Node Classification Using Graph Neural Network (GraphSAGE)

Name: Nadine Rasmy

ID: 2205203

1. Introduction

This task focuses on applying a Graph Neural Network (GNN) model using the GraphSAGE technique to perform node classification on a small graph dataset. The main goal of this work is to classify each node as either a benign user or a malicious user based on its features and its connections with other nodes in the graph.

2. Graph Data Description

2.1 Node Features

The dataset consists of a simple graph with six nodes. Each node is represented by two numerical features that describe its type. The encoding used is simple and easy to understand:

- [1, 0] represents a benign user
- [0, 1] represents a malicious user

These values are stored in a tensor and used as input to the neural network model.

2.2 Edges (Node Connections)

The relationships between nodes are defined using an edge index. This edge index specifies which nodes are connected to each other and forms the structure of the graph. These connections allow the model to learn not only from the node itself but also from its neighbors.

2.3 Node Labels

Each node is assigned a true label that indicates its real class. The value 0 is used for benign users and the value 1 is used for malicious users. These labels are used during the training process to help the model learn correctly.

3. Model Architecture

The Graph Neural Network model is built using two GraphSAGE convolution layers. Between these layers, a ReLU activation function is applied to introduce non-linearity into the model. At the output layer, a LogSoftmax function is used to produce classification scores for each class.

4. Training Process

The model is trained using the Adam optimizer, which updates the model weights efficiently during training. The loss function used is Negative Log Likelihood (NLLLoss), which is suitable for classification problems.

The training process runs for 50 epochs. In each epoch, the model performs the following steps:

1. Generate predictions for all nodes.
2. Calculate the loss by comparing predictions with true labels.
3. Update the model weights using backpropagation.

5. Testing and Results

After completing the training, the model is switched to evaluation mode. The trained model is then used to predict the class of each node in the graph. The predicted class for each node represents whether it is classified as benign or malicious.

6. Conclusion

In this task, we successfully applied a Graph Neural Network using GraphSAGE for node classification. We created a small graph dataset, defined node features and connections, trained the model, and tested its performance. This experiment shows how GNNs can be effectively used in applications such as malicious user detection, social network analysis, and cybersecurity.