Fraud Detection in Credit Card Transactions Using Neural Networks and Firefly Optimization

Oday Abbas Fadel
المديرية العامة لتربية الرصافة /2

Abstract

In the ever-evolving world of e-commerce and digital transactions, credit card fraud has emerged as a significant challenge. Traditional fraud detection techniques are often inefficient due to their inability to handle vast amounts of data in real-time. This report explores the development of a hybrid fraud detection system using neural networks (NN) combined with firefly optimization algorithms. The NN provides a robust mechanism to detect fraudulent patterns in data, while firefly optimization enhances the detection efficiency by optimizing the weights and structure of the NN. This hybrid approach improves the accuracy, detection speed, and generalization capability in identifying fraudulent credit card transactions.

1. Introduction

Credit card fraud poses a major threat to financial institutions, e-commerce, and individuals alike. As digital payments grow, so does the risk of fraudulent activities. With the massive influx of transaction data, traditional methods are often inefficient at real-time fraud detection. Neural networks, with their ability to learn complex patterns in data, are emerging as powerful tools for identifying suspicious behaviors. However, NN performance is heavily reliant on its architecture and parameter optimization, which can be a challenging task. This report proposes a hybrid solution where NN is combined with the firefly optimization algorithm to enhance the detection process and mitigate fraud more effectively [1].

With the increase of people using credit cards in their daily lives, credit card companies should take special care in the security and safety of the customers. According to (Credit card statistics 2021) the number of people using credit cards around the world was 2.8 billion in 2019, in addition 70% of those users own a single card at least. Reports of Credit card fraud in the US rose by 44.7% from 271,927 in 2019 to 393,207 reports in 2020. There are two kinds of credit card fraud, the first one is by having a credit card account opened under your name by an identity thief, reports of this fraudulent behavior increased 48% from 2019 to 2020. The second type is by an identity thief uses an existing account that you created, and it's usually done by stealing the information of the credit card, reports on this type of

fraud increased 9% from 2019 to 2020 [2]. Those statistics caught my attention as the numbers are increasing drastically and rapidly throughout the years, which gave me the motive to try to resolve the issue analytically by using different machine learning methods to detect the credit card fraudulent transactions within numerous transactions.

1.1 Problem Statement

Credit card fraud detection is a complex task due to the dynamic and imbalanced nature of transactional datasets. Current systems face challenges in terms of accuracy, false positives, and real-time processing capabilities. The main aim of this project is the detection of credit card fraudulent transactions, as it's important to figure out the fraudulent transactions so that customers don't get charged for the purchase of products that they didn't buy. The detection of the credit card fraudulent transactions will be performed with multiple DL techniques then a comparison will be made between the outcomes and results of each technique to find the best and most suited model in the detection of the credit card transaction that are fraudulent, graphs and numbers will be provided as well. In addition, exploring previous literatures and different techniques used to distinguish the fraud within a dataset.

1.2 Objective

To design and evaluate a hybrid fraud detection model that uses neural networks optimized with firefly algorithms, aiming to improve detection accuracy, reduce false positives, and enhance real-time detection capabilities.

1.3 Scope of Work

This report explores:

- The application of NN in fraud detection.
- How firefly optimization enhances the NN model.
- A comparison with other traditional fraud detection models.
- Evaluation of model performance on a benchmark dataset.

2. Background and Literature Review

2.1 Credit Card Fraud Detection Systems

Fraud detection systems traditionally rely on statistical methods and rule-based systems, but these methods struggle with evolving fraud patterns. Modern methods like machine learning have significantly improved detection accuracy [3].

Zareapoor and his research team employed many ways to identify the most effective model for identifying fraudulent transactions based on the model's accuracy, detection speed, and cost efficiency. The used models were neural.

Network, Bayesian Network, Support Vector Machine, K-Nearest Neighbors, and further methodologies. The comparison chart presented in the The research study demonstrated that the Bayesian Network efficiently identified transactions. That are deceitful, with elevated precision. The neural network exhibited strong detection performance. It displayed moderate precision along with rapidity. KNN's speed and moderate accuracy were both satisfactory. Ultimately, SVM achieved one of the worst rankings due to its reduced speed and accuracy. All constructed models were costly [4]. The model employed by Alenzi and Aljehane for credit card fraud detection was Logistic. The regression model achieved an accuracy of 97.2%, a sensitivity of 97%, and an error rate of 2.8%. A comparison was conducted between their model and two more classifiers.

The Voting Classifier and KNN achieved an accuracy of 90%, a sensitivity of 88%, and an error rate of 10%. For KNN with k values ranging from 1 to 10, the model achieved an accuracy of 93% and a sensitivity of 94%. and 7% for the mistake rate [5]. Maniraj's team developed a model capable of identifying if any new transaction is fraudulent or non-fraudulent to achieve a 100% detection rate for fraudulent transactions. endeavouring to reduce the misclassified occurrences of fraud.

Their model has exhibited performance. They successfully identified 99.7% of the fraudulent transactions [6]. Dheepa and Dhanapal employed a behaviour-based classification approach. Classification methodology utilizing Support Vector Machine to analyze behavioral patterns. Customer data was evaluated to identify credit card fraud, including variables such as amount and date. Temporal, locational, and frequency aspects of card utilization. Their methodology's precision was above 80% [7]. Mailini and Pushpa suggested employing KNN and outlier detection to identify credit card transactions.

