IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 29 August 2024, accepted 6 September 2024, date of publication 12 September 2024, date of current version 25 September 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3458944

RESEARCH ARTICLE

OptDevNet: A Optimized Deep Event-Based Network Framework for Credit Card Fraud Detection

MUHAMMAD ADIL[1], (Senior Member, IEEE), ZHANG YINJUN[2], (Member, IEEE), MONA M. JAMJOOM[3], (Member, IEEE), AND ZAHID ULLAH[4], (Member, IEEE)

[1]Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY 14260, USA
[2]Department of Physics and Information Engineering, Guangxi Science and Technology Normal University, Laibin 546100, China
[3]Department of Computer Sciences, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University, Riyadh 11564, Saudi Arabia
[4]Department of Information Systems, College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University, Riyadh 11432, Saudi Arabia

Corresponding author: Zhang Yinjun (zhangyinjun@gxstnu.edu.cn)

This work was supported by Princess Nourah Bint Abdulrahman University, Riyadh, Saudi Arabia, through the Princess Nourah Bint Abdulrahman University Researchers Supporting Project under Grant (PNURSP2024R104).

ABSTRACT In recent times, credit card fraud has emerged as a substantial financial challenge for both cardholders and the issuing authorities. To address this demanding issue, researchers have employed machine learning techniques to identify fraudulent activities within labeled transaction records. However, these techniques have primarily been evaluated on limited or specific datasets, which may not adequately represent the broader real-world scenario. These limitations motivated us to comprehensively assess the existing machine learning classifiers and propose an Optimized Deep Event-based Network (OptDevNet) framework capable of addressing these challenges. To evaluate the performance of the proposed model, we implemented and assessed five different machine learning classifiers using the well-known Credit Card Fraud Detection (CCFD) Dataset. Upon careful analysis, we found that our model surpasses these classifiers in terms of fraudulent transaction detection accuracy. Given these findings, we are confident that our proposed model has the potential for effective real-world deployment in detecting and preventing malicious transactions.

INDEX TERMS Credit card fraud detection, OptDevNet framework, automatic fraud detection, malicious transactions, credit card security.

I. INTRODUCTION

In the rapidly evolving financial sector, machine learning techniques have emerged as a game-changer, particularly in credit risk assessment and fraud detection. The rise of fintech and the availability of vast amounts of data have paved the way for leveraging advanced algorithms to tackle complex financial challenges that were once seen as too difficult to overcome [1]. At the forefront of this revolution is credit risk assessment, where machine learning models can analyze intricate data patterns and extract valuable insights, enabling more accurate risk profiling and credit scoring. Techniques like extremely randomized trees (XRT), support vector machines (SVM), and deep neural networks (DNN) have demonstrated superior performance compared to traditional statistical methods, ushering in a new era of precision and efficiency in lending decisions [2]. Moreover, machine learning has proven to be an invaluable ally in the battle against fraudulent credit card transactions. Convolutional neural networks, for instance, can effectively learn and identify fraudulent patterns of transactions in large datasets. Therefore, these models often outperform traditional systems, which may struggle to keep pace with the ever-evolving strategies of cybercriminals [3]. Their ability to continuously learn and adapt to changing situations gives them an advantage over traditional techniques, ensuring that financial institutions remain vigilant and proactive in protecting their customers' assets [4].

The associate editor coordinating the review of this manuscript and approving it for publication was Parul Garg.

However, using machine learning algorithms in finance sector creates many problems. Some of these concerns revolve around the interpretability of systems and applications, which is followed by some additional challenges such as biases and regulatory standards, etc. Therefore, addressing these issues is imperative to guarantee the conscientious and ethical implementation of these algorithms in technological advancements [5], [6]. Nonetheless, the potential benefits of these algorithms—such as improved risk management, enhanced fraud detection, and optimized lending decision processes—make them an indispensable tool for the future financial sector. As fintech continues to disrupt traditional finance, the synergy between cutting-edge machine learning techniques and traditional finance increases, helping to secure the financial sector. The deployment of new algorithms enables more secure, efficient, and reliable financial systems worldwide. This symbiotic relationship will not only strengthen the financial sector but also foster economic growth and stability, benefiting individuals and businesses [7], [8].

However, machine learning (ML), deep reinforcement learning (DRL), deep learning (DL), and transfer learning (TL) algorithms face challenges related to the occurrence of 'false negatives' (FN) and 'false positives' (FP) [9], [10]. False negatives occur when the algorithm fails to detect fraudulent transactions, which can negatively impact an organization's revenue. On the other hand, false positives happen when a legitimate customer or user transaction is mistakenly flagged as fraudulent or illegitimate, resulting in customer dissatisfaction, loss of trust, and financial losses. Notably, as indicated in [11], FRISS has pinpointed that false positives (FPs) represent one of the most formidable challenges in detecting fraud in online transactions. This challenge is particularly relevant in the financial sector, where the adoption of machine learning techniques for credit risk assessment and fraud detection is gaining traction. While machine learning models have demonstrated superior performance compared to traditional statistical methods in credit risk profiling and credit scoring, the issue of false positives remains a critical consideration. Inaccurately flagging legitimate transactions as fraudulent can erode customer trust and loyalty, ultimately undermining the legitimacy of financial institutions that aim to safeguard their clients' assets.

In the domain of fraud detection, the prowess of machine learning techniques like CNN has shown valuable results in the identification of frauds. However, their usefulness diminishes if they generate too many false alerts, leading to unnecessary investigations and potentially alienating customers. Tackling the issue of false positives is paramount for the conscientious and ethical application of these innovations in the financial sector. Credit card fraud happens when someone uses a credit card without permission of the owner to make an unauthorized transaction. It is a common form of financial deception where individuals misuse someone else's card without their knowledge/consent [12]. Financial organizations continually seek novel strategies to combat fraud by considering the potential risks of fraudulent transactions. However, fraudsters are also working to exploit newly adopted technologies using innovative techniques. This ongoing challenge necessitates the development of highly efficient and effective algorithms [13]. In this scenario, the utilization of machine learning algorithms play a vital role to identifying fraudulent transactions. However, as discussed earlier, the occurrence of false negatives and false positives poses a significant challenges to these algorithms.

To address these challenges, we evaluate various machine learning algorithms using the well-known "CCFD" dataset to assess their performance. Furthermore, we propose an Optimized Deep Event-based Network (OptDevNet) to mitigate the issues of false negatives and false positives. The main highlights of this work are summarized as follows:

1) We start by conducting a comprehensive performance evaluation of various machine learning algorithms on the well-known Credit Card Fraud Detection (CCFD) dataset. This rigorous evaluation helps and provides valuable insights about the efficiency of these algorithms in detecting fraudulent credit card transactions.
2) Subsequently, we propose an Optimized Deep Event-based Network (OptDevNet) framework designed to address the limitations of existing algorithms in online credit card fraud detection. Furthermore, the proposed model enhances the accuracy and efficiency of fraudulent transaction identification by harnessing the capabilities of optimized event-based deep neural network architectures.
3) We also assess how well the proposed model can reduce false positives and false negatives. This evaluation helps determine how effectively the model identifies fraud, which is important for increasing customer satisfaction and revenue.
4) Through rigorous evaluation, OptDevNet demonstrates significant improvements in fraud detection accuracy, ultimately benefiting financial organizations by mitigating financial losses and enhancing customer satisfaction.
5) Finally, we comprehensively assess the effectiveness of our proposed model against existing machine learning models using various comparative metrics, demonstrating its superior performance and robustness in CCFD.

A. PAPER STRUCTURE

The remainder of the paper is structured as follows: Section II discusses the existing state-of-the-art schemes that have been employed in recent years to counter fraudulent transactions. Section III is dedicated to the discussion of our proposed model, while Section IV contains the comparative results and statistics of the proposed model, along with rival algorithms. Finally, Section V provides a comprehensive summary for the work and concludes the paper.

II. RELATED WORK

In the recent times, the application of data mining and machine learning techniques has shown significant advancements in detecting and preventing fraudulent credit card transactions. Credit card fraud involves deceptive practices aimed at achieving unauthorized financial gains by stealing money from individuals or institutions. Traditional methods for credit card fraud detection typically entail manual procedures conducted by audit teams, a process that can be notably time-consuming when dealing with evidence pertaining to embezzlement. In [14], the authors discussed comprehensively the state-of-the-art research that follow these kind of techniques. This review categorizes different studies based on various fraud typologies and data mining methodologies to identify potential instances of fraudulent activities. To improve this process, alternative technologies such as machine learning (ML) and deep learning strategies have been introduced. Notably, the recent study conducted in [15], explored the advancements in fraud detection. The authors used the Kitchenham method to scrutinize the onging research on fraud detection by evaluating a decade literature. In [16], the authors propose another research initiative that sought to conduct an exhaustive literature review of current SoA approaches used in CCFD.

