# CS244W: Bitcoin Network Fraud Detection using Machine Learning with Graphs

**Sebastijan Trojer**
University of Ljubljana
st5804@student.uni-lj.si

**Matej Vrečar**
University of Ljubljana
mv90948@student.uni-lj.si

## 1 Application domain

### 1.1 Dataset

We are going to use the public Elliptic++ dataset [**?** **?** ], consisting of over 200 thousand Bitcoin (BTC) transactions and over 800 thousand different wallet addresses, which allows the detection of fraudulent transactions and addresses. The dataset is split into two parts, specifically the Elliptic++ Transactions dataset and the Elliptic++ Actors dataset.

#### 1.1.1 Elliptic++ Transactions

The Elliptic++ Transactions dataset consists of 49 graphs, recorded at different time steps (essentially 49 snapshots), where each node is a Bitcoin transaction and each edge represents money flow, where edges are labeled as either licit, illicit, or unknown, where the class proportions are 21%, 2% and 77% respectively. Each transaction consists of 183 features, 166 of which are inherited from the Elliptic dataset [**?** ], which include time step, 93 local features and 72 aggregate features. The remaining 17 node features are added in the Elliptic++ dataset and are referred to as augmented features, adding information about the node degrees (in/out edges), total amounts of BTC sent/received etc.

The authors also provide a detailed statistical analysis of the data, along with results of several experiments for fraud detection. They also recommend metrics, such as precision, recall, F1 score and Matthews Correlation Coefficient.

| Mean Nodes | Mean Edges | Avg. Node degree |
|---|---|---|
| 4158 | 4782 | 2.25 |

Table 1: Average nodes, edges and node degrees in Transactions dataset

#### 1.1.2 Elliptic++ Actors

The Elliptic++ Actors dataset is a network of over 800,000 different wallet addresses as nodes, where each node consists of 56 features, describing the amount of BTC received and sent, transaction times, and address interactions. Each node is also annotated in the same way as in the Elliptic++ Transactions dataset, and the labels are derived from the transaction labels. This dataset is also highly unbalanced, with only 2% of the wallets tagged as illicit and 31% as licit.

The dataset specifies two graphs, an Actor interaction graph and an Address-Transaction graph. The authors also provide two more graphs generated from the same data, specifically the Money Flow Transaction Graph, that shows BTC flow between transactions and the User Entity Graph, an address cluster graph, which allows potential address linking, to identify addresses controlled by the same user.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

### 1.2 Tasks

This specific dataset was chosen because it is a much richer version of the original Elliptic dataset [**?** ] and allows for both transaction- and wallet-level classification, as well as larger potential for temporal modeling, since wallet identities are provided and transaction features include block number of execution. Having such a large variety of data allows us to try different approaches, before deciding on one. This dataset also allows for semi-supervised learning since a large proportion of data is labeled as unknown, however, the annotated subset is presumably large enough, to potentially allow the data to be reduced to only that subset.

Given the rich dataset, the goal of the project is to focus on the Address-Transaction graph and apply graph learning methods for two separate tasks: In the first experiment, we want to try to determine whether an actor (wallet) is illicit or not, and in the second one we want to use the labels from the Elliptic++ Transactions to determine illicitness of transactions. The first task then represents a node-level classification experiment, while the second is an edge-level classification experiment. The reason for defining two tasks is that we hope that the difference in performance and difficulty might offer some insight into which task is generally harder and whether the two can complement each other.

Since the Address-Transaction graph is relatively large, and the majority of features are transaction features (edges in this case), we also plan to explore opportunities for transaction-based fraud detection from the Transactions Dataset, and finally work on the more promising data.

Another advantage of this dataset is that the authors performed experiments on transaction classification using ensemble models, which can serve as a baseline for our work.

