Nanotechnology Perceptions
ISSN 1660-6795
www.nano-ntp.com

Intrusion Detection System Using WSN Novel Deep Learning Algorithms

Er. Priyanka Pande[1], Dr. Harsh Mathur[2], Dr. Lalit Kumar Gupta[3]
[1]*Research Scholar,Department of Computer Science & Engineering,Rabindranath Tagore University, Bhopal, M.P. - 464993, India*
E-mail:priyankapande05@gmail.com
[2]*Associate Professor, Department of Computer Science & Engineering, Rabindranath Tagore University, Bhopal, M.P. - 464993, India*
E-mail: harshmathur786@gmail.com
[3]*Assistant Professor, Department of Computer Science & Engineering, Bundelkhand University, Jhansi Uttar Pradesh – 284128, India*
E-mail:dr.lalitgupta.bu@gmail.com

Protecting networks and systems against malicious cyber activity is of the utmost significance in the rapidly changing and highly linked digital environment of today. Intrusion Detection Systems, often known as IDS, are very important tools for determining whether or not actions on a computer network are authorised or malicious. Despite the fact that classic rule- based IDS have been shown to be successful, the ever-increasing complexity of cyber threats requires solutions that are more adaptable and intelligent. The creation of an intrusion detection system that makes use of machine learning methods to improve detection capabilities is the focus of this abstract. The article discusses the process of data collecting and pre-processing, the extraction of features, the training and evaluation of machine learning models, the integration of real-time data, and the ongoing development of the system to combat new threats. It discusses the difficulties and factors to take into account while developing such a system and shows some real-world instances of how it may be put to use. This IDS paradigm gives a viable option for enhancing network security in the face of growing cyber hazards by using the power of machine learning. This is an important step that has to be taken. Several machine learning algorithms have been applied to the BoT-IoT dataset and the performance parameters are better for our proposed 1D-CNN algorithm. The MSE, Accuracy, Recall, Precision, and F1-score of the proposed algorithm are 3.0782%, 96.9218%, 96.9218%, 97.1816%, and 96.9688% respectively.

Keywords: Intrusion Detection System, Machine Learning, Cybersecurity, Network Security. Anomaly Detection, Threat Detection, Data Preprocessing, Feature Extraction, Supervised Learning. Unsupervised Learning.

1. Introduction

Protecting the integrity and security of these assets has become a top priority in today's digitally linked world as information travels across networks and systems. Intrusion Detection Systems (IDS), created to identify and counteract unauthorised or hostile actions inside computer networks and systems, have become essential elements of cybersecurity measures. IDS concentrates on locating and reacting to possible threats that have gotten past the perimeter defences, in contrast to conventional perimeter security systems that put their attention on preventing unauthorised access [1-2].

A. Definition and Purpose

A software or hardware solution known as an intrusion detection system (IDS) analyses network and system activity to look for indications of unauthorised or malicious behaviour. By continually monitoring network traffic, system logs, and user activity for any aberrant patterns that could point to a security breach, an IDS primarily serves to offer another layer of defence. IDS can be broadly categorized into two main types: Signature-Based Detection (Misuse Detection): This approach involves comparing incoming data with known patterns or signatures of known attacks. If a match is found, an alert is generated. Signature- based detection is effective against well-known attacks but might struggle with new or previously unseen threats.

Anomaly-Based Detection: Anomaly detection involves creating a baseline of normal network behavior and identifying deviations from this baseline. Unusual patterns or activities that deviate from the norm are flagged as potential intrusions. Anomaly-based detection is more adaptable to new threats but can result in false positives due to variations in legitimate behavior.

B. Challenges in Intrusion Detection

While Intrusion Detection Systems provide a valuable layer of defence against cyber threats, they are not without challenges. Some of these challenges include:

False Positives and False Negatives: Striking the right balance between detecting actual threats and minimizing 4 false alarms is a challenge. False positives (identifying legitimate activities as threats) and false negatives (failing to detect actual threats) can impact the effectiveness of an IDS.

Evolving Threat Landscape: Cyber threats are constantly evolving and becoming more sophisticated. IDS must keep up with new attack techniques, malware variants, and evasion methods to remain effective.

Complexity of Network Traffic: Modern networks generate vast amounts of data and traffic, making it challenging to distinguish between normal and malicious activities, especially in high-traffic environments.

Anomaly Detection's Complexity: Creating accurate baselines of normal behavior for anomaly-based detection can be difficult, and the system must adapt to legitimate changes in network behavior.

Privacy and Data Protection: IDS involves monitoring and analysing network traffic and user activities, which raises concerns about user privacy and the protection of sensitive data.

Resource Consumption: IDS can consume significant computational resources, impacting network performance and system responsiveness.

Integration with Security Ecosystem: Integrating IDS with other security tools and processes to form a comprehensive security ecosystem can be complex and require careful planning.

In Intrusion Detection Systems play a critical role in identifying potential security breaches and abnormal activities within computer networks, By understanding their definition, purpose, and the challenges they face, organizations can better appreciate the importance of deploying effective IDS solutions to enhance their overall cyber-security posture.

2. TRADITIONAL VS. MACHINE LEARNING-BASED IDS

The topic of cybersecurity is always evolving, and with it come new approaches for identifying threats and reducing their effects. conventional rule-based Intrusion Detection Systems (IDS) have been successful to a certain degree; however, the growing complexity of cyber threats has led to the study of more sophisticated techniques, including machine learning-based IDS. This is due to the fact that conventional IDS have been able to keep up with the evolving nature of cyber threats [3].