The authors in [17] combined a genetic algorithm with a neural network to predict the likelihood of fraudulent transactions. Building on this, the study in [18] introduced a data analytics-based approach for credit card fraud detection (CCFD). Similarly, the research in [19] highlighted the importance of data analytics and artificial intelligence, focusing on their potential effectiveness in future internal audits. Further advancing the field, the study in [20] examined various data mining techniques and their use in fraud detection. These researchers created profiles of suspicious activities and used software that could learn rules to spot signs of fraud in a large database of transactions.

In [21], the authors utilized a apriori algorithm to analyze the patterns of new transactions in relation to existing ones for each customer. They meticulously crafted both fraud and legitimate transaction patterns for individual customers to ensure the safe operation of the credit card industry. Throughout this process, they employed various steps such as pattern creation and transaction categorization, to separate transactions into two pools: legal and illegal transactions. In [22], the authors conducted experiments on a dataset both before and after pre-processing with six classifiers. Their findings revealed a remarkable improvement in results when they implemented the undersampling technique with the dataset. For evaluation of the classifiers' performance, they used precision and recall metrics and showed valuable results. In [23], the authors assess the outcomes of various algorithms, including Logistic Regression, Apriori Algorithm, Naïve Bayes, Deep Learning, Decision Trees, SVM, K-Nearest Neighbor, and Neural Network models. Specifically, they focus on Chebyshev Functional Link Artificial Neural Networks (CFLANN) and MLP. This evaluation is conducted using the same dataset to lay-down the foundation for futuristic research. In [24], the authors used SVM in coordination with the Radial Basis Function (RBF) to detect fraudulant transactions with improved efficiency. In [25], the authors introduced an advanced approach to detect CCF in online transactions. They integrated multiple classification methods using a stacking ensemble model to improve efficiency. Additionally, they utilized the Sequential Minimal Optimization and Fuzzy-Rough Nearest Neighbor as base classifiers, along with LR to achieve better detection results.

Considering the technological advancements, researchers have underscored the inherent challenges in credit card industry. They have underlined that one of these challenges is the difficulty in differentiating between deprenciate and legitimate credit card transactions, as both exhibit similar patterns. To address this challenge, major credit card companies, including Paypal, MasterCard, Debit-Card, Visa, and American Express, have effectively employed artificial intelligence (AI) to detect fraudulent or deceptive patterns transactions, and mitigate the risks associated with CC fradulent transactions [26]. However, it is worth noting that the understanding and utilization of AI in the realm of credit card fraud detection still face limitations, mainly due to the diverse range of techniques and algorithms. Furthermore, machine learning and deep learning approaches have demonstrated their ability to learn from historical fraud patterns and insights provided by domain experts, such as forensic accountants, to proactively prevent similar fraudulent activities in the future [27]. This characteristic empowers disruptive technology to efficiently and effectively process extensive datasets, ultimately leading to the development of sophisticated algorithms.

III. PROPOSED MODEL

In this section, we examine the execution process of the proposed OptDevNet model and its operation within the context of the undermentioned problem. Before jumping into the operational process, we have added Figure 1 in the paper to provide a visual overview of holistic structure of the proposed model.

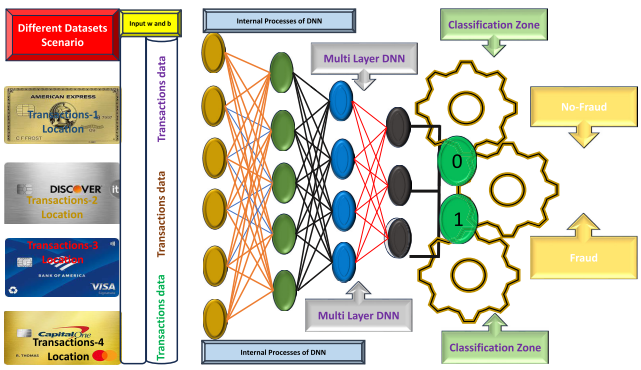

FIGURE 1. Overall Structure diagram of the proposed model.

Problem Formulation: Most of the existing credit card fraud detection techniques use a two-stage process. Initially, they need new data representations and then apply fraud indices to these representations to compute fraud scores. Given that, these techniques require a large number of labeled data, which most of the time is computationally intensive. To address these limitations, we aim in this work to simplify the process by directly using a minimal set of labeled data to calculate fraud scores.

To determine the minimal labeled dataset for the OptDevNet model, we use an iterative process of active learning. Initially, we started with a small set of labeled fraudulent transactions (K) and gradually increased it while monitoring the model's performance. The optimal size of K is determined when the model's accuracy stabilizes or when the improvement in performance becomes marginal with additional labeled data.

Moreover, our problem statement focuses on the optimization of the fraud detection process within a given dataset $X = x_1, x_2, x_3, x_4, x_5, \ldots, x_{N+K}$, where each $x_i \in \mathbb{R}^D$ represents a transaction with D features. In addition, we used $U = x_1, x_2, x_3, x_4, x_5, \ldots, x_N$ to represent the set of unlabeled data, which contain N transactions, and $L = x_{N+1}, x_{N+2}, x_{N+3}, x_{N+4}, x_{N+5}, \ldots, x_{N+K}$, where $K \ll N$, includes a small subset of K labeled fraud transactions that provide an initial overview of the fraudulent transaction patterns. To ensure the accuracy of the model while reducing the amount of labeled data, we implement a semi-supervised learning approach. This method leverages the large unlabeled dataset U to enhance the model's understanding of the overall transaction patterns, while the small labeled dataset L provides the necessary guidance for fraud detection.

Following that, our objective is very clear to develop an optimized deep learning enabled fraud detection function $\phi : X \rightarrow \mathbb{R}$ to calculate fraud score, and efficiently assigns fraud scores directly based on the minimal labeled dataset, bypassing the need for initial extensive representation learning. Furthermore, we designed this function to ensure that $\phi(x_i) > \phi(x_j)$ when x_i represents a fraudulent credit card transaction and x_j represents a normal transaction. This means the function assigns higher fraud scores to fraudulent transactions compared to legitimate ones. Given that, it helps to effectively identify immediate and accurate fraudulent transactions in real time.

In the proposed model, we combined neural networks with the initial probability distribution function of fraud scores to introduce a novel loss function. This helps fraud detector in the training process to ensure that fraudulent transactions are assigned higher fraud scores compared to normal transactions. By incorporating this probabilistic feature, our model can better differentiate between various transaction types and adapt to evolving fraud patterns. The anticipated results of the model not only generates more precise fraudulent scores but also exhibits greater efficiency compared to conventional methods.

To explore the internal structure of the model,it has three major modules such as summarized as follows. Initially, we develop a fraud scoring network, denoted as function ϕ, to generate a scalar fraud score for each input X in the model.

To guide the scoring process, a reference score generator is used to produce a scalar reference score, which is defined as the average of the fraud scores such as $U = R = \{r_1, r_2, r_3, r_4, \ldots, r_l\}$ for a group of l randomly chosen normal samples, represented by μ_R. This reference score is derived through the model prior probability F (fraud data) to produce μ_R and facilitate interpretable fraud scoring.

Next, the fraud scoring function $\phi(x)$, and reference score μ_R are integrated with its corresponding standard deviation σ_R for the deviation function L calculation. The objective of this optimization process is to ensure that the scores for fraudulent transactions significantly exceed μ_R in the upper distribution tail while maintaining the scores for normal items as close as possible to μ_R. However, this process will raise a challenge such as the collection of an adequate number of normal transactions for training our model, given the limited number of labeled transactions in K and the unknown class labels of objects in U. This process is summarized in Figure 2.

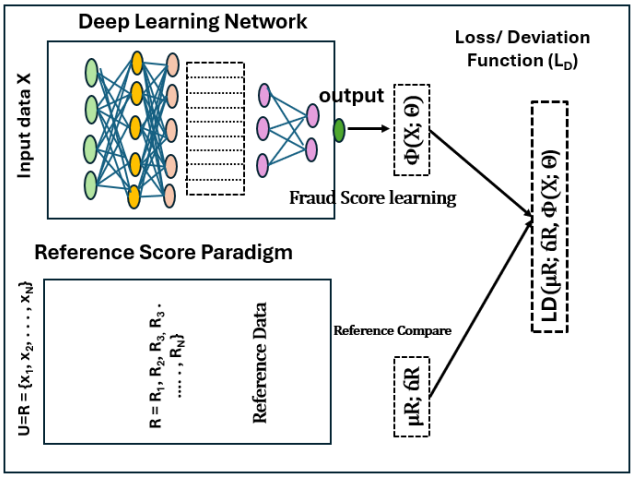

FIGURE 2. OptDevNet model operational steps.

