**Genetic Algorithm With Deep Learning For Fraud Ethereum Transaction Detection**

Radhika Amar Desai , 22BCE1114

**ABSTRACT**

Recently blockchain has emerged as the leading technology in online trades. Ethereum smart contracts are particularly popular in the blockchain community. With expansion of online trading using Ethereum smart contracts, new fraudulent activities have also emerged. This study presents a deep learning model to detect fraud transactions. We have used the Ethereum Fraud Detection Datasetfrom Kaggle to train our model. Our model uses genetic algorithm to decide the hyperparameters for our Artificial Neural Network like learning rate and batch size that significantly influence the accuracy of the model. The deep learning model used is a 3 layered fully connected neural network. The loss function used here is the negative log likelihood loss. The model has obtained 99.991 percent accuracy. The model’s performance is also evaluated using other parameters like accuracy, precision and f1 score for the dataset. We have also compared our results to classic machine learning models like SVM, Random Forest (RF), Naïve Bayes and Logistic regression as well as to models proposed by other research papers listed in the literature review section. The proposed methodology outperforms all the models mentioned in the reviewed papers.

**INTRODUCTION**

In recent years, the rapid growth of digital transactions has brought immense convenience to consumers and businesses alike, but it has also given rise to increased fraudulent activities. Detecting and preventing fraudulent transactions is crucial to maintain trust and security in financial systems. Traditional rule-based approaches often fall short due to their inability to adapt to evolving fraud tactics. Therefore, developing an effective model for fraud transaction detection is essential to identify suspicious activities with higher accuracy and speed. This study aims to explore and propose a machine learning model tailored for fraud detection that can analyze transaction patterns, identify anomalous behavior, and adapt to new fraud strategies. By leveraging large datasets and applying advanced algorithmic techniques, this study will address the challenges of reducing false positives, increasing detection rates, and improving model efficiency. We compared various machine learning models and techniques for fraud transaction detection.

**LITERATURE REVIEW**

One of the most popular fraud schemes used in Ethereum is the Ponzi Scheme. Weili Chen et al. [1] presented a classification model to detect smart Ponzi schemes. They obtained 200 Ponzi schemes by examining more than 3000 open source smart contracts on the Ethereum platform. Their model increases the precision to 0.95 and outperforms the traditional models like XG Boost and one-class SVM.

Aljofey et al. [2] uses an ensemble classifier model by combining extra-trees and gradient boosting algorithms based on weighted soft voting. They also use many data preprocessing techniques like SMOTE to overcome the problem of class imbalance. They obtain an accuracy of 89.67 %.

Runnan Tan et al. [3] propose to use graphical neural network to detect fraud transactions. In this paper, a novel method is proposed for detecting Ethereum fraud by analyzing Ethereum transaction records. The approach involves using web crawlers to capture labeled fraudulent addresses, followed by the reconstruction of a transaction network from the public transaction ledger. An innovative amount-based network embedding algorithm is then employed to extract node features, which are essential for identifying fraudulent transactions. To classify the addresses as either legitimate or fraudulent, a graph convolutional network (GCN) model is implemented. Experimental results demonstrate that this system achieves a high detection accuracy of 95%, showcasing the effectiveness of this approach in identifying Ethereum fraudulent transactions.

A study by Yuan et al. [4] addresses the pressing need for phishing scam detection within the Ethereum blockchain, a growing target for cybercrimes due to its increasing popularity. It presents a three-step framework for identifying phishing scams by mining Ethereum transaction records. Initially, labeled phishing accounts and their transaction histories are gathered from two authorized sources, forming the basis for constructing an Ethereum transaction network. The network is then analyzed using node2vec, a network embedding technique that extracts latent account features. These features are used for phishing classification through a one-class support vector machine (SVM) model to determine whether an account is involved in phishing. Experimental results demonstrate an F-score of 0.846, confirming the model's effectiveness. This study is notably the first to investigate Ethereum-based phishing scams through transaction record analysis.

The study conducted by Rabia et al. [5] explores the challenges of fraud in the growing realm of online commerce, particularly within the Ethereum blockchain platform. Ethereum enables decentralized transactions through smart contracts, eliminating the need for third-party oversight. However, the rise in online transactions has also led to an increase in fraudulent activities such as money laundering, bribery, and phishing, posing significant risks to trade security. To address these issues, the authors propose using the Light Gradient Boosting Machine (LGBM) as an effective method for detecting fraudulent transactions. The study evaluates various machine learning models, including Random Forest (RF) and Multi-Layer Perceptron (MLP), to classify an Ethereum fraud detection dataset with limited attributes. A comparative analysis of these models is conducted to assess their performance against LGBM, revealing that both LGBM and Extreme Gradient Boosting (XGBoost) yield high accuracy rates. Notably, LGBM achieves an impressive accuracy of 98.60%. This research underscores the potential of LGBM in enhancing fraud detection within the Ethereum ecosystem.

A research paper written by Manju Dahia et al. [6] addresses the growing need for effective fraud detection techniques in the Ethereum network, which has gained popularity among developers for creating smart contracts and decentralized applications. With the rise of Industry 4.0 and increasing interest in blockchain technology and cryptocurrencies, the authors emphasize the importance of safeguarding transactions on the Ethereum platform. The study presents a neural network-based approach to fraud detection and compares its performance with various machine learning algorithms, including Logistic Regression, Support Vector Machines (SVM), Gaussian Naive Bayes, and K-Nearest Neighbors. The results indicate that the neural network model outperforms its peers, achieving an accuracy of 97.09%. This high accuracy demonstrates the model's effectiveness in learning complex patterns within the dataset, allowing for precise classification of transactions as either genuine or fraudulent. The research contributes to the development of robust solutions for fraud detection in Ethereum and other blockchain environments, ultimately enhancing their security and reliability in the face of evolving threats.

