SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V11I11P114

Volume 11 Issue 11, 153-167, November 2024
© 2024 Seventh Sense Research Group®

Original Article

Navigating the IoT Landscape: A Deep Dive into Anomaly Classification for IoT Security

Jisha Jose[1], J.E. Judith[2]

[1,2]*Department of Computer Science and Engineering, Noorul Islam Centre for Higher Education, Tamil Nadu, India.*

[1]*Corresponding Author : jisha.jose.321@outlook.com*

Received: 15 September 2024 Revised: 14 October 2024 Accepted: 13 November 2024 Published: 03 December 2024

Abstract - *The Internet of Things (IoT) is crucial for technological advancement, attracting significant interest from researchers worldwide. However, the exponential growth of IoT devices and their huge volumes of data introduce substantial risks related to several security threats and vulnerabilities. The increasing implementation of IoT infrastructure has led to challenges such as device failures, elevated risks, and greater exposure to attacks, anomalies, and potential security breaches. Tackling and alleviating these concerns represent a critical area of focus within the broader field of IoT. By utilizing the IoTID20 dataset, precisely designed for IoT anomaly detection, the study suggests a novel approach for anomaly classification in IoT environments using a Deep Learning (DL) model optimized by a Normalized Bayesian Optimization Algorithm (NBOA) and a Convolutional Neural Network (CNN) architecture is employed for classifying anomaly, benefiting from the spatial pattern recognition capabilities of CNNs. The study employs a Machine Learning (ML) approach that utilizes Decision Tree (DT) and feature selection through the Harris Hawk Optimization (HHO) algorithm. The study validates the efficiency of using NBOA and HHO in boosting anomaly classification, ensuring faster convergence and improved accuracy. The outcomes demonstrate the superior performance of the DL model with 95.67% accuracy, surpassing the proposed ML and other state-of-the-art models. Combining these advanced optimization techniques with the DL and ML models, the study addresses the security challenges in the rapidly expanding IoT landscape, offering a robust solution for real-time anomaly detection.*

Keywords - *Anomalies, Internet of things, Deep learning, Machine learning, Optimization.*

1. Introduction

In the past few decades, internet services and products have experienced significant growth. The exponential growth of the IoT is one such significant and beneficial breakthrough [1]. The IoT is a decentralized network that links physical objects to the internet through various network devices or routers. These all-devices exchange data autonomously without human intervention, increasing automation, data capture, and speed of operation. Several sensors are utilized in IoT to sense various objects and automate the process [2]. The sensed and collected data are then analyzed to produce appropriate information for optimal policy decisions. IoT is becoming a key technology in smart homes, industries, retail, transportation, and more due to its rapid ascent. Despite its benefits, the phenomenal expansion of network infrastructure has significantly increased landscape vulnerability, making interception a constant possibility during these situations due to weak authentication policies [3].

The absence of strong security and the increase in viruses explicitly created for security devices become very alarming. Intrusion occurs when an unauthorized user accesses the privacy, dignity, availability, and protection of resources connected to a network. An Intrusion Detection System (IDS) has been developed to detect these intrusions accurately. IoT networks are susceptible to various security threats, like Denial of Service (DoS), Mirai, and Man-in-the-Middle (MITM) ARP Spoofing attacks, each exploiting specific weaknesses in IoT devices and their communication protocols.

Mirai, for instance, is a self-propagating malware that targets IoT devices with weak or default credentials, using them as botnets to launch large-scale DDoS attacks. DoS attacks flood a network with traffic, rendering devices or services unavailable, while MITM ARP Spoofing enables attackers to intercept or alter communications between IoT devices. These vulnerabilities often arise from weak authentication protocols, outdated firmware, and poor security practices in IoT device design [4]. Detection and classification of these anomalies are crucial to mitigate such attacks in IoT applications across various sectors. In a DoS attack, the attackers deny all the network services to the original users. During the scanning process, the hardware and system information are collected. Mirai is a self-propagating worm that exploits security to infect as many devices as possible.

Conventional security measures for IoT networks are often insufficient because of their processing power and energy consumption. Therefore, advanced anomaly classification techniques using artificial intelligence are increasingly being utilized [5]. These techniques analyze huge streams of IoT data and help identify deviations from normal behavior to avert security incidents. Effective anomaly classification allows overall resilience in IoT systems, permits early threat detection and response, and ensures data confidentiality, integrity, and availability. This study proposes a DL and ML model with optimized feature selection methods for effective anomaly classification in IoT security. The main contributions of the research are listed below:

- To develop a Deep Learning (DL) model using Bayesian optimization for anomaly classification in IoT security.
- To develop a Machine Learning (ML) model using Harris Hawk Optimizer (HHO) for anomaly classification in IoT security.
- To implement multiclass classifiers to categorize the anomalies present in IoT security.
- To assess and compare the efficiency of the suggested methods.

The structure of the remaining section of the paper is organized as follows: Section 2 discusses related works. The suggested models with enhanced feature optimization techniques are explained in Section 3. The results and a discussion of the suggested models are presented in Section 4. The study is finally summarized in Section 5 with a conclusion of the contributions.

2. Literature Review

Altulaihan et al. [6] proposed an IDS mechanism using anomaly detection and ML to detect DoS attacks. The study employed the IoTID20 dataset alongside four supervised classification algorithms to observe network traffic continuously for deviations from standard profiles. The models used various feature selection algorithms and compared the results. The results demonstrated that DT and Random Forest (RF) enhanced with Genetic Algorithm (GA) provide superior performance. The study had limitations in varying computational resource requirements for different classifiers. Guan et al. [7] introduced a two-tiered framework for classifying anomalies using a hybrid DL model that integrates Bidirectional Long Short-Term Memory (Bi-LSTM) and CNN. Particle Swarm Optimization (PSO) was employed for feature selection, and data imbalance was addressed. The model surpassed other machine learning models with an accuracy of 88%. The study faced challenges in detecting new or unusual anomalies, which were not represented in the training process.

Xin et al. [8] investigated CNN and Variational Autoencoders (VAE) to improve anomaly detection. The CNN model demonstrated superior performance, and the VAE model effectively detected anomalies and irregularities in the data. The study had limitations in data balance, leading to biased classification outcomes. Khan and Alkhathami [9] utilized a publicly available IoT dataset of 33 types of IoT attacks with non-biased supervised ML models. The models were analyzed by eliminating highly correlated features and speeding up training time. The RF model outperformed other models, demonstrating superior accuracy in reduced and all feature spaces. The study faced challenges in varying network conditions and real-time operational constraints.

Lai et al. [10] analyzed traditional and ensemble ML methods and Bayesian Optimization (BO) to detect cybersecurity attacks in IoT. The models were evaluated across diverse datasets, highlighting the set of influencing configurations and optimal hyperparameter tuning. The study suggested that Extreme Gradient Boosting (XGB) and Gradient Boosting Machines (GBM) achieved high accuracy. The study did not investigate the interaction of different IoT devices with various network protocols. Tahir et al. [11] proposed RF, DT, SVM, and GBM models for ML-based anomaly detection. The abnormity negotiation function and the self-adaptive defense procedures were combined to analyze the strength of IoT networks. The GBM model surpassed other models with 89.34% accuracy. The study lacked assessments of scalability and overfitting.

Sharma et al. [12] utilized a Deep Neural Network (DNN) model incorporating filter-based feature selection techniques. The UNSW-NB15 dataset, which included four attack classes, was used in the study. The model achieved 84% accuracy. To generate synthetic data of minority attacks, Generative Adversarial Networks (GANs) were used, and the model achieved 91% accuracy with a balanced class dataset. Senthil et al. [13] developed a hybrid DL model combining CNN and XGB to detect anomaly-based intrusions in IoT environments. Principal Component Analysis (PCA) extracted features from three publicly available datasets. The results demonstrated that the hybrid model using the CICDDoS 2019 dataset achieved 93.21 % accuracy.

Rahim et al. [14] explored the DL model for face recognition and anomaly detection in IoT devices. Six models were proposed, and the model combining the effectiveness of Logistic Regression (LR), HistGB classifiers, and CNN excelled with an accuracy of 94% and an AUC of 0.92 for anomaly detection. The challenges posed in interpreting predictions potentially limit the trustworthiness and transparency of the model. Lawal et al. [15] analyzed the security of network anomaly mitigation methods in IoT networks. The UNSW-NB15 dataset, which contains nine categories of attacks, was employed to evaluate the performance of various classification algorithms. With an accuracy of 94.38% and 94.71%, respectively, for the information gain feature selection methods and the correlation coefficient, the outcomes presented that the kNN model outpaced other models.

A critical gap identified across the studies is the challenge of generalizing anomaly detection models in dynamic and diverse IoT environments, particularly in relation to detecting novel or evolving threats. While several studies demonstrate high accuracy using specific datasets and models, they often face limitations in addressing real-world scenarios where new attack vectors are not represented in the training data. Additionally, many approaches do not sufficiently explore the interaction between various IoT devices and network protocols, which can significantly influence detection capabilities. The dependence on static datasets raises concerns about the adaptability and scalability of these models in practice. Furthermore, issues such as data imbalance, overfitting, and the interpretability of model predictions remain inadequately addressed, hindering the deployment of these systems in operational settings. Addressing these gaps improves the strength and pertinence of anomaly detection mechanisms in IoT security.

