allows T and W_p^s to have complex values. Hereinafter we denote the space of all complex-valued matrices with shape $d_1 \times d_2$ by $\mathbb{C}^{d_1 \times d_2}$.

For preciseness, we recall Jordan's decomposition theorem below. Let $J(\lambda)$ be Jordan block matrix $J(\lambda) = \begin{bmatrix} \lambda & 1 & & & \\ & \lambda & 1 & & \\ & & \lambda & & \\ & & & \ddots & 1 \\ & & & & \lambda \end{bmatrix}$. We have:

Fact A.5 (Jordan) *For every $W \in \mathbb{R}^{d\times d}$, there exists a complex-valued invertible matrix $P \in \mathbb{C}^{d\times d}$ and complex-valued block-diagonal matrix $\Omega = \begin{bmatrix} J(\lambda_1) & & & \\ & J(\lambda_2) & & \\ & & \ddots & \\ & & & J(\lambda_k) \end{bmatrix}$ such that $W = P\Omega P^{-1}$, where $\lambda_j \in \mathbb{C}$ is the j-th eigenvalue of W and the size of $J(\lambda_j)$ is the algebraic multiplicity of λ_j.*

Corollary A.1.1 *If a square matrix $W \in \mathbb{R}^{d\times d}$ has d distinct eigenvalues the there exists $P, \Lambda \in \mathbb{C}^{d\times d}$ where P is invertible and Λ is diagonal, such that $W = P\Lambda P^{-1}$.*

Then, we shall show that actually on the complex domain "almost" every matrix can be diagonalized, which means, for each matrix W, either it itself can be diagonalized, or there's a diagonalizable matrix W' that is arbitrarily closed to W.

Corollary A.1.2 *For every square matrix $W \in \mathbb{R}^{d\times d}$ and any $\epsilon > 0$, there exists a diagonalizable square matrix $W' \in \mathbb{R}^{d\times d}$ such that $\|W' - W\| \leq \epsilon$.*

Proof Let $W = R\Omega R^{-1}$ and $E = \operatorname{diag}[\epsilon_1 \quad \epsilon_2 \quad \cdots \quad \epsilon_n]$ such that the diagonal of $\Omega + E$ is distinct and $\epsilon_j^2 < \frac{\epsilon^2}{d\|R\|^2\|R^{-1}\|^2}$.

Since Ω is upper-triangle, its eigenvalues are the elements in its diagonal, which are distinct. Thus according to Corollary A.1.1, $\Omega + E$ is diagonalizable, suppose $\Omega + E = Q\Lambda Q^{-1}$. Let $W' = R(\Omega + E)R^{-1} = RQ\Lambda Q^{-1}R^{-1}$, it is apparent W' also diagonalizable.

Now consider the difference between W' and W. We have $\|W' - W\|^2 = \|PEP^{-1}\|^2 \leq \|E\|^2\|R\|^2\|R^{-1}\|^2 < \epsilon^2$.

■

Note the norm in Corollary A.1.2 can be any matrix norm.

Lemma A.2 *The solution of equation $PYW + XW_x = Y$ is continuous respect to W as long as a unique solution exists.*

Proof The solution of this equation is $\text{vec}(Y) = \left(I - W^\top \otimes P\right)^{-1} \text{vec}(XW_x)$ is continuous.

■

Theorem A.3 *$\forall W_p \in \mathbb{R}^{d\times d}$ that admits unique fixed point for IGNN and $W_x \in \mathbb{R}^{d_0\times d}$, suppose $Y \in \mathbb{R}^{n\times d}$ is the only solution of $PYW + XW_x = Y$, then there exists a $Y' \in \mathbb{C}^{n\times d}$, $\bar{W}_x \in \mathbb{C}^{d_0\times d'}$, right-invertible $\bar{T} \in \mathbb{C}^{d\times d'}$ and herimitian $\bar{W}_p^s \in \mathbb{C}^{d'\times d'}$, such that*

$$PY'\bar{T}\bar{W}_p^s + X\bar{W}_x = Y'\bar{T}, \text{ with } \|Y' - Y\| < \epsilon, \forall\epsilon > 0, Y' \in \mathbb{C}^{n\times d}. \tag{45}$$

Proof By Corollary A.1.2 and Lemma A.2, there exists diagonalizable $W' \in \mathbb{R}^{d\times d}$ such that $PY'W' + XW_x = Y'$ for some Y' satisfies $\|Y' - Y\| < \epsilon$.

