The Effects of Activation Functions on the Over-smoothing Issue of Graph Convolutional Networks

Bao Wang
Department of Mathematics
Scientific Computing and Imaging Institute
The University of Utah

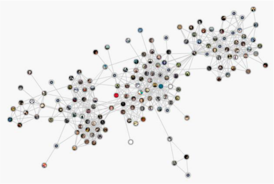
CBMS Conference: Deep Learning and Numerical PDEs
Morgan State University

Partially supported by DOE and NSF.

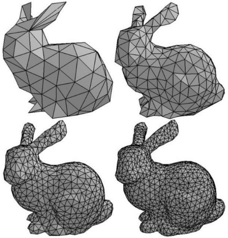

Shih-Hsin Wang*, Justin Baker*, Cory Hauck, and Bao Wang, The Effects of Activation Functions on the Over-smoothing Issue of Graph Convolutional Networks, submitted.

Learning Non-Euclidean Data?

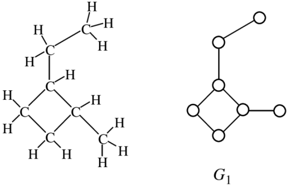
- Graph is a flexible structure to represent non-Euclidean data.



Social Graph (Facebook, Wikipedia)



3D Mesh



Molecular Graph

Graph convolutional networks

- Let $G = (V, E)$ be an undirected graph where $V = \{v_i\}_{i=1}^n$ is the set of nodes and E is the set of edges.

- Let $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ be the adjacency matrix of G.

- Let $\boldsymbol{G} := (\boldsymbol{D} + \boldsymbol{I})^{-\frac{1}{2}}(\boldsymbol{I} + \boldsymbol{A})(\boldsymbol{D} + \boldsymbol{I})^{-\frac{1}{2}} = \tilde{\boldsymbol{D}}^{-\frac{1}{2}} \tilde{\boldsymbol{A}} \tilde{\boldsymbol{D}}^{-\frac{1}{2}}$ be the (augmented) normalized adjacency matrix.

- Graph convolutional layer (GCL):

$$\boldsymbol{H}^l = \sigma(\boldsymbol{W}^l \boldsymbol{H}^{l-1} \boldsymbol{G}),$$

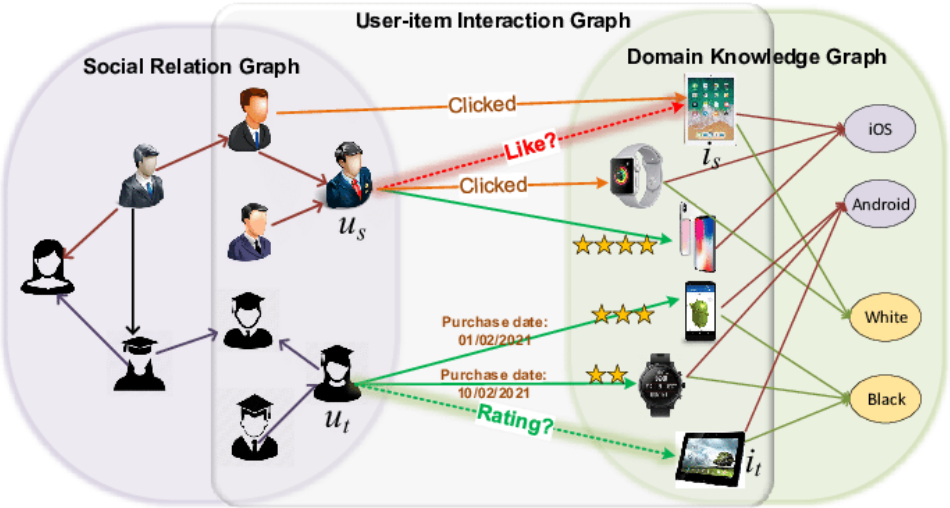
where σ is the activation function, $\boldsymbol{W}^l \in \mathbb{R}^{d \times d}$ is a learnable weight matrix, and $\boldsymbol{H}^0 := [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_n] \in \mathbb{R}^{d \times n}$ with $\boldsymbol{h}_i$ being the i^{th} node feature. A message-passing scheme rather than exact convolution.

Kipf and Welling, ICLR, 2017.

Graph learning tasks

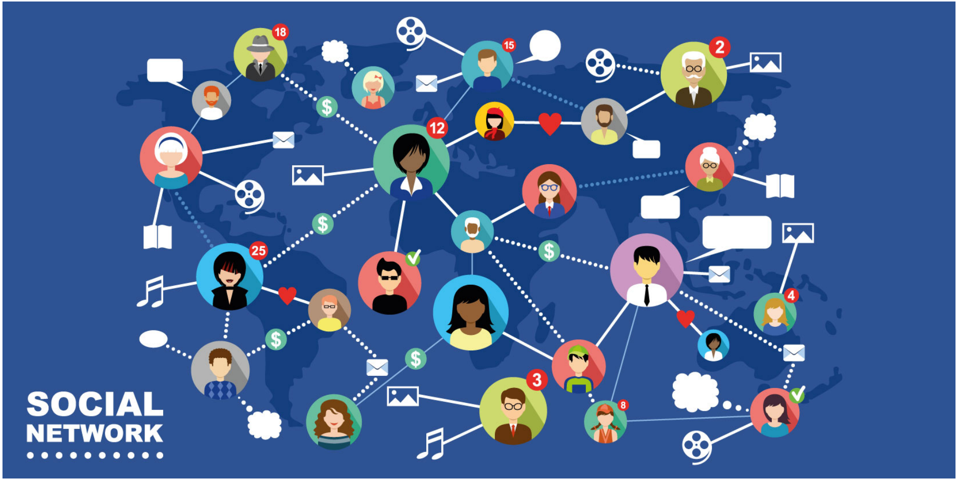
- Node classification
- Link prediction
- Graph classification and generation

Applications: Recommender system



Link prediction

Applications: Social network



Node classification

Over-smoothing of GNN

- All eigenvalues of $\boldsymbol{G}$ lie in the interval $(-1, 1]$.

- $\boldsymbol{H}^l = \boldsymbol{W}^l \boldsymbol{H}^{l-1} \boldsymbol{G}$, i.e., $\mathrm{vec}(\boldsymbol{H}^l) = \boldsymbol{G}^\top \otimes \boldsymbol{W}^l \mathrm{vec}(\boldsymbol{H}^{l-1})$, can be considered as a low-pass filter, indicating that each GCL "smooths" node features.

- As the GCN architecture gets deep, all nodes' representation – within each connected component – will become "indistinguishable", which is referred to as *over-smoothing*.

- Learning long-range dependencies ("long-range interaction") is hard.

Existing Theory

Mathematical characterization of the over-smoothing - I (Oono & Suzuki, ICLR, 2019.)

- Distance of the node features $\boldsymbol{H}^l$ to the eigenspace $\mathcal{M}$ – the eigenspace corresponding to the largest eigenvalue of $\boldsymbol{G}$ – goes to zero.

  > Suppose the graph G has m connected components, i.e. we can decompose $V = \bigcup_{i=1}^m V_i$. Let $\boldsymbol{u}_i = (1_{\{k \in V_i\}})_{1 \le k \le n}$ be the indicator vector of the i^{th} component V_i.

