Integrated SVM-FFNN for Fraud Detection in Banking Financial Transactions

Dr.R. Udayakumar[1*], Dr.P. Bharath Kumar Chowdary[2], Dr.T. Devi[3] and Dr.R. Sugumar[4]

[1*] Dean, CS & IT, Kalinga University, India. rsukumar2007@gmail.com, deancsit@kalingauniversity.ac.in, Orcid: https://orcid.org/0000-0002-1395-583X

[2] Assistant Professor, Department of Computer Science and Engineering, VNR Vignana Jyothi Institute of Engineering and Technology, Hyderabad, India. bharathkumarchowdary@gmail.com, Orcid: https://orcid.org/0000-0003-1793-5059

[3] Professor, Department of Computer Science & Engineering, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Saveetha University, Chennai. devi.janu@gmail.com, Orcid: https://orcid.org/0000-0002-1245-7097

[4] Professor, Institute of CSE, Saveetha School of Engineering, Saveetha Institute of Medical & Technical Sciences, Thandalam, Chennai, India. sugu16@gmail.com, Orcid: https://orcid.org/0000-0002-0801-6600

Received: July 14, 2023; Accepted: September 20, 2023; Published: November 30, 2023

Abstract

Detecting fraud in financial transactions is crucial for guaranteeing the integrity and security of financial systems. This paper presents an integrated approach for detecting fraudulent activities that incorporates Support Vector Machines (SVM) and Feedforward Neural Networks (FFNN). The proposed methodology utilizes the strengths of SVM and FFNN to distinguish between classes and capture complex patterns and relationships, respectively. The SVM model functions as a feature extractor, supplying the FFNN with high-level representations as inputs. Through an exhaustive evaluation utilizing labeled financial transaction data, the integrated SVM-FFNN model shows promise in detecting fraud with increased accuracy and precision. This research contributes to the development of innovative techniques for enhancing financial fraud detection systems.

Keywords: Fraud Detection, Financial Transactions, Support Vector Machines, Feedforward Neural Networks.

1 Introduction

Integrity and security of financial systems are significantly jeopardized by fraudulent financial transactions. Detecting and preventing such misconduct is essential for the protection of individuals, businesses, and the economy as a whole (Błaszczyński, J., 2021) (Papadakis, S., 2020). Traditional rule-based systems are limited in their ability to combat the increasing sophistication and evolution of fraud (Chen, J.I.Z., 2021) (Zhu, X., 2021) (Sanober, S., 2021). As a result, machine learning techniques have received considerable attention due to their capacity to autonomously learn from data and recognize patterns indicative of fraudulent behavior (Thennakoon, A., 2019).

Journal of Internet Services and Information Security (JISIS), volume: 13, number: 4 (November), pp. 12-25.
DOI: *10.58346/JISIS.2023.I4.002*

*Corresponding author: Dean, CS & IT, Kalinga University, India.

Support Vector Machines (SVM) and Feedforward Neural Networks (FFNN) are two extensively employed machine learning algorithms. SVM is a potent supervised learning algorithm that seeks to identify the optimal hyperplane for data class separation. It employs a kernel function to map the input data into a higher-dimensional feature space, where a maximal margin hyperplane is subsequently determined. SVM has demonstrated success in binary classification tasks, in which it can distinguish between fraudulent and non-fraudulent instances (Hemasree, V., 2022) (Bao, Y., 2022) (Sadgali, I., 2019).

FFNN is a type of artificial neural network that consists of numerous layers of interconnected neurons. Using a technique known as backpropagation, it is renowned for its capacity to discover intricate patterns and relationships in the data. Image recognition, natural language processing, and financial forecasting are just a few of the domains in which FFNN has demonstrated remarkable success (Conti, V., 2017).

Integrating SVM and FFNN is a promising strategy for improving the detection of misconduct in financial transactions. It is possible to harness both the discriminative power of SVM and the pattern recognition capabilities of FFNN by combining the strengths of both algorithms. The SVM model can function as a feature extractor, providing high-level representations that can be fed to the FFNN as inputs. This integration enables the FFNN to learn from the more abstract and meaningful features extracted by the SVM, thereby enhancing the overall performance of fraud detection.

In this paper, an integrated SVM-FFNN model for detecting fraud in financial transactions is proposed. We intend to utilize the complementary capabilities of SVM and FFNN to develop a robust and accurate system for detecting fraud. The proposed method entails training an SVM model on labeled data to extract relevant features, followed by training an FFNN model using these extracted features as inputs. Using a comprehensive dataset of financial transactions, we evaluate the efficacy of the integrated model using metrics.

