Anomaly Detection for Credit Card Fraud Detection : A Deep Learning Approach

Karthvik Sarvade
New York University
ks6807@nyu.edu

Mannal Kamble
New York University
mk8475@nyu.edu

Abstract

This project aims to develop a deep learning model capable of detecting fraudulent transactions in credit card data. Inspired by "Locally Interpretable One-Class Anomaly Detection for Credit Card Fraud Detection" by Tung-Yu Wu and You-Ting Wang, the project faces the challenge of testing and enhancing the model's robustness against sophisticated, deliberately manipulated adversarial data sets. Additionally, it focuses on evaluating the model's performance in various anomaly detection scenarios to understand its adaptability and accuracy in different environments. The project utilizes a Credit Card Fraud Detection dataset, encompassing transactions made in September 2013 by European cardholders, to address the significant skewness challenge inherent in credit card data. The proposed model employs an AutoEncoder for input-output reconstruction and a fully-connected classifier for fraud detection, trained adversarially and unsupervised to handle class imbalances in fraud detection datasets. GitHub Repo

1 Introduction

In the rapidly evolving domain of digital finance, the escalating incidence of credit card fraud presents a formidable challenge, directly affecting consumers and financial institutions on a global scale. The proliferation of online financial transactions has led to a corresponding rise in fraudulent activities, highlighting the critical need for advanced fraud detection solutions. This project report introduces a deep learning-based model specifically designed to tackle this challenge, drawing inspiration from 'Locally Interpretable One-Class Anomaly Detection for Credit Card Fraud Detection' by Tung-Yu Wu and You-Ting Wang. The primary objective of our project is to develop and rigorously test a model that effectively identifies fraudulent transactions within credit card data, while also assessing its resilience against sophisticated, adversarial data manipulations aimed at evading detection.

The project utilizes the Credit Card Fraud Detection dataset from Kaggle, which contains detailed transactions from European cardholders during a two-day period in September 2013. This dataset is characterized by a significant class imbalance, a common hurdle in fraud detection, where genuine transactions greatly outnumber fraudulent ones. To address this, our model integrates an AutoEncoder for the precise reconstruction of legitimate transaction patterns and a fully-connected classifier capable of discerning anomalies in the heavily imbalanced datasets typical of credit card transactions.

This report meticulously outlines the development process of our deep learning model, including the experimental approach, setup, execution, and comprehensive analysis methods. Our goal is to validate the model's effectiveness not just against the Kaggle credit card fraud detection dataset but also across other anomaly detection scenarios, thereby confirming its robustness and practical applicability in diverse real-world contexts. Through this endeavor, we aim to contribute significantly to the field of financial cybersecurity, enhancing the detection and prevention of credit card fraud, and providing a robust foundation for future research and technological advancements.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

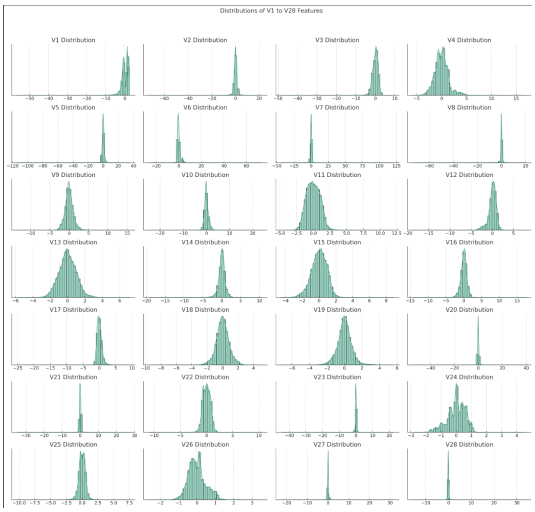

Figure 1: Distribution of features in the dataset

2 Literature Review

The application of deep learning in fraud detection has garnered significant attention in recent years. Abbas et al. (2021) and Thanh Thi Nguyen et al. (2020) emphasize the potential of deep learning techniques, particularly neural networks, in efficiently processing large datasets and uncovering complex patterns inherent in credit card transactions. These methods surpass traditional statistical approaches in detecting fraudulent activities due to their advanced pattern recognition and learning capabilities.

