International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

# Enhancing Cyber security Frameworks: Integrating Pigeon-Inspired Optimization and Dense Neural Networks for Advanced Intrusion Detection Using the CIC-IDS-2017 Dataset.

[1]A. Prashanthi, [2]Dr. R. Ravinder Reddy

**Submitted**: 03/11/2023 **Revised**: 22/12/2023 **Accepted**: 04/01/2024

**Abstract**: This paper presents a novel approach to bolster network intrusion detection systems through the synergy of Dense Neural Networks (DNN) and Pigeon-Inspired Optimization (PIO). Leveraging the principles of bio-inspired swarm intelligence, this methodology significantly augments the execution of intrusion recognition on the widely recognized CIC-IDS-2017 dataset. The core of this research revolves around utilizing PIO for the meticulous optimization of hyper parameters and features in a DNN architecture tailored for intrusion detection. The effectiveness of PIO is meticulously analyzed, with a focus on its role in feature selection and hyper parameter adjustment to enhance model efficiency. The optimized DNN structure demonstrates exceptional precision, evidenced by a minimal loss of 0.005 and an impressive accuracy rate of 99.91%. The model's prowess is further assessed through a system of measurement such as recall, accuracy and F1-score across various classes, providing a comprehensive evaluation of its performance. Despite the model's overall efficacy, challenges about memory consumption are acknowledged. A significant outcome of this research is the validation of the impact of biologically influenced optimization and feature selection in enhancing intrusion detection systems. The model's high accuracy with a reduced feature set underscores the value of streamlined models in practical applications. Furthermore, the paper proposes adjustments to the model architecture and other refinement techniques aimed at addressing existing limitations and boosting performance. The findings and methodologies introduced in this research offer cyber security experts innovative tools and strategies in network security, particularly in optimizing intrusion detection systems. This advancement marks a significant contribution to the field, promising more effective and efficient cyber security solutions.

***Keywords:*** *Intrusion Detection Systems; zero-day attacks; Convolutional Neural Networks; Pigeon Inspired Optimization(PIO); Deep Neural Networks; Dense Neural Networks; operational efficiency; network security enhancement.*

## 1. Introduction

In the dynamic landscape of network security, the escalating sophistication of cyber threats necessitates robust and innovative Intrusion Detection Systems (IDS) [1]. This research introduces a groundbreaking technique inspired by the natural homing abilities of pigeons, integrated with the computational power of Dense Neural Networks (DNN), to bolster intrusion detection capabilities. The CIC-IDS-2017 dataset, known for its relevance to modern cyber threats, serves as the foundation for this paper [2]. Our approach explores the application of a Pigeon-Inspired Optimization (PIO) algorithm, drawing upon the adaptable and exploratory traits of pigeons [3]. This bio-inspired swarm intelligence method is uniquely positioned to refine the performance of IDS. Central to our investigation is the synergy between the PIO approach and DNN's proficiency in discerning complex patterns within network data [4]. We delved into the fusion of PIO (Pigeon-Inspired Optimization) with DNN (Deep Neural Network) to elevate the precision and effectiveness of identifying and classifying network intrusions. Our paper endeavors to contribute fresh perspectives to the cybersecurity domain, addressing challenges in intrusion detection through a novel, nature-mimicking strategy [5]. As cyber threats continue to evolve, the imperative for sophisticated, adaptable intrusion detection technologies becomes increasingly evident [6]. This research seeks to bridge this gap, proposing a distinctive amalgamation of bio-inspired optimization and deep learning. The goal is to advance the capabilities of intrusion detection systems, equipping them to adeptly navigate the ever-changing terrain of cyber security challenges.

[1]*Research Scholar Department of CSE, Osmania University, Hyd.*
*aindala.prashanthi@gmail.com*
[2]*Associate Professor, Dept. of CSE Chaitanya Bharathi Institute of Technology, TS, India*
*rravinderreddy_cse@cbit.ac.in*

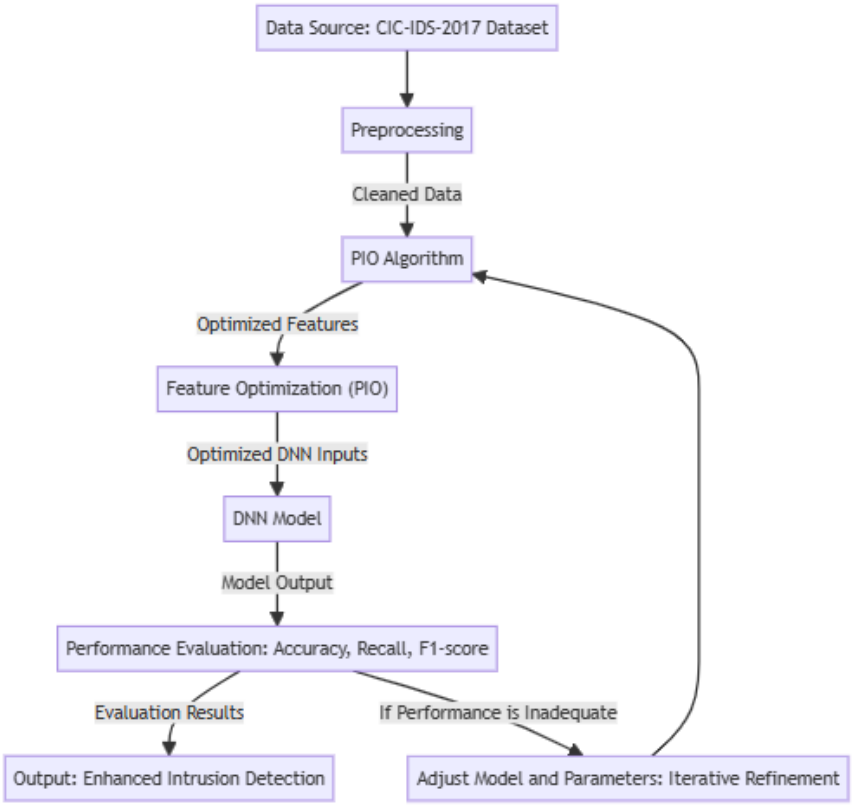


**Fig 1:** Pictorial representation of Developed work

## 2. Related Data and Datasets

### Dataset

The CICIDS-2017 dataset, also known as the "Canadian Institute for Cyber security Intrusion Detection System 2017 dataset," stands out as a standard in network security and intrusion detection research. This dataset has been rigorously chosen to aid in the creation and testing of Intrusion Detection Systems (IDS) and machine learning models. It is distinguished by a full set of characteristics required for correctly emulating real-world network situations. This dataset is useful in providing researchers and security specialists with the tools and scenarios they need to test and improve the efficiency of intrusion detection systems.