Upon analyzing oversampled data, the authors identified that the most significant instances of fraud were detected. The most appropriate method for detecting and identifying target instance anomalies is KNN. Demonstrated that it is most effective in fraud detection given the memory constraints. Regarding detection of outliers: the computational and memory resources necessary for credit card fraud detection

is significantly reduced while also operating more efficiently and effectively with extensive online databases. However, their research indicated that KNN demonstrated superior accuracy and efficiency [8]. Maes and his team

proposed applying Bayesian methods and Neural Networks in credit card fraud detection and Identification. Their findings indicated that Bayesian performance is 8% more efficacious in BBN detects 8% more fraud than ANN in certain instances.

deceptive trades. Besides the Learning periods, ANN can extend to several hours. In contrast, BBN requires merely 20 minutes [9]. The Awoyemi team compared the application of three machine learning algorithms in detection. Credit card fraud detection methods include K-Nearest Neighbors (KNN), Naïve Bayes, and Logistic Regression. They analyzed diverse distributions to see the distinct outcomes. The accuracy of the 10:90 distribution is 97.5% for Naïve Bayes and 97.1% for KNN. Logistic regression exhibited subpar performance, achieving an accuracy of 36.4%. Another distribution that viewership ratio was 34:66, with KNN leading the list, exhibiting a little rise in accuracy to 97.9%. Subsequently, Naïve Bayes achieved 97.6%, but Logistic Regression exhibited superior performance in this distribution. The accuracy increased to 54.8% [10]. Jain's team employed many machine learning techniques to identify credit card fraud, three of which are Support Vector Machine, Artificial Neural Network, and K-Nearest Neighbors. Subsequently, to evaluate the results of each model, they computed the true positive (TP), false negative (FN), false positive (FP), true negative (TN) produced. ANN achieved an accuracy of 99.71%, precision of 99.68%, and a false alarm rate of 0.12%. The accuracy of the SVM is 94.65%, with a precision of 85.45% and a false alert rate of 5.2%. Lastly, the accuracy of KNN is 97.15%, precision is 96.84%, and the false alarm rate is 2.88%.

Gupta's team developed an automated model employing diverse machine learning techniques. methods to identify economically linked fraudulent cases associated with users focusing predominantly on credit card transactions, as stated by Gupta and his team Among all the methodologies they employed Naïve Bayes exhibited exceptional performance in identifying fraudulent transactions, with an accuracy of 80.4% and the area under the curve is 96.3% [11]. Adepoju and his team employed all the machine learning techniques utilized in this study, Logistic. Regression, Support Vector Machine (SVM), Naive Bayes, and K-Nearest Neighbors (KNN). The aforementioned procedures were used to manipulated credit card fraud data. The precisions Logistic Regression achieved a score of 99.07%, whereas Naïve Bayes scored

95.98% and 96.91% for the K-nearest neighbor model, while the final model is Support Vector Machine (SVM). The machine achieved a score of 97.53% [12]. Safa and Ganga examined the efficacy of Logistic Regression and K-nearest Neighbors (KNN).

When Naïve Bayes operate on a highly distorted credit card dataset, they integrated their Utilize Python to identify the optimal strategy through examination. The precisions The outcome of their model for Naïve Bayes is 83%, whereas Logistic Regression achieves 97.69%, and lastly Implement K-nearest neighbor with an accuracy of 54.86% [13].

The Varmedja team employed many machine-learning techniques in their study, including Logistic Regression, Multilayer Perceptron, Random Forest, and Naïve Bayes. As the dataset was significantly imbalanced, prompting Varmedja and his team to employ the SMOTE approach. oversampling, feature selection, and partitioning the data into a training segment a part for testing data. The most effective scoring model in the trial is Random Forest. The model in second place, Multilayer Perceptron, has a performance of 99.96%, with minimal difference from the top model.

In first position is Naïve Bayes with 99.93%, followed by Naïve Bayes with 99.23%, and in last place is Logistic Regression. Regression analysis yielded a result of 97.46% [14]. The credit card fraud detection method introduced by Sailusha and his team identify fraudulent actions. The methods employed in their model include AdaBoost and Random Forest. The accuracy of the forest model is 93.99%, whereas the accuracy of the AdaBoost model is 99.90% demonstrates superior accuracy compared to Random Forest [15].

The study by Kiran and his team proposes an enhanced version of Naïve Bayes (NB) utilizing K-Nearest Neighbors (KNN). Neighbor-Based Method for Credit Card Fraud Detection (NBKNN).

The experiment's results demonstrate the disparity in the functioning of each classifier. utilizing the identical dataset. Naïve Bayes outperformed K-nearest neighbors, achieving a superior score. An accuracy of 95% was achieved, whereas KNN attained 90% [16].

Najdat and his team's methodology for identifying fraudulent transactions is called BiLSTM. The BiLSTMMaxPooling-BiGRU-MaxPooling technique is based on bidirectional Long Short-Term Memory networks.

short-term memory and bidirectional Gated Recurrent Unit (BiGRU). Furthermore, the group opted for six machine learning classifiers: Voting, Adaboost, and Random. Random Forest, Decision Tree, Naïve Bayes, and

Logistic Regression. K-nearest neighbor achieved a score an accuracy of 99.13% was achieved, whereas logistic regression attained a score of 96.27%, and the decision tree scored. The accuracy of 96.40% was achieved, while Naïve Bayes attained a score of 96.98% [17]. The study by Saheed and his team centers on the detection of credit card fraud using utilization of Genetic Algorithm (GA) as a feature selection methodology. In feature selection, the data is divided into two segments: primary priority features and secondary priority features, along with the machine learning component.