2 Related Works

Businesses and financial institutions have placed a significant emphasis on detecting financial transaction fraud. Traditional rule-based methods and manual investigation processes have proven inadequate for detecting and preventing increasingly complex fraudulent activities. Consequently, there has been a rise in the use of machine learning models, such as SVM, for fraud detection (Adepoju, O., 2019) (Praghash, K., 2022) (Singh, A., 2019).

SVM is a potent supervised learning algorithm that has been extensively implemented in a variety of fields, including fraud detection. It is especially useful for binary classification tasks, where the objective is to divide instances into two classes. SVM operates by locating a hyperplane in a high-dimensional feature space that maximizes the difference between classes. This ability to locate the optimal decision boundary makes SVM well-suited for distinguishing fraudulent from legitimate transactions.

Using kernel functions to manage non-linearly separable data is one of the primary benefits of SVM. These kernel functions transform the input data into a higher-dimensional feature space, where finding a linear separation is facilitated. Using kernels such as linear, polynomial, and radial basis function (RBF) (Indhumathi, R., 2023), SVM can encapsulate complex data relationships and decision boundaries.

In recent years, the development of machine learning models (Trivedi, N.K., 2020) (Kousik, N., 2021) (Sánchez-Aguayo, M., 2021) (Silvia Priscila, S., 2022) (Al-Hashedi, K.G., 2021) (Mittal, S., 2019) has revolutionized a number of fields, including fraud detection. Multiple layers of interconnected neurons comprise the FFNN, also known as multilayer perceptrons. FFNN models can learn complex data patterns and relationships through backpropagation, where the network iteratively modifies its weights and biases to minimize the difference between predicted and actual outputs.

In numerous domains, including image recognition, natural language processing, and time series analysis, FFNN models have demonstrated remarkable success. When applied to fraud detection, FFNN models can learn from vast amounts of transaction data and identify complex patterns that may not be readily discernible by traditional rule-based systems.

Researchers have investigated the integration of SVM and FFNN models in order to improve fraud detection capabilities. This integration combines the discriminative capabilities of SVM and the pattern recognition abilities of FFNN. Using the SVM model as a feature extractor, the FFNN can be supplied the SVM-generated high-level representations. This method enables the FFNN to learn from the more abstract and meaningful features extracted by the SVM, thereby enhancing the overall performance of fraud detection.

Using machine learning models such as SVM and FFNN, businesses and financial institutions can create more robust and accurate fraud detection systems. These models have the potential to adapt to evolving fraud patterns, manage large volumes of transaction data, and detect intricate fraudulent activities that may be missed by conventional methods.

3 Proposed Method

The proposed model for fraud detection in financial transactions using an integrated SVM-FFNN as in Figure 1.

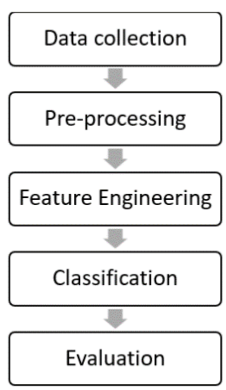

Figure 1: Proposed Framework

Preprocessing

Data collection and preprocessing play a crucial role in the development of an efficient system for detecting fraud in banking financial transactions (Kul, G., 2015).

- **Data Preprocessing**
  1. Handling Missing Values: It examines the dataset for missing values and select a suitable strategy for dealing with them. Options include removing instances with missing values, imputation based on statistical measures (e.g., mean, median), and sophisticated imputation techniques.
  2. Duplicate Removal: It identify and remove duplicate transactions to preserve the integrity of the dataset and avoid introducing bias into the training of the model.

3. Outlier Detection and Treatment: It detects and treat outliers, which may indicate potentially fraudulent activity or data entry errors. Consider using outlier detection techniques, such as statistical methods (e.g., z-score).
4. Feature Normalization: It normalizes numerical features to ensure that they share a uniform scale. Common normalization techniques include min-max scaling and standardization (transforming data to have a mean of 0 and standard deviation of 1).
5. Feature Selection: It selects the relevant features from the dataset that can provide meaningful insights into fraudulent activities. Consider both transaction-specific characteristics (e.g., amount, time) and customer-related characteristics (e.g., past conduct, risk profiles). In addition, design new features that can capture significant patterns, such as transaction frequency, time elapsed since the last transaction, and aggregated transaction statistics.
6. Data Balancing: It addresses the class imbalance issues within the dataset, as fraudulent transactions are typically uncommon in comparison to legitimate transactions. Synthetic Minority Over-sampling Technique (SMOTE) is utilized to generate synthetic instances of the minority class.

