# COMP8221 ADVANCED ML

# ASSIGNMENT 2

ADITI ANAND
47699752

## Section 1: Motivation & explanation of data

I chose the Cora dataset for this assignment due to its suitability for graph-based machine learning tasks, particularly for graph and node classification. The dataset consists of 2708 scientific publications classified into seven categories, with 5429 citation links between them. Each publication is represented by a bag-of-words feature vector, indicating the presence or absence of 1433 unique words from a dictionary.

The Cora dataset is widely used as a benchmark in research for several reasons:

**Small and Easily Interpretable:** Its small size and simple structure make it easy to understand and work with, making it ideal for experimentation and prototyping.

**Representation of Real-World Networks:** The citation graph closely resembles real-world networks like social networks or citation networks, making it a valuable dataset for studying graph-based machine learning methods.

The preprocessing of data includes the following steps:

- Print the string representation of dataset object, length of dataset, number of classes and features.

```
Dataset: Cora()
Number of graphs: 1
Number of features: 1433
Number of classes: 7
Data(x=[2708, 1433], edge_index=[2, 10556], y=[2708], train_mask=[2708], val_mask=[2708], test_mask=[2708])
```

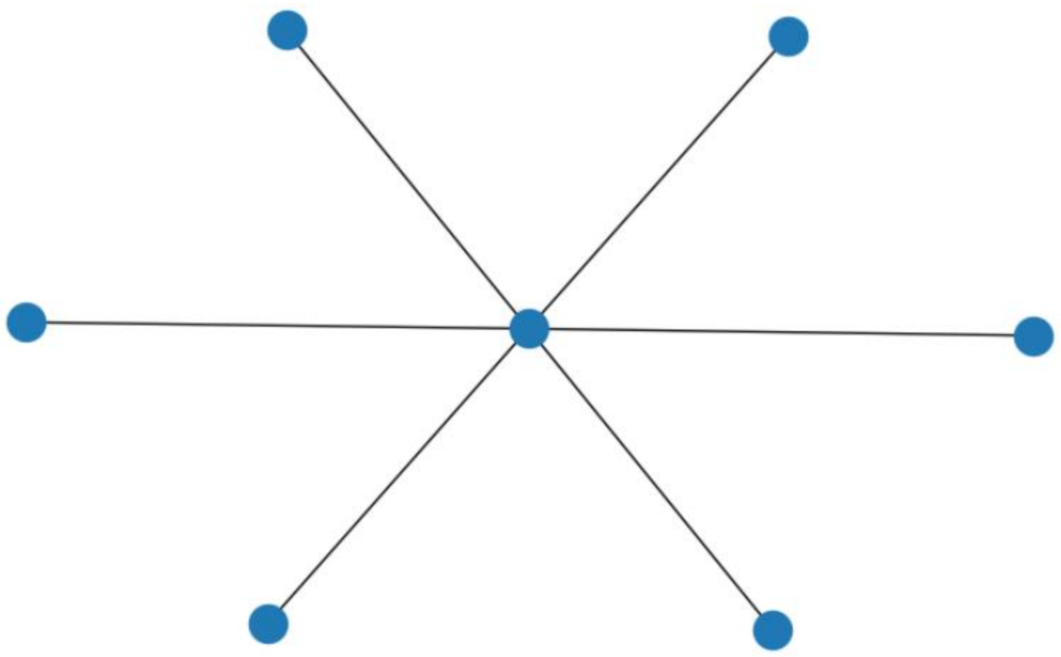
- Retrieve the node features, edge indices and target labels. Normalize the node features and print the tensor shapes.

```
Node features shape: torch.Size([2708, 1433])
Edge indices shape: torch.Size([2, 10556])
Target labels shape: torch.Size([2708])
```

- Confirm if there are nodes not connected by edges, self-loops and are not directional.

```
Has isolated nodes: False
Has self-loops: False
Is undirected: True
```

