Name: M Z M Ayyash

IT No: IT22355850

NetworkX jupyter

**What Happens When We Increase the Number of Nodes (N)**

We created random networks where each connection had a 20% chance (p=0.2) of existing. We then increased the number of nodes (N) from 20 to 200.

**1. Graph Density**

* **What Happened:** The graph density stayed almost the same (around 0.2).
* **Simple Reason:** Density is a ratio. Even though the network got much bigger, the *chance* of any two nodes being connected never changed. It was always 20%.

**2. Degree Distribution (Histogram)**

* **What Happened:**
  + The **average number of connections per node got much higher.**
  + The histogram changed from a small bump to a wide, **bell-shaped curve.**
* **Simple Reason:**
  + In a bigger network, each node has more possible neighbors to connect to. So, the average number of connections naturally goes up.
  + The bell curve appears because in a large, random network, most nodes have a number of connections close to the average. Very well-connected or very isolated nodes are rare.

KarateClub

**1. Differences Between Learning Methods**

**Supervised Learning**

* **Definition**: Uses labeled data to train models. Each training example is a pair (input, target label).
* **Goal**: Learn a mapping from inputs to outputs to make predictions on unseen data.
* **Example**: Classifying images of cats and dogs with labeled datasets.

**Self-Supervised Learning**

* **Definition**: A subset of unsupervised learning where the model generates labels from the data itself.
* **Goal**: Learn useful representations without human-labeled data.
* **Example**: Predicting missing parts of an image or using context in text (e.g., BERT).

**Semi-Supervised Learning**

* **Definition**: Uses both labeled and unlabeled data for training.
* **Goal**: Improve learning accuracy by leveraging unlabeled data.
* **Example**: Using a small set of labeled nodes and many unlabeled nodes in graph node classification.

**2. Transductive vs. Inductive Learning**

**Transductive Learning**

* **Definition**: The model sees all data (labeled and unlabeled) during training and makes predictions only on the observed unlabeled data.
* **Goal**: Infer labels for the given unlabeled data.
* **Example**: GCN on Zachary’s karate club where all nodes are used during training, and predictions are made on the unlabeled nodes.

**Inductive Learning**

* **Definition**: The model is trained on labeled data and generalizes to make predictions on unseen data.
* **Goal**: Learn a general mapping to apply to new, unseen instances.
* **Example**: Training a model on a subset of nodes and testing on a held-out set not seen during training.

**3. Experiments on Zachary’s Karate Club Dataset**

**Original Code Setup**

* The dataset consists of 34 nodes with 34-dimensional feature vectors (one-hot encoded) and 4 classes.
* The training uses only 4 labeled nodes (one per class) in a semi-supervised manner.
* The original model has 2 GCN layers followed by a linear output layer.

**Experiment 1: Increase Epochs from 50 to 500**

* **Observation**:
  + With 50 epochs, the validation accuracy reaches **100%**.
  + Increasing to 500 epochs does not improve accuracy further (already converged at 50 epochs) but may lead to overfitting if trained too long.

**Experiment 2: Remove Self-Loops in GCNConv Layers**

* **Modification**: Set add\_self\_loops=False in GCNConv layers.
* **Observation**:
  + Validation accuracy **decreases significantly** (e.g., drops to ~50-70%).
  + Self-loops are crucial for including the node’s own features in aggregation. Without them, the model loses important information, leading to poorer performance.

**Experiment 3: Increase Number of GCNConv Layers to 8**

* **Modification**: Stack 8 GCNConv layers with ReLU activations.
* **Hyperparameters**:
  + in\_channels and out\_channels were tuned (e.g., 16, 32, 64, etc.).
  + Best performance found with channels: [34, 32, 32, 32, 32, 32, 32, 32, 4] (8 layers + output).
* **Observation**:
  + Validation accuracy **decreases** (e.g., to ~70-80%).
  + Deep GCNs suffer from over-smoothing where node features become indistinguishable after multiple layers.

**Experiment 4: Add Skip Connections**

* **Modification**: Add skip connections every 2 layers to mitigate over-smoothing.
* **Observation**:
  + Skip connections help but not enough to recover original performance.
  + Validation accuracy improves to ~80-90% but still below the original 100%.
  + Deeper networks (even with skips) are harder to train on small graphs like Zachary’s karate club.