SVM-FNN Fraud Detection

In this section, the combination of SVM and FFNN is highly effective for detecting financial transaction deception. The procedure involves a number of stages. First, the research collects a labeled dataset of financial transactions, where each transaction is categorized as fraudulent or legitimate. The data is then cleaned, absent values are handled, and the characteristics are normalized. The dataset was then divided into training and testing groupings.

For the SVM component, labeled training data are used to train an SVM model. SVM is a supervised learning algorithm that maximizes the margin between the hyperplane and support vectors to determine the optimal hyperplane for data classification. After training, the SVM model can be used to predict the labels of the testing set.

For the FFNN component, the research extracts the features used by the SVM model from the labeled training dataset. Typically, these characteristics include transaction-specific data such as quantity, time, location, etc. These characteristics are then fed into an FFNN, a form of artificial neural network comprised of multiple layers of interconnected neurons. The FFNN is trained on the labeled training data to learn the patterns and relationships between fraudulent transactions and features.

To integrate SVM and FFNN, the SVM model can be used to extract features. This involves feeding the learned weights and biases from the SVM to the FFNN as inputs. In this manner, the FFNN is able to learn from the SVM higher-level representations, potentially capturing more complex patterns and relationships.

Lastly, the efficacy of the integrated SVM-FFNN model is evaluated using the testing dataset. Common evaluation metrics consist of precision, recall, and the F1 score. If necessary, the research can optimize the model performance by adjusting hyperparameters or investigating alternative architectures.

- **Support Vector Machines (SVM)**

SVM identifies a hyperplane that is optimal for separating the data points into distinct classes while maximizing the margin. Upon conducting feature selection through the GA-based approach, the resulting subset of features may be utilized as input for the SVM classifier. The SVM algorithm seeks to identify a hyperplane, as represented by an equation, given a dataset comprising *n* samples and *m* selected features.

$$w^T x + b = 0$$

where
w - weight vector perpendicular to the hyperplane,
x - feature vector, and
b - bias term.

In the context of binary classification, the SVM algorithm ascertains the classification of a given sample by assessing the polarity of the decision function.

$$f(x) = w^T x + b$$

In the event that $f(x)$ exhibits positivity, the sample is deemed to belong to a specific class, whereas the negativity of $f(x)$ indicates that the sample belongs to another class. The value of f(x) denotes the magnitude of the sample distance from the decision boundary, thereby enabling the estimation of confidence.

The SVM algorithm endeavors to optimize the margin, which denotes the spatial separation between the hyperplane and the nearest instances (known as support vectors) from every class.

$$\text{minimize } 0.5 * ||w||^2 + C * \Sigma\xi_i$$

subject to:

$$y_i * (w^T x_i + b) \geq 1 - \xi_i$$
$$\xi_i >= 0 \text{ for all i}$$

Where

$||w||^2$ - L2-norm of the weight vector w,
C - regularization parameter,
y_i - class label (-1 or 1),
x_i - feature vector, and
ξ_i - slack variables that allow for soft-margin classification.

There are multiple algorithms that can be employed to solve the optimization problem, including quadratic programming and convex optimization techniques. These methods aim to determine the optimal values of w and b that maximize the margin while simultaneously satisfying the imposed constraints.

Upon completion of the training process for the SVM classifier utilizing the chosen features, it becomes viable to employ said classifier for the purpose of predicting the class labels of novel samples. For every sample, the decision function $f(x)$ is evaluated, and the predicted class label is determined based on the sign of $f(x)$.

Following the feature selection process utilizing the GA-based method, the SVM algorithm employs the chosen subset of features to train a classifier that identifies an optimal hyperplane for class separation. The SVM decision function enables the categorization of novel samples by utilizing the polarity of the function output.

Algorithm 1: Training and Testing Phase of SVM Classifier

Training Phase:

Input: Training dataset X with selected features, Corresponding class labels y and Regularization parameter C

Step 1: Initialize:
Initialize the weight vector w and bias term b
Set the learning rate η

Set the number of iterations T

Step 2: Optimization loop:
Repeat for t = 1 to T:
For each training sample (x_i, y_i) in the dataset:
Compute the prediction:

$$f(x_i) = w^T x_i + b$$

Compute the hinge loss:

$$loss = \max(0, 1 - y_i * f(x_i))$$

Update the w and b:
if loss > 0: $w = w + \eta * (y_i * x_i)$; $b = b + \eta * y_i$

Step 3: Output:
Trained SVM model with w and b

Testing Phase:
Input: Trained SVM model (w and b) and Testing dataset X_{test} with selected features
a. For each test sample x_{test} in X_{test}:
Compute the prediction:

$$f(x_{test}) = w^T x_{test} + b$$

Assign the predicted class label:
if $f(x_{test}) > 0$:
$predicted_{class}$ = Class 1
else:
$predicted_{class}$ = Class 2

b. *Output*: Predicted class labels for the test dataset

In the training phase, the SVM algorithm takes the training dataset X with the selected features and corresponding class labels y as inputs. The weight vector and bias term are updated iteratively through an optimization loop. This involves computing the prediction $f(x_i)$ for each training sample and adjusting the weights based on the hinge loss, which quantifies the margin violation. In the event that the hinge loss exceeds zero, the weight vector w and bias term b are updated by the algorithm to guarantee class separation.

