Break the Ceiling: Stronger Multi-scale Deep Graph Convolutional Networks

Sitao Luan & **Mingde Zhao**, Xiao-Wen Chang, Doina Precup

Abstract

We analyze how existing Graph Convolutional Networks (GCNs) have limited expressive power due to the constraint of the activation functions and their architectures. We generalize spectral graph convolution and deep GCN in block Krylov subspace forms and devise 2 architectures, both with the potential to be extended deeper but each making use of the multi-scale information differently. On several node classification tasks, with or without validation set, the 2 proposed architectures achieve SOTA performance.

Problem of GCN: activation function

Deepened GCN can be described as
$\mathbf{Y}' = \mathrm{L}\ \mathrm{ReLU}(\cdots \mathrm{L}\ \mathrm{ReLU}(\mathrm{L}\ \mathrm{ReLU}(\mathrm{L}\mathbf{X}W_0)\ W_1)\ W_2 \cdots)\ W_n$
where $L = \tilde{D}^{-1/2}\tilde{A}\ \tilde{D}^{-1/2}$, $\tilde{A}$=$A + I$ and $\tilde{D}$=$\sum_j \tilde{A}_{ij}$, $A \in \mathbb{R}^{N\times N}$ is the adjacency matrix.

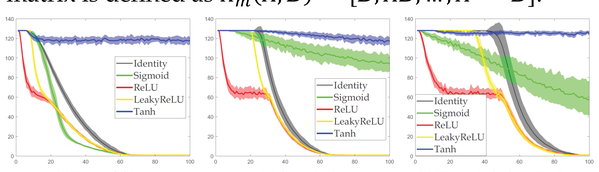
Theorem 1. Suppose the graph $\mathcal{G}$ has k connected components and the diffusion operator L is defined as above. Let $\mathbf{X}$ be any block vector sampled from a continuous distribution in the space $\mathbb{R}^{N\times F}$ and $\{W_0, W_1, \dots, W_n\}$ be any set of parameter matrices, If $\mathcal{G}$ has no bipartite components, then $\lim_{n\to\infty} rank(Y') \le k$.

Theorem 2. Suppose we sample $x, y \in \mathbb{R}^N$ from a continuous distribution and we apply pointwise $Tanh$ on $[x, y]$, then
$\mathbb{P}\big(rank\big(Tanh([x,y])\big) \le rank([x,y]) \big| x, y \in \mathbb{R}^N\big) = 0$

Problem of GCN: architecture

Definition Let $\mathbb{S}$ be a block vector subspace of $\mathbb{R}^{F\times F}$. Given a set of block vectors $\{\boldsymbol{X}_k\}_{k=1}^m \subset \mathbb{R}^{N\times F}$, the $\mathbb{S}$-span of $\{\boldsymbol{X}_k\}_{k=1}^m$is defined as $span^{\mathbb{S}}\{\boldsymbol{X_1}, \boldsymbol{X_2}, \dots, \boldsymbol{X_m}\} \coloneqq \{\sum_{k=1}^m X_k C_k : C_k \in \mathbb{S}\}$.

The order- m block Krylov subspace *w.r.t.* $A \in \mathbb{R}^{N\times N}, \boldsymbol{B} \in \mathbb{R}^{N\times F}$ and $\mathbb{S}$ is defined as $\mathcal{K}_m^{\mathbb{S}}(A, \boldsymbol{B}) \coloneqq span^{\mathbb{S}}\{\boldsymbol{X_1}, \boldsymbol{X_2}, \dots, \boldsymbol{X_m}\}$. The corresponding block Krylov matrix is defined as $K_m(A, \boldsymbol{B}) \coloneqq [\boldsymbol{B}, A\boldsymbol{B}, \dots, A^{m-1}\boldsymbol{B}]$.



(a) Deep GCN (b) Snowball (c) Truncated Block Krylov

Figure 1: Changes in the number of independent features with the increment of network depth

Block Krylov Forms

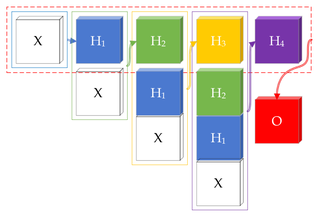
Graph convolution with any well-defined analytic spectral filter g defined on L can be written as the product of a block Krylov matrix with a specific parameter matrix.

$$g(L)\boldsymbol{X} = \sum_{n=0}^{\infty} \frac{g^{(n)}(0)}{n!} L^n \boldsymbol{X} = [\boldsymbol{X}, L\boldsymbol{X}, L^2\boldsymbol{X}, \cdots][\frac{g^{(n)}(0)}{0!} I_F, \frac{g^{(n)}(0)}{1!} I_F, \frac{g^{(n)}(0)}{2!} I_F, \cdots]^T = A'B'$$

where $\rho(L) < R$, R is the convergence radius and $\rho(L)$ denotes the spectrum radius of L. $Range(A'B') = Range(A')$ and exists a smallest m *s.t.* $span^{\mathbb{S}}\{\boldsymbol{X}, L\boldsymbol{X}, L^2\boldsymbol{X} \dots\} = span^{\mathbb{S}}\{\boldsymbol{X}, L\boldsymbol{X}, \dots, L^{m-1}\boldsymbol{X}\}$. The hidden layer can be written as $g(L)\boldsymbol{X}W' = K_m(L, \boldsymbol{X})W^{\mathbb{S}}$.