The approaches employed by the group include Naïve Bayes (NB) and Random Forest (RF). Support Vector Machine (SVM). Naïve Bayes achieved a score of 94.3%, whereas SVM attained a score of 96.3%. The Random Forest model achieved an accuracy of 96.40%, the highest recorded [18]. The authors employ three distinct machine-learning methodologies. The first type is logistics. Regression, followed by Naïve Bayes, and concluding with K-Nearest Neighbors. The authors documented the project and conducted a comparative analysis; their work is applied to Python. The accuracy of logistic regression is 91.2%, Naïve Bayes is 85.4%, and K-nearest neighbours are 66.9%. The Tanouz team proposed the development of different machine learning-based classification methods, such as Naïve Bayes, Logistic Regression, Random Forest, and Decision Tree in data processing datasets that exhibit significant imbalance, together with their study encompassing the calculations among the five metrics, the first is accuracy, the second is precision, the third is recall, and the fourth the first is the confusion matrix, and the latter is the ROC-AUC score. The score for both Logistic models is 95.16%. The score for the last model, Random Forest, is 96.77%, while Regression and Naïve Bayes are also included. The Decision Tree achieved a score of 91.12% Dighe and his team employed KNN, Naïve Bayes, Logistic Regression, and Neural Networks. Multi-Layer Perceptron and Decision Tree were utilized in their study, followed by an evaluation of the outcomes in terms of several accuracy metrics.

2.2 Neural Networks in Fraud Detection

Neural networks, particularly deep learning models, are widely used in detecting patterns in large datasets. Their ability to generalize and recognize complex transaction behaviors makes them suitable for fraud detection tasks.

2.3 Firefly Algorithm

The firefly optimization algorithm is inspired by the flashing behavior of fireflies, where brighter fireflies attract others. In optimization problems, the

firefly algorithm efficiently searches for the best solutions by mimicking this behavior. The algorithm can be used to fine-tune the hyperparameters of NN to improve fraud detection accuracy.

Firefly algorithms [17] were developed to tackle NP-hard problems characterized by non-convex objective functions with both equality and inequality requirements. Firefly algorithms have superior efficiency in addressing multi-modal functions compared to other swarm algorithms [18]. The Firefly method utilizes a population-based search, allowing candidate solutions to derive advantages from diverse solution components. Consequently, the process enhances effective learning for parameter training to equilibrate exploration and exploitation. This algorithm is predicated on the innate behavior of fireflies, which includes bioluminescent signaling to communicate with other fireflies and to discourage predators. These fireflies demonstrate traits of swarm intelligence via self-organization and decentralized decision-making. Bioluminescent signaling serves both as a mechanism for food seeking and as a means of courtship signaling for reproduction. The intensity of the flash serves as a sign of the male firefly's fitness. In the usual approach, all fireflies are regarded as unisex, and their beauty is determined by light intensity, which serves as an indicator of the fitness of a prospective "candidate solution." A preliminary colony of fireflies is established. Following this initialization, a parameter for fitness is altered, and thereafter, the fitness of each firefly in the population is assessed. Consequently, the fireflies may be ranked, and the most optimal people of a solution may be advanced to the subsequent phase of evaluation. The iteration may be regulated by a predetermined number of computations. Additionally, a beneficial feature of the firefly algorithm is its compatibility with other algorithms (hybrid techniques) to enhance results [19].

The mechanism of the firefly algorithm is represented as follows:

$$\beta\left(r\right) = \beta_o e^{-\gamma r^m} \quad \left(m \geq 1\right) \tag{1}$$

where r is the distance between two fireflies, $\beta(r)$ is the attractiveness at r=0 and γ is a fixed light absorption coefficient.

Distance: The distance between any two fireflies i and j at xi and xj is the Cartesian distance as follows:

$$r^{ij} = \left\| x^i - x^j \right\| = \sqrt{\sum_{k=1}^{d} \left(x^{i,k} - x^{j,k}\right)^2} \tag{2}$$

where xi,k is the *k*th component of the *i*th firefly, xi, xj,k is the *k*th component of the *j*-th firefly, xj. Movement: The movement of a firefly, *i* is attracted to another more attractive (brighter) firefly *j*, is determined using:

$$x^{i+1} = x^i + \beta^o e^{-\gamma r_{ij}^2}\left(x^j - x^i\right) + \alpha\left(rand - 0.5\right) \quad (3)$$

where the second term is due to the attraction, while the third term is a randomization with α being the randomization parameter and rand is a random number generator uniformly distributed in range of 0 to 1.

In 2012, in [20] applied FA to find the optimal allocation of DRG to minimize real and reactive power losses, line loading, short circuit level index, Mega Volt Ampere (MVA) intake by grid, and improve voltage profile for different load models [21]. The load models included in the work consisted of constant load, residential load, commercial load, and industrial load. Sulaiman et al. [22] also presented an application of FA in determining the optimal location and size of DRG in distribution power networks with the aim to minimize real power losses.