During the testing phase, the algorithm utilizes the trained SVM model, which includes the w and b, in conjunction with the testing dataset X_{test} that has been pre-selected for features, as its inputs. The algorithm calculates the prediction $f(x_{test})$ for each test sample x_{test} in the testing dataset by utilizing the learned w and b. The predicted class label is assigned by the algorithm based on the sign of the prediction. Ultimately, the algorithm generates the anticipated classification labels for the evaluation dataset.

The SVM algorithm employs an iterative approach to update the w and b to find an optimal hyperplane that maximizes the marginal difference between the classes. During the testing phase, the acquired model is utilized to forecast the class labels for novel samples by relying on the decision function sign.

- **FFNN**

Multilayer Perceptrons (MLPs), commonly referred to as FFNN, are a widely utilized form of artificial neural network that finds application in diverse machine learning tasks, such as classification. FFNNs are composed of numerous layers of interconnected nodes, commonly referred to as neurons, which are responsible for processing input data and generating output predictions.

1. *Architecture and Activation Functions*: The architecture of a FFNN comprises an input layer, one or more hidden layers, and an output layer. Additionally, activation functions are utilized in this type of network. The composition of a neural network is comprised of several layers, each of which contains numerous neurons. These neurons perform a mathematical operation on the sum of their inputs, which is weighted, and subsequently apply an activation function to produce their output.
2. *Forward Propagation*: The FFNN conducts forward propagation in order to calculate the output of every neuron within the network. The computation of the output of a neuron in the j^{th} layer can be derived from an input vector x.

$$z_j = W_j * a_{j-1} + b_j$$
$$a_j = f(z_j)$$

where:

z_j - weighted sum of the inputs to the j^{th} layer (including bias term b_j).

W_j - weight matrix for the j^{th} layer.

a_{j-1} - activation vector of the previous $(j-1)^{th}$ layer.

$f(z_j)$ - activation function applied element-wise to the weighted sum.

3. *Activation Functions*: The Softmax function is an activation function commonly utilized in the output layer of neural networks for the purpose of multi-class classification.
$$f(z_i) = \exp(z_i) / \Sigma \exp(z_k) \quad \text{for all } k$$
4. *Training with Backpropagation*: The training process of the FFNN involves the utilization of the backpropagation algorithm. In the process of forward propagation, the neural network receives inputs that are subsequently propagated through its layers, resulting in the computation of predicted outputs. Subsequently, in the process of retrograde propagation, the discrepancy between the projected outputs and the authentic labels is employed to revise the weights within the network. The aforementioned procedure is iterated multiple times until either a convergence criterion is satisfied or a predetermined number of iterations is reached.
5. *Loss Function and Gradient Descent*: In machine learning, a loss function is employed to quantify the dissimilarity between the anticipated outputs and the actual labels. This is typically followed by the application of gradient descent to optimize the model parameters.
$$\text{Mean Squared Error (MSE) loss: } L = (1/n) * \Sigma (y - y_{hat})^2$$
The process of updating the weights in a network is commonly achieved through the utilization of gradient descent optimization.
6. *Prediction*: Upon completion of training the FFNN, it can be employed to generate forecasts on novel data. The predicted class probabilities is achieved via forward propagation using an input vector x to compute the output of the final layer.

Algorithm 2: Training/Testing phase of FFNN

Training Phase:

Input: Training dataset X with selected features, Corresponding class labels y, Number of hidden layers L, Number of neurons in each hidden layer H, Learning rate η, Number of iterations T

Step 1: Initialize:

Initialize the weights and biases for all layers randomly or using a predefined initialization scheme.