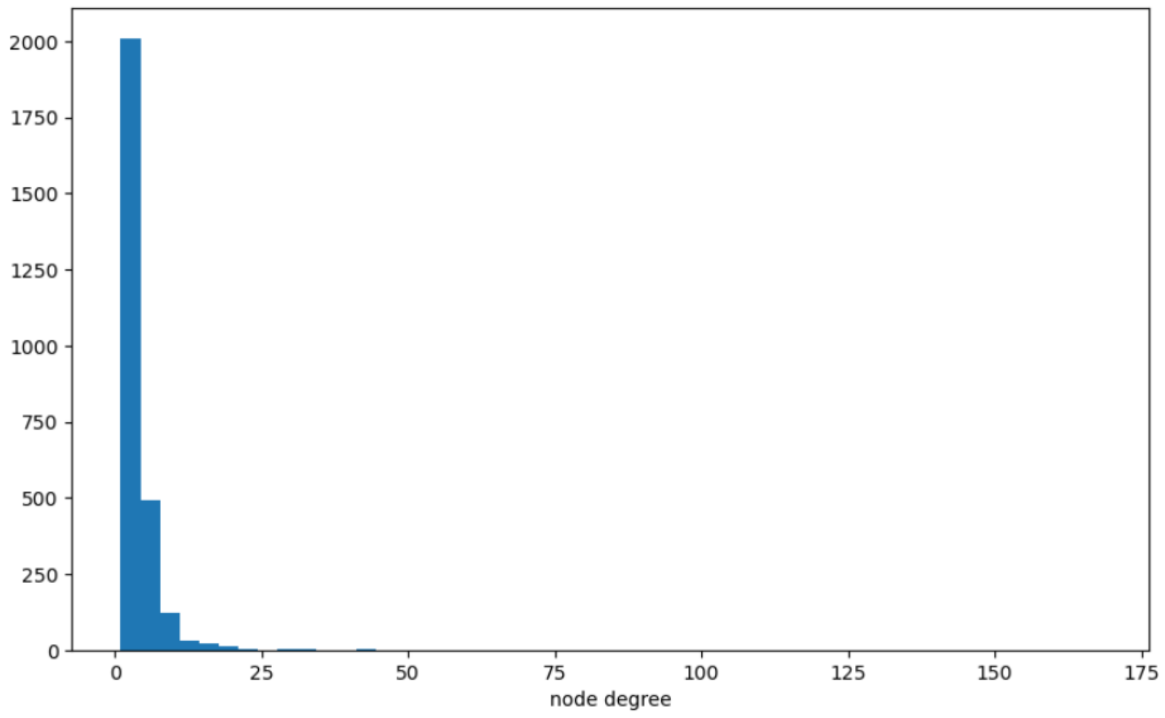
- Create a graph of edge indices with nodes as the circles and edges as the line connecting them.



- Converts the graph in “data” to NetworkX graph and get insights on distribution of node degree, descriptive statistics and histogram visualization.

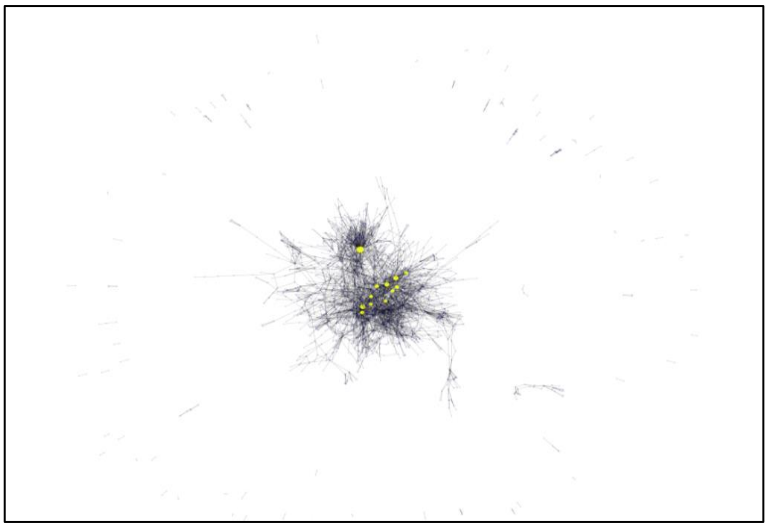
```
    count  mean   std  min  25%  50%  75%    max
0  2708.0   3.9  5.23  1.0  2.0  3.0  5.0  168.0

2708
10556
```



