Notes on Semi-Supervised Classification with Graph Convolutional Networks

Rom Parnichkun

August 18, 2022

1 Introduction

- Consider the problem of node classification (where labels are only available for a small subset of nodes)
- Instead of using graph-based regularization, this work only trains of nodes with a supervised target $\mathcal{L}_0$, and shares the same set of parameters across the graph for inference.
- Main contributions:
  1. Introduce a simple and well-behaved layer-wise propagation rule for neural network models which operate directly on graphs
  2. Demonstrate how this form of graph-based model can be used for fast and scalable semi-supervised node classification

2 Graph Convolutional Network Model

Layer-wise propagation rule for graph convolutional network model $f(X, A)$ (X is the input and A is the adjacency matrix):

$$H^{l+1} = \sigma\big(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^l W^l\big). \tag{1}$$

Here, $\tilde{A} = A + I_N$ is the adjacency matrix of the undirected graph $\mathcal{G}$ with added self-connections, $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$ and W^l is a layer specific trainable weight matrix. $\sigma(\cdot)$ denotes an activation function. $H^l \in \mathbb{R}^{N\times D}$ are hidden latent representations at layer l and $H^0 = X$.

2.1 Breaking it down

- $H^l W^l$ converts the hidden representation of size $N \times D^l$ to $N \times D^{l+1}$
- $\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$ normalizes the elements in the adjacency matrix $\tilde{A}_{ij}$ by $\frac{1}{\sqrt{\tilde{D}_{ii}}\sqrt{\tilde{D}_{jj}}}$.
- $\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^l W^l$ represents the resultant hidden features after message passing of each node's latent representation to their one-hop neighbors weighted by each other's degree (D_{ii} and D_{jj}).
- Finally, $\sigma(\cdot)$ introduces a non-linearity.

3 Semi-supervised Node Classification

Setting $\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$ a forward model of a 2-layer GCN takes the following form:

$$Z = f(X, A) = \text{softmax}\big(\hat{A}\ \text{ReLU}(\hat{A}XW^0)W^1\big) \tag{2}$$

For multi-class classification, the following loss function can be computed.

$$\mathcal{L} = -\sum_{l\in\mathcal{Y}_L}\sum_{f=1}^{F} Y_{lf}\ln Z_{lf}, \tag{3}$$

with F being the feature size of the output and $\mathcal{Y}_L$ is the set of node indices that have labels.