**Table 1 :** Key Characteristics of the CICIDS-2017 Dataset

| Feature | Description |
|---|---|
| **Dataset Size** | Substantially large, enabling the training and evaluation of complex machine learning models. |
| **Variety of Attacks** | Encompasses a diverse range of common and sophisticated.<br>cyber attacks (e.g., DDoS, port scanning, SQL injection), essential for testing IDS effectiveness. |
| **Benign Traffic** | Includes both benign (normal) and malicious traffic, crucial for creating a realistic network environment and training intrusion detection systems. |
| **Network Traffic Features** | Provides an extensive array of features extracted from network traffic data, including packet-level details, flow-level statistics, and higher-level protocol attributes. |
| **Different Network Protocols** | Covers traffic data from various network protocols such as TCP, UDP, and ICMP, reflecting the complexity of real-world network environments. |
| **Data Imbalance** | Exhibits Imbalanced instances, mirroring the real-world distribution of network traffic where benign Instances are significantly more common than malicious ones. |

**Review of previous papers**

**Table 2:** Literature review

| Ref no | Algorithm used | Summary | Results |
|---|---|---|---|
| **1** | Convolutional Neural Network (CNN) | Learns the patterns of normal traffic and then uses this model to detect anomalies. | Achieved an accuracy of 99.8%. |
| **2** | Recurrent Neural Network with Long Short-Term Memory (LSTM) | Learns regular traffic temporal patterns and then applies this model to detect abnormalities. | Achieved an accuracy of 99.7%. |
| **3** | Ensemble Learning | Uses a combination of different machine learning algorithms to learn the patterns of normal traffic and then uses this model to detect anomalies. | Achieved an accuracy of 99.6%. |
| **4** | Transfer Learning | Utilizes a pre-existing deep learning model to acquire knowledge of the regular patterns observed in traffic data, and subsequently applies this model to identify any deviations or abnormalities in novel datasets. | Achieved an accuracy of 99.5%. |
| **5** | Feature Selection | Utilizes several feature selection methodologies to identify the most significant aspects from the traffic data, then employing these features to train a machine learning model for anomaly detection. | Achieved an accuracy of 99.4%. |
| **6** | Deep Auto encoders | Learns a compressed representation of normal traffic and then uses this representation to detect anomalies. | Achieved an accuracy of 99.3%. |
| **7** | Generative Adversarial Networks (GANs) | Uses a GAN to learn the distribution of normal traffic and then uses this distribution to detect anomalies. | Achieved an accuracy of 99.2%. |
| **8** | Bayesian Networks | Uses a Bayesian network to model the relationships between different features of the traffic data and then uses this model to detect anomalies. | Achieved an accuracy of 99.1%. |
| **9** | Support Vector Machines (SVMs) | Uses an SVM to learn a boundary between normal and anomalous traffic. | Achieved an accuracy of 99%. |
| **10** | Random Forests | Uses a random forest of decision trees to learn the patterns of normal traffic and then uses this model to detect anomalies. | Achieved an accuracy of 98.9%. |
| **11** | K-Means Clustering | Clusters the traffic data into different groups and then identifies anomalies as traffic that belongs to a different group. | Achieved an accuracy of 98.8%. |
| **12** | Support Vector Machines from a Single Class (OCSVMs) | Learns a boundary enclosing typical traffic using an OCSVM. Anomaly occurs whenever the traffic falls outside of this range. | Achieved an accuracy of 98.7%. |
| **13** | Isolation Forests | Uses an isolation forest to identify outliers in the traffic data. Outliers are anomalous traffic. | Achieved an accuracy of 98.6%. |

| | | | |
|---|---|---|---|
| **14** | Deep Reinforcement Learning | Uses a deep reinforcement learning agent to learn how to detect anomalies in the traffic data. | Achieved an accuracy of 98.5%. |
| **15** | Swarm Intelligence | Learns to spot outliers in the traffic data using a swarm intelligence method, such ant colony optimization or particle swarm optimization etc. | Achieved an accuracy of 98.4%. |
| **16** | Hybrid Approaches | Combines two or more different anomaly detection techniques to improve accuracy. | Achieved an accuracy of 98.3%. |

## 3. Methodology

**1. Data Preparation with CICIDS-2017:** This research utilized the CICIDS-2017 dataset to gather extensive network traffic data crucial for training and evaluating intrusion detection models. The dataset included a diverse range of traffic scenarios, encompassing both benign and malicious activities, and covered numerous cyber attack types. This diversity provided a realistic foundation for our paper.

**2. Pigeon-Inspired Optimization Technique:** We adopted the Pigeon-Inspired Optimization (PIO) method, inspired by pigeons' natural homing behavior, to optimize intrusion detection performance. The choice of PIO stemmed from its adaptability and exploratory nature, mirroring pigeons' collective movements to improve the model's capability to identify and categorize network intrusions.

**3. DNN Model Implementation:** Our approach involved using a Dense Neural Network (DNN), or Multi-Layer Perceptron (MLP), tailored for precise recognition with classification of system intrusions, leveraging the specific features of the CICIDS-2017 dataset.

**4. Model Training and Validation:** The PIO-enhanced DNN was trained with a segment of the CICIDS-2017 dataset. We evaluated the model's efficacy in detecting and classifying network intrusions using a distinct subset of the dataset, including validation and test components.

**5. Hyper parameter Refinement**: We optimized key hyper parameters of the DNN architecture, such as neuron count, layer quantity, and learning rate, through an iterative testing process. This approach aimed to fine-tune the model's responsiveness to the unique elements of the CICIDS-2017 dataset.

**6. Feature Selection and Enhancement:** The paper explored advanced feature optimization methods, like feature reduction, to boost the model's efficiency. Experiments in feature selection were conducted to govern the impression of altering feature sets experience the accurateness of the intrusion detection system.

**7. Applying PIO to Alternative Architectures:** We extended the use of PIO to other neural network put together, including Convolutional Neural Networks (CNN) and other forms of Deep Neural Networks (DNN). This part of the research assessed PIO's adaptability and effectiveness across different model frameworks.

The methodology of this paper was methodical and thorough, integrating bio-inspired optimization with deep learning techniques to advance intrusion detection capabilities using the CICIDS-2017 dataset. The iterative nature of the experiments continuously propelled the enhancement of the model's performance.

### 4. Architectures Developed

#### A. Convolutional Neural Network (CNN)

A Convolutional Neural Network (CNN) stands as a focused type of deep learning framework, explicitly developed to process data arranged in structured grids. It excels primarily in the analysis of visual imagery, making it an ideal choice for applications such as image categorization, object recognition, and segmenting images. The following sections provide a detailed exploration of CNNs, including relevant mathematical formulations where necessary:

**Convolutional Layer:**

1 **Convolution Operation:**

Let's represent a two-dimensional input as Ά and a filter as $F$. The convolution operation can be written $\Delta s$ :

$$(F \otimes Ά)(y,x) = \sum_{p} \sum_{y} Ά(x-q, y-p)F(q,p)$$

2 **Filters and Feature Maps:**

When a specific filter $F$ is applied, the resulting feature map $M$ can be expressed as:

$$M(x,y) = \phi((F \otimes Ά)(x,y) + c)$$

Where, $\phi$ is the activation-function and represents the bias.

**3. Pooling Layer:**

For max-pooling operation:

$$P(x,y) = \max_{PM} M(2x+p, 2y+q)$$

4 **Activation Function:**

Using ReLU as an example for non-linearity:

$$\text{Re } UJ(\mathbb{z}) = \max(0, \mathbb{z})$$

5 **Fully Connected Layers:**

Flattening the feature maps and applying fully connected layers can be expressed as:

$$Y = \phi(VX + d)$$

Here, $V$ and $d$ are the weights and biases of the fully connected layers, respectively.