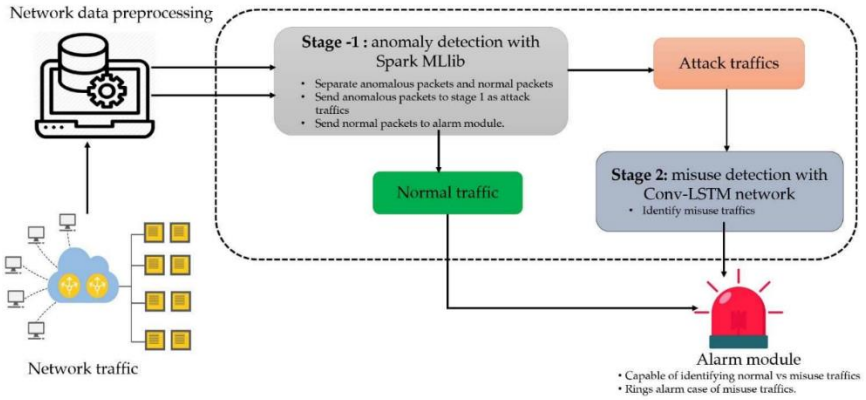

Figure 1: Graphical Design of IDS

A. Limitations of Rule-Based Systems:

Rule-based IDS rely on predefined patterns, signatures, or rules that characterize known attack patterns. While they have their merits, they come with several limitations:

Limited to Known Signatures: Rule-based systems are effective at detecting well-known 2 attacks for which signatures have been defined. However, they struggle to detect new or evolving attacks that don't match any predefined rules.

False Negatives: If an attack does not match any known rule. Rule-based IDS may fail to detect it, leading to false negatives.

Inflexibility: Maintaining and updating rule sets can be time- consuming and requires constant monitoring of emerging threats. This approach struggles to adapt to rapidly evolving attack techniques.

False Positives: Overly complex or broad rules can result in false positives, flagging legitimate activities as threats and inundating security teams with alerts.

Limited Context: Rule-based systems often lack the ability to consider the broader context of activities and connections, potentially leading to misinterpretation of benign activities.

B. Advantages of Machine Learning-Based Approaches:

Machine learning-based IDS leverage the capabilities of artificial intelligence to overcome some of the limitations of traditional rule- based systems. Here are some advantages:

Adaptability to New Threats: Machine learning models can learn from data and adapt to new and previously unseen attack patterns, making them more effective at detecting novel threats.

Anomaly Detection: Machine learning can excel at identifying anomalies in data, making it suitable for detecting sophisticated attacks that deviate from normal behavior.

Reduced False Positives: ML models can analyse vast amounts of data and discem complex patterns, leading to better detection accuracy and fewer false positives.

Contextual Analysis: Machine learning models can consider various attributes and contextual information when making decisions, improving the accuracy of threat detection.

Continuous Learning: Machine learning models can be updated with new data, allowing them to evolve and adapt to changing threat landscapes without requiring manual rule updates.

Efficient Resource Utilization: Machine learning models can optimize resource u sage, leading to improved performance and reduced impact on network and system operations.

Behavioral Profiling: ML-based IDS can create profiles of normal behavior, allowing them to identify deviations over time rather than relying solely on predefined rules.

DATA COLLECTION AND PREPROCESSING

The steps of data collecting and preprocessing are essential phases in building an effective Intrusion Detection System (IDS). When data is handled correctly, machine learning models can learn from relevant information and make accurate predictions [4].

A. Types of Data Sources

Network Traffic Logs: These logs capture network activities, including incoming and outgoing packets, source and destination IP addresses, port numbers, and protocols used.

System Logs: These logs record system-level events, such as login attempts, file access, process executions, and system configuration changes.

Application Logs: Application-specific logs provide insights into the behavior of specific software or services running on the network or system.

User Activities: User-related data, such as login patterns, authentication attempts, and user actions, can be important for identifying anomalies.

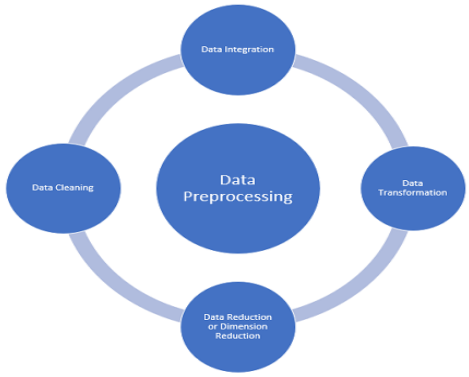

Figure 2: Data Pre-processing in IDS

B. Data Cleaning and Transformation

Data Cleaning: Remove duplicate entries, handle missing values, and address inconsistencies to ensure data quality.

Data Transformation: Normalize or scale numerical features to bring them to a common scale. Convert categorical features into numerical representations using techniques like one-hot encoding.

C. Feature Extraction and Selection:

Feature extraction involves converting raw data into meaningful features that can be used by machine learning models. Feature selection aims to identify the most relevant features to improve the model's efficiency and performance.

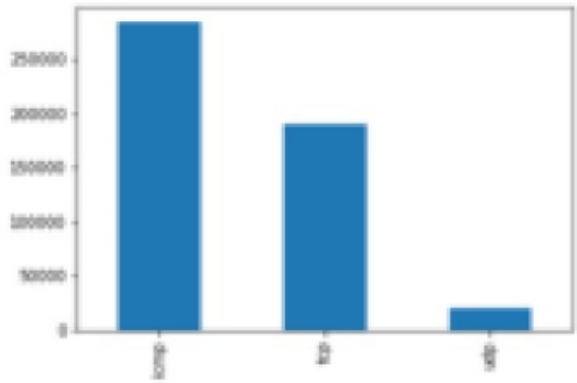

Figure 3: Histogram of Transformation

1) Important Features for IDS:

IP Addresses and Ports: Source and destination IP addresses and port numbers can help identify unusual communication patterns.

