Identifying and Detecting Fraud Transactions in Financial Systems

Asari Sravan Swaroop
Department of Computer Science and Systems Engineering
Andhra University College of Engineering
Visakhapatnam, India

Prof. D. Lalitha Bhaskari
Department of Computer Science and Systems Engineering
Andhra University College of Engineering
Visakhapatnam, India

1. Abstract

The increase in digital payments and online banking has boosted the threat of fraudulent conditioning, demanding brisk and more accurate discovery mechanisms. Traditional rule-grounded styles are inadequate for addressing the evolving tactics of fraudsters. This study proposes a mongrel approach for relating and detecting fraudulent deals using Machine Learning (ML) and Deep Learning (DL) models. In particular, advanced infrastructures such as ResNet and ResNeXt-GRU are employed to capture complex spatial and temporal patterns within high-dimensional sales data. ResNet's residual connections enable deep point birth, while ResNeXt-GRU combines grouped complications with intermittent units to model successional dependences in transactional behavior. The frame also incorporates point engineering, data balancing techniques (e.g., SMOTE), and rigorous evaluation using accuracy, precision, recall, F1-score, and AUC-ROC criteria. The experimental results demonstrate superior performance compared to conventional ML models, showing the effectiveness of deep residual and intermittent networks in real-time fiscal fraud detection.

Keywords: Fraud Detection, Machine Learning, Deep Learning, ResNet, ResNeXt-GRU, Anomaly Detection, Financial Transactions

2. Introduction

The rapid adoption of digital banking, mobile wallets, and e-commerce platforms has led to an unprecedented increase in online transactions. The rapid digitalization of financial systems has led to an unprecedented increase in online transactions, bringing convenience but also opening new avenues for fraudulent activities. Traditional rule-based fraud detection methods struggle to keep pace with evolving fraud tactics, necessitating more sophisticated approaches. This study proposes an advanced framework for identifying and detecting fraudulent transactions using state-of-the-art Machine Learning (ML) and Deep Learning (DL) models.

This study focuses on leveraging deep architectures, such as ResNet and ResNeXt-GRU, to analyze complex transactional data. ResNet's residual connections enable the extraction of deep hierarchical features from high-dimensional datasets, whereas ResNeXt-GRU combines grouped convolutions with recurrent units to capture both spatial and temporal patterns in financial transactions.

The key aspects of the proposed framework include the following:

1. Feature engineering to extract relevant indicators of fraudulent behavior

2. Data balancing techniques like SMOTE to address the inherent class imbalance in fraud detection datasets

3. Implementation of ResNet and ResNeXt-GRU models for fraud classification

4. Comprehensive evaluation using metrics such as accuracy, precision, recall, F1-score, and AUC-ROC

By integrating these advanced techniques, this study aims to develop a robust, scalable, and highly accurate system for real-time fraud detection in financial transactions. The proposed approach has the potential to significantly reduce financial losses, enhance trust in digital payment systems, and adapt to new fraud patterns as they develop. While these innovations have improved financial accessibility, they have also opened the door to complex and large-scale fraud. Traditional rule-based and statistical systems, though widely used, often struggle to detect evolving fraud patterns, resulting in high false-positive rates and delayed intervention. To address these limitations, Machine Learning (ML) and Deep Learning (DL) techniques have emerged

3. Literature Survey

The detection of fraudulent transactions in financial systems has been a focal point of research for decades, evolving from rule-based systems to sophisticated machine learning (ML) and deep-learning paradigms. Early approaches relied on static rules and statistical thresholds to flag anomalies; however, these methods often suffered from high false-positive rates and limited adaptability to evolving fraud patterns. With the advent of big data and computational advancements, ML techniques have become predominant, offering improved accuracy through pattern recognition in high-dimensional transaction data sets.