3. Materials and Methods

In IoT devices, anomaly classification is essential for ensuring security and reliability by maintaining data integrity. The system detects abnormal behavior in potential security breaches and prevents failures by examining data for deviations from normal behavior and operational efficiency. An ML model is suggested in this study to classify anomalies in IoT security. The dataset with four categories of anomalies and one normal is preprocessed and feature optimized. The output is fed into the DL and ML classifiers to classify between the categories. Figure 1 represents the block diagram of the suggested model.

3.1. Dataset

A well-designed dataset is required for new techniques and detection algorithms for IoT security. The study utilized a new IoTID20 dataset [16] comprising more comprehensive network and flow-based features. A typical smart home environment uses an artificial intelligence speaker, laptops, smartphones, and Wi-Fi cameras (EZVIZ) as interconnected network components. SKT NGU speakers and EZVIZ are the IoT victim devices used to generate the IoTID20 dataset, while other devices are the attacking equipment. With its high rank features, this new IoT botnet dataset provides a reference point for recognizing anomalies across various IoT networks. This dataset consists of four anomaly categories (Mirai, scan, DoS, and MITM ARP Spoofing) and one normal category. The dataset consists of 77 features. Figure 2 illustrates the sample data and dataset description. Tables 1 and 2 provide the data distribution for binary and category label distributions.

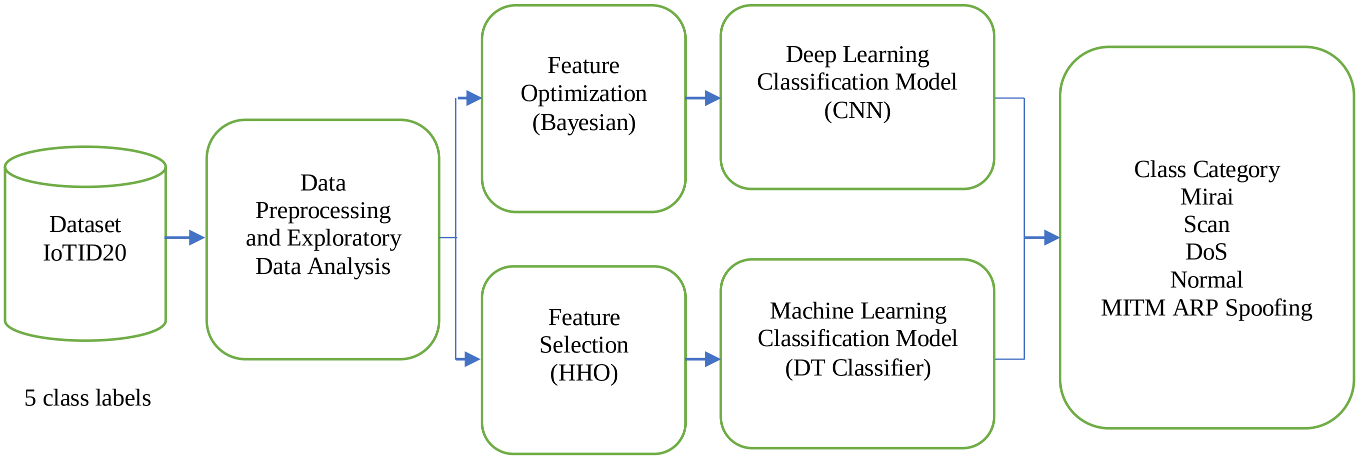

Dataset
IoTID20
5 class labels
Data Preprocessing and Exploratory Data Analysis
Feature Optimization (Bayesian)
Feature Selection (HHO)
Deep Learning Classification Model (CNN)
Machine Learning Classification Model (DT Classifier)
Class Category
Mirai
Scan
DoS
Normal
MITM ARP Spoofing

Fig. 1 Block diagram of the suggested model

Table 1. Binary label distribution

Binary label	Count
Anomaly	585,710
Normal	40,073

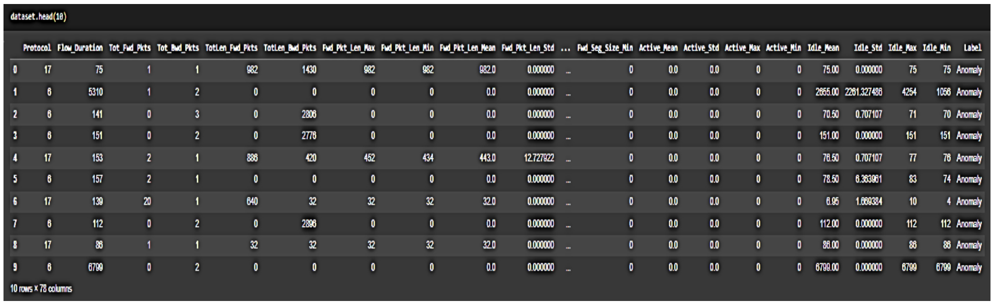
Table 2. Category label distribution

Category label	Count
Mirai	415,677
MITM ARP Spoofing	35,377
Scan	75,265
DoS	59,391
Normal	40,073

3.2. Preprocessing and Exploratory Data Analysis

Several techniques are applied to enhance the training process during data preprocessing, including data cleaning, normalization, and encoding. In the data cleaning phase, it is essential to remove null values, as missing data can result in erroneous predictions and adversely affect model performance.

No null points were presented in the dataset. Each categorical (string) value is converted into a numerical (integer) value using a label encoder. Table 3 illustrates the label encoding.

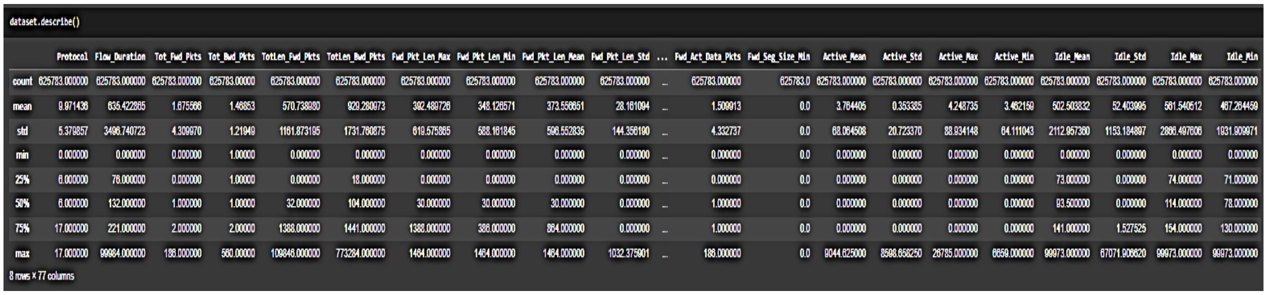


```
dataset.head(10)
```

	Protocol	Flow_Duration	Tot_Fwd_Pkts	Tot_Bwd_Pkts	TotLen_Fwd_Pkts	TotLen_Bwd_Pkts	Fwd_Pkt_Len_Max	Fwd_Pkt_Len_Min	Fwd_Pkt_Len_Mean	Fwd_Pkt_Len_Std	...	Fwd_Seg_Size_Min	Active_Mean	Active_Std	Active_Max	Active_Min	Idle_Mean	Idle_Std	Idle_Max	Idle_Min	Label
0	17	75	1	1	982	1430	982	982	982.0	0.000000	...	0	0.0	0.0	0	0	75.00	0.000000	75	75	Anomaly
1	6	5310	1	2	0	0	0	0	0.0	0.000000	...	0	0.0	0.0	0	0	2655.00	2261.327486	4254	1056	Anomaly
2	6	141	0	3	0	2806	0	0	0.0	0.000000	...	0	0.0	0.0	0	0	70.50	0.707107	71	70	Anomaly
3	6	151	0	2	0	2776	0	0	0.0	0.000000	...	0	0.0	0.0	0	0	151.00	0.000000	151	151	Anomaly
4	17	153	2	1	886	420	452	434	443.0	12.727922	...	0	0.0	0.0	0	0	76.50	0.707107	77	76	Anomaly
5	6	157	2	1	0	0	0	0	0.0	0.000000	...	0	0.0	0.0	0	0	78.50	6.363961	83	74	Anomaly
6	17	139	20	1	640	32	32	32	32.0	0.000000	...	0	0.0	0.0	0	0	6.95	1.669384	10	4	Anomaly
7	6	112	0	2	0	2896	0	0	0.0	0.000000	...	0	0.0	0.0	0	0	112.00	0.000000	112	112	Anomaly
8	17	86	1	1	32	32	32	32	32.0	0.000000	...	0	0.0	0.0	0	0	86.00	0.000000	86	86	Anomaly
9	6	6799	0	2	0	0	0	0	0.0	0.000000	...	0	0.0	0.0	0	0	6799.00	0.000000	6799	6799	Anomaly