A major challenge in credit card fraud detection is the highly imbalanced nature of datasets, where legitimate transactions far outnumber fraudulent ones. This imbalance can lead to inaccuracies in detection. Research has focused on developing models that can effectively learn from such skewed data. Techniques such as oversampling of minority classes and synthetic data generation (like SMOTE) are explored to mitigate this issue. The aim is to enhance the model's sensitivity towards rare fraudulent transactions without compromising the overall accuracy.

Anomaly detection has proven to be a crucial strategy in fraud detection, as it focuses on identifying patterns that deviate from the norm. Traditional methods like clustering and nearest-neighbor techniques have evolved with the integration of deep learning. AutoEncoders, specifically, have shown promise in fraud detection, as indicated in studies by Pumsirirat and Yan (2018). They excel in reconstructing normal transaction patterns and identifying anomalies, making them particularly useful in handling imbalanced datasets.

AutoEncoders play a pivotal role in reconstructing and learning normal transaction behaviors, while simultaneously identifying deviations indicative of fraud. The use of adversarial networks, as explored in research like Ba (2019), adds another layer of sophistication. These networks, especially Generative Adversarial Networks (GANs), are utilized for data augmentation to balance class distributions, thereby improving the detection accuracy of fraud detection systems.

Adversarial attacks present a significant challenge in fraud detection, where attackers deliberately manipulate data to evade detection. These attacks are designed to exploit vulnerabilities in machine learning models, rendering them less effective. Recent research in this domain focuses on developing models resilient to such attacks. Techniques for enhancing model robustness include adversarial training, where models are trained with adversarially manipulated data to improve their resistance to such attacks, ensuring the reliability of fraud detection systems in real-world scenarios.

3 Model Architecture

3.1 AutoEncoder for Anomaly Detection:

The core of the model architecture is an AutoEncoder, a type of neural network that is adept at learning a compressed representation of the input data. In this project, the AutoEncoder is designed to learn and reconstruct normal transaction patterns. It comprises two main parts: the encoder, which compresses the input data into a lower-dimensional representation, and the decoder, which reconstructs the data back to its original form. The AutoEncoder is trained primarily on legitimate transactions to learn the standard patterns of normal transactions.

3.2 Classifier for Fraud Detection:

Alongside the AutoEncoder, a fully-connected neural network classifier is employed. This classifier is tasked with distinguishing between normal and fraudulent transactions. It analyzes the output of the AutoEncoder and classifies each transaction based on the degree of deviation from the learned normal patterns. The classifier operates on the principle that fraud transactions, which are not part of the training set for the AutoEncoder, will exhibit significant reconstruction errors, signaling potential fraud.

3.3 Adversarial Training:

To enhance the model's robustness against adversarial attacks, the training incorporates adversarial examples. This approach involves exposing the model to manipulated data during training, enabling it to learn and recognize various forms of data manipulation typically used in fraudulent activities. Adversarial training is critical in developing a model that remains effective in the ever-evolving landscape of credit card fraud.

3.4 Loss Functions and Optimization:

The model uses specific loss functions to optimize both the AutoEncoder and the classifier. The AutoEncoder employs a reconstruction loss (like Mean Squared Error) to minimize the difference between the original and reconstructed data. The classifier, on the other hand, utilizes a classification loss (like Binary Cross-Entropy) to accurately classify transactions. These loss functions are crucial in fine-tuning the model for high accuracy and reliability in fraud detection.