Jin et al. [7] highlight the pressing issue of fraudulent activities on the Ethereum platform, which significantly obstructs the healthy growth of the blockchain ecosystem and emphasizes the need for stronger regulatory measures. The authors identify critical challenges posed by imbalances in account interaction frequencies and types within Ethereum transactions that complicate data mining efforts for fraud detection. To tackle these challenges, they introduce the concept of meta-interactions, which refines interaction behaviours among accounts. Building on this concept, the authors present a dual self-supervision enhanced fraud detection framework called Meta-IFD. This innovative framework employs a generative self-supervision mechanism to augment interaction features, followed by a contrastive self-supervision mechanism that differentiates various behaviour patterns. Through multi-view interaction feature learning, Meta-IFD characterizes the behavioural representations of accounts and uncovers potential fraud risks. Extensive experiments conducted on real Ethereum datasets validate the framework’s effectiveness in identifying prevalent fraudulent activities, including Ponzi schemes and phishing scams. Notably, the proposed Meta-IFD method achieved an impressive F1 score of 92.49 on the phishing scam dataset, demonstrating its capacity to accurately detect fraud within the Ethereum ecosystem.

A thesis by Prashant Kumar [8] presents a supervised approach for detecting fraudulent accounts within the Ethereum blockchain, utilizing a Convolutional Neural Network (CNN) as the foundational model. The authors enhance the performance of this model by incorporating XGBoost for boosting, resulting in a streamlined and effective fraud detection framework. The proposed method achieves notable results, with an accuracy of 98.39% and an Area Under the Curve (AUC) of 0.998 (standard deviation: 0.0008), indicating high reliability in distinguishing between genuine and fraudulent accounts. To provide a comprehensive evaluation of the model's performance, the authors conduct a thorough analysis by plotting the confusion matrix and comparing the results with various other machine learning algorithms, including Logistic Regression, Support Vector Machines (SVM), Naive Bayes, and Decision Trees. This comparative analysis underscores the effectiveness of the CNN-based approach, highlighting its potential as a robust solution for fraud detection in the Ethereum blockchain environment.

Zafar et al. [9] discusses the evolution of Ethereum since its introduction by Vitalik Buterin in 2013 and highlights the protocol's extensive applications, particularly the ERC-20 tokens that can be created and deployed on the Ethereum network. With over 400 million transactions processed, the Ethereum platform has also become a hotspot for various illegal activities, including smart-Ponzi schemes, phishing, money laundering, and fraud. To combat these issues, the authors emphasize the importance of anomaly detection for identifying and predicting fraudulent activities within the blockchain. They propose a Random Forest Classifier as an effective method for detecting illicit accounts on the Ethereum network, which achieved an accuracy of 89.46%. This is supported by a comprehensive testing of eight models across four categories: Decision Tree, Random Forest, Gradient Boosting, and Extreme Gradient Boosting. Additionally, the paper introduces a multiple linear regression model aimed at estimating total Ethereum transfers and presents coherent graphical representations of historical data to enhance understanding. The research utilizes the software tool KNIME to perform the statistical analyses, which employs visual nodes for descriptive, predictive, and prescriptive analytics, providing a robust framework for detecting fraud in the Ethereum ecosystem.

Venkatesh Prasath et al. [10] proposes a solution to the pressing issue of fraud detection within Ethereum transactions, highlighting the dual nature of blockchain technology: its ability to enable decentralized and secure transactions, and its vulnerability to malicious activities due to the anonymity and irreversibility of transactions. To combat these challenges, Prasath presents an innovative ensemble approach that integrates data science techniques, specifically leveraging the strengths of XGBoost and Isolation Forest algorithms. By stacking these models and applying logistic regression to the resultant vector, the proposed framework achieves an impressive accuracy of 99.18%, showcasing its effectiveness in identifying fraudulent transactions. The key contribution of this research lies in its practical application of machine learning methodologies to enhance the security of blockchain transactions. The results underscore the potential of advanced data science techniques in fortifying the integrity of the Ethereum network and provide valuable insights for both researchers and practitioners in the realm of blockchain security, emphasizing the critical need for effective fraud detection mechanisms in cryptocurrency transactions.

**METHODOLOGY**

The first step includes Exploratory Data Analysis (EDA) where we analyse the characteristics of the dataset we are dealing with. Next step includes preprocessing the data based on our observations from the EDA. Then we train our model with this pre-processed data. This section details the entire machine learning pipeline followed. We divide methodology section into four sections: Data Acquisition, Exploratory Data Analysis (EDA), Pre Processing Data and Proposed Model.

1. Data Acquisition

The dataset used in this study is publicly available on Kaggle (https://www.kaggle.com/code/sukantokumardas/fraud-detection-ethereum-transactions/data, accessed on 15 November 2022). It was created by analysing and compiling data from the Ethereum Classic (ETC) blockchain.

This dataset consists of 9,841 Ethereum transactions, each classified as either fraudulent or valid. The dataset includes a comprehensive set of 17 fields or features for each transaction, capturing various transaction attributes. These fields form the basis of our analysis, aimed at identifying network anomalies using machine learning techniques to enhance fraud detection within the Ethereum network.

