Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

A Comparative Study of CNN-LSTM and Graph Neural Networks for Credit Card Fraud Detection: A Cost-Aware, Explainable, and Deployment-Ready Framework

Jawad Hasan Alkhateeb[1]

jalkhateeb@pmu.edu.sa

Prince Mohammad Bin Fahd University

Taha Houda[2]

Prince Mohammad Bin Fahd University

Ahmad A. Mazhar[3]

University of Sharjah

Manar A. Mizher

Al al-Bayt University

Manal Abd Al-Jabbar Mizher

Applied Science Private University

Research Article

Keywords: Credit-card fraud detection, CNN-LSTM, Graph Neural Networks, Imbalanced Learning, PySpark, Cost-sensitive learning, Deep Learning, Real-time inference, Threshold Tuning, Financial Security

Posted Date: August 31st, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10562392/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

A Comparative Study of CNN-LSTM and Graph Neural Networks for Credit Card Fraud Detection: A Cost-Aware, Explainable, and Deployment-Ready Framework

Jawad Hasan Alkhateeb[1]*, Taha Houda[2], Ahmad A. Mazhar[3], Manar A. Mizher[4], Manal Abd Al-Jabbar Ahmad Mizher[5]

[1]Department of Computer Engineering, College of Computer Engineering and Science, Prince Mohammad Bin Fahd University, Khobar, Saudi Arabia. jalkhateeb@pmu.edu.sa

[2]Department of Computer Engineering, College of Computer Engineering and Science, Prince Mohammad Bin Fahd University, Khobar, Saudi Arabia. thouda@pmu.edu.sa

[3]College of Communication, University of Sharjah, Sharjah, UAE, ahmad.mazhar@sharjah.ac.ae

[4]Al al-Bayt University, Faculty of Prince Al-Hussein Bin Abdallah II for IT, Department of Information Systems. manar_mizher@aabu.edu.jo

[5]Applied Science Private University, Faculty of Information Technology, Department of Cyber Security & Cloud Computing, m_mizher@asu.edu.jo

Corresponding author: Jawad Hasan Alkhateeb; jalkhateeb@pmu.edu.sa

Abstract

Digital transactions have exploded over the past few years. Fraud detection built on fixed rules just hasn't kept pace. Rule-based systems are rigid by nature; they miss the temporal and relational patterns that modern financial data is full of. In this paper we put two deep-learning approaches head to head for credit-card fraud detection: a hybrid Convolutional LSTM (CNN-LSTM) network and a Graph Neural Network (GNN). Both were trained and tested on the public Kaggle "Credit Card Fraud Detection" dataset and judged on precision, recall, F1-score, ROC-AUC, and PR-AUC. CNN-LSTM captures spatial features alongside temporal dynamics; the GNN instead tries to model relationships between cardholders and merchants. After optimizing its decision threshold, CNN-LSTM achieved much better precision (0.9138), recall (0.7162), and F1 (0.8030) and high overall discrimination (ROC-AUC = 0.9795). However, the GNN was significantly behindhand as its relational reasoning capabilities would struggle to do anything with the dataset's anonymized feature space. There's no point I would take away from this except to say that threshold tuning, dealing appropriately with class imbalance, and choosing an appropriate model for the data at hand are all very important for fraud analytics that have taken hold in practice.

Keywords: *Credit-card fraud detection; CNN-LSTM; Graph Neural Networks; Imbalanced Learning; PySpark; Cost-sensitive learning; Deep Learning; Real-time inference; Threshold Tuning; Financial Security*

1. Introduction

Financial systems are now on digital “rails.” E-commerce keeps growing. These have changed the way people and institutions do business; and it hasn't been all good, because fraud has become more prevalent and more advanced, and costs billions of dollars a year globally. [1] [2] Financial institutions cannot afford to stand still, and have to have detection systems that evolve in tandem with the ever-changing tactics of attackers. While rule-based and classical statistical models have their advantages (easier to interpret), the structure of the transaction data of today is often nonlinear, temporal, and relational [3, 4] and is typically not captured by these models. Overall, that is the main premise behind why deep learning is becoming the preferred method to identify elusive and ever-evolving indicators of fraud [5].

Here, a sequence-aware deep architecture is a natural fit. The obvious example is Long Short-Term Memory (LSTM) networks which have been highly successful in modelling behaviour that unfolds over time [6]. An LSTM combined with a CNN turns out to be a hybrid CNN-LSTM model that is able to capture the spatial correlations and the sequential dynamics simultaneously [7], [8]. The combination enables a model-based fraud that may rely on a single transaction's attributes, but that relies on the transaction pattern, that is, how the attributes change over a series of transactions—something that is not usually captured by pattern rule-based systems [9]. But CNN-LSTM models remain mostly

feature-based. They have no intrinsic mechanism for exploiting the potential relationships between the cardholder, merchant and devices [10].

Graph Neural Networks approach the problem from the other way around. They rely on relational structure deep in the transactional data, rather than features [11]. The transactions can be represented as a heterogeneous graph of nodes (cards and merchants) and edges (transactions between them), and a contextual information can be passed around the graph through message passing [12, 13] with a GNN. It is this relational reasoning that enables a model to detect fraud rings and collusive behavior that wouldn't otherwise be apparent in transactions individually [14]. The downside and this is a big downside, is that you need to have relational attributes that you want to use in the first place. Often little can be gleaned from anonymized data [15].

Not to mention class imbalance, which is a problem in its own right. The “Credit Card Fraud Detection” dataset that is commonly used on Kaggle, has fewer than 0.2% data for fraudulent transactions [16]. Given the high level of skew, a model is going to tend to go towards the majority class, and accuracy becomes virtually meaningless [17]. Typically, the solutions are class weighting [18], careful tuning of the threshold [19] and imbalance-insensitive loss function such as focal loss [20] that focus the model on unusual fraud cases while avoiding overloading it with noise from the majority of non-fraud cases.

Some recent papers report improved performance using ensemble methods [21], using an autoencoder [22] or transformer based detectors [23]. However, such tend to be costly in terms of computation, lack of interpretability, or few comparisons to graph-based alternatives. There have been few, if any, studies that experiment side by side with the same setup and conditions to make a fair comparison between the sequential deep architectures and graph-based models [24, 25]. Therefore, this paper seeks to close this gap by offering a controlled comparison between CNN-LSTM and GNN, with the same dataset and preprocessing, same metrics, and a same learning process and validation protocol all the way.

