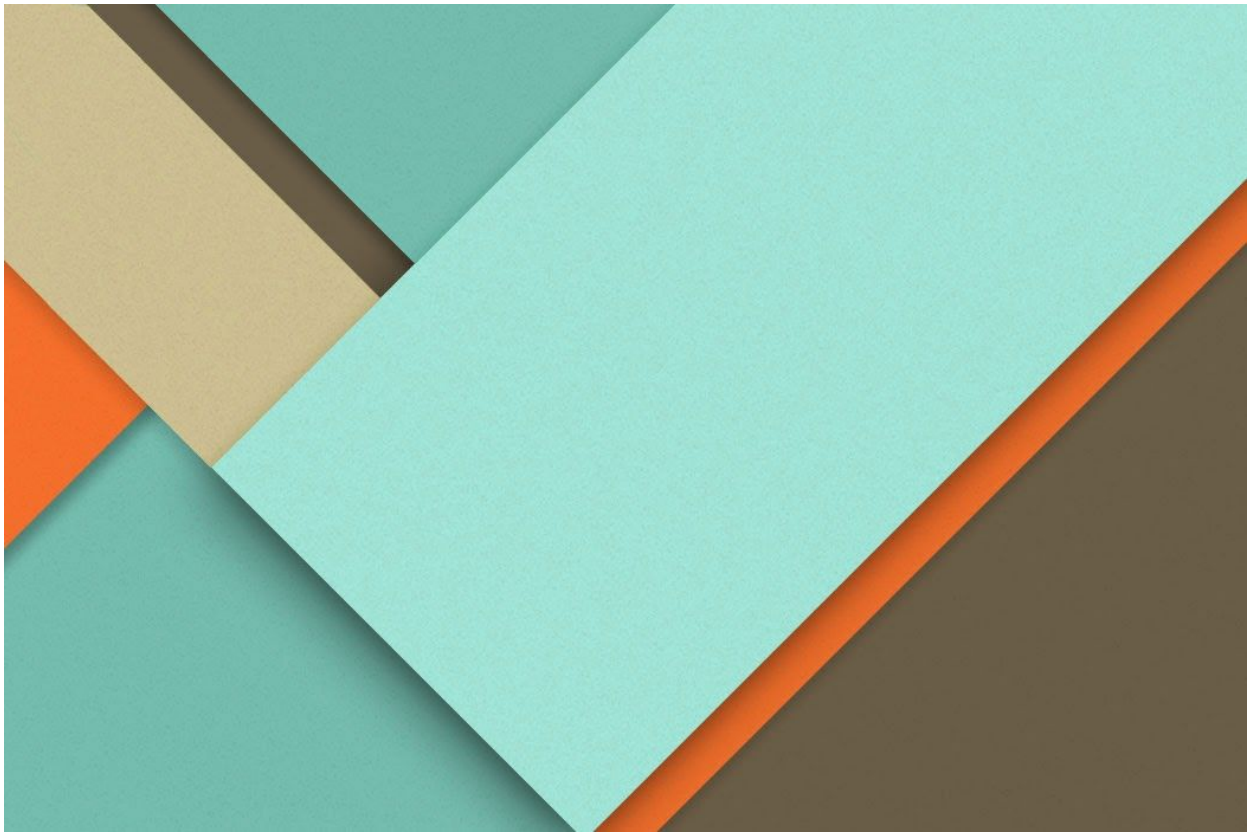


COSI126a_HW5

11.20.2020

Menghong Han, MSBA

International Business School

Brandeis University

Implement Graph Convolutional Networks to solve the node classification problem, and test it on Cora. Report the accuracy of your prediction, and compare it with the result of Node2vec in Assignment 4.

GCN

Graph Convolutional Networks (GCNs) is convolutional and can filter parameters are typically shared over all locations in the graph (or a subset thereof as in Duvenaud et al., NIPS 2015).