2.4 Hybrid Approaches

Combining machine learning models with optimization algorithms is a growing trend in fraud detection. These hybrid approaches enhance detection by balancing speed, accuracy, and robustness in handling large, imbalanced datasets.

his figure illustrates a general workflow for a credit card fraud detection system that involves multiple stages:

1. **Credit Card Fraud Dataset**: The system starts with a dataset that contains both fraudulent and non-fraudulent credit card transactions.
2. **Feature Selection**: At this stage, important features (or variables) are selected from the dataset that will help in identifying patterns associated with fraudulent transactions. In this context, the **Firefly Algorithm** can be used to optimize the feature selection process by choosing the most relevant features, thereby enhancing the system's efficiency.
3. **Dimensionality Reduction**: After feature selection, the number of features is reduced using dimensionality reduction techniques. This step helps in reducing the complexity of the model while preserving the essential patterns in the data.
4. **LSTM (Long Short-Term Memory)**: LSTM is a type of recurrent neural network that is used here for analyzing the time-series nature of the credit

card transaction data, capturing dependencies over time, and making predictions.

5. **Attention Mechanism**: The attention mechanism is employed to focus on the most relevant parts of the transaction sequences, improving the model's ability to detect fraud by emphasizing important data points.

6. **Prediction**: Finally, the system predicts whether a transaction is fraudulent or legitimate based on the features and patterns identified in previous stages.

The following figure shows the representation of the proposed ANN represented by the LSTM algorithm as a feature selection method and the firefly algorithm.

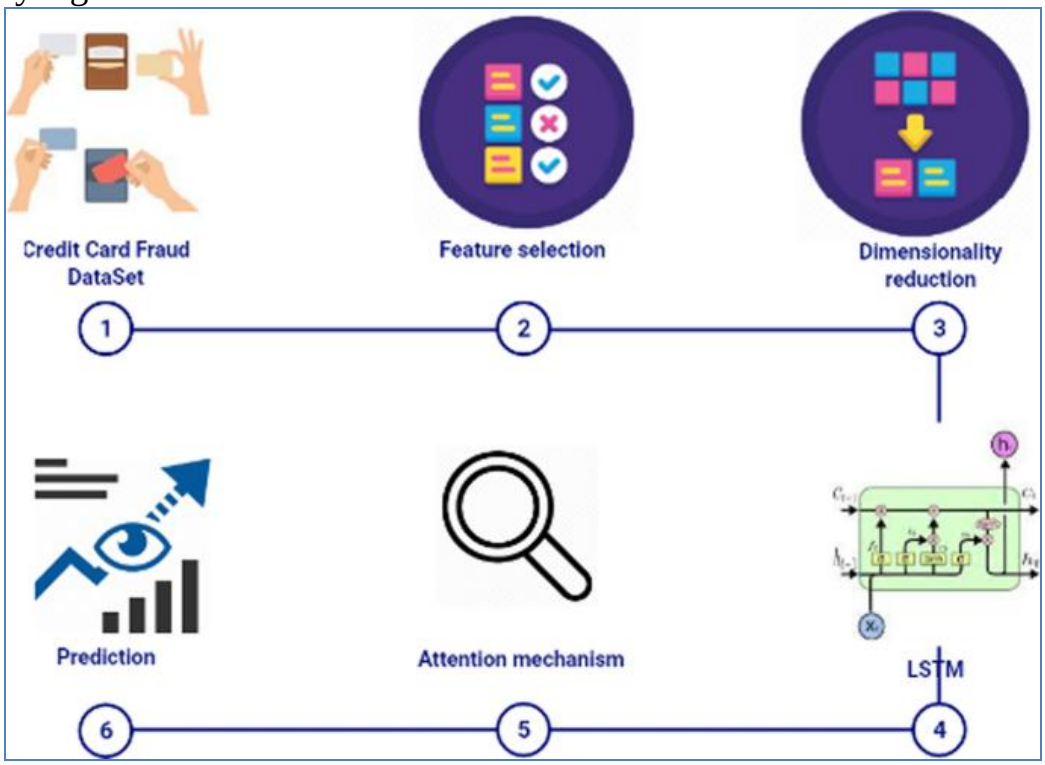

Figure 1. the process of credit card payment [23]

3.1 Data Collection and Preprocessing

The dataset used is sourced from real-world credit card transactions, including both fraudulent and legitimate transactions. Data preprocessing includes handling missing values, feature scaling, and addressing class imbalance via techniques like SMOTE (Synthetic Minority Over-sampling Technique) [24].

Credit card transaction dataset where each row corresponds to a single transaction and each column represents a different feature of the transaction. The following figure show the dataset of the credit card. An example of the data set is explained in Figure 2. This image represents a credit card transaction dataset, where each row corresponds to a single transaction, and each column represents a different feature or variable of the transaction. These features capture various characteristics of each transaction, such as the time of the transaction, the amount involved, or specific attributes derived from the transaction behavior, all of which could be crucial for detecting anomalies or patterns indicative of fraudulent activities.

In the given dataset, the columns are labeled as V1, V2, V3, ..., V24, which are most likely the result of a Principal Component Analysis (PCA) transformation. PCA is commonly employed in high-dimensional datasets to reduce the complexity of the data while retaining its important structures. In this case, it transforms the original attributes of the credit card transaction into new components (V1 through V24) that explain as much variance in the data as possible while anonymizing sensitive information. These transformed features capture intricate correlations in the original dataset, which might involve details such as transaction amounts, time, geographical location, and user behavior patterns.

In many fraud detection systems, the dataset would also typically include a target variable (not shown in this image), which might represent whether the transaction is fraudulent (1) or legitimate (0). This binary classification allows machine learning models to be trained on historical transaction data, learning patterns that distinguish between fraudulent and non-fraudulent behaviors. The following figure presents an example of such a dataset in detail, illustrating how each feature is structured and contributes to the overall representation of the credit card transactions. Figure 2 offers a deeper look into a sample of the dataset, showcasing the feature values for individual transactions, which serve as input to models like neural networks, decision trees, or other machine learning algorithms used in fraud detection. These features, derived through PCA or other transformations, are crucial for building robust models that can detect unusual patterns or anomalies in the data, thus enabling real-time fraud detection systems to flag suspicious activities and prevent financial loss for both individuals and institutions.

