Anti-Money Laundering: Feedforward Neural Network-based approach

Arslonbek Ishanov - 001196493 - ai4771p

Abstract

This project aims to outline the advantages of using Feedforward Neural Network to detect Money Laundering transactions over common Graph Neural Networks such as Graph Isomorphism Network and Principal Neighbourhood Aggregation.

1. Introduction

1.1. Problem Domain

Money Laundering is defined as the transfer of wealth derived from illegal activities to conceal its illicit origin or assist any persons involved in such pursuit. UN estimates that up to $2 trillion USD is laundered annually. Furthermore, these funds could be used to finance terrorism, proliferation (spread of biological, chemical, or nuclear weapons), drug and people trafficking. UNODC.Thus, financial institutions could benefit from an AI model that detects illegal activities.

1.2. Background research

Most of the banks are legally not allowed to publicly share their customer's data. Therefore, current research on the topic is limited due to sparse data availability. However, there are several studies on the subject with the most notable being IBM's synthetic dataset of Transactions for Anti-Money Laundering (AML) (**syntheticdata**). It specifically implements a small fraction of transactions as laundering, mimicking real life. While access to their algorithm which generates such data is restricted, IBM published 6 datasets that mimic real-world transactions. Out of these 6 transactions, Hi-Small_Trans.csv will be used for this project.(Altman 2019)

1.3. GNNs and FNNs in AML

The consensus is that to detect money laundering transactions, the use of a Graph Neural Network (GNN) model is required. However, GNNs require enormous computational power and long training time. Additionally, such models require the transactional data to be converted into a graph, further increasing computational requirements. This paper aims to analyse the viability of such consensus by applying Feedforward Neural Network (FNN) to the task of detecting money laundering activity.

1.4. Overall results

FNN performed extremely well when classifying regular transactions but did not fully detect illicit activities. The model could be used by smaller institutions (e.g., in poor countries which have limited computational capacity) to significantly reduce money laundering. It can compete with GNNs and sometimes outcompete them.

2. Methodology

2.1. The algorithm

Feedforward Neural Network (written with the help of ChatGPT for this project (OpenAI 2024)) is a specific type of supervised Artificial Neural Network, first proposed by Frank Rosenblatt in 1958. The model typically consists of 2-3 layers: input, hidden, and output, where the data flows in one direction: from the input layer, where the data is received, through the hidden layers, where the patterns are identified, and the output layer which provides the result. It is worth noting that hidden layers can consist of just one or multiple layers. Furthermore, each layer consists of at least one neuron which is essentially a mathematical function designed to mimic the work of a neuron in a human brain. Additionally, neurons are interconnected through weights which numerically represent the strength of the connection. (Rosenblatt 1958) The layers are then activated with either sigmoid which is described as:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

Or Rectified Linear Unit (ReLU):

$$\text{ReLU(x)} = \max(0,x) = \frac{x+|x|}{2} = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}$$

(Dubey, Singh, and Chaudhuri 2021) During training, the model learns through the change of weights after each piece of data is processed and the respective amount of error compared to the actual result. This process can be divided into 4 steps:

1. Calculate the final output after the first data is processed from the last hidden layer which can be represented as:

$$a_l = f\left(W_{l-1} \cdot a_{l-1} + b_{l-1}\right)$$

(Ang et al. 2020)

2. Calculate the loss (in this case, through binary cross-entropy loss):

$$L = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)]$$

(Ang et al. 2020)

3. Calculate the gradient of the loss function with respect to each weight using the chain rule of calculus which states that the derivative of a composite function equals the product of the derivative of the outer function and the derivative of the inner. In the case of FNNs, it is applied to the whole network, where the network serves as the outer function, and the layers – the inner functions, starting with the output layer and propagating backward.(Rumelhart et al. 1986)

4. Next, gradient descent adjusts the weights based on the gradients from backpropagation to minimise the loss. It adjusts the weights in the opposite direction of the result:

$$W_l = W_l - \eta \cdot \frac{\partial L}{\partial W_l}$$

Finally, the model convergence to a state where the model performs well on unseen data.(Ang et al. 2020)

While this model is not commonly used to detect money laundering activity, especially with the data designed for GNNs, the model's main strength lies in its quick training time and lower computational complexity compared to GNNs. The final FNN model's time complexity is

$$O(E \cdot T/B \cdot 97)$$

where E is the number of Epochs.(Ang et al. 2020)

T is the number of training samples.

B is the batch size.