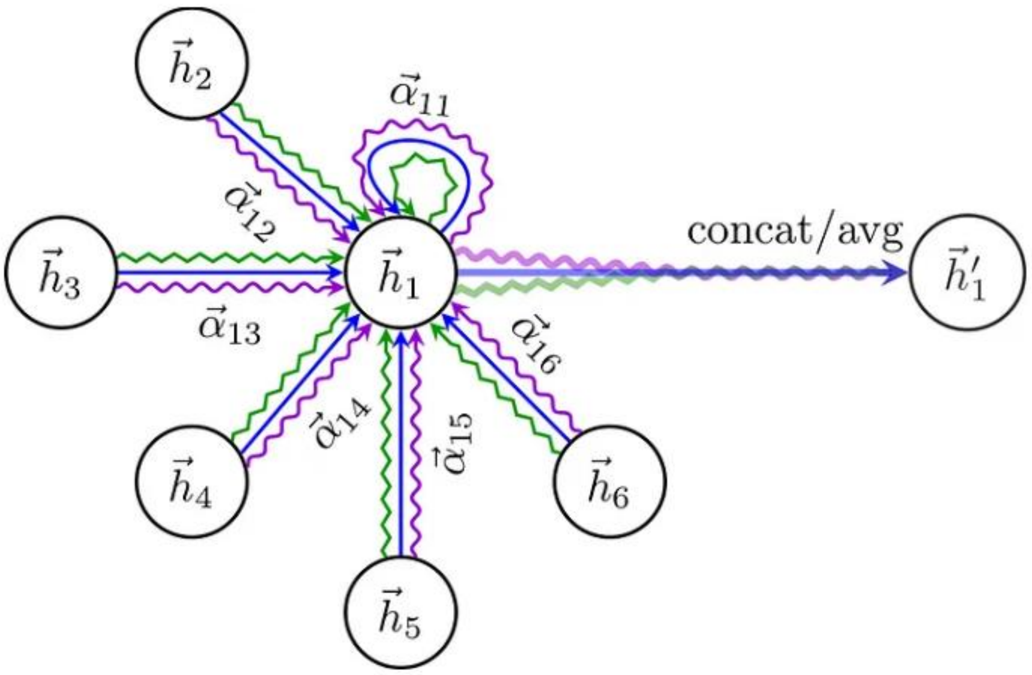
**Note:** The higher the degree, the greater number of nodes it is connected to. It is a good indication of the quality of paper by knowing the number of times the paper has been cited.

- Create the graph in which nodes size is node centrality and node colors is whether the centrality is above a certain threshold.



## Section 2: Appropriateness & explanation of model(s)

The graph ML model that I used is Graph attention mechanism (GAT). This model assigns an attention coefficient which enables the calculation of attention score for each pair of nodes which are then normalized using SoftMax function. It also allocated different weights to different neighbours which provides more clarity on how important every neighbour is. Each node retrieves and combine the features from its neighbours for the graph structure.



$$h_i^{(l+1)} = \sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(l)} \cdot W^{(l)} \cdot h_j^{(l)}$$

$$\alpha_{ij}^{(l)} = \mathrm{softmax}_j(e_{ij}^{(l)})$$

$$e_{ij}^{(l)} = a(W^{(l)} \cdot h_i^{(l)}, W^{(l)} \cdot h_j^{(l)})$$

- hi(l): Input node feature for layer l
- $W(l)$: Weight matrix for layer l
- $\alpha ij(l)$: Attention coefficient for nodes i and j for layer l
- $hi(l$+1): Output node at layer l+1

Key components of the GAT model include:

- **Model Definition (GAT class):** The GAT model consists of two GATConv layers. GATConv is a graph convolutional layer that employs attention mechanisms for message passing. The hidden_channels parameter sets the number of output channels in the first GATConv layer, while num_heads specify the number of attention heads. Dropout is used for regularization.
- **Dataset Split:** The dataset is split into training and test sets, with 80% of the nodes allocated for training and the remaining 20% for testing.
- **Optimizer:** The model parameters are optimized using the Adam optimizer, with learning rate of 0.01 and weight decay of 5e-4.