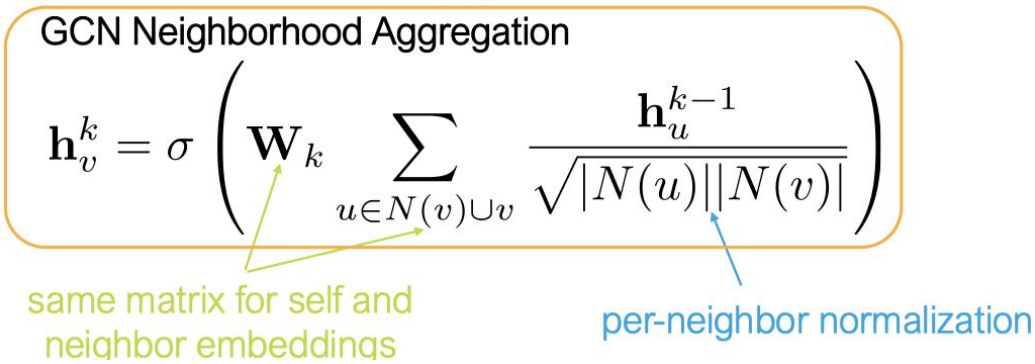
The goal is then to learn a function of signals/features on a G=(V,E) which takes as input:

1. A feature description xi for every node i; summarized in a N×D feature matrix X (N: number of nodes, D: number of input features)
2. A representative description of the graph structure in matrix form; typically in the form of an adjacency matrix A (or some function thereof) and produces a node-level output Z (an N×F feature matrix, where F is the number of output features per node).

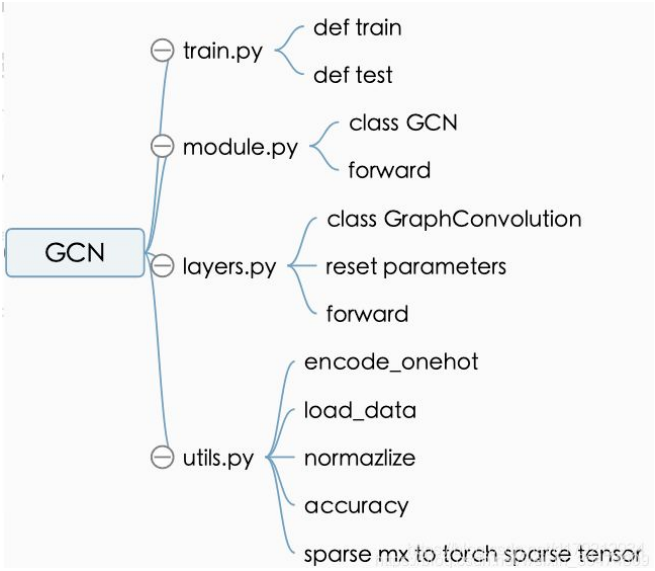
Graph-level outputs can be modeled by introducing some form of pooling operation (see, e.g. Duvenaud et al., NIPS 2015).

Every neural network layer can then be written as a nonlinear function H(l+1)=f(H(l),A), with H(0)=X and H(L)=Z (or z for graph-level outputs), L being the number of layers. The specific models then differ only in how $f(\cdot,\cdot)$ is chosen and parameterized.

The neighborhood aggregation function is as following:



The following is the GCN implementation code structure:



Result

Total time elapsed: 2.2331s

Test set results: loss= 0.7302 accuracy= 0.8260

Comparison

The accuracy for GCN on Cora is 0.8260, while the best accuracy of Node2vec is 0.77 where (p = 1, q = 0.5).

p	q	num_walk	walk_length	accuracy	time
1	1	10	40	0.747	10.76s
1	0.5	10	40	0.77	9.98s
2	0.5	10	40	0.749	10.2s
0.5	1	10	40	0.747	9.89s
0.5	2	10	40	0.743	9.81

Node2vec Classification Result

From the above result, we can see **GCN** performs better.