The total number of neurons in the model is 97.

Or the general time complexity of an FNN model is O(N).

The time complexity of a GNN is typically:

$$O(E \cdot L \cdot (V + C)$$

where E – is the number of Epochs.

V – number of Nodes.

C – number of edges of a vertex.

The general time complexity of a GNN model is $O(N^2)$.

Thus, it is clear that the computational time complexity of FNN is far superior to GNN's.

3. Experiments

3.1. Dataset

The dataset used to train, validate, and test the model was obtained from IBM's publication on Kaggle, called "IBM Transactions for Anti Money Laundering (AML)". For this project specifically HI-Small_Trans.csv (Altman 2019) file was used. It contains a higher ratio of illicit transactions but is overall smaller than the other datasets. After downloading the dataset (and storing it in the same folder as the main code) the processed data (through IBM's provided code on GitHub), the data is loaded, reshuffled, reduced by sampling (to reduce the training time), formatted, split in the 60-20-20 ratio, and standardised.

3.2. Experimental Settings

To optimise the model some settings were adjusted. Specifically, the optimiser, activation functions, and epochs.

3.3. Evaluation criteria

To evaluate the model's performance, 3 criteria were used:

1. Accuracy: to determine, how accurately can the model classify transactions

2. F1-score: to calculate the model's precision using the formula:

$$F1 = \frac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}}$$

(Ang et al. 2020)

3. Confusion matrix: to better understand where the model underperformed.

3.4. Results

The base model with 3 layers, and 10 epochs showed 99.91% accuracy and provided an F1-score of 0.41. After some experimentation with the parameters, the model's accuracy was increased to 99.92% and its F1-score – to 0.52. The final model's confusion matrix was:

$$\begin{bmatrix} 253641 & 14 \\ 191 & 72 \end{bmatrix}$$

3.5. Discussion

This signifies that the model can accurately classify legitimate transactions, however, it struggles with identifying illicit ones. The main reason for this is that the illicit transactions comprise a tiny fraction of the entire dataset. Even with SMOTE technique and Oversampling the model did not improve. Reshuffling the data increased the identification of illegal activity from 72 to 86. The reduction in the number of epochs expectedly reduced the model's performance, while increasing them, improved it to 0.51 (F1-score) at 20 epochs. However, a further increase in the number of epochs led to the degradation of the model. The model performed better compared to Graph Isomorphism Network (GIN) which showed an F1-score of 0.29 (Wang and al. 2023), and was on-par with Principal Neighborhood Aggregation (PNA) with the F1-score of 0.57 (Wang and al. 2023).

4. Conclusion

To conclude, FNN is a strong model with a variety of applications. Its use in AML can compete with specific GNN models such as PNA and outcompete them (such as GIN) with far faster training time making it a good choice for AML. To further improve the AML, FNN could be transformed into a Deep Neural Network to capture more complex patterns and combined with other models (such as LightGBM) to improve its detection of illicit transactions as shown by one of the authour's group members, Sadiq Warsi.

References

Altman, E. (2019). *IBM Transactions for Anti-Money Laundering (AML)*. URL: https://www.kaggle.com/datasets/ealtman2019/ibm-transactions-for-anti-money-laundering-aml.

Ang, Koon Meng et al. (2020). "Optimal Training of Feedforward Neural Networks Using Teaching-Learning-Based Optimization with Modified Learning Phases". In: *Proceedings of the 2020 International Conference on Intelligent Computing*. Springer, pp. 259–264. URL: https://link.springer.com/chapter/10.1007/978-981-16-2406-3_65.

Dubey, Shiv Ram, Satish Kumar Singh, and Bidyut Baran Chaudhuri (2021). "Activation Functions in Deep Learning: A Comprehensive Survey and Benchmark". In: *arXiv*. URL: https://ar5iv.org/html/2109.14545.

OpenAI (2024). *ChatGPT*. URL: https://openai.com/chatgpt.

Rosenblatt, Frank (1958). "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain". In: *Psychological Review* 65.6, pp. 386–408.

Rumelhart, David E. et al. (1986). *Backpropagation: The Basic Theory*. URL: https://cpb-us-e2.wpmucdn.com/labs.utdallas.edu/dist/e/71/files/2020/12/BackpropagationTheBasicTheory.pdf.

Wang, J. and et al. (2023). "Understanding Anti-Money Laundering with Machine Learning". In: *arXiv*. URL: https://arxiv.org/pdf/2306.11586.