Data Engineering project
Anti Money laundering in Bitcoin
Experimenting with Graph Convelational Network and Fedarated learning on graph

Kaitha Lavanya Lahari (20MCME23)
School of Computer and Information Sciences
University of Hyderabad
20mcme23@uohyd.ac.in

December 28, 2023

ABSTRACT Blockchain provides the unique and accountable channel for financial forensics by mining its open and immutable transaction data. A recent surge has been witnessed by training machine learning models with cryptocurrency transaction data for anomaly detection, such as money laundering and other fraudulent activities.This paper presents a holistic applied data science approach to fraud detection in the Bitcoin network with the help of elliptic dataset which contains 203,769 transaction nodes, 234,355 edges , 167 features per transaction.This project is to develop a federated learning model for predicting whether Bitcoin trans- actions are illicit or licit using a Graph Convolutional Network (GCN). The federated learning approach is employed to enable decentralized training on individual clients' datasets, ensuring pri- vacy and security while collectively improving the global model's accuracy.

Keywords: Bitcoin transactions, fraud detection, anomaly detection, machine learning models, cryptocurrency, blockchain, money laundering, illicit activities, ensemble learning, predictive analytics.

Contents

1 CONTRIBUTION IN PROJECT

1.1 WHY GCN FOR THE ELLIPTIC DATASET

Graph Convolutional Networks (GCNs) are particularly well-suited for tasks involving graph-structured data, and they have been shown to be effective in various domains, including social network analysis, biology, and citation networks. However, whether a GCN is the "best" model for predicting elliptic datasets depends on the specific characteristics of the data and the task at hand.

Performance Metrics

The final performance results, as reported, indicate that GCN outperforms other approaches. Specifically, GCN provides the best results in terms of recall (0.790) and F1-score (0.844). While precision is slightly lower than Decision Tree and Random Forest, it still outperforms other approaches with a precision of 0.906.

Comparison with Other Models

The text mentions a comparison with Decision Tree, Random Forest, and Graph Attention Network (GAT). GCN outperforms Decision Tree and Random Forest in terms of recall and F1-score. GAT, another graph-based approach, is mentioned to perform better than a simple dense network but falls short of GCN and Random Forest in terms of overall performance. While Graph Convolutional Networks (GCNs) have demonstrated success in handling graph-structured data, including various applications, it doesn't mean that they are always the best choice for every dataset or prediction task. The choice of a model depends on the characteristics of the data, the nature of the problem, and various other factors.

Dataset Characteristics

The dataset used for the analysis is described as unbalanced, with more licit than illicit transactions in a ratio of approximately 1 to 10. Despite the dataset being unbalanced, GCN still demonstrates superior performance, particularly in terms of recall and F1-score.

Graph-Based Advantage

The paper emphasizes that, from an operational standpoint, graph-based methods (including GCN) perform better than baseline approaches. GCN is explicitly mentioned as performing better than Random Forests, encouraging further experimentation with GCN for AML/CFT (Anti-Money Laundering/Counter Financing of Terrorism) purposes.

1.2 ellipticBitcoinDatase over EllipticPlusPlus

Elliptic Bitcoin Transaction Dataset

Pros:

- **Real-world Relevance:** The dataset is derived from real-world Bitcoin transactions, reflecting the dynamics and characteristics of actual transactions on the Bitcoin blockchain.
- **Widely Used in Research:** It has been widely used in research, establishing itself as a benchmark for developing and evaluating models related to cryptocurrency transaction analysis.
- **Graph Structure:** Includes features related to the graph structure of transactions, making it suitable for graph-based machine learning models like Graph Convolutional Networks (GCNs).

- **Anomaly Detection and Classification:** Labels for transactions, such as "licit" and "illicit," allow for tasks like anomaly detection and classification.

Cons:

- **Limited to Bitcoin:** Specific to Bitcoin transactions, limiting its representativeness for transactions on other blockchains or cryptocurrencies.
- **Single Source:** May have limitations in representing the diversity of transactions, originating from a single source.
- **Potential Biases:** Biases in the dataset may impact the generalizability of models trained on it, depending on the source and collection methodology.
- **Fixed Time Period:** Data collected within a specific time period, and the dynamics of Bitcoin transactions may evolve over time, affecting model performance on more recent data.

