Simplifying Graph Convolutional Networks

Felix Wu*  Tianyi Zhang*  Amauri Holanda de Souza Júnior*  Christopher Fifty  Tao Yu  Kilian Q. Weinberger

*: Equal contribution
Graph Convolutional Networks Applications
(Kipf and Welling, ICLR 2017)

- Social Networks & Citation Networks
- Applied Chemistry
- Natural Language Processing
- Computer Vision
- …
Node Classification

Input Graph
Node Classification

Input Graph

Features: \( X = [x_1, \ldots, x_n]^T \)
Node Classification

Features: \( X = [x_1, \ldots, x_n]^T \)

(Partially labelled)
Node Classification

Features: $X = [x_1, \ldots, x_n]^T$

(Partially labelled)
Graph Convolutional Network (GCN)

\[ H^{(0)} = X = [x_1, \ldots, x_n]^T \]
Graph Convolutional Network (GCN)

\[ H^{(0)} = X = [x_1, \ldots, x_n]^T \]

Feature Propagation

\[ \overline{H}^{(k)} \leftarrow S H^{(k-1)} \]
Graph Convolutional Network (GCN)

\[ H^{(0)} = X = [x_1, \ldots, x_n]^T \]

Input Graph

Feature Propagation
\[ \mathbf{H}^{(k)} \leftarrow \mathbf{SH}^{(k-1)} \]

Linear Transformation
\[ \bar{\mathbf{H}}^{(k)} \leftarrow \bar{\mathbf{H}}^{(k)} \Theta^{(k)} \]

Class +1: \[
\begin{array}{c}
& \black\text{Feature Vector:} & \black\text{Feature Value:} \\
& -1 & 0 & +1 \\
& \end{array}
\]

Class -1:
Graph Convolutional Network (GCN)

\[ \mathbf{H}^{(0)} = \mathbf{X} = [x_1, \ldots, x_n]^T \]

Input Graph

Feature Propagation

\[ \mathbf{H}^{(k)} \leftarrow \mathbf{S} \mathbf{H}^{(k-1)} \]

Linear Transformation

\[ \mathbf{H}^{(k)} \leftarrow \mathbf{H}^{(k)} \Theta^{(k)} \]

Nonlinearity

\[ \mathbf{H}^{(k)} \leftarrow \text{ReLU}(\mathbf{H}^{(k)}) \]

Class +1: blue
Class -1: red
Feature Vector: grey

Feature Value:
Graph Convolutional Network (GCN)

\[ \mathbf{H}^{(0)} = \mathbf{X} = [x_1, \ldots, x_n]^T \]

Input Graph

Feature Propagation
\[ \mathbf{H}^{(k)} \leftarrow \mathbf{S} \mathbf{H}^{(k-1)} \]

Linear Transformation
\[ \mathbf{H}^{(k)} \leftarrow \mathbf{H}^{(k)} \Theta^{(k)} \]

Nonlinearity
\[ \mathbf{H}^{(k)} \leftarrow \text{ReLU}(\mathbf{H}^{(k)}) \]

Class +1: 🔛  Class -1: 🔜  Feature Vector: □

Feature Value: -1 0 +1
Graph Convolutional Network (GCN)

\[ H^{(0)} = X = [x_1, \ldots, x_n]^T \]

\[ \hat{Y}_{GCN} = \text{softmax}(SH^{(K-1)}\Theta^{(K)}) \]

**Input Graph**

- \( x_1, x_2, x_3 \)
- \( x_4, x_5, x_6, x_7 \)

**GCN**

1. **Feature Propagation**
   \[ \hat{H}^{(k)} \leftarrow SH^{(k-1)} \]

2. **Linear Transformation**
   \[ \hat{H}^{(k)} \leftarrow \hat{H}^{(k)}\Theta^{(k)} \]

3. **Nonlinearity**
   \[ H^{(k)} \leftarrow \text{ReLU}(\hat{H}^{(k)}) \]

4. **Predictions**
   \[ \hat{Y}_{GCN} = \text{softmax}(SH^{(K-1)}\Theta^{(K)}) \]
Graph Convolutional Network (GCN)

Is the nonlinearity necessary?

