Intern Name: Meklit Habtamu

1. Introduction

The objective of this experiment is to implement and compare multiple node classification models on a citation network dataset, **Cora**, using PyTorch Geometric. The models explored include Multi-Layer Perceptron (MLP), Graph Convolutional Network (GCN), GraphSAGE and Graph Attention Network v2 (GATv2). The goal is to understand how different architectures perform on graph-structured data and to analyze the impact of graph-based neighborhood aggregation.

2. Dataset

The Cora dataset was used which is a citation network consisting of

- Nodes: 2,708 scientific publications
- Edges: 5,429 citation links
- Node features: 1,433-dimensional bag-of-words vectors
- Labels: 7 classes corresponding to research topics
- Masks: Training, validation, and test splits provided

3. Model Formulation

Each model is formulated in terms of Encoder, Decoder, Ground Truth and Loss

Model	**Encoder**	**Decoder**	**Ground Truth**	**Loss**
MLP	Fully connected layers transforming node features	Final Dense layer with softmax	Node labels	NLL loss on training nodes
GCN	GCNConv layers aggregating neighbor features	Final GCNConv layer for classification	Node labels	NLL loss on training nodes

GraphSAGE	GraphSAGE layers aggregating neighbor info	Final GraphSAGE layer for classification	Node labels	NLL loss on training nodes
GATv2	Graph attention layers weighting neighbors	Final GATv2 layer producing logits	Node labels	NLL loss on training nodes

4. Experimental Setup

- Framework: PyTorch + PyTorch Geometric
- Training: Adam optimizer with learning rate 0.01 and weight decay 5e-4
- Hidden Dimensions: 64, 128
- Number of Layers: 2, 3
- Dropout: 0.5 applied after each hidden layer
- Epochs: 100
- Evaluation Metrics: Accuracy, Macro F1-score, Log Loss

5. Results

Model	**Hidden**	**Layers**	**Accuracy**	**Macro F1**	**Log Loss**
MLP	64	2	0.578	0.564	1.323
GCN	64	2	0.813	0.804	0.601
GraphSAGE	64	2	0.799	0.790	0.643
GATv2	64	2	0.786	0.775	0.804

MLP	64	3	0.555	0.546	1.900
GCN	64	3	0.784	0.779	0.876
GraphSAGE	64	3	0.793	0.787	1.019
GATv2	64	3	0.802	0.792	1.169
MLP	128	2	0.568	0.558	1.351
GCN	128	2	0.806	0.801	0.615
GraphSAGE	128	2	0.804	0.798	0.635
GATv2	128	2	0.769	0.771	0.960
MLP	128	3	0.558	0.550	1.850
GCN	128	3	0.800	0.795	0.819
GraphSAGE	128	3	0.784	0.774	1.070
GATv2	128	3	0.749	0.753	1.834

6. Analysis

1. **Graph Structure Matters**
   MLP, which ignores graph connectivity, performs poorly compared to graph-based models. Graph aggregation improves predictive performance.
2. **GCN Performance**
   GCN consistently achieves the highest accuracy and lowest log loss across most configurations, demonstrating the effectiveness of simple neighbor averaging on citation networks.
3. **GraphSAGE**
   Slightly behind GCN, GraphSAGE is robust and can handle inductive settings, where nodes unseen during training can still be classified.
4. **GATv2 Attention Mechanism**
   While attention allows weighting neighbors differently, on Cora the benefit is modest. Small datasets with uniform connectivity may not fully exploit attention mechanisms.
5. **Hidden Dimensions & Layers**
   Increasing hidden dimensions or adding layers does not always improve performance; overfitting or oversmoothing can occur. For Cora, **64 hidden units with 2 layers** is optimal.
6. **Loss and Ground Truth**
   NLL loss on training nodes provides a reliable supervision signal. Validation accuracy was used to select the best epoch for evaluation.

7. Conclusion

This study implements MLP, GCN, GraphSAGE, and GATv2 models on the Cora dataset, formulates each model in terms of Encoder, Decoder, Ground Truth, and Loss, and performs a full hyperparameter sweep. Key observations:

- Graph-based models outperform MLP significantly.
- GCN achieves the highest accuracy and lowest log loss, making it the most effective model on Cora.
- GraphSAGE and GATv2 provide alternative approaches with their own strengths for more complex or inductive tasks.
- Careful tuning of layers and hidden dimensions is crucial to avoid overfitting or oversmoothing.

Overall, the experiments demonstrate the importance of graph structure in node classification and provide insights into the performance of various graph neural network architectures.