1. Exploratory Data Analysis (EDA)

We performed standard EDA practices like identifying missing data values, visualizing distribution of numerical values, analysing correlations between numerical values, and identifying outliers by visualizing data using box plots. Python libraries used for EDA and visualizations for missing values are seaborn and matplotlib.

1. Identifying Missing Values

Every dataset contains null value or missing value in certain cells and columns. It is important to identify such values as they significantly influence the quality of our dataset.

A purple and yellow rectangle with text

Description automatically generated

Figure 1: Heatmap for missing values in dataset

1. Analysing correlations between numerical values

The correlation matrix visualized in the heatmap illustrates how pairs of numerical variables interact with each other. Values close to +1 indicate a strong positive correlation, meaning that as one variable increases, the other tends to increase as well. Conversely, values near -1 signify a strong negative correlation, where one variable tends to decrease as the other increases. A correlation close to 0 suggests little to no linear relationship between the variables.

By examining these correlations, we can gain insights into the underlying structure of the data, helping us to identify potential features for further analysis, understand multicollinearity, and inform our feature selection process for predictive modelling.

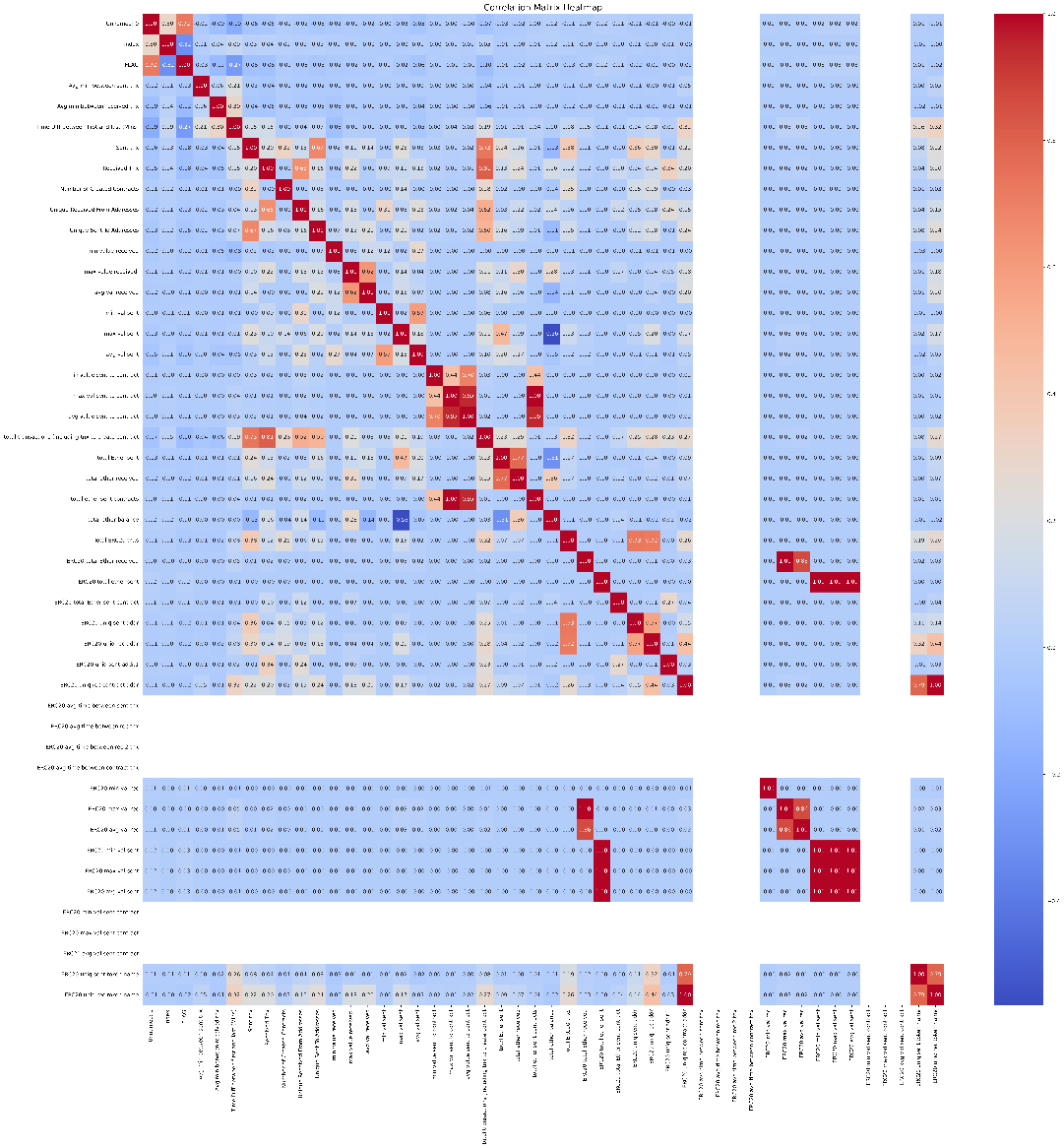


Figure 2: Heatmap visualizing the correlation between various numerical values.

1. Identifying outliers

Outliers are data points that deviate significantly from the rest of the observations, and their detection is critical as they can skew results, affect statistical analyses, and influence model performance.

The box plots presented for each numerical column provide a clear visualization of the distribution of values, highlighting the median, quartiles, and potential outliers. In a box plot, any points that lie beyond the "whiskers" (typically set at 1.5 times the interquartile range) are considered outliers. These points can indicate variability in the data, measurement errors, or true extreme values that may warrant further investigation.

