# Fraud Detection in Elliptical Bitcoin Dataset Using GNN

**Section 1: Motivation and Explanation of data**

In this assignment, Elliptical Bitcoin dataset is considered, where it’s combination of the 3 csv files, Edgelist, classes, and features. This dataset is collected from bitcoin blockchain. Nodes in this dataset are transaction information. An edge represents relation between one transaction and another. Each node has 166 features and is labelled as either scam, licit or unknown.

The motivation behind selecting this dataset is to combat the threat of hackers attempting to infiltrate the blockchain and steal funds, making it challenging to trace the original IP address of the device used. Identifying illicit transactions is crucial for maintaining the integrity of the network and ensuring user trust.

The preprocessing of the dataset involves assigning edges, classes, and features. The dataset is divided into three parts based on the transaction ID, and the class labels are encoded to distinguish between illicit, licit, and unknown transactions. Due to the dataset's density, batch normalization is employed to alleviate computational power constraints. This technique calculates the average of each batch and divides it by each element, thereby optimizing processing efficiency.

A blue dots in a circle

Description automatically generated

**Fig 1**: Visualisation of the Elliptical bitcoin dataset.

Since the dataset is too dense, considering only 100 nodes from the dataset for visualisation. Here the nodes represent the entities, and edges represent the connection between the nodes.

**Section 2: Appropriateness and Explanation of the model**

**Architecture of the GAT model:**

**A diagram of a graph

Description automatically generated with medium confidence**

Here GAT architecture is used, previously due to the model complexity the number epochs the data trained on was high (400 epochs) which required 2 TPUs running for almost 10 minutes, due to lack of computational power, here the model is run only for 100 epochs but for the trade-off additional input layer is added and dropout rate is increased as to extract hidden features from the transaction and the dropout rate is increased so the model is not over fit.

Attention Coefficient Normalisation:

A mathematical equation with a line of letters

Description automatically generated with medium confidence

Weighted Sum of Neighbours:

A number of mathematical equations

Description automatically generated with medium confidence

Batch Normalisation:

A mathematical equation with black text

Description automatically generated

Loss Function:

Here BCE loss function is used as the output label is classified into 2 categories. It helps to measure the difference between predicted class and the actual class label. This loss function is used with sigmoid function in the final layer.

A black text on a white background

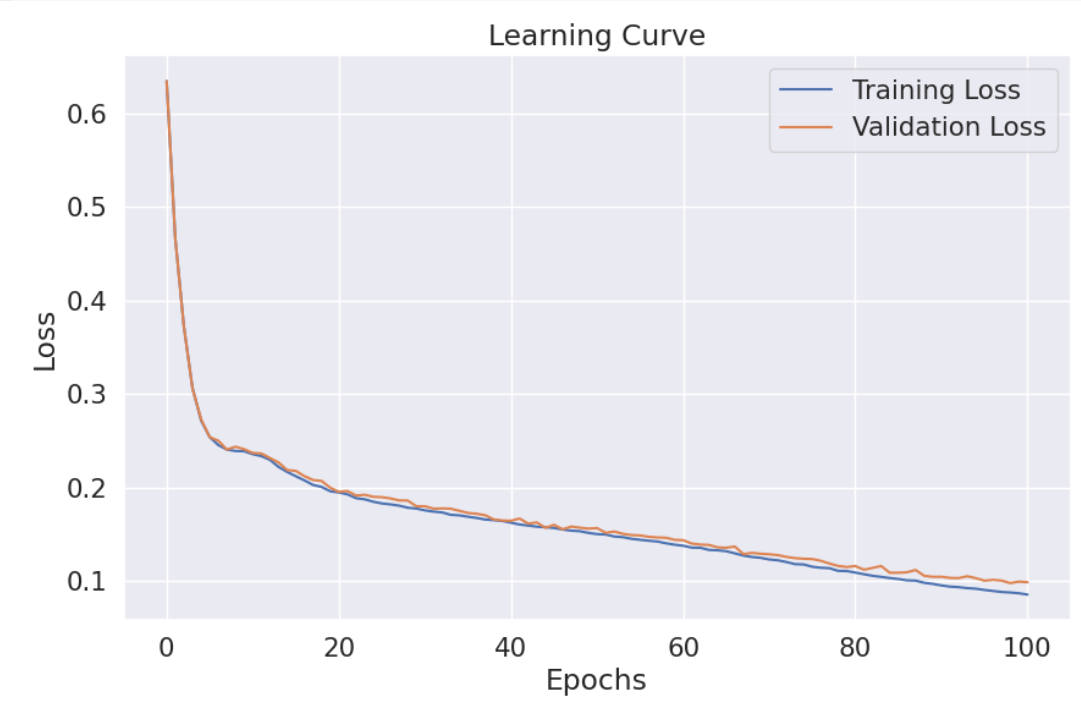
Description automatically generated

In this GNN architecture, GAT (Graph Attention network) is used, as GATs capture local and global dependencies. Attention mechanism in GATs helps them giving different attention or weights to their neighbours based on the relational/ relevance. By giving attention to different nodes, it can efficiently capture important feature information. In addition to that, given the complexity of the dataset the GATs work best in accurately detecting illicit transactions.