We need to concatenate multi-scale information in each hidden layer so that multi-hop neighborhood features , especially some distinct features for each node.

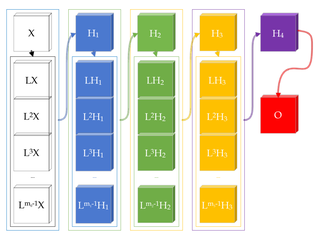
Proposed Architecture: Snowball



$\boldsymbol{H_0} = X, H_{l+1} = f(L\{H_0, H_1, \dots, H_l\}W_l), l = 0,1, \dots, n-1$
$\boldsymbol{C} = g(\{H_0, H_1, \dots, H_n\}W_n)$
Output = softmax$(L^p C W_C)$
$W^l \in \mathbb{R}^{(\sum_{i=0}^{l} F_i)\times F_{l+1}}, W_n \in \mathbb{R}^{(\sum_{i=0}^{n} F_i)\times F_C}, W_C \in \mathbb{R}^{F_C\times F_O}$.
f, g are activation functions.
$\mathbf{C}$ is a classifier of any kind, *e.g.* neural networks or even an identity layer $\mathbf{C} = [H_0, H_1, \dots, H_l]$.
$p \in \{0,1\}$

Snowball GCN stacks multi-scale information incrementally.

Proposed Architecture: Truncated Krylov



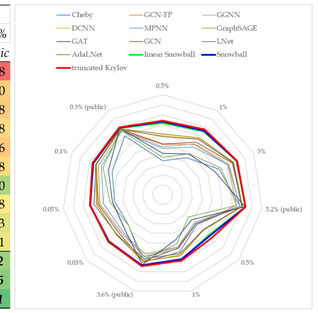
$\boldsymbol{H_0} = \boldsymbol{X}, \boldsymbol{H_{l+1}} = f(L\{\boldsymbol{H_l}, \dots, L^{m_l-1}\boldsymbol{H_l}\}W_l), l = 0,1,2, \dots, n-1$
$\boldsymbol{C} = g(\boldsymbol{H_n}W_n)$
Output = softmax$(L^p \boldsymbol{C} W_C)$
$W^l \in \mathbb{R}^{(m_l F_l)\times F_{l+1}}, W_n \in \mathbb{R}^{F_n\times F_C}, W_C \in \mathbb{R}^{F_C\times F_O}$;
f, g are activation functions.
$\boldsymbol{C}$ is a classifier of any kind, e.g. neural networks or even an identity layer $\mathbf{C} = [H_0, H_1, \dots, H_l]$.
$p \in \{0,1\}$

In truncated Krylov, the local information will not be diluted because in each layer l, we start the concatenation from $L^0 H_l$. It can be shown that snowball GCN with linear activation function is equivalent to a one-layer truncated block Krylov network. The 2 architectures have some inherent connections for feature extractions.

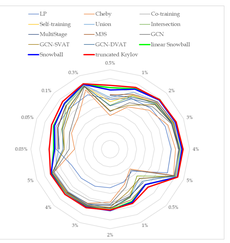
Truncated Krylov GCN stacks multi-scale information directly in each layer.

Experiments: Node Classification

Algorithms	Cora				CiteSeer			PubMed			
	0.5%	1%	3%	5.2% *public*	0.5%	1%	3.6% *public*	0.03%	0.05%	0.1%	0.3% *public*
Cheby	33.9	44.2	62.1	78.0	45.3	59.4	70.1	45.3	48.2	55.2	69.8
GCN-FP	50.5	59.6	71.7	74.6	43.9	54.3	61.5	56.2	63.2	70.3	76.0
GGNN	48.2	60.5	73.1	77.6	44.3	56.0	64.6	55.8	63.3	70.4	75.8
DCNN	59.0	66.4	76.7	79.7	53.1	62.2	69.4	60.9	66.7	73.1	76.8
MPNN	46.5	56.7	72.0	78.0	41.8	54.3	64.0	53.9	59.6	67.3	75.6
GraphSAGE	37.5	49.0	64.2	74.5	33.8	51.0	67.2	45.4	53.0	65.4	76.8
GAT	41.4	48.6	56.8	83.0	38.2	46.5	72.5	50.9	50.4	59.6	79.0
GCN	50.9	62.3	76.5	80.5	43.6	55.3	68.7	57.9	64.6	73.0	77.8
LNet	58.1	66.1	77.3	79.5	53.2	61.3	66.2	60.4	68.8	73.4	78.3
AdaLNet	60.8	67.5	77.7	80.4	53.8	63.3	68.7	61.0	66.0	72.8	78.1
linear Snowball	***71.7***	***75.2***	***81.5***	***83.7***	***61.2***	***67.3***	***73.4***	***71.7***	***73.2***	***75.7***	***79.2***
Snowball	***72.9***	***76.2***	***81.1***	***83.3***	***62.4***	***67.0***	***73.2***	***70.9***	***73.0***	***76.1***	***79.5***
truncated Krylov	***74.1***	***77.6***	***81.8***	***83.5***	***65.3***	***68.2***	***74.2***	***71.5***	***73.2***	***77.0***	***80.1***



