Graph Attentional Networks

Convolutional Neural Networks have been quite successful at tasks like image classification, image segmentation, etc where underlying data is in a grid structure. However, in many tasks, the data is in a graph structure example, 3D meshes, social networks, telecommunication networks, biological networks, or brain connectomes. To solve this, Graph Neural Networks (GNNs) were introduced by Gori et al. (2005) and Scarselli et al. (2009) as a generalization of recursive neural networks that can directly deal with a more general class of graphs.

GNNs consist of an iterative process, which propagates the node states until equilibrium; followed by a neural network, which produces an output for each node based on its state. This idea was adopted and improved by Li et al. (2016), who proposed to use of gated recurrent units (Cho et al., 2014) in the propagation step.

There have been two approaches for making models on graphs. Spectral and Non-Spectral. Spectral approaches involve using the Laplacian Matrix of the graph for convolutional operations and Non-Spectral data defines convolutions directly on the spatial structure of the graph.

Inspired by previous works' where it had been established that attention mechanisms were well suited for variable-sized inputs, and were able to selectively focus on the most relevant parts of data, this paper introduced a Non-spectral Attention-based architecture to perform node classification of graph-structured data. The idea is to compute the hidden representations of each node in the

graph, by attending over its neighbors, following a self-attention strategy. The attention architecture has several interesting properties:

(1) the operation is efficient since it is parallelizable across node neighbor pairs;

(2) it can be applied to graph nodes having different degrees by specifying arbitrary weights to the neighbors; and

(3) the model is directly applicable to inductive learning problems, including tasks where the model has to generalize to completely unseen graphs.

The proposed approach was validated on four challenging benchmarks - Cora, Citeseer, and Pubmed citation networks as well as an inductive protein-protein interaction dataset achieving state-of-the-art results.

GAT Architecture

- **Parametrised Linear Transformation**

A parametrized linear transformation is applied to every node's feature vector by using a trainable weight matrix which is multiplied with the node features to produce node-specific embeddings. This is done to increase the dimensionality of input to be able to learn complex relations.

- **Computing Self-Attention**

Self-attention mechanism calculates attention coefficients for each pair of nodes in the graph to allow the nodes to weigh the importance of a node's features relative to its neighbors.

- **Normalization using custom softmax function**

This is just so that the sum of attention coefficients of a node's neighbors sum up to one so we can interpret them as probabilities.

- **Attention Mechanism with a Leaky ReLU Function**

This introduces non-linearity into the model and allows the network to learn sparse attention patterns. The leaky ReLU function ensures that even small inputs result in non-zero output. This prevents nodes from ignoring their less relevant neighbors.

- **Aggregate Multi-Head Attention**

aggregated features from each head are concatenated or averaged so that a node can attend to multiple aspects of its neighborhood simultaneously.