EllipticPlusPlus Dataset (Hypothetical)

Note: The following points are hypothetical and based on common considerations for evaluating datasets.

Pros:

- **Expanded Coverage:** If "EllipticPlusPlus" includes data from multiple cryptocurrencies or a broader range of transactions, it may offer more comprehensive insights.
- **Recent Data:** If regularly updated, it may provide more recent and relevant information about cryptocurrency transactions.

Cons:

- **Limited Adoption:** If ”EllipticPlusPlus” is a new dataset, it might not have gained the same level of adoption and scrutiny as the ”Elliptic Bitcoin Transaction” dataset.
- **Data Quality:** The quality of data collection and labeling processes may vary, potentially affecting the reliability of the dataset for machine learning tasks.
- **Compatibility with Existing Research:** Researchers and practitioners might prefer datasets with a history of use in the literature for confidence in relevance and comparability to existing studies.

1.3 Understanding Bitcoin Transactions

Bitcoin transactions are a fundamental aspect of the Bitcoin network, enabling the transfer of value between participants. Here is a detailed explanation of how Bitcoin transactions are conducted:

- **Wallets:**
  - *Digital Wallets:* Users need a digital wallet to send and receive bitcoins. Wallets store private keys, which are cryptographic keys that prove ownership of bitcoins and facilitate transactions.
- **Addresses:**
  - *Public Addresses:* Each wallet has a public address, a unique identifier derived from the wallet's public key. This is the address to which others can send bitcoins.
- **Transaction Inputs and Outputs:**
  - *Inputs:* When a user initiates a transaction, they reference unspent transaction outputs (UTXOs) from previous

transactions. These UTXOs serve as the inputs for the new transaction.

  - *Outputs:* Transactions have one or more outputs, each specifying an amount and a recipient's address. The sum of the inputs must equal or exceed the sum of the outputs, with the difference (change) going back to the sender.

- **Transaction Signing:**
  - *Digital Signatures:* To prove ownership and authorize a transaction, the sender uses their private key to create a digital signature. The signature is unique to the transaction and ensures that only the owner of the private key can spend the bitcoins.

- **Broadcasting the Transaction:**
  - *Network Propagation:* The signed transaction is broadcast to the Bitcoin network. Nodes in the network verify the transaction's validity, including checking the digital signature and ensuring that the inputs are unspent.

- **Confirmation:**
  - *Mining:* Valid transactions are bundled into blocks by miners, who compete to solve complex mathematical problems to add the block to the blockchain.
  - *Confirmation:* Once a block is added to the blockchain, the transaction is considered confirmed. The more blocks added after a transaction, the more secure and irreversible it becomes.

- **Verification:**
  - *Node Validation:* All nodes on the network independently validate and store the entire blockchain. This decentralized validation ensures the integrity of the entire system.

- **Privacy and Anonymity:**
  - *Pseudonymity:* While Bitcoin transactions are recorded on the public blockchain, users are identified by cryptographic addresses, providing a degree of privacy. However, it's not entirely anonymous.

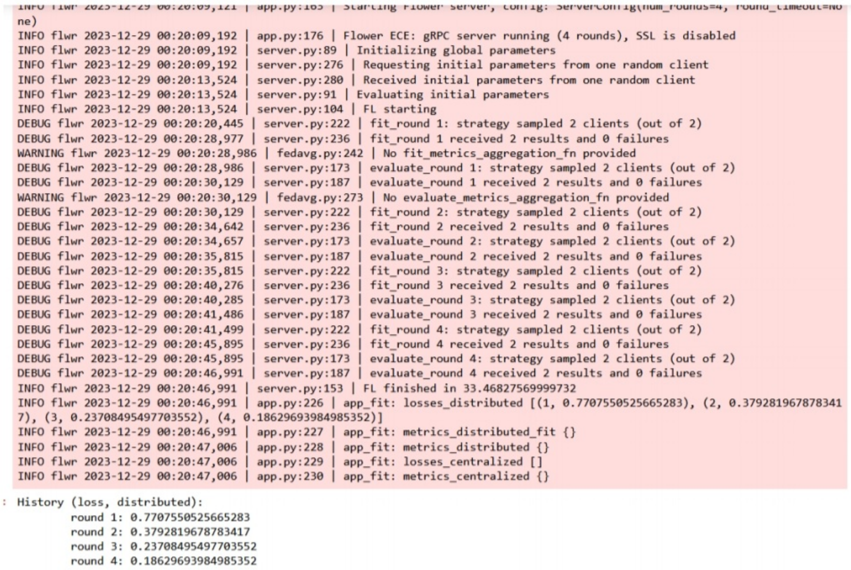
1.4 Work Done Related to Federated Learning