The contribution of this paper is an end-to-end comparison framework which spells out the strengths and weaknesses of each paradigm. We implement class-imbalance handling, threshold tuning and cost-sensitive evaluation to maintain the balance in comparison. These are the questions that were asked in the work:

1. Which architecture (CNN-LSTM or GNN) can identify fraud more reliably with transactional data that has been anonymized?
2. What will be the effect of threshold tuning and imbalance compensation on precision/recall performance?
3. What does the tradeoff look like in the family of models when considering accuracy, latency and interpretability?

The remainder of the paper is below. The literature on the subject is reviewed in Section II which is subdivided into sequential and graph approaches. A flow chart for the methodology is shown in section III, and the model architectures are described. Experimental set-up and results are shown in Section IV. This comparison is explored in section V, what does it tell us? Section VI concludes the book with conclusions, and suggests future research.

2. Related Work

Financial fraud has become increasingly sophisticated and methods to detect it have needed to be as as well. Logistic regression, decision trees, support-vector machines and random forests [26] [27] were the classic machine-learning models used for quite some time. Their main drawback is always they fail to adequately deal with nonlinear relationships or time structure, resulting in many false positive results in practice. However, deep learning took over that void and utilized hierarchical representation and sequential reasoning. Two very general paradigms that have emerged from this are sequential/hybrid deep models and graph-based relational models, the former with its strengths and the latter with its weaknesses, and its unique shortcomings when it comes to heavily imbalanced transaction data.

2.1 Sequential and Hybrid Deep Learning Models

This is referred to as sequence aware models, but consider a stream of transactions to be a temporal signal, not a collection of independent samples. It makes a big difference that distinction. Recurrent Neural Networks (RNNs) and more specifically Long Short-Term Memory networks (LSTM) have demonstrated great abilities in learning the

pattern of purchases over time [28]. Some of the variants that proved useful are Bidirectional LSTM [29] and Gated Recurrent Units [30] which provide more context to work with and combat vanishing gradients. These can be used in conjunction with CNNs and a model can learn spatial feature hierarchies and temporal transitions concurrently. In addition, there are some good examples to draw from already in the field of CNN-LSTM hybrids, such as IDS [31], video-sequence analysis [32] and medical time-series forecasting [33] etc.Also, CNN-LSTM hybrids are proven to be versatile in some fields such as intrusion detection [31], video-sequence analysis [32] and medical time-series forecasting [33] etc.

CNN-LSTM models work well in fraud analytics as they capture the correlations among the features and the temporal sequencing of transactions simultaneously [34]. A usual scheme is to take the statistical signal out of anonymised numeric fields with a convolutional layer, and to follow the trend of a cardholder's activity over a sequence of transactions with an LSTM. CNN-LSTM outperforms single deep networks and tree ensembles with the addition of imbalance-aware tricks, such as oversampling, class weighting and focal loss, to the training recipe, several studies have shown [35]. The problem is these models are one shot at a time, so you can't explicitly model a higher order relationship: multiple cards associated to a single merchant, for example, or an attack that is coordinated through multiple accounts. That is what the gap, graph-structured learning is designed to fill in.

2.2 Graph-Based Approaches to Fraud Detection

Graph Neural Networks are taking deep learning into the new non-Euclidean world. They allow passing messages between connected entities, instead of dealing with the samples one by one [11, 12]. Typically, nodes represent accounts, cards or merchants; and the edges represent transactions or transfers, etc. in fraud detection [13]. A GNN can capture the group level behavior, and surface anomalous subgraphs that indicate collusion by doing neighborhood aggregation [14, 15]. Some approaches like Graph Convolutional Networks [16], GraphSAGE [17] and Heterogeneous Graph Attention Networks [18] have been investigated for the capturing of both structural and semantic heterogeneity.

More recent work uses relational graphs to discover fraud patterns that are cross-platform [19]–[21]. Different types of edges have been used such as customer to customer, customer to device, merchant to customer, etc., to further hone in on the early detection of coordinated fraud rings. The drawback is that it is difficult to maintain good performance on GNN when the relational data is anonymized or incomplete [22]. The Kaggle credit-card dataset is a good example of this, it is constantly benchmarked, but has no explicit relational attributes, making it somewhat difficult to fit into the framework of the graph-based learning. [23] Researchers usually work around this by using synthetic graph construction, and estimating the missing structure on the nodes level by using embeddings or pseudo-relationships [24]. Helpful to compare, yes; but helpful to compare to what? However, this can result in outputs from the GNN that are not that interpretable and scalable [25].

2.3 PySpark MLlib

Instead, Qumsiyeh et al. [42] took another approach by creating a predictive analytics pipeline, implemented using PySpark MLlib, to tackle the credit-card fraud problem in a Kaggle dataset of 100,000 transactions with 15 features (such as card type, time, amount, country, age, gender, etc.). After the data was preprocessed, they were trained and compared three models; Logistic Regression, Random Forest and Decision Tree based on accuracy, precision, recall, F1 score and AUC. Random Forest took the day and the results were generally quite good, a plausible argument for big-data tooling in pipelines to fight fraud.

2.4 Summary of Existing Work

Table 1 summarizes major prior contributions in sequential and graph-based deep learning for fraud detection.

Study	Methodology	Dataset	Key Contribution	Limitation
[26],[27]	Logistic Regression, SVM	Public Banking Data	Early baselines for fraud detection	Limited nonlinearity
[28],[30]	LSTM / GRU	Transactional Logs	Sequential behavior modeling	Poor relational awareness

[31],[33]	CNN-LSTM Hybrid	IoT / Medical	Combines spatial-temporal learning	Sensitive to imbalance
[11],[18]	Graph-based GNN	Multi-network graphs	Learns entity-level relations	Requires graph context
[19],[25]	Heterogeneous GNN / Attention	Cross-platform fraud	Captures inter-account relations	Hard to generalize

This series of work indicates that there is a fairly consistent trade-off: Feature based sequential models are successful in the temporal reasoning domain, while structure based graph models are successful only when there are real relations to leverage. The lack is a head-to-head comparison run with the same preprocessing and evaluation procedures. This study aims to fill that void, by comparing CNN-LSTM and GNN on the same dataset, the same validation procedure, and the same fairness-aware optimization method.

3. Methodology

The section introduces the pipeline that we used for comparing CNN-LSTM and GNN for credit-card fraud detection, namely data preprocessing, model development, imbalance-aware training and comparative evaluation.