Protocol Types: Different proto- cols (TCP, UDP, ICMP) exhibit different behaviors, and analysing protocol distributions can reveal anomalies.

Packet Sizes: Anomalies in packet sizes might indicate data exfiltration or denial of service attacks.

Time Intervals: Detecting deviations in time intervals between actions can help identify

brute force attacks or unauthorized access.
User Behavior: Monitoring user activity patterns can help spot suspicious login attempts or unusual behaviors.
System Calls: For host-based IDS, tracking system calls and their frequencies can provide insights into potentially malicious activities.

2) *Techniques for Feature Engineering:*
Dimensionality Reduction: Principal Component Analysis (PCA) is one technique that can decrease the degree of features while keeping important information.
Aggregation: Group related data together to form aggregated features, such as counting failed login attempts within a time window.
Feature Scaling: Normalize features to the same range to prevent certain features from dominating others during model training.
Feature Interactions: Create new features by combining existing ones, such as the ratio of successful logins to failed logins.

3. LABELING DATA FOR TRAINING

Labeling data involves categorizing instances as either normal or representing some form of intrusion. This labelled data is used to train machine learning models to differentiate between normal and malicious activities within an Intrusion Detection System (IDS) [5-6].

A. Anomaly Detection vs. Signature-Based Detection
1) **Anomaly Detection:** Anomaly detection involves identifying deviations from normal behavior without relying on predefined attack signatures. Instances that significantly differ from the established normal baseline are flagged as anomalies. Effective for detecting previously unseen attacks, but may lead to false positives due to legitimate variations.
2) Signature-Based Detection: Signature-based detection re- lies on predefined patterns or signatures of known attacks. Instances matching these signatures are labelled as malicious. Effective for detecting well-known attacks, but limited to known threats and vulnerable to new attack variants.

B. Creating Labelled Datasets
1) Historical Data: Utilize historical intrusion data that has already been identitled and labelled. Machine learning models are trained using this data in order to identify similar patterns.
2) Manual Labeling: Manually label instances in a dataset as normal or intrusive. Requires domain expertise to accurately identify intrusions and normal activities.
3) Simulation: Set up controlled environments to simulate at tacks and normal behaviors. This approach provides labelled data for both normal and intrusive instances.
4) Hybrid Approaches: Combine historical data with manual labelling to ensure a comprehensive dataset. This approach helps bridge the gap between known and novel threats. Unlabelled Data with Semi-Supervised Learning:
Use a limited set of labelled data alongside a large of unlabelled data Semi-supervised learning leverages the labelled data to guide the model's learning from the unlabelled data.

4. DATA SPLITTING AND EVALUATION METRICS

In the process of developing an Intrusion Detection System (IDS) using machine learning, data splitting and evaluation metrics are crucial to ensure that the model's performance is accurately assessed and validated [7].

A. Training and Testing Sets

1) Training Set: A subset of the labelled dataset utilized for training the machine learning model is called the training set. From this data, the model learns the relationships between features and labels. It's important that the training set is representative of the data's distribution and includes a mix of normal and intrusive instances.

2) Testing Set: The testing set is a separate portion of the labelled dataset that the model has not seen during training. The model's performance is evaluated on this set to compute the degree to which it generalizes to current, untested data. The testing set helps detect overfitting, where the model performs well on training data but poorly on new data.

B. Performance Metrics for IDS

Evaluating the performance of an IDS is crucial to assess its ability to accurately identify intrusions and minimize false positives/negatives. Several performance metrics are commonly used:

1) Accuracy: The percentage of cases that are accurately classified is known as accuracy (both normal and intrusive) to the total instances in the testing set. It can be misleading in cases of imbalanced datasets where one class dominates

2) Precision: Precision calculates the proportion of correctly classified intrusive instances to all instances predicted as intrusive. Useful when false positives need to be minimized.

3) Recall (Sensitivity or True Positive Rate): Recall measures the proportion of correctly classified intrusive instances to all actual intrusive instances. Important when it's crucial to detect as many intrusions as possible, even at the cost of more false positives.

4) FI-Score: The Fl-score is calculated by taking the harmonic mean of recall and precision. It works well with unbalanced datasets because it strikes a compromise between precision and recall. Area Under Curve (AUC) and Receiver Operating Characteristic (ROC) Curve.

At different thresholds, the true positive rate (recall) is shown versus the false positive rate using a ROC curve. AUC, or area under the ROC curve, is a single statistic that can be used to evaluate the model's overall performance.

Confusion Matrix: The actual and expected classifications are shown in tabular form in a confusion matrix. It's a helpful tool for displaying the model's performance across several classes.

5. MACHINE LEARNING ALGORITHMS

Selecting the right machine learning algorithms is facial step in building an effective Intrusion Detection System (IDS). The choice of algorithms depends on the type of learning (supervised, unsupervised, or semi-supervised) and the characteristics of the data [8].

A. Supervised. Unsupervised, and Semi-Supervised Learning:

1) Supervised Learning: In supervised learning, the algorithm is trained on labelled data, where each instance is associated with its corresponding class or label. Common for signature-based detection and certain types of anomaly detection where labelled data is

available.

2) Unsupervised Learning: Unsupervised learning doesn't use labelled data. Instead, it identifies patterns and structures within the data without predefined categories Useful for anomaly detection when the types of intrusions are unknown.