- **Training Loop:** The model is trained for 200 epochs. In each epoch, the train function updates the model, and the test function evaluates its performance on the test set. The loss and accuracy are recorded and printed for each epoch.

I chose the GAT model to compare with GCN because of its ability to learn different weights for each neighbour, potentially capturing more nuanced relationships in the graph. However, I used fewer layers (2) for GAT compared to GCN to keep the model simpler and avoid overfitting due to the higher complexity of GAT's attention mechanism.

## Section 3: Insights & results

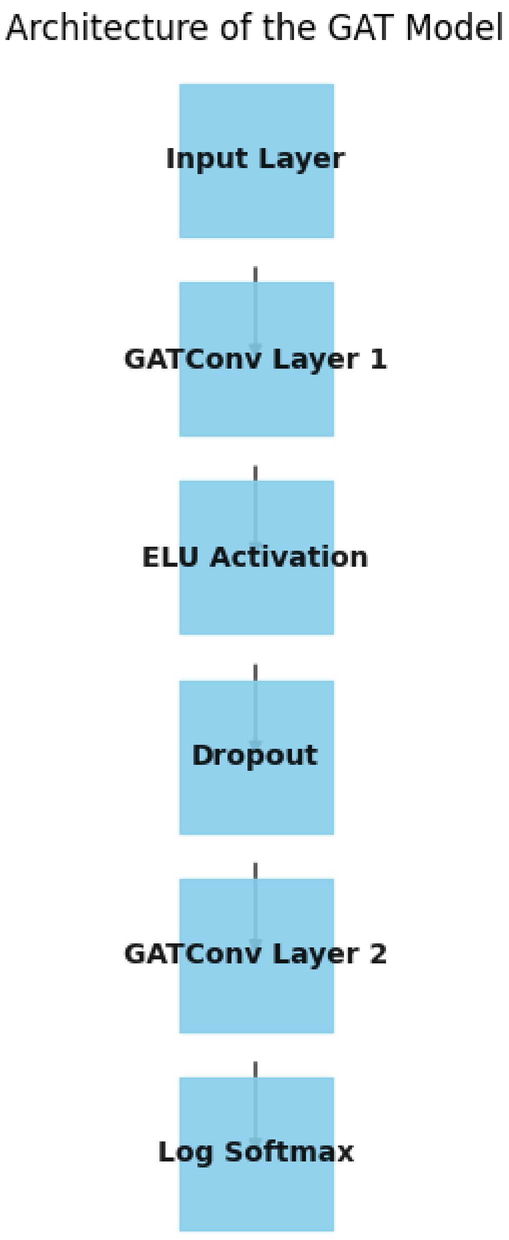
The chosen evaluation metrics and their analysis of GAT model are:

1. Accuracy: The accuracy of the model is 88.75% showing that model has learned significant patterns from Cora dataset.
2. Precision: The precision of the model is 89.36% indicating that the model predicts the correct node for 89.36% of the time. This hsows that the model has relatively low false negative errors.
3. Recall: The recall is 88.75% which means model has 88.75% chances of detecting actual positive instances.
4. F1 Score: The F1 score is 88.83% suggesting that the model is effective in identifying the correct nodes.