Suppose now $W' = R\Lambda R^{-1}$ where Λ is diagonal and R is invertible. We have

$$PY'R\Lambda R^{-1} + XW_x = Y', \tag{46}$$

and right-multiply by R at each side of this equation we get

$$P(Y'R)\Lambda + XW_xR = Y'R. \tag{47}$$

By choosing $\bar{T} = R, \bar{W}_p^s = \Lambda, \bar{W}_x = W_xR^{-1}$, we proved the proposition. ■

Next, we consider restricting this result to the real domain. This is straight-forward since every complex linear transformation can be converted to real linear transformation by consider real and imaginary parts separately. Hereinafter, we denote the real part of an complex-valued matrix M by $M_{(r)}$ and imaginary part by $M_{(i)}$.

Proof of Theorem 6.1 Let $\bar{Y}$, $\bar{T}$, $\bar{W}_p^s$, $\bar{W}_x$ are matrices output by Theorem A.3. Note the last three are complex-valued, $\bar{T}$ is invertible and $\bar{W}_p^s$ is hermitian.

Consider $T = \begin{bmatrix}\bar{T}_{(r)} & \bar{T}_{(i)}\end{bmatrix}$. Since $\bar{T}$ is invertible, let $\bar{L} = \bar{T}^{-1}$, then we have $I = (\bar{T}\bar{L})_{(r)} = \bar{T}_{(r)}\bar{L}_{(r)} - \bar{T}_{(i)}\bar{L}_{(i)}$. Thus $T\begin{bmatrix}\bar{L}_{(r)} \\ -\bar{L}_{(i)}\end{bmatrix} = \bar{T}_{(r)}\bar{L}_{(r)} - \bar{T}_{(i)}\bar{L}_{(i)} = I$. This means T is right-invertible.

Then let $W_s^p = \begin{bmatrix}\bar{W}_{p(r)}^s & \bar{W}_{p(i)}^s \\ -\bar{W}_{p(i)}^s & \bar{W}_{p(r)}^s\end{bmatrix}$. Since $\bar{W}_p^s$ is hermitian, we have that $\bar{W}_{p(r)}^s = \bar{W}_{p(r)}^{s\top}$ and $\bar{W}_{p(i)}^s = -\bar{W}_{p(i)}^{s\top}$, which ensures W_p^s constructed here is symmetric.

At last, let $\tilde{W}_x = \begin{bmatrix}\bar{W}_{x(r)} & \bar{W}_{x(i)}\end{bmatrix}$.

Now it's easy to verify that $PY'TW_p^s + X\tilde{W}_x = Y'T$.

■

A.8 Proof of Theorem 6.2

First we will illustrate that with some value of the parameters in UGNN, the iteration step from (8) can form a GCN layer with symmetric shared weights, with or without residual connections. We set ρ to be identity (so that $\Gamma = I$), $\tilde{B}$ satisfies that $\tilde{B}^\top\tilde{B} = I - P$ (e.g. if $P = \hat{A} = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$ is normalized adjacency, then $\tilde{B}$ is normalized incidence matrix $\tilde{B} = \tilde{D}^{-1/2}B$).

For a GCN model without residual connections, let $W_f^s = I - W_p^s$, then we have $Y^{(k+1)} = \sigma(PY^{(k)}W_p^s)$. Consider $Y^{(0)} = \begin{bmatrix} Y_{\mathrm{GCN}}^{(0)} & 0 & 0 & \cdots & 0 \end{bmatrix}$ where $Y_{\mathrm{GCN}}^{(0)}$ is the input of GCN, and the ith 0 here denote a block matrix of all zero with the same size as $Y_{\mathrm{GCN}}^{(i)}$. When we let

$$W_p^s = \begin{bmatrix} 0 & W_p^{(1)} & 0 & 0 & \cdots & 0 \\ W_p^{(1)\top} & 0 & W_p^{(2)} & 0 & \cdots & 0 \\ 0 & W_p^{(2)\top} & 0 & W_p^{(3)} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & W_p^{(k)} \\ 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix}, \tag{48}$$

it is not difficult to verify that the kth block of $Y^{(k)}$ is $Y_{\mathrm{GCN}}^{(k)}$. Note the 0s in the (48) are block matrices of all zeros and with proper size.