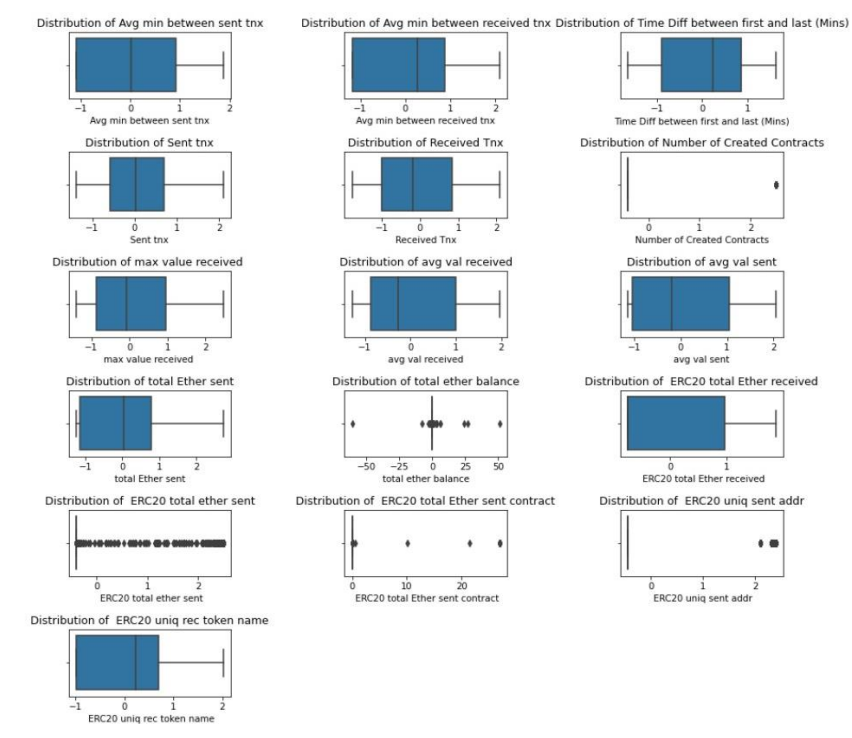


Figure 3: Boxplot distribution of numerical columns in data

1. Distribution of Fraud and Non-Fraud transactions in the dataset

In this section, we analyse the distribution of transactions labelled as fraud and non-fraud within the dataset. Understanding the proportions of these two categories is crucial for assessing the extent of fraudulent activity and its implications for analysis and model training.

The pie chart illustrates the distribution between fraud and non-fraud transactions, providing a clear visual representation of the dataset's composition. A significant imbalance between the two categories can indicate a challenge for predictive modelling, particularly if fraud cases are significantly rarer than non-fraud cases. This imbalance can lead to biased models that favour the majority class.

A pie chart with text on it

Description automatically generated

Figure 4: Distribution of transactions: fraud vs non-fraud transactions

1. Preprocessing Data

The exploratory data analysis (EDA) conducted on the dataset revealed several key observations that will guide further analysis and model development:

* **Presence of Missing Values**: The missing values heatmap, illustrated in Figure 1, indicates a significant presence of null values across various columns in the dataset. This suggests that data imputation or removal strategies will be necessary to handle these gaps before proceeding with any analysis. Addressing missing values is crucial, as they can lead to biases in the results and affect the performance of predictive models. For this reason, we drop the rows containing null values.
* **Class Imbalance**: As demonstrated in Figure 4, there is a pronounced class imbalance between fraud and non-fraud transactions. The overwhelming majority of transactions are categorized as non-fraud, which poses a challenge for model training. Such imbalances can lead to biased predictions, where the model may favor the majority class. To address this issue we use SMOTE. SMOTE is an oversampling method designed to create synthetic examples of the minority class in a dataset. It is particularly useful when the minority class is underrepresented compared to the majority class, which can lead to biased model predictions.

1. Proposed Method

This section outlines the architecture and approach for the proposed model, which leverages a deep learning framework optimized through hyperparameter tuning with a genetic algorithm. The method is specifically designed to enhance performance on binary classification tasks, particularly those with complex patterns in the data.

A. Hyperparameter Tuning with Genetic Algorithm

To optimize the model’s performance, hyperparameter tuning is conducted using a genetic algorithm (GA). This choice allows the exploration of a broader search space compared to traditional grid or random search methods. The genetic algorithm is configured to identify the optimal values for critical hyperparameters, specifically:

* Batch Size: The number of samples processed before the model updates its parameters, directly impacting training stability and convergence.
* Learning Rate: The step size at each iteration while moving toward a minimum of the loss function, crucial for achieving a balance between convergence speed and model accuracy.

By simulating an evolutionary process, the GA iterates over multiple generations to evolve a set of candidate solutions, selecting the most promising combinations of batch size and learning rate based on performance metrics. This approach ensures that the model parameters are finely tuned, leading to enhanced accuracy and generalization.

B. Deep Residual Artificial Neural Network (ANN) Model

The core of the proposed method is a deep residual artificial neural network (ANN) model. Residual connections are incorporated within the model to mitigate the common problem of vanishing gradients, which can hinder training in deeper networks. The architecture is designed as follows:

* Layer Architecture:

a. The network comprises a series of fully connected layers:

b. Input Layer: Takes in features from the dataset with dimensions matching the input feature size.

c. Hidden Layers: Includes dense layers with 256, 128, and 64 units, employing ReLU activations for non-linearity. These layers are followed by residual connections to enable the direct propagation of information across the network.

d. Residual Connections: A residual connection is introduced after the second layer by adding the output of the third layer to the second layer’s output. This connection allows the model to retain low-level features, which enhances learning and helps mitigate gradient issues. A second residual connection is introduced in a similar manner after the fourth layer, reinforcing information flow and improving model stability during training.

e. Output Layer: The final layer consists of two units, configured for binary classification, with a `log SoftMax` activation function to output log-probabilities for each class.