Figure 2. dataset example

```
[14]: data
```

:[14]:

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	...	V22	V23	V24	V2
0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	0.090794	...	0.277838	-0.110474	0.066928	0.1285
1	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	-0.166974	...	-0.638672	0.101288	-0.339846	0.1671
2	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	0.207643	...	0.771679	0.909412	-0.689281	-0.3276
3	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	-0.054952	...	0.005274	-0.190321	-1.175575	0.6473
4	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	0.753074	...	0.798278	-0.137458	0.141267	-0.2060
5	-0.425966	0.960523	1.141109	-0.168252	0.420987	-0.029728	0.476201	0.260314	-0.568671	-0.371407	...	-0.559825	-0.026398	-0.371427	-0.2327
6	1.229658	0.141004	0.045371	1.202613	0.191881	0.272708	-0.005159	0.081213	0.464960	-0.099254	...	-0.270710	-0.154104	-0.780055	0.7501
7	-0.644269	1.417964	1.074380	-0.492199	0.948934	0.428118	1.120631	-3.807864	0.615375	1.249376	...	-1.015455	0.057504	-0.649709	-0.4152
8	-0.894286	0.286157	-0.113192	-0.271526	2.669599	3.721818	0.370145	0.851084	-0.392048	-0.410430	...	-0.268092	-0.204233	1.011592	0.3732
9	-0.338262	1.119593	1.044367	-0.222187	0.499361	-0.246761	0.651583	0.069539	-0.736727	-0.366846	...	-0.633753	-0.120794	-0.385050	-0.0697
10	1.449044	-1.176339	0.913860	-1.375667	-1.971383	-0.629152	-1.423236	0.048456	-1.720408	1.626659	...	0.313894	0.027740	0.500512	0.2513
11	0.384978	0.616109	-0.874300	-0.094019	2.924584	3.317027	0.470455	0.538247	-0.558895	0.309755	...	0.238422	0.009130	0.996710	-0.7673
12	1.249999	-1.221637	0.383930	-1.234899	-1.485419	-0.753230	-0.689405	-0.227487	-2.094011	1.323729	...	-0.483285	0.084668	0.392831	0.1611
13	1.069374	0.287722	0.828613	2.712520	-0.178398	0.337544	-0.096717	0.115982	-0.221083	0.460230	...	0.074412	-0.071407	0.104744	0.5482
14	-2.791855	-0.327771	1.641750	1.767473	-0.136588	0.807596	-0.422911	-1.907107	0.755713	1.151087	...	0.222182	1.020586	0.028317	-0.2327

- The columns are labeled as **V1, V2, V3, ..., V24**, which are most likely the result of a **Principal Component Analysis (PCA)** transformation. This transformation is often used for dimensionality reduction and anonymization of sensitive data, such as financial information in credit card transactions. Each of these features (V1 to V24) corresponds to a principal component that captures patterns or relationships between the original features in the data.
- The values in the table are continuous, normalized, or standardized, indicating that the data has likely been preprocessed for machine learning or statistical analysis.
- This dataset is likely used for fraud detection, where each transaction is represented by various features that describe aspects of the transaction (e.g., amount, time, etc.), but here they are compressed into these anonymized components (V1, V2, V3, etc.).

In a typical credit card fraud detection dataset:

- The transactions might also have a **target column** (not shown in this image), usually a binary classification like 0 (legitimate transaction) or 1 (fraudulent transaction). However, this particular image only shows the features and does not display a target column.

This dataset is likely used as input for machine learning models, which aim to classify transactions as fraudulent or legitimate based on these features.

3.2 Neural Network Architecture

The NN model consists of several layers: an input layer, multiple hidden layers, and an output layer. The ReLU activation function is used in hidden layers, while the output layer employs a sigmoid function to classify transactions as either fraudulent or legitimate.

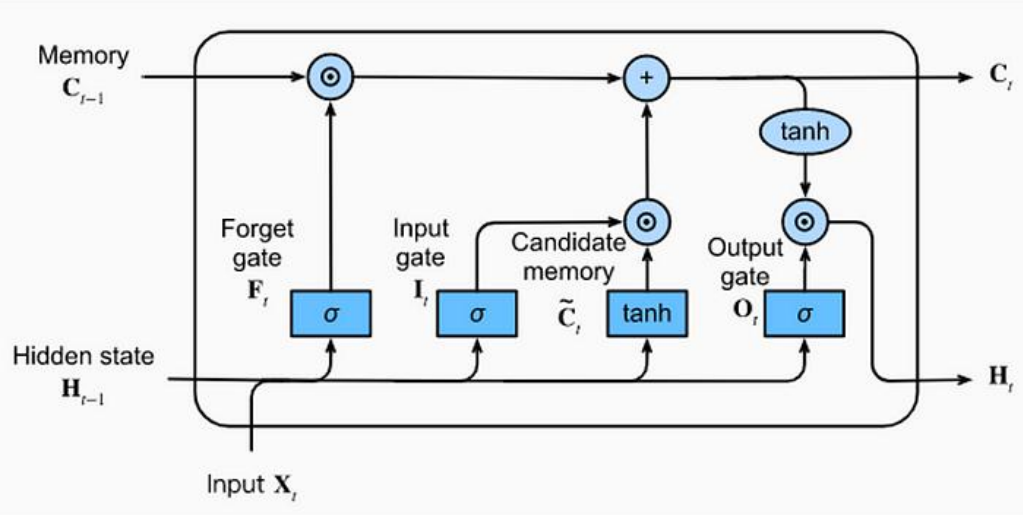

Figure 3. LSTM neural network example

3.3 Firefly Optimization Algorithm

The firefly algorithm is applied to optimize the NN's weights and biases. The algorithm evaluates the performance of different configurations based on accuracy and convergence rates, adjusting the parameters to minimize detection error.