Algorithms	Cora						CiteSeer						PubMed			
	0.5%	1%	2%	3%	4%	5%	0.5%	1%	2%	3%	4%	5%	0.03%	0.05%	0.1%	0.3%
LP	56.4	62.3	65.4	67.5	69.0	70.2	34.8	40.2	43.6	45.3	46.4	47.3	61.4	66.4	65.4	66.8
Cheby	38.0	52.0	62.4	70.8	74.1	77.6	31.7	42.8	59.9	66.2	68.3	69.3	40.4	47.3	51.2	72.8
Co-training	56.6	66.4	73.5	75.9	78.9	80.8	47.3	55.7	62.1	62.5	64.5	65.5	62.2	68.3	72.7	78.2
Self-training	53.7	66.1	73.8	77.2	79.4	80.0	43.3	58.1	68.2	69.8	70.4	71.0	51.9	58.7	66.8	77.0
Union	58.5	69.9	75.9	78.5	80.4	81.7	46.3	59.1	66.7	66.7	67.6	68.2	58.4	64.0	70.7	79.2
Intersection	49.7	65.0	72.9	77.1	79.4	80.2	42.9	59.1	68.6	70.1	70.8	71.2	52.0	59.3	69.7	77.6
MultiStage	61.1	63.7	74.4	76.1	77.2		53.0	57.8	63.8	68.0	69.0		57.4	64.3	70.2	
M3S	61.5	67.2	75.6	77.8	78.0		56.1	62.1	66.4	70.3	70.5		59.2	64.4	70.6	
GCN	42.6	56.9	67.8	74.9	77.6	79.3	33.4	46.5	62.6	66.9	68.7	69.6	46.4	49.7	56.3	76.6
GCN-SVAT	43.6	53.9	71.4	75.6	78.3	78.5	47.0	52.4	65.8	68.6	69.5	70.7	52.1	56.9	63.5	77.2
GCN-DVAT	49	61.8	71.9	75.9	78.4	78.6	51.5	58.5	67.4	69.2	70.8	71.3	53.3	58.6	66.3	77.3
linear Snowball	***69.5***	***74.1***	***79.4***	***80.4***	***81.3***	***82.4***	***56.8***	***64.3***	***68.8***	***71.0***	***72.2***	***72.3***	***64.1***	***69.5***	***72.9***	***79.3***
Snowball	***67.2***	***73.5***	***78.5***	***80.0***	***81.5***	***82.3***	***56.7***	***65.0***	***69.5***	***71.1***	***72.3***	***72.6***	***62.9***	***68.3***	***73.3***	***79.5***
truncated Krylov	***72.2***	***75.7***	***79.2***	***80.9***	***82.5***	***82.8***	***60.9***	***66.2***	***69.0***	***71.8***	***72.4***	***73.2***	***69.1***	***71.8***	***76.1***	***80.1***



(a) Linear Snowball (b) Snowball (c) Truncated Krylov

Figure 3: t-SNE for the extracted features trained on Cora (7 classes) public (5.2%).

We use RMSprop for tests with validation and Adam for those without. For each case, all the compared methods are fine-tuned.

The performance of tuned instances of the proposed architectures is better than the tuned baselines in EVERY test case.

Surprisingly better performance can be observed in smaller splits: the proposed architectures are particularly well-performing while the train split is small.

References

[1] A. Frommer, K. Lund, and D. B. Szyld. Block Krylov subspace methods for functions of matrices. *Electronic Transactions on Numerical Analysis*, 47:100–126, 2017.

[2] M. H. Gutknecht and T. Schmelzer. The block grade of a block krylov space. *Linear Algebra and its Applications*, 430(1):174–185, 2009.

[3] R. Liao, Z. Zhao, R. Urtasun, and R. S. Zemel. Lanczosnet: Multi-scale deep graph convolutional networks. *ICLR*, 2019.

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