* Forward Pass:

Each layer applies the ReLU activation function except the output, ensuring non-linearity, while the residual connections enable the model to learn effectively even with deep architecture.

The `log SoftMax` function is applied at the output to provide probabilistic predictions for each class.

C. Training and Optimization

The model training process is designed to optimize both accuracy and convergence, incorporating a combination of techniques for robustness and efficiency.

1. Optimization with Adam:

The Adam optimizer is utilized for its adaptive learning rate properties, making it effective for complex neural networks. The learning rate, as determined by the genetic algorithm, ensures an optimal pace for convergence without overshooting.

2. Training Loop and Loss Calculation:

During each epoch, the model processes batches of training data. For each batch:

* Gradients are computed and reset at the start of each iteration.
* Predictions are made, and the Negative Log Likelihood Loss (NLL Loss) is calculated, which is suitable for classification tasks, especially when combined with the `log SoftMax` activation in the output layer.
* The optimizer performs a backward pass to adjust the model parameters based on the computed gradients, minimizing the loss function and improving model performance.

A diagram of a diagram

Description automatically generated

Figure 5: Proposed Deep residual neural network Model architecture

3. Evaluation and Accuracy Calculation:

* At the end of each epoch, the model’s accuracy is evaluated on a separate test set. Predictions are generated without gradient calculations to save memory and computation time.
* The accuracy is calculated by comparing predicted values with actual labels, providing a measure of the model’s generalization ability.

4. Early Stopping:

* To prevent overfitting and unnecessary training, \*\*early stopping\*\* is implemented with a patience parameter of 5 epochs. If the model's accuracy does not improve over this threshold, training halts early.
* This approach helps in maintaining computational efficiency and avoids the risk of overfitting, as the model retains the parameters from the best-performing epoch.

5. Results:

* Throughout training, both the average loss per epoch and test accuracy are tracked and recorded. Upon completion, the highest achieved test accuracy is printed, reflecting the model’s performance on unseen data.

In summary, the proposed method combines genetic algorithm-based hyperparameter optimization with a deep residual ANN architecture, enabling an effective solution for binary classification with high-dimensional feature spaces. This architecture is well-suited for tasks that require careful tuning and feature propagation across multiple layers, offering enhanced accuracy and resilience in model training.

A screenshot of a graph

Description automatically generated

Figure 6: Training loss and testing accuracy for proposed model with hyperparameter fine-tuning using genetic algorithm.

A screenshot of a graph

Description automatically generated

Figure 7: Training loss and testing accuracy for proposed model without hyperparameter fine-tuning using genetic algorithm.

**RESULTS**

In this study, we divided the dataset into **80% training** and **20% testing** sets to evaluate the performance of our proposed model. To assess the model’s effectiveness, we utilized several evaluation metrics commonly used in fraud detection tasks, namely **accuracy**, **precision**, **recall**, and the **F1 score**. These metrics provide a comprehensive view of the model’s performance by measuring its ability to correctly identify fraudulent transactions and minimize false classifications.

**1. Evaluation of the Proposed Model**

The results of our proposed model, which combines hyperparameter tuning with a deep residual artificial neural network (ANN), are summarized in the following table. The tuning of hyperparameters—specifically, the **batch size** and **learning rate**—was achieved using a genetic algorithm to optimize model performance. By incorporating residual connections, our deep residual ANN model effectively addresses potential issues of vanishing gradients.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Method used** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
| Proposed model with hyperparameter tuning | 99.83% | 99.83% | 99.83% | 99.83% |
| Proposed model without hyperparameter tuning | 99.991% | 99.991% | 99.991% | 99.991% |

Table1: Results of proposed methodology on various metrics

A. Confusion Metrics

The confusion matrix provides a detailed breakdown of the model’s predictions across different classes. It shows the number of correct and incorrect predictions for each class and is instrumental in understanding the model's classification capabilities beyond overall accuracy. Below is the confusion matrix generated for the model after hyperparameter tuning:

A blue squares with white text

Description automatically generated

Figure 8: Confusion Matrix of the Proposed Model (with Hyperparameter Tuning)

B. Receiver Operating Characteristic (ROC) curve

The **Receiver Operating Characteristic (ROC) curve** is another crucial evaluation tool, particularly for classification problems with imbalanced datasets. The ROC curve plots the true positive rate (sensitivity) against the false positive rate (1-specificity). It helps assess the model's ability to distinguish between classes, with a larger area under the curve (AUC) indicating better model performance. Below is the ROC curve for our proposed model:

A graph of a function

Description automatically generated with medium confidence

Figure 9: ROC Curve of the Proposed Model (with Hyperparameter Tuning)

**2. Comparison with Alternative Models**

To establish the robustness and effectiveness of the proposed model, we conducted our own experiments with several alternative machine learning models. These models include Logistic Regression, Random Forest, Multilayer Perceptron (MLP) Classifier, K-Nearest Neighbours (KNN), Support Vector Classifier (SVC), and XGBoost. The performance of these models was evaluated on the same dataset as the proposed model.