Here in Figure 2, we can see that $\phi(x; \Theta)$ functions work to calculate the fraudulent transaction score utilizing the parameters Θ. The mean μ_R of the fraud scores for specific normal objects is established by a prior probability function F. Moreover, the σ_R in the given scenario represents the standard deviation associated with μ_R. The loss function L_f associated with $\phi(x; \Theta)$, μ_R, σ_R is formulated to ensure that the fraudulent transaction scores for transaction diverge markedly from μ_R towards the upper tail while striving to align the fraud scores of normal entities closely with μ_R. This optimization based on the deviation loss function encourages the normal transactions to converge around L_f according to their fraudulent scores. Moreover, this approach can be falter under highly fraudulent conditions by considering the unlabeled training dataset U. Unlike traditional deep learning

model, our OptDevNet approach utilizes this training process to ensure fraudulent transactions in the testing phase.

A. FEATURE EXTRACTION AND DATA PRE-PROCESSING

Before feeding the data into the Deep Event Optimized Network (DevOptNet), it is essential to extract pertinent features and prepare the data adequately. Let us define the initial dataset as $X = \{x_1, x_2, \ldots, x_n\}$, where each $x_i \in \mathbb{R}^p$ signifies the i-th data point, and p is the dimensionality of the input space. We employ a feature transformation function $\varphi : \mathbb{R}^p \rightarrow \mathbb{R}^q$ that maps raw data into a feature space of higher dimensionality, where q denotes the number of features derived. The resulting feature set F_{set} can be expressed as:

$$F_{set} = \{\varphi(x_1), \varphi(x_2), \ldots, \varphi(x_n)\}$$

This transformed set F_{set} is subsequently used for training the fraud detection model.

$$F = \{\varphi(x_i)\}_{i=1}^{n} = \{\phi_1(x_i), \phi_2(x_i), \ldots, \phi_q(x_i)\}_{i=1}^{n} \quad (1)$$

In Equation (1), $\phi_j : \mathbb{R}^p \rightarrow \mathbb{R}$ signifies the j-th feature extraction function, where $\varphi(x_i) = [\phi_1(x_i), \phi_2(x_i), \phi_3(x_i), \phi_4(x_i), \ldots, \phi_q(x_i)]^T \in \mathbb{R}^q$ is the feature vector for the i-th data point. After feature extraction, the feature set F_{set} is normalized to ensure compatibility with the DevOptNet model. Let F'_{set} denote the normalized feature set, where $F'_{set} = \Omega(\varphi(x_i))_{i=1}^{n}$, with $\Omega : \mathbb{R}^q \rightarrow \mathbb{R}^q$ representing the normalization function.

We employ the min-max normalization technique, described by the function $\Omega_{\text{norm}} : \mathbb{R}^q \rightarrow \mathbb{R}^q$, defined as:

$$\omega_{\text{norm}}(\varphi(x_i)) = \frac{\varphi(x_i) - \min(\varphi(X))}{\max(\varphi(X)) - \min(\varphi(X))} \quad (2)$$

In Equation (2), $\min(\varphi(X))$ and $\max(\varphi(X))$ denote the minimum and maximum values of the extracted features across all data points respectively. The normalized feature set F' is given by:

$$F'_{set} = \omega_{\text{norm}}(\varphi(x_i))_{i=1}^{n} \quad (3)$$

These normalized features F'_{set} are then fed into the DevOptNet model, denoted as f_Θ, which is parameterized by Θ representing the weights and biases of the model for the training. The training of the model involves minimizing a specified loss function $L(\Theta; D_{\text{train}})$ over the training dataset X_{train}.

B. DATA SPLITTING

After preprocessing, the dataset X is partitioned for training and testing to collectively evaluation of the model's performance:

$$\begin{aligned} &X = X_{\text{train}} \cup X_{\text{val}} \cup X_{\text{test}}, \\ &X_{\text{train}} \cap X_{\text{val}} = \emptyset, \\ &X_{\text{train}} \cap X_{\text{test}} = \emptyset, \\ &X_{\text{val}} \cap X_{\text{test}} = \emptyset \end{aligned} \quad (4)$$

The training subset X_{train} is utilized to adjust the model parameters Θ, defined as:

$$X_{\text{train}} = \{(\omega_{\text{norm}}(\varphi(x_i)), y_i)\}_{i=1}^{N_{\text{train}}} \quad (5)$$

Here, $\omega_{\text{norm}}(\varphi(x_i))$ denotes the normalized feature vector for the i-th data point, $y_i \in \{0, 1\}$ indicates the label (0 for normal, 1 for fraud), and N_{train} represents the count of training examples. The validation set X_{val} is used for tuning hyperparameters and for periodic evaluation during the training phase, defined as:

$$X_{\text{val}} = \{(\omega_{\text{norm}}(\varphi(x_i)), y_i)\}_{i=N_{\text{train}}+1}^{N_{\text{train}}+N_{\text{val}}} \quad (6)$$

In Equation 6, N_{val} indicates the number of validation samples. Lastly, the test subset D_{test} evaluates the final model performance on unseen data, detailed as:

$$X_{\text{test}} = \{(\omega_{\text{norm}}(\varphi(x_i)), y_i)\}_{i=N_{\text{train}}+N_{\text{val}}+1}^{N} \quad (7)$$

In Equation 7, $N = N_{\text{train}} + N_{\text{val}} + N_{\text{test}}$ denoting the total number of instances in X.

C. MODEL TRAINING

Our proposed OptDevNet uses a score deviation loss function with a gaussian prior probability function to facilitate the end-to-end optimization of fraud scores using a dedicated neural network for fraudulent score analysis.

Let $P \in \mathbb{R}^M$ represent a latent representation space. A fraud detection framework $\phi(\cdot; \Theta) : X \rightarrow \mathbb{R}$ can be conceptualized as a fusion of a feature extraction module $\psi(\cdot; \Theta_f) : X \rightarrow P$ and a scoring mechanism $\xi(\cdot; \Theta_r) : P \rightarrow \mathbb{R}$, where $\Theta = \Theta_f, \Theta_r$. Specifically, $\psi(\cdot; \Theta_f)$ acts as a neural network for feature learning with $G \in \mathbb{N}$ hidden layers, characterized by the weights $\Theta_f = W_1, W_2, W_3, W_4, \ldots, W_G$, defined by:

$$p = \psi(x; \Theta_f) \quad (8)$$

In Equation 8, $x \in X$ and $p \in P$. The network architectures follow the recurrent networks for time-series data. The scoring function $\xi(\cdot, \Theta_r) : P \rightarrow \mathbb{R}$, utilizing a simple linear neural unit at their output to calculate the fraud score based on the derived feature representations such as:

$$\xi(p; \Theta_r) = \sum_{i=1}^{M} v_i p_i + v_{M+1} \quad (9)$$

In Equation 9, $p \in P$ and $\Theta_r = v$ (with v_{M+1} as the bias component). Therefore, $\phi(\cdot; \Theta)$ is expressed as:

$$\phi(x; \Theta) = \xi(\psi(x; \Theta_f); \Theta_r), \quad (10)$$

Thus providing a direct mapping from input data to fraud scores, enabling training in a cohesive end-to-end manner.

After calculating the fraudulent transaction scores via $\phi(x; \Theta)$, the network utilizes a reference score, $\mu_R \in \mathbb{R}$, computed as the average of fraud scores from a selection of normal objects R. This reference score μ_R is important for guiding the optimization process. Two principal techniques

for determining μ_R are utilized such as data-driven and prior-driven. Data-driven approaches learn μ_R from the data set X, whereas prior-driven strategies derive μ_R from a predefined prior probability distribution F. For this implementation, the prior-driven approach is preferred because it offers clear interpretability for fraud scores and generates μ_R consistently by proving more efficiency. Moreover, we adopt a Gaussian prior probability function for defining the reference score: $r_1, r_2, r_3, r_4, \ldots, r_l \sim \mathcal{N}(\mu, \sigma^2)$, leading to:

$$\mu_R = \frac{1}{l}\sum_{i=1}^{l} r_i, \tag{11}$$

In Equation 11, each r_i follows the normal distribution $\mathcal{N}(\mu, \sigma^2)$, representing a fraudulent transaction score for a randomly selected normal object. We set $\mu = 0$ and $\sigma = 1$ in our experiments to maintain consistent detection across different datasets.