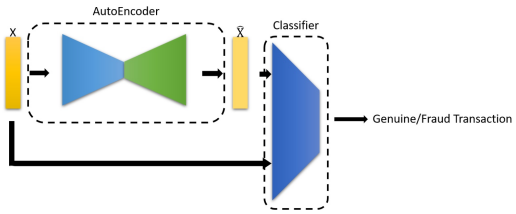

Figure 2: Model Architecture Diagram

4 Methodology

4.1 Dataset Preparation and Splitting:

4.1.1 Custom Dataset Class 'SplitedDataSet':

Purpose: To segregate credit card transactions into two categories: fraud and non-fraud.
Data Loading: Reads data from a CSV file, ensuring all transactions are loaded correctly.

Class Filtering: Depending on the mode ('non-fraud or 'fraud'), the dataset filters out the opposite class of transactions to create a focused dataset.
Data Conversion: Each transaction's features and labels are converted from string format to float for numerical processing.

4.1.2 Aggregated Dataset 'DataSet':

Function: To combine multiple subsets of data into a single dataset.
Data Aggregation: Appends features and labels from different datasets, ensuring that the combined dataset contains a balanced representation of both fraudulent and non-fraudu- lent transactions.
Torch Conversion: Features and labels are converted to FloatTensors for compatibility with PyTorch models.

4.1.3 Dataset Splitting 'getDatasets':

Data Splitting: Splits the non-fraudulent and fraudulent data into training and testing subsets.
Training and Testing Proportions: Determines the number of data points for each subset, ensuring a representative sample for both training and evaluation.
Random Splitting: Utilizes random split for a randomized division of data, enhancing the model's generalizability.

4.2 Neural Network Architecture:

4.2.1 Weight Initialization Function:

Goal: To optimize the initial weights of the neural network layers.
Layer-specific Initialization: Applies different initialization strategies for convolutional, linear, and BatchNorm1d layers to ensure efficient learning.

4.2.2 FCNN (Discriminator):

Architecture: Comprises linear layers, batch normalization, and ReLU activations in a sequential manner.
Role: To classify the input transactions into binary classes (fraudulent or non-fraudulent).

4.2.3 Autoencoder (Generator):

Encoder and Decoder: The encoder compresses the input data, and the decoder reconstructs it, aiming to capture and reproduce the underlying patterns of the transactions.
Purpose: To learn the normal pattern of transactions and assist in anomaly detection.

4.3 Model Training and Evaluation Setup

4.3.1 Setting Training Parameters:

Batch Size: Set to 4096, determining the number of samples processed in one iteration.
Learning Rate (lr): A value of 2e-4 is chosen, balancing the speed and stability of the training process.
Epochs: The number of complete passes through the entire dataset is set to 1 for initial training.
Normalization: Z-score normalization is used to standardize features, ensuring uniform data scaling.
Reconstruction Loss Type: Specified as 'SmoothL1', a combination of L1 and L2 losses, which is less sensitive to outliers.

4.3.2 Initialization of Models:

Autoencoder (Generator) and FCNN (Discriminator) are initialized and set to training mode, enabling specific behaviors like dropout during the training phase.

4.3.3 Defining Optimizers:

Adam Optimizers: Separate optimizers for the generator and discriminator, with weight decay set to 1e-4 to help prevent overfitting.

4.3.4 Loss Function Setup:

Reconstruction Loss: nn.SmoothL1Loss() is chosen for the generator, measuring the difference between reconstructed and original data.
Adversarial Loss Functions: Binary Cross-Entropy (nn.BCEWithLogitsLoss()) and Mean Squared Error (nn.MSELoss()) for the discriminator.

4.4 Training Loop

Iterating Over Epochs: The loop runs for a specified number of epochs, processing the training data in batches.

4.4.1 Generator Training:

Reconstruction: The generator creates a reconstructed version of the features.
Loss Calculation: Reconstruction loss (SmoothL1Loss) and Binary Cross-Entropy loss are computed based on the reconstructed and real data.
Backpropagation and Weight Update: The generator's weights are updated using its optimizer to minimize the combined loss.

4.4.2 Discriminator Training:

Real and Fake Predictions: The discriminator makes predictions on both real and reconstructed (fake) data.
Losses for real and fake predictions are computed separately using Binary Cross-Entropy.
Backpropagation and Weight Update: The discriminator's weights are updated to improve its ability to distinguish real from fake data.