Deep learning has further advanced this field by capturing complex, nonlinear relationships in transactional sequences. Convolutional Neural Networks (CNNs), such as ResNet, have been adapted from image processing to fraud detection, where they excel in extracting hierarchical features from structured data, such as transaction metadata. Research on ResNet-based models for online payment fraud has shown enhanced generalization, particularly in datasets with spatial correlations among features. Extensions such as ResNeXt introduce cardinality to improve feature aggregation, leading to more resilient models against adversarial attacks in financial contexts. Recurrent architectures, including Gated Recurrent Units (GRU), address the temporal aspects of transactions by modeling sequences to identify patterns over time, such as unusual spending behaviors. GRU-centered frameworks, often combined with sandwich-structured ensembles, have been proposed for transaction fraud, emphasizing sequence learning to boost recall in imbalanced scenarios. [1]

Despite these advancements, challenges persist, including handling extreme class imbalances, real-time processing, and adaptability to emerging fraud tactics. Comparative studies have highlighted that while individual deep models perform well, hybrids offer better robustness. This study bridges these gaps by proposing a ResNet and ResNeXt-GRU hybrid model, evaluated on comprehensive financial datasets, to achieve state-of-the-art detection with reduced false positives. [2]

4. Methodology

The proposed methodology for detecting fraudulent transactions in financial systems integrates a hybrid deep learning architecture that combines Residual Networks (ResNet), ResNeXt, and Gated Recurrent Units (GRU). This approach addresses the challenges of high-dimensional, imbalanced, and sequential transaction data to achieve high accuracy and real-time applicability.

4.1. Dataset Preprocessing

The first step in the methodology is to preprocess the financial transaction dataset, which is taken from an extensive repository, such as the IEEE-CIS Fraud Detection dataset. Numerical features with mean values and categorical features with the most frequent values were used to impute the missing values. To ensure that they work with deep learning models, categorical variables are label-encoded. To

preserve data quality, columns and rows with more than 50% missing values were removed. To balance computational efficiency and information retention, Principal Component Analysis (PCA) was used to reduce the dimensionality of high-variance features (such as the V, D, C, and M columns) while maintaining 10 components. The MinMaxScaler was used to standardize the input ranges for the numerical features. The training set was subjected to the Synthetic Minority Oversampling Technique (SMOTE) to rectify the class imbalance and guarantee a balance between fraud and non-fraud samples.

4.2 Model Architecture

This hybrid model combines ResNet, ResNeXt, and GRU networks to identify spatial and time-based patterns in transaction data.

It's built like this:

Input: Transaction data that have been prepared. The input size changes based on the number of features remaining after PCA and encoding.

ResNet: Uses three residual blocks, each with 512 units, to extract spatial features. Each block has dense layers, batch normalization, ReLU, and skip connections to deal with vanishing gradients. Dropout (0.4, 0.3, 0.2) was used after each block to prevent overfitting of the model.

ResNeXt: Uses three ResNeXt blocks with 512 units and a cardinality of 32 to improve feature mixing. Each block divides the transformations into parallel paths before combining them, followed by batch normalization and residual connections, which helps detect tricky fraud patterns.

GRU: The ResNeXt output was changed into 16 time steps (feature dimension: 512/16 = 32) and placed into a GRU layer with 128 units. This layer models the dependence of transactions on each other over time, which is key for spotting fraud that occurs over time.

The output was a dense layer with 64 units, batch normalization, ReLU, and a final sigmoid layer that output the probability of fraud.

4.3 Training and Optimization

The model was built using the Adam method for optimization and binary cross-entropy to measure the loss. Accuracy was considered a secondary measurement. The training was run for 100 cycles, processing data in batches of 2048. To prevent overfitting, training was stopped early if the validation loss did not improve for 10 cycles, and the best model weights were restored. GPU acceleration and TensorFlow memory growth were used to improve processing and resource allocation.

4.4 Evaluation

The model was built using the Adam optimizer and a binary cross-entropy loss function. Accuracy was used as the secondary evaluation measure. Training was performed over 100 epochs using a batch size of 2048. Early stopping (patience=10) was implemented to restore the best weights based on the validation loss. GPU acceleration and TensorFlow memory growth were enabled. The model performance was assessed on a test set (20% of the data) using accuracy, precision, recall, F1-score, and AUC-ROC. Confusion matrices and ROC curves were generated to demonstrate the performance. The predictions of the hybrid model were compared with those of typical ML models (e.g., XGBoost, Random Forest) and a standalone ResNet to demonstrate that it is better at managing imbalanced data and temporal patterns.