Subsequently, a deviation loss function is established to refine the performance of the fraud scoring in the network. This loss function uses the deviation metric, which is expressed as a Z-Score, and generalized as:

$$\text{dev}(x) = \frac{\phi(x;\Theta) - \mu_R}{\sigma_R}, \tag{12}$$

In Equation 12, σ_R represents the standard deviation of the anomaly scores set $\{r_1, r_2, r_3, r_4, \ldots, r_l\}$. Next, this deviation is modified to contrastive loss function, which is defined as.

$$L(\phi(x;\Theta), \mu_R, \sigma_R) = (1-y)|\text{dev}(x)| + y\max(0, a - \text{dev}(x)), \tag{13}$$

In Equation 13, $y = 1$ indicates the fraud instance, and $y = 0$ denotes a normal data point. The parameter a acts as a threshold that aligns with a Confidence Score level, and aim to align the fraud scores of normal data as close as possible with μ_R while maintaining a minimum deviation of a for the fraudulent instances. Notably, a negative deviation in the score for an fraud results in a significantly large loss that promotes large positive deviations. However, the challenge with this is the absence of labeled normal data. To overcome this, we assume that the unlabeled data in U are normal. Moreover, the training process of the model are generalized in pseudo code 1.

In pseudo-code 1, we summarized the whole training process. Initially, the model starts with the Θ, the parameters of the fraud scoring function, which are generally the weights in a neural network and are initialized randomly. Thereafter, the training process begins and continues for several epochs, where each epoch consists of multiple batches. Next, a specific number of data samples, b, are chosen from the training data X, and a set of l fraud scores are drawn from a normal distribution, $\mathcal{N}(\mu, \sigma^2)$. These scores represent a prior understanding of what constitutes fraud. The mean μ_R and standard deviation σ_R of these fraud scores are computed. These statistics help assess how far any given data point deviates from what is considered normal. Next,

Algorithm 1 Training the EventOptNet Model for Fraud Detection

Require: Training set $X \subseteq \mathbb{R}^D$, where $X = V \cup W$ and $\Theta = V \cap W$
Ensure: Fraud scoring function $\phi : X \mapsto \mathbb{R}$
Initialize parameters Θ randomly
for $i = 1$ to n_{epochs} **do**
  for $j = 1$ to n_{batches} **do**
    Sample b training, half from both W and V
    Draw l Fraud scores from $\mathcal{N}(\mu, \sigma^2)$
    Calculate the mean μ_R and standard deviation σ_R of these l scores
    Compute the loss: $L = \frac{1}{b}\sum_{x\in B} L(\phi(x;\Theta), \mu_R, \sigma_R)$
    Update Θ using gradient descent to minimize L
  end for
end for
return $\phi = 0$

a loss function is computed for each data point in the batch to measure the difference between the predicted fraud score by the model and the expected score based on the sample of l fraud scores. The goal of the loss function is to adjust the fraud scoring function such that the predicted scores for normal objects are close to μ_R and those for fraudulent transactions are significantly different, ideally higher. Based on the computed loss, the parameters of the fraud scoring model are updated using a gradient descent optimization technique. The process then repeats until the training is complete. Finally, the trained model can be used to evaluate new data points and determine if they are normal or fraudulent based on the learned patterns.

D. CLASSIFICATION AND EVALUATION

In this section, we discuss the OptDevNet framework for testing data set X_{test} to validate its effectiveness in identifying malicious versus normal transactions. This verification involves calculating the fraud scores $\phi(d_i;\Theta)$ for each data point d_i within X_{test}:

$$\hat{p}_i = \xi(\psi(d_i;\Theta_f);\Theta_r) \tag{14}$$

To obtain the binary classification labels,, a threshold τ is applied to the fraud scores:

$$\hat{y}_i = \begin{cases} 1, & \text{if } \hat{p}_i \geq \tau \\ 0, & \text{otherwise} \end{cases}$$

Here, $\hat{y}_i \in \{0, 1\}$ represents the predicted classification for the i-th data object (0 for normal, 1 for fraud). To quantify the effectiveness of the OptDevNet in classifying data objects, various metrics are computed based on the confusion matrix:

$$\begin{bmatrix} \text{TN} & \text{FP} \\ \text{FN} & \text{TP} \end{bmatrix}$$

The metrics considered in confusion matrix are essential for calculating performance of OptDevNet tp provide an

overview of the model's capability to distinguish between normal and fradulent transactions effectively.

IV. RESULT EVALUATION OF DIFFERENT ALGORITHMS

In this section, we discuss different algorithms that has been check on the same dataset to acknowledge why we choose the optimized deep learning model over them. Despite that we will talk about the comparative the algorithms that have been already checked on the same dataset, and compared their results.

A. REGRESSION MODEL IMPLEMENTATION

In this section, we discuss the operational steps of the logistic regression model and present the achieved results based on the considered datasets. Let $D = \{d_1, d_2, \ldots, d_n\}$ denote the input dataset, where $d_i \in \mathbb{R}^p$ represents the i-th input feature vector with p dimensions. We label the data with two output classes, 0 and 1, denoted by $y_i \in \{0, 1\}$. To probabilistically predict the output classes, we employ the logistic function $\lambda : \mathbb{R} \to (0, 1)$, defined as:

$$\lambda(z) = \frac{1}{1 + e^{-z}} \tag{15}$$

In Equation 15, z is a linear combination of the input features, given by:

$$z = \beta_0 + \beta_1 \cdot d_i^{(1)} + \beta_2 \cdot d_i^{(2)} + \ldots + \beta_p \cdot d_i^{(p)} \tag{16}$$

In Equation 16, β_0 is the bias term, and β_j represents the coefficient of the j-th feature in the input vector d_i. The hypothesis function $h_\beta : \mathbb{R}^p \to (0, 1)$ of the logistic regression model is defined as:

$$h_\beta(d_i) = \lambda(\beta^T \cdot d_i) \tag{17}$$

In Equation 17, $h_\beta(d_i)$ denotes the anticipated likelihood of input data d_i being associated with the affirmative category. To quantify the disparity between the projected probabilities and the factual labels, we utilize the binary cross-entropy loss function.

$$\begin{aligned} Loss(Y, Hypothesis) = -\frac{1}{n}\sum_{i=1}^{n}\Big[&\log\left(\text{Hypothesis}(d_i)\right) \cdot y_i \\ &+ \log\left(1 - \text{Hypothesis}(d_i)\right) \\ &\cdot (1 - y_i)\Big] \end{aligned} \tag{18}$$

In Equation 18, n represents the quantity of training samples, while y_i denotes the authentic label pertaining to the i-th sample, and $h_\beta(d_i)$ signifies the anticipated probability associated with the i-th sample. Our approach to minimizing the loss function involves fine-tuning the model parameters β through gradient descent. The formula for updating the j-th parameter β_j is articulated as:

$$\beta_j \leftarrow \beta_j - \alpha \cdot \frac{1}{n}\sum_{i=1}^{n}(h_\beta(d_i) - y_i) \cdot d_i[j] \tag{19}$$

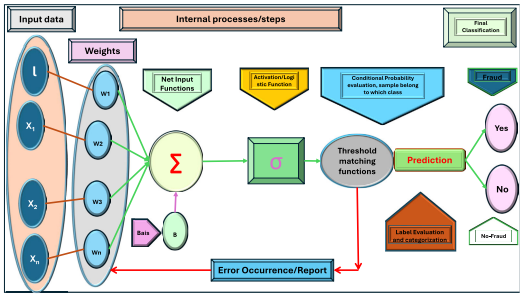

FIGURE 3. Basic structure diagram of the logistic regression model.

In Equation 19, α is the learning rate, and $d_i[j]$ is the j-th feature of the i-th sample. We initialize the model parameters β with small random values and repeatedly update them using the gradient descent rule until convergence. Figure 3 provides a visual representation of the logistic regression model for enhanced clarity and understanding.

In Figure 3, the input features $d_i, d_i, \ldots, d_i[p]$ are combined linearly with the weights β_1, β_2, $\ldots$, β_p and the bias term β_0. The resulting linear combination z is then passed through the logistic function $\lambda(z)$ to obtain the predicted probability $h_\beta(d_i)$. The predicted probability is compared with the true label y_i to compute the loss, which is minimized using gradient descent to update the model parameters β.

1) LOGISTIC REGRESSION RESULT STATISTIC EVALUATION

In this segment, we talk and showcase the assessment outcomes of a logistic regression model, which is used as a fundamental reference point for comparison. The model's performance is assessed through confusion matrices, classification metrics, and discrimination ability curves for both training and test datasets. The confusion matrices visualize the model's correct and incorrect predictions by highlighting TP, TN, FP, and FN. The classification metrics provide a comprehensive overview, including precision, recall, F1-score, and overall accuracy for each class. Furthermore, the discrimination ability curves illustrate the trade-off between TP and FP values that allow a thorough evaluation of the model's discriminative power. The area under these curves serves as a valuable metric for comparison with other models, including our optimized deep-learning approach. Through this analysis, we aim to highlight the potential advantages of our deep learning model, which leverages advanced techniques to capture complex patterns and relationships, potentially outperforming the logistic regression model and other traditional methods. Moreover, the results obtained during evaluation are shown in the form of confusion metrics and ROC plots, Fig 4.