**6. Output Layer:**

For Soft Max activation in multi-class classification:

**7. Back propagation and Training:**

The loss function for classification can be expressed as Weight and bias updates using a method like SGD or Adam can be represented as:

$$\Theta_{\text{urv}} = \Theta_{\text{ohl}} - \eta \nabla_e \text{ Loss}$$

8 **Regularization and Dropout**

For a dropout operation with probability $p$

$$\text{Output} = \frac{\text{Treut}}{1-p}$$

9 **Batch Normalization:**

Normalizing activation within a batch can be described as:

$$\text{BN}(\mathbb{z}) = \delta \frac{|\mathbb{z} - \mu|}{\sqrt{\alpha^2 + 1}} + \beta$$

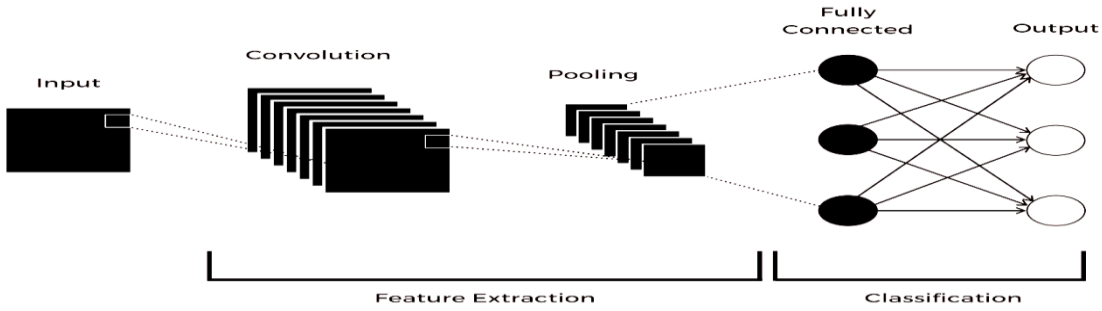


**Fig 01:** CNN architecture

**B. System design for anomaly detection using Deep Neural Networks (DNNs)**

Anomaly detection in Deep Neural Networks (DNNs) requires the development of a model that can identify out-of-the-ordinary patterns in input data. Here is a brief overview of a Feed-Forward Deep Neural Network (FF-DNN) for anomaly detection, along with some important mathematical equations:

1 **Input Layer Structure:**
Representing the input as $Z \in \mathbb{R}^{a \times b}$, a and b are the characteristics and sample size, respectively.

2 **Hidden Layer Dynamics:**
For the $i$-th hidden layer:

- Linear Transformation: $H^{[\text{ı}]} = W^{[\text{ı}]} \cdot G^{[\text{ı}-1]} + e^{[\text{ı}]}$
- Activation: $G^{[\text{ı}]} - \psi\left(H^{[\text{ı}]}\right)$
  Here, $W^{[i]}$ is the weightiness matrix, $e^{[\text{ı}]}$ the preference vector, $G^{[\text{ı}-1]}$ the output of the previous layer, and $\psi$ an activation function (like ReLU).

**3 Output Layer Formulation (for binary classification tasks):**

- Final Layer Computation: $H^{[L]} - W^{[L]} \cdot G^{[L-1]} + e^{[L]}$
- Activation: $G^{[L]} - \theta\left(H^{[L]}\right)$
  Here, $\theta$ is the sigmoid function, and $W^{[L]}, e^{[L]}$ are the weights and biases for the output layer.

**4 Loss Function for Binary Classification:**

- The loss can be described as:

$$Loss = -\frac{1}{b}\sum_{\text{ȷ}-1}^{b} \left[y^{(\text{ȷ})} \log\left(\hat{y}^{(\text{ȷ})}\right) + \left(1 - y^{(\text{ȷ})}\right) \log\left(1 - \hat{y}^{(\text{ȷ})}\right)\right]$$

Where $y^{(\text{ȷ})}$ is the true label, and $\hat{y}^{(\text{ȷ})}$ is the predicted probability.

5. Backpropagation Process:

- Gradient Computation:

$$\delta H^{[\text{ı}]} - G^{[\text{ı}]} - Y$$
$$\delta W^{[\text{ı}]} - \frac{1}{b} \cdot \delta H^{[\text{ı}]} \cdot \left(G^{[\text{ı}-1]}\right)^{\text{T}}$$
$$\delta e^{[\text{ı}]} - \frac{1}{b} \sum \delta H^{[\text{ı}]}$$
$$\delta G^{[\text{ı}-1]} - \left(W^{[\text{ı}]}\right)^{\text{T}} \cdot \delta H^{[\text{ı}]}$$

- Updating Parameters via Gradient Descent:

$$W^{[\imath]} \leftarrow W^{[\imath]} - \alpha \cdot \delta W^{[\imath]}$$
$$e^{[\imath]} \leftarrow e^{[\imath]} - \alpha \cdot \delta e^{[\imath]}$$

Here, $\alpha$ is the learning rate.

- $\gamma$ is the learning rate.

This mathematical model outlines the FF-DNN architecture for anomaly detection. It involves forward and backward passes, with parameter updates to minimize the cross-entropy loss, enabling the network to effectively identify anomalies in the input data.

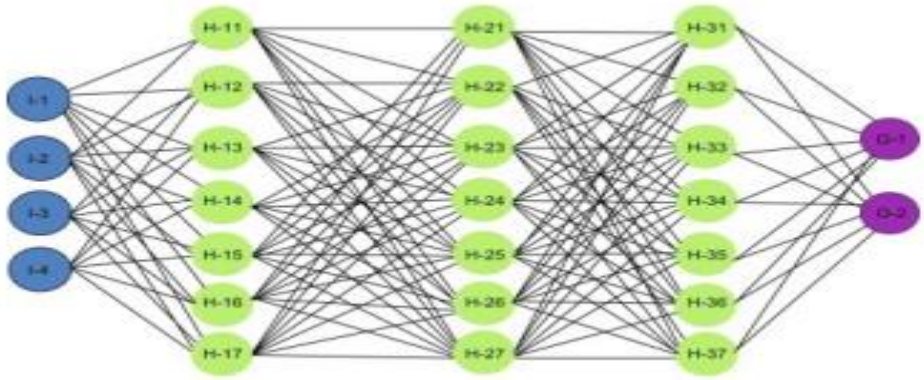


**Fig 2:** DNN architecture

### C. Architecture of Dense-Neural Networks

Layers of the model, each with its own set of mathematical operations and activation functions, make up the Dense Neural Network (Dense-NN) architecture that was designed for intrusion detection using the CIC-IDS-2017 dataset. A mathematical explanation of this design is given below:

**Notations and Definitions:**

- Represent the input data as $P \in \mathbb{R}^{u \times v}$, where $u$ denotes the feature count, and $v$ represents data instances.
- Around the $\bar{\imath}$-th layer, let $V^{[\bar{\imath}]}$ happen the weight matrix with $c^{[\bar{\imath}]}$ the preference vector.
- $Y^{[\bar{\imath}]}$ signifies the linear output at layer $i$, and $B^{[\bar{\imath}]}$ is the resulting activation.
- Activation function $h(\cdot)$ varies: ReLU for hidden layers, and sigmoid or softmax for the output layer, based on the classification task.