Federated Learning Workflow

- Edge devices receive a copy of a global model from a central server.
- The model is trained locally on the data residing on the edge devices.
- Global model weights are updated during training on each worker.
- A local copy is sent back to the central server.
- The server receives various updated models and aggregates the updates, thereby improving the global model and preserving the privacy of data on which it was being trained.

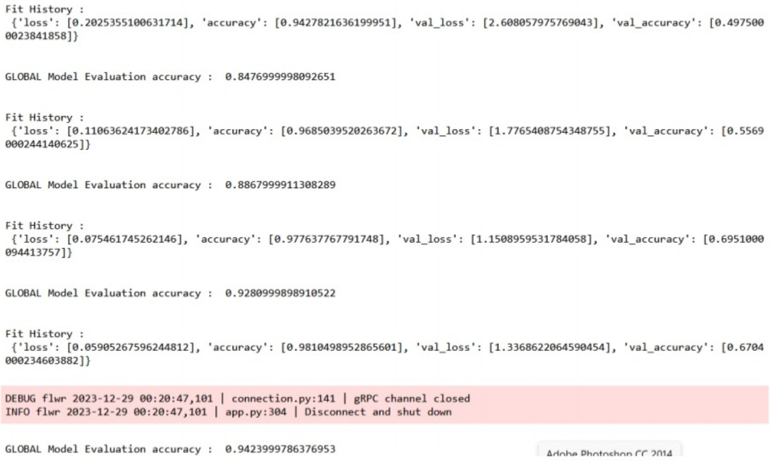
Federated Learning Environment using FLOWER Framework