  > The nonegative vectors $\{\tilde{\boldsymbol{D}}^{\frac{1}{2}} \boldsymbol{u}_i / \|\tilde{\boldsymbol{D}}^{\frac{1}{2}} \boldsymbol{u}_i\|\}_{1 \le i \le m}$ form an orthonormal basis of $\mathcal{M}$.

- Let $\mathbb{R}^d \otimes \mathcal{M}$ be the subspace of $\mathbb{R}^{d \times n}$ consisting of the sum $\sum_{i=1}^m \boldsymbol{w}_i \otimes \boldsymbol{e}_i$ where $\boldsymbol{w}_i \in \mathbb{R}^d$ and $\{\boldsymbol{e}_i\}_{i=1}^m$ is an orthonormal basis of the eigenspace $\mathcal{M}$. Then the distance of $\boldsymbol{H}^l$ to $\mathcal{M}$ is

$$\|\boldsymbol{H}^l\|_{\mathcal{M}^\perp} := \inf_{\boldsymbol{Y} \in \mathbb{R}^d \otimes \mathcal{M}} \|\boldsymbol{H}^l - \boldsymbol{Y}\|_F = \left\| \boldsymbol{H}^l - \sum_{i=1}^m \boldsymbol{H}^l \boldsymbol{e}_i \boldsymbol{e}_i^\top \right\|_F.$$

- $\|\boldsymbol{H}^l\|_{\mathcal{M}^\perp} \le s_l \lambda \|\boldsymbol{H}^{l-1}\|_{\mathcal{M}^\perp}$ when σ is ReLU. Here, $\lambda = \max\{|\lambda_i| \mid \lambda_i < 1\}$ is the second largest magnitude of $\boldsymbol{G}$'s eigenvalues, and s_l is the largest singular value of $\boldsymbol{W}^l$.

Effects of ReLU

- $\|\sigma(\boldsymbol{Z})\|_{\mathcal{M}^\perp} \leq \|\boldsymbol{Z}\|_{\mathcal{M}^\perp}$ for any matrix $\boldsymbol{Z}$ when σ is ReLU, i.e. ReLU reduces the distance to eigenspace $\mathcal{M}$. – Oono & Suzuki, ICLR, 2019

Mathematical characterization of the over-smoothing - II (Cai & Wang, arXiv:2006.13318)

- Dirichlet energy of node features:

$$\|\boldsymbol{H}\|_E^2 := \mathrm{Trace}(\boldsymbol{H}\tilde{\Delta}\boldsymbol{H}^\top),$$

where $\tilde{\Delta} = \boldsymbol{I} - \boldsymbol{G}$ is the (augmented) normalized Laplacian.

- $\|\boldsymbol{H}^l\|_E \leq s_l \lambda \|\boldsymbol{H}^{l-1}\|_E$ when σ is ReLU or leaky ReLU.

Effects of activation function: Existing theory

- $\|\sigma(\boldsymbol{Z})\|_{\mathcal{M}^\perp} \leq \|\boldsymbol{Z}\|_{\mathcal{M}^\perp}$ for any matrix $\boldsymbol{Z}$ when σ is ReLU, i.e. ReLU reduces the distance to eigenspace $\mathcal{M}$. – Oono & Suzuki, ICLR, 2019

- $\|\sigma(\boldsymbol{Z})\|_{E} \leq \|\boldsymbol{Z}\|_{E}$ for any matrix $\boldsymbol{Z}$ when σ is ReLU or leaky ReLU. – Cai & Wang, arXiv:2006.13318, 2020

- $\|\boldsymbol{H}\|_{\mathcal{M}^\perp}$ and $\|\boldsymbol{H}\|_{E}$ are two equivalent seminorms, i.e. there exist two constants $\alpha, \beta > 0$ s.t. $\alpha\|\boldsymbol{H}\|_{\mathcal{M}^\perp} \leq \|\boldsymbol{H}\|_{E} \leq \beta\|\boldsymbol{H}\|_{\mathcal{M}^\perp}$ for any $\boldsymbol{H} \in \mathbb{R}^{d\times n}$.
$>$ $\|\sigma(\boldsymbol{Z})\|_{\mathcal{M}^\perp} \leq \|\boldsymbol{Z}\|_{\mathcal{M}^\perp}$, when σ is ReLU or leaky ReLU.

Bottlenecks of the existing theory

- Existing smoothness notions – distant to $\mathcal{M}$ and Dirichlet energy of node features – do not take the magnitude of feature vectors into account and they are not scaling free. Multiplying feature vectors by a constant will result in corresponding changes in their distance to $\mathcal{M}$ and their Dirichlet energy but do not affect graph node classification.

- Existing theory do not reveal a mechanism to control the smoothness of the learned node features when taking the activation functions into consideration.

Geometry Underlying the Input & Output of ReLU and Leaky ReLU

Geometric characterization of the effect of ReLU

- We have the decomposition $\boldsymbol{H} = \boldsymbol{H}_{\mathcal{M}} + \boldsymbol{H}_{\mathcal{M}^\perp}$ for any matrix $\boldsymbol{H} := [\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_n] \in \mathbb{R}^{d \times n}$

$$\boldsymbol{H}_{\mathcal{M}} = \sum_{i=1}^{m} \boldsymbol{H}\boldsymbol{e}_i\boldsymbol{e}_i^\top, \quad \text{and} \quad \boldsymbol{H}_{\mathcal{M}^\perp} = \sum_{i=m+1}^{n} \boldsymbol{H}\boldsymbol{e}_i\boldsymbol{e}_i^\top.$$

- Let $\boldsymbol{Z} \in \mathbb{R}^{d \times n}$ be an arbitrary matrix and $\boldsymbol{H} = \sigma(\boldsymbol{Z})$ with $\sigma(x) = \max\{0, x\}$ being ReLU.

- Proposition 1. For any $\boldsymbol{Z} = \boldsymbol{Z}_{\mathcal{M}} + \boldsymbol{Z}_{\mathcal{M}^\perp} \in \mathbb{R}^{d \times n}$, let $\boldsymbol{H} = \sigma(\boldsymbol{Z}) = \boldsymbol{H}_{\mathcal{M}} + \boldsymbol{H}_{\mathcal{M}^\perp}$ with σ being ReLU, then $\boldsymbol{H}_{\mathcal{M}^\perp}$ lies on the high dimensional sphere centered at $\boldsymbol{Z}_{\mathcal{M}^\perp}/2$ with the radius

$$r := \left(\|\boldsymbol{Z}_{\mathcal{M}^\perp}/2\|_F^2 - \langle \boldsymbol{Z}_{\mathcal{M}}^+, \boldsymbol{Z}_{\mathcal{M}}^- \rangle_F \right)^{1/2}.$$

In particular, $\boldsymbol{H}_{\mathcal{M}^\perp}$ lies inside the ball centered at $\boldsymbol{Z}_{\mathcal{M}^\perp}/2$ with radius $\|\boldsymbol{Z}_{\mathcal{M}^\perp}/2\|_F$ and hence we have $\|\boldsymbol{H}_{\mathcal{M}^\perp}\|_F \leq \|\boldsymbol{Z}_{\mathcal{M}^\perp}\|_F$. [Reduced distance to $\mathcal{M}$!]

