Simplifying Graph Convolutional Networks

ICML'19(?), Felix Wu et al.

Hoang NT

Murata Laboratory
Tokyo Tech

2019/04/16

Overview

1. GCN Introduction
   - Problems
   - Approaches
2. Simplifying GCN
   - Simplest Formulation
3. Experiments & Results
   - Datasets
   - Results
   - Conclusion

GCN Introduction

There are two main problems:

Popular Problems on Graph-Structured Data

There are two main problems:

- Answering questions on vertices.

Popular Problems on Graph-Structured Data

There are two main problems:

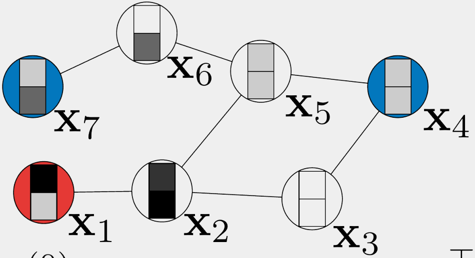
- Answering questions on vertices.
- Answering questions about the graph.

There are two main problems:

- Answering questions on vertices.
- Answering questions about the graph.

This paper is about classifying vertices

Given a graph structured data $\mathcal{G} = (A, \mathcal{X}, \mathcal{C}, J_{\text{train}}, \mathcal{C})$, we want to find: $\mathcal{C}_i \ \forall i \in V(A) - J_{\text{train}}$.



$$\mathbf{H}^{(0)} = \mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^\top$$

Figure: Input Graph Example

Graph representation learning:

Graph representation learning:

- Spectral Clustering [6], Modularity [5]

Approach: Only Use Graph Structure

Graph representation learning:

- Spectral Clustering [6], Modularity [5]
- Label Propagation [11]

Approach: Only Use Graph Structure

Graph representation learning:

- Spectral Clustering [6], Modularity [5]
- Label Propagation [11]
- Deepwalk [4], node2vec [1]

Approach: Only Use Graph Structure

Graph representation learning:

- Spectral Clustering [6], Modularity [5]
- Label Propagation [11]
- Deepwalk [4], node2vec [1]
- (and many more)

Graph representation learning:

- Spectral Clustering [6], Modularity [5]
- Label Propagation [11]
- Deepwalk [4], node2vec [1]
- (and many more)

Common theme: Graph

Approach: Only Use Graph Structure

Graph representation learning:

- Spectral Clustering [6], Modularity [5]
- Label Propagation [11]
- Deepwalk [4], node2vec [1]
- (and many more)

Common theme: Graph $\rightarrow$ Surrogate Structure

Approach: Only Use Graph Structure

Graph representation learning:

- Spectral Clustering [6], Modularity [5]
- Label Propagation [11]
- Deepwalk [4], node2vec [1]
- (and many more)

Common theme: Graph $\rightarrow$ Surrogate Structure $\rightarrow$ Manifold Learning

Graph representation learning:

- Spectral Clustering [6], Modularity [5]
- Label Propagation [11]
- Deepwalk [4], node2vec [1]
- (and many more)

Common theme: Graph $\rightarrow$ Surrogate Structure $\rightarrow$ Manifold Learning $\rightarrow$ Node Representaions.

Approach: Graph Structure + Other Information

Using $\mathcal{G} = (A, \mathcal{X}, \mathcal{C}, J_{\text{train}}, \mathcal{C})$

Using $\mathcal{G} = (A, \mathcal{X}, \mathcal{C}, J_{\text{train}}, \mathcal{C})$

- SemiEmb [8], Planetoid [10]

Approach: Graph Structure + Other Information

Using $\mathcal{G} = (A, \mathcal{X}, \mathcal{C}, J_{\text{train}}, \mathcal{C})$

- SemiEmb [8], Planetoid [10]
- GCN [3], **SGC** (this paper)

Using $\mathcal{G} = (A, \mathcal{X}, \mathcal{C}, J_{\text{train}}, \mathcal{C})$

- SemiEmb [8], Planetoid [10]
- GCN [3], **SGC** (this paper)
- GraphSAGE [2], DGI [7]

Using $\mathcal{G} = (A, \mathcal{X}, \mathcal{C}, J_{\text{train}}, \mathcal{C})$

- SemiEmb [8], Planetoid [10]
- GCN [3], **SGC** (this paper)
- GraphSAGE [2], DGI [7]
- (and a few more)

Approach: Graph Structure + Other Information

Using $\mathcal{G} = (A, \mathcal{X}, \mathcal{C}, J_{\text{train}}, \mathcal{C})$

- SemiEmb [8], Planetoid [10]
- GCN [3], **SGC** (this paper)
- GraphSAGE [2], DGI [7]
- (and a few more)

Common theme: Graph + Features

Using $\mathcal{G} = (A, \mathcal{X}, \mathcal{C}, J_{\text{train}}, \mathcal{C})$