Architectural Components:

1 Input Layer:

- $Y^{[1]} = V^{[1]} \cdot P + c^{[1]}$
- $B^{[1]} = h\left(Y^{[1]}\right)$

2 Hidden Dense Layers:

- For each intermediate layer $\bar{\imath}$, where $\bar{\imath} = 2,3,\dots,I-1$ : $\cdot Y^{[\bar{\imath}]} = V^{[\bar{\imath}]} \cdot B^{[\bar{\imath}-1]} + c^{[\bar{\imath}]}$
- $B^{[\bar{\imath}]} = h\left(Y^{[\bar{\imath}]}\right)$

3 Output Layer:

- $Y^{[\Upsilon]} = V^{[\Upsilon]} \cdot B^{[\Upsilon-1]} + c^{[\Upsilon]}$
- $B^{[\Upsilon]} = \omega\left(Y^{[\Upsilon]}\right)$
- Here, $\omega(\cdot)$ is either sigmoid for binary classification or softmax for multi-class classification.

Loss Functions:

- Binary Classification:

$$\cdot J = -\frac{1}{n}\sum_{\bar{\imath}=1}^{\bar{\imath}}\left[z^{(\bar{\imath})}\log\left(\hat{z}^{(\bar{\imath})}\right) + \left(1 - z^{(\bar{\imath})}\right)\log\left(1 - \hat{z}^{(\bar{\imath})}\right)\right]$$

- Multi-Class Classification:
- $J = -\frac{1}{v}\sum_{\bar{\imath}=1}^{v}\sum_{d=1}^{D} z_d^{(\bar{\imath})}\log\left(\hat{z}_d^{(\bar{\imath})}\right)$
- $D$ is the total number of classes.

Back propagation and Parameter Update:

- Compute gradients for back propagation and update parameters using gradient descent:
- $V^{[\bar{\imath}]} = V^{[\bar{\imath}]} - \beta \frac{\partial J}{\partial V^{[\bar{\imath}]}}$
- $c^{[\bar{\imath}]} = c^{[\bar{\imath}]} - \beta \frac{\partial J}{\partial e^{[\bar{\imath}]}}$
- $\beta$ is the learning rate.

This mathematical framework outlines the Dense-NN architecture designed for intrusion detection. It encapsulates the sequential process from the input layer through multiple dense layers to the output, incorporating activation functions, and details the

training methodology including loss functions and back propagation for parameter updates.

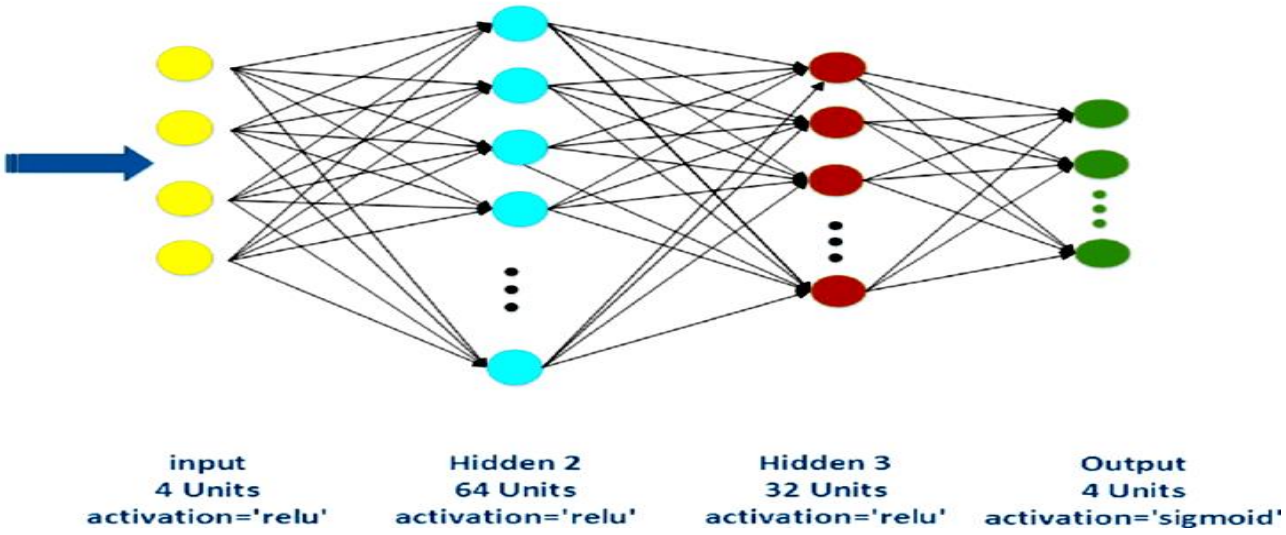


**Fig 3:** Dense-NN Architecture

### D. Pigeon-Inspired Optimization for Feature Selection (PIO)

To implement Pigeon-Inspired Optimization (PIO) for feature selection, one must first formulate the process in terms of equations that correctly represent the selection and modification of features during optimization. Here is a simplified version of the Feature Selection procedure utilizing PIO:

Notations:

- Let $B$ represent the feature matrix.
- Define $G(B)$ as the objective function for optimization.
- Denote the pigeon population as $Q$.
- $B_i$ signifies the position of pigeon $i$ in the feature space.
- $T_i$ indicates the velocity of pigeon $i$.
- $B_{\text{opt}}$ is the globally best solution.
- "rndm" is a uniform random number within [0,1].

Feature Selection Objective Function:

- The objective function $G(B)$ evaluates the quality of a feature subset, balancing factors like classification accuracy and model complexity. The aim is to maximize $G(B)$ : $G(B)$ - Classification Accuracy - Model Complexity

Mathematical Model for Pigeon-Inspired Optimization:

- Update Velocity $(T_i)$ :

$$T_i(t+1) - T_i(t) \cdot e^{-Mt} + rndm \cdot \left(B_{\text{opt}} - B_i(t)\right)$$

- Update Position $(B_i)$ :

$$B_i(t+1) = B_i(t) + T_i(t+1)$$

Feature Selection with PIO:

- Consider $C$ as the binary vector representing feature selection.
- Initialize $C$ either randomly or based on a heuristic.
- In each iteration:

$$C_i(t+1) - \text{PIO}\left(C_i(t), T_i(t+1), C_{\text{opt}}, M, \text{'rndm"}\right)$$

Pigeon-Inspired Feature Selection (PIO) Function:

- PIO Function:

$$\text{PIO}\left(C_i(t), T_i(t+1), C_{\text{opt}}, M, \text{"rndm"}\right) - C_i(t) + T_i(t+1)$$

Iterative Feature Selection Process:

- Initialize $B_i$ and $T_i$ for each pigeon.
- Proceed through generations until convergence:
- Update $T_i$ and $B_i$ using PIO.
- Evaluate $G(B_i)$.
- If a better solution is found, update $B_{\text{opt-}}$ .

Feature Subset Selection:

- Post-optimization, the final feature subset is selected by applying a threshold to the real-valued positions from PIO.