- $\boldsymbol{Z}^+ = \max(\boldsymbol{Z}, 0)$ and $\boldsymbol{Z}^- = \max(-\boldsymbol{Z}, 0)$.

Geometric characterization of the effect of leaky ReLU

- Let $\boldsymbol{Z} \in \mathbb{R}^{d \times n}$ be an arbitrary matrix and $\boldsymbol{H} = \sigma_a(\boldsymbol{Z})$ with σ_a being leaky ReLU:

$$\sigma_a(x) = \begin{cases} x & \text{if } x \geq 0, \\ ax & \text{otherwise,} \end{cases}$$

where $0 < a < 1$ is a positive scalar.

- Proposition 2. For any $\boldsymbol{Z} = \boldsymbol{Z}_{\mathcal{M}} + \boldsymbol{Z}_{\mathcal{M}^\perp}$, let $\boldsymbol{H} = \sigma_a(\boldsymbol{Z}) = \boldsymbol{H}_{\mathcal{M}} + \boldsymbol{H}_{\mathcal{M}^\perp}$ with σ_a being leaky ReLU, then $\boldsymbol{H}_{\mathcal{M}^\perp}$ lies on the high dimensional sphere centered at $(1+a)\boldsymbol{Z}_{\mathcal{M}^\perp}/2$ with radius

$$r_a := \left(\|(1-a)\boldsymbol{Z}_{\mathcal{M}^\perp}/2\|_F^2 - (1-a)^2 \langle \boldsymbol{Z}_{\mathcal{M}}^+, \boldsymbol{Z}_{\mathcal{M}}^- \rangle_F \right)^{1/2}.$$

In particular, $\boldsymbol{H}_{\mathcal{M}^\perp}$ lies inside the high-dimensional ball centered at $(1+a)\boldsymbol{Z}_{\mathcal{M}^\perp}/2$ with radius $\|(1-a)\boldsymbol{Z}_{\mathcal{M}^\perp}/2\|_F$ and hence we see that $a\|\boldsymbol{Z}\|_{\mathcal{M}^\perp} \leq \|\boldsymbol{H}\|_{\mathcal{M}^\perp} \leq \|\boldsymbol{Z}\|_{\mathcal{M}^\perp}$.

Geometric characterization of the effect of activation functions

- σ: center $\boldsymbol{Z}_{\mathcal{M}^\perp}/2$, radius $r := \left(\|\boldsymbol{Z}_{\mathcal{M}^\perp}/2\|_F^2 - \langle \boldsymbol{Z}_{\mathcal{M}}^+, \boldsymbol{Z}_{\mathcal{M}}^- \rangle_F\right)^{1/2}$.

- σ_a: center $(1+a)\boldsymbol{Z}_{\mathcal{M}^\perp}/2$, radius $r_a := \left(\|(1-a)\boldsymbol{Z}_{\mathcal{M}^\perp}/2\|_F^2 - (1-a)^2\langle \boldsymbol{Z}_{\mathcal{M}}^+, \boldsymbol{Z}_{\mathcal{M}}^- \rangle_F\right)^{1/2}$.

- Prop. 1 and 2 imply the precise location of $\boldsymbol{H}_{\mathcal{M}^\perp}$ (or the smoothness $\|\boldsymbol{H}_{\mathcal{M}^\perp}\|_F = \|\boldsymbol{H}\|_{\mathcal{M}^\perp}$) depends on the center and the radius of the spheres. Given a fixed $\boldsymbol{Z}_{\mathcal{M}^\perp}$, the center of the spheres remains unchanged and their radii r and r_a are only affected by changes in $\boldsymbol{Z}_{\mathcal{M}}$.

- Next, we focus on analyzing how **changes in $\boldsymbol{Z}_{\mathcal{M}}$ impact** $\|\boldsymbol{H}\|_{\mathcal{M}^\perp}$, i.e. the smoothness of node features.

How changes in $Z_{\mathcal{M}}$ impact $\|H\|_{\mathcal{M}^\perp}$?

Distance to the eigenspace $\mathcal{M}$

- Prop 1 and 2 show that both ReLU and leaky ReLU reduce the distance of node features to the eigenspace $\mathcal{M}$, i.e. $\|\boldsymbol{H}\|_{\mathcal{M}^\perp} \leq \|\boldsymbol{Z}\|_{\mathcal{M}^\perp}$.

- Consider $\boldsymbol{Z}, \boldsymbol{Z}' \in \mathbb{R}^{d\times n}$ s.t. $\boldsymbol{Z}_{\mathcal{M}^\perp} = \boldsymbol{Z}'_{\mathcal{M}^\perp}$ but $\boldsymbol{Z}_{\mathcal{M}} \neq \boldsymbol{Z}'_{\mathcal{M}}$. Let $\boldsymbol{H}, \boldsymbol{H}'$ be the output of $\boldsymbol{Z}, \boldsymbol{Z}'$ via ReLU or leaky ReLU, respectively.

  $>$ We have $\|\boldsymbol{H}\|_{\mathcal{M}^\perp} \leq \|\boldsymbol{Z}\|_{\mathcal{M}^\perp}$ and $\|\boldsymbol{H}'\|_{\mathcal{M}^\perp} \leq \|\boldsymbol{Z}'\|_{\mathcal{M}^\perp}$.
  $>$ $\boldsymbol{Z}_{\mathcal{M}^\perp} = \boldsymbol{Z}'_{\mathcal{M}^\perp}$ implies that $\|\boldsymbol{Z}\|_{\mathcal{M}^\perp} = \|\boldsymbol{Z}'\|_{\mathcal{M}^\perp} \Rightarrow \|\boldsymbol{H}'\|_{\mathcal{M}^\perp} \leq \|\boldsymbol{Z}\|_{\mathcal{M}^\perp}$.

- In other words, when $\boldsymbol{Z}_{\mathcal{M}^\perp} = \boldsymbol{Z}'_{\mathcal{M}^\perp}$ is fixed, changing $\boldsymbol{Z}_{\mathcal{M}}$ to $\boldsymbol{Z}'_{\mathcal{M}}$ can not affect the fact that ReLU and leaky ReLU smooth node features. — Resonating with existing theories (Oono & Suzuki, ICLR 2019, Cai & Wang, arXiv:2006.13318).

Altering the eigenspace projection

- Let $\boldsymbol{z}$ be a vector with z_i being the feature of the i^{th} node, we consider

$$\boldsymbol{z}(\alpha) = \boldsymbol{z} - \alpha \boldsymbol{e},$$

where $\boldsymbol{e}$ is the only eigenvector of $\boldsymbol{G}$ associated with the eigenvalue 1.

- It is clear that

$$\boldsymbol{z}(\alpha)_{\mathcal{M}^\perp} = \boldsymbol{z}_{\mathcal{M}^\perp} \text{ and } \boldsymbol{z}(\alpha)_{\mathcal{M}} = \boldsymbol{z}_{\mathcal{M}} - \alpha \boldsymbol{e},$$

where we see that α only alters $\boldsymbol{z}_{\mathcal{M}}$ while preserves $\boldsymbol{z}_{\mathcal{M}^\perp}$.