4.5 Testing and Performance Evaluation Loop

4.5.1 Setting Models to Evaluation Mode:

The generator and discriminator are switched to evaluation mode, ensuring dropout and batch normalization layers are done correctly during testing.

4.5.2 Iterating Over Different Thresholds:

The loop tests the model's performance at varying thresholds to determine the best balance for fraud detection.

4.5.3 Processing Test Data:

For each batch in the test dataset, the generator reconstructs the features, and the discriminator predicts the probability of fraud.

4.5.4 Calculating Confusion Matrix Elements:

True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN) are calculated based on the discriminator's predictions and actual labels.

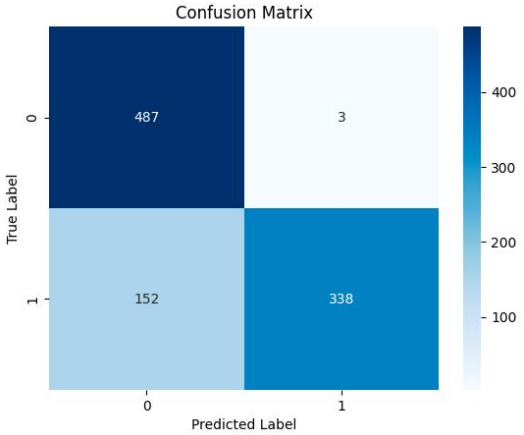

Figure 3: Confusion Matrix

4.5.5 Computing Performance Metrics:

Key metrics such as accuracy, recall, precision, F1 score, and Matthews Correlation Coefficient (MCC) are calculated and displayed.

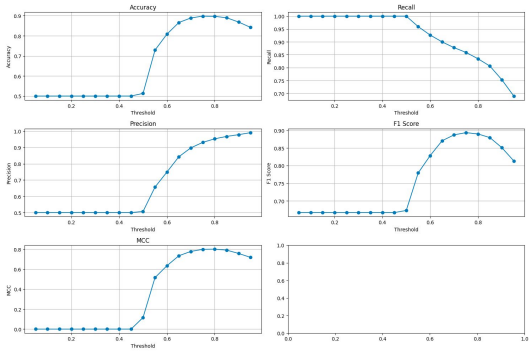

Figure 4: Performance metrics at different thresholds

4.5.6 ROC Curve and AUC Visualization:

The Receiver Operating Characteristic (ROC) curve is plotted, and the Area Under the Curve (AUC) is calculated for visual analysis of the model's performance. Key metrics such as accuracy, recall, precision, F1 score, and Matthews Correlation Coefficient (MCC) are calculated and displayed.

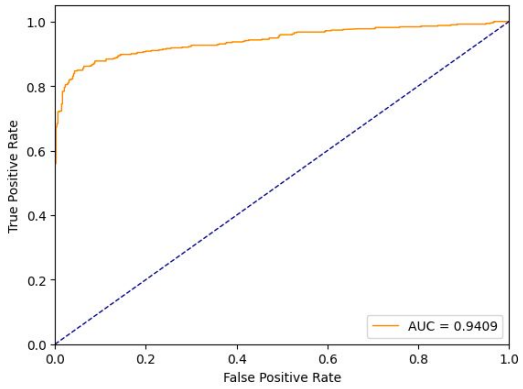

Figure 5: ROC Curve

4.6 Testing for Adversarial Data:

4.6.1 Fast Gradient Sign Method (FGSM) Attack:

- The FGSM attack is implemented to generate adversarial examples by perturbing the original data. This tests the model's robustness against subtle manipulations that mimic potential fraud strategies.

4.6.2 Evaluation on Adversarial Data:

- The generator (Autoencoder) and discriminator (Fully Connected Neural Network) models are set to evaluation mode.
- The test dataset is passed through the models, and adversarial examples are generated using the FGSM method with varying levels of epsilon (0.01 to 0.2).
- Performance metrics, including accuracy, recall, precision, F1 score, and Matthews Correlation Coefficient (MCC), are computed for each epsilon value to assess the impact of adversarial attacks on the model's ability to detect fraud.

