Graph Attention Network

The layer corresponding to this network is known as graph attention layer. The input to our layer is a set of node features **h**, **N** is the number of nodes and **F** is the number of features in each node. The layer produces a new set of node features known as **h’.** For this purpose we have the following equations:

Note:- We are starting with only single graph attention layer.

1. First we need to find the attention coefficients which is given by equation:-



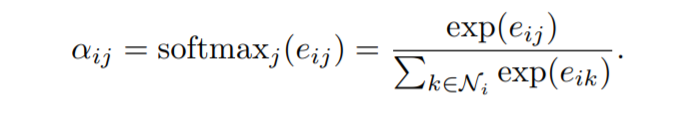
Here,

a= the attention mechanism which is a single-layer feedforward neural network.

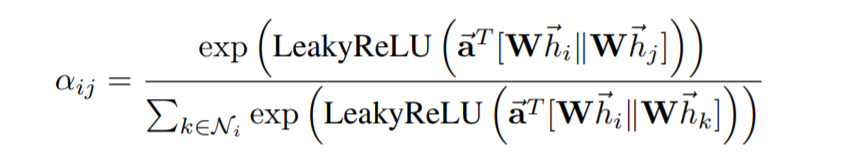
W= learnable weight matrix

The attention coefficient indicates the importance of node j’s features to node i. In its most general formulation, the model allows every node to attend on every other node, dropping all structural information. We inject the graph structure into the mechanism by performing masked attention—we only compute eij for nodes j ∈ Ni , where Ni is some neighborhood of node i in the graph.

1. Now to make the coefficients easily comparable across different nodes, we normalize them across all choices of j using softmax function:



If we expand the above equation and taking LeakyReLU nonlinearity, it can be expressed as:



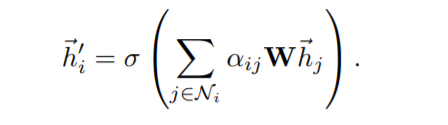
Here,

Vector(a)= learnable weight vector

.T represents transposition.

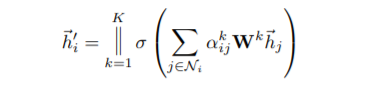
|| is the concatenation operation.

1. The normalized attention coefficients are used to compute a linear combination of the features corresponding to them, to serve as the final output features for every node applying a nonlinearity, σ:

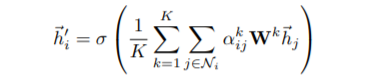


To stabilize the learning process of self-attention, employment of multi-head attention is beneficial. In that case, each attention head has its own parameters and their outputs can be merged in two ways:

1)Concatenation:



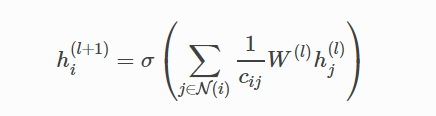
2)Average:



where K is the number of heads. We can use concatenation for intermediary layers and average for the final layer.

**Comparison:**

GAT is similar to GCN and GraphSage. The difference lies in equation 3. For both the normalized sum of the node features of neighbors is given by:



For GCN value of cij is given by:-



For GraphSage value of cij is given by:-

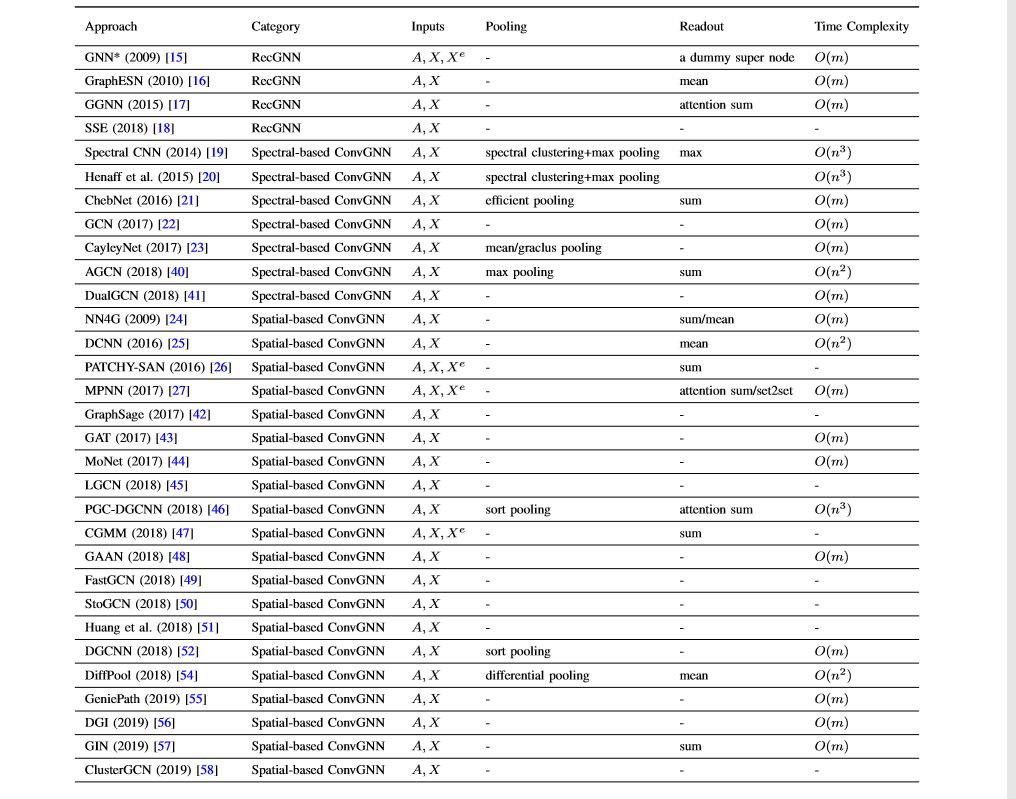


Here,

cij= normalization constant

N(i)=one-hop neighbor (directly connected vertex in graph)

Comparison between different approaches:



**Implementation:**

For implementation of GAT we can take help from following links:

1) <https://github.com/PetarV-/GAT>

2) <https://docs.dgl.ai/en/0.4.x/tutorials/models/1_gnn/9_gat.html>

3) <https://github.com/danielegrattarola/keras-gat/blob/master/keras_gat/graph_attention_layer.py>

Other links that can be useful:-

1. <https://github.com/shenweichen/GraphNeuralNetwork/blob/master/gnn/gat.py>
2. <https://github.com/HazyResearch/hgcn>

**Reference:**

1. <https://arxiv.org/pdf/1710.10903.pdf>