For GCN model with residual connection, let $W_f^s = I - W_p^s - W_r^s$, then $Y^{(k+1)} = \sigma(PY^{(k)}W_p^s + Y^{(k)}W_r^s)$. We take the same W_p^s as before (48), and let

$$W_r^s = \begin{bmatrix} 0 & I & 0 & 0 & \cdots & 0 \\ I & 0 & I & 0 & \cdots & 0 \\ 0 & I & 0 & I & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & I \\ 0 & 0 & 0 & 0 & \cdots & 0 \end{bmatrix}. \tag{49}$$

Since W_r^s and W_p^s are both symmetric, apparently W_f^s is also symmetric. Also notice that since residual connection exists, the size of all the $W_p^{(i)}$s are the same, so the size of I in the (49) is the same as $W_p^{(i)}$.

B A SUFFICIENT CONDITION OF EQUIVALENCE

In this section, we consider general nonlinear functions σ and penalty term ϕ. Hereinafter we denote prox_ϕ by $\tilde{\sigma}$. We add mild assumption on σ and consider a general type of $\tilde{\sigma}$.

Constraint on σ To simplify the discussion, we assume $\sigma(\boldsymbol{x})$ is a proximal operator of some penalty function and is differentiable with respect to x.

This assumption is consistent with practice. Firstly, commonly used non-linearities like $\tanh$, $\mathrm{sigmoid}$ and ReLU are all proximal operators since they are all continuous and component-wise non-decreasing (By Lemma 4.2). Also, although at some point ReLU and Leaky-ReLU are not differentiable, we can approximate them by

$$\text{ReLU}(x) \approx \frac{1}{r}\log\left(\exp(rx) + 1\right)$$

and

$$\text{Leaky-ReLU}(x, p) \approx \frac{1}{r}\log\left(\exp(rx) + \exp(prx)\right)$$

with big enough r, which are differentiable and do not affect their practical attributes.

$\tilde{\sigma}$ ***Considered***

Fact B.1 *A twice-differentiable function ψ is convex iff its Hessian H_ψ is positive semi-definite.*

As discussed in the main paper, it would cause some degenerated cases if we allow a general class of proximal operators. Also it would too difficult to handle in this case. Therefore, we only consider proximal operators in a special while general family

$$\mathcal{S} = \{\tilde{\sigma} : \boldsymbol{x} \mapsto G\sigma(C\boldsymbol{x}) | G, C \text{ are chosen such that } \tilde{\sigma} \text{ is proximal operator}\}$$

Now we consider under what value of G and C we can ensure $\tilde{\sigma}$ is proximal operator. From Fact A.2, we know $\exists \phi, \tilde{\sigma} = \operatorname{prox}_\phi$ iff $\tilde{\sigma}$ is subgradient of some l.s.c convex function. Since we have assumed σ is continuous and differentiable, $\int \sigma(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}$ is twice differentiable. Thus from Fact B.1 we know $H(\boldsymbol{x}) = \frac{\partial \sigma(\boldsymbol{x})}{\partial \boldsymbol{x}}$ is positive semi-definite, and so do Hessian of $\tilde{\sigma}(\boldsymbol{x})$, i.e. $\tilde{H}(\boldsymbol{x}) = \frac{\partial \tilde{\sigma}}{\partial \boldsymbol{x}}$. From the chain rule we know

$$\tilde{H}(\boldsymbol{x}) = \frac{\partial \tilde{\sigma}(\boldsymbol{x})}{\partial \boldsymbol{x}} = \frac{\partial G\sigma(C\boldsymbol{x})}{\partial \boldsymbol{x}} = G\frac{\partial \sigma(C\boldsymbol{x})}{\partial C\boldsymbol{x}}\frac{\partial C\boldsymbol{x}}{\partial \boldsymbol{x}} = GH(C\boldsymbol{x})C.$$