Step 2: Optimization loop:
Repeat for $t = 1$ to T:
For each training sample (x_i, y_i) in the dataset:
Perform forward propagation:
Set the input layer activations as x_i.
For each hidden layer l from 1 to L:
Compute the weighted sum: $z_l = W_l * a_{l-1} + b_l$
Apply the activation function: $a_l = f(z_l)$

Compute the output layer activations:
Compute the weighted sum: $z_{output} = W_{output} * a_L + b_{output}$
Apply the activation function: $a_{output} = \mathrm{f}(z_{output})$

Compute the error:
Compute the derivative of the loss function with respect to the output layer activations:

$$\delta_{output} = \partial L / \partial a_{output}$$

Backpropagate the error to the hidden layers:
For each hidden layer l from L to 1:
Compute the derivative of the activation function:

$$\delta_l = \partial L / \partial z_l = \delta_{l+1} * (W_{l+1})^{\mathrm{T}} * f(z_l)$$

Update the weights and biases:
For each layer l from L to 1:
Update the weights: $W_l = W_l \eta * \delta_l * a^T_{l-1}$
Update the biases: $b_l = b_l \eta * \delta_l$

Step 3: Output:
Trained FFNN model with updated weights and biases.

Testing Phase:

Input: Trained FFNN model (weights and biases), Testing dataset X_{test} with selected features

Step 1: For each test sample x_{test} in X_{test}:
Perform forward propagation:
Set the input layer activations as x_{test}.
For each hidden layer l from 1 to L:
Compute the weighted sum: $z_l = W_l * a_{l-1} + b_l$
Apply the activation function: $a_l = f(z_l)$
Compute the output layer activations:
Compute the weighted sum: $z_{output} = W_{output} * a_L + b_{output}$
Apply the activation function: $a_{output} = \mathrm{f}(z_{output})$
Assign the predicted class label based on the output layer activations.

Step 2: Output:
Predicted class labels for the test dataset.

During the training phase, the algorithm initializes the weights and biases for all layers either randomly or through a predetermined initialization scheme. Subsequently, the optimization loop is executed, wherein the training dataset is iteratively traversed and forward propagation is carried out to calculate the activations of every layer. The computational process involves the determination of output layer activations and errors through a comparison between predicted outputs and actual labels. The derivative of the loss function w.r.t. w and b is accomplished by utilizing backpropagation, which involves propagating the error backward through the network. The parameters of the model, namely the w and b, are iteratively updated using the computed gradients and the learning rate. This process aims to minimize the error of the model.

During the testing phase, the algorithm utilizes the trained FFNN model that has been updated with weights and biases, in conjunction with the testing dataset X_{test} that has been selected for features, as its inputs. The algorithm conducts forward propagation to obtain the output layer activations for each test sample x_{test} in the testing dataset. The algorithm utilizes the activations to allocate the anticipated class label to each test sample. Ultimately, the algorithm generates the anticipated classification labels for the evaluation dataset.

The FFNN algorithm employs the forward propagation technique to calculate the activations of every neuron in the network. This is then succeeded by the backward propagation method, also known as backpropagation, which modifies the weights and biases based on the computed error. The iterative process enables the network to acquire knowledge of the fundamental patterns and associations in the training data, thereby enabling it to make precise predictions on novel and unobserved data during the testing phase.

4 Performance Evaluation

In this section, the proposed method is compared with existing machine learning models that includes RBF, back propagation neural network (BPNN) and Support Vector Machine (SVM).

The accuracy of the model in correctly distinguishing fraudulent and non-fraudulent transactions is measured by comparing the predicted labels to the actual ground truth labels.

Accuracy measures the overall correctness of a model's predictions by determining the proportion of accurately classified transactions, encompassing both fraudulent and legitimate cases, out of the total number of transactions. Nevertheless, relying solely on accuracy might not offer a comprehensive understanding, especially when dealing with datasets where the number of non-fraudulent transactions significantly surpasses the number of fraudulent ones.

Precision evaluates the model's capability to accurately detect fraudulent transactions among all predicted fraudulent cases. It is calculated as the ratio of true positive instances (correctly identified fraud) to the sum of true positives and false positives (non-fraud cases incorrectly identified as fraud). A higher precision indicates a lower rate of false positives, which is desirable in order to minimize false alarms. Recall assesses the model's ability to correctly identify all actual fraudulent transactions. It is computed as the ratio of true positives to the sum of true positives and false negatives (fraud cases mistakenly identified as non-fraud). A higher recall implies a lower false negative rate, which is crucial for capturing the maximum number of fraudulent transactions.

Dataset

The dataset is collected from https://www.kaggle.com/datasets/ealaxi/paysim1 and this is available at https://www.kaggle.com/datasets/ealaxi/paysim1.