• Consider a connected graph with 100 nodes with each being assigned a random degree between 2 to 10. Then we assign an initial node feature $\boldsymbol{x} \in \mathbb{R}^{100}$, sampled uniformly on the interval $[-1.5, 1.5]$, with each node feature being a scalar; we study the smoothness of node features $\boldsymbol{z}_\alpha = \boldsymbol{x} + \alpha \boldsymbol{e}$, where $\alpha \in [-1.5, 1.5]$ is the smoothness control parameter.

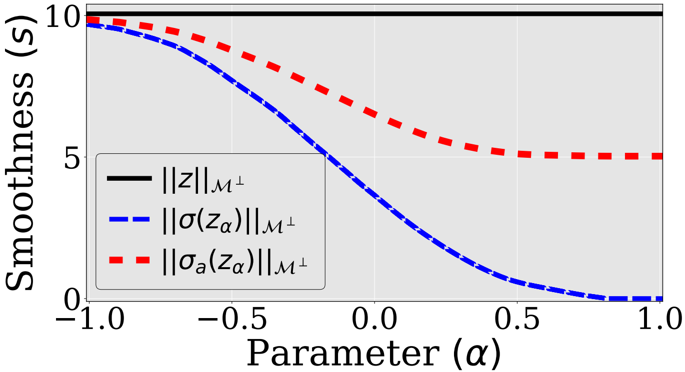


Figure: Effects of varying parameter α on the smoothness of output features $\sigma(\boldsymbol{z}_\alpha)$ and $\sigma_a(\boldsymbol{z}_\alpha)$.

Normalized Smoothness

- For the sake of simplicity, we assume the graph is connected, i.e. $m = 1$.

- Definition. Let $\boldsymbol{Z} \in \mathbb{R}^{d \times n}$ be the features over n nodes with $\boldsymbol{z}^{(i)} \in \mathbb{R}^n$ $(i = 1, \ldots, d)$ being the i^{th} row vector of $\boldsymbol{Z}$, i.e. the i^{th} dimension of the features over all nodes. Then we define the normalized smoothness of $\boldsymbol{z}^{(i)}$ as follows:

$$s(\boldsymbol{z}^{(i)}) := \frac{\|\boldsymbol{z}_{\mathcal{M}}^{(i)}\|}{\|\boldsymbol{z}^{(i)}\|} \in [0, 1],$$

where we set $s(\boldsymbol{z}^{(i)}) = 1$ when $\boldsymbol{z}^{(i)} = 0$.

Altering the eigenspace projection

- Let $\boldsymbol{z}$ be a vector with z_i being the feature of the i^{th} node, we consider

$$\boldsymbol{z}(\alpha) = \boldsymbol{z} - \alpha \boldsymbol{e},$$

where $\boldsymbol{e}$ is the only eigenvector of $\boldsymbol{G}$ associated with the eigenvalue 1.

- It is clear that

$$\boldsymbol{z}(\alpha)_{\mathcal{M}^\perp} = \boldsymbol{z}_{\mathcal{M}^\perp} \text{ and } \boldsymbol{z}(\alpha)_{\mathcal{M}} = \boldsymbol{z}_{\mathcal{M}} - \alpha \boldsymbol{e},$$

where we see that α only alters $\boldsymbol{z}_{\mathcal{M}}$ while preserves $\boldsymbol{z}_{\mathcal{M}^\perp}$.

• Consider a connected graph with 100 nodes with each being assigned a random degree between 2 to 10. Then we assign an initial node feature $\boldsymbol{x} \in \mathbb{R}^{100}$, sampled uniformly on the interval $[-1.5, 1.5]$, with each node feature being a scalar; we study the smoothness of node features $\boldsymbol{z}_\alpha = \boldsymbol{x} + \alpha \boldsymbol{e}$, where $\alpha \in [-1.5, 1.5]$ is the smoothness control parameter.

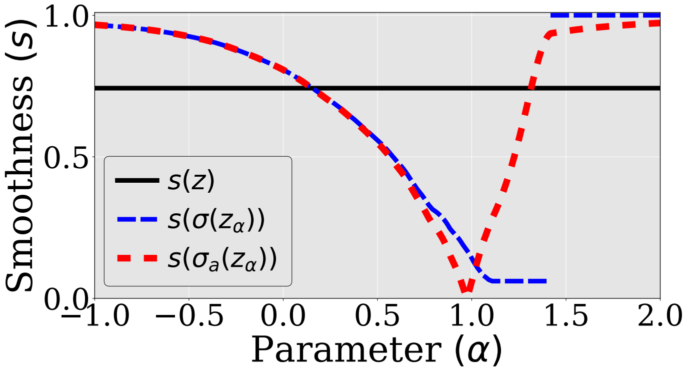


Figure: Effects of varying α on the normalized smoothness of output features $\sigma(\boldsymbol{z}_\alpha)$ and $\sigma_a(\boldsymbol{z}_\alpha)$.

Proposition 3. (ReLU) Suppose $\boldsymbol{z}_{\mathcal{M}^\perp} \neq 0$. Let $\boldsymbol{h}(\alpha) = \sigma(\boldsymbol{z}(\alpha))$ with σ being ReLU, then

$$\min_\alpha s(\boldsymbol{h}(\alpha)) = \sqrt{\frac{\sum_{x_i=\max \boldsymbol{x}} d_i}{\sum_{j=1}^n d_j}} \quad \text{and} \quad \max_\alpha s(\boldsymbol{h}(\alpha)) = 1,$$

where $\boldsymbol{x} := \tilde{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{z}$, $\max \boldsymbol{x} = \max_{1\le i\le n} x_i$, and $\tilde{\boldsymbol{D}} = \mathrm{diag}(d_1, d_2, \ldots, d_n)$. Also, the normalized smoothness $s(\boldsymbol{h}(\alpha))$ is monotone increasing as α decreases whenever $\alpha < \|\tilde{\boldsymbol{D}}^{\frac{1}{2}}\boldsymbol{u}_n\| \max \boldsymbol{x}$ and it has range $[\min_\alpha s(\boldsymbol{h}(\alpha)), 1]$.

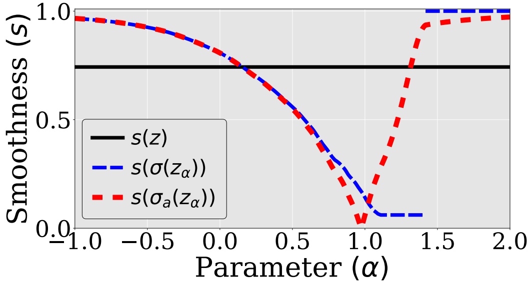


Figure: Effects of varying α on the normalized smoothness of output features $\sigma(\boldsymbol{z}_\alpha)$ and $\sigma_a(\boldsymbol{z}_\alpha)$.

Proposition 4. (Leaky ReLU) Suppose $\boldsymbol{z}_{\mathcal{M}^\perp} \neq 0$. Let $\boldsymbol{h}(\alpha) = \sigma_a(\boldsymbol{z}(\alpha))$ with σ_a being leaky ReLU, then 1) $\min_\alpha s(\boldsymbol{h}(\alpha)) = 0$, and 2) $\sup_\alpha s(\boldsymbol{h}(\alpha)) = 1$. Also, $s(\boldsymbol{h}(\alpha))$ has range $[0, 1)$.