From the deduction above, we can conclude that:

Lemma B.1 *$\tilde{\sigma} : \mathbb{R}^d \to \mathbb{R}^d, \boldsymbol{x} \mapsto G\sigma(C\boldsymbol{x})$ is proximal operator iff $\forall \boldsymbol{x} \in C\mathbb{R}^d, GH(\boldsymbol{x})C$ is positive semi-definite where $H(\boldsymbol{x}) = \frac{\partial \sigma(\boldsymbol{x})}{\partial \boldsymbol{x}}$.*

The Condition of General Fixed-Point Alignment

Theorem B.2 *For σ satisfying the assumptions above, $W_p \in \mathbb{R}^{d\times d}$ that admits unique fixed point for IGNN, and $W_x \in \mathbb{R}^{d_0\times d}$, suppose Y is the only solution of $Y = \sigma(PYW + XW_x)$, a sufficient condition of*

$$\exists \phi,\ \textit{right invertible } T \in \mathbb{R}^{d\times d'},\ \textit{symmetric } W_p^s \in \mathbb{R}^{d'\times d'} \textit{ and } \tilde{W}_x \in \mathbb{R}^{d\times d} \textit{ such that}$$
$$\tilde{\sigma}(PYTW_p^s + XW_x) = YT$$

is that

$$\exists \textit{ right-invertible } T \in \mathbb{R}^{d\times d'}, C \in \mathbb{R}^{d'\times d}, W_p^s \in \mathbb{R}^{d'\times d'} \textit{ such that } TW_p^sC = W_p$$
$$\textit{and } \forall \boldsymbol{x} \in C^\top\mathbb{R}, T^\top H(\boldsymbol{x})C^\top \textit{ is P.S.D,}$$

where $\tilde{\sigma}(\boldsymbol{x}) = \operatorname{prox}_\phi(\boldsymbol{x})$ and $H(\boldsymbol{x}) = \frac{\partial \sigma(\boldsymbol{x})}{\partial \boldsymbol{x}}$.

Proof We have assumed $\tilde{\sigma} \in \mathcal{S}$, thus $\tilde{\sigma}(\boldsymbol{x}) = G\sigma(C\boldsymbol{x})$. We want to prove that

$$\begin{aligned} YT &= \tilde{\sigma}(PYTW_p^s + X\tilde{W}_x) & (50)\\ &= \sigma(PYTW_p^sC + X\tilde{W}_xC)G & (51)\\ &= \sigma(PYTW_p^sC + XW_x)G \text{ (Assume } \tilde{W}_xC = W_x). & (52)\end{aligned}$$

A sufficient condition for this equation to hold is $T = G$ and $Y = \sigma(AYTW_p^sC + XW_x) = \sigma(AYW_p + XW_x)$. Comparing the last two terms, apparently it holds when $TW_p^sC = W_p$. And to ensure $\tilde{\sigma} \in \mathcal{S}$, from Lemma B.1 we know it means $T^\top H(C^\top x)C^\top$ be positive semi-definite for all x.

■

We can verify this Theorem B.2 by linear case. If σ is linear, then $H(Cx) = I$, thus simply taking $C = T^{-1}$ (on complex field) we have $TW_p^sT^{-1}$ can generate any real matrix W and $T^\top H(C^\top x)C^\top = T^\top T^{-\top} = I$ is positive semi-definite.

C Experiment Details

C.1 Datasets

In this paper, we used Cora, Citeseer and Pubmed (Sen et al., 2008) and OGBN-arxiv(Hu et al., 2020) which are commonly used datasets to evaluate GNN models in general purpose. We also used Wisconsin, Texas, Actor and Cornell datasets introduced by (Pei et al., 2019), which are small but with a strong property of heterophily, and is used to test models' performance on heterophily graphs. We also include results on amazon co-purchase which is originally used in IGNN(Gu et al., 2020) and TWIRLS(Yang et al., 2021). The experiment settings generally follows (Yang et al., 2021).