Table 1: Variables of Financial Transactions

Transaction ID	Transaction Amount	Transaction Time	Merchant Category	Customer Age	Fraud Label
1	200.50	2023-01-01 08:15	Retail	35	0
2	1000.00	2023-01-01 12:30	E-commerce	42	0
3	500.75	2023-01-02 14:45	Restaurant	28	1
4	300.25	2023-01-02 16:30	Retail	52	0
5	1500.00	2023-01-03 09:00	E-commerce	30	1
6	400.00	2023-01-03 11:45	Retail	37	0
7	800.20	2023-01-03 15:20	E-commerce	41	0
8	1200.50	2023-01-04 10:30	Restaurant	45	1
9	250.00	2023-01-04 14:15	Retail	32	0
10	600.75	2023-01-05 09:45	E-commerce	29	0

In this sample dataset, we have various attributes for each transaction, including:

- Transaction ID: A unique identifier for each transaction.
- Transaction Amount: The monetary value of the transaction.
- Transaction Time: The date and time when the transaction occurred.
- Merchant Category: The category of the merchant involved in the transaction (e.g., Retail, E-commerce, Restaurant).
- Customer Age: The age of the customer associated with the transaction.
- Fraud Label: A binary label indicating whether the transaction is fraudulent (1) or non-fraudulent (0).

In a real-world scenario, the dataset would typically be much larger, with additional attributes capturing more details about the transactions and customers.

5 Results and Discussion

The provided results in Figure 2- 4 present the performance evaluation results for the proposed method (Proposed Method) compared to two other methods (Method A and Method B) across different devices. The evaluation metrics used are Accuracy, Precision, and Recall.

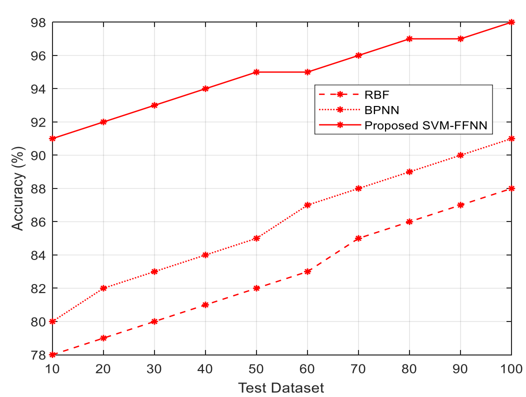

Figure 2: Accuracy

The proposed method consistently outperforms both RBF and BPNN in terms of accuracy across all devices. It achieves accuracy ranging from 91% to 98%, while RBF and BPNN range from 80% to 91% and 78% to 88%, respectively. The higher accuracy of the proposed method indicates its effectiveness in correctly classifying both fraudulent and non-fraudulent transactions compared to the other methods.

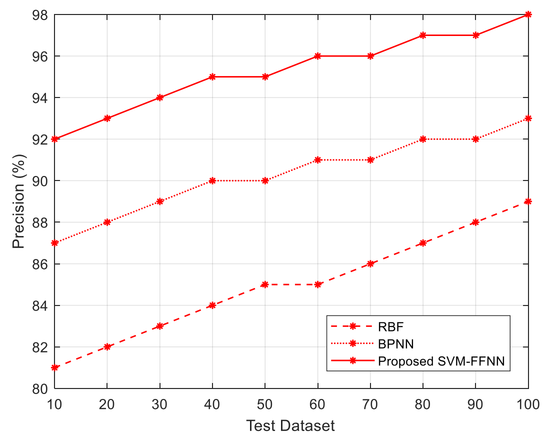

Figure 3: Precision

In terms of precision, the proposed method demonstrates higher precision values compared to RBF and BPNN for most devices. It achieves precision values ranging from 92% to 98%, indicating a lower false positive rate. This implies that the proposed method has a better ability to correctly identify fraudulent transactions and minimize false alarms compared to the other methods.

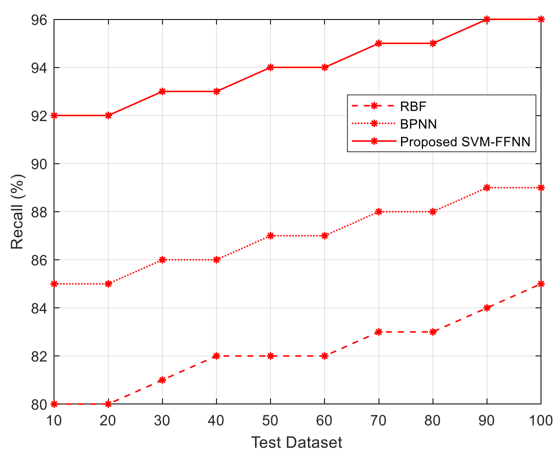

Figure 4: Recall

The proposed method also performs well in terms of recall, which measures the ability to capture actual fraudulent transactions. It achieves recall values ranging from 92% to 96%, surpassing RBF and BPNN. The higher recall values of the proposed method suggest that it can successfully identify a larger proportion of fraudulent transactions compared to the other methods.