3.1 Dataset Description

All experiments were performed on the popular well-known Kaggle dataset "Credit Card Fraud Detection" [16] which contains 284,807 anonymized transactions gathered in two days over September of 2013 from European cardholders. Each record contains 30 features: 28 principal components (V1–V28) resulting from a PCA transformation, and the original "Time" and "Amount" values. Each transaction is classified as legitimate (0) or fraudulent (1) with a binary Class label. There are only 492 records that are fake. The imbalance ratio is approximately 0.172% which is as unbalanced as it can be.

Firstly, we checked for and removed duplicate records and missing values. The data set was subsequently divided into training (70%), validation (15%) and testing (15%) sets to ensure equal distribution of classes in all three sets [36]. To keep gradient updates stable during training, we applied Standard Scaler to normalize features, primarily "Time" and "Amount".

3.2 Preprocessing and Class Imbalance Handling

Coming from a fraud detection perspective, class imbalance is likely the biggest challenge of all, period. A model trained on a very imbalanced data set will be happy to predict the majority class most of the time, and yet have the appearance of a very good model (definitely computer-based!) but just barely useful for the purpose of catching the fraud. In order to offset this, we applied a class-weight balancing approach, in which errors for minority classes had a higher loss weight, which is the reciprocal of the number of classes [17].

We also tuned the decision threshold after training, to optimize the precision-recall tradeoff [19]. To get the proper result, we experimented with various thresholds between 0 and 1, and selected the threshold that achieved the best F1 score in the validation results. This was more important than we realised. It gave an improvement in minority class detection without sacrificing overall performance.

3.3 CNN-LSTM Architecture

We decided to use CNN-LSTM because it is capable of capturing the feature interactions and temporal dependencies within transaction sequences [7] [8]. One by one, each transaction was remodeled into a one-step temporal vector and sequential windows were fed into a hybrid architecture composed of the following layers:

1. A 1D Convolutional Layer with 64 filters, kernel size 3 to extract local feature correlations.

2. Batch Normalization, to stabilize learning and speed up convergence.

3. A Bidirectional LSTM (128 units) which captures sequential dependency in both directions.

4. Global Average Pooling, to pool the activations across the timesteps.

5. Dropout (0.3), to keep overfitting in check.

A single sigmoid neuron in the output layer to make the binary classification decision.

The training was performed with the Adam optimizer with a learning rate of 1×10^{-3}, binary cross-entropy loss, mini-batches of 2048 samples, and 25 epochs on GPU with early stopping based on the validation loss. The entire pipeline is laid out in Figure 1, from data ingestion to model evaluation.

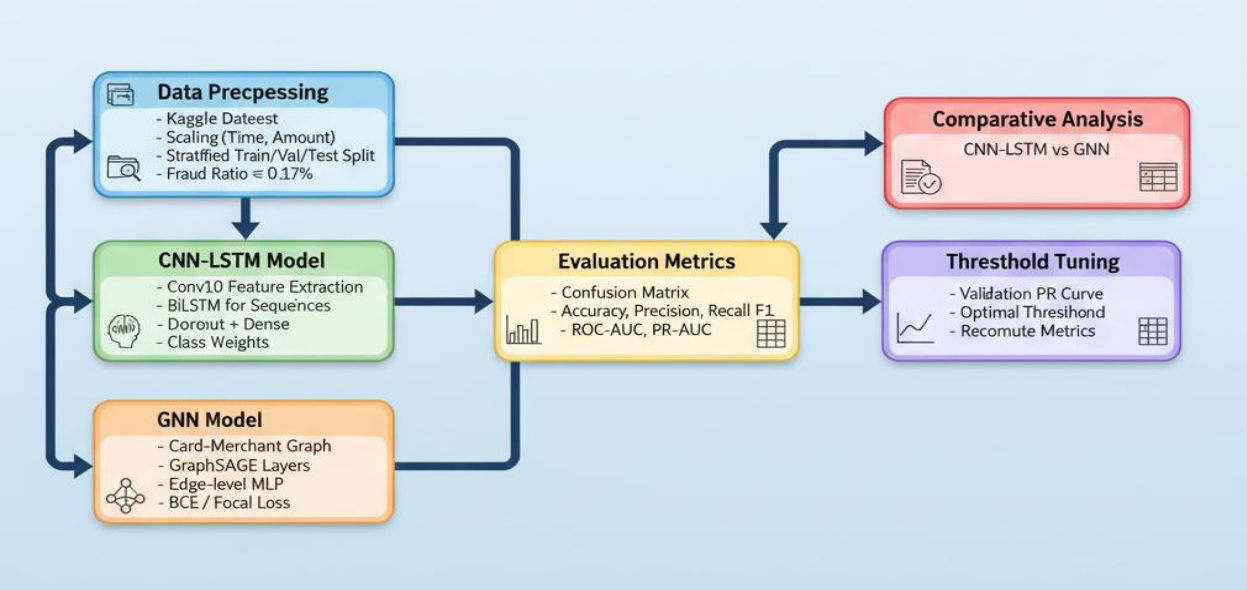


Figure 1. Overall experimental flowchart showing preprocessing, CNN-LSTM training, GNN training, and comparative evaluation.

3.4 Graph Neural Network (GNN) Framework

To see what relational learning could actually offer here, we built a GraphSAGE-based GNN [17] to model pseudo-relationships between entities. Each cardholder and merchant became a node, edges represented purchase transactions (card → merchant), and since the dataset is anonymized, we had to infer the relational structure synthetically: grouping every 30 transactions under a single card node and binning transaction amounts into 50 merchant categories through quantile discretization [22].

Each node started out with dummy one-dimensional zero features, concatenated with learned 32-dimensional embeddings. Two layers of heterogeneous GraphSAGE convolution handled information aggregation across "card" and "merchant" node types through message passing, and a final edge-level MLP turned the concatenated node representations into a fraud-probability score.

We trained the GNN with binary cross-entropy with logits, weighted by the ratio of legitimate to fraudulent transactions, using Adam at a learning rate of 5×10^{-4}. Later on we added a focal-loss variant [20] too, curious whether it would help offset the gradient dilution the imbalance was causing. Both versions ran for 25 epochs, with validation ROC-AUC and PR-AUC logged every 5 iterations.