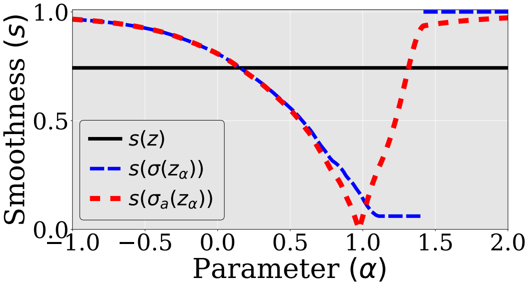


Figure: Effects of varying α on the normalized smoothness of output features $\sigma(\boldsymbol{z}_\alpha)$ and $\sigma_a(\boldsymbol{z}_\alpha)$.

Theorem 1. Suppose $\boldsymbol{z}_{\mathcal{M}^\perp} \neq 0$. Let $\boldsymbol{h}(\alpha) = \sigma(\boldsymbol{z}(\alpha))$ or $\sigma_a(\boldsymbol{z}(\alpha))$ with σ being ReLU and σ_a being leaky ReLU. Then we have $\|\boldsymbol{z}\|_{\mathcal{M}^\perp} \geq \|\boldsymbol{h}(\alpha)\|_{\mathcal{M}^\perp}$ for any $\alpha \in \mathbb{R}$. However, $s(\boldsymbol{h}(\alpha))$ can be smaller than, larger than, or equal to $s(\boldsymbol{z})$ for different values of α.

Controlling the Smoothness of Node Features

Controlling the smoothness of node features

- Our proposed smoothness control term (SCT):

$$\boldsymbol{B}_{\boldsymbol{\alpha}^l} = \sum_{i=1}^{m} \boldsymbol{\alpha}_i^l \boldsymbol{e}_i^\top,$$

where l is the layer index, $\{\boldsymbol{e}_i\}_{i=1}^m$ is the orthonormal basis of the eigenspace $\mathcal{M}$, and $\boldsymbol{\alpha}^l$ is a collection of learnable vectors $\{\boldsymbol{\alpha}_i^l\}_{i=1}^m$ with $\boldsymbol{\alpha}_i^l \in \mathbb{R}^d$ being approximated by an MLP.

- GCN-SCT:

$$\boldsymbol{H}^l = \sigma(\boldsymbol{W}^l \boldsymbol{H}^{l-1} \boldsymbol{G} + \boldsymbol{B}_{\boldsymbol{\alpha}^l}).$$

- GCNII-SCT:

$$\boldsymbol{H}^l = \sigma(((1-\alpha_l)\boldsymbol{H}^{l-1}\boldsymbol{G} + \alpha_l \boldsymbol{H}^0)((1-\beta_l)\boldsymbol{I} + \beta_l \boldsymbol{W}^l) + \boldsymbol{B}_{\boldsymbol{\alpha}^l}),$$

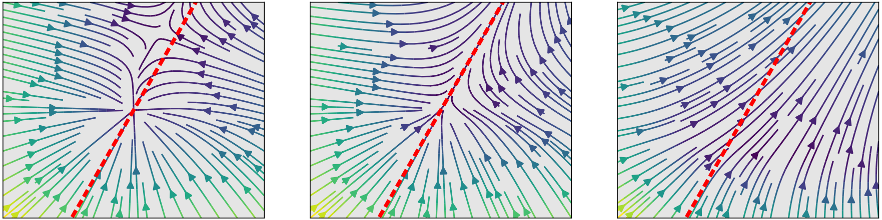
where the residual connection and identity mapping are consistent with GCNII.

Node feature trajectory

- Consider a connected graph with two nodes with 1D node features. GCL becomes

$$\boldsymbol{h}^1 = \sigma(w\boldsymbol{h}^0\boldsymbol{G} + \boldsymbol{b}_\alpha),$$

where $w = 1.2$, $\boldsymbol{h}^0, \boldsymbol{h}^1, \boldsymbol{b}_\alpha \in \mathbb{R}^2$, and $\boldsymbol{G} \in \mathbb{R}^{2\times 2}$. We select a positive definite matrix $\boldsymbol{G}$ with the largest eigenvalue 1; $\boldsymbol{G}$ is defined to be $[0.592, 0.194; 0.194, 0.908]$. Twenty initial node feature vectors $\boldsymbol{h}^0$ are sampled evenly in the domain $[-1, 1] \times [-1, 1]$.



a) $\alpha = -0.25$ b) $\alpha = 0.0$ c) $\alpha = 1.0$

Figure: Node feature trajectories, with colorized magnitude, for varying smoothness control parameter α. For classical GCN b), the node features converge to the eigenspace $\mathcal{M}$ (red dashed line).

Layers	2	4	16	32
Cora				
GCN/GCN-SCT	81.1/**82.9**	80.4/**82.8**	64.9/**71.4**	60.3/**67.2**
GCNII/GCNII-SCT	82.2/**83.8**	82.6/**84.3**	84.6/**84.8**	85.4/**85.5**
EGNN/EGNN-SCT	83.2/**84.1**	84.2/**84.5**	**85.4**/83.3	**85.3**/82.0
Citeseer				
GCN/GCN-SCT	**70.3**/69.9	67.6/**67.7**	18.3/**55.4**	25.0/**51.0**
GCNII/GCNII-SCT	68.2/**72.8**	68.9/**72.8**	72.9/**73.8**	**73.4**/**73.4**
EGNN/EGNN-SCT	72.0/**73.1**	71.9/**72.0**	72.4/**72.6**	72.3/**72.9**
PubMed				
GCN/GCN-SCT	79.0/**79.8**	76.5/**78.4**	40.9/**76.1**	22.4/**77.0**
GCNII/GCNII-SCT	78.2/**79.7**	78.8/**80.1**	80.2/**80.7**	79.8/**80.7**
EGNN/EGNN-SCT	79.2/**79.8**	79.5/**80.4**	80.1/**80.3**	80.0/**80.4**
Coauthor-Physics				
GCN/GCN-SCT	92.4/**92.6**	92.1/**92.5**	13.5/**50.9**	13.1/**43.6**
GCNII/GCNII-SCT	92.5/**94.4**	92.9/**94.2**	92.9/**93.7**	92.9/**94.1**
EGNN/EGNN-SCT	92.6/**93.9**	92.9/**94.1**	93.1/**94.0**	93.3/**93.8**
Ogbn-arxiv				
GCN/GCN-SCT	70.4/**72.1**	71.7/**72.7**	70.6/**72.3**	68.5/**72.3**
GCNII/GCNII-SCT	70.1/**72.0**	71.4/**72.1**	71.5/**72.4**	70.5/**72.1**
EGNN/EGNN-SCT	68.4/**68.5**	71.1/**71.3**	72.7/**72.8**	**72.7**/72.3

Table: Test accuracy for models of varying depth on citation networks with benchmark splits. (Unit:%)

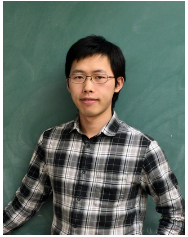
Cornell	Texas	Wisconsin	Chameleon
52.70/**55.95** (0.007/0.018)	52.16/**62.16** (0.007/0.008)	45.88/**54.71** (0.007/0.008)	28.18/**38.44** (0.006/0.007)
74.86/**75.41** (0.020/0.020)	69.46/**83.34** (0.031/0.020)	74.12/**86.08** (0.020/0.015)	60.61/**64.52** (0.015/0.013)

Table: Mean test accuracy results and average computational time per epoch (in the parenthesis) for the WebKB and WikipediaNetwork datasets with fixed 48/32/20% splits. First row: GCN/GCN-SCT. Second row: GCNII/GCNII-SCT. (Unit:% (second))

References

Shih-Hsin Wang*, Justin Baker*, Cory Hauck, and Bao Wang, The Effects of Activation Functions on the Over-smoothing Issue of Graph Convolutional Networks, preprint, 2023.