10 rows × 78 columns

(a)



```
dataset.describe()
```

	Protocol	Flow_Duration	Tot_Fwd_Pkts	Tot_Bwd_Pkts	TotLen_Fwd_Pkts	TotLen_Bwd_Pkts	Fwd_Pkt_Len_Max	Fwd_Pkt_Len_Min	Fwd_Pkt_Len_Mean	Fwd_Pkt_Len_Std	...	Fwd_Act_Data_Pkts	Fwd_Seg_Size_Min	Active_Mean	Active_Std	Active_Max	Active_Min	Idle_Mean	Idle_Std	Idle_Max	Idle_Min
count	625783.000000	625783.000000	625783.000000	625783.00000	625783.000000	625783.000000	625783.000000	625783.000000	625783.000000	625783.000000	...	625783.000000	625783.0	625783.000000	625783.000000	625783.000000	625783.000000	625783.000000	625783.000000	625783.000000	625783.000000
mean	9.971436	635.422865	1.675566	1.46853	570.738980	929.280973	392.489726	348.126571	373.556651	28.161094	...	1.509913	0.0	3.764405	0.353385	4.248735	3.462159	502.503832	52.403995	561.540512	467.264459
std	5.379857	3496.740723	4.309970	1.21949	1161.873195	1731.760875	619.575865	588.161845	596.552835	144.356190	...	4.332737	0.0	68.064508	20.723370	88.934148	64.111043	2112.957360	1153.184897	2866.497606	1931.909971
min	0.000000	0.000000	0.000000	1.00000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	6.000000	76.000000	0.000000	1.00000	0.000000	18.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	73.000000	0.000000	74.000000	71.000000
50%	6.000000	132.000000	1.000000	1.00000	32.000000	104.000000	30.000000	30.000000	30.000000	0.000000	...	1.000000	0.0	0.000000	0.000000	0.000000	0.000000	93.500000	0.000000	114.000000	78.000000
75%	17.000000	221.000000	2.000000	2.00000	1388.000000	1441.000000	1388.000000	386.000000	864.000000	0.000000	...	1.000000	0.0	0.000000	0.000000	0.000000	0.000000	141.000000	1.527525	154.000000	130.000000
max	17.000000	99984.000000	186.000000	560.00000	109846.000000	773284.000000	1464.000000	1464.000000	1464.000000	1032.375901	...	186.000000	0.0	9044.625000	8598.658250	26785.000000	6659.000000	99973.000000	67071.908620	99973.000000	99973.000000

8 rows × 77 columns

(b)

Fig. 2 (a) Sample dataset, and (b) Dataset description.

Data normalization is structuring data within a database to eliminate redundancy and enhance data integrity, ensuring the information is stored efficiently and consistently [17]. Z-score normalization transforms features into a '0' mean and standard deviations of '1'. The process helps to speed up convergence during training. The normalization function is represented by Equation 1.

$$x' = \frac{x-\mu}{\sigma} \tag{1}$$

Where x represents the original value of a feature, μ represents the mean, and σ is the standard deviation of the feature. Exploratory Data Analysis (EDA) involves examining the distribution and features of the data through various data visualization techniques [18]. Figure 3 represents the count of different anomaly categories. Figure 4 provides the feature distribution of the dataset.

Table 3. Label feature of the IoTID20 dataset

Label	Category
0	Mirai
1	Scan
2	DoS
3	Normal
4	MITM ARP Spoofing

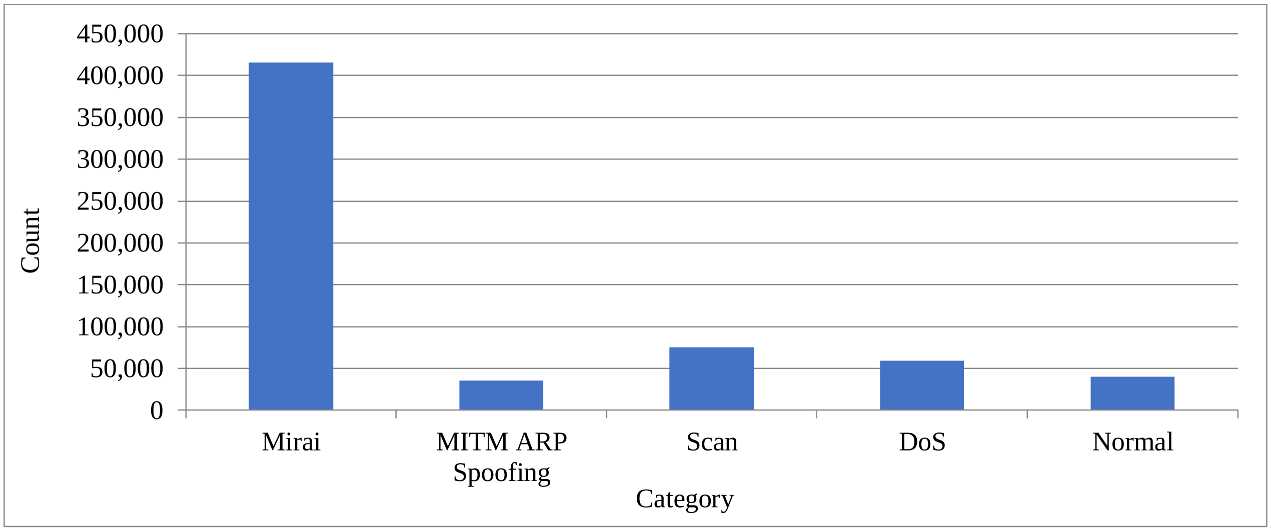


Fig. 3 Visualization of the count of different anomaly categories

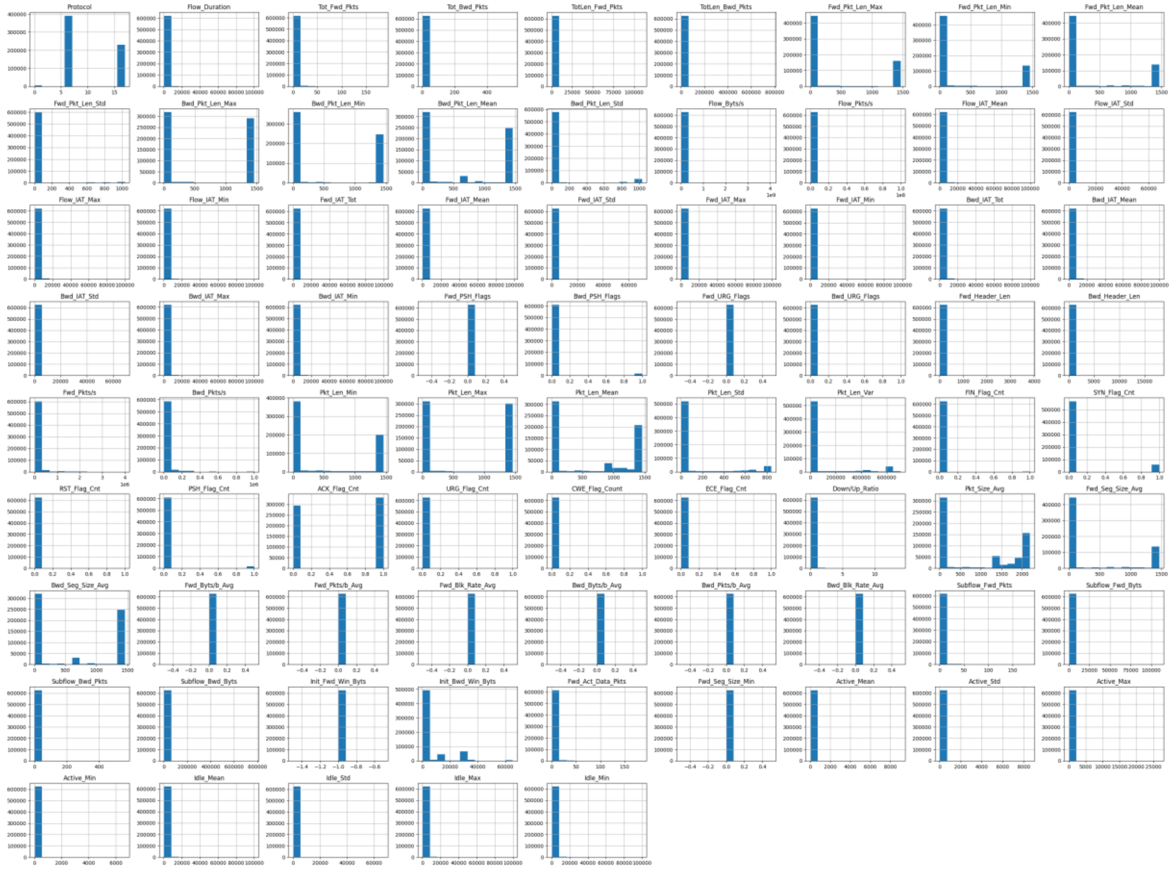


Fig. 4 Feature distribution of the dataset

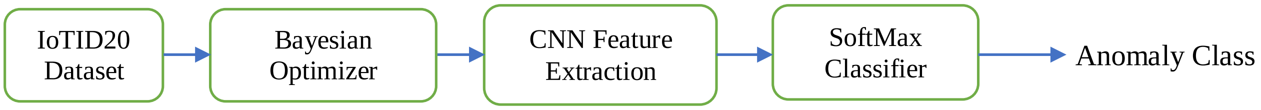


Fig. 5 Workflow of the suggested DL model

3.3. Proposed Deep Learning Model Using Normalized Bayesian Optimization Algorithm

In an era where the IoT is rapidly expanding, ensuring robust security against anomalies has become paramount. This section presents a proposed deep learning model that utilizes a normalized Bayesian optimization algorithm to classify anomalies in IoT environments effectively. Figure 5 demonstrates the workflow of the suggested DL model.