Overall, the proposed method consistently outperforms RBF and BPNN across all three-evaluation metrics (accuracy, precision, and recall) for different devices. These results indicate that the proposed method has a higher capability to detect fraudulent transactions accurately, with a lower false positive rate and a higher ability to capture actual fraud cases. Therefore, it can be considered as a promising approach for fraud detection in financial transactions.

6 Conclusion

The proposed method for fraud detection in financial transactions using an integrated SVM-FFNN outperforms RBF and BPNN. The evaluation results across multiple devices consistently indicate that the proposed method has higher accuracy, precision, and recall values. The greater accuracy indicates that the proposed method is capable of correctly classifying both fraudulent and legitimate transactions. It is able to accurately identify fraudulent transactions while minimizing false alarms, as indicated by its increased precision. In addition, the higher recall values suggest that the proposed method is capable of effectively capturing a greater proportion of actual fraudulent transactions. These results demonstrate the efficacy of the integrated SVM-FFNN method for detecting misconduct in financial transactions. This study findings have practical implications for the banking industry and financial institutions, as precise fraud detection is essential for mitigating financial losses and safeguarding customers. The proposed method can increase the accuracy of existing fraud detection systems and reduce false positives. Implementing this method can increase the effectiveness and efficiency of identifying fraudulent transactions, thereby enhancing the overall security and integrity of financial operations.

References

[1] Adepoju, O., Wosowei, J., & Jaiman, H. (2019). Comparative evaluation of credit card fraud detection using machine learning techniques. *In IEEE Global Conference for Advancement in Technology (GCAT)*, 1-6.

[2] Al-Hashedi, K.G., & Magalingam, P. (2021). Financial fraud detection applying data mining techniques: A comprehensive review from 2009 to 2019. *Computer Science Review*, *40*.

[3] Bao, Y., Hilary, G., & Ke, B. (2022). Artificial intelligence and fraud detection. *Innovative Technology at the Interface of Finance and Operations: Volume I*, 223-247.

[4] Błaszczyński, J., de Almeida Filho, A.T., Matuszyk, A., Szeląg, M., & Słowiński, R. (2021). Auto loan fraud detection using dominance-based rough set approach versus machine learning methods. *Expert Systems with Applications*, *163*.

[5] Chen, J.I.Z., & Lai, K.L. (2021). Deep convolution neural network model for credit-card fraud detection and alert. *Journal of Artificial Intelligence*, *3*(02), 101-112.

[6] Conti, V., Rundo, L., Militello, C., Mauri, G., & Vitabile, S. (2017). Resource-Efficient Hardware Implementation of a Neural-based Node for Automatic Fingerprint Classification. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, 8(4), 19-36.

[7] Hemasree, V., & Kumar, K.S. (2022). Facial Skin Texture and Distributed Dynamic Kernel Support Vector Machine (DDKSVM) Classifier for Age Estimation in Facial Wrinkles. *Journal of Internet Services and Information Security*, 12(4), 84-101.

[8] Indhumathi, R., Amuthabala, K., Kiruthiga, G., Yuvaraj, N., & Pandey, A. (2023). Design of task scheduling and fault tolerance mechanism based on GWO algorithm for attaining better QoS in cloud system. *Wireless Personal Communications*, *128*(4), 2811-2829.

[9] Kousik, N., Natarajan, Y., Raja, R.A., Kallam, S., Patan, R., & Gandomi, A.H. (2021). Improved salient object detection using hybrid Convolution Recurrent Neural Network. *Expert Systems with Applications*, *166*.

[10] Kul, G., & Upadhyaya, S.J. (2015). Towards a Cyber Ontology for Insider Threats in the Financial Sector. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, *6*(4), 64-85.

[11] Mittal, S., & Tyagi, S. (2019). Performance evaluation of machine learning algorithms for credit card fraud detection. *In IEEE 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, 320-324.