4.7 Training the Model:

4.7.1 Adversarial Training with FGSM:

The generator and discriminator are retrained using both original and adversarial examples. The adversarial examples are created using the FGSM method with a selected epsilon value (0.2 in this case).

4.8 Combined Testing for Original and Adversarial Data:

4.8.1 Extensive FGSM Attack Testing:

A comprehensive test is conducted using a wider range of epsilon values (from 0 to 0.5) to generate adversarial examples using the FGSM method.

4.8.2 Performance Evaluation at Different Thresholds:

For each epsilon value, the model's performance is evaluated at various thresholds to determine the best threshold for maximizing accuracy, recall, precision, F1 score, and MCC.

4.8.3 Combining Original and Adversarial Data:

Predictions and labels for both original and adversarial data are combined to assess the model's overall performance in a scenario that mimics real-world conditions where both normal and manipulated data are present.

Table 1: Comparison of Benchmarks

Metric	Our benchmarks	Research Paper Benchmarks	After Adversarial Attack	After Adversarial Testing
Accuracy	0.8969	0.9061	0.5	0.8464
Precision	0.9534	0.9216	0.5	0.7929
Recall	0.8347	0.8878	1	0.8377
F1-score	0.8901	0.9044	0.6658	0.8377
MCC	0.8001	0.8128	0	0.6969

5 Results

In our credit card fraud detection study, the deep learning model initially achieved promising results comparable to benchmarks set in the referenced research paper, particularly at a threshold of 0.8, where the Matthews Correlation Coefficient (MCC) reached 0.8001, closely approaching the research paper's MCC of 0.8128. However, upon testing against Fast Gradient Sign Method (FGSM) adversarial attacks, the model initially displayed vulnerabilities with low accuracy (around 50%) and high recall (nearly 100%) across various epsilon levels. Subsequent training with adversarial data significantly enhanced the model's robustness, as evidenced by improved performance metrics even under adversarial conditions, with accuracy, precision, recall, and MCC values demonstrating the model's resilience and effectiveness in accurately detecting fraudulent transactions in a challenging adversarial environment.

6 Conclusion

In this study, we developed a deep learning model for credit card fraud detection, achieving metrics closely aligned with existing research benchmarks, especially in terms of the Matthews Correlation Coefficient (MCC). However, the model's initial vulnerabilities were exposed under Fast Gradient Sign Method (FGSM) adversarial attacks, leading us to enhance its robustness through adversarial training. This approach significantly improved performance against adversarial conditions. Furthermore, we attempted to extend the model's application to other datasets for broader anomaly detection. Despite these efforts, the results were not as promising, highlighting the challenges and complexities of adapting anomaly detection models to different contexts. This experience underlines the importance of tailored solutions in anomaly detection and contributes valuable insights into developing versatile and reliable systems for financial cybersecurity.

7 References

[1] S. Abbas and S. Nahavandi. Uncertainty-aware credit card fraud detection using deep learning. arXiv preprint arXiv:2107.13508, 2021.

[2] M. T. Thanh, M. Abdelrazek, and A. Babar. Deep learning methods for credit card fraud detection. arXiv preprint arXiv:2012.03754, 2020.

[3] T. Wu, and Y. Wang. Locally interpretable one-class anomaly detection for credit card fraud detection. arXiv preprint arXiv:2108.02501, 2021.

[4] H. Ba. Improving detection of credit card fraudulent transactions using generative adversarial networks. *arXiv preprint arXiv:1907.03355 [cs.LG]*, 2019.

8 Acknowledgments

We thank Prof. Marco Romanelli, Prof Alexandre Araujo, Prof Naman Patel, and Prof Chinmay Nerurkar and the teaching assistants for their guidance, mentorship, and expertise throughout this project.