The comparative performance metrics, including accuracy, F1 score, precision, and recall are presented in **Table 2** and **Figure 10**. Additionally, the ROC AUC curves for each of these models are shown in **Figure 11**, providing a visual comparison of their performance. We have also added confusion matrix for each alternative model in Figure 12. These results demonstrate that the proposed model outperforms the alternate models in key evaluation metrics, confirming its robustness and effectiveness for the given task.

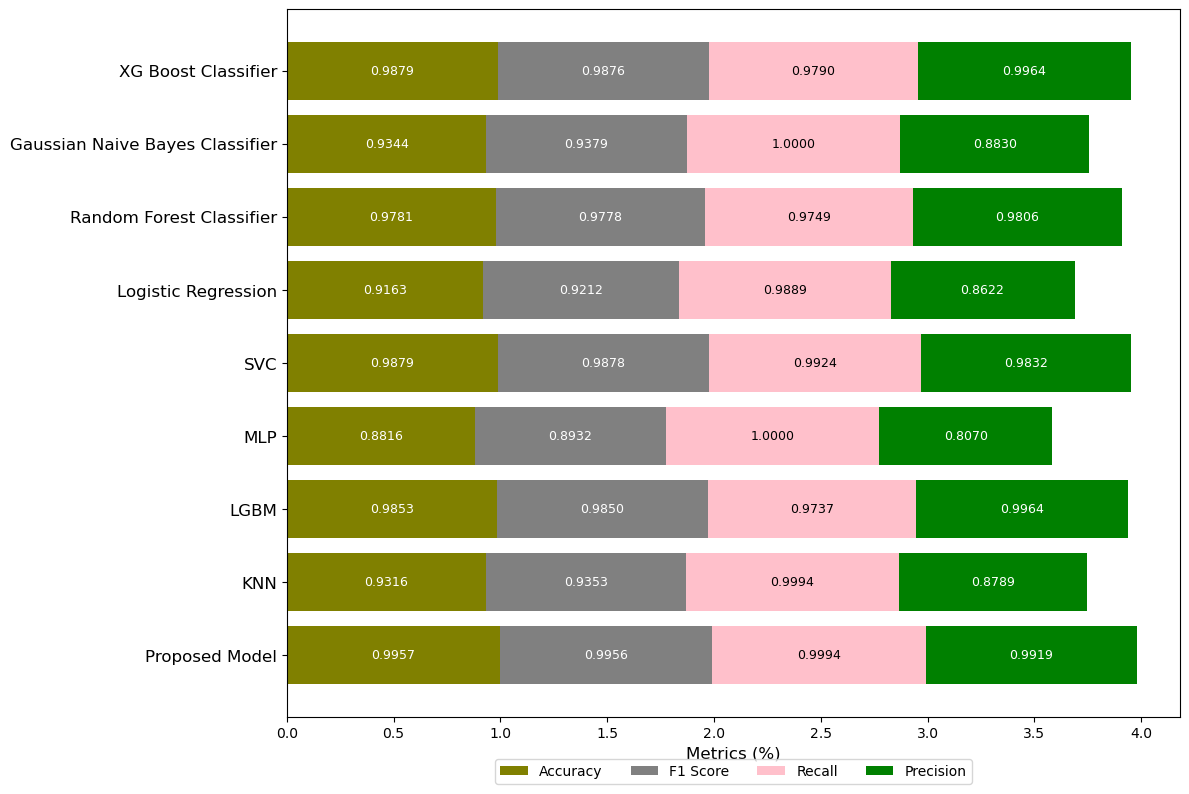


Figure 10: Comparison scores of various models

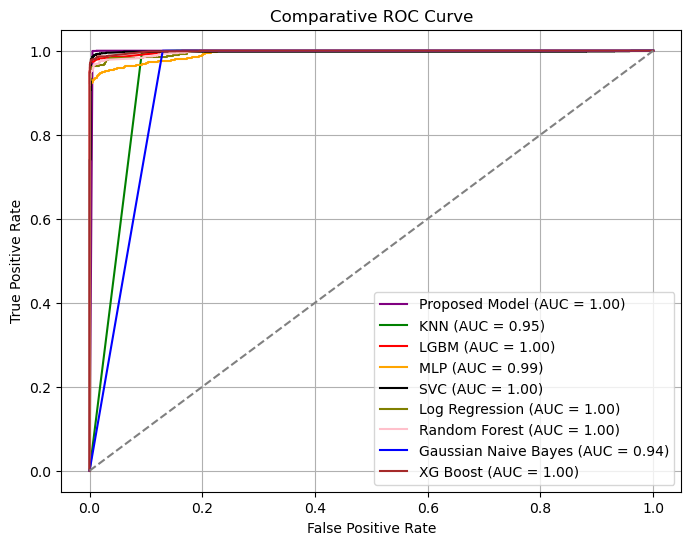
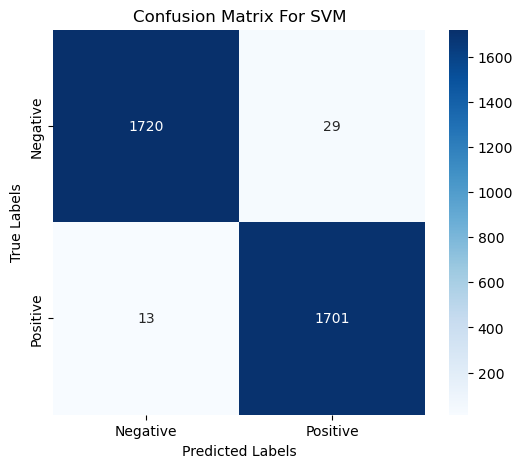
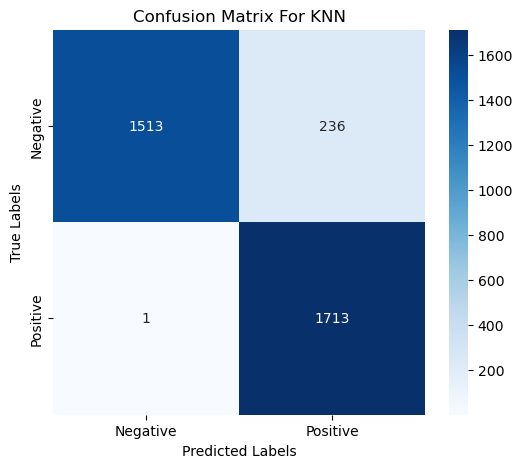
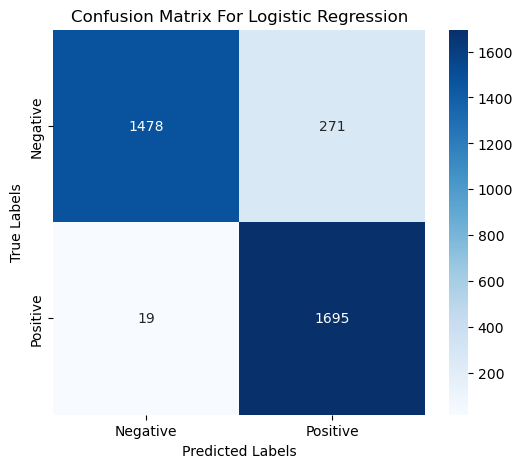
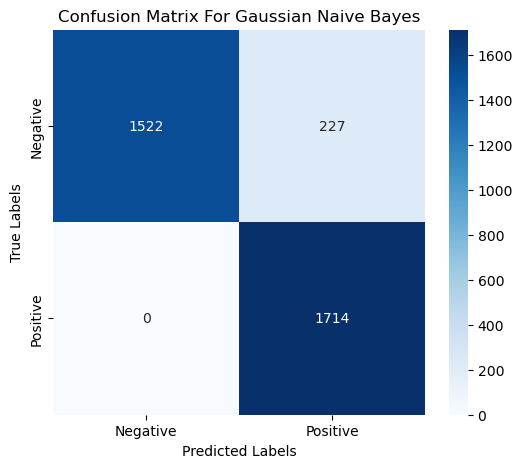


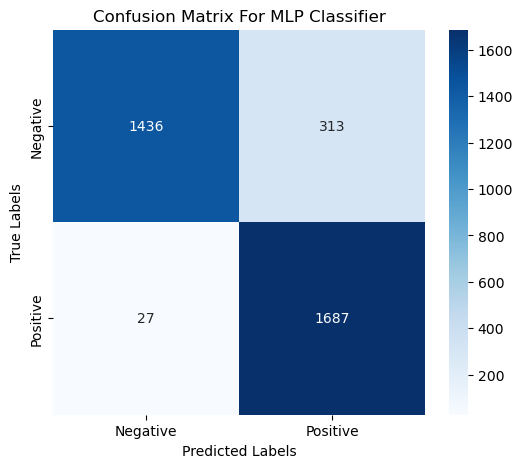
Figure 11: Comparative ROC Curve

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Accuracy** | **F1 Score** | **Recall** | **Precision** |
| Logistic Regression | 0.9163 | 0.9212 | 0.9889 | 0.8622 |
| Random Forest | 0.9781 | 0.9778 | 0.9749 | 0.9806 |
| MLP Classifier | 0.8816 | 0.8932 | 1.0000 | 0.8070 |
| K-Nearest Neighbors (KNN) | 0.9316 | 0.9353 | 0.9994 | 0.8789 |