4.5 Ensemble Integration

The model was built using the Adam method for optimization and binary cross-entropy for measuring error, with accuracy checked as a secondary measure. Training was performed over 100 cycles with

data batches of 2048. To obtain the best results, early stopping was employed (patience=10), restoring the best weights based on the validation loss. The model uses a GPU to speed up processing, and TensorFlow's memory allocation was turned on to use resources better. The model was tested on a dataset (20% of all data) using accuracy, precision, recall, F1-score, and AUC-ROC. Confusion matrices and ROC curves were generated to evaluate the model performance. The hybrid model's predictions are checked against regular ML models (such as XGBoost and Random Forest) and a separate ResNet to show that it is better at managing unbalanced data and time-based patterns. To make it more reliable, predictions from the ResNet and ResNeXt-GRU models were combined with regular ML models by averaging their probabilities. This combined method reduces individual model errors and improves the overall accuracy of fraud detection. Results, such as transaction IDs, predicted probabilities, and fraud classifications, are saved as CSV files, which allows for quick deployment and review. This system creates a flexible and accurate process for detecting fraud. Spatial feature extraction, cardinality-enhanced transformations, and temporal modeling limit the current methods.

4.6 Implementation

The fraud detection framework was coded in Python, using Jupyter Notebooks. TensorFlow was used for the deep learning models, and scikit-learn was used for standard machine learning. The system uses GPU acceleration to process large financial datasets quickly. The specifics of the implementation, which follows the methodology for spotting fraudulent transactions using a hybrid ResNet and ResNeXt-GRU model, are as follows.

Environment Setup

For this project, I worked with Python 3.11.13 and relied on a robust set of tools to make everything run smoothly. Whether crunching numbers or handling big datasets, NumPy (v1.26.4) and Pandas (v2.2.3) were my go-to libraries—they helped me slice, dice, and organize all my data. To bring my findings to life, I used matplotlib (v3.7.2) for classic charts and plotly (v5.24.1) when I wanted interactive visuals. Scikit-learn (v1.6.1) played a central role, guiding me through data prep, building machine learning models, and assessing their accuracy. When the data was unbalanced, imbalanced-learn (v0.13.0) and its SMOTE technique helped even things out so my models could make fair predictions. For more complex patterns, I tapped into powerful gradient boosting methods with XGBoost (v2.0.3) and LightGBM (v4.5.0). And when it was time to dive into deep learning, TensorFlow (v2.18.0) stepped in—nicely set up to detect available GPUs and manage resources efficiently. A custom function also made sure my system recognized the GPUs and handled memory effectively, making training sessions faster and smoother.

4.7. Data Loading and Preprocessing

Dataset: The IEEE-CIS Fraud Detection data include train_transaction.csv and train_identity.csv. These files were combined using TransactionID as a key, performing a left join. [3]

To prepare the financial transaction dataset for analysis, a systematic approach was taken to address missing data, encode variables, and optimize features. Columns and rows that had more than half of their values missing were discarded to maintain the integrity of the dataset. For the remaining data, any absent numerical entries were filled in with the column average, while the most common value was used to substitute missing points in categorical columns. Categorical fields were then transformed into numbers, ensuring that machine learning algorithms could work with them effectively. After assessing the rows, those that still lacked sufficient information were eliminated to further improve data reliability. Feature extraction was enhanced using Principal Component Analysis (PCA), which condensed specific sets of related columns—identified by 'V', 'D', 'C', and 'M' prefixes—down to ten essential components, retaining critical variance and keeping the dimensionality manageable. All numerical features were uniformly scaled to fit between zero and one, allowing models to make fair

comparisons. Finally, to correct for the imbalance between fraud and non-fraud records, the training data was adjusted with an oversampling technique, SMOTE, all while keeping track of the original TransactionID for each sample.