- SemiEmb [8], Planetoid [10]
- GCN [3], **SGC** (this paper)
- GraphSAGE [2], DGI [7]
- (and a few more)

Common theme: Graph + Features $\rightarrow$ Average Features

Using $\mathcal{G} = (A, \mathcal{X}, \mathcal{C}, J_{\text{train}}, \mathcal{C})$

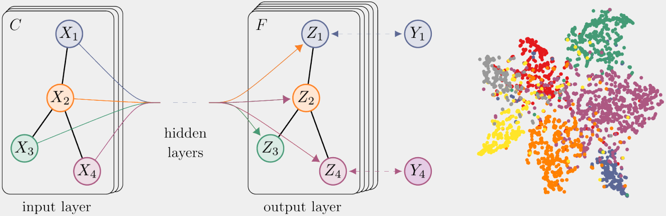
- SemiEmb [8], Planetoid [10]
- GCN [3], **SGC** (this paper)
- GraphSAGE [2], DGI [7]
- (and a few more)

Common theme: Graph + Features $\rightarrow$ Average Features $\rightarrow$ Loss Backpropagation

Using $\mathcal{G} = (A, \mathcal{X}, \mathcal{C}, J_{\text{train}}, \mathcal{C})$

- SemiEmb [8], Planetoid [10]
- GCN [3], **SGC** (this paper)
- GraphSAGE [2], DGI [7]
- (and a few more)

Common theme: Graph + Features $\rightarrow$ Average Features $\rightarrow$ Loss Backpropagation $\rightarrow$ Node Representations.



(a) Graph Convolutional Network

(b) Hidden layer activations

Figure: A Doodle of GCN

Essentially, there are three steps to perform GCN:

Steps for GCN

Essentially, there are three steps to perform GCN:

1. Feature propagation: $\bar{H}^{(k)} = SH^{(k-1)}$, where $S = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$ and $H^{(k)}$ is the activation at the $k-$th layer.

Steps for GCN

Essentially, there are three steps to perform GCN:

1. Feature propagation: $\bar{H}^{(k)} = SH^{(k-1)}$, where $S = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$ and $H^{(k)}$ is the activation at the $k-$th layer.
2. Activation: $H^{(k)} = \mathrm{ReLU}(\bar{H}^{(k)}\Theta^{(k)})$

Essentially, there are three steps to perform GCN:

1. Feature propagation: $\bar{H}^{(k)} = SH^{(k-1)}$, where $S = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$ and $H^{(k)}$ is the activation at the $k-$th layer.
2. Activation: $H^{(k)} = \text{ReLU}(\bar{H}^{(k)}\Theta^{(k)})$
3. Classification: $\hat{Y}_{\text{gcn}} = \text{softmax}(SH^{(K-1)}\Theta^{(K)})$

Simplifying GCN

Simplifying GCN" addresses the excess complexity in recent GCN models.

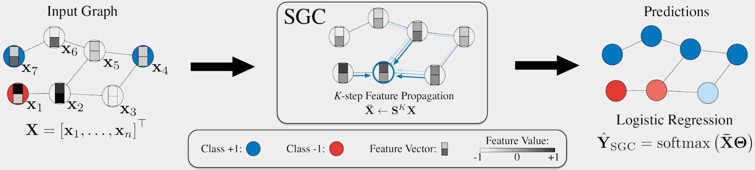


Figure: SGC model

Simplifying GCN" addresses the excess complexity in recent GCN models.

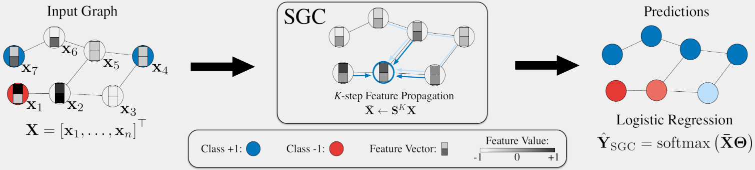


Figure: SGC model

Main experiment

Input: $\mathcal{G} = (A, \mathcal{X}, \mathcal{C}, J_{\text{train}}, \mathcal{C})$
Output: $\mathcal{C}_i \ \forall i \in V(A) - J_{\text{train}}$

Observation

Hypothesis: Nonlinearity is not important

The main factor in the power of a GNN is the feature averaging step (GCN step 1), *not the nonlinearity between layers* (GCN step 2).