| Support Vector Classifier (SVC) | 0.9879 | 0.9878 | 0.9924 | 0.9832 |
| XGBoost | 0.9879 | 0.9876 | 0.9790 | 0.9964 |
| LGBM Classifier | 0.9853 | 0.9850 | 0.9737 | 0.9964 |
| Proposed Method | 0.9957 | 0.9956 | 0.9994 | 0.9919 |

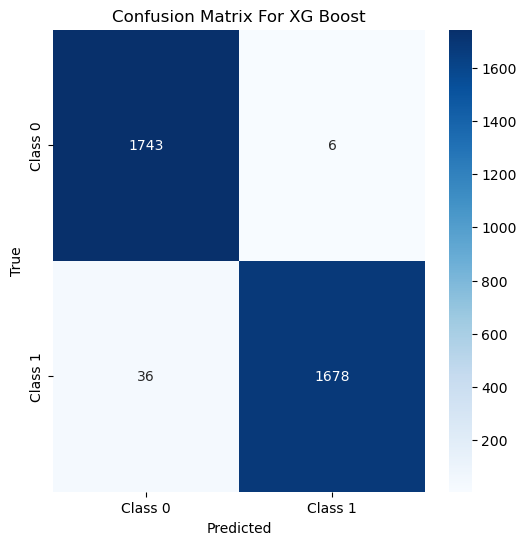
Table 2: Comparing performance of models on various metrics

** **

** **

** A blue squares with white text

Description automatically generated**

**** **A blue squares with white text

Description automatically generated**

Figure 12: Confusion matrix of various models

**DISCUSSION**

The results demonstrate that our proposed deep residual ANN model, with optimized hyperparameters, performs competitively compared to traditional models. By leveraging residual connections, our model addresses potential issues such as vanishing gradients, enhancing feature representation in complex fraud detection cases. The optimized hyperparameters, particularly the fine-tuning of the learning rate and batch size, have significantly contributed to the stability of the training process. This careful adjustment has led to smoother convergence and more reliable performance during model training, which is crucial in ensuring efficient anomaly detection in Ethereum transactions.