3) Semi-Supervised Learning: Semi-supervised learning combines labelled and unlabelled data for training. Can be effective when labelled intrusion data is limited but can provide valuable guidance.

B. Algorithms for IDS:

1) Random Forest: Several decision trees are used in the Random Forest ensemble learning technique to increase accuracy and decrease overfitting, efficient for detection based on anomalies as well as signatures.

In practical decision-making applications, the most common classifiers or regression are used. The decision tree calculation is enlarged into RF calculation. The structure known as the "forest" or RF is made up of a variety of trees. The main difference between both directed learning models is that RF has a minimal level of complexity and is easy to implement by selecting the best arrangements based on the sacking strategy.

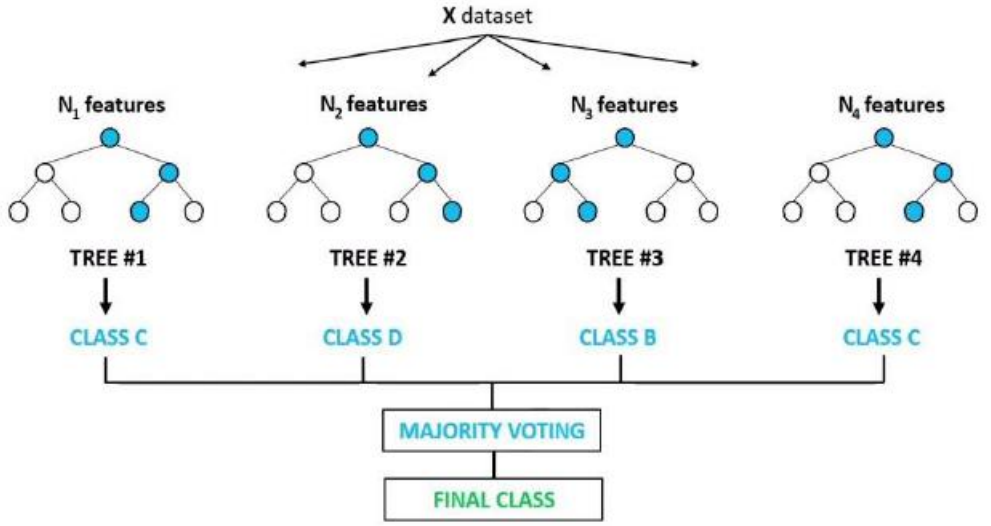

Figure 4: RF Classifier Algorithm

The main goal of using the trees in combination as a sorting approach is to achieve more accurate results in terms of prediction or misfortune accuracy. According to the image below, the classifier that uses the X-dataset should be divided into three different classes using RF. The RF classification algorithm is displayed in Figure 4 [9].

2) Decision Trees: Decision trees, which are frequently employed for classification and regression, are another supervised learning model. In contrast to the SVM approach, the DT is a non-parametric model. The DT structure that is best appropriate for business, risk management analysis is framed by the parent node with several offspring nodes (i.e. branches). Time complexity, which is quite high compared to other ML algorithms' preparation and testing phases, is DT's main drawback. Figure 5 depicts an example decision

tree classification structure [10].

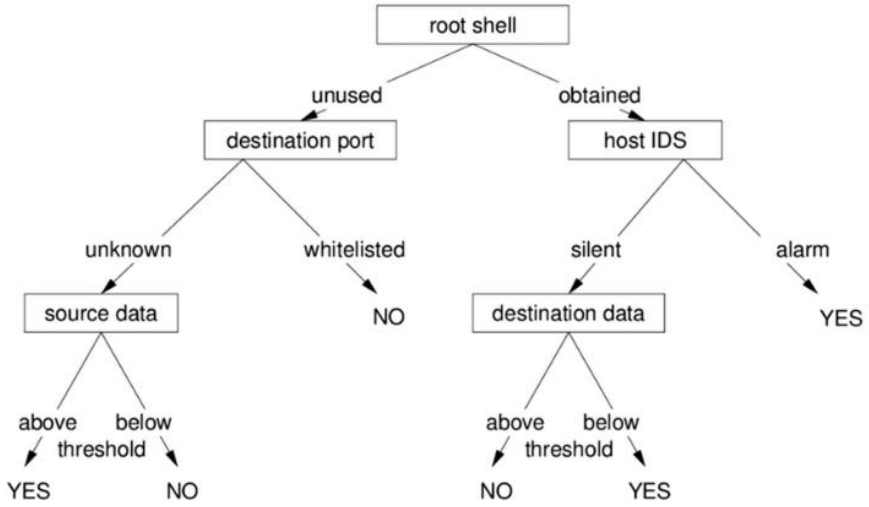

Figure 5: Decision Tree Sample Classification Structure

Records in decision trees are composed of attribute vectors, which include a set of classification characteristics that characterise the vector and a class attribute that associates the input data with a particular class. By constantly searching the database for the attribute that best separates the data into the several subclasses, up until a halting condition is satisfied, a decision tree is created. Customers can quickly summarise the facts in the presentation form since decision trees may be shown in an understandable tree-structured way. The parent and child nodes that are linked to the root node of DT are known as leafs.

3) Support Vector Machines (SVM): SVM aims to find a hyperplane that best separates classes in a high-dimensional space. Well-suited for binary classification tasks and can be used in both supervised and semi-supervised settings.