Implicit Graph Neural Networks: A Monotone Operator Viewpoint

Justin Baker*, Qingsong Wang*, Cory Hauck, and Bao Wang, Implicit Graph Neural Networks: A Monotone Operator Viewpoint, ICML, 2023.

Implicit GNNs

- Implicit GNN (IGNN)

$$\boldsymbol{Z}^{(k+1)} = \sigma\big(\boldsymbol{W}\boldsymbol{Z}^{(k)}\boldsymbol{G} + g_{\boldsymbol{B}}(\boldsymbol{X})\big), \quad \text{for } k = 0, 1, 2, \cdots,$$

where $g_{\boldsymbol{B}}$ is a function parameterized by $\boldsymbol{B}$, e.g. $g_{\boldsymbol{B}}(\boldsymbol{X}) = \boldsymbol{B}\boldsymbol{X}\boldsymbol{G}$ with $\boldsymbol{B} \in \mathbb{R}^{d \times d}$.

- Finding the fixed point $\boldsymbol{Z}^*$ as the representation of input graph.

Gu et al. Implicit graph neural networks, NeurIPS 2020.

Issue 1: Well-posedness of IGNN limits its expressivity of IGNN

- Well-posedness, i.e. the fixed point exists and is unique

$$\lambda_1(|\boldsymbol{W}|) < 1.$$

Or all eigenvalues of $\boldsymbol{W}$ are less than one in magnitude.

- The selection of $\boldsymbol{W}$ is limited, limiting the expressivity of IGNN.

Notice that all eigenvalues of $\boldsymbol{G} = \hat{\boldsymbol{A}}$ are in $[-1, 1]$ with $\lambda_1(\boldsymbol{G}) = 1$.

Issue 2: When IGNN learns long-range dependencies (LRD)

- Learning LRD: each node can aggregate information from the nodes that are far apart.

- To learn LRD, $\lambda_1(|\boldsymbol{W}|)$ needs to be close to one in magnitude; otherwise, the Picard iteration converges too fast, and each node only aggregates nearby nodes' features.

- Training IGNN with $\lambda_1(|\boldsymbol{W}|) \to 1$, starting from random initialization, may not happen.

- Picard iteration converges slowly when $\lambda_1(|\boldsymbol{W}|) \to 1$

A monotone operator theory viewpoint of IGNN

- Notice that $\boldsymbol{Z}^{(k+1)} = \sigma\big(\boldsymbol{W}\boldsymbol{Z}^{(k)}\boldsymbol{G} + g_{\boldsymbol{B}}(\boldsymbol{X})\big)$ can be rewritten as the following vectorized equation

$$\mathrm{vec}(\boldsymbol{Z}^{(k+1)}) = \sigma\big(\boldsymbol{G}^\top \otimes \boldsymbol{W}\mathrm{vec}(\boldsymbol{Z}^{(k)}) + \mathrm{vec}(g_{\boldsymbol{B}}(\boldsymbol{X}))\big), \tag{1}$$

where $\boldsymbol{G}^\top \otimes \boldsymbol{W}$ denotes the Kronecker product between $\boldsymbol{G}$ and $\boldsymbol{W}$.

- Finding a fixed point of (1) is equivalent to solving the monotone inclusion problem

$$\text{find } 0 \in (\mathcal{F}+\mathcal{G})(\mathrm{vec}(\boldsymbol{Z})^*),$$

where

$$\mathcal{F}(\mathrm{vec}(\boldsymbol{Z})) = (\boldsymbol{I} - \boldsymbol{G}^\top \otimes \boldsymbol{W})\mathrm{vec}(\boldsymbol{Z}) - \mathrm{vec}(g_{\boldsymbol{B}}(\boldsymbol{X})) \ \text{ and } \ \mathcal{G} = \partial f,$$

where f is a convex closed proper (CCP) function such that

$$\sigma(x) = \mathrm{prox}_f^1(x) = \operatorname*{argmin}_z \left\{ \frac{1}{2}\|x-z\|^2 + f(z) \right\}.$$

- Notice that when σ is ReLU, then $\sigma = \mathrm{prox}_f^\alpha$ for $\forall \alpha > 0$ with f being the indicator of the positive octant, i.e. $f(x) = I\{x \geq 0\}$.

Well-posedness of MIGNN

- MIGNN: monotone operator theory viewpoint of IGNN.
- The fixed point $\boldsymbol{Z}^*$ exists and is unique if $\mathcal{F}$ is strongly monotone.
- If $\boldsymbol{I} - \boldsymbol{G}^\top \otimes \boldsymbol{W} \succeq m\boldsymbol{I}$ for some $m > 0$, then $\mathcal{F}$ is strongly monotone.

Monotone parameterization of MIGNN: Enhancing expressivity of IGNN

- We consider the following MIGNN model

$$\boldsymbol{Z}^{(k+1)} = \sigma\big(\boldsymbol{W}\boldsymbol{Z}^{(k)}\boldsymbol{G} + g_{\boldsymbol{B}}(\boldsymbol{X})\big).$$

- We let $\boldsymbol{G} = \frac{\boldsymbol{L}}{2}$ where $\boldsymbol{L} := \boldsymbol{D}^{-1/2}(\boldsymbol{D} - \boldsymbol{A})\boldsymbol{D}^{-1/2}$ is the normalized Laplacian.

- We parameterize $\boldsymbol{W}$ with the following monotone parameterization

$$\boldsymbol{W} = (1-m)\boldsymbol{I} - \boldsymbol{C}\boldsymbol{C}^\top + \boldsymbol{F} - \boldsymbol{F}^\top,$$

where $\boldsymbol{C}, \boldsymbol{F} \in \mathbb{R}^{d\times d}$ are arbitrary matrices, and $m > 0 \in \mathbb{R}$.

Monotone parameterization of MIGNN: Enhancing expressivity of IGNN

- The monotone parameterization guarantees the operator $\mathcal{F}$ to be strongly monotone.

- The monotone parameterization allows the eigenvalues of $\boldsymbol{W}$ to be much less than -1, which is more flexible than IGNN.