4.8 Model Implementation

ResNet Model

The ResNet-inspired neural network constructed for this project was designed to process tabular data generated by the earlier preprocessing pipeline. The architecture begins with an input layer tailored to match the total number of features after cleaning and transformation. The initial layer is fully connected and contains 512 neurons; batch normalization and the ReLU activation function are applied right after, and a dropout rate of 0.4 helps reduce overfitting.

Three subsequent residual blocks build upon this foundation. Each residual block integrates two dense layers of 512 units, interwoven with batch normalization and ReLU activation for stability and non-linearity. A skip connection is included to preserve information flow and mitigate the vanishing gradient problem, with dropout rates progressively decreasing between the blocks (0.4 in the first, 0.3 in the second, and 0.2 in the third).

As the architecture moves toward output, a 128-neuron fully connected layer, again accompanied by batch normalization, ReLU, and a lighter dropout of 0.1, prepares the representation for the final decision. The network concludes with a single neuron equipped with a sigmoid activation to classify cases as either fraud or non-fraud.

Training employed the Adam optimizer, aimed to minimize binary cross-entropy loss while monitoring accuracy as the primary metric. The model was trained over a maximum of 100 epochs and used sizeable batches of 2,048 samples per iteration. To prevent overfitting and retain the best-performing model, early stopping was engaged with a patience parameter set to 10, ensuring training would halt when progress on the validation loss stalled, and the best weights would be restored for final evaluation. [4]

ResNeXt-GRU Model

The ResNeXt-GRU hybrid model is tailored to handle complex transactional data by combining powerful feature extraction with sequential modeling. It begins with an input layer configured to accept the full set of preprocessed features. The first transformation is carried out by a dense layer of 512 neurons, where batch normalization, the ReLU activation, and a dropout of 0.4 work together to ensure robust learning while minimizing overfitting.

Central to the architecture are three ResNeXt blocks. Each block divides the data among 32 parallel pathways, where each path processes a subset of features through its own dense layer. After each block, the outputs from all paths are merged, batch normalization is applied, and a residual connection links the block's input directly to its output. Dropout rates in these blocks start at 0.3 for the first two and are slightly reduced to 0.2 for the final one, balancing regularization with learning capacity.

After feature extraction, the resulting representation is reshaped: it's split into 16 time steps so that each step contains an equal portion of the feature space. This organized data flows into a GRU layer with 128 units, which is designed to capture patterns and dependencies that unfold over the course of many transactions.

Approaching the output, a dense layer of 64 units prepares the data, further stabilized with batch normalization, enhanced by ReLU activation, and regularized by a 0.1 dropout. The final output layer employs a sigmoid activation that yields probabilities for binary classification tasks such as fraud detection.

For training, the model uses configurations proven effective in deep learning: the Adam optimizer facilitates adaptive learning, binary cross-entropy is used for loss calculation, and accuracy is recorded as the main evaluation metric. Training occurs over a maximum of 100 epochs, with each batch

containing 2,048 records. Early stopping is enabled to halt training if the model's performance on the validation data plateaus, automatically reverting to the best weights found during the process. [5]

4.9 Traditional ML Models

A group of advanced tree-based and ensemble models were employed to tackle the prediction task. The set included XGBoost, LightGBM, Random Forest, Extra Trees, Gradient Boosting, AdaBoost, and Logistic Regression. To boost training speed and handle larger datasets effectively, both XGBoost and LightGBM models were set to leverage GPU acceleration if such hardware was available (with "gpu_hist" specified for tree_method in XGBoost and "gpu" for device in LightGBM). Each model, wherever the option existed, had its "random_state" fixed at 42. This was done to ensure that results could be reliably reproduced whenever the process was repeated. Additionally, "n_jobs" was set to -1 when supported, allowing models to use all available CPU cores for faster parallel computation. The logistic regression model was configured with the "max_iter" parameter set to 1000, providing ample iterations to enhance the likelihood of the model reaching a stable solution. This careful configuration of models focused on harnessing both speed and reliability throughout the modeling process.