3.3.1. Normalized Bayesian Optimization Algorithm

The NBOA was chosen for its capability to balance exploration and exploitation in high-dimensional parameter spaces, ensuring efficient convergence to optimal solutions. Unlike traditional optimization techniques like particle swarm optimization or genetic algorithms, NBOA leverages a Gaussian process to model the objective function and refine predictions iteratively. This makes it particularly suited for tuning the hyperparameters of complex deep learning models, as it minimizes computational costs while maximizing performance. The BOA aims to minimize the scalar objective function $f(x)$ for variable x, producing different outputs based on whether the function is deterministic or stochastic [19]. The minimization process comprises three key elements: a Gaussian process model representing $f(x)$, a Bayesian update that refines the Gaussian model with each new evaluation of $f(x)$, and an acquisition function $a(x)$ that directs the search for optimal solutions. The next evaluation point is identified by maximizing $a(x)$. A gaussian process is a gathering of random variables with a specified mean $m(x)$ and covariance $k(x, x')$ expressed as Equation 2.

$$f(x) = \mathcal{GP}(m(x), k(x, x')) \tag{2}$$

Equation 3 defines the mean and covariance functions.

$$m(x) = \mathbb{E}[f(x)]$$

$$k(x, x') = \sigma^2 \exp\left(-\frac{(x-x')^2}{2l^2}\right) \tag{3}$$

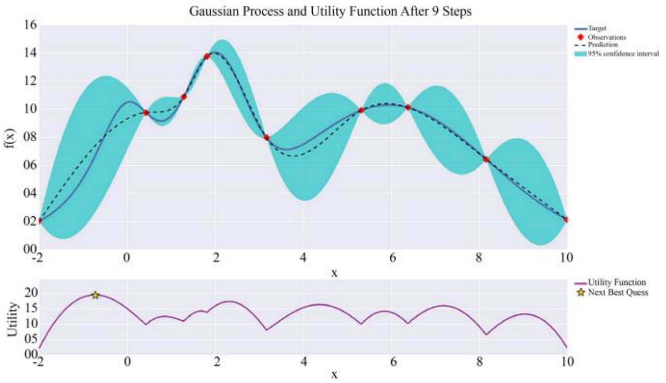


Fig. 6 Gaussian function and utility function after 9 steps

Given a set of observed data $\mathbb{X} = \{x_1, x_2, \ldots \ldots x_n\}$ having corresponding function values $\mathbb{y} = \{f(x_1), f(x_2), \ldots \ldots f(x_n)\}$, the Gaussian process provides a posterior distribution, as in Equation 4, over the function values at new points x^*. The posterior distribution improves as the number of observations increases, exploring the more valuable region in the parameter space.

$$f^* | \mathbb{X}, \mathbb{y}, x^* \sim \mathcal{N}(\mu^*, \Sigma^*) \tag{4}$$

Where $\mu^* = \mathbb{K}_*^T \mathbb{K}^{-1} y$, $\Sigma^* = \mathbb{K}_{**} - \mathbb{K}_*^T \mathbb{K}^{-1} \mathbb{K}_*$ and f^* is the best observed value. Here, K is the covariance matrix, K_* is the covariance between observed and new points and K (**) is the covariance between new points. The Gaussian process and the utility function are shown in Figure 6.

By balancing between exploration and exploitation, the acquisition function decides the position of the next sample [20]. At each step, the Gaussian process fits the explored points, and the next points to be explored are determined according to the corresponding distribution combined with the exploration strategy.

The expected improvements are defined by Equation 5.

$$EI(x) = \mathbb{E}[\max\left(0, f^* - \hat{f}(x)\right)] \tag{5}$$

Where $\hat{f}(x)$ is the Gaussian process prediction at x. The probability of improvement is expressed in Equation 6.

$$PI(x) = \varphi\left(\frac{\mu(x) - f^* - \xi}{\sigma(x)}\right) \tag{6}$$

Where φ is the cumulative distribution function, and ξ is a small positive constant. The upper confidence bound is given by Equation 7.

$$U(x) = \mu(x) + \kappa\sigma(x) \tag{7}$$

The acquisition function is maximized to find x^* as in Equation 8.

$$x^* = arg \max_x \alpha(x) \tag{8}$$

The true objective function is evaluated, the gaussian process is updated with the new data, and the process repeats until the stopping criterion. Figure 7 illustrates the action of Bayesian optimization.

To normalize the Bayesian optimizer, it is necessary to normalize the $Pr(data)$ of the Bayes' Theorem in Variational Bayes. The Bayes theorem denotes the posterior distribution as expressed in Equation 9. The variational approximation seeks to minimize the Kullback-Leibler (KL) divergence between the approximate distribution and the actual posterior, effectively refining the estimate of the true distribution by finding the closest approximation.

$$\Pr(\text{params}|\text{data}) = \frac{\Pr(\text{data}|\text{params})\Pr(\text{params})}{\Pr(\text{data})} \tag{9}$$

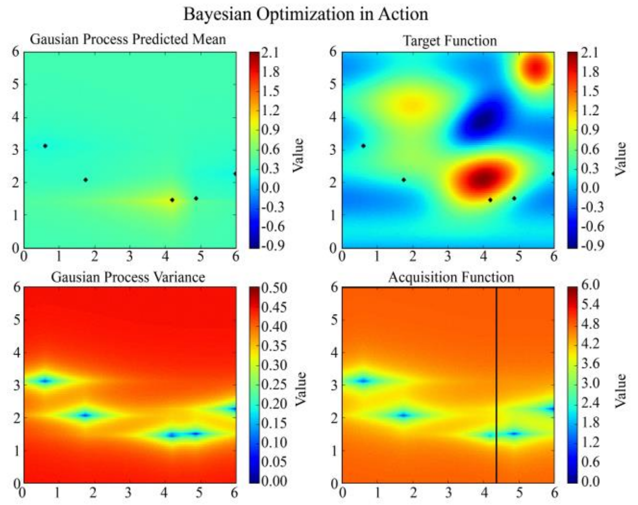


Fig. 7 Bayesian optimization in action

3.3.2. Deep Learning Model for Anomaly Classification

CNN can automatically classify and extract significant features from network traffic datasets, effectively distinguishing between normal and malicious activities. [21] Due to this, the model does exceptionally well to capture any spatial patterns, making them ideal for analyzing any visual data. The features to be fed to the CNN are optimized using NBOA. A basic CNN is structured with several layers, as illustrated in Figure 8.

CNN extracts the features from the given input to fed into the next layer. The pooling operation is done by applying a filter over the feature map and calculating the average of the elements within each patch, which filter overlaps. Each input node is now connected to the preceding output layer to form a fully connected network. The basic convolution operation is given by Equation 10.

$$y[a, b] = \sum_{i=0}^{k-1} \sum_{j=0}^{k-1} a[a + i, b + j] . w[i, j] \tag{10}$$

Where $w\,[i,j]$ represents the filter values at position (i,j), $a\,[a+i,b+j]$ is the input feature map, and $y\,[a,b]$ represents the output feature map at position (i,j). A dense-type CNN with a ReLU activation function is used in this study. The last dense layer employs A softmax activation function to produce the predicted class probabilities. The suggested architecture for the CNN model is shown in Figure 9.

3.4. Proposed Machine Learning Model Using Harris Hawk Optimization Algorithm

With the evolvement of the IoT landscape, safeguarding these interconnected systems from anomalies is critical for maintaining security and functionality. This section outlines a proposed machine learning model that utilizes the HHO algorithm to enhance anomaly classification in IoT security. Figure 10 illustrates the workflow of the suggested ML model.

3.4.1. HHO Feature Selection

The HHO algorithm is employed for feature selection to improve classification accuracy and training speed. Feature selection involves eliminating inappropriate and redundant features to select the most important one. The population-based, nature-inspired HHO slope optimization method mimics the chasing style of Harris hawks' birds [22]. Harris Hawks' attacks use a variety of chasing styles, originating from ma+ny directions to surprise their prey.