The statistic of different datasets are summarized in Table 3.

Table 3: Dataset statistics. Note that Amazon co-purchase does not have node features.

Dataset	# Nodes	# Edges	Feature Dim	# Classes
Cora	2,708	5,429	1,433	7
Citeseer	3,327	4,732	3,703	6
Pubmed	19,717	44,339	500	3
OGBN-Arxiv	169,343	1,166,243	128	40
Texas	183	309	1,703	5
Wisconsin	251	499	1,703	5
Actor	7,600	33,544	931	5
Cornell	183	295	1,703	5
Amazon	334,863	2,186,607	-	58

C.2 Model Implementation

For model implementation, we directly adopt IGNN's code and make a small modification of TWIRLS's code to make it a general UGNN.

In UGNN, since it is too general to consider all the choice of ρ and ϕ, we adopt $\rho(Y) = Y$ which produces $\Gamma = I$ and $\phi(x) = \mathcal{I}_\infty[x < 0]$ which produces $\text{prox}_\phi = \text{ReLU}$, as mentioned in section 4.2. Also, We introduce another hyper-parameter λ and let $W_f^s = (1 + \lambda)I - W_p^s$ and $W_p^s = \lambda \tilde{W}$, thus the iteration becomes

$$Y^{(k+1)} \leftarrow (1 - \alpha - \alpha\lambda)Y^{(k)} + \alpha\lambda PY^{(k)}\tilde{W} + \alpha f(X; W), \tag{53}$$

which covers model described in (11) when $\lambda = 1$ and $\alpha = \frac{1}{1+\lambda}$[5] but is more general.

When projecting W_p (or $\tilde{W}$ in UGNN) to the space that admits unique fixed point, in IGNN's original paper it uses $\|\cdot\|_{\text{inf}}$ (Gu et al., 2020), which would break the symmetry, based on our result in Section 5.1, in our implementation when the model weight is symmetric we project $\|\cdot\|_2$ instead of $\|\cdot\|_{\text{inf}}$.

For propagation matrix we always use normalized adjacency $P = \hat{A}$.

C.3 Further Details about Label Recovering Task

In this task, we use model with $f(X) = XW_x$ and $g\left[\boldsymbol{y}_i^*(\mathcal{W}); \theta\right] = W_g \boldsymbol{y}_i^*(\mathcal{W})$ where $W_g \in \mathbb{R}^{c\times d}$ is a learned matrix that maps the propagated node features to the output space. We set the hidden size to 32 in asymmetric case and 34 in symmetric case to ensure nearly the same number of parameters (we deliberately set the hidden size to this small to ensure the models do not overfit). For generating models, we first train them using the original labels of the dataset by 500 steps. The number of propagation steps is set to 2, λ is set to 1, α is set to $\frac{1}{2}$ for UGNN.

C.4 Further Details about Memory and Timing

In this section we adopt amazon co-purchase dataset with label ratio = 0.6. The hyper-parameters are set the same as the original IGNN paper (Gu et al., 2020) for all the models. As mentioned before, in UGNN we use $\lambda = 1, \alpha = \frac{1}{2}$ and explicitly times the parameter matrix by 2 to exactly match the iteration of (11), under consideration that not hurting the training dynamics.

C.5 Further Details about Best-Model Comparing

It is worth mentioning that we did not traverse all possible implementations of the two models (especially for UGNN since it has more general forms), and therefore, the comparisons have some

[5]Actually, they differ by a coefficient of 2 which can be absorbed into $\tilde{W}$ and W_x since they are all learned parameters. When comparing UGNN with IGNN, we explicitly multiplies this 2 to eliminate the potential impact on training dynamics

limitations. The hyper-parameters that we considered include the specific implementation of $f(X;W)$ ($f(X;W) = XW$ or $f(X;W) = AXW$), the number of MLP layers in f and g, the way of initialization ($Y^{(0)} = f(X)$ or $Y^{(0)} = 0$) and learning rate, weight decay rate and dropout rate. And for UGNN the number of propagation steps, values of α and λ are also considered. Also, UGNN w/I results in the table come from results of TWIRLS(Yang et al., 2021), while the other three were conducted by us.