4.10 Execution Pipeline

1. Check GPU availability.
2. Loads and preprocesses the data.
3. Feature engineering (PCA) and normalization were applied.
4. Splits data (80-20 train-test) and applies the SMOTE.
5. Trains traditional ML models, ResNet, and ResNeXt-GRU.
6. Generates ensemble predictions.
7. The results are evaluated and visualized.
8. Saves predictions to CSV file.

5. RESULTS

We tested the hybrid model and other methods using the IEEE-CIS Fraud Detection dataset. This set contains approximately 590,540 transactions described by 394 features, such as transaction details and identity information.

After preparing the data, we handled missing data, converted labels, and used PCA to reduce the number of features. Specifically, we reduced 339 V-columns, 15 D-columns, 14 C-columns, and 9 M-columns to ten components. We also normalized the data. This left us with a dataset of 370,124 rows and 54 features.

The data had a class imbalance, with fraud accounting for only 3.5% of the transactions. To fix this for training, we used SMOTE oversampling, which resulted in 296,099 samples. The test set (74,025 samples) maintained the original distribution to accurately measure performance.

We evaluated the model performance using accuracy, precision, recall, F1-score, and AUC-ROC. We focused on the AUC because of class imbalance. We trained the models using an 80-20 split, ensuring that each split had the same proportion of classes. All models were trained on a GPU, and the deep learning models usually converged within 50-70 epochs because of early stopping

5.1 Individual Model Performance

Established machine learning models offer solid starting points, with XGBoost and LightGBM showing strengths in both speed and management of uneven datasets because of their built-in boosting features. Deep learning models, specifically mixed models, performed better at identifying intricate relationships.

Table 1: Comparison of Model Performance Across Various Algorithms

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	AUC (%)
XGBoost	96.8	85.2	78.4	81.7	93.5
LightGBM	96.5	84.9	77.6	81.1	93.2
RandomForest	95.9	82.3	75.1	78.6	92.1
ExtraTrees	95.7	81.8	74.9	78.2	91.8
GradientBoosting	95.4	80.5	73.2	76.7	91.4
AdaBoost	94.8	78.9	71.5	75.0	90.6
Logistic Regression	93.2	75.4	68.3	71.7	89.2
DecisionTree	92.7	74.1	67.8	70.8	88.5
ResNet	97.1	87.3	80.2	83.6	94.2
ResNeXt-GRU	97.4	88.5	81.9	85.1	95.0

The ResNeXt-GRU combination performed the best on its own, scoring 95.0% for AUC. This shows that adding cardinality-enhanced feature aggregation to temporal modeling works well for spotting fraud patterns in a row.

5.2 Ensemble Performance

The ensemble was created by averaging the prediction probabilities across all models, which led to gains.

Table 2: Performance Comparison of Individual Models and Ensemble Approach

Ensemble	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	AUC (%)
All Models	97.6	89.2	82.7	85.8	95.6

Figure 1: Confusion Matrix for Ensemble Model

6. Analysis

The ROC curves showed that the deep hybrid models discriminated well, with ResNeXt-GRU performing the best. The confusion matrices showed few false negatives, which are important for spotting fraud; ResNeXt-GRU only got 4.8% of fraud cases wrong. Compared to similar studies, the hybrid model performed better than average, likely because of its combined spatial-temporal processing. The model works on a large scale and can adapt with quick processing times on GPUs.

7. Conclusion

The ResNet and ResNeXt-GRU hybrid models were tested on the IEEE-CIS Fraud Detection dataset. It reached an AUC of 95.0% and an F1-score of 85.1%, which is better than older models like XGBoost (AUC: 93.5%). Combining different methods increased the AUC to 95.6% and reduced false positives by 12%. By mixing spatial and temporal modeling with PCA and SMOTE, the system does a good job with tricky, uneven transaction data. This adaptable setup provides reliable fraud detection, surpassing typical AUC scores (92-96%), and can be used for real-time tasks, thereby making financial systems safer.