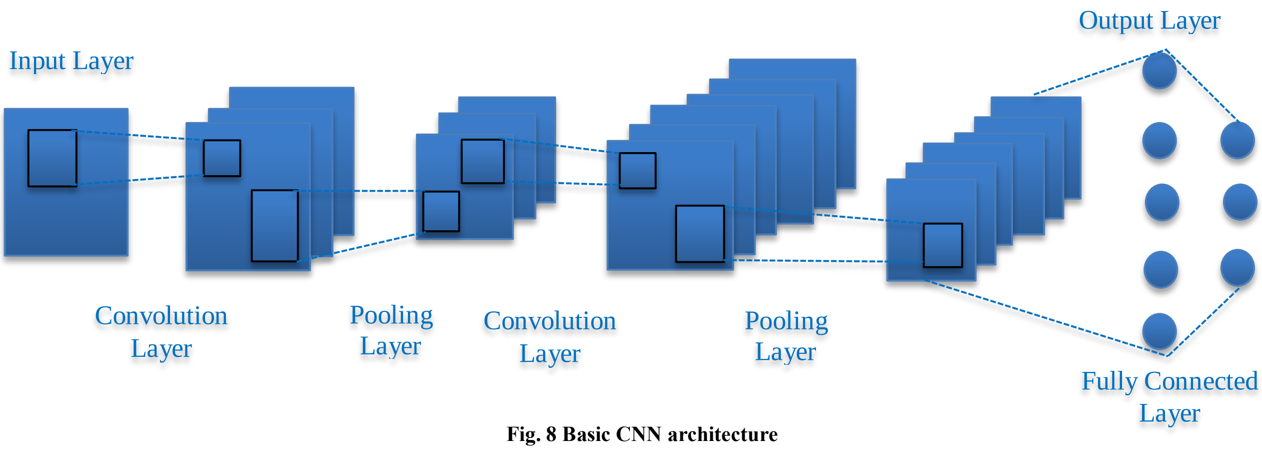


Fig. 8 Basic CNN architecture

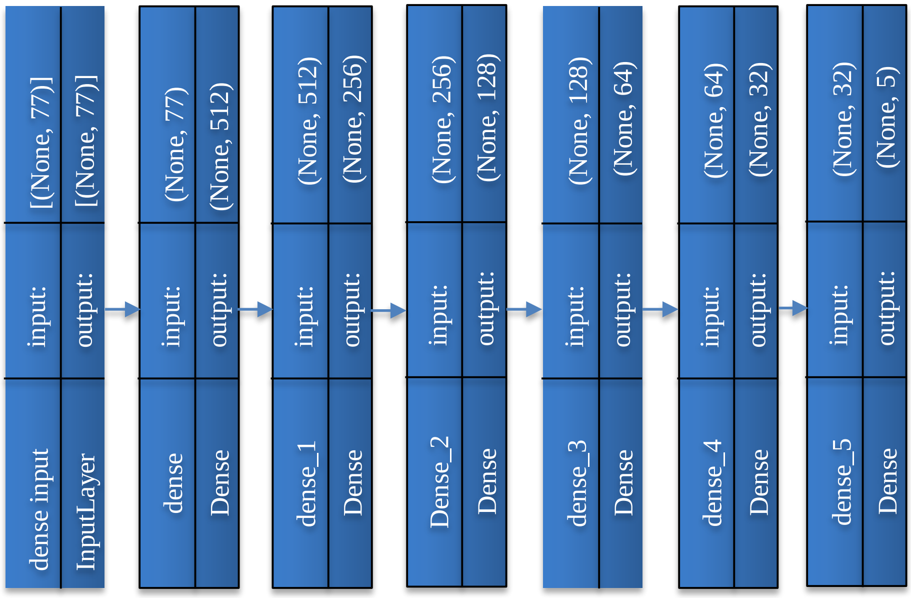


Fig. 9 Proposed CNN architecture

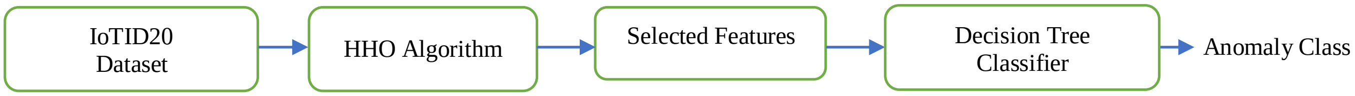


Fig. 10 Workflow of the suggested ML model

The exploration and exploitation strategies are derived from a standard HHO algorithm grounded on the attacking behaviors of Harris Hawks', like predation, preaching, and surprise pounce strategies. There are four phases for exploitation and two for exploration in HHO. Figure 11 illustrates the various phases of HHO.

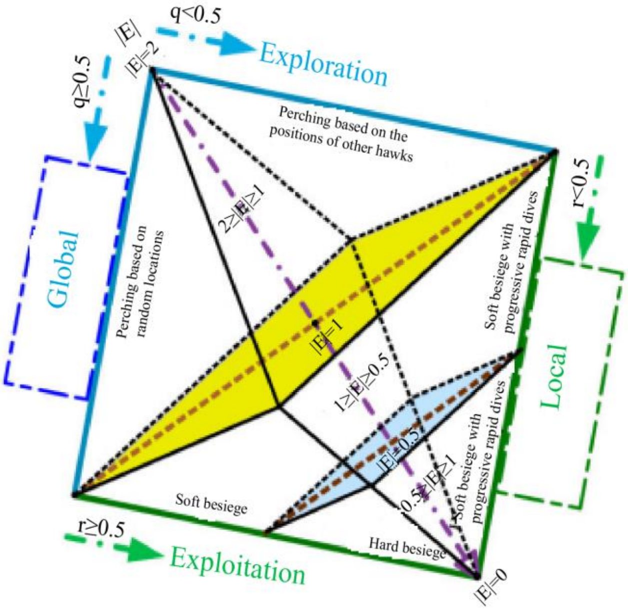

q<0.5
Exploration
|E|
|E|≥2
q≥0.5
Global
Perching based on the positions of other hawks
Perching based on random locations
2≥|E|≥1
|E|=1
1≥|E|≥0.5
|E|=0.5
0.5≥|E|≥1
r<0.5
Soft besiege with progressive rapid dives
Soft besiege with progressive rapid dives
Local
Soft besiege
Hard besiege
r≥0.5
Exploitation
|E|=0

Fig. 11 Various phases of HHO

During the initialization phase, the initial population assigns parameter values, defining the solution space of the objective function.

During the exploration phase, the Harris Hawks actively hunt for the prey. Although the hawks' attractive eyes aid in locating and following their prey, it is not always easy to realize the prey.

Here, the hawks watch and wait in hopes of seeing their prey. In every iteration, all Harris Hawks represent candidate solutions, with their fitness values assessed based on the targeted prey. Based on Equation 11, the Harris hawks wait at specific locations to find their prey.

$$X(t+1) = \begin{cases} X_{rand}(t) - r_1|X_{rand}(t) - 2r_2X(t) & q < 0.5 \\ (X_{prey}(t) - X_m(t)) - r_3(LB + r_4(UB - LB)) & q \geq 0.5 \end{cases} \qquad (11)$$

Where $X(t)$ and $X(t+1)$ represents the Hawks' position in iteration t and $(t+1)$ iterations, respectively, X_{prey} denotes the position of prey in the current population, having chosen a random solution $X_{rand}(t)$, $r_1, r_2, r_3, r_4, q \in [0,1]$ are the random scaled factors, which are updated in each iteration. UB and LB represent the upper and lower bounds of variables. $X_m(t)$ is the average number of solutions given by Equation 12.

$$X_m(t) = \frac{1}{N}\sum_{i=1}^{N} X_i(t) \qquad (12)$$

Where, according to chaos theory, $X_i(t)$ denotes the position of each solution in iteration t. The next phase is the evolution from exploration to exploitation, depicted in Figure 12, which shows how HHO moves in response to the prey's energy E.

HHO assumes that the prey's energy diminishes progressively due to its escaping actions, as in Equation 13.

$$E = 2E_0\left(1 - \frac{t}{T}\right), E_0 \in [-1,1] \qquad (13)$$

Where E_0 is the prey's initial energy, and T represents the maximum number of iterations. In the exploitation phase, HHO employed four possible approaches as follows for mimicking the attacking strategy, relying on two variables: the probability of escaping, r, and the escaping energy $|E|$ [23].

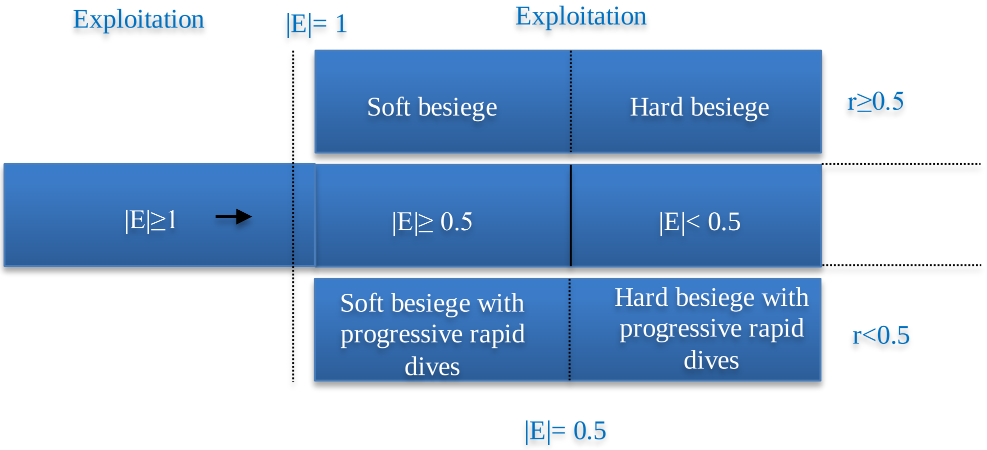


Fig. 12 Phase transitions based on energy

Equation 14 provides the condition of escaping probability.

$$r = \begin{cases} r < 0.5, & higher\ chance\ of\ successful\ \ escape \\ r \geq 0.5 & higher\ chance\ of\ unsuccessful\ escape \end{cases} \quad (14)$$

In the soft besiege scenario, where $r \geq 0.5$ and $|E| \geq 0.5$, the prey possesses adequate energy to evade capture, while the hawks gently encircle the prey to deplete its energy further before executing a surprise attack. Equation 15 shows the mathematical expression for soft besiege.

$$X(t+1) = \Delta X(t) - E\left|JX_{prey} - X(t)\right|$$

$$\Delta X(t) = X_{prey} - X(t),$$

$$J = 2(1 - r_5), r_5 \in [0,1] \quad (15)$$

In this case, r_5 is a random variable, J is the prey's jump strength, and ΔX(t) is the difference between the prey's location at iteration t and the current position vector.

In hard besiege, where $r < 0.5$ and $|E| \geq 0.5$, the prey has less possibility to escape and is tired. Here, the hawks are circling their prey close to prepare for one last surprise attack. The updated solution is represented by Equation 16.

$$X(t+1) = X_{prey}(t) - E|\Delta X(t)| \quad (16)$$

In soft besiege with progressive rapid dives, where $r < 0.5$ and $|E| \geq 0.5$, the prey retains energy to avoid capture. The hawk skilfully navigates around the prey, waiting patiently before making a sudden dive for the surprise attack.