The model worked well due to following factors:

- ✓ Attention Mechanism in GAT: The allocation of weights to nodes helps the model to focus on most relevant nodes and give higher overall accuracy.
- ✓ Dropout regularization: The use of dropout layers prevents the overfitting in the model by zeroing out the element in input.
- ✓ Hyperparameters: The chosen hyperparameters like dropout=0.6, learning rate=0.01, weight_decay=5e-4 tuned well with the dataset and model. The presence of large hidden layers also allows the model to learn complex representations.

The model architecture is:



**Section 4: Comprehensive analysis**

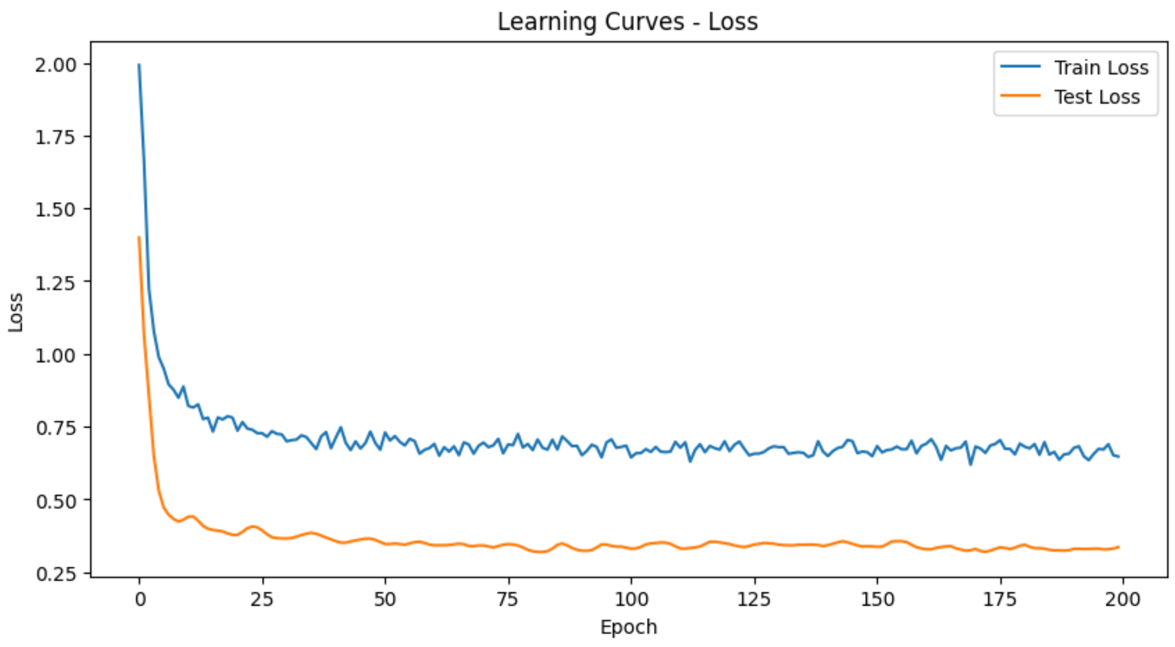
I conducted 3 Ablation studies on the proposed model and the output is given in the table below:

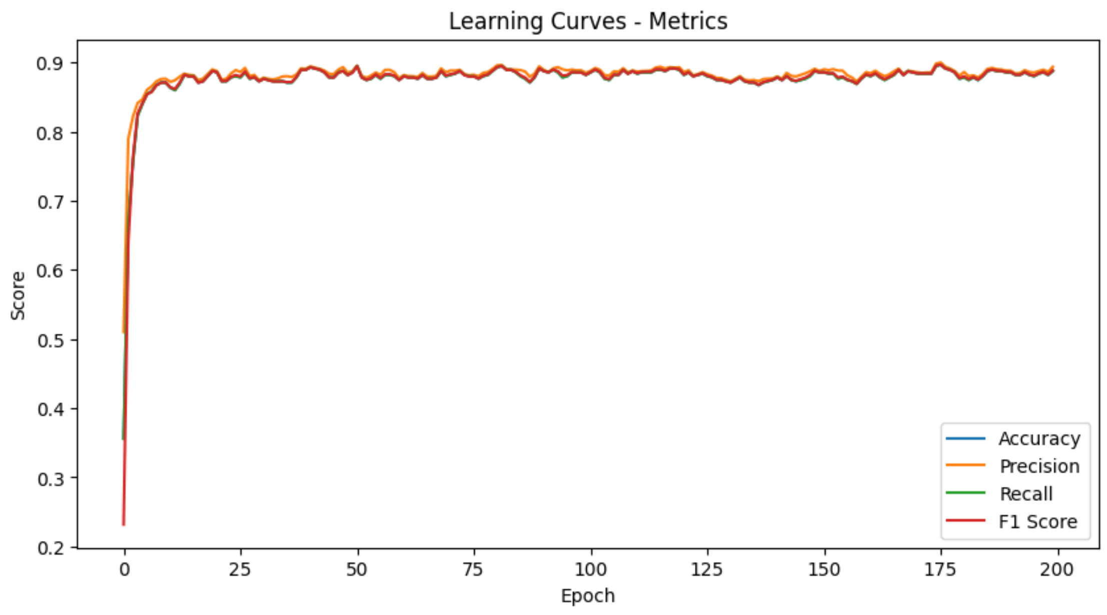
| **Original Model** | **ABLATION 1: Remove Dropout layer** |
|---|---|
| `Epoch: 200, Loss: 0.6474, Accuracy: 0.8875 Precision: 0.8936, Recall: 0.8875, F1 Score: 0.8883` | `Loss: 0.6825, Accuracy: 0.8875 Precision: 0.8936, Recall: 0.8875, F1 Score: 0.8883` |
| **Original Model** | **ABLATION 2: Change in activation function** |
| `Epoch: 200, Loss: 0.6474, Accuracy: 0.8875` | `[ReLU Activation] Epoch: 200, Loss: 0.6676, Accuracy: 0.8875` |

| | |
|---|---|
| `Precision: 0.8936, Recall: 0.8875, F1 Score: 0.8883` | `Precision: 0.8936, Recall: 0.8875, F1 Score: 0.8883` |
| **Original Model** | **ABLATION 3:Reduce Number of Attention Heads** |
| `Epoch: 200, Loss: 0.6474, Accuracy: 0.8875 Precision: 0.8936, Recall: 0.8875, F1 Score: 0.8883` | `[Fewer Heads] Epoch: 200, Loss: 0.6995, Accuracy: 0.8875 Precision: 0.8896, Recall: 0.8875, F1 Score: 0.8872` |

Comparison between multiple model architectures:

| **GCN** | **GAT** |
|---|---|
| `Loss: 0.1236` | `Loss: 0.6474` |
| `Accuracy: 0.8727` | `Accuracy: 0.8875` |
| `Precision: 0.8743` | `Precision: 0.8936` |
| `Recall: 0.8708` | `Recall: 0.8875` |
| `F1 Score: 0.8717` | `F1 Score: 0.8883` |





The metrics of GAT shows better model efficiency than GCN. The accuracy and precision are higher for GAT which indicates that it predicts the correct nodes and less false positives.

**References:**

- Karami, F. (2023, July 20). Understanding Graph Attention Networks: A Practical Exploration. Medium. https://medium.com/@farzad.karami/understanding-graph-attention-networks-a-practical-exploration-cf033a8f3d9d
- Noda, K. (2023, January 2). Hands-on Graph Neural Networks with PyTorch Geometric (1): Cora Dataset. Medium. https://medium.com/@koki_noda/ultimate-guide-to-graph-neural-networks-1-cora-dataset-37338c04fe6f
- Hsu, V. (2024, April 15). [AI] Train the first GNN model for Cora data with PyTorch. Medium. https://medium.com/@garraypierce/train-the-first-gnn-model-for-cora-data-with-pytorch-58a7f3706183
- Futia, G. (2022, January 23). Graph Attention Networks Under the Hood. Medium. https://towardsdatascience.com/graph-attention-networks-under-the-hood-3bd70dc7a87