Boser et al. (1996) invention of the support vector classifier or repressor is used for non-linear cataloguing issues [11]. Kernel, hyperplane, margin, and data points are the main SVM buzzwords. Finding the closest support vectors in the hyperplane using SVM is superior to using linear regression. This results in the best global minimum value being found inside the search space. Different kernels are used in SVM to reduce the dimensionality of the raw input data, including linear, radial bias, and polynomial functions (i.e. frames the pattern or structure for input data). Real-time cataloguing results with zero error or higher accuracy are frequently obtained using kernel-based SVM models. SVM's working method is typically used to divide raw input into various clusters utilising hyperplane and margins. The network is trained using a variety of activation functions according to the demands of the applications. Below are some simple mathematical equations.

$$\rho^T * X_i + \delta = 0 \qquad (100)$$

Were , $'\rho'$ is the weight value which is more important in training the neural network and $'\delta'$ bias value with sum of input features denoted by $'Xi'$.

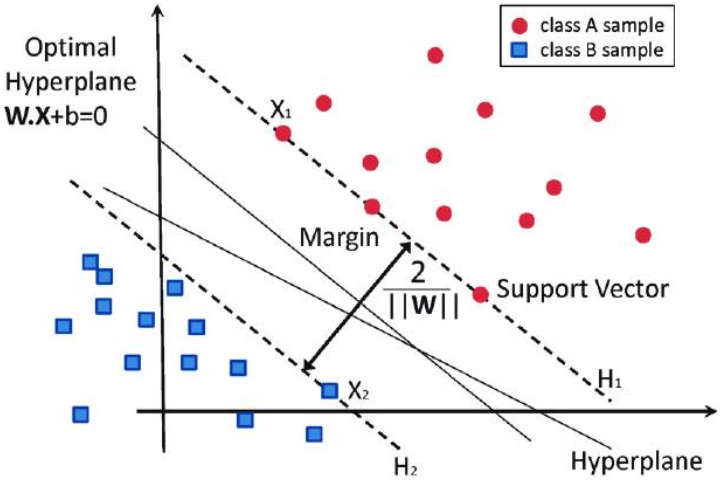

Figure 6: SVM Classifier

C. Neural Networks

Deep learning neural networks, especially convolational and recerrent neural networks, can learn complex patterns in data. Suitable for image-based intrusion detection, text analysis, and feature-rich data [12].

1) K-Nearest Neighbors (KNN): KNN classifies an instance by considering the classes of its k-nearest neighbors. Effective for both supervised and unsupervised settings, especially when the decision boundaries are complex.

2) Clustering Algorithms (e.g., K-Means, DBSCAN): Clustering algorithms group similar data points together based on their features. Useful for unsupervised anomaly detection, identifying groups of similar behavice

3) Isolation Forest: Isolation Forest is specifically designed for anomaly detection and is based on the idea that anomalies are isolated instances in the data.

4) One-Class SVM: One-Class SVM is suitable for binary classification tasks where one class (normal) is significantly more prevalent than the other (intrusive). Ensemble Methods (eg. AdaBoost, Gradient Boosting).

Ensemble methods combine multiple models to enhance overall performance. Can improve detection accuracy and reduce false positives.

6. MODEL TRAINING AND OPTIMIZATION

In the process of building an Intrusion Detection System (IDS) using machine learning algorithms, model training and optimization are essential steps to ensure that the model performs well and effectively detects intrusions [13].

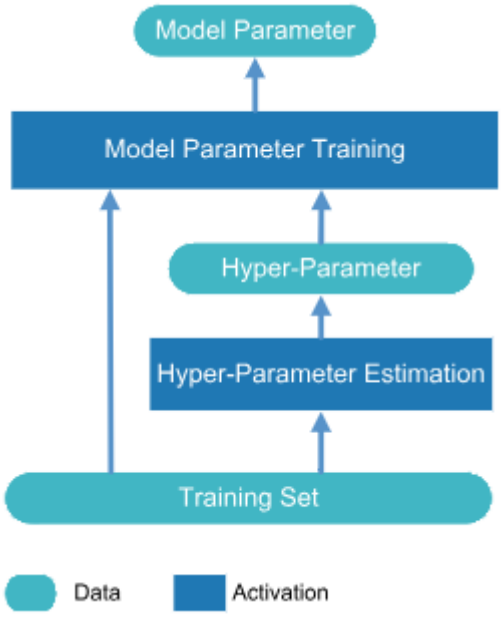

Figure 7: Model Training Procedure

A. Training Process

Data Preparation: Prepare the labelled dataset, separating features (input variables) and labels (intrusion or normal).

Algorithm Selection: Choose the appropriate machine learning algorithm based on the nature of your data and the goals of your IDS (e.g., supervised, unsupervised, or semi-supervised).

Training Set: Split the labelled dataset into a training set and a testing/validation set. The training set is used to train the model, while the testing/validation set is used to evaluate its performance.

Feature Scaling: Normalize or scale the features to ensure that they are on similar scales. This can help improve the convergence of certain algorithms.

Model Initialization: Initialize the chosen algorithm with default parameters.

Model Training: Train the algorithm using the training set. The algorithm learns to recognize patterns and features that distinguish between normal and intrusive instances.

Validation: Use the testing/validation set to assess the model's performance during training. This step helps identify potential over- fitting, where the model becomes too tailored to the training data.

B. *Hyperparameter Tuning and Cross-Validation*

Hyperparameters: These are parameters that are set before the training process begins. They control aspects of the algorithm's learning process, such as the number of hidden layers in a neural network or the depth of a decision tree.

Hyperparameter Tuning: Hyperparameters significantly impact the model's performance. Experiment with different values of hyperparameters to find the combination that optimizes performance. Techniques include grid search, random search, and more advanced methods

like Bayesian optimization.