This approach involves a two-step update of the hawk's position. Initially, the hawks approach the prey by predicting its subsequent movement, as represented by Equation 17.

$$Y = X_{prey}(t) - E\left|JX_{prey}(t) - X(t)\right| \quad (17)$$

Then, the hawk evaluates whether to dive by contrasting the results of earlier dives with the expected outcomes. If it decides against diving, the hawks engage in irregular dives, guided by the principle of Lévy Flight (LF), as described in Equation 18.

$$LF(x) = 0.001 \times \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}}, \sigma = \left(\frac{\Gamma(1+\beta) \times sin\frac{\pi\beta}{2}}{\Gamma(\frac{1+\beta}{2}) \times \beta \times 2^{(\frac{\beta-1}{2})}} \right)^{\frac{1}{\beta}}$$

$$Z = Y + S \times LF(\mathrm{Dim}) \quad (18)$$

Where Dim denotes the dimension of the solution, S represents a random vector of size $1 \times dim$, β is a constant with value 1.5 and $u, v \in [0,1]$. Thus, the update in the hawk's position with progressive rapid dives is expressed in Equation 19.

$$X(t+1) = \begin{cases} Z & if\ F(Z) < F(X(t)) \\ Y & if\ F(Y) < F(X(t)) \end{cases} \quad (19)$$

Where Y and Z are evaluated using Equations 17 and 18, respectively. In hard besiege with progressive rapid dives, where $r < 0.5$ and $|E| < 0.5$, the prey cannot escape due to insufficient energy, prompting the hawks to perform swift dives to capture it, leading to a successful surprise pounce. Equation 20 represents the movement pattern of hawks in this condition.

$$X(t+1) = \begin{cases} Z & if\ F(Z) < F\big(X(t)\big) \\ Y & if\ F(Y) < F(X(t)) \\ & \end{cases} \quad (20)$$

Where Z and Y are evaluated using Equations 21 and 22, respectively.

$$Y = X_{prey}(t) - E\left|JX_{prey}(t) - X_m(t)\right|$$

$$Z = Y + S \times LF(\mathrm{Dim}) \quad (21)$$

Subsequently, the classification error rate, represented as in Equation 22, and the selected features' minimum number are included in the fitness function calculation.

$$Fitness = \propto \gamma_R(D) + \beta \frac{|R|}{|N|} \quad (22)$$

Where $\propto \in [0,1]$and $\beta = (1 - \alpha)$, $\gamma_R(D)$ denotes the classifier error rate, $|R|$ is the number of features that have been selected, and $|N|$ is the features' total number. Here, the HHO algorithm selected 21 features out of the total available features in the dataset, reducing the dimensionality of the data while improving the model's performance.

3.4.2. Machine Learning Model for Anomaly Classification

DT are utilized for anomaly classification due to their applicability in regression and classification tasks [24]. It consists of a tree structure where the root node is the starting point on behalf of the entire dataset, making it simple to identify anomalies in the data. It splits into two or more subsets based on the attribute that maximizes the separation between normal and anomalous instances, determined using an Attribute Selection Measure (ASM). The internal nodes that result from these splits are known as decision nodes, which make decisions based on an attribute and can have multiple branches after pruning. Each branch represents a different decision rule that connects a parent node to its child nodes. At the terminal end of the tree are the leaf nodes, which represent the outcome and do not split further, as shown in Figure 13. Splitting refers to dividing a node into sub-nodes based on attribute values. At the same time, pruning removes unnecessary branches to simplify the tree and prevent overfitting, making it more generalizable [25]. Thus, the decision nodes facilitate the decision-making process with multiple branches, while the leaf nodes signify the final outcomes without further subdivisions.

To determine which feature to use for splitting, Decision Trees use different criteria to measure the purity or homogeneity of the nodes. Two popular methods are information gain and the Gini index. Information gain measures the reduction in entropy (a measure of impurity) after splitting the data based on a feature as given in Equation 23. The feature that maximizes information gain is selected for splitting.

$$G(D,A) = Entropy\ (D) - \sum_{v \in Values(A)} \frac{|D_v|}{|D|} Entropy(D_v) \tag{23}$$

Where D is the dataset, A is the feature, and entropy measures the randomness or impurity in the dataset, as in Equation 24.

$$Entropy(D) = -\sum_{i=1}^{C} p_i log_2(p_i) \tag{24}$$

Where p_i is the probability of class i. Equation 25 represents the Gini impurity index.

$$Gini(S) = 1 - \sum_{j=1}^{k} P_j^2 \tag{25}$$

Where k denotes the classes' number. The attribute with the smallest Gini index is chosen for the split. Thus, by selecting the 21 most relevant features using HHO, the DT classifier focuses on the most important variables, improving model interpretability.

The proposed anomaly detection model demonstrates significant potential for real-world applications across various sectors, including smart homes, industrial IoT, and healthcare systems.

In smart homes, the model could be deployed to monitor device interactions, identifying unauthorized access or unusual behaviors, thereby confirming the privacy, veracity, and accessibility of sensitive data. For example, it could detect abnormal patterns in home security systems or connected appliances, alerting users to potential security breaches.

The model could prevent downtime in industrial IoT settings by detecting network intrusions or device malfunctions before they escalate into major issues. Real-time detection of anomalous behavior could help reduce operational disruptions and improve the overall efficiency of critical infrastructure. Similarly, in healthcare systems, the model could safeguard medical devices and networks from cyber threats, ensuring patient safety and the protection of sensitive health data. Applying this model in these practical scenarios can make IoT systems more resilient, improving safety, operational efficiency, and reliability.

3.5. Hardware and Software Setup

A comprehensive setup is used for this study to ensure a well-equipped environment to handle the demand of neural network training and deployment. It consists of an Intel Core i7 processor, NVIDIA GeForce GTX 1080Ti GPU, 32GB of RAM, and the Python-based Keras library integrated with the TensorFlow framework. Google Colab's extensive computing resources and Keras's user-friendly interface simplified the procedure of building models and ensured the successful training and execution of complex structures.

The dataset was partitioned as 70% for training, 20% for validation and 10% for testing. Hyperparameters are critical parameters that specify the operation and functions of a DL framework throughout the training. Table 4 demonstrates hyperparameters, which are user-specified prior to training, in contrast to the model's parameters, which are determined by the data.

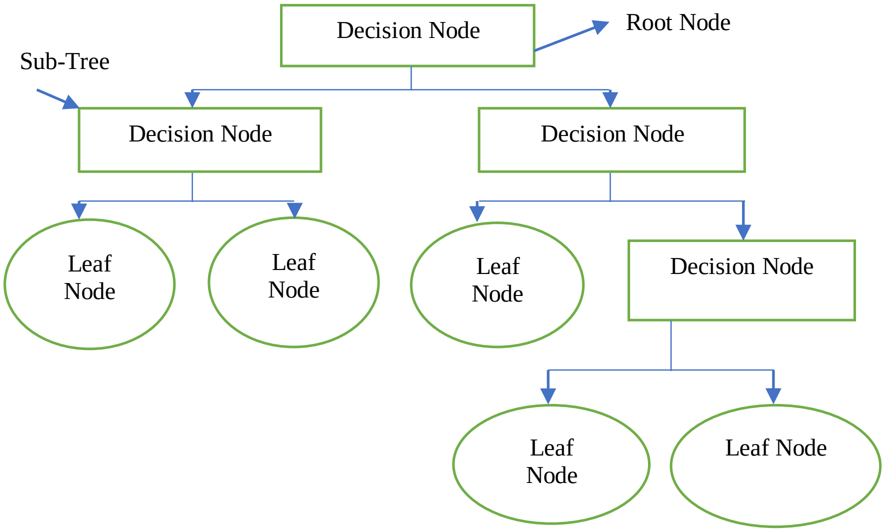


Fig. 13 Visualization of DT

Table 4. Hyperparameter specifications

Hyperparameters	Values
Optimizer	Adam
Activation function	ReLU
Epochs	500
Loss function	Categorical cross-entropy
Batch size	5000

4. Results and Discussion

4.1. Evaluation of Performance

Performance evaluation of the suggested model was conducted to ensure an inclusive understanding of its effectiveness using a variety of metrics. As shown in Table 5, the primary metrics highlight different aspects of the model's performance. The classification report for the suggested DL and ML models is illustrated in Table 6, and Figure 14 provides a visual representation of the results.

Table 5. Evaluation metrics

Performance Metrics	Equations
Accuracy	$(TP + TN) / (TP + TN + FP + FN)$
Precision	$TP / (TP + FP)$
Recall	$TP / (TP + FN)$
F1 Score	$2 * (Precision * recall) / (Precision + recall)$
Where, TP-True Positives, FP-False pOsitives, TN-True Negatives and FN-False Negatives	

Table 6. Classification report of suggested models

Evaluation parameters	Proposed Machine Learning Model	Proposed Deep Learning Model
Accuracy	92.65%	95.67%
Precision	91.37%	94.8%
Recall	92.42%	95.4%
F1 score	91.89%	95.09%