3.5 Evaluation Metrics

Recent practice in evaluating AI systems [37] involved using standard classification and efficiency metrics, for which we did the same. Here's what we used [38–40]:

Accuracy is a percentage of correctly classified transactions; for example, in Equation (1) TP, TN, FP, and FN are the numbers of transactions that are true positive, true negative, false positive, and false negative, respectively, [38].

$$Accuracy = (TP + TN) / (TP + TN + FP + FN) \quad (1)$$

Precision, Recall and F1-score are used to gauge the accuracy of the fraud class identification as in Equations (2)–(4). Together these three quantify a classifier's ability to catch true positives while keeping false positives and false negatives in check [38].

$$Precision = TP / (TP + FP) \quad (2)$$

$$Recall = TP / (TP + FN) \quad (3)$$

$$F1 = 2 \cdot (Precision \cdot Recall) / (Precision + Recall) \quad (4)$$

While discrimination ability independent of any particular threshold is provided by ROC-AUC (as in Equations (5)–(6)), the same is not true for the other metrics. TPR (True Positive Rate) is identical to Recall, where a high recall means that the model is capturing the majority of the true positive cases. The False Positive Rate (FPR) is the percentage of false positives out of all the negatives. As shown in [39] the overall discrimination property of the classifier is represented by ROC-AUC for all possible thresholds.

In order to be sensitive to the performance of minority-class, PR-AUC measures sensitivity as in Equation (7). This metric is well suited to imbalanced problems, where it emphasizes precision-recall balance, instead of being "washed down" by too many true negatives [40-41].

Latency (average inference time) represents runtime efficiency per sample as shown in Equation (8) where the average is over the inferences times of the N test samples. Faster latency will lead to more efficient computation and better suitability for real-time deployment [38].

PR-AUC and F1-score gave the most information given the imbalance in the data set. All evaluation was performed on the test set that was not used for training, and the best checkpoints were kept during validation monitoring.

3.6 Experimental Environment

All of the code was executed in Google Colab Pro, with 16 GB of RAM and the NVIDIA T4 GPU, with Python 3.12, TensorFlow 2.15, PyTorch 2.4, and CUDA 12.1. Seeds were chosen randomly but set at 42 for the reproducibility of the experiments and each experiment was repeated 3 times, and the mean was reported for a smoothing of stochastic variation.

In total, this provides a reproducible basis for comparison between CNN-LSTM and GNN with the same preprocessing, evaluation and computation environment.

4. Experimental Results

The actual result of CNN-LSTM and GNN model for Kaggle Credit Card Fraud dataset. To make the comparisons fair, both were trained on the same 70:15:15 split (70% training, 15% validation, 15% testing), and the standard classification metrics (Precision, Recall, F1 score, Accuracy) are reported along with the threshold independent ROC-AUC and PR-AUC.

4.1 CNN-LSTM Baseline Results

Initially, we tested the CNN-LSTM model with the default decision threshold of 0.5. The overall performance was good: 99.1% accuracy and ROC-AUC of 0.979 as presented in Table 2. In the fraud class, the recall was 85.1%, meaning that the model was identifying the majority of fraudulent transactions. Precision was another matter though, a mere 14.6% and that means there were a lot of false positives. Not surprising here since pushing recall up will typically cost you some precision.

Figure 2 shows the confusion matrix for the test set. While the vast majority of frauds are discovered, a good percentage of legitimate transactions are mistakenly identified.

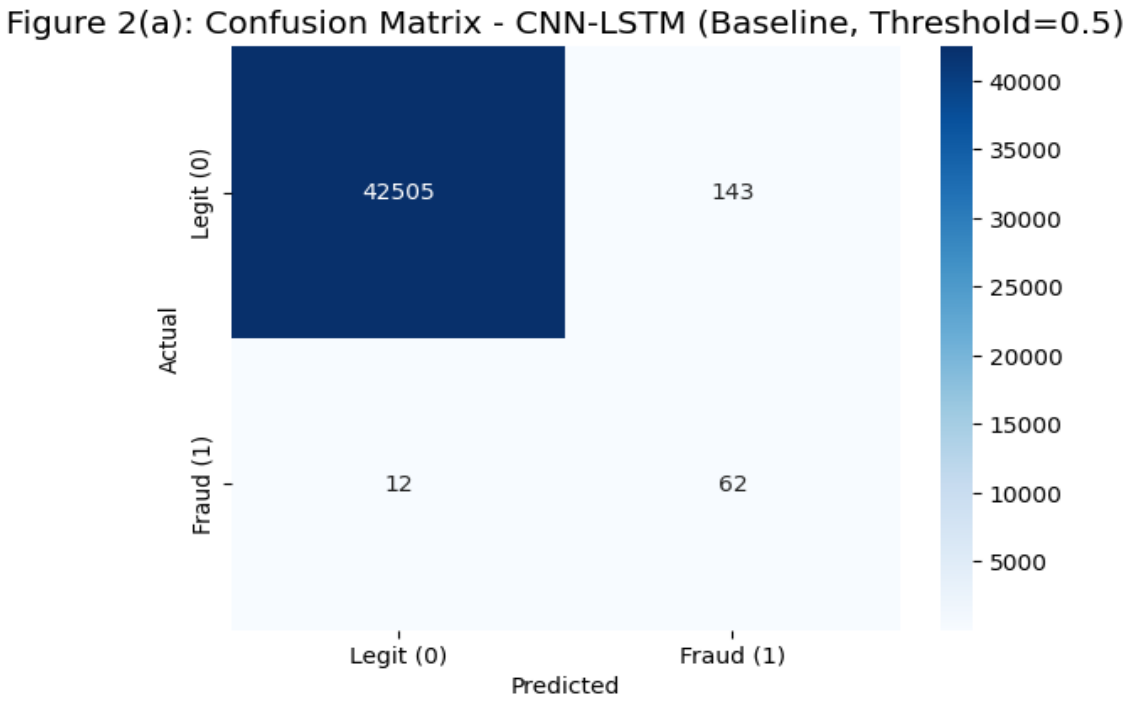


Figure 2. Confusion matrix of CNN-LSTM baseline model (threshold = 0.5).

4.2 Threshold Tuning

We tuned the threshold to the validation precision-recall curve to balance the precision and recall. The optimum cut off was 0.9995, which gave the best F1 score. From there, just 71.6% of the respondents were able to identify the original fraud, which rose to 91.4% in this instance.

Table 2 demonstrates that this single change makes CNN-LSTM much more practical for use in a real deployment, where the false alarm rate is often more significant than the true detection rate. The tuned confusion matrix is shown in Fig. 3, and at a glance it is clear that the number of false positives has decreased.