Cross-Validation: Cross-validation assesses the model's generalization performance by splitting the dataset into multiple subsets (folds). The model is trained on one subset and validated on the others in a rotating manner. This helps mitigate the risk of overfitting by evaluating the model on multiple validation sets.

Regularization Techniques: Regularization methods like L1 and 12 regularization can help prevent overfitting by adding penalties to the model's complexity.

In the context of hyperparameter tuning, a separate validation set can be used to prevent data leakage and ensure unbiased tuning.

7. PROPOSED 1-D CNN MODEL

The deep learning intrusion detection model is proposed for computing environment using deep learning techniques of Convolutional Neural Network (CNN). The aim of this work is to improve the attack detection accuracy and reduce the false alarm rate by combing CNN model. The inherent characteristics of proposed model is that the correlation of features is learned by the 1-D CNN Layers and then long-range time dependent features are learned by the subsequent layer. The CNN model is performed very well. This integrated grouping operations are used to derive the relationship of the features between the results [14]. The model can automatically determine the efficient properties of the intrusion samples so that the intrusion samples can be classified accurately. The basic concepts of CNN is discussed in following sections. The proposed deep learning intrusion detection model namely Integrated Convolutional Neural Network is discussed. The experimental results and performance metrics are discussed.

8. CONVOLUTIONAL NEURAL NETWORKS (CNN)

Yann LeCun was the first to introduce Convolutional Neural Networks, or ConvNets, in the 1980s. Multiple layers of artificial neurons make up a convolutional neural network. As seen in Figure 8, artificial neurons carry out mathematical operations that determine the weighted sum of several inputs and produce an activation value. Convolution is the process of multiplying inputs by weights and adding them up [15].

One type of neural network that typically consists of a series of layers is called a CNN. A CNN consists of an input layer, an output layer, and a hidden layer that contains multiple convolutional, pooling, and fully connected layers. The output of the last convolution layer is used as input by the CNN's classification layer, which is its last layer.

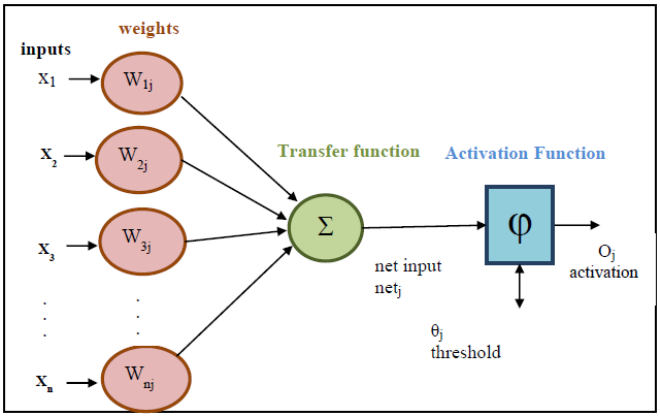

Figure 8: Artificial Neuron Model

Input Layer

This layer serves as the CNN's input. It brings the initial data into the CNN for further processing by subsequent layers because this is the very first layer. It may contain one dimensional, two dimensional or three dimensional inputs.

Convolutional Layer

This layer is the primary building block and is also known as the feature extractor where the feature data is extracted. This is the core component of the CNN that performs the convolution process. Neural networks are hence referred to by their name. An array of weights known as a filter or kernel that are smaller than the input data is used in a convolution procedure. The filters and an array of input data are multiplied together. Often called the scalar product or dot product, this operation takes a filter-sized patch of input and filter and uses an element-wise multiplication function to produce a single value. The same filter (set of weights) can be multiplied by the input array several times at various locations on the input because a filter is smaller than the input. In particular, the filter is applied methodically, from top to bottom and left to right, to every overlapping portion or filter-sized patch of the incoming data. The output obtained by this operation is called feature map. By making the kernel smaller than input, it is needed to store fewer parameters. Hence, the memory requirement of the model is reduced and it also improves the statistical efficiency of the model [16].

At every CNN layer, a collection of n kernels and biases—W={w1, w2,…,wn} and B ={b1,b2,…, bn}, respectively—are convolved with input data. As seen in Figure 6, the convolution between the data and each kernel creates a new feature map Xk.

The transformation for each convolution layer l is determined by

$$X_k^l = f\left(W_k^{l-1} * X^{l-1} + b_k^{l-1}\right) \qquad (1)$$

Where f and l denote activation function and layer respectively.

Once a feature map is created, each value in the feature map is passed to the subsequent layers through the activation function.

Stride and Padding

Stride denotes the motion of the filter i.e., how many steps it moves in each pass in convolution. By default, the kernel takes one step at a time. Padding is a process of adding zeros to the input matrix symmetrically.

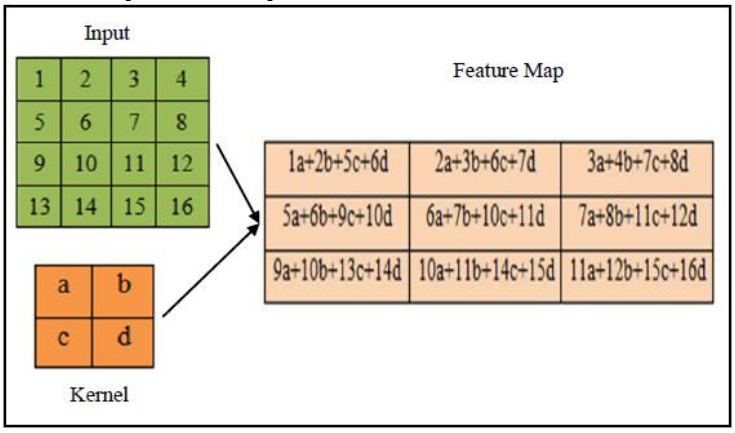

Figure 9: Convolution Operation

The convolution output dimension is calculated when an input passes through it as

$$W_2 = \frac{W_1 - F + 2*P}{S} + 1 * D \qquad (2)$$

Where, W2 is the output width / height, W1 is the width / height of the input, F is the width / height of the kernel, P is the padding, D is the depth/number of feature maps/activation maps and S is the stride [17].