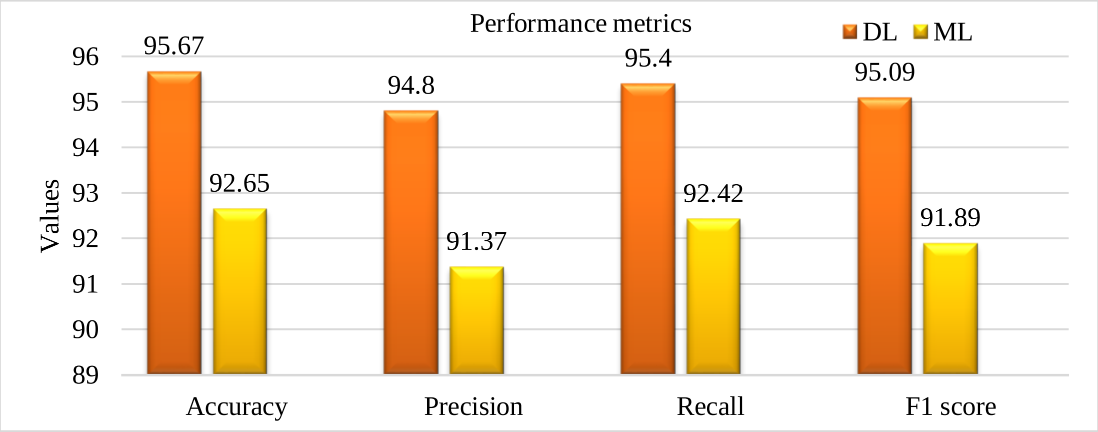


Fig. 14 Graphical representation of classification report

Table 6 highlights the efficiency of the DL and ML models in recognizing anomalies within IoT security. With an accuracy of 95.67%, the DL model exhibits significantly higher accuracy than the ML model. This suggests that the DL model demonstrates greater effectiveness in accurately classifying instances within the dataset. High accuracy is crucial, especially in IoT applications where correct predictions are paramount. A higher precision of 94.8% specifies that it is more likely to be correct when the DL model forecasts a positive class. The DL model demonstrates superior performance by capturing true positives with a recall value of 95.4%. This is critical in contexts of anomaly detection, where failing to identify positive cases (false negatives) poses a risk. With a higher F1 score of 95.09%, the DL model demonstrates a better balance between recall and precision, making it more reliable for practical applications. The study's accuracy and loss plots were crucial for assessing the model's performance throughout training. The accuracy plot demonstrated the model's learning progress, which showed strong generalization as training accuracy increased gradually and validation accuracy followed suit. The consistent decay in the loss plot indicates that the model has fewer errors and is more efficient in learning. No divergence between the training and validation metrics indicates no overfitting. The model's accuracy and loss plot are shown in Figure 15. The confusion matrix shown in Figure 16 illustrates the model performance, where predicted labels are compared to actual labels, with accurate predictions shown along the diagonal and incorrect classifications shown by off-diagonal parts.

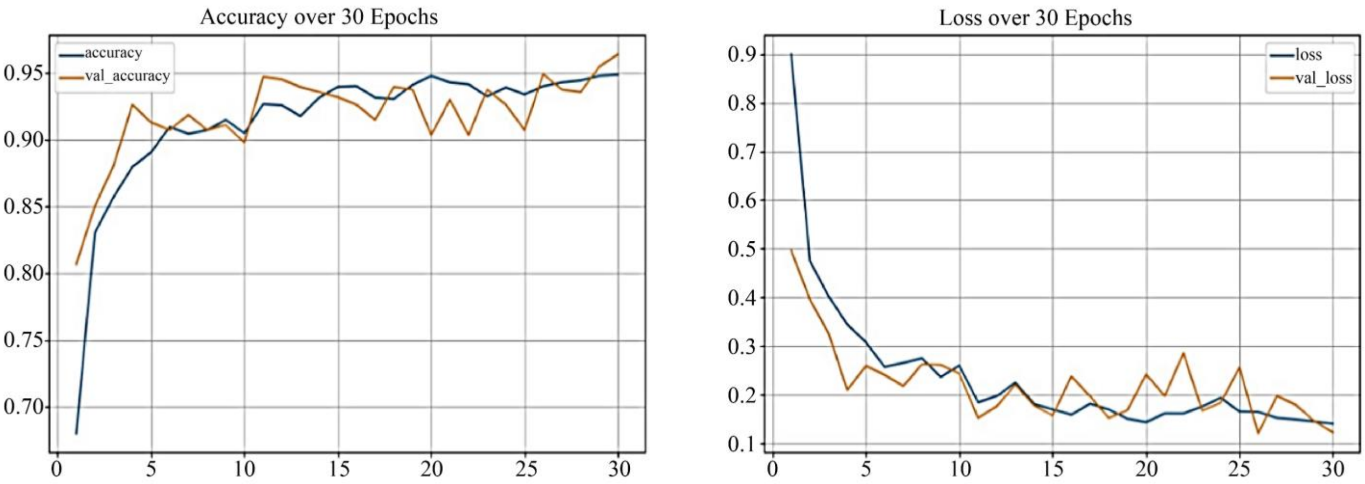


Fig. 15 Accuracy and loss plot of the suggested DL model

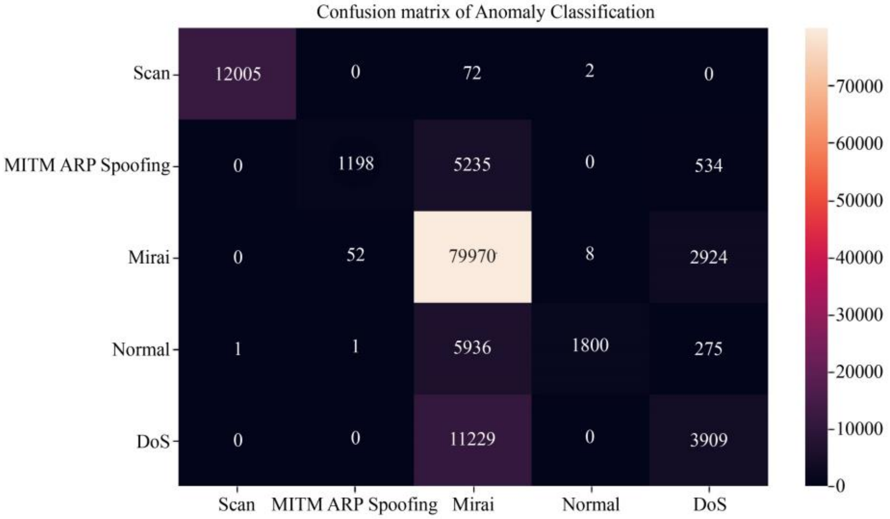


Fig. 16 Confusion matrix of the suggested DL model

Figure 17 represents the convergence plot, illustrating the fitness value progression of the HHO algorithm over several iterations. The plot shows a downward trend in fitness values, which confirms that the HHO algorithm is successfully optimizing the feature selection process. The lower the fitness value, the better the selected features are for the DT classifier. The algorithm consistently improves over the iterations, leading to more effective feature selection.

4.2. Performance Comparison

Table 7 and Figure 18 show the efficiency of the suggested model when compared to traditional models. The comparative analysis of various models on different datasets reveals that the suggested DL model achieves the highest accuracy of 95.67% on the New IoTID20 dataset, surpassing other methodologies. Notably, the CNN-BiLSTM model achieved 88%, and the GBM recorded 89.34%, indicating substantial improvements. The DNN on the UNSW-NB15 dataset achieved 91%, while the CNN-XGBoost model on the CICDDoS 2019 dataset attained 93.21%. The proposed ML model demonstrates commendable performance with 93.65% accuracy on the same dataset. This indicates that the proposed DL model not only outperforms existing models but also underscores the efficiency of DL techniques in addressing complex challenges in anomaly classification in the IoT environment.

The study's dependence on a single dataset, IoTID20, represents a limitation in terms of the model's generalizability. While IoTID20 is designed explicitly for anomaly detection, it does not fully capture the vast diversity and complexity of real-world IoT environments, particularly in terms of device types, communication protocols, and attack vectors. This limitation could affect the model's performance when applied to IoT networks with different characteristics or evolving threats. Furthermore, the computational complexity of the deep learning model, particularly the Convolutional Neural Network (CNN), may pose challenges in large-scale IoT deployments where resources such as processing power, memory, and energy are limited. Real-time processing of vast amounts of IoT data could lead to high latency or excessive computational overhead, which may hinder the model's practical application in time-sensitive environments. Acknowledging these challenges is essential for providing a balanced perspective on the study's findings. Future work could explore strategies to address these challenges, such as employing lightweight models or edge computing solutions, to make the system more scalable and suitable for deployment in diverse IoT settings.

Table 7. Comparison with the existing models

Methodology	Dataset	Accuracy (%)
CNN-BiLSTM [6]	New IoTID20	88
GBM [11]	Network Traffic Dataset	89.34
DNN [12]	UNSW-NB15	91
CNN-XGB [13]	CICDDoS 2019	93.21
Proposed DL model	**New IoTID20**	**95.67**
Proposed ML model	**New IoTID20**	**93.65**