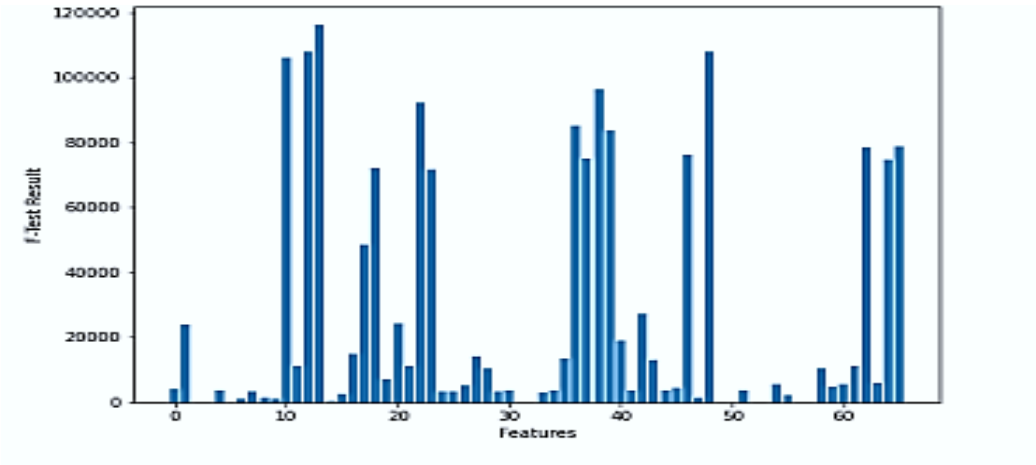


**Fig 4:** Features selected by PIO

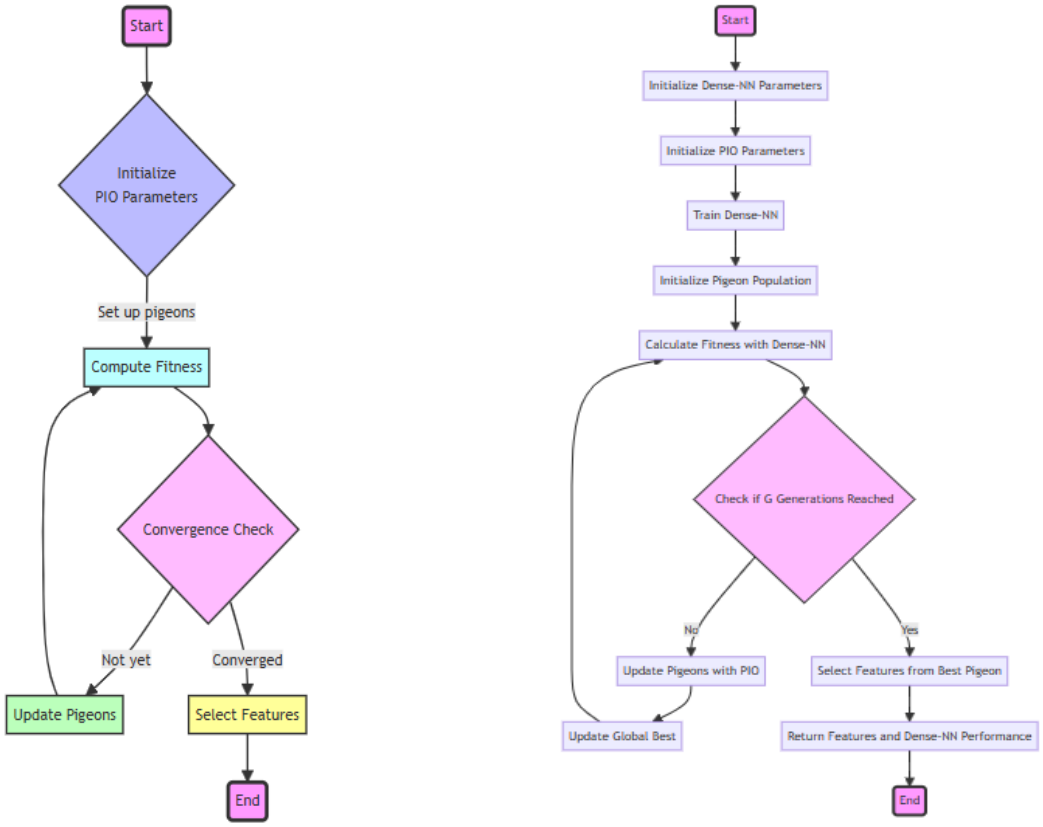


**Fig 5:** (a)Features selection by PIO ; (b) Dense-NN with PIO

**Algorithm: Dense-NN with PIO for Feature Selection**

**Initialization:**

1 Initialize the Dense-NN architecture parameters (weights and biases) randomly.

2 Initialize PIO parameters: ***R*** (map and compass factor), ***N*** (number of pigeons), ***G*** (maximum number of iterations).
Dense-NN Training:

3 Train the Dense-NN using the selected features.

- Use the standard back propagation algorithm to update weights and biases.

**PIO Optimization for Feature Selection:**

4. Initialize the population of pigeons with random positions representing features. Pigeon $_i$ = $[f_1, f_2, \ldots, f_n]$, $i - 1, 2, \ldots, N$ where $f_j \in \{0, 1\}$ represents the selection status of feature $j$.

1 Evaluate the fitness of each pigeon using the Dense-NN performance. Fitness $_i$ - Dense-NN_Performance (Pigeon $_i$)

2 Identify the global best position $(X_g)$ among all pigeons based on fitness.

3 Iterate through $G$ generations:

**Update the velocity and position of each pigeon using PIO equations:**

$$U_i(t+1) - U_i(t) \cdot e^{-Itt} + \text{rand} \cdot (X_g - \text{Pigeon}_i)$$
$$\text{Pigeon}_i(t+1) - \text{Pigeon}_i(t) + U_i(t+1)$$

4 After $G$ iterations, the final selected features are based on the best pigeon's position. Output:

5 Return the selected features and the corresponding Dense-NN performance.

**End**

## 4. Results and Discussion

In this research, we undertake an exhaustive evaluation of various neural network architectures, including Convolutional Neural Network (CNN), Deep Neural Network (DNN), and Dense Neural Network (Dense-NN), using the CIC-IDS-2017 dataset. The aim is to deliver a detailed examination of the performance characteristics of each model. The following sections detail and analyze the evaluation metrics applied to each of these architectures.

**Architecture of Convolutional Neural Network (CNN) Evaluation Measures:**

**Loss:** The minimal loss number (0.028) signifies precise predictions, demonstrating the model's efficacy.

**Accuracy:** The exceptional precision of 99.68% indicates a strong level of overall correctness.

**Report on Classification:**

The CNN demonstrates exceptional performance in accurately categorizing specific classes (such as "0," "2," "4," and "10") with elevated levels of precision, recall, and F1-score metrics.

Nevertheless, there is scope for enhancing the identification of particular categories (such as "1," "5," "6," "8," "9," "11," "12," "13," and "14") when the recall rates are comparatively lower.

**Summary:** The CNN exhibits a high level of accuracy, especially in specific categories. Nevertheless, specific enhancements are required for classes that have lower recall rates. Optimizing the model architecture or training approach can improve its performance.