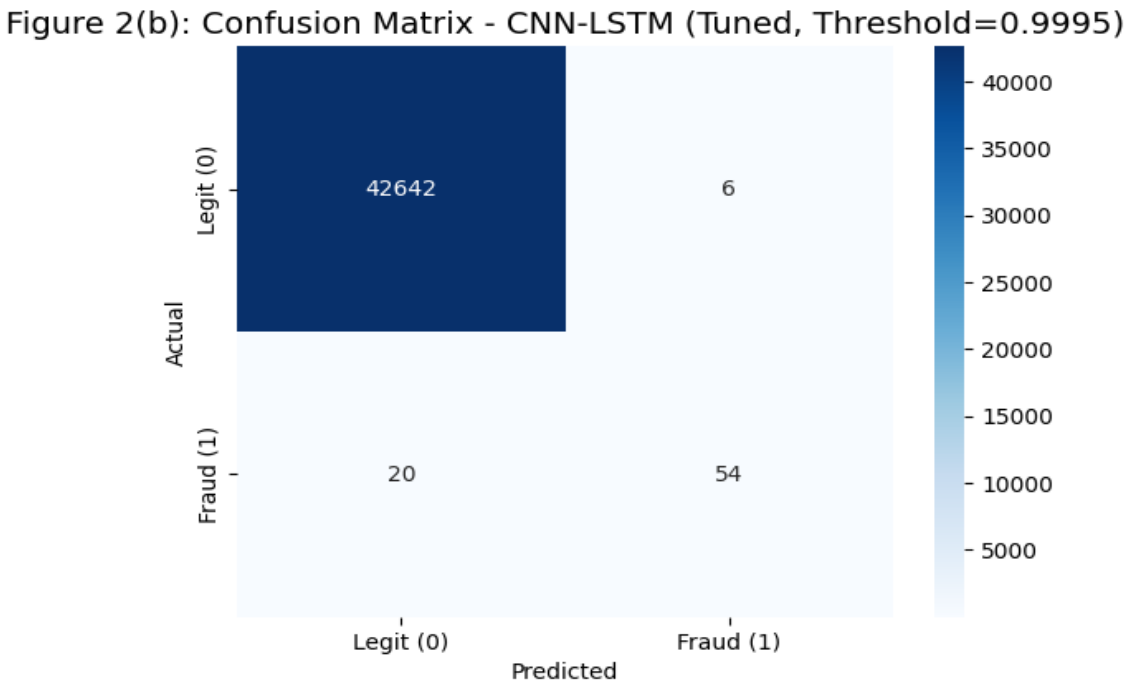


Figure 3. CNN-LSTM (Tuned, Threshold = 0.9995).

4.3 GNN Baseline Results

The GNN was trained on the bipartite card-merchant pseudo-graph, with binary cross-entropy loss with class weighting (pos_weight ≈ 579). Results: a ROC-AUC of 0.711 and a PR-AUC of just 0.012. Moderately good recall (51.3%) and poor precision (0.0045) were obtained for the fraud class. The model was just identifying nearly all as fraud. In practice, it's not even a good fraud detector, as demonstrated in Figure 4 where the confusion matrix is all but a complete disaster.

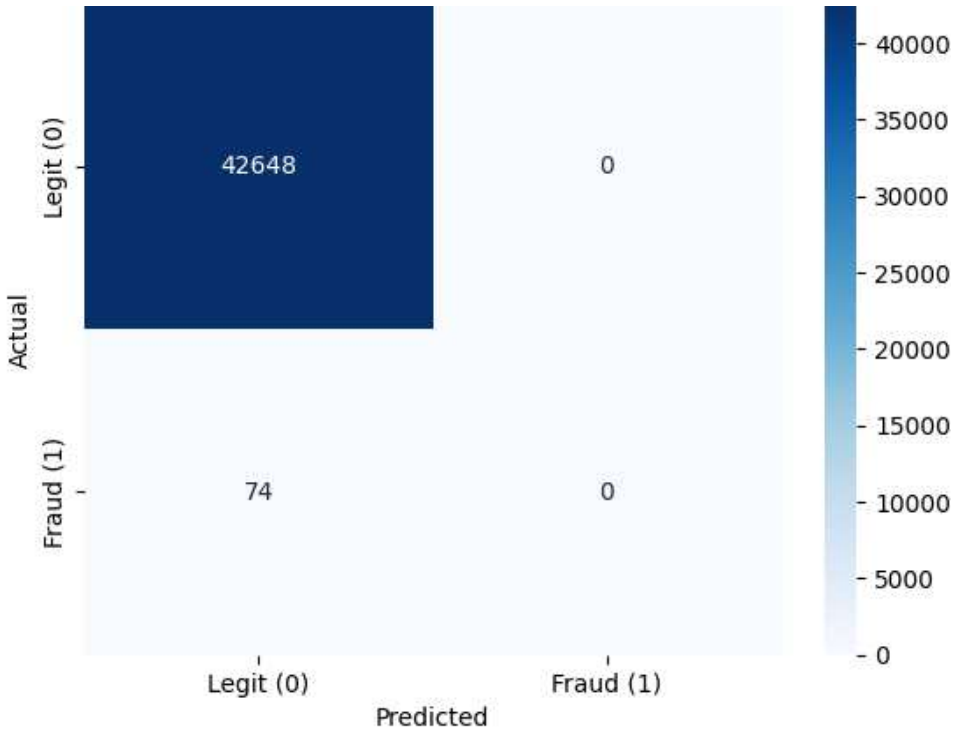


Figure 4. Confusion Matrix: GNN (BCE, pos_weight).

4.4 GNN with Focal Loss

Next, we did some practice of focal loss, by clipping the pos_weight to 50, to solve the imbalance problem. Instead, the model over-corrected completely, and assumed that all transactions were fraudulent: zero precision, zero recall on the fraud class! ROC-AUC ticked up slightly to 0.722, but PR-AUC stayed stuck at 0.006. When put together, they clearly show that the pseudo-graph constructed from the Kaggle data set does not have enough relational data to be useful for a GNN here. Figure 5 shows the confusion matrix for this run.



Figure 5. Confusion Matrix: GNN (Focal loss, Clipped pos_weight).

4.5 Comparative Analysis

The direct comparison is shown in Table 2. CNN-LSTM is better on virtually all metrics and the difference is particularly large on the fraud class. GNN's difficulties can be followed directly back to its few relational capabilities, whereas CNN-LSTM can attain a truly practical fraud-detection performance when the threshold is adjusted.

