# DL Lab 06

## NetworkX jupyter Notebook

### The observations related to the change of graph density and degree distribution

1. **Graph Density**
   * The **density** will decrease as N increases. This is because density measures how many of the possible edges in a graph are actually present. As the number of nodes increases, the number of possible edges grows faster than the actual number of edges, which results in lower density.
2. **Degree Distribution**
   * The **degree distribution** will be broader. In smaller graphs, many nodes may have similar degrees, but as N increases, the randomness in the connections will lead to more variation in the degree distribution.
   * The histogram of degrees may show a more skewed distribution, where a few nodes have a higher degree, and most have lower degrees.
   * For an Erdos-Renyi graph, the degree distribution roughly follows a binomial distribution, which becomes more apparent as N increases.

## KarateClub dataset based GCN

### **Differences Between Supervised, Self-Supervised, and Semi-Supervised Learning Methods**

1. Supervised learning requires a fully labeled dataset, which can be costly.
2. Self-supervised learning does not require external labels but generates its own from the data.
3. Semi-supervised learning uses both labeled and unlabeled data, making it useful in cases where labeling is expensive or time-consuming.

### Differences Between Transductive and Inductive Learning

#### **Inductive Learning**

* In inductive learning, the model is trained on a given dataset (training data) and the goal is to generalize to new, unseen data (test data). The model creates a general rule or function that can be applied to new instances.
* The model learns a function f(x) that maps input x to output y during training, which can be applied to unseen test data.
* **Examples**:
  + Training a classifier on a dataset and then using it to predict labels for new, unseen samples.
  + Most machine learning methods like decision trees, neural networks, and SVMs are inductive.

#### **Transductive Learning**

* In transductive learning, the model is trained on both the training data and the test data simultaneously. However, unlike inductive learning, it doesn't aim to create a generalized function for all future instances. The goal is to make predictions specifically for the given test set.
* The model tries to infer the labels of the test data directly based on both the training and test data.
* **Examples**:
  + Label propagation in semi-supervised learning is a classic transductive learning method where the labels of test data are inferred using both labeled and unlabeled data.

### Observations for GCN Model Experiments

1. Increase the Number of Epochs from 50 to 500

Initially, training and validation accuracy improved significantly. However, after reaching a certain threshold, further increases in the number of epochs result in diminishing returns in accuracy improvement.

1. Experiment Without Self-Loops in GCNConv Layers

When **self-loops** are removed from the GCNConv layers, the model's ability to aggregate information locally from the node itself is diminished. This often results in slightly lower model validation accuracy, as the graph convolution lacks the influence of the node's own features during updates.

1. Increase the Number of GCNConv Layers (Up to 8 Layers)

Deeper GCNs (with more than 3 layers) tend to suffer from the over-smoothing problem, where node representations from different classes become indistinguishable as the number of layers increases. Increasing to 8 layers resulted in lower validation accuracy than the original 3-layer setup due to over-smoothing and reduced representation capacity.

1. Add Skip Connections Between GCN Layers

Adding **skip connections** helps mitigate the **over-smoothing** issue by allowing direct connections between non-adjacent layers. This enables the model to retain information from earlier layers, thereby improving its capacity to model complex relationships without causing the node embeddings to become too similar. Skip connections resulted in **better accuracy** compared to a deep GCN without skip connections, especially for deeper models (4+ layers). For the 8-layer model, adding skip connections improved both **training and validation accuracy**, preventing the steep decline seen in the model without skip connections.

### Differences Between **Message Passing GNN, Graph Convolution Network (GCN), Graph Attention Network (GAT),** and **GraphSAGE**

1. Message Passing GNN
   * Message Passing GNN is a general framework where nodes aggregate information from their neighbors by exchanging messages. This process is repeated over multiple layers, with each node updating its representation based on the messages received from its neighbors.
2. Graph Convolution Network (GCN)
   * GCN is a type of Message Passing GNN that performs graph convolutions, where each node updates its feature by averaging its neighbors' features. The aggregation process is weighted by the graph's adjacency matrix, normalized to ensure equal contributions.
3. Graph Attention Network (GAT)
   * GAT uses attention mechanisms to assign different weights to a node's neighbors, determining which neighbors are more important for updating the node's representation. This attention-based aggregation makes GAT more flexible than GCN when capturing node relationships.
4. GraphSAGE
   * GraphSAGE (Sample and Aggregate) learns aggregation functions that sample a fixed number of neighbors for each node and combine their features. Unlike GCN, which operates over the entire graph, GraphSAGE is designed for inductive learning and can scale better for large graphs by using neighborhood sampling.