DL lab 6 – Graph Neural Networks

1. Upload the NetworkX jupyter notebook file (i.e.,NetworkX\_tutorial.ipynb) to google colab root directory.
   * Run the above code and understand it.
   * Complete the code sections to get the degree matrix and Laplacian matrix of the created random graph.
   * Calculate the graph density of the random graph in the code. Use the below equation (D = graph density, |V| = number of nodes and |E| = number of edges).
   * Increase the N value from 20 (original value) to 200 with multiple N values in between and observe the change of graph density and degree distribution (i.e., histogram plot). Explain what you observe and write the answer in a word file.



**Graph Density**: Graph density decreases as N increases due to a fixed number of edges.

**Degree Distribution**: Degree distribution becomes more heterogeneous, with a broader range of node degrees.

**Histogram Plot**: The histogram plot of the degree distribution widens, showing a long-tailed distribution.

**Sparsity**: The graph becomes sparser, with fewer connections relative to possible connections.

1. In the KarateClub dataset based GCN code, we use semi-supervised training approach along with the transductive leaning method.
   * Explain the differences between supervised learning, self-supervised learning and semi-supervised learning methods

**Supervised Learning:** Model is trained on labeled data to predict the labels of new data.

**Self-Supervised Learning:** The model generates its own labels from the data, often used for representation learning.

**Semi-Supervised Learning:** Uses both labeled and unlabeled data to improve learning accuracy.

* + Explain the differences between transductive learning and inductive learning.

**Transductive Learning:** Trains on specific data and only makes predictions for that data.

**Inductive Learning:** Trains on data and generalizes to new, unseen data.

1. Upload the KarateClub dataset based GCN jupyter notebook file (i.e., KarateClub\_GCN\_introduction.ipynb ) to google colab root directory.
   * In this code, we use Zachary’s karate club network dataset.
   * Run the above code and understand it.
   * Increase the number of epochs from 50 to 500 and observe the change in validation accuracy and write what you observe in the word file.
   * Experiment without self-loops added to GCNConv() layers in the GCN() model and detail the model accuracy increase/decrease in the word file.
   * Increase the number of GCNConv() layers in the GCN() model upto 8 layers from original 3 layers. Detail the accuracy increase/decrease in the word file.
     1. In\_channels and out\_channels in GCNConv() can be considered as hyper-parameters and you can use the best performing values you find.
     2. Add skip connections between some of the GCNConv() layers and try to see if that can improve the model performance.
     3. Detail what you observe in the word file.

**By raising the GCNConv() layers to 8, the GCN() model's capacity can be significantly increased. There is a potential that it will overfit as well due to the added complexity. Carefully choosing the in\_channels and out\_channels for GCNConv() will maximize performance. Skip links between layers may be added to lessen overfitting and boost performance.**

1. Explain the differences between Message Passing GNN, graph convolution network (GCN), graph attention network (GAT) and GraphSAGE. Write the answers in the word file.

**Message Passing GNN:** Nodes aggregate information from neighbors using message passing.

**GCN (Graph Convolution Network):** Applies convolutional operations on the graph.

**GAT (Graph Attention Network):** Uses attention mechanisms to weigh the importance of neighbors.

**GraphSAGE:** Samples and aggregates neighbors to handle large graphs more efficiently.

**Submission.**

Download the final modified notebook files (all 2 jupyter notebooks). Add these notebooks and the word file to a new zip file. Upload this zip file to the courseweb submission link. The file name should be your registration number.