B. SUPPORT VECTOR MACHINE

In this section, we explore the inner workings of the Support Vector Machine (SVM) algorithm and its application in classifying credit card transactions. The SVM model tries to find the best line that separates different groups of data points with some gap between them. This line is defined by

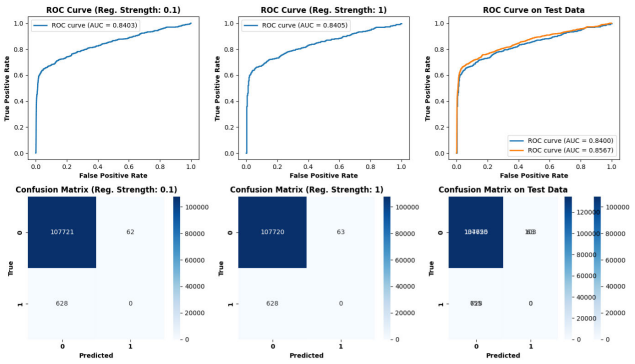

FIGURE 4. ROC curve and confusion matrix for regression model.

the undermentioned equation.

$$w^T \cdot x + b = 0 \tag{20}$$

In Equation 20, $\mathbf{w}$ signifies the weight vector, orthogonal to the hyperplane. The symbol $\mathbf{x}$ denotes the feature vector of the data point, and b represents the bias parameter. However, the margin of classification error is defined as $|\mathbf{w}|$, and expressed as:

$$\text{Margin} = \|w\|_1 \tag{21}$$

The SVM model enforces constraints to ensure that data points are correctly classified while maximizing the margin. Let $(\mathbf{x}_i, y_i)$ denote a training sample, where $\mathbf{x}_i$ denotes the feature vector and $y_i \in -1, 1$ represents the class label. The constraints can be represented as:

$$y_i(w^\top x_i + b) \geq 1 \tag{22}$$

The SVM model's optimization problem can be expressed as:

$$\begin{aligned} &\text{minimize}_{w,b} \quad \frac{1}{2}\|w\|^2 \\ &\text{subject to } y_i(w^\top x_i + b) \geq 1, \text{ for } i = 1, \ldots, N. \end{aligned}$$

After determining the optimal values of $\mathbf{w}$ and b, the decision boundary can be computed as:

$$\mathbf{w}^\top \mathbf{x} + b = 0$$

For the given "CCFD" (Credit Card Fraud Detection) dataset, denoted as $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^N$, the steps adopted for training and evaluating the SVM model are summarized in the following algorithm:

The algorithm's primary purpose is to work with the "CCFD" dataset $\mathcal{X}$ containing various transactions, both legitimate and fraudulent. It accomplishes this by training an SVM classifier, which is capable of predicting the class label (legitimate or fraudulent) of each transaction $\mathbf{x}_i$. The algorithm splits the dataset into two such as training and testing. Next, it trains the SVM model on the training data by solving the optimization problem, and obtains the optimal weight vector $\mathbf{w}^*$ and bias b^*. For each transaction $\mathbf{x}_i$ in the testing subset, the algorithm computes the predicted class label $\hat{y}_i$ using the sign of the decision function $\mathbf{w}^{*\top}\mathbf{x}_i + b^*$. The predicted labels $\{\hat{y}_i\}$ are then used to evaluate the model's performance for comparative metrics. This algorithm is designed to handle multiple transactions sequentially and deliver precise and interpretable classification results, aiding in the identification of whether each transaction can be considered legitimate or potentially fraudulent. Furthermore, the general diagram of SVM is shown in figure 5, for visual evaluation.

Algorithm 2 SVM for Credit Card Fraud Detection

Require: Transaction dataset $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^N$, labels $\{y_i\}_{i=1}^N$, where $y_i \in \{-1, 1\}$
Ensure: Predicted class labels $\hat{y}_i$ for transactions
  Split $\mathcal{X}$ into two subsets (train & test)
  Train the SVM model for solving the optimization problem
  Obtain the optimal weight vector $\mathbf{w}^*$ and bias b^*
  for $i = 1$ TO N_{test} **do**
    $\hat{y}_i \leftarrow \text{sign}(\mathbf{w}^{*\top}\mathbf{x}_i + b^*)$ Predict class label
  end for
  Evaluate the model's performance using appropriate metrics **return** Predicted class labels $\{\hat{y}_i\}$ for the testing subset $= 0$

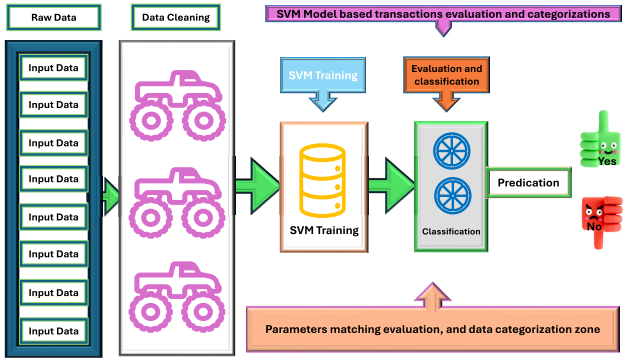

FIGURE 5. Basic structural diagram of support vector machine.

1) SVM: RESULT STATISTICS EVALUATION

In this section, we present a comprehensive evaluation of the Support Vector Machine (SVM) model by highlighting its performance in comparison to traditional ML algorithms, moderately optimized ML models, and our optimized deep learning approach. Furthermore, this analysis will help to ensure why our fine-tuned DNN model is better than SVM in the considered scenario. The evaluation results are presented through insightful visualizations, including confusion matrices and Receiver Operating Characteristic (ROC) curves. The confusion matrices provide a clear understanding of the model's misclassifications by enabling the readers to identify areas for improvement and potential biases. Additionally, the ROC curves offer a robust measure of the model's discriminative ability, which shows a fair comparison against other modeling approaches. Through

this rigorous evaluation, we demonstrate the strengths and limitations of the SVM model to pave the way for our optimized deep-learning solution. The findings not only highlight the SVM's performance but also serve as a benchmark for assessing the potential improvements offered by our deep learning approach, which leverages advanced architectures and techniques to achieve superior accuracy and generalization capabilities. Furthermore, we present the evaluation results in the form of confusion matrices, along with ROC curves, as depicted in Figure 6.

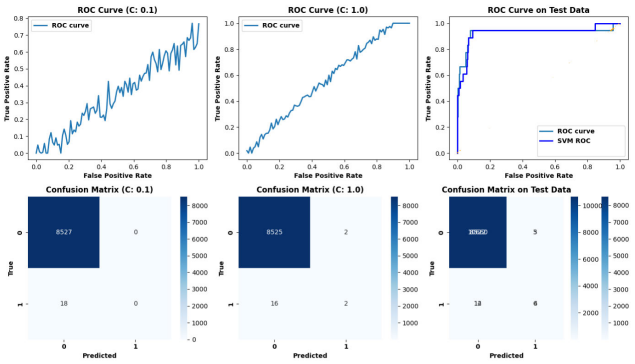

FIGURE 6. Result statistics for SVM model's evaluation.

C. TRADITIONAL KNN MODEL

In this section, we explain how the traditional KNN model works in the CCFD scenario. Let $\mathcal{X} = (x_i, y_i)_{i=1}^{N}$ denote the dataset, where $x_i \in \mathbb{R}^M$ illustrate the i-th point in the data (transaction) with M features, and $y_i \in 0, 1$ is the corresponding class label (0 for legitimate, 1 for fraudulent). The KNN algorithm, which is a non-parametric approach, always allocates class labels to the data points they are new data. These new data points are represented by x_{new}, and the algorithm determines their class label by considering the majority class among their k nearest neighbors in the feature space. The value of k is typically chosen through cross-validation or domain knowledge. To assess the resemblance or dissimilarity between different data points, KNN utilizes the Euclidean distance, denoted as $d(x_i, x_j)$. Here, x_i and x_j denote individual data points, and the distance is calculated between them. The KNN algorithm follows these steps for classifying a new data point x_{new}:

- *Distance Computation:* Compute the Euclidean distances between x_{new} and all other data points in the dataset:

$$d(\mathbf{x}_i, \mathbf{x}_{\text{new}}) = \sum_{j=1}^{M} (x_{ij} - x_{\text{new}j})^2, \quad i = 1, 2, \ldots, N \tag{23}$$

- *Nearest Neighbor Selection:* Determine the k data points with the smallest distances to x_{new}. These are the k nearest neighbors of x_{new}.
- *Class Voting:* Determine the class labels of the k nearest neighbors and tally the occurrences of neighbors from each class using the following equation:

$$\text{Count}_c = \sum_{i=1}^{k} \delta(y_i, c), \quad c \in \{0, 1\} \tag{24}$$

In Equation 24, the Kronecker delta function $\delta(y_i, c)$ evaluates to 1 when y_i is part of class c, and 0 otherwise.

