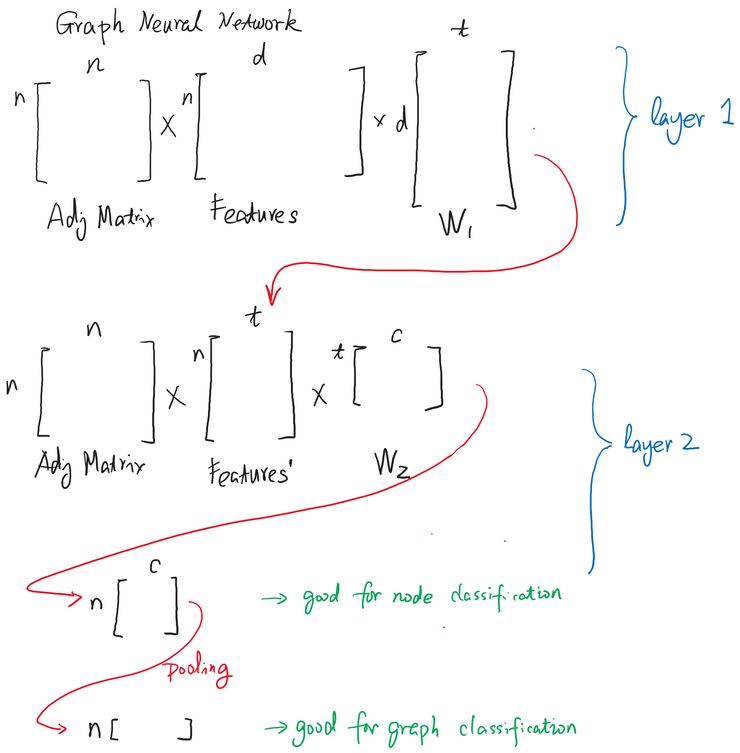


GCN does not need to care about the size of the graph. In fact, what to be learnt are W_1 and W_2 in this example.

- W_1 is decided by the number of features and the size of the hidden layer.
- W_2 is decided by the size of the hidden layer and the num of classes.

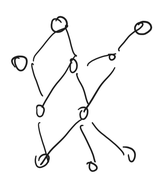
  d: # of features

  t: size of the hidden layer

  C: # of classes.

This GCN does not care about the size of the graph because the adj matrix and features of nodes are used as inputs. However, d, t, and C should be fixed for a training task.

for example:



$G_1 \begin{cases} A_1 \\ X_1 \end{cases} \Rightarrow GCN(A_1, X_1) \Rightarrow$

$$\underset{n_1\times t}{O} = \underset{n_1\times n_1}{A_1} \times \underset{n_1\times d}{X_1} \times \underset{d\times t}{W_1}$$

$$\underset{n_1\times c}{O} = \underset{n_1\times n_1}{A_1} \times \underset{n_1\times t}{O} \times \underset{t\times c}{W_2}$$

$G_2 \begin{cases} A_2 \\ X_2 \end{cases} \Rightarrow GCN(A_2, X_2) \Rightarrow$

$$\underset{n_2\times t}{O} = \underset{n_2\times n_2}{A_2} \times \underset{n_2\times d}{X_2} \times \underset{d\times t}{W_1}$$

$$\underset{n_2\times c}{O} = \underset{n_2\times n_2}{A_2} \times \underset{n_2\times t}{O} \times \underset{t\times c}{W_2}$$

So, what really matters for GCN are d, t, and C.