Removing all activation function between K layers:

$$\hat{Y} = \text{softmax}(SS...S\mathcal{X}\Theta^{(1)}\Theta^{(2)}...\Theta^{(K)})$$

Since everything is matrix multiplications:

Simple Graph Convolution

$$\hat{Y}_{\text{sgc}} = \text{softmax}(S^K\mathcal{X}\Theta)$$

Experiments & Results

Datasets

Dataset	# Nodes	# Edges	Train/Dev/Test Nodes
Cora	$2,708$	$5,429$	$140/500/1,000$
Citeseer	$3,327$	$4,732$	$120/500/1,000$
Pubmed	$19,717$	$44,338$	$60/500/1,000$
Reddit	233K	11.6M	152K/24K/55K

Figure: Datasets

Competitors

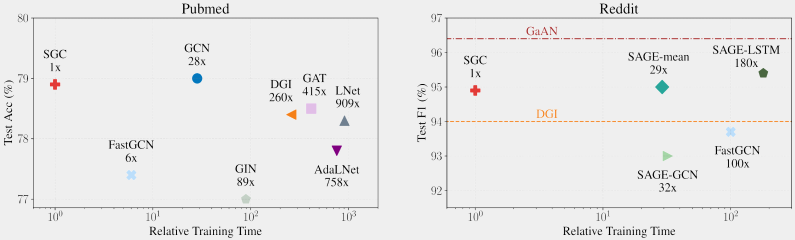


Figure: Competitors and their performance

	Cora	Citeseer	Pubmed
Numbers from literature:			
GCN	81.5	70.3	79.0
GAT	83.0 ± 0.7	72.5 ± 0.7	79.0 ± 0.3
GLN	81.2 ± 0.1	70.9 ± 0.1	78.9 ± 0.1
AGNN	83.1 ± 0.1	71.7 ± 0.1	79.9 ± 0.1
LNet	79.5 ± 1.8	66.2 ± 1.9	78.3 ± 0.3
AdaLNet	80.4 ± 1.1	68.7 ± 1.0	78.1 ± 0.4
DeepWalk	70.7 ± 0.6	51.4 ± 0.5	76.8 ± 0.6
DGI	82.3 ± 0.6	71.8 ± 0.7	76.8 ± 0.6
Our experiments:			
GCN	81.4 ± 0.4	70.9 ± 0.5	79.0 ± 0.4
GAT	83.3 ± 0.7	72.6 ± 0.6	78.5 ± 0.3
FastGCN	79.8 ± 0.3	68.8 ± 0.6	77.4 ± 0.3
GIN	77.6 ± 1.1	66.1 ± 0.9	77.0 ± 1.2
LNet	$80.2 \pm 3.0^{\dagger}$	67.3 ± 0.5	$78.3 \pm 0.6^{\dagger}$
AdaLNet	$81.9 \pm 1.9^{\dagger}$	$70.6 \pm 0.8^{\dagger}$	$77.8 \pm 0.7^{\dagger}$
DGI	82.5 ± 0.7	71.6 ± 0.7	78.4 ± 0.7
SGC	81.0 ± 0.0	71.9 ± 0.1	78.9 ± 0.0

(a) Classic Datasets for GNNs

Setting	Model	Test F1
Supervised	GaAN	96.4
	SAGE-mean	95.0
	SAGE-LSTM	95.4
	SAGE-GCN	93.0
	FastGCN	93.7
	GCN	**OOM**
Unsupervised	SAGE-mean	89.7
	SAGE-LSTM	90.7
	SAGE-GCN	90.8
	DGI	94.0
No Learning	Random-Init DGI	93.3
	SGC	94.9

(b) Reddit Dataset

Figure: Results for SGC

Conclusion

Conclusion

The authors have demonstrated:

Conclusion

The authors have demonstrated:

1. The activations might not be needed between graph layers.

Conclusion

The authors have demonstrated:

1. The activations might not be needed between graph layers.
2. Simple formulation can give fast and high accuracy results.

Thanks for listening!

Aditya Grover and Jure Leskovec.
node2vec: Scalable feature learning for networks.
In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864. ACM, 2016.

Will Hamilton, Zhitao Ying, and Jure Leskovec.
Inductive representation learning on large graphs.
In *Advances in Neural Information Processing Systems*, pages 1024–1034, 2017.

Thomas N Kipf and Max Welling.
Semi-supervised classification with graph convolutional networks.
arXiv preprint arXiv:1609.02907, 2016.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena.
Deepwalk: Online learning of social representations.
In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014.

L. Tang and H. Liu.
Efficient graphlet kernels for large graph comparison.
pages 817–826, 2009.

L. Tang and H. Liu.
Reconstructing markov processes from independent and anonymous experiments.
pages 447–478, 2011.

Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm.
Deep graph infomax.
arXiv preprint arXiv:1809.10341, 2018.

Jason Weston, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert.
Deep learning via semi-supervised embedding.
In *Neural Networks: Tricks of the Trade*, pages 639–655. Springer, 2012.

Felix Wu, Tianyi Zhang, Amauri Holanda de Souza Jr, Christopher Fifty, Tao Yu, and Kilian Q Weinberger.
Simplifying graph convolutional networks.
arXiv preprint arXiv:1902.07153, 2019.

Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov.
Revisiting semi-supervised learning with graph embeddings.
arXiv preprint arXiv:1603.08861, 2016.

Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty.
Semi-supervised learning using gaussian fields and harmonic functions.
pages 912–919, 2003.