Activation Functions

Activation functions are the most crucial part of any neural network in deep learning. It is used to perform a nonlinear transformation on the input.

They basically decide to activate neurons or deactivate them to get the desired output by defining the output of input or set of inputs.

The neuron performs a linear transformation on its input using the weights and biases as

$$X = (weight * input) + bais \qquad (3)$$

An activation function is applied on the above result for non-linear transformation which is expressed as

$$Y = Activation(E(weight * input) + bais) \qquad (4)$$

Several different types of activation functions are used in deep learning. The most famous activation functions are described below,

- Binary step
- Linear
- Sigmoid
- ReLU
- Leaky ReLU
- Tanh

- Softmax

Pooling Layer

Following the convolutional layer comes a pooling layer, which is another CNN building piece. Its purpose is to gradually shrink the representation's spatial size in order to lower the network's computation and parameter count. Every feature map produced by the convolutional layer is individually processed by the pooling layer. Compared to the feature map, the pooling operation or filter is smaller [18]. A H×W "block," with H representing the block's height and W its width, slides over the input data during the pooling operation. In order to decrease the height and width, the stride—that is, the number of steps taken during the sliding operation—is frequently equal to the size of the pool. To summarize the lowest value in the region, the average presence of a feature, and the most activated presence of a feature, respectively, three different pooling types are used: min pooling, average pooling, and max pooling. The Max pooling and Min pooling processes are displayed in Figure 10. Max pooling and average pooling are the two most used pooling types.

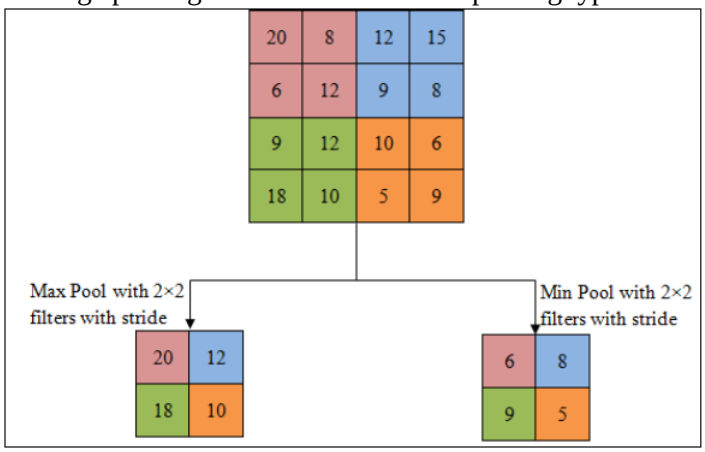

Figure 10: Max and Min pooling operations

Max Pooling

Max pooling chooses the highest element from the feature map area that the filter covers. A feature map with the most noticeable features from the prior feature map would be the result of the Max-pooling layer.

Average Pooling

For every patch on the feature map, it determines the average value. A lot of information about a block's or pool's "less important" components is preserved by average pooling. However, Max Pooling picks the maximum value and ignores them. Following the pooling layer, the feature map's size is

$$Feature\ map\ size = \frac{l-f+1}{s} * \frac{w-f+1}{s} * C \qquad (5)$$

Where l is the length of the feature map, w is the width of the feature map, f is the dimensions of the filter, c is the number of channels of the feature map and s is the stride.

Fully-Connected Layer

The name of the full-connected layer accurately describes its nature. Simply said, feed

forward neural networks, which make up the network's final few layers, are the fully linked layer. Every node in the output layer has a direct connection to a node in the layer above it in the fully-connected layer [19]. After being flattened, the output from the last pooling or convolutional layer is passed into the fully connected layer as its input. Through the output of the convolutional layer, it learns non-linear combinations of the high-level features. Using the features that were retrieved from the earlier layers and their various filters, this layer carries out the classification process. As seen in Figure 11, the output from the last (and any) pooling and convolutional layer is flattened, or all of its values are unrolled into a vector.

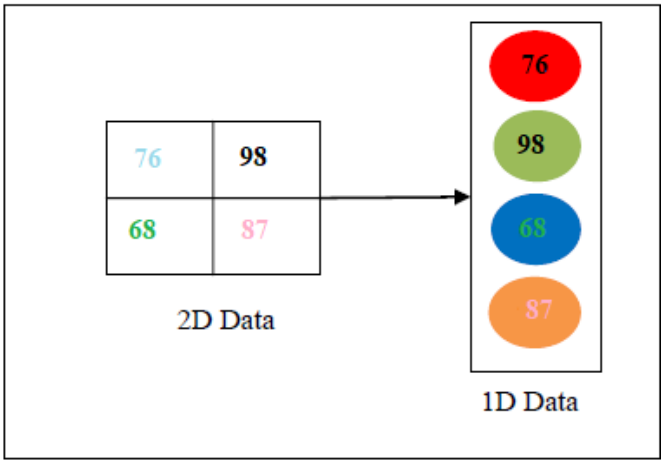

Figure 11: Fully Connected Layer

A few fully connected layers that carry out the same mathematical operations as ANNs are then connected to this flattened vector. The final layer, which comes after the fully linked layers, use the softmax activation function (rather than ReLU) to classify the inputs appropriately by calculating the likelihood that they belong to a specific class (from 0 to 1).