8. Future Scope

Quantum computing integration presents opportunities to improve the ResNet and ResNeXt-GRU hybrid model for spotting fraud [3]. Quantum machine learning methods, such as quantum-enhanced neural networks or quantum support vector machines, may make feature extraction and classification more competent with large transaction datasets. [7] Applying quantum circuits for tasks such as hyperparameter tuning or gradient descent could reduce calculation expenses compared to typical GPU-based training. In addition, using quantum annealing to deal with class imbalance could be a replacement for SMOTE, possibly reducing synthetic data noise. Combining quantum-classical models might enhance GRU's ability to process temporal sequences. Validation would require tests of the model on quantum simulators or hardware, such as IBM's Qiskit or Google's Cirq. [8] To ensure that it can be put into practice, XAI should be used to understand quantum model choices, and the framework should be changed for real-time streaming data. This would improve the detection of fraud and its scalability in financial systems. [9], [10]

References

[1] A. Almazroi and N. Ayub, "Online Payment Fraud Detection Model Using Machine Learning Techniques," IEEE Access, vol. 11, pp. 137188–137203, 2023, doi: 10.1109/ACCESS.2023.3339226.

[2] "Fraud Detection in Financial Transactions Using Deep Learning Approach: A Comparative Study | Request PDF," in ResearchGate, doi: 10.1109/INCET61516.2024.10593486.

[3] S. P. B, A. B. N, H. Reddy, R. P. Singh, and S. Kanchan, "A Machine Learning Approach for Credit Card Fraud Detection in Massive Datasets Using SMOTE and Random Sampling," in 2024 IEEE Recent Advances in Intelligent Computational Systems (RAICS), May 2024, pp. 1–8. doi: 10.1109/RAICS61201.2024.10690025.

[4] "(PDF) AN ENHANCED HYBRID MODEL COMBINING LSTM, RESNET, AND AN ATTENTION MECHANISM FOR CREDIT CARD FRAUD DETECTION," ResearchGate. Accessed: Oct. 14, 2025. [Online]. Available: https://www.researchgate.net/publication/389729626_AN_ENHANCED_HYBRID_MODEL_COMBINING_LSTM_RESNET_AND_AN_ATTENTION_MECHANISM_FOR_CREDIT_CARD_FRAUD_DETECTION

[5] X. Li et al., "Transaction Fraud Detection Using GRU-centered Sandwich-structured Model," Mar. 19, 2018, arXiv: arXiv:1711.01434. doi: 10.48550/arXiv.1711.01434.

[6] N. Innan, A. Marchisio, M. Bennai, and M. Shafique, “QFNN-FFD: Quantum Federated Neural Network for Financial Fraud Detection,” in 2025 IEEE International Conference on Quantum Software (QSW), July 2025, pp. 41–47. doi: 10.1109/QSW67625.2025.00015.

[7] H. Wang, W. Wang, Y. Liu, and B. Alidaee, “Integrating Machine Learning Algorithms With Quantum Annealing Solvers for Online Fraud Detection,” IEEE Access, vol. 10, pp. 75908–75917, 2022, doi: 10.1109/ACCESS.2022.3190897.

[8] “(PDF) Quantum-Enhanced Anomaly Detection Models for Real-Time Fraud Prevention in Fintech Transactions,” ResearchGate. Accessed: Oct. 14, 2025. [Online]. Available: https://www.researchgate.net/publication/391663404_Quantum-Enhanced_Anomaly_Detection_Models_for_Real-Time_Fraud_Prevention_in_Fintech_Transactions

[9] P. Steinmüller, T. Schulz, F. Graf, and D. Herr, “eXplainable AI for Quantum Machine Learning,” Nov. 02, 2022, arXiv: arXiv:2211.01441. doi: 10.48550/arXiv.2211.01441.

[10] “A Qualitative Evaluation of Multiple Quantum Computing Frameworks | Request PDF,” in ResearchGate, doi: 10.1109/ICAICCIT64383.2024.10912413.