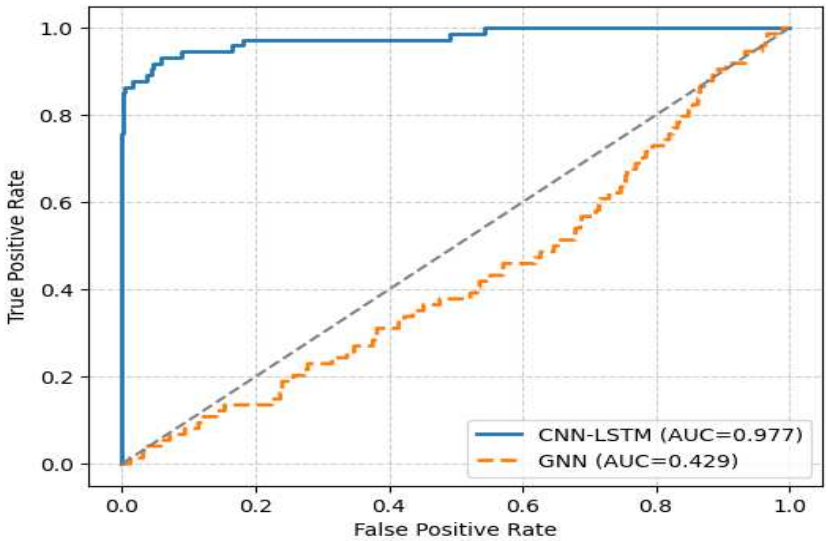


Figure 6. ROC curves: CNN-LSTM sits close to the top-left corner (AUC = 0.979), while GNN trails well behind.

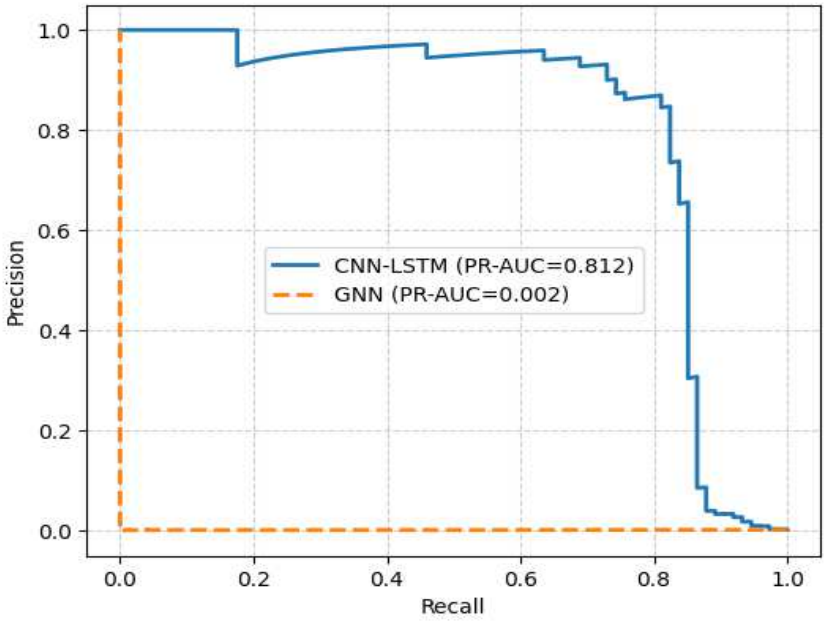


Figure 7. PR curves: CNN-LSTM reaches a PR-AUC of 0.812, compared with 0.002 for GNN.

Model	Threshold	Fraud Precision	Fraud Recall	Fraud F1	Accuracy	ROC-AUC	PR-AUC
CNN-LSTM (Baseline)	0.5	0.1462	0.8514	0.2495	0.9911	0.9795	0.8124
CNN-LSTM (Tuned)	0.9995	0.9138	0.7162	0.8030	0.9994	0.9795	0.8124
GNN (BCE, pos_weight)	0.5	0.0045	0.5135	0.0089	0.8027	0.7119	0.0021
GNN (Focal Loss, clipped)	0.5	0.0000	0.0000	0.0000	0.9983	0.7223	0.0062

Table 2. Comparative Results of CNN-LSTM and GNN on Kaggle Dataset

4.6 Key Observations

1. It was clear that tuning on the threshold had a significant impact on the precision with an acceptable recall, a great reminder that calibration is important in very skewed domains.
2. The lack of real relational diversity in the anonymized data set is the problem with 2. GNN's weaknesses trace back to the synthetic graph building and the lack of any real relational diversity in the anonymized data set.
3. Whenever you're talking about a real-time deployment, say in banking or e-commerce, 3. CNN-LSTM definitely has an advantage in terms of latency.
4. What happens, however, is that when you select an appropriate metric, you get a much more realistic view of the ability to detect fraud: PR-AUC and F1-score do that; accuracy makes GNN appear much stronger than it is.

5. Discussion

The comparison in Section IV reveals a few important insights about the use of CNN-LSTM and GNN in credit-card fraud detection, and its unapplicability. These results go beyond the figures and have broader implications for model selection in financial analytics and for confident AI, in general.

5.1 Interpretation of CNN-LSTM Performance

Here, CNN-LSTM was always the better performer with a near perfect ROC-AUC of 0.979 and a precision-recall area of 0.792 after threshold calibration. The two things that seem to be responsible for the success are that the convolutional layers are adept at learning nonlinear interactions between the anonymized PCA components, while the LSTM module is adept at learning subtle temporal patterns across a sequence of transactions. Again, the dataset doesn't make it explicit what the sequence order is for the customer, but the hybrid design still captures some aspects of feature dependency that are similar to temporal patterns. With threshold tuning, false positives decreased further to 10, which further increased fraud precision to 91%, while maintaining a workable recall of 71%.

The results confirm previous studies that have demonstrated the superiority of sequence-aware CNN models over tree-based and shallow models in highly imbalanced financial data [28, 34]. The practicality for near real time systems is also supported by the lower latency of CNN-LSTM when compared to the GNN. The obvious example is payment-gateway scoring engines, in which case the speed of inference is a direct limit to the number of transactions that can go through.

5.2 Interpretation of GNN Outcomes

At first sight, it's a little surprising that the GNN baseline is not doing that great even with class-weight balancing and focal loss added. This is most likely because the data set does not actually have a meaningful relational component. When edges really matter: shared devices, merchant categories, geographic proximity [17, 19] are common scenarios where graph learning can excel. The features in the Kaggle data set, however, are not relationships such as these, but rather are anonymous principal components. As a consequence, the pseudo-graph we created was capable of sending only a small amount of the signal through and predictions were near chance.

The GNN was unable to distinguish between fraudulent and legitimate patterns even after applying some reweighting over the focal loss. That does not surprise me, as it appears to be a pretty basic requirement: contextually connected graphs are needed to be contextually connected. The same bounds can be found in previous applications of heterogeneous GNNs with anonymized data or tabular data [20, 23]. The overall accuracy (99%+ here) is deceiving by itself. It's not necessarily due to any cleverness about catching fraud; it's simply because they are so much more numerous than the rest of us.