## 2 Methodology

### 2.1 Baseline

For a baseline we decided to use two basic models. The first was a simple Graph Convolutional Network (GCN), a layer of which consisted of a GCNConv operator, layer normalization, and a ReLU activation function. In addition, a dropout was included in the end to avoid overfitting. The second baseline model was a model based on the SAGEConv operator. The rest of the model was identical to the first one. Both models were optimized using a grid search on the same parameter space:

- num_layers: [1, 3, 5]
- hidden_dim: [128, 256, 512]
- dropout: [0.1, 0.3, 0.5]
- learning rate: [0.0001, 0.001, 0.01, 0.1]

### 2.2 Anomaly detection on subgraphs

Prior research shows that it is worth treating fraud detection as an subgraph anomaly detection task, rather than node or edge classification task, as it yields better results. One option of doing that is to extract k-node graphlets from the original graphs and perform graph-level classification. Since the transaction graphs are relatively sparse (Table 1), graphlets consisting of 6 nodes were extracted using C++ graphlet sampling tool BLANT [1]. Since it is a sampling tool, we sampled up to 10 million graphlets for each transaction graph in the original dataset and removed duplicates. Graphlet counts per graph are shown in Figure 1. Since sampling 6-node graphlets gives us an abundance of samples, only the labeled samples were kept.

Graphlet labels were acquired using node labels, specifically, if any node in the graphlet was labeled as ilicit, the graphlet label was set to ilicit, and if not, graphlet label was set to the majority class of the nodes (licit or unlabeled). After that, the unlabeled graphlets were discarded. When using samples from the graph in the first time step, we kept around 400k labeled samples out of 3 million.

GCN was used for training, with the same methodology as for the baseline, with the addition of global mean pooling, which allowed for graph-level classification.

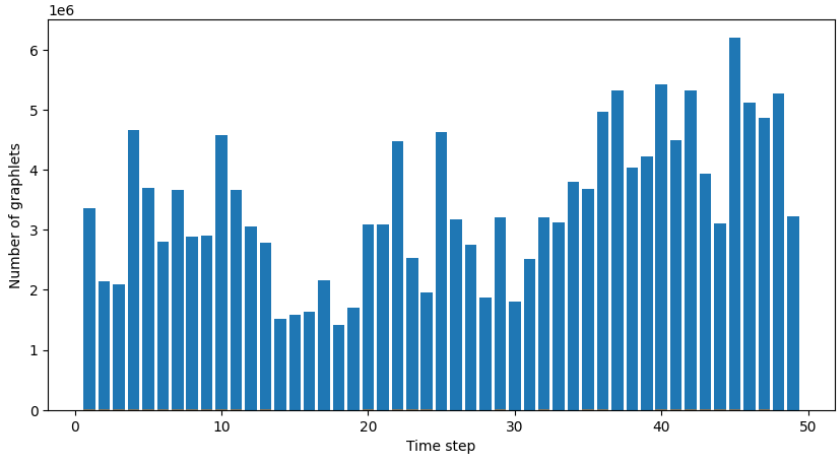


Figure 1: Graphlet counts per graph in each time step.

# 3 Results

## 3.1 Baseline

We will now discuss the quality of the two baseline models.

Table 2: Baseline model results.

| Method | Precision | Recall | F1 |
|---|---|---|---|
| GCN | 0.273 | 0.408 | 0.327 |
| SAGE | **0.737** | **0.445** | **0.555** |

As can be observed in Table 2, the SAGE approach performed significantly better than the GCN based approach. Overall, the result is not stellar, but still serves as a good baseline given the 10/90 positive/negative label imbalance.

## 3.2 Graph-level classification

Preliminary results show that high precision is achievable, however low recall renders the model useless. We plan on exploring other models, as well as approaches to balance the problem by using various modifications of the loss function.

# References

[1] Ahmet Hasan, Peter-Chung Chung, and Wayne Hayes. Graphettes: Constant-time determination of graphlet and orbit identity including (possibly disconnected) graphlets up to size 8. *PLoS ONE*, 12(8):e0181570, 2017.