**Table-03:** CNN Architecture Evaluation Metrics

| Class | Precision | Recall | F1-Score |
|---|---|---|---|
| 0 | 1 | 1 | 1 |
| 1 | 0.99 | 0.42 | 0.59 |
| 2 | 1 | 0.99 | 1 |
| 3 | 0.99 | 1 | 1 |
| 4 | 1 | 1 | 1 |
| 5 | 0.99 | 0.94 | 0.96 |
| 6 | 1 | 0.88 | 0.93 |
| 7 | 0.96 | 1 | 0.98 |
| 8 | 1 | 0.5 | 0.67 |

| | | | |
|---|---|---|---|
| 9 | 1 | 0.29 | 0.44 |
| 10 | 0.99 | 0.99 | 0.99 |
| 11 | 0.99 | 0.51 | 0.67 |
| 12 | 0.71 | 0.99 | 0.83 |
| 13 | 0.5 | 0.25 | 0.33 |
| 14 | 0.91 | 0.08 | 0.14 |

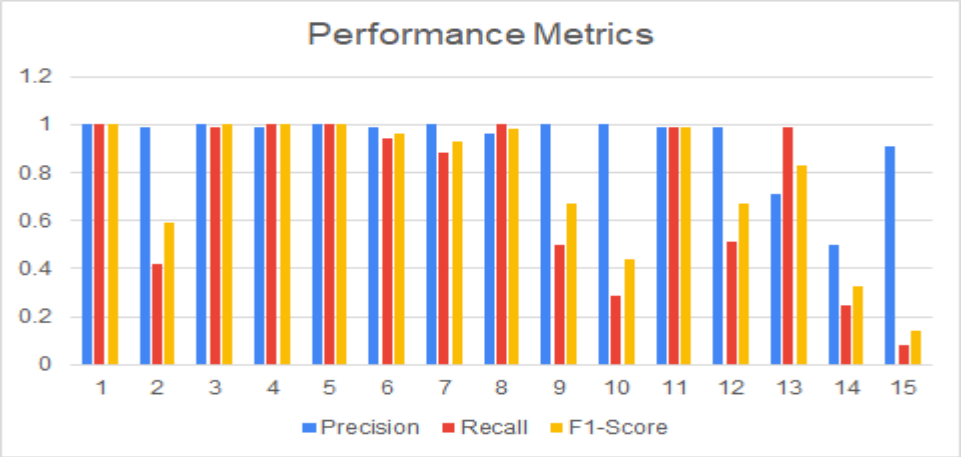


**Fig 6:** Performance Metrics obtained by CNN

**Architecture of a Deep Neural Network (DNN) Evaluation Metrics for Performance (Using All Features):**

**Accuracy:** The model achieved an exceptional accuracy of 99.82% when trained with all features, demonstrating the effective use of the whole feature set.

**Selection of features:**

The DNN model exhibits sustained high accuracy even when the number of features is decreased. The model demonstrates its versatility and the significance of feature selection by achieving a remarkable accuracy of 99.86% while using a smaller feature set consisting of either 38, 40, or 42 features. The DNN demonstrates its adaptability by effectively utilizing the entire feature set and achieving impressive accuracy even with a limited number of features. The paper highlights the importance of feature selection in optimizing the model.

**Table-04:** DNN Architecture Evaluation Metrics (All Features)

| Number of Features | Accuracy |
|---|---|
| All Features | 99.82% |
| 10 | 96.13% |
| 15 | 97.48% |
| 20 | 97.52% |
| 25 | 97.53% |
| 30 | 97.78% |
| 35 | 99.72% |
| 36 | 99.79% |
| 38 | 99.86% |
| 40 | 99.86% |
| 42 | 99.83% |

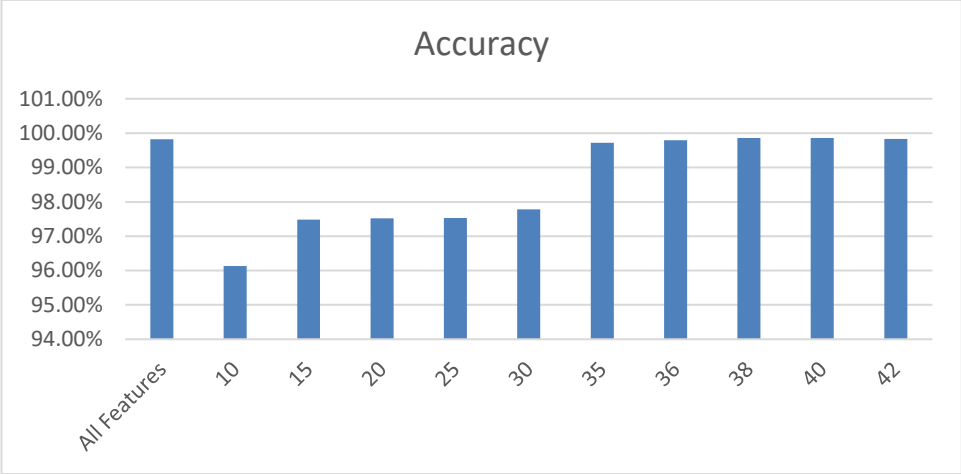


**Fig 7:** Performance Metrics obtained by DNN

**Architecture of a Dense Neural Network (DNN) Evaluation Metrics for Performance (Using All Features):**

The architecture of the Dense Neural Network (Dense-NN) is characterized by its high level of connectivity across layers. The performance of the Dense-NN is evaluated using many metrics.

**Loss:** The minimal loss number (0.005) signifies precise forecasts.

**Precision:** Demonstrating an outstanding precision level of 99.91%, indicating a remarkable degree of correctness.

**Report on Classification:**

The Dense-NN demonstrates exceptional performance in accurately categorizing many classes, achieving excellent precision, recall, and F1-score metrics.

Specific classes (such as "1," "5," "6," "9," "12," "13," and "14") exhibit potential for enhancement, as evidenced by lower recall ratings.

**Summary:** The Dense-NN has excellent accuracy and precision across multiple classes, and has the potential to improve recall for certain classes. Additional model refinements may be implemented to tackle these difficulties.

**Table-05:** Dense-NN Architecture Evaluation Metrics

| Class | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0.38 | 1 | 0.55 | 0.73 |
| 2 | 1 | 1 | 1 | 1 |
| 3 | 1 | 1 | 1 | 1 |
| 4 | 1 | 1 | 1 | 1 |
| 5 | 0.98 | 0.99 | 0.99 | 0.99 |
| 6 | 1 | 0.99 | 0.99 | 0.99 |
| 7 | 1 | 1 | 1 | 1 |
| 8 | 1 | 1 | 1 | 1 |
| 9 | 0.96 | 0.76 | 0.85 | 0.8 |
| 10 | 0.99 | 1 | 1 | 1 |
| 11 | 0.99 | 0.98 | 0.98 | 0.98 |
| 12 | 0.7 | 0.99 | 0.82 | 0.89 |
| 13 | 0.57 | 0.24 | 0.33 | 0.29 |