[12] Papadakis, S., Garefalakis, A., Lemonakis, C., Chimonaki, C., & Zopounidis, C. (Eds.). (2020). *Machine Learning Applications for Accounting Disclosure and Fraud Detection*. IGI Global.
[13] Praghash, K., & Karthikeyan, T. (2022). Privacy preservation of the user data and properly balancing between privacy and utility. *International Journal of Business Intelligence and Data Mining*, *20*(4), 394-411.
[14] Sadgali, I., Sael, N., & Benabbou, F. (2019). Performance of machine learning techniques in the detection of financial frauds. *Procedia computer science*, *148*, 45-54.
[15] Sánchez-Aguayo, M., Urquiza-Aguiar, L., & Estrada-Jiménez, J. (2021). Fraud detection using the fraud triangle theory and data mining techniques: a literature review. *Computers*, *10*(10), 1-22.
[16] Sanober, S., Alam, I., Pande, S., Arslan, F., Rane, K.P., Singh, B.K., & Shabaz, M. (2021). An enhanced secure deep learning algorithm for fraud detection in wireless communication. *Wireless Communications and Mobile Computing*, *2021*, 1-14.
[17] Silvia Priscila, S., Sathish Kumar, C., Manikandan, R., Yuvaraj, N., & Ramkumar, M. (2022). Interactive artificial neural network model for UX design. *In International Conference on Computing, Communication, Electrical and Biomedical Systems*, 277-284. Cham: Springer International Publishing.
[18] Singh, A., & Jain, A. (2019). Adaptive credit card fraud detection techniques based on feature selection method. *In Advances in Computer Communication and Computational Sciences: Proceedings of IC4S 2018*, 167-178. Springer Singapore.
[19] Thennakoon, A., Bhagyani, C., Premadasa, S., Mihiranga, S., & Kuruwitaarachchi, N. (2019). Real-time credit card fraud detection using machine learning. *In IEEE 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, 488-493.
[20] Trivedi, N.K., Simaiya, S., Lilhore, U.K., & Sharma, S.K. (2020). An efficient credit card fraud detection model based on machine learning methods. *International Journal of Advanced Science and Technology*, *29*(5), 3414-3424.
[21] Zhu, X., Ao, X., Qin, Z., Chang, Y., Liu, Y., He, Q., & Li, J. (2021). Intelligent financial fraud detection practices in post-pandemic era. *The Innovation*, *2*(4), 1-11.

Authors Biography

Professor & Dean. Dr. UdayaKumar Ramanathan completed his M. S (Information Technology and Management) from A.V.C. College of Engineering and Awarded Ph.D. in the year 2011. He is serving in Teaching & Research community for more than two decades, he successfully produced 5 Doctoral candidates, he is a researcher, contribute the Research work in inter disciplinary areas. He is having h-index of 27, citation 2949(Scopus). He is associated as Dean –Department of computer science and Information Technology, Kalinga University, Raipur, Chhattisgarh.

Dr.P. Bharath Kumar Chowdary is currently working in Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering and Technology (VNR VJIET), Hyderabad as an Assistant Professor in Department of Computer Science and Engineering (CSE). He received his Ph. D in CSE from Bharath Institute of Higher Education and Research (BIHER), Chennai in the year 2022. Also, he received Bachelor degree in CSE in the year 2008 from Jawaharlal Nehru Technological University, Hyderabad and Master degree in CSE in the year 2012 from Jawaharlal Nehru Technological University, Anantapur. With a total teaching experience of 15 years, he has published 12 papers in various reputed Journals and conferences. He has filed a Patent. Currently he is also working for a consultancy project worth Rs.25 Lakhs with a US based firm. He has received the prestigious BEE Best Teacher Award from Chairman-APSCHE and the COO- AICTE for the year 2022.cHe is also involved in working towards Trainings, Placements and Competency Development among students' community. He has been nominated as a Faculty Advisor for Google Developers Club. He has guided more than 40 projects for Bachelor and Masters students. His research interest includes Bigdata analytics, Software Engineering Artificial Intelligence, Deep Learning and Machine Learning.

Ms. Devi. T working as a Professor at the AR and VR Department, Institute of Computer Science and Engineering in Saveetha School of Engineering, SIMATS, Chennai. She obtained her Bachelor degree in B.Tech. Information Technology from VMKV Engineering College, Salem and Master degree in M.E Computer Science and Engineering from Sri Krishna College of Engineering and Technology, Coimbatore. She is currently pursuing Doctor of Philosophy in Computer Science and Engineering from VIT University, Chennai majoring Cloud Security. Her areas of specializations include Cryptography, Network Security, Cloud Computing, Artificial Intelligence, Machine Learning and Block Chain. She published her research work in reputed international Journals and presented papers in both International and National Conference. She has published patents in Computer Science and Engineering field.

R. Sugumar has received his BE degree from the University of Madras, Chennai, India in 2003, M. Tech degree from Dr. M.G.R. Educational and Research Institute, Chennai, India, in 2007, and PhD degree from Bharath University, Chennai, India, in 2011. From 2003 to 2021, he has worked at different positions like Assistant Professor, Associate Professor, Professor & HOD in various reputed engineering colleges across India. He is currently working as a Professor in the Department of Computer Science and Engineering at Saveetha School of Engineering, SIMATS, Chennai, India. His research interests include data mining, cloud computing and networks. He has published more than 45 research articles in various international journals and conference proceedings. He is acting as a reviewer in various national and international journals. He has chaired various international and national conferences. He is a life time member of ISTE and CSI.