I have developed a Federated Learning environment using the FLOWER framework. I took a model from TensorFlow Keras (an open-source neural networks library) and implemented it in a federated learning setup. The MNIST dataset has been used to train this model, focusing on the task of Image Classification. The Federated system includes a minimum of 2 clients. These clients train local models on their respective machines with non-IID subsets of the MNIST dataset. (Non-IID data distribution implies that the clients have different subsets of the data, ensuring a more realistic and challenging Federated Learning scenario). The clients then send back their trained models to the server. Upon receiving all the models from the clients, the server coordinates the training process by aggregating model updates and broadcasting the global model updates.



```
INFO flwr 2023-12-29 00:20:09,121 | app.py:163 | Starting Flower server, config: ServerConfig(num_rounds=4, round_timeout=No
ne)
INFO flwr 2023-12-29 00:20:09,192 | app.py:176 | Flower ECE: gRPC server running (4 rounds), SSL is disabled
INFO flwr 2023-12-29 00:20:09,192 | server.py:89 | Initializing global parameters
INFO flwr 2023-12-29 00:20:09,192 | server.py:276 | Requesting initial parameters from one random client
INFO flwr 2023-12-29 00:20:13,524 | server.py:280 | Received initial parameters from one random client
INFO flwr 2023-12-29 00:20:13,524 | server.py:91 | Evaluating initial parameters
INFO flwr 2023-12-29 00:20:13,524 | server.py:104 | FL starting
DEBUG flwr 2023-12-29 00:20:20,445 | server.py:222 | fit_round 1: strategy sampled 2 clients (out of 2)
DEBUG flwr 2023-12-29 00:20:28,977 | server.py:236 | fit_round 1 received 2 results and 0 failures
WARNING flwr 2023-12-29 00:20:28,986 | fedavg.py:242 | No fit_metrics_aggregation_fn provided
DEBUG flwr 2023-12-29 00:20:28,986 | server.py:173 | evaluate_round 1: strategy sampled 2 clients (out of 2)
DEBUG flwr 2023-12-29 00:20:30,129 | server.py:187 | evaluate_round 1 received 2 results and 0 failures
WARNING flwr 2023-12-29 00:20:30,129 | fedavg.py:273 | No evaluate_metrics_aggregation_fn provided
DEBUG flwr 2023-12-29 00:20:30,129 | server.py:222 | fit_round 2: strategy sampled 2 clients (out of 2)
DEBUG flwr 2023-12-29 00:20:34,642 | server.py:236 | fit_round 2 received 2 results and 0 failures
DEBUG flwr 2023-12-29 00:20:34,657 | server.py:173 | evaluate_round 2: strategy sampled 2 clients (out of 2)
DEBUG flwr 2023-12-29 00:20:35,815 | server.py:187 | evaluate_round 2 received 2 results and 0 failures
DEBUG flwr 2023-12-29 00:20:35,815 | server.py:222 | fit_round 3: strategy sampled 2 clients (out of 2)
DEBUG flwr 2023-12-29 00:20:40,276 | server.py:236 | fit_round 3 received 2 results and 0 failures
DEBUG flwr 2023-12-29 00:20:40,285 | server.py:173 | evaluate_round 3: strategy sampled 2 clients (out of 2)
DEBUG flwr 2023-12-29 00:20:41,486 | server.py:187 | evaluate_round 3 received 2 results and 0 failures
DEBUG flwr 2023-12-29 00:20:41,499 | server.py:222 | fit_round 4: strategy sampled 2 clients (out of 2)
DEBUG flwr 2023-12-29 00:20:45,895 | server.py:236 | fit_round 4 received 2 results and 0 failures
DEBUG flwr 2023-12-29 00:20:45,895 | server.py:173 | evaluate_round 4: strategy sampled 2 clients (out of 2)
DEBUG flwr 2023-12-29 00:20:46,991 | server.py:187 | evaluate_round 4 received 2 results and 0 failures
INFO flwr 2023-12-29 00:20:46,991 | server.py:153 | FL finished in 33.46827569999732
INFO flwr 2023-12-29 00:20:46,991 | app.py:226 | app_fit: losses_distributed [(1, 0.7707550525665283), (2, 0.379281967878341
7), (3, 0.23708495497703552), (4, 0.18629693984985352)]
INFO flwr 2023-12-29 00:20:46,991 | app.py:227 | app_fit: metrics_distributed_fit {}
INFO flwr 2023-12-29 00:20:47,006 | app.py:228 | app_fit: metrics_distributed {}
INFO flwr 2023-12-29 00:20:47,006 | app.py:229 | app_fit: losses_centralized []
INFO flwr 2023-12-29 00:20:47,006 | app.py:230 | app_fit: metrics_centralized {}
: History (loss, distributed):
        round 1: 0.7707550525665283
        round 2: 0.3792819678783417
        round 3: 0.23708495497703552
        round 4: 0.18629693984985352
```