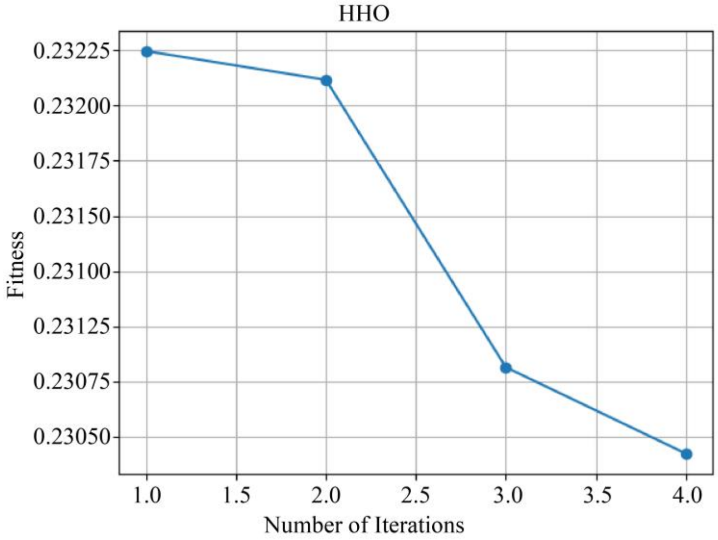


Fig. 17 Convergence plot of HHO algorithm

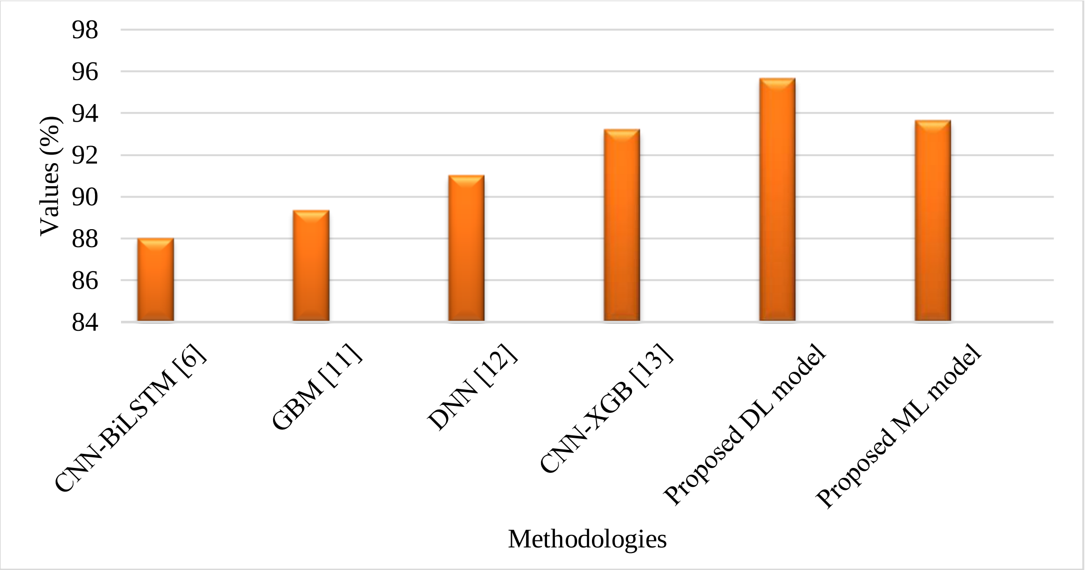


Fig. 18 Performance comparison of the suggested model with the existing methods

5. Conclusion

IoT systems are vulnerable to numerous security threats that can hinder legitimate users from accessing their services. Consequently, it is crucial to implement strong techniques and mechanisms to safeguard systems, devices, and data against attacks targeting IoT networks. By utilizing the IoTID20 dataset, the study proposed a DL model optimized by a normalized Bayesian optimization algorithm, and a CNN architecture was employed for classification, benefiting from the spatial pattern recognition capabilities of CNNs. The study also employed an ML approach that utilizes DT and feature selection through the HHO algorithm. The proposed DL methodology bids a scalable and robust solution for real-time IoT anomaly detection, outperforming conventional models in classification accuracy, having a value of 95.67% and feature selection efficiency. Future work should evaluate the proposed model across diverse IoT datasets to assess its adaptability and generalizability. It should also integrate transfer learning and advanced optimization methods to enhance efficiency and performance. Additionally, addressing challenges like computational overhead and latency will be crucial for optimizing the model for real-time deployment in resource-constrained IoT environments, enabling broader adoption in large-scale applications.

Acknowledgments

I want to express my sincere gratitude to all those who contributed to completing this research paper. I extend my heartfelt thanks to my supervisor, family, colleagues and fellow researchers for their encouragement and understanding during the demanding phases of this work.

References

[1] Martha Rodríguez, Diana P. Tobón, and Danny Múnera, “Anomaly Classification in Industrial Internet of Things: A Review,” *Intelligent Systems with Applications*, vol. 18, pp. 1-13, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[2] Fereshteh Abbasi, Marjan Naderan, and Seyed Enayatallah Alavi, “Anomaly Detection in Internet of Things Using Feature Selection and Classification Based on Logistic Regression and Artificial Neural Network on N-BaIoT Dataset,” *2021 5th International Conference on Internet of Things and Applications (IoT)*, Isfahan, Iran, pp. 1-7, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[3] Khaled A. Alaghbari et al., “Deep Autoencoder-Based Integrated Model for Anomaly Detection and Efficient Feature Extraction in IoT Networks,” *IoT*, vol. 4, no. 3, pp. 345-365, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[4] Himani Tyagi, and Rajendra Kumar, “Attack and Anomaly Detection in IoT Networks Using Supervised Machine Learning Approaches,” *Revue d'Intelligence Artificielle*, vol. 35, no. 1, pp. 11-21, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[5] Redhwan Al-amri et al., “A Review of Machine Learning and Deep Learning Techniques for Anomaly Detection in Iot Data,” *Applied Sciences*, vol. 11, no. 12, pp. 1-23, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[6] Esra Altulaihan, Mohammed Amin Almaiah, and Ahmed Aljughaiman, “Anomaly Detection IDS for Detecting DoS Attacks in IoT Networks Based on Machine Learning Algorithms,” *Sensors*, vol. 24, no. 2, pp. 1-30, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[7] Yue Guan, Morteza Noferesti, and Naser Ezzati-Jivan, “A Two-Tiered Framework for Anomaly Classification in IoT Networks Utilizing CNN-BiLSTM Model,” *Software Impacts*, vol. 20, pp. 1-8, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[8] Qi Xin et al., “IoT Traffic Classification and Anomaly Detection Method Based on Deep Autoencoders,” *Preprints.org*, pp. 1-13, 2024. [Google Scholar] [Publisher Link]

[9] Maryam Mahsal Khan, and Mohammed Alkhathami, “Anomaly Detection in IoT-Based Healthcare: Machine Learning for Enhanced Security,” *Scientific Reports*, vol. 14, no. 1, pp. 1-16, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[10] Tin Lai et al., “Ensemble Learning Based Anomaly Detection for IoT Cybersecurity via Bayesian Hyperparameters Sensitivity Analysis,” *Cybersecurity*, vol. 7, no. 1, pp. 1-18, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[11] Usama Tahir et al., “Enhancing IoT Security through Machine Learning-Driven Anomaly Detection,” *VFAST Transactions on Software Engineering*, vol. 12, no. 2, pp. 1-13, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[12] Bhawana Sharma et al., “Anomaly Based Network Intrusion Detection for IoT Attacks Using Deep Learning Technique,” *Computers and Electrical Engineering*, vol. 107, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[13] R.S. Kamalakannan et al., “Anomaly based Intrusion Detection System Using Hybrid Machine Learning Approach in IoT Environment,” *Journal of Electrical Systems*, vol. 20, no. 10s, pp. 2763-2771, 2024. [Google Scholar] [Publisher Link]

[14] Asif Rahim et al., “Enhancing Smart Home Security: Anomaly Detection and Face Recognition in Smart Home IoT Devices Using Logit-Boosted CNN Models,” *Sensors*, vol. 23, no. 15, pp. 1-42, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[15] Muhammad Aminu Lawal, Riaz Ahmed Shaikh, and Syed Raheel Hassan, “Security Analysis of Network Anomalies Mitigation Schemes in IoT Networks,” *IEEE Access*, vol. 8, pp. 43355-43374, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[16] IoT Intrusion Dataset. [Online]. Available: https://sites.google.com/view/iot-network-intrusion-dataset/home

[17] Imtiaz Ullah, and Qusay H. Mahmoud, “Design and Development of a Deep Learning-Based Model for Anomaly Detection in IoT Networks,” *IEEE Access*, vol. 9, pp. 103906-103926, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[18] Dukka Karun Kumar Reddy et al., “Ensemble Bagging Approach for IoT Sensor Based Anomaly Detection,” *Proceeding of the First International Conference on Intelligent Computing in Control and Communication*, pp. 647-665, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[19] Xilu Wang et al., “Recent Advances in Bayesian Optimization,” *ACM Computing Surveys*, vol. 55, no. 13s, pp. 1-36, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[20] Madeline E. Scyphers et al., “Bayesian Optimization for Anything (BOA): An Open-Source Framework for Accessible, User-Friendly Bayesian Optimization,” *Environmental Modelling & Software*, vol. 182, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[21] Shaik Althaf V. Shajihan et al., “CNN Based Data Anomaly Detection Using Multi-Channel Imagery for Structural Health Monitoring,” *Smart Structures and Systems*, vol. 29, no. 1, pp. 181-193, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[22] Ali Asghar Heidari et al., “Harris Hawks Optimization: Algorithm and Applications,” *Future Generation Computer Systems*, vol. 97, pp. 849-872, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[23] B.K. Tripathy et al., “Harris Hawk Optimization: A Survey Onvariants and Applications,” *Computational Intelligence and Neuroscience*, vol. 2022, no. 1, pp. 1-20, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[24] Amuthan Prabakar Muniyandi, R. Rajeswari, and R. Rajaram, “Network Anomaly Detection by Cascading K-Means Clustering and C4. 5 Decision Tree Algorithm,” *Procedia Engineering*, vol. 30, pp. 174-182, 2012. [CrossRef] [Google Scholar] [Publisher Link]

[25] Panagiotis I. Radoglou-Grammatikis, and Panagiotis G. Sarigiannidis, “An Anomaly-Based Intrusion Detection System for the Smart Grid Based on Cart Decision Tree,” *2018 Global Information Infrastructure and Networking Symposium (GIIS)*, Thessaloniki, Greece, pp. 1-5, 2018. [CrossRef] [Google Scholar] [Publisher Link]