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

\[ \mathbf{H}^{(k)} \leftarrow \mathbf{SH}^{(k-1)} \]

\[ \mathbf{H}^{(k)} \leftarrow \text{ReLU}(\mathbf{H}^{(k)}) \]

\[ \hat{\mathbf{Y}}_{GCN} = \text{softmax}(\mathbf{SH}^{(K-1)} \Theta^{(K)}) \]

Class +1: ⬝ Class -1: ⬤ Feature Vector: □
Graph Convolutional Network (GCN)

Is the nonlinearity necessary?

Feature Propagation
\[ \mathbf{H}^{(k)} \leftarrow \mathbf{S} \mathbf{H}^{(k-1)} \]

Linear Transformation
\[ \tilde{\mathbf{H}}^{(k)} \leftarrow \tilde{\mathbf{H}}^{(k)} \mathbf{\Theta}^{(k)} \]

Nonlinearity
\[ \mathbf{H}^{(k)} \leftarrow \text{ReLU} (\tilde{\mathbf{H}}^{(k)}) \]

Predictions
\[ \hat{\mathbf{Y}}_{\text{GCN}} = \text{softmax} (\mathbf{S} \mathbf{H}^{(K-1)} \mathbf{\Theta}^{(K)}) \]

\[ \hat{\mathbf{Y}} = \text{softmax} (\mathbf{S} \ldots \mathbf{S} \mathbf{X} \mathbf{\Theta}^{(1)} \mathbf{\Theta}^{(2)} \ldots \mathbf{\Theta}^{(K)}) \]

Class +1: 🔵 Class -1: 🔴 Feature Vector: □
Graph Convolutional Network (GCN)

Is the nonlinearity necessary?

\[ \hat{Y} = \text{softmax} \left( S \ldots SSX\Theta^{(1)}\Theta^{(2)}\ldots\Theta^{(K)} \right) \]

\[ \hat{Y}_{\text{GCN}} = \text{softmax}(SH^{(K-1)}\Theta^{(K)}) \]
Simple Graph Convolution (SGC)

\[ \hat{Y}_{SGC} = \text{softmax}(\tilde{X}\Theta) \]

Input Graph

\[ X = [x_1, \ldots, x_n]^T \]

SGC

K-step Feature Propagation

\[ \tilde{X} \leftarrow S^K X \]

Predictions

Logistic Regression

Class +1:  
Class -1:  
Feature Vector:  
Feature Value:
Simple Graph Convolution (SGC)

1. Interpretability
2. Mini-batch training on large graphs
3. Second-order optimization methods

$$\hat{Y}_{SGC} = \text{softmax} (\bar{X}\Theta)$$

\[ X = [x_1, \ldots, x_n]^T \]
Performance vs. Training Time on Reddit Dataset
(Hamilton et al., NeurIPS 2017)

Reddit dataset contains 233K nodes and 11.6M edges

Within 2.7 seconds.
Comparable or Better Results

• SGC performs on par with or better than GCN across 5 tasks including 14 datasets.
  ‣ Graph classification on citation and social networks: Cora, Citeseer, Pubmed, and Reddit datasets
  ‣ Text classification: 20NG, R8, R52, Ohsumed, MR
  ‣ Semi-supervised user geolocation: GEOTEXT, TWITTER-US, TWITTER-WORLD
  ‣ Relation extraction: TACRED
  ‣ Zero-shot image classification: ImageNet
Comparable or Better Results

• SGC performs on par with or better than GCN across 5 tasks including 14 datasets.
  ▸ Graph classification on citation and social networks: Cora, Citeseer, Pubmed, and Reddit datasets
  ▸ Text classification: 20NG, R8, R52, Ohsumed, MR
  ▸ Semi-supervised user geolocation: GEOTEXT, TWITTER-US, TWITTER-WORLD
  ▸ Relation extraction: TACRED
  ▸ Zero-shot image classification: ImageNet

But not on graph classification task! (Nonlinearity is still beneficial on this task)
Thank You!

• A linear model (SGC) is sufficient on many graph tasks.

• Our official code is available at: https://github.com/Tiiiger/SGC

• We thank *Deep Graph Library*, *PyTorch Geometric*, *Spektral*, and *StellarGraph* for including SGC in their library.

• Please feel free to come by our poster section if you have questions:

  **Tonight 06:30~09:00 PM @ Pacific Ballroom #267**