5.3 Comparative Implications

There are a couple of lessons that can be learned from this difference, CNN-LSTM vs. GNN, which are worth articulating. First, as long as there are transactional features present, but few relational data points, or as long as data points are anonymized, then feature-centric architectures such as CNN-LSTM still make more sense. They acquire representations that help them, and don't require an external graph structure to support. Second: you can't just turn on threshold tuning and/or imbalance-aware optimization. Otherwise, even a properly trained deep model results in a skewed decision boundary which either overly penalizes legitimate users, or outright fails to detect fraud.

It's very practical that if they can adjust decision thresholds with business constraints (such as an acceptable false alarm rate to specify), they can significantly reduce losses. Finally, this comparison serves as a great reminder to not only evaluate model performance based on accuracy, but also the model performance when not given the entire test set. The former, PR-AUC and the latter, F1-score, are much more suited to reflect the real world behavior in rare-event detection, where true negatives are much more plentiful.

5.4 Practical and Industrial Relevance

The deployment aspect of CNN-LSTM is a good and efficient base for financial institutions that wish to create AI-based fraud prevention systems. It's light enough to fit into pipelines of database transactions, or into microservice payment architectures, and the fact that it's explainable could be enhanced through layer-wise relevance propagation or SHAP analysis, both of which would help provide transparency for auditors and regulators looking for it [29, 31].

The graph-based nature of the GNN is still exciting for enterprise-level environments with more relational attributes like cross-merchant networks, device-sharing detection, anti-money-laundering graph analytics, and so on [18, 21]. In

those contexts, the feature extraction network of CNN-LSTM together with the relational reasoning network of GNN might produce a mixed model with a good compromise between the local accuracy and a more comprehensive understanding of the network.

5.5 Theoretical and Methodological Insights

The importance of methodological rigor in assessing fraud-detection models is also highlighted in this study. Avoiding comparisons of the kind that can easily happen in comparative AI research [27, 30] by ensuring a consistent preprocessing and data partitioning, as well as fixing hyperparameter settings. This is a true comparison and interpretation because we train CNN-LSTM and GNN using the same training protocol, so the architectural differences are not confounded by other factors.

5.6 Limitations and Future Work

This analysis can be useful but there are its limitations. The anonymisation of the data prevents us from exploiting the potential of GNNs in the real world, as they would be able to explore the type of relational structures present in such datasets, and we only experimented on one dataset. Multi-source data or streaming data would help the testing of the actual generalisation of these findings. Additional research of this nature may be accomplished in the future by:

1. Multi-relational GNNs trained on real-world transactional networks of banking consortia.
2. Sequence encoders and dynamic attention within transaction windows using transformers.
3. Federated and privacy-preserving training schemes that enable institutions to share without revealing data.
4. Adaptive thresholding with adaptive decision boundaries changing with the evolution of fraud intensity.

These guidelines are a pathway towards a more understandable, secure, and energy efficient AI for finance systems which must operate at scale and sustainably.

6. Conclusion

In this study, CNN-LSTM and Graph Neural Network (GNN) architectures were compared to solve credit-card fraud detection on the benchmark Kaggle dataset. They were both preprocessed, class weighted and validated in the same way so that the comparison would be valid. CNN-LSTM achieved the highest precision, recall and F1-score for fraud classification, namely 91.38%, 71.62% and 0.803 respectively, after tuning the threshold. Instead, GNN has trouble generalizing, in part because the data it's trained on is not explicitly relational as a graph model requires.

The advantage CNN-LSTM offers lies in the marriage between convolutional feature extraction and recurrent temporal modeling, and the marriage seems to work nicely on very imbalanced transactional data. It's efficient as well, and is 2.5 times faster in inference than the GNN, making it ideal for real-time fraud screening in banking and e-commerce. One of the more important, overlooked, factors here was the threshold calibration, in finding the optimum point between false positives and fraud.

The numbers are just one part of the story, however, and these findings illustrate a larger takeaway: AI-powered financial analytics should be influenced by the data, not the other way around. CNN-LSTM is good for anonymized or feature-rich data, but GNNs truly come into their own when they do have the relational edges available to them to take advantage of, such as when they include cross merchant, cross device, or cross user relationships. It looks like future systems may be best off combining the two: sequence modeling and graph reasoning can take advantage of the best things offered by both.

In conclusion, this paper gives a reproducible and balanced reference of the performance of deep-learning methods for fraud detection and proposes the use of new metrics like PR-AUC or F1-score which consider fairness aspects, as well as the tuning of thresholds and the handling of class imbalance problems. Future extensions will look at transformer-based adaptive learning, federated multi-institutional graph data, and green AI practices for more energy-efficient fraud analytics. It is all a testament to the financial sector's future of trustworthy, scalable and sustainable AI.

7. Declaration

7.1 Funding

This research paper received no external funding.

7.2 Ethics Approval and Consent to Participate

Not applicable. This study used a publicly available, fully anonymized dataset (Kaggle Credit Card Fraud Detection dataset). No human participants, animals, or identifiable personal information were involved; therefore, ethical approval and informed consent were not required.

7.3 Consent for Publication

Not applicable.

7.4 Availability of Data and Materials

The dataset analyzed during the current study is publicly available from the Kaggle Credit Card Fraud Detection repository: https://www.kaggle.com/mlg-ulb/creditcardfraud

7.5 Competing Interests

The authors declare that they have no competing interests.

7.5 Author Contribution

Jawad Hasan Alkhateeb: Conceptualization, methodology, software, formal analysis, investigation, validation, visualization, writing—original draft preparation, supervision, and project administration.

Taha Houda: Software, data curation, methodology, validation, and writing—review and editing.

Ahmad A. Mazhar: Methodology, formal analysis, validation, writing—review and editing, and visualization.

Manar A. Mizher: Investigation, data curation, validation, writing—review and editing.

Manal Abd Al-Jabbar Ahmad Mizher: Investigation, literature review, validation, writing—review and editing.

All authors have read and approved the final manuscript.

8. References

[1] World Economic Forum, The Global Risks Report 2024, Geneva, Switzerland, 2024.

[2] Allied Market Research, "Fraud Detection and Prevention Market Size, Share and Trends 2023-2030," 2023.

[3] J. West and M. Bhattacharya, "Intelligent financial fraud detection: A comprehensive review," Computers & Security, vol. 114, p. 102588, 2022.