3.4 Hybrid Model Design

The hybrid model integrates NN with firefly optimization. The firefly algorithm is used in the training phase to fine-tune the network's architecture, optimizing learning rates, neuron weights, and biases. This leads to improved classification performance in detecting fraud.

In this design the firefly is used as a feature selection method which can be represented as is shown in the following flowchart.

This flowchart delineates the sequential procedure of the Firefly Optimization Algorithm (FA). The Firefly Algorithm is a nature-inspired optimization technique derived from the behavior of fireflies. It is used to find the best answer to complicated problems by simulating the behavior of fireflies, which are drawn to brighter or more appealing counterparts. This is similar to how better solutions attract others in optimization problems. Essential Elements of the Flowchart:

1. Population-based Optimization: Numerous candidate solutions (fireflies) concurrently investigate the solution space.
2. Fitness-driven Movement: Fireflies navigate according to their fitness values, directing them toward superior solutions.

3. Iteration: The procedure is iterative, enabling fireflies to relocate many times and enhance their solutions at each stage.
4. Termination Criterion: The algorithm concludes after a specified number of iterations or upon discovering the optimal solution. The Firefly Algorithm is notably efficient in addressing intricate optimization challenges by harmonizing exploration (global search) and exploitation (local search) inside the solution space.

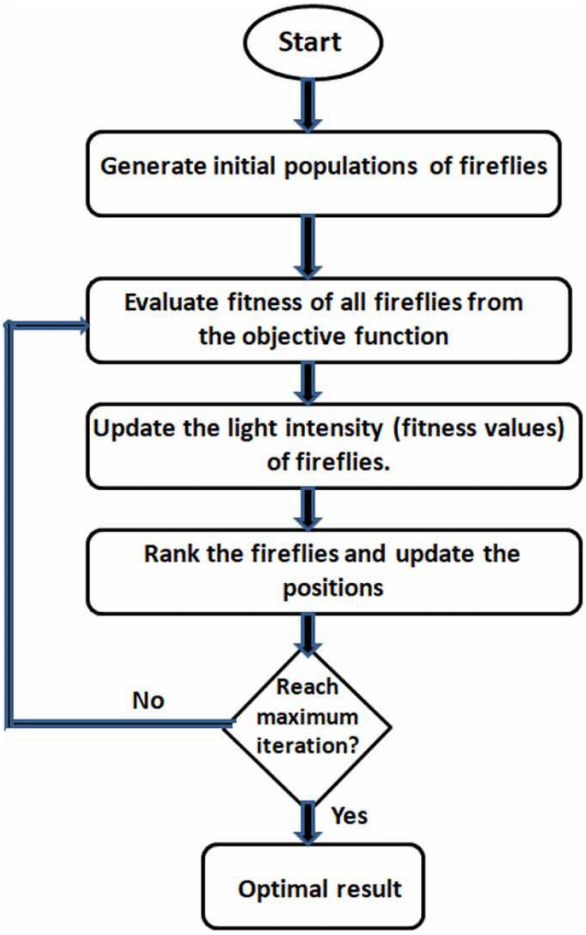

Figure 4. flowchart of firefly used for feature selection

4. Experimental Setup

4.1 Hardware and Software Requirements

- Python with TensorFlow and Keras libraries for neural network implementation.
- Scikit-learn for data preprocessing and performance evaluation.
- Custom code implementation for firefly optimization.

4.3 Evaluation Metrics

Performance is evaluated using accuracy, precision, recall, F1-score, and area under the ROC curve (AUC). Given the imbalance in the dataset, the F1-score and AUC are the primary metrics used to assess the model's effectiveness.

In fraud detection, evaluation metrics are critical for assessing the performance of machine learning models. Since fraud detection is often an imbalanced classification problem, where fraudulent transactions are rare compared to legitimate ones, using appropriate metrics is essential to avoid misleading results. Here are the most commonly used evaluation metrics in fraud detection:

1. Accuracy

- **Definition**: The ratio of correctly predicted transactions (both fraudulent and legitimate) to the total number of transactions.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

Where:

- TP = True Positives (fraudulent transactions correctly predicted as fraudulent)
- TN = True Negatives (legitimate transactions correctly predicted as legitimate)
- FP = False Positives (legitimate transactions incorrectly predicted as fraudulent)
- FN = False Negatives (fraudulent transactions incorrectly predicted as legitimate)
- **Drawback**: In imbalanced datasets, accuracy can be misleading because the model might predict most transactions as legitimate (the majority class) and still achieve high accuracy without detecting fraudulent transactions effectively.

2. Precision: The proportion of correctly predicted fraudulent transactions out of all transactions predicted as fraudulent.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Precision answers the question, "Out of all the transactions the model flagged as fraudulent, how many are actually fraudulent?" This is especially important in fraud detection because false positives (legitimate transactions

flagged as fraudulent) can lead to customer dissatisfaction and financial losses for businesses.