Orthogonal parameterization of MIGNN: Stabilizing learning LRD

- Consider the following MIGNN model

$$\mathbf{Z}^{(k+1)} = \sigma\big(\mathbf{W}\mathbf{Z}^{(k)}\mathbf{G} + g_{\mathbf{B}}(\mathbf{X})\big).$$

- We parameterize $\mathbf{W}$ using the following scaled Cayley map

$$\mathbf{W} = \phi(\gamma)(\mathbf{I} - \mathbf{S})(\mathbf{I} + \mathbf{S})^{-1},$$

where $\phi(\cdot)$ is the sigmoid function. $\mathbf{S} = \mathbf{C} - \mathbf{C}^\top$ is a skew-symmetric matrix with $\mathbf{C} \in \mathbb{R}^{d\times d}$ an arbitrary matrix.

- Notice that the matrix $(\mathbf{I} - \mathbf{S})(\mathbf{I} + \mathbf{S})^{-1}$ is orthogonal.

Finding the fixed point of the equilibrium equation

- Picard iteration may not converge for MIGNN with monotone parameterization, i.e., $\boldsymbol{W} = (1-m)\boldsymbol{I} - \boldsymbol{C}\boldsymbol{C}^\top + \boldsymbol{F} = \boldsymbol{F}^\top$.

- Picard iteration suffers from slow convergence for MIGNN with orthogonal parameterization, i.e., $\boldsymbol{W} = (\boldsymbol{I} - \boldsymbol{S})(\boldsymbol{I} + \boldsymbol{S})^{-1}$ with $\boldsymbol{S} = \boldsymbol{C} - \boldsymbol{C}^\top$.

- Need new algorithms to find the fixed point of MIGNN.

Forward-backward splitting (FB): MIGNN with monotone parameterization

- Finding the fixed point of MIGNN, $\boldsymbol{Z}^{(k+1)} = \sigma\big(\boldsymbol{W}\boldsymbol{Z}^{(k)}\boldsymbol{G} + g_{\boldsymbol{B}}(\boldsymbol{X})\big)$, with monotone parameterization

$$\boldsymbol{Z}^{(k+1)} := F_{\alpha}^{\mathrm{FB}}(\boldsymbol{Z}^{(k)}) := \mathrm{prox}_f^{\alpha}\left(\boldsymbol{Z}^{(k)} - \alpha \cdot \left(\boldsymbol{Z}^{(k)} - \boldsymbol{W}\boldsymbol{Z}^{(k)}\boldsymbol{G} - g_{\boldsymbol{B}}(\boldsymbol{X})\right)\right),$$

where $\alpha > 0$ is an appropriate constant.

- $$\boldsymbol{Z}^{(k+1/2)} = \boldsymbol{Z}^{(k)} - \alpha \cdot \left(\boldsymbol{Z}^{(k)} - \boldsymbol{W}\boldsymbol{Z}^{(k)}\boldsymbol{G} - g_{\boldsymbol{B}}(\boldsymbol{X})\right)$$
$$\boldsymbol{Z}^{(k+1)} = \mathrm{prox}_f^{\alpha}(\boldsymbol{Z}^{(k+1/2)}).$$

- Resulting the model MIGNN-Mon.

Peaceman-Rachford splitting (PR): MIGNN with orthogonal parameterization

- PR finds the solution $\boldsymbol{Z}^*$ of the MIGNN by letting

$$\boldsymbol{Z}^* = \mathrm{prox}_f^{\alpha}(\boldsymbol{U}^*),$$

where $\boldsymbol{U}^*$ is the solution of the following fixed point iterations:

$$\mathrm{vec}(\boldsymbol{U}^{(k+1)}) = F_{\alpha}^{\mathrm{PR}}(\mathrm{vec}(\boldsymbol{U}^{(k)})) := \mathcal{C}_{\mathcal{F}}\mathcal{C}_{\mathcal{G}}(\mathrm{vec}(\boldsymbol{U}^{(k)})),$$

where

$$\mathcal{R}_{\mathcal{T}} = (\mathcal{I} + \alpha\mathcal{T})^{-1},$$

and

$$\mathcal{C}_{\mathcal{T}} = 2\mathcal{R}_{\mathcal{T}} - \mathcal{I}.$$

Peaceman-Rachford splitting (PR): MIGNN with orthogonal parameterization

- Let $\boldsymbol{u}^k := \mathrm{vec}(\boldsymbol{U}^{(k)})$, then we can formulate PR as follows

$$\boldsymbol{u}^{k+1} := F_\alpha^{\mathrm{PR}}(\boldsymbol{u}^k) = 2\boldsymbol{V}\Big(2\mathrm{prox}_f^\alpha(\boldsymbol{u}^k) - \boldsymbol{u}^k + \alpha\,\mathrm{vec}(g_{\boldsymbol{B}}(\boldsymbol{X}))\Big) - 2\mathrm{prox}_f^\alpha(\boldsymbol{u}^k) + \boldsymbol{u}^k,$$

where the matrix $\boldsymbol{V} := (\boldsymbol{I} + \alpha(\boldsymbol{I} - \boldsymbol{G}^\top \otimes \boldsymbol{W}))^{-1}$ and $\boldsymbol{u}^0$ is the zero vector.

- Computing $\boldsymbol{V}(\boldsymbol{x}^k)$ is expensive:

  $>$ Bartels–Stewart algorithm, which requires diagonalizing the matrix $\boldsymbol{G}$ and $\boldsymbol{W}$.

PR with Neumann series approximation

- Notice that

$$\boldsymbol{V}(\boldsymbol{u}^k) = (\boldsymbol{I} + \alpha(\boldsymbol{I} - \boldsymbol{G}^\top \otimes \boldsymbol{W}))^{-1}(\boldsymbol{u}^k)$$
$$= \frac{1}{1+\alpha}\left(\boldsymbol{I} - \frac{\boldsymbol{G}^\top \otimes \boldsymbol{W}}{1+1/\alpha}\right)^{-1}(\boldsymbol{u}^k)$$
$$= \frac{1}{1+\alpha}\sum_{i=0}^{\infty} \frac{\mathrm{vec}\left(\boldsymbol{W}^i \boldsymbol{U}^{(k)} \boldsymbol{G}^i\right)}{(1+1/\alpha)^i}.$$

- K-th order Neumann series approximation of $\boldsymbol{V}(\boldsymbol{u}^k)$:

$$\boldsymbol{N}_K(\mathrm{vec}(\boldsymbol{U}^k)) := \frac{1}{1+\alpha}\sum_{i=0}^{K} \frac{\mathrm{vec}\left(\boldsymbol{W}^i \boldsymbol{U}^k \boldsymbol{G}^i\right)}{(1+1/\alpha)^i}.$$

- K-th order Neumann series approximation of PR iteration

$$\boldsymbol{u}^{k+1} := \tilde{F}_\alpha^{\mathrm{PR,K}}(\boldsymbol{u}^k) = 2\boldsymbol{N}_K\Big(2\mathrm{prox}_f^\alpha(\boldsymbol{u}^k) - \boldsymbol{u}^k + \alpha\,\mathrm{vec}(g_{\boldsymbol{B}}(\boldsymbol{X}))\Big) - 2\mathrm{prox}_f^\alpha(\boldsymbol{u}^k) + \boldsymbol{u}^k.$$

MIGNN with diffusion convolution

- We can set $\boldsymbol{G}$ to be the combination of higher powers of $\hat{\boldsymbol{A}}$ or $\boldsymbol{L}$, making each node to aggregate multi-hops neighbors' features in each iteration.

