Name: Ahmed Haitham

ID: 2205241

Report: Explanation of the Graph Neural Network (GNN) Code

1. Introduction

This project demonstrates a simple example of using a Graph Neural Network (GNN) — specifically GraphSAGE — to classify nodes in a small graph.

Each node represents a user, and the goal is to determine whether the user is benign or malicious based on node features and graph connections.

2. Required Libraries

The code imports PyTorch, torch_geometric, and the SAGEConv layer, which allows the model to aggregate information from neighboring nodes.

3. Graph Construction

The graph contains 6 nodes. Each node has two features. Nodes 0–2 represent benign users, and nodes 3–5 represent malicious users.

The edges define two tightly connected communities (benign cluster and malicious cluster) with one weak connection between them.

4. Labels

The labels y = [0,0,0,1,1,1] identify which nodes are benign (0) and which are malicious (1).

5. GraphSAGE Model

The model uses two SAGEConv layers:

- First layer: 2 → 4 dimensions

- Second layer: 4 → 2 dimensions

ReLU activation is used between layers, and log_softmax is used at the output.

6. Training

The model is trained for 200 epochs using Adam optimizer and negative log-likelihood loss.

During training, the model learns patterns in node features and graph structure.

7. Evaluation

After training, the model predicts each node's class. Typically, the model successfully classifies all nodes as [0,0,0,1,1,1].

8. Conclusion

This example demonstrates how Graph Neural Networks can combine node features with graph structure to make accurate predictions.

It illustrates the full pipeline: graph creation, model building, training, and evaluation.