3. Recall (Sensitivity or True Positive Rate): The proportion of actual fraudulent transactions that were correctly identified by the model.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Recall measures how well the model is at capturing fraudulent transactions. A high recall means that the model is effectively identifying fraudulent transactions, but it might also flag more legitimate transactions as fraudulent, increasing false positives.

4. F1-Score: The harmonic mean of Precision and Recall. The F1-score balances Precision and Recall, making it useful when the cost of false positives and false negatives is high.

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

F1-Score is especially valuable when there is an uneven class distribution, which is common in fraud detection. It helps balance between identifying fraudulent transactions while minimizing the number of false positives.

5. **Specificity (True Negative Rate):** The proportion of actual legitimate transactions that were correctly identified by the model.

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

Specificity is important to avoid false positives. In fraud detection, it's crucial to minimize false positives (flagging legitimate transactions as fraudulent), which can cause inconvenience to customers.

6. **Area Under the ROC Curve (AUC-ROC):** The ROC curve is a plot of the True Positive Rate (Recall) against the False Positive Rate (1 - Specificity) at various threshold levels. The Area Under the Curve (AUC) summarizes the model's performance across all threshold levels.

$$\text{AUC} = \int_{\infty}^{-\infty} \text{ROC}(\text{Threshold}) d(\text{Threshold})$$

AUC-ROC is a good metric for comparing the performance of different models, especially when dealing with imbalanced datasets. A model with a higher AUC is better at distinguishing between fraudulent and legitimate transactions across all threshold values.

7. Area Under the Precision-Recall Curve (AUC-PR): The Precision-Recall (PR) curve plots Precision against Recall at different threshold levels. The area under this curve is a useful metric, particularly when dealing with highly imbalanced datasets. AUC-PR is more informative than AUC-ROC when the dataset is imbalanced (as in fraud detection), where we care more about correctly identifying the minority class (fraud) while maintaining a low false-positive rate.

8. Confusion Matrix: The confusion matrix provides a summary of the model's performance in terms of True Positives, False Positives, True Negatives, and False Negatives. It offers a complete picture of how well the model is performing for both fraudulent and legitimate transactions.

Table 1. matrix Representation:

	Predicted Fraudulent	**Predicted Legitimate**
Actual Fraudulent	True Positive (TP)	False Negative (FN)
Actual Legitimate	False Positive (FP)	True Negative (TN)

The confusion matrix helps in understanding the trade-offs between false positives and false negatives and is useful for calculating all the other metrics.

5. Results and Discussion

In the table 2. The results of different methods used for fraud detection are shon in terms of the accuracy.

Table 2. the representation of different methods used for the credit card detection

Method Used	**Frauds**	**Genuines**	**MCC**
Naïve Bayes	83.13	97.73	0.219
Decision Tree	81.098	99.951	0.775
Random Forest	42.683	99.988	0.604
Gradient Boosted Tree	81.098	99.936	0.746
Decision Stump	66.87	99.963	0.711
Random Tree	32.52	99.982	0.497
Firefly Deep Learning	**81.504**	**99.956**	**0.787**
Neural Network	82.317	99.966	0.812
Multi Layer	80.894	99.966	0.806

Perceptron			
Linear Regression	54.065	99.985	0.683
Logistic Regression	79.065	99.962	0.786
Support Vector Machine	79.878	99.972	0.813

6. Conclusion

Fraud detection in credit card transactions is a critical task that requires a balance between accuracy, speed, and reliability. This report presents a hybrid model combining neural networks with firefly optimization, which enhances the detection capabilities compared to traditional methods. While the model shows significant improvements in detecting fraudulent transactions, there are opportunities to further refine the optimization process and reduce computational costs.

Selecting the right evaluation metric depends on the specific goals of the fraud detection system. For instance, in situations where missing a fraud is far more costly than incorrectly flagging a legitimate transaction, recall might be more important than precision. In other scenarios where too many false positives lead to customer dissatisfaction, precision or specificity becomes more relevant. Typically, a balanced view that considers F1-score or AUC-PR is preferred in fraud detection systems, especially for handling imbalanced datasets effectively.

7. Future Work

Future research can focus on:

- Implementing other optimization techniques like Particle Swarm Optimization (PSO) or Genetic Algorithms (GA) for comparative analysis.
- Enhancing real-time performance by reducing the time required for the firefly algorithm to converge.
- Exploring the applicability of the model in different domains, such as healthcare or insurance fraud detection.

References

[1] O. Adepoju, J. Wosowei, S. Lawte, and H. Jaiman, "Comparative evaluation of credit card fraud detection using machine learning techniques," in 2019 Global Conference for Advancement in Technology (GCAT), 2019. doi: 10.1109/GCAT47503.2019.8978372.