Figure 1: Federated Learning Workflow - server output

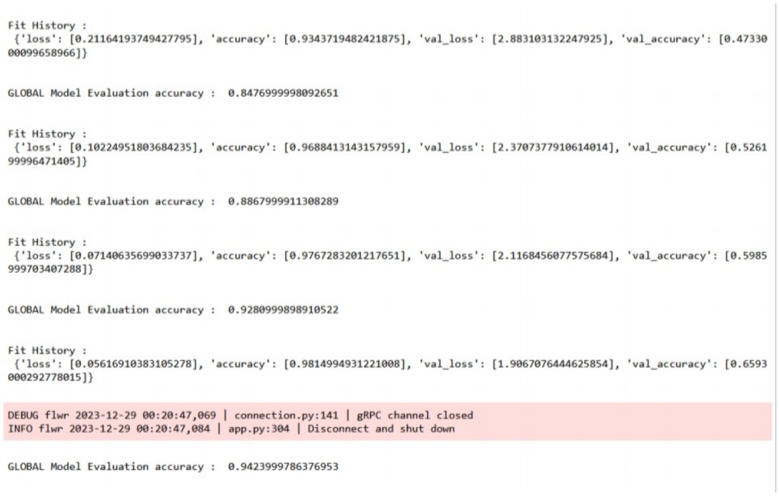


```
Fit History :
 {'loss': [0.2025355100631714], 'accuracy': [0.9427821636199951], 'val_loss': [2.608057975769043], 'val_accuracy': [0.497500
0023841858]}


GLOBAL Model Evaluation accuracy :  0.8476999998092651


Fit History :
 {'loss': [0.11063624173402786], 'accuracy': [0.9685039520263672], 'val_loss': [1.7765408754348755], 'val_accuracy': [0.5569
000244140625]}


GLOBAL Model Evaluation accuracy :  0.8867999911308289


Fit History :
 {'loss': [0.075461745262146], 'accuracy': [0.977637767791748], 'val_loss': [1.1508959531784058], 'val_accuracy': [0.6951000
094413757]}


GLOBAL Model Evaluation accuracy :  0.9280999898910522


Fit History :
 {'loss': [0.05905267596244812], 'accuracy': [0.9810498952865601], 'val_loss': [1.3368622064590454], 'val_accuracy': [0.6704
000234603882]}

DEBUG flwr 2023-12-29 00:20:47,101 | connection.py:141 | gRPC channel closed
INFO flwr 2023-12-29 00:20:47,101 | app.py:304 | Disconnect and shut down


GLOBAL Model Evaluation accuracy :  0.9423999786376953
```

Figure 2: Federated Learning Workflow - client1 output

Federated learning using elliptic dataset with GCN model

Taking example as the above model we have implemented as the same using GCN model. we are getting parameters but the model



```
Fit History :
 {'loss': [0.21164193749427795], 'accuracy': [0.9343719482421875], 'val_loss': [2.883103132247925], 'val_accuracy': [0.47330
00099658966]}

GLOBAL Model Evaluation accuracy :  0.8476999998092651

Fit History :
 {'loss': [0.10224951803684235], 'accuracy': [0.9688413143157959], 'val_loss': [2.3707377910614014], 'val_accuracy': [0.5261
99996471405]}

GLOBAL Model Evaluation accuracy :  0.8867999911308289

Fit History :
 {'loss': [0.07140635699033737], 'accuracy': [0.9767283201217651], 'val_loss': [2.1168456077575684], 'val_accuracy': [0.5985
999703407288]}

GLOBAL Model Evaluation accuracy :  0.9280999898910522

Fit History :
 {'loss': [0.05616910383105278], 'accuracy': [0.9814994931221008], 'val_loss': [1.9067076444625854], 'val_accuracy': [0.6593
000292778015]}

DEBUG flwr 2023-12-29 00:20:47,069 | connection.py:141 | gRPC channel closed
INFO flwr 2023-12-29 00:20:47,084 | app.py:304 | Disconnect and shut down

GLOBAL Model Evaluation accuracy :  0.9423999786376953
```

Figure 3: Federated Learning Workflow -client2 output

was not fitting into the frame work. Because as i mentioned earlier the latest tenserflow version was not allowing to create pickle file so, we are unable to fit the model into the frame work.

2 Conclusion

The purpose of this report was to summarize my contribution to the Data Engineering project. Along with my work, I was also helping my teammate in integration process. We worked together for the integration and did pair-programming most of the time for better communication and understanding of each other's work. I thank my teammates who motivated me and their hard work inspired me to contribute more to this project. Also, I would like to thank Dr. Mridula Verma for giving us an opportunity to learn about various crucial technologies and implement them to form a cohesive application

3 References

Codebase Repository

GitHub Repository: `https://github.com/Federatedlearning1/GCN`

Dataset

Dataset: `https://www.kaggle.com/datasets/ellipticco/elliptic-data-set/data`