- *Classification:* Assign x_{new} to the class with the majority of votes among its k nearest neighbors:

$$\text{Predicted Class}(x_{\text{new}}) = \text{argmax}_c \, \text{Count}_c \tag{25}$$

Upon completion of the classification process, x_{new} is allocated to either the legitimate or illegitimate data class, considering the prevalence of classes among its k neighbors. It is crucial to acknowledge that the KNN algorithm is very lazy in the learning process. Moreover, it retains the entire dataset and performs computations and estimation to achieve desired results. Moreover, the selection of k and the distance metric can exert a significant influence on the model's efficacy. For visual representation and evaluation, we added figure 7. We summarized the operational steps of KNN in algorithms 3.

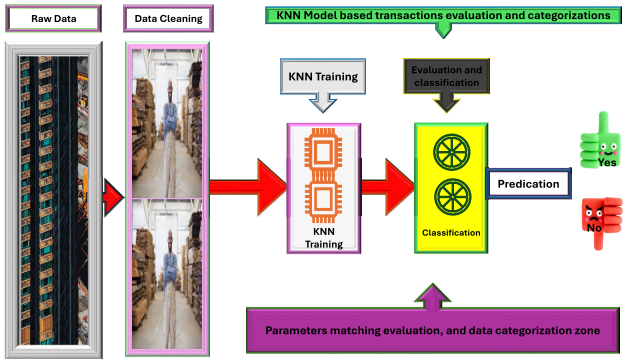

FIGURE 7. Basic structural diagram of KNN Model.

1) KNN: RESULT STATISTICS EVALUATION

In this section, we explore the results statistics of the KNN model. Through a comparative analysis against conventional machine learning algorithms, moderately tuned models, and our sophisticated deep learning methodology, our objective is to emphasize the significance and efficacy of the KNN classifier. We used informative visualizations like confusion matrices and ROC curves to provide a thorough examination of the KNN model's performance. The confusion matrices offer insights into the model's misclassifications by facilitating the identification of potential biases and areas for enhancement. Meanwhile, the ROC curves provide a robust metric for assessing the model's discriminative capability by enabling a fair comparison with alternative modeling strategies. This meticulous evaluation not only helps with the KNN model's strengths and weaknesses but also establishes a baseline for evaluating the potential enhancements achievable through our optimized deep-learning solution. Leveraging advanced architectures and

Algorithm 3 CCFD: Pseudo code of KNN

Require: $X = \{(x_i, y_i)\}_{i=1}^{N}$: Dataset of N transactions with features x_i and labels y_i
Ensure: Predicted class label for x_{new}

Function KNN_Classify(X, x_{new}, k):
distances = []
for $i = 1$ to N **do**
    d = Euclidean_Distance(x_i, x_{new})
    distances.append((d, y_i))
end for
distances.sort(by=distance) {Sort distances in ascending order}
nearest_neighbors = distances[:k] {Take the k nearest neighbors}
count_0 = 0
count_1 = 0
for d, y in nearest_neighbors **do**
    if $y == 0$ **then**
        count_0 += 1
    else
        count_1 += 1
    end if
end for
if count_0 > count_1 **then**
    return 0 {Legitimate transaction}
else
    return 1 {Fraudulent transaction}
end if
Function Euclidean_Distance(x_i, x_j):
distance = 0
for $m = 1$ to M **do**
        {M is the number of features}
    distance += $(x_{im} - x_{jm})^2$
end for
return $\sqrt{\text{distance}}$ =0

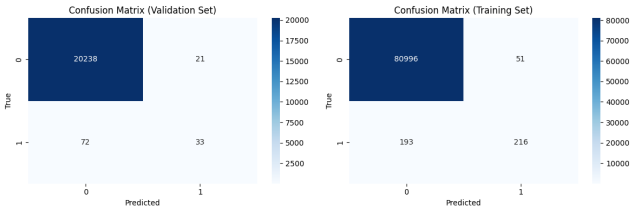

FIGURE 8. Confusion matrix present the result statistics of KNN model.

methodologies, our deep learning approach attains superior accuracy and generalization capabilities, outperforming the KNN model and other conventional techniques. Through this exhaustive analysis, we offer valuable insights into the performance characteristics of the KNN classifier, as depicted in Figure 8.

D. RANDOM FOREST ALGORITHM

In this section, we discuss the Random Forest (RF) algorithm and its application in a CCFD scenario by focusing on how it utilizes an ensemble of decision trees to enhance prediction accuracy and robustness. The core idea behind RF is to harness the combined insights of several weak learners (decision trees) to construct a robust and dependable predictive model. The key steps involved are outlined as follows:

Let $X = \{x_1, x_2, \ldots, x_n\}$ be the input feature space, and $Y = \{y_1, y_2, \ldots, y_n\}$ are used to demonstrate the target variable (fraud or non-fraud). The Random Forest model comprises an ensemble of B decision trees, denoted as $T_1(X), T_2(X), \ldots, T_B(X)$. Every decision tree $T_b(X)$ is constructed using a bootstrap sample obtained by randomly selecting instances with replacement from the initial training dataset. This method, referred to as bagging (Bootstrap Aggregating), fosters heterogeneity among the trees, thereby reducing the likelihood of overfitting. Additionally, during the construction of each decision tree, a random subset of features is considered at each node split, a process known as feature bagging. This amplifies diversity within the trees and diminishes their correlation with each other. The RF model consolidates the outcomes of each individual tree to determine the final prediction for a given input x, employing either majority voting for classification assignments or averaging for regression tasks. In the context of classification tasks, the RF prediction can be articulated as:

$$\hat{y}(x) = \text{majority\_vote}\{T_b(x)\}_{b=1}^{B} \tag{26}$$

In Equation 26, the predicted class label $\hat{y}(x)$ corresponds to the input $\mathbf{x}$. The expression $\hat{y}(x) = \text{majority\_vote}\{T_b(x)\}_{b=1}^{B}$ denotes the majority vote among the predictions of all B decision trees. Utilizing the ensemble nature of Random Forest, the algorithm adeptly captures intricate patterns and interactions within the data, rendering it a robust tool for CCFD. It is important to consider and acknowledge that the performance of RF can be affected by various factors such as the number of trees, and maximum tree depth followed by the number of features considered for each node split. Therefore, effective hyperparameter tuning and careful model selection techniques are essential to achieve optimal performance in CCFD applications.

E. RESULT STATISTICS OF RANDOM FOREST ALGORITHM

In this section, we outline the comparative outcomes of our RF model, an ensemble learning technique that merges multiple decision trees for predictions. By analyzing its performance in comparison to traditional machine learning algorithms, moderately optimized models, and our advanced deep learning approach, we aim to highlight the strengths and limitations of this versatile model.

We conduct a thorough evaluation of the model, considering the comparative metrics set for comparison. Confusion matrix results furnish an understanding of the model's misclassifications and help the readers to pinpoint the biases area. Meanwhile, ROC curves serve as a reliable metric for assessing the model's discriminative power that allows a fair comparison with alternative modeling techniques. This

comprehensive assessment not only highlights the strengths of the RF model but also serves as a benchmark for gauging the enhancements achievable through our optimized deep-learning approach. Additionally, Figure 9 illustrates the outcomes acquired from the analysis.

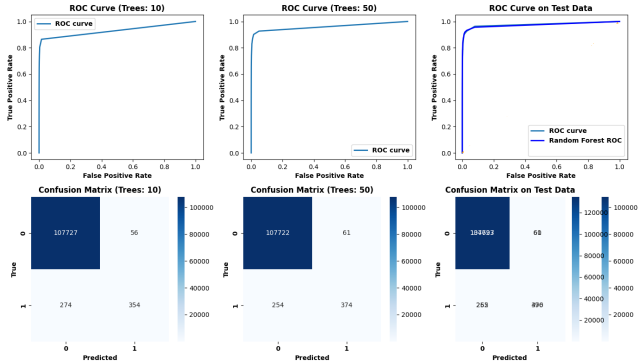

FIGURE 9. Result statistics of random forest algorithms.

F. PROPOSED OptDevNet MODEL RESULT STATISTICS

In this section, we shift our focus towards the results obtained from our optimized deep neural network (DNN) model to highlight its unique characteristics and performance in comparison to traditional machine learning algorithms used in the domain. During analysis, the proposed DNN model demonstrates exceptional capabilities in achieving remarkable results, particularly in terms of both training and validation accuracy metrics, even when using a small number of training examples with a large number of hidden layers (model fine-tuning). This indicates the model's effectiveness in adapting to the data with a reasonable number of training epochs to ensure its robust convergence. Furthermore, we check the DNN model's performance by presenting training accuracy graph followed by validation accuracy, as shown in Figure 10. We also shown the confusion matrix for result statistics evaluation in Figure 11. These results not only reveal the model's strengths and limitations but also highlight its relevance and effectiveness in the classification task. The superior performance achieved by our fine-tuned DNN classifier reaffirms its significance in handling complex data and underscores its potential as a valuable tool for various classification tasks such as CCFD.