Algorithm No-1

-Bot-IoT dataset
-Extract labels & Features
-Split dataset for training/testing in ratio of 80:20
-Resize feature matrix to row (feat)*col (feat)*1
-Sequential model
-1-D Convolution -→ Max pooling
-1-D Convolution -→ Max pooling
- Flatten
- Dense
- Train the model using train data
- Performance calculation using test data.

9. MODEL EVALUATION AND VALIDATION

Evaluating and validating the performance of your machine learning model for Intrusion Detection Systems (IDS) is essential to ensure that it effectively detects intrusions and operates reliably in real-world scenarios [20].

A. Confusion Matrix Analysis:

A tabular representation of a classification model's performance is called a confusion matrix. It offers valuable information on false positives (FP), false negatives (FN), true positives (TP), and true negatives (TN).

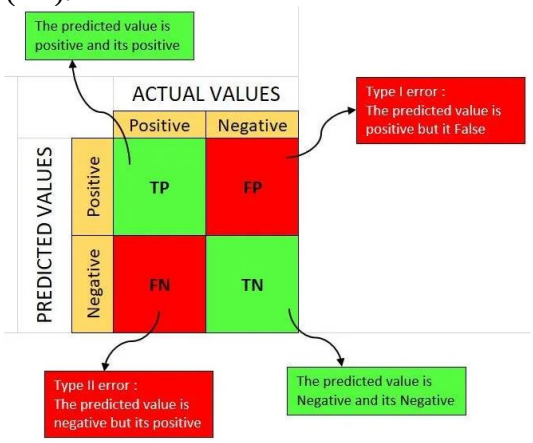

Figure 12: Confusion Matrix for Training

True Positives (TP): Instances correctly classified as intrusive.
True Negatives (TN): Instances correctly classified as normal.
False Positives (FP): Instances incorrectly classified as intrusive when they are normal (Type I error).
False Negatives (FN): Instances incorrectly classified as normal when they are intrusive (Type II error).
A key tool for evaluating the model's performance is the confusion matrix, which computes measures including specificity, recall, accuracy, precision, and Fl-score.

B. ROC Curves and AUC:

At various categorization thresholds, the trade-off between the true positive rate (recall) and the false positive rate (FPR) is shown visually using receiver operating characteristic (ROC) curves. They help determine an appropriate threshold for classifying instances based on the model's output probabilities.
True Positive Rate (Recall):

$$TPR = \frac{TP}{(TP+FN)} \qquad (6)$$

False Positive Rate (FPR):

$$FPR = \frac{FP}{(FP+TN)} \qquad (7)$$

The ROC curve's overall performance is gauged by the Area Under the Curve (AUC). Better performance is indicated by higher AUC values, which range from 0 to 1. A perfect classifier has an AUC of 1, whereas random guessing has an AUC of 0.5.

A model with an ROC curve that consistently stays above the diagonal line (random guessing) and a high AUC value indicates that the model is performing well in

distinguishing between normal and intrusive instances.
By analysing the confusion matrix, ROC curves, and AUC, you can gain insights into how well your IDS model is performing and make informed decisions about model thresholds and adjustments to optimize its performance.

10. CASE STUDIES AND EXAMPLES

Real-world implementations of Machine Learning (ML)-based Intrusion Detection Systems (IDS) have revolutionized the way organizations combat cyber threats. By harnessing the power of AI and ML algorithms, these systems are capable of efficiently detecting both known and emerging security breaches. Let's explore a few examples of such implementations and their impact [21].

A. Snort and Suricata:

Snort and Suricata are open-source network IDS solutions that leverage signature-based detection, making them suitable for identifying known attacks. However, both systems have evolved to in- corporate some machine learning aspects. They can analyze network traffic patterns and apply heuristics to identify potential anomalies that might indicate new or previously unseen threats.

1) Darktrace: Darktrace utilizes unsupervised machine learning and AI algorithms to detect anomalies across networks, cloud environments, and IoT devices. Its "Enterprise Immune System" approach adapts to the changing behaviors of users and systems. The system learns what is "normal" and flags deviations from the baseline behavior as potential threats, allowing for real-time response to novel attacks.

2) Vectra AI: Vectra AI employs AI-driven threat detection to identify attacker behaviors within network traffic. It leverages unsupervised machine learning to analyse large amounts of data and create behavioral profiles of users and devices. By detecting subtle deviations from these profiles, Vectra AI identifies potentially malicious activities, helping security teams respond quickly.

B. CylancePROTECT:

CylancePROTECT uses machine learning algorithms to prevent both known and unknown threats by examining files and programs. The system's AI model learns from a vast dataset of known malware samples and their attributes, enabling it to detect and prevent new, similar threats before they execute [22].

C. XDR Solutions (Extended Detection and Response):

Many XDR solutions integrate machine learning algorithms to provide enhanced threat detection and response across multiple attack vectors. They aggregate data from various sources, such as endpoint, network, and cloud, and employ ML algorithms to identify patterns, indicative of attacks, enabling comprehensive threat hunting and incident response.

D. User and Entity Behavior Analytics (UEBA) Systems:

UEBA solutions like Splunk's User Behavior Analytics (UBA) use machine learning to monitor user and entity behaviors, detect anomalies, and identify potential insider threats. These systems analyse user activities, logins, and access patters to identify deviations from typical behaviors.

E. Industrial Control Systems (ICS) Security:

ML-based IDS are also making