When we look into GCN architecture, these models find it difficult to capture the long range dependencies which makes the model to perform bad.

**Section 3: Insights and Results**

Learning Curve:



Confusion Matrix:

A diagram of a confusion matrix

Description automatically generated

Precision, Recall, Accuracy, f1-score:

A screenshot of a graph

Description automatically generated

Model insights signify that the accuracy of the model is around 97%. Precision is 98% for licit transactions and 89% for illicit transactions. Precision is a metric used to measure positive predictions made from the model. The recall is around 99% and 81% respectively. High recall signifies model identifies most of the true positives.

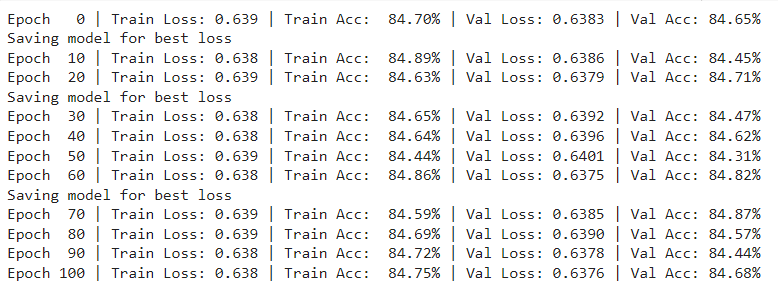
Here, combining the GAT layer with a fully connected layer to enhance the model performance with smaller computing power. Applying a single graph convolutional layer can help learn node embeddings from the data. Using this layer the information between nodes is captured based on the adjacency matrix.

Adding a fully connected dense layer can help the model learn complex interactions and perform well for transaction classification. The fully connected layer reduces dimensionality and it is connected to the ReLu activation layer to minimize the loss and update weights.

**Section 4: Comprehensive Analysis**

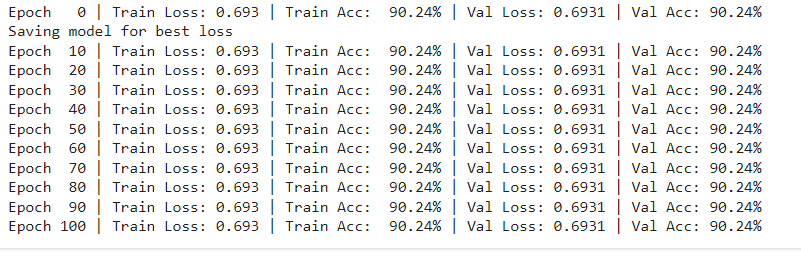
For the ablation model, the batch normalization is removed and the ReLu activation function is added instead of Leaky\_ReLu. From the results, the ablation model training loss and accuracy remains constant at 0.638 and 84.9% respectively. The model achieves stability at the early stage of around 10 epochs. There are no significant improvements in the results. It has achieved good accuracy for training and validation sets.

Below are results of the Ablation model:



The GAT model is compared with GCN architecture and GraphSage architecture. The code is taken from the baseline. GAT employs attention mechanisms to assign weights to different neighbors during the aggregation process, allowing the model to focus more on relevant neighbors [1]. The architecture typically involves multiple attention heads, which enable the model to capture diverse aspects of the neighborhood information. It aggregated information from attention heads to improve representation. When it comes to GCN, these models train based on the adjacency matrix and gather information from the neighbor nodes. Each layer of a GCN applies a linear transformation followed by an aggregation function, typically a summation or mean, to combine the features of neighboring nodes.  GraphSage works best when working on large graphs offering scalability and robust performance. The input layers aggregate information from the sampled neighbors.

Results for GCN model:



Results for GraphSage model:

**A screenshot of a computer

Description automatically generated**

**This model code was taken from baseline models from GitHub.**

**Source :**  [https://medium.com/@adjcs224w/illicit-transaction-detection-in-graph-networks-c0d381d85999](https://www.google.com/url?q=https%3A%2F%2Fmedium.com%2F%40adjcs224w%2Fillicit-transaction-detection-in-graph-networks-c0d381d85999)

**References:**

* Weber, M., Domeniconi, G., Chen, J., Weidele, D. K., Bellei, C., Robinson, T., & Leiserson, C. E. (2019). Anti-Money Laundering in Bitcoin: Experimenting with Graph Convolutional Networks for Financial Forensics. *ArXiv*. /abs/1908.02591
* AmolDominicJosh. (2023, December 15). *Illicit Transaction Detection in Graph Networks*.
* GAT fraud detection. illicit-f1=0.89. (n.d.). Kaggle.com
* Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2017). Graph Attention Networks. *ArXiv*. /abs/1710.10903
* Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Inductive Representation Learning on Large Graphs. *ArXiv*. /abs/1706.02216