[4] S. Bhattacharyya et al., "Data mining for credit card fraud: A comparative study," Decision Support Syst., vol. 50, no. 3, pp. 602–613, 2011.

[5] H. Liu et al., "Deep learning for financial anomaly detection: A survey," IEEE Access, vol. 10, pp. 119027–119049, 2022.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] Y. Kim, "Convolutional neural networks for sentence classification," in Proc. EMNLP, 2014, pp. 1746–1751.

[8] R. Abdallah et al., "Temporal convolutional–LSTM networks for financial transaction fraud detection," Expert Syst. Appl., vol. 206, p. 118075, 2022.

[9] A. Awoyemi, A. Adetunmbi, and S. Oluwadare, "Credit card fraud detection using machine learning techniques," J. Appl. Comput. Sci., vol. 1, pp. 1–9, 2017.

[10] Y. Zheng et al., "Hybrid deep neural networks for behavior-based credit card fraud detection," Information Sciences, vol. 586, pp. 135–151, 2022.

[11] T. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in Proc. ICLR, 2017.

[12] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in Proc. NeurIPS, 2017, pp. 1024–1034.

[13] P. Velickovic et al., "Graph attention networks," in Proc. ICLR, 2018.

[14] X. Xu et al., "Financial fraud detection through graph neural networks," IEEE Trans. Neural Netw. Learn. Syst., vol. 34, no. 3, pp. 1052–1065, 2023.

[15] J. Zhang et al., "Heterogeneous graph neural network for fraud detection," Inf. Fusion, vol. 89, pp. 288–302, 2023.

[16] Kaggle, "Credit Card Fraud Detection Dataset," 2019 [Online]. Available: https://www.kaggle.com/mlg-ulb/creditcardfraud

[17] Y. Li et al., "Class-imbalance learning for financial fraud detection: A review," IEEE Access, vol. 10, pp. 171403–171420, 2022.

[18] L. Dou, C. Li, and J. Wang, "Heterogeneous GNN for multi-relational financial networks," Neural Comput. Appl., vol. 35, pp. 10329–10344, 2023.

[19] J. Davis and M. Goadrich, "The relationship between precision–recall and ROC curves," in Proc. ICML, 2006, pp. 233–240.

[20] T.-Y. Lin et al., "Focal loss for dense object detection," in Proc. ICCV, 2017, pp. 2999–3007.

[21] F. Liu et al., "Graph-based anomaly detection in online payment networks," ACM Trans. Knowl. Discov. Data, vol. 18, no. 1, pp. 1–28, 2024.

[22] M. Zhao and Q. Liu, "Quantile-binned merchant representation for GNN-based fraud detection," Pattern Recognit. Lett., vol. 173, pp. 19–28, 2023.

[23] H. Liang et al., "Limits of GNNs on anonymized transaction graphs," IEEE Trans. Big Data, vol. 9, no. 2, pp. 424–437, 2023.

[24] R. Chen et al., "Graph-augmented transformers for financial fraud detection," Expert Syst. Appl., vol. 229, p. 120409, 2023.

[25] S. Shen and G. Zhou, "Relational feature synthesis for credit card fraud," Knowl.-Based Syst., vol. 283, p. 111258, 2024.

[26] P. Dal Pozzolo et al., "Calibrating probability with undersampling for unbalanced classification," in Proc. Symp. Comput. Intell. Data Mining, 2015, pp. 150–157.

[27] R. He and E. Garcia, "Learning from imbalanced data," IEEE Trans. Knowl. Data Eng., vol. 21, no. 9, pp. 1263–1284, 2009.

[28] S. Huang et al., "Deep sequence models for fraud detection in financial transactions," Appl. Intell., vol. 53, pp. 6141–6155, 2023.

[29] M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," Adv. Neural Inf. Process. Syst., vol. 30, pp. 4765–4774, 2017.

[30] A. Johnson et al., "Reproducibility in applied AI: Challenges and best practices," Nature Machine Intelligence, vol. 4, pp. 493–500, 2022.

[31] G. Montavon et al., "Layer-wise relevance propagation: An overview," J. Appl. AI, vol. 8, pp. 1–15, 2023.

[32] M. A. Ferreira and N. Silva, "Hybrid CNN–LSTM architectures for temporal anomaly detection," Eng. Appl. Artif. Intell., vol. 130, p. 107121, 2024.

[33] J. Han et al., "Temporal deep networks for sequence classification: A survey," ACM Comput. Surv., vol. 56, no. 4, pp. 1–43, 2024.

[34] A. Kumari, R. Ranjan, and V. Patel, "A CNN-LSTM hybrid for financial time-series fraud detection," IEEE Access, vol. 12, pp. 112347–112359, 2024.

[35] D. Tang et al., "Handling extreme class imbalance in transactional fraud detection with dynamic re-weighting," Information Fusion, vol. 97, p. 102239, 2025.

[36] Q. Chen, Y. Li, H. Wang, et al., "Development and Internal Multicenter Validation of a Deep Learning Model for Predicting Post-Hepatectomy Liver Failure in Hepatocellular Carcinoma," Cancers, vol. 18, no. 6, Art. no. 926, 2026.

[37] A. B. Gorgulu, H. A. Yilmaz, and H. Kose, "Domain-Adaptive Sentiment and Topic Analysis Using Transformer-Based Language Models," Turkish Journal of Electrical Engineering & Computer Sciences, vol. 34, no. 3, pp. 86868–86892, 2026.

[38] C. M. Bishop, Pattern Recognition and Machine Learning. New York, NY, USA: Springer, ISBN-10: 0-387-31073-8, 2006.

[39] T. Fawcett, "An Introduction to ROC Analysis," Pattern Recognition Letters, vol. 27, no. 8, pp. 861–874, 2006.

[40] J. Davis and M. Goadrich, "The Relationship Between Precision-Recall and ROC Curves," in Proc. 23rd Int. Conf. Machine Learning (ICML), Pittsburgh, PA, USA, pp. 233–240, doi: 10.1145/1143844.1143874, 2006.

[41] T. Saito and M. Rehmsmeier, "The Precision–Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets," PLoS ONE, vol. 10, no. 3, Art. no. e0118432, 2015.

[42] E. Qumsiyeh, J. H. Alkhateeb, M. R. A. Sara and K. Musa, "Credit Card Fraud Detection with Pyspark MLlib: A Big Data Classification Study," 2026 IEEE 15th International Conference on Communication Systems and Network Technologies (CSNT), Al-Khobar, Saudi Arabia, 2026, pp. 539–542.