- We let $\boldsymbol{G} = \tilde{\boldsymbol{D}}^{-1/2}(\boldsymbol{A} + \cdots + \boldsymbol{A}^P)\tilde{\boldsymbol{D}}^{-1/2}$ for any positive integer P, where $\tilde{\boldsymbol{D}}$ is the degree matrix with $\tilde{D}_{ii} = \sum_{j=1}^{n}\sum_{k=1}^{P}(\boldsymbol{A}^k)_{ij}$.

- MIGNN with P-th order diffusion matrix $\boldsymbol{G}$

$$\boldsymbol{Z} = \sigma(\boldsymbol{W}\boldsymbol{Z}\tilde{\boldsymbol{D}}^{-1/2}(\boldsymbol{A} + \boldsymbol{A}^2 + \cdots + \boldsymbol{A}^P)\tilde{\boldsymbol{D}}^{-1/2} + g_{\boldsymbol{B}}(\boldsymbol{X})).$$

- We denote the model as MIGNN-NKDP when it is implemented by using the P-th order diffusion and the K-th order Neumann series approximated PR iteration.

Directed chain classification

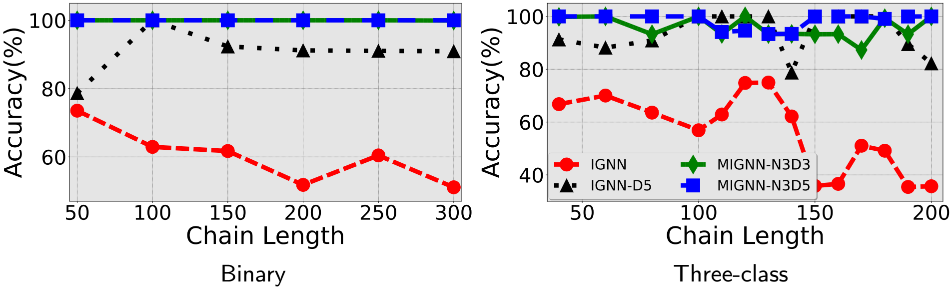


Figure: The accuracy of IGNN and MIGNN for classifying directed chains of different lengths.

Directed chain classification

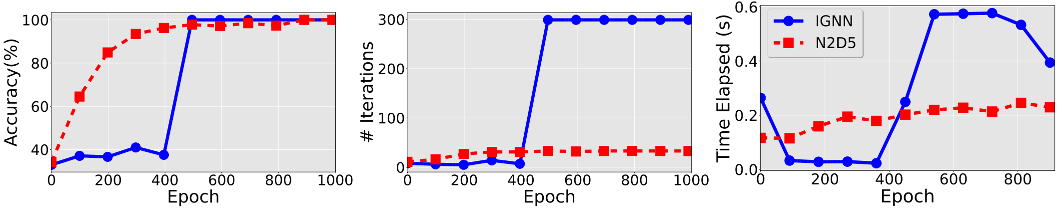


Figure: MIGNN-N2D5 vs. IGNN for three class chains classification (length: 140).

Graph node classification: Citation networks

Datasets	Cora	Citeseer	Pubmed
Geom-GCN	85.27	**77.99**	**90.05**
GCNII	**88.49**	77.08	89.57
APPNP	85.09	75.73	79.73
GCN+GDC	83.58	73.35	78.72
GIND	88.25	76.81	89.22
IGNN	85.80	75.24	87.66
EIGNN (Ours)	85.89	75.31	87.92
MIGNN-Mon (Ours)	86.82	76.59	88.00
MIGNN-N5D1	87.04	74.91	83.55

Table: Node classification mean accuracy (%) for 10-fold cross-validation.

Graph classification: bioinformatics-related tasks

Datasets	MUTAG	PTC	COX2	PROTEINS	NCI1
# graphs/Avg # nodes	188/17.9	344/25.5	467/41.2	1113/39.1	4110/29.8
WL	84.1 ± 1.9	58.0 ± 2.5	83.2 ± 0.2	74.7 ± 0.5	84.5 ± 0.5
DCNN	67.0	56.6	—	61.3	62.6
DGCNN	85.8	58.6	—	75.5	74.4
GIN	89.4 ± 5.6	64.6 ± 7.0	—	76.2 ± 3.4	82.7 ± 1.7
FDGNN	88.5 ± 3.8	63.4 ± 5.4	83.3 ± 2.9	76.8 ± 2.9	77.8 ± 1.6
IGNN	76.0 ± 13.4	60.5 ± 6.4	79.7 ± 3.4	76.5 ± 3.4	73.5 ± 1.9
GIND	89.3 ± 7.4	66.9 ± 6.6	84.8 ± 4.2	77.2 ± 2.9	78.8 ± 2.9
GSN	92.2 ± 7.5	68.2 ± 7.2	—	76.6 ± 5.0	83.5 ± 2.0
SIN	—	—	—	76.5 ± 3.3	82.8 ± 2.2
CIN	92.7 ± 6.1	68.2 ± 5.6	—	77.0 ± 4.3	83.6 ± 1.4
MIGNN-Mon	81.8 ± 9.1	72.6 ± 6.7	85.0 ± 5.3	77.9 ± 3.4	73.6 ± 2.0
MIGNN-N1D1	86.1 ± 9.1	70.9 ± 6.5	86.5 ± 2.8	79.0 ± 3.3	78.4 ± 1.2
MIGNN-N3D1	91.4 ± 7.5	71.2 ± 3.2	88.2 ± 4.1	80.1 ± 3.8	80.8 ± 1.81

Table: Graph classification mean accuracy (%) $\pm$ standard deviation for 10-fold cross-validation.

Graph classification: bioinformatics-related tasks

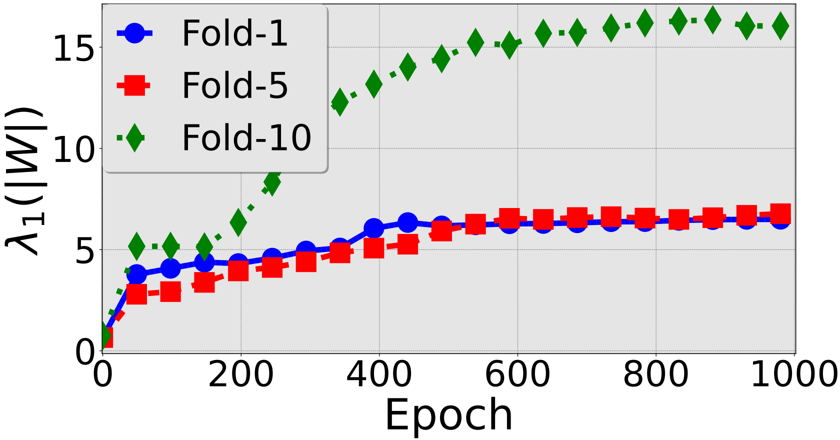


Figure: $\lambda_1(|\boldsymbol{W}|)$ of MIGNN-Mon vs. Epoch on MUTAG.

Summary

I. How activation functions affect the smoothness of node features.

- I.1 Geometric characterization
- I.2 Smoothness control

II. Monotone operator-based implicit graph neural networks

- I.1 Stable and accurate graph deep learning
- I.2 Fast convergence and learning long-range dependencies