G. OVERALL RESULTS EVALUATION WITH EXISTING STUDIES

In this section, we comprehensively examine of the current state-of-the-art articles published on this topic. Our aim is to conduct a thorough comparative analysis by shedding light on why this paper is essential within the context of existing research. We accomplish this by meticulously evaluating each of the considered paper by emphasizing their respective contributions, and subsequently elucidating how our model outperforms them in various aspects.

Fang et al. [28], introduced a Light Gradient Boosting Machine model for the detection of fraudulent transactions

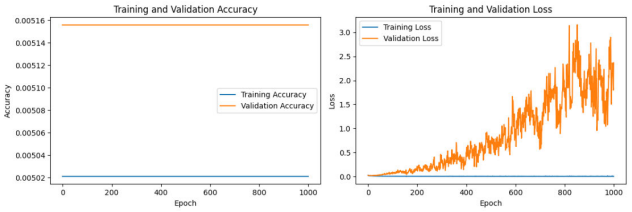

FIGURE 10. The proposed DNN result statistics for training accuracy and validation accuracy.

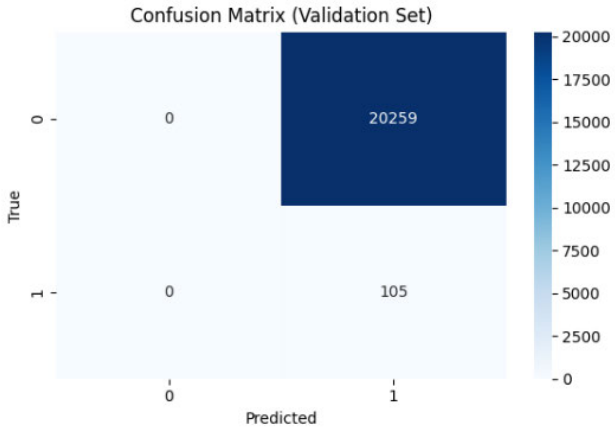

FIGURE 11. The proposed DNN result statistics "Confusion Matrix."

in CC transactions by achieving an impressive F1 score of 99%. Additionally, they conducted a comparative analysis, pitting their obtained results against those achieved by the Random Forest algorithms followed by Gradient Boosting algorithms to demonstrate that their model is superior than them in certain aspects. In contrast, our model obtained results of 99.8% with considerably fewer training iterations and epochs when compared to this model and its competing schemes. This clearly illustrates the superior performance of our model. In [29], the author introduced an ensemble model that combines DRN network with an innovative DNN using the voting mechanism to detect fraudulent actions. The authors claimed that their experimental results on real-world datasets are better than the existing models. However, the paper does not comprehensively discuss the main factors that make this work superior to existing state-of-the-art schemes. In contrast, our model demonstrates remarkable results, particularly in terms of F1 score, training and validation accuracy, and efficiency, with a shorter training time that shows its superiority over the existing models.

In [30], the authors evaluated several algorithms performance such as Random Forest, SVM, and KNN, for CC deceptive transactions by achieving better results with accuracy rates of 94.84% and 89.46% for Random Forest and KNN, respectively. Additionally, they declared that Random Forest exhibited rapid predictions for new fraud cases. However, our model achieved an accuracy of 99.89%, surpassing the results of the mentioned model by a significant margin. In [31], the authors address the credit card fraud detection issue at the application level through feature

selection methods. They used J48 decision tree, Random Forest, AdaBoost and Naive Bayes algorithms to detect financial frauds and compares their performance based on precision metrics. We achieved better statistical results compared to their findings. In [32], the authors introduced a hybrid technique that combines supervised and unsupervised algorithms to detect fraudulent transactions. The experimental results demonstrated that their model outperforms existing algorithms in terms of both efficiency and accuracy. However, the complex integration method, lengthy training times, and the absence of disclosed accuracy scores make it challenging to apply this model in real-world scenarios. Conversely, our model boasts shorter training times and high training and validation scores, affirming its superiority. In [33], the authors compiled common attributes, features, and available data of CC transactions to facilitate new research. They evaluated existing fraud detection methods for their effectiveness in addressing challenges such as real-time detection, imbalanced datasets, concept drift, and classifier adaptability. However, their achieved results for all algorithms are not convincing, whereas our model performs significantly better than their evaluated algorithms in terms of comparative metrics.

V. CONCLUSION

In this study, we introduced an Optimized Deep Event-based Network (OptDevNet) framework for detecting and preventing fraudulent transactions. The motivation for this work comes from the increasing security threat posed by credit card fraud (CCF), which presents a significant challenge to financial institutions. Fraudsters continuously employ new techniques to compromise the security of these systems. Moreover, we noted in the literature that the effectiveness of these systems relies on machine learning (ML)-)-enabled algorithms, which privately depend on specific use cases and the features of the input data to perform this task. Traditional deep learning (DL) algorithms, such as convolutional neural networks (CNNs), have shown promising results compared to ML algorithms, but they have yet to achieve remarkable results. We evaluated the proposed model against rival algorithms, including support vector machines (SVM), logistic regression, random forest, and K-nearest neighbors (KNN) on the CCFD dataset. Our analysis achieves an exceptional accuracy by surpassing both the evaluated algorithms and existing state-of-the-art schemes. Notably, we observed that the performance on unseen data improved as class imbalance increased. Given that, we are confident that the proposed model will effectively meet the requirements of involved stakeholders.

REFERENCES

[1] R. B. Sulaiman, V. Schetinin, and P. Sant, "Review of machine learning approach on credit card fraud detection," *Hum.-Centric Intell. Syst.*, vol. 2, nos. 1–2, pp. 55–68, 2022.

[2] A. Cherif, A. Badhib, H. Ammar, S. Alshehri, M. Kalkatawi, and A. Imine, "Credit card fraud detection in the era of disruptive technologies: A systematic review," *J. King Saud Univ., Comput. Inf. Sci.*, vol. 35, no. 1, pp. 145–174, Jan. 2023.

[3] Y. Zhang, M. Jamjoom, and Z. Ullah, "Double deep Q-network next-generation cyber-physical systems: A reinforcement learning-enabled anomaly detection framework for next-generation cyber-physical systems," *Electronics*, vol. 12, no. 17, p. 3632, Aug. 2023.

[4] N. S. Alfaiz and S. M. Fati, "Enhanced credit card fraud detection model using machine learning," *Electronics*, vol. 11, no. 4, p. 662, Feb. 2022.

[5] E. Esenogho, I. D. Mienye, T. G. Swart, K. Aruleba, and G. Obaido, "A neural network ensemble with feature engineering for improved credit card fraud detection," *IEEE Access*, vol. 10, pp. 16400–16407, 2022.

[6] J. F. Roseline, G. B. S. R. Naidu, V. S. Pandi, S. A. A. Rajasree, and N. Mageswari, "Autonomous credit card fraud detection using machine learning approach," *Comput. Elect. Eng.*, vol. 102, Sep. 2022, Art. no. 108132.

[7] I. Park, D. Kim, J. Moon, S. Kim, Y. Kang, and S. Bae, "Searching for new technology acceptance model under social context: Analyzing the determinants of acceptance of intelligent information technology in digital transformation and implications for the requisites of digital sustainability," *Sustainability*, vol. 14, no. 1, p. 579, Jan. 2022.

[8] T. Hewavitharana, S. Nanayakkara, A. Perera, and P. Perera, "Modifying the unified theory of acceptance and use of technology (UTAUT) model for the digital transformation of the construction industry from the user perspective," *Informatics*, vol. 8, no. 4, p. 81, Nov. 2021.

[9] P. K. Sadineni, "Detection of fraudulent transactions in credit card using machine learning algorithms," in *Proc. 4th Int. Conf. I-SMAC*, Oct. 2020, pp. 659–660.

[10] J. R. D. Kho and L. A. Vea, "Credit card fraud detection based on transaction behavior," in *Proc. IEEE Region 10 Conf. (TENCON)*, Nov. 2017, pp. 880–884.

[11] Y. Timofeyev and T. Busalaeva, "Current trends in insurance fraud in Russia: Evidence from a survey of industry experts," *Secur. J.*, vol. 34, no. 1, pp. 1–25, Mar. 2021.