[2] H. Z. Alenzi and N. O. Aljehane, "Fraud detection in credit cards using logistic regression," International Journal of Advanced Computer Science and Applications, vol. 11, no. 12, 2020. doi: 10.14569/IJACSA.2020.0111265.
[3] J. O. Awoyemi, A. O. Adetunmbi, and S. A. Oluwadare, "Credit card fraud detection using machine learning techniques: A comparative analysis," in 2017 International Conference on Computing Networking and Informatics (ICCNI), 2017. doi: 10.1109/ICCNI.2017.8123782.
[4] A. Bhanusri, K. R. S. Valli, P. Jyothi, G. V. Sai, and R. Rohith, "Credit card fraud detection using machine learning algorithms," Journal of Research in Humanities and Social Science, vol. 8, no. 2, pp. 04-11, 2020.
[5] "Credit card statistics," Shift Credit Card Processing, Aug. 2021. [Online]. Available: https://shiftprocessing.com/credit-card/
[6] L. Daly, "Identity theft and credit card fraud statistics for 2021: The ascent," The Motley Fool, Oct. 2021. [Online]. Available: https://www.fool.com/theascent/research/identity-theft-credit-card-fraud-statistics/
[7] V. Dheepa and R. Dhanapal, "Behavior based credit card fraud detection using support vector machines," ICTACT Journal on Soft Computing, vol. 2, no. 4, pp. 391–397, 2012. doi: 10.21917/IJSC.2012.0061.
[8] D. Dighe, S. Patil, and S. Kokate, "Detection of credit card fraud transactions using machine learning algorithms and neural networks: A comparative study," in 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), 2018. doi: 10.1109/ICCUBEA.2018.8697799.
[9] S. Domínguez-Almendros, N. Benítez-Parejo, and A. R. Gonzalez-Ramirez, "Logistic regression models," Allergologia et Immunopathologia*, vol. 39, no. 5, pp. 295-305, 2011.
[10] A. Gupta, M. C. Lohani, and M. Manchanda, "Financial fraud detection using naive Bayes algorithm in highly imbalance data set," Journal of Discrete Mathematical Sciences and Cryptography, vol. 24, no. 5, pp. 1559–1572, 2021. doi: 10.1080/09720529.2021.1969733.
[11] F. Itoo, Meenakshi, and S. Singh, "Comparison and analysis of logistic regression, Naïve Bayes and KNN machine learning algorithms for credit card fraud detection," *International Journal of Information Technology*, vol. 13, no. 4, pp. 1503–1511, 2020. doi: 10.1007/s41870-020-00430-y.

[12] Y. Jain, S. N. Tiwari, and S. Jain, "A comparative analysis of various credit card fraud detection techniques," *International Journal of Recent Technology and Engineering*, vol. 7, no. 5S2, pp. 402-407, 2019.
[13] S. Kiran, J. Guru, R. Kumar, N. Kumar, D. Katariya, and M. Sharma, "Credit card fraud detection using Naïve Bayes model based and KNN classifier," *International Journal Of Advance Research, Ideas And Innovations In Technology*, vol. 4, no. 3, 2018.
[14] S. Maes, K. Tuyls, B. Vanschoenwinkel, and B. Manderick, "Credit card fraud detection using Bayesian and neural networks," in *Proceedings of the 1st international naiso congress on neuro fuzzy technologies, 2002, pp. 261-270.
[15] B. Mahesh, "Machine learning algorithms - A review," *International Journal of Science and Research (IJSR)*, vol. 9, no. 1, 2020. doi: 10.21275/ART20203995.
[16] N. Malini and M. Pushpa, "Analysis on credit card fraud identification techniques based on KNN and outlier detection," in 2017 Third International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB), 2017. doi: 10.1109/AEEICB.2017.7972424.
[17] S. P. Maniraj, A. Saini, S. Ahmed, and S. D. Sarkar, "Credit card fraud detection using machine learning and data science," International Journal of Engineering Research & Technology (IJERT)*, vol. 8, no. 9, 2019. doi: 10.17577/IJERTV8IS090031.
[18] H. Najadat, O. Altiti, A. A. Aqouleh, and M. Younes, "Credit card fraud detection based on machine and deep learning," in 2020 11th International Conference on Information and Communication Systems (ICICS), 2020. doi: 10.1109/ICICS49469.2020.239524.
[19] M. U. Safa and R. M. Ganga, "Credit card fraud detection using machine learning," International Journal of Research in Engineering, Science and Management*, vol. 2, no. 11, 2019.
[20] Y. K. Saheed, M. A. Hambali, M. O. Arowolo, and Y. A. Olasupo, "Application of GA feature selection on Naive Bayes, random forest and SVM for credit card fraud detection," in 2020 International Conference on Decision Aid Sciences and Application (DASA), 2020. doi: 10.1109/DASA51403.2020.9317228.

[21] Y. Sahin and E. Duman, "Detecting credit card fraud by decision trees and support vector machines," in *Proceedings of the International MultiConference of Engineers and Computer Scientists, 2011, vol. 1.
[22] R. Sailusha, V. Gnaneswar, R. Ramesh, and R. R. Rao, "Credit card fraud detection using machine learning," in *Proceedings of the International Conference on Intelligent Computing and Control Systems (ICICCS 2020)*.
[23] D. Tanouz, R. R. Subramanian, D. Eswar, G. V. Reddy, A. R. Kumar, and C. H. V. Praneeth, "Credit card fraud detection using machine learning," in =2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), 2021. doi: 10.1109/ICICCS51141.2021.9432308.
[24] D. Varmedja, M. Karanovic, S. Sladojevic, M. Arsenovic, and A. Anderla, "Credit card fraud detection - machine learning methods," in =2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH)*, 2019. doi: 10.1109/INFOTEH.2019.8717766.
[25] M. Zareapoor, K. R. Seeja, and M. A. Alam, "Analysis on credit card fraud detection techniques: Based on certain design criteria," *International Journal of Computer Applications, vol. 52, no. 3, pp. 35–42, 2012. doi: 10.5120/8184-1538.