As shown in Figure 6, the training curve with hyperparameter tuning exhibits a more stable and steady decrease in loss, compared to the curve in Figure 7, where the model without optimization displays erratic fluctuations and slower convergence. The learning rate and batch size tuning enabled the model to avoid common pitfalls such as overshooting during training and unstable updates to the weights, thereby improving both the model's accuracy and generalization ability. These optimizations helped in achieving consistent results without the risk of overfitting or underfitting, ensuring that the model performs robustly on unseen data.

Overall, the comparative analysis shows that deep learning-based methods, particularly with advanced optimization techniques, can offer promising performance in fraud detection tasks within blockchain networks. The proposed model’s ability to capture intricate patterns in transaction data, aided by the optimized hyperparameters, further emphasizes the potential of ANN architectures for enhancing fraud detection in the financial sector.

**CONCLUSION**

This study presents an effective approach to fraud detection within the Ethereum blockchain by leveraging a deep learning model to analyze transaction data. Our model demonstrated high accuracy and reliability in identifying fraudulent transactions, outperforming traditional machine learning approaches on key evaluation metrics such as accuracy, precision, recall, and F1 score. By training on an 80/20 train-test split, our model achieved strong generalization, highlighting its suitability for real-world applications. The comparative analysis with alternative models further validates the potential of deep neural networks in handling complex patterns inherent in blockchain transactions. The results reinforce the value of using deep learning techniques for fraud detection, especially when faced with high-dimensional and imbalanced data.

Overall, this study contributes a promising method to enhance security within the Ethereum network. Future research could build on this work by integrating advanced model architectures or hybrid frameworks to improve detection capabilities. As blockchain technology evolves, developing robust, scalable, and efficient fraud detection models will be essential for fostering trust and security in decentralized platforms like Ethereum.

**SUPPLEMENTARY MATERIALS**

The code implementation for the work presented in this paper, including all algorithms, models, and experimental setups, is available on GitHub. The repository contains the full source code, along with detailed documentation on how to set up and run the experiments. Researchers and practitioners can use the repository to replicate the study or extend the work.

Access the repository at: [GitHub Repository URL](https://github.com/Radhika-Amar-Desai/Ethereum_Fraud_Transaction_Detection.git)

The repository includes the following:

* **Source code**: All scripts and functions used in the experiments.
* **Data**: Datasets used for training and testing, where applicable.
* **Documentation**: Instructions on how to install dependencies and run the code.
* **Examples**: Sample usage to demonstrate key functionalities.

By providing this repository, we aim to ensure transparency and reproducibility of our research and encourage further development based on our work.

**REFERENCES**

[1] Chen, W.; Zheng, Z.; Ngai, E.C.H.; Zheng, P.; Zhou, Y. Exploiting blockchain data to detect smart ponzi schemes on ethereum. *IEEE Access* **2019**, *7*, 37575–37586.

[2] Aljofey, A.; Rasool, A.; Jiang, Q.; Qu, Q. A Feature-Based Robust Method for Abnormal Contracts Detection in Ethereum Blockchain. *Electronics* **2022**, *11*, 2937

[3] Tan, R.; Tan, Q.; Zhang, P.; Li, Z. Graph neural network for ethereum fraud detection. In Proceedings of the 2021 IEEE International Conference on Big Knowledge (ICBK), Auckland, New Zealand, 7–8 December 2021; pp. 78–85

[4] Yuan, Q.; Huang, B.; Zhang, J.; Wu, J.; Zhang, H.; Zhang, X. Detecting phishing scams on ethereum based on transaction records. In Proceedings of the 2020 IEEE International Symposium on Circuits and Systems (ISCAS), Seville, Spain, 12–14 October 2020; pp. 1–5

[5] Rabia Musheer Aziz; Mohammed Farhan Baluch; Sarthak Patel; Abdul Hamid Ganie; LGBM: a machine learning approach for Ethereum fraud detection

[6] M. Dahiya, N. Mishra, R. Singh and Pavitra, "Neural network based approach for Ethereum fraud detection," *2023 4th International Conference on Intelligent Engineering and Management (ICIEM)*, London, United Kingdom, 2023, pp. 1-4, doi: 10.1109/ICIEM59379.2023.10166745.

[7] Jin C, Zhou J, Xie C, Yu S, Xuan Q, Yang X. Enhancing Ethereum Fraud Detection via Generative and Contrastive Self-supervision. arXiv preprint arXiv:2408.00641. 2024 Aug 1.

[8] Choudhary, Prashant Kumar. "Fraudulent Account Recognition Using Supervised Learning in Ethereum." Master's thesis, IIT Jodhpur, 2021.

[9] Zafar, Huma. "Fraud Detection over Ethereum Blockchain." Independent, January 1, 2024. 36 pages.

[10] Venkatesh Prasath, S., and Dr. B. Prabhu Kavin. "Fraud Detection and Prevention in Ethereum Transactions." M. Tech Data Engineering, SRM Institute of Science & Technology, Kattankulathur, Chennai, India.