[12] N. Tyagi, A. Rana, S. Awasthi, and L. K. Tyagi, "Data science: Concern for credit card scam with artificial intelligence," in *Cyber Security in Intelligent Computing and Communications*. Singapore: Springer, 2022, pp. 115–128.

[13] M. J. Madhurya, H. L. Gururaj, B. C. Soundarya, K. P. Vidyashree, and A. B. Rajendra, "Exploratory analysis of credit card fraud detection using machine learning techniques," *Global Transitions Proc.*, vol. 3, no. 1, pp. 31–37, Jun. 2022.

[14] A. Ali, S. A. Razak, S. H. Othman, T. A. E. Eisa, A. Al-Dhaqm, M. Nasser, T. Elhassan, H. Elshafie, and A. Saif, "Financial fraud detection based on machine learning: A systematic literature review," *Appl. Sci.*, vol. 12, no. 19, p. 9637, Sep. 2022.

[15] G. K. Kulatilleke, "Challenges and complexities in machine learning based credit card fraud detection," 2022, *arXiv:2208.10943*.

[16] J. K. Afriyie, K. Tawiah, W. A. Pels, S. Addai-Henne, H. A. Dwamena, E. O. Owiredu, S. A. Ayeh, and J. Eshun, "A supervised machine learning algorithm for detecting and predicting fraud in credit card transactions," *Decis. Anal. J.*, vol. 6, Mar. 2023, Art. no. 100163.

[17] D. Prusti, J. K. Rout, and S. K. Rath, "Detection of credit card fraud by applying genetic algorithm and particle swarm optimization," in *Proc. 3rd Int. Conf. MIND*. Singapore: Springer, Jan. 2023, pp. 357–369.

[18] A. Singh, R. K. Ranjan, and A. Tiwari, "Credit card fraud detection under extreme imbalanced data: A comparative study of data-level algorithms," *J. Exp. Theor. Artif. Intell.*, vol. 34, no. 4, pp. 571–598, Jul. 2022.

[19] Y. Y. Dayyabu, D. Arumugam, and S. Balasingam, "The application of artificial intelligence techniques in credit card fraud detection: A quantitative study," in *Proc. E3S Web Conf.*, vol. 389. Les Ulis, France: EDP Sciences, 2023, p. 07023.

[20] A. Razaque, M. B. H. Frej, G. Bektemyssova, F. Amsaad, M. Almiani, A. Alotaibi, N. Z. Jhanjhi, S. Amanzholova, and M. Alshammari, "Credit card-not-present fraud detection and prevention using big data analytics algorithms," *Appl. Sci.*, vol. 13, no. 1, p. 57, Dec. 2022.

[21] Z. Zhu, "Research on long-term care insurance fraud early warning based on Apriori algorithm," *Proc. SPIE*, vol. 12249, pp. 216–221, May 2022.

[22] F. A. Ghaleb, F. Saeed, M. Al-Sarem, S. N. Qasem, and T. Al-Hadhrami, "Ensemble synthesized minority oversampling-based generative adversarial networks and random forest algorithm for credit card fraud detection," *IEEE Access*, vol. 11, pp. 89694–89710, 2023.

[23] M. Chaudhry, I. Shafi, M. Mahnoor, D. L. R. Vargas, E. B. Thompson, and I. Ashraf, "A systematic literature review on identifying patterns using unsupervised clustering algorithms: A data mining perspective," *Symmetry*, vol. 15, no. 9, p. 1679, Aug. 2023.

[24] S. Kumar, V. K. Gunjan, M. D. Ansari, and R. Pathak, "Credit card fraud detection using support vector machine," in *Proc. 2nd Int. Conf. Recent Trends Mach. Learn., IoT, Smart Cities Appl.* Singapore: Springer, 2022, pp. 27–37.
[25] M. Naushin, A. K. Das, J. Nayak, and D. Pelusi, "Rough-fuzzy based synthetic data generation exploring boundary region of rough sets to handle class imbalance problem," *Axioms*, vol. 12, no. 4, p. 345, Mar. 2023.
[26] S. P. Maniraj, A. Saini, S. Ahmed, and S. Sarkar, "Credit card fraud detection using machine learning and data science," *Int. J. Eng. Res.*, vol. 8, no. 9, pp. 110–115, 2019.
[27] O. Adepoju, J. Wosowei, S. Lawte, and H. Jaiman, "Comparative evaluation of credit card fraud detection using machine learning techniques," in *Proc. Global Conf. Advancement Technol. (GCAT)*, Oct. 2019, pp. 1–6.
[28] Y. Fang, Y. Zhang, and C. Huang, "Credit card fraud detection based on machine learning," *Comput., Mater. Continua*, vol. 61, no. 1, pp. 185–195, 2019, doi: 10.32604/cmc.2019.06144.
[29] J. Forough and S. Momtazi, "Ensemble of deep sequential models for credit card fraud detection," *Appl. Soft Comput.*, vol. 99, Feb. 2021, Art. no. 106883.
[30] N. Kousika, G. Vishali, S. Sunandhana, and M. A. Vijay, "Machine learning based fraud analysis and detection system," *J. Phys., Conf. Ser.*, vol. 1916, no. 1, May 2021, Art. no. 012115.
[31] A. Singh and A. Jain, "Adaptive credit card fraud detection techniques based on feature selection method," in *Advances in Computer Communication and Computational Sciences*. Singapore: Springer, 2019, pp. 167–178.
[32] F. Carcillo, Y.-A. Le Borgne, O. Caelen, Y. Kessaci, F. Oblé, and G. Bontempi, "Combining unsupervised and supervised learning in credit card fraud detection," *Inf. Sci.*, vol. 557, pp. 317–331, May 2021.
[33] S. Mittal and S. Tyagi, "Computational techniques for real-time credit card fraud detection," in *Handbook of Computer Networks and Cyber Security: Principles and Paradigms*, B. Gupta, G. Perez, D. Agrawal, and D. Gupta, Eds., Cham, Switzerland: Springer, 2020, pp. 653–681, doi: 10.1007/978-3-030-22277-2_26.

MUHAMMAD ADIL (Senior Member, IEEE) is currently with the Department of Computer Science and Engineering, University at Buffalo, and The State University of New York, USA. He has CCNA and CCNP certifications. He has many publications in prestigious journals, such as IEEE Internet of Things, IEEE Transactions of Intelligent Transportation Systems, IEEE Transactions on Industrial Informatics, IEEE Transactions on Network Science and Engineering, IEEE Transactions on Vehicular Technology, IEEE Transactions on Intelligent Vehicles, IEEE Transactions on Green Communication Networking, IEEE Sensor Journal, *IEEE Network Magazine*, *IEEE SMS Magazine*, *IEEE Micro Magazine*, *ACM Computing Surveys*, *ACM Transactions on Sensor Networks*, *Computer Networks* (Elsevier), and *Sustainable Cities and Societies*. His research interests include networking, cybersecurity, cyber-physical systems (CPS), unnamed aerial vehicles (UAVs), the Internet of Things (IoT), and wireless sensor networks (WSN). In addition, he is a member of the IEEE Computer Society, IEEE Industrial Electronics, IEEE Cybersecurity, IEEE Young Professionals, IEEE Industrial Electronic Ambassador (Region-1), USA and London Journal Press ClubUK, and as an Honorary member. He is reviewing for prestigious journals, such as IEEE Internet of Things Journal, IEEE Sensors, IEEE Systems, IEEE Transactions on Industrial Informatics, IEEE Transactions on Cognitive Communications and Networking, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Intelligent Vehicles, IEEE Transactions on Artificial Intelligence, IEEE Transactions on Systems, Man, and Cybernetics: Systems, IEEE Transactions on Green Communications and Networking, IEEE Wireless Communications Letters, *IEEE Communication Magazine*, *IEEE Network Magazine*, *IET Communication*, *Computer Networks* (Elsevier), JNCA, FGCS, and *Computer Security*.

ZHANG YINJUN (Member, IEEE) received the master's degree from the Department of Control Science and Engineering, in 2011, and the Ph.D. degree from the Department of Computer Security, in 2019. His current research interests include neural networks, deep learning, and reinforcement learning.

MONA M. JAMJOOM (Member, IEEE) received the Ph.D. degree in computer science from King Saud University. She is currently an Associate Professor with the Department of Computer Sciences, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University, Riyadh, Saudi Arabia. She has published several research articles in her field. Her research interests include artificial intelligence, cognitive computing, and machine learning.

ZAHID ULLAH (Member, IEEE) received the master's degree in computer science from the University of Peshawar, Pakistan. He was a Researcher with King Saud University, Riyadh, Saudi Arabia, for five years. Currently, he is with the Department of Information Systems, College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University, Riyadh. He has published several articles in his research areas. His research interests include cognitive computing, business-IT alignment, IT business values, customer relationship management (CRM), and enterprise resource planning (ERP).