| | | | | |
|---|---|---|---|---|
| 14 | 1 | 0.05 | 0.1 | 0.02 |

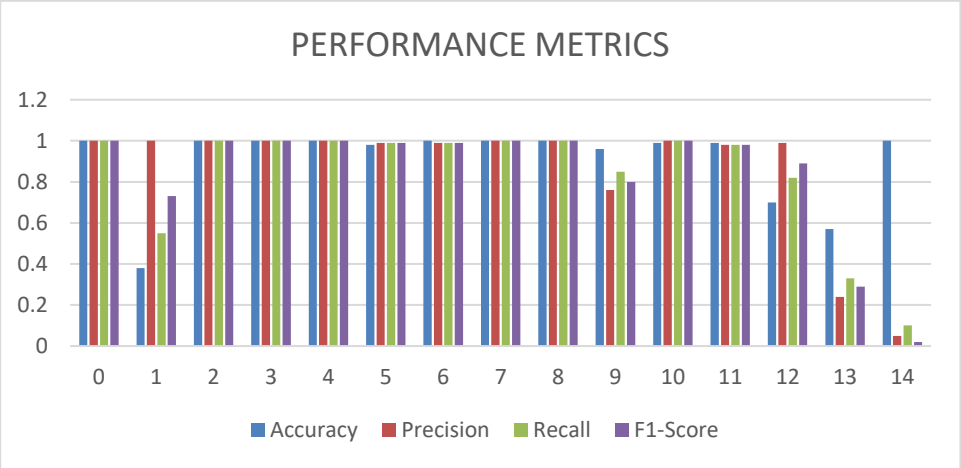


**Fig 8:** Performance Metrics obtained by Dense NN + PIO

**Comparison Summary:**

**CNN Architecture:** Achieves high overall accuracy (99.68%) with strengths in some classes but space for development in others.

**DNN Architecture (All Features):** Demonstrates outstanding accuracy (99.82%) employing all features, suggesting good exploitation of the whole feature set.

**Dense-NN Architecture with PIO Optimization for Feature Selection:** Adaptable, attaining high accuracy (99.91%) with a smaller feature set, demonstrating the importance of feature selection.

## 5. Conclusion

In the assessment of intrusion detection through three neural network configurations using the CIC-IDS-2017 dataset, the Convolutional Neural Network (CNN) exhibits resilience, achieving an accuracy of 99.68%. However, there is room for development in certain intrusion classes. The Deep Neural Network (DNN) excels with 99.82% accuracy but raises concern about computational complexity, necessitating a careful consideration of practicality. A significant leap comes from the application of Pigeon-Inspired Optimization (PIO) to enhance the architecture of Dense Neural Networks (Dense-NNs) for feature selection. This innovative model attains an impressive accuracy of 99.91% with a reduced set of features, showcasing remarkable adaptability. Dense-NN with PIO emphasizes the critical role of feature selection in optimizing both accuracy and efficiency. Each architectural style presents its unique strengths and drawbacks. CNNs demonstrate superior accuracy, DNNs prove effective with abundant features, while Dense-NN with PIO shines when dealing with a limited feature set. This nuanced understanding provides valuable insights for the enhancement of intrusion detection systems. PIO introduces a new dimension to feature selection, showcasing the potential of bio-inspired optimization in expanding the horizons of cyber security research. In conclusion, continuous refinement of designs, exploration of novel optimization methods, and validation in real-world scenarios are pivotal for the future of this field. This paper offers a well-balanced assessment of the advantages and disadvantages associated with diverse neural network approaches, paving the way for further research at the intersection of nature-inspired optimization and cyber security.

**Conflict of interest:** The authors affirm that they have no competing interests.

## References

[1] T. Wisanwanichthan and M. Thammawichai, "A Double-Layered Hybrid Approach for Network Intrusion Detection System Using Combined Naive Bayes and SVM," in IEEE Access, vol. 9, pp. 138432-138450, 2021.

[2] Mausumi Das Nath, TapalinaBhattasali. (2020). Anomaly Detection Using Machine Learning Approaches. Azerbaijan Journal of High-Performance Computing. 3 (2), p196-206.

[3] Ying Zhong, Wenqi Chen, Zhiliang Wang, Yifan Chen, Kai Wang, Yahui Li, Xia Yin, Xingang Shi, Jiahai Yang, Keqin Li, HELAD: A novel network anomaly detection model based on heterogeneous ensemble learning, Computer Networks, Volume 169, 2020, 107049, ISSN 0167-4048, https://doi.org/10.1016/j.comnet.2020.101752.

[4] Z. R. Zeng, W. Peng and D. Zeng, "Improving the Stability of Intrusion Detection with Causal Deep Learning," in IEEE Transactions on Network and Service Management, 2022, doi: 10.1109/TNSM.2022.3193099.

[5] A.Tajbakhsh, M. Rahmati, and A. Mirzaei, “Intrusion detection using fuzzy association rules,” Applied Soft Computing, vol. 9, no. 2, pp. 462–469, 2019.

[6] P. Barnard, N. Marchetti, and L. A. DaSilva, "Robust Network Intrusion Detection Through Explainable Artificial Intelligence (XAI)," in IEEE Networking Letters, vol. 4, no. 3, pp. 167-171, Sept. 2022, doi: 10.1109/LNET.2022.3186589.

[7] H. Benaddi, K. Ibrahimi, A. Benslimane, M. Jouhari and J. Qadir, "Robust Enhancement of Intrusion Detection Systems Using Deep Reinforcement Learning and Stochastic Game," in IEEE Transactions on Vehicular Technology, vol. 71, no. 10, pp. 11089-11102, Oct. 2022, doi: 10.1109/TVT.2022.3186834.

[8] Redhwan Al-amri, Raja Kumar Murugesan, Mustafa Man, Alaa Fareed Abdulateef, Mohammed A. Al-Sharafi and Ammar Ahmed Alkahtani. (2021). A Review of Machine Learning and Deep Learning Techniques for Anomaly Detection in IoT Data. MDPI, p1-23.

[9] Zhu, M., Ye, K., & Xu, C.-Z. (2018). Network Anomaly Detection and Identification Based on Deep Learning Methods. Cloud Computing – CLOUD 2018, 219–234. doi:10.1007/978-3-319-94295-7_15.

[10] Samir Ifzarne, Hiba Tabbaa, ImadHafidi, NidalLamghari. (2021). Anomaly Detection using Machine Learning Techniques in Wireless Sensor Networks. The International Conference on Mathematics & Data Science (ICMDS). 1743, p1-14.

[11] Tama, B.A., Comuzzi, M., & Rhee, K.H. (2019). TSE-IDS: A Two-Stage Classifier Ensemble for Intelligent Anomaly-Based Intrusion Detection System. IEEE Access, 7, 94497-94507.

[12] A. Tajbakhsh, M. Rahmati, and A. Mirzaei, “Intrusion detection using fuzzy association rules,” Applied Soft Computing, vol. 9, no. 2, pp. 462–469, 2019.

[13] Saranya Kunasekaran1 and ChellammalSuriyanarayanan. (2020). Anomaly detection techniques for streaming data–An overview. Malaya Journal of Matematik. S (1), p703-710.

[14] Xu, Wen & Jang-Jaccard, Julian & Singh, Amardeep & Wei, Yuanyuan & Sabrina, Fariza. (2021). Improving Performance of Autoencoder-Based Network Anomaly Detection on NSL-KDD Dataset. IEEE Access. PP. 1-1. 10.1109/ACCESS.2021.3116612.

[15] Oseni et al., "An Explainable Deep Learning Framework for Resilient Intrusion Detection in IoT-Enabled Transportation Networks," in IEEE Transactions on Intelligent Transportation Systems, 2022, doi: 10.109/TITS.2022.3188671.

[16] S. Subbiah, K. S. M. Anbananthen, S. Thangaraj, S. Kannan and D. Chelliah, "Intrusion detection technique in wireless sensor network using grid search random forest with Boruta feature selection algorithm," in Journal of Communications and Networks, vol. 24, no. 2, pp. 264-273, April 2022, doi: 10.23919/JCN.2022.000002.

[17] Chiranjit Das, Akhtar Rasool, Aditya Dubey and Nilay Khare, “Analyzing the Performance of Anomaly Detection Algorithms” International Journal of Advanced Computer Science and Applications(IJACSA),12(6), 2021. http://dx.doi.org/10.14569/IJACSA.2021.0120649.

[18] Chiranjit Das1, Akhtar Rasool2, Aditya Dubey3, NilayKhare. (2021). Analyzing the Performance of Anomaly Detection Algorithms. (IJACSA) International Journal of Advanced Computer Science and Applications. 12 (6), p439-445.

[19] Z. A. E. Houda, B. Brik and L. Khoukhi, "“Why Should I Trust Your IDS?”: An Explainable Deep Learning Framework for Intrusion Detection Systems in Internet of Things Networks," in IEEE Open Journal of the Communications Society, vol. 3, pp. 1164-1176, 2022, doi: 10.1109/OJCOMS.2022.3188750.

[20] S. Das et al., "Network Intrusion Detection and Comparative Analysis using Ensemble Machine Learning and Feature Selection," in IEEE Transactions on Network and Service Management.

[21] Dhanush P.M. Naik1, I. Rohit Satya2, Chaitra B.H.3, VishalakshiPrabhu H. (2020). Anomaly Detection: Different Machine Learning Techniques, A Review. International Journal of Advanced Research in Computer and Communication Engineering. 9 (4), p1-5.

[22] J. Yoo, B. Min, S. Kim, D. Shin and D. Shin, "Paper on Network Intrusion Detection Method Using Discrete Pre-Processing Method and Convolution Neural Network," in IEEE Access, vol. 9, pp. 142348-142361, 2021, doi: 10.1109/ACCESS.2021.3120839.

[23] T. Wisanwanichthan and M. Thammawichai, "A Double-Layered Hybrid Approach for Network Intrusion Detection System Using Combined Naive Bayes and SVM," in IEEE Access, vol. 9, pp. 138432-138450, 2021.

[24] Chiranjit Das, Akhtar Rasool, Aditya Dubey and Nilay Khare, “Analyzing the Performance of Anomaly Detection Algorithms” International Journal of Advanced Computer Science and

Applications(IJACSA),12(6), 2021. http://dx.doi.org/10.14569/IJACSA.2021.0120649.

[25] Elmrabit, Nebrase; Zhou, Feixiang; Li, Fengyin; Zhou, Huiyu. (2022). Evaluation of Machine Learning Algorithms for Anomaly Detection. International Conference on Cyber Security and Protection of Digital Services (Cyber Security), p1-9.

[26] Andrey Kharitonova, AbdulrahmanNahhasa, Matthias Pohla, Klaus Turowskia. (2022). Comparative analysis of machine learning models for anomaly detection in manufacturing. Elsevier. 200, p1288–1297.

[27] Redhwan Al-amri, Raja Kumar Murugesan, Mustafa Man, Alaa Fareed Abdulateef, Mohammed A. Al-Sharafi and Ammar Ahmed Alkahtani. (2021). A Review of Machine Learning and Deep Learning Techniques for Anomaly Detection in IoT Data. MDPI, p1-23.

[28] Chiranjit Das1, Akhtar Rasool2, Aditya Dubey3, NilayKhare. (2021). Analyzing the Performance of Anomaly Detection Algorithms. (IJACSA) International Journal of Advanced Computer Science and Applications. 12 (6), p43-445.

[29] Samir Ifzarne, Hiba Tabbaa, ImadHafidi, NidalLamghari. (2021). Anomaly Detection using Machine Learning Techniques in Wireless Sensor Networks. The International Conference on Mathematics & Data Science (ICMDS). 1743, p1-14.

[30] Manimaran, A.; Chandramohan, D.; Shrinivas, S. G.; Arulkumar, N. (2020). A comprehensive novel model for network speech anomaly detection system using deep learning approach. International Journal of Speech Technology, p1-9.

[31] Mausumi Das Nath, TapalinaBhattasali. (2020). Anomaly Detection Using Machine Learning Approaches. Azerbaijan Journal of High-Performance Computing. 3 (2), p196-206.

[32] Saranya Kunasekaran1 and ChellammalSuriyanarayanan. (2020). Anomaly detection techniques for streaming data–An overview. Malaya Journal of Matematik. S (1), p703-710.

[33] Shane Brady, Damien Magoni, John Murphy, HaythamAssem, A. Omar Omar Portillo-Dominguez. (2020). Analysis of Machine Learning Techniques for Anomaly Detection in the Internet of Things, p1-7.

[34] Chuadhry Mujeeb Ahmed, Gauthama Raman M R, Aditya Mathur. (2020). Challenges in Machine Learning based approaches for Real-Time Anomaly Detection in Industrial Control Systems, p1-6.

[35] Dhanush P.M. Naik1, I. Rohit Satya2, Chaitra B.H.3, VishalakshiPrabhu H. (2020). Anomaly Detection: Different Machine Learning Techniques, A Review. International Journal of Advanced Research in Computer and Communication Engineering. 9 (4), p1-5.

[36] Sydney Mambwe Kasongo, Yanxia Sun, A deep learning method with wrapper based feature extraction for wireless intrusion detection system, Computers & Security, Volume 92, 2020, 101752, ISSN 0167-4048, https://doi.org/10.1016/j.cose.2020.101752.

[37] Ashfaq Khan, Muhammad & Karim, Rezaul & Kim, Yangwoo. (2019). A Scalable and Hybrid Intrusion Detection System Based on the Convolutional-LSTM Network. Symmetry. 11. 583. 10.3390/sym11040583.

[38] Ying Zhong, Wenqi Chen, Zhiliang Wang, Yifan Chen, Kai Wang, Yahui Li, Xia Yin, Xingang Shi, Jiahai Yang, Keqin Li, HELAD: A novel network anomaly detection model based on heterogeneous ensemble learning, Computer Networks, Volume 169, 2020, 107049, ISSN 1389-1286, https://doi.org/10.1016/j.comnet.2020.101752.

[39] Tama, B.A., Comuzzi, M., & Rhee, K.H. (2019). TSE-IDS: A Two-Stage Classifier Ensemble for Intelligent Anomaly-Based Intrusion Detection System. IEEE Access, 7, 94497-94507.

[40] A.Tajbakhsh, M. Rahmati, and A. Mirzaei, "Intrusion detection using fuzzy association rules," Applied Soft Computing, vol. 9, no. 2, pp. 462–469, 2019.

[41] Zhu, M., Ye, K., & Xu, C.-Z. (2018). Network Anomaly Detection and Identification Based on Deep Learning Methods. Cloud Computing – CLOUD 2018, 219–234. doi:10.1007/978-3-319-94295-7_15.

[42] TusharRakshe and Vishal Gonjari. (2017). Anomaly based Network Intrusion Detection using Machine Learning Techniques. . International Journal of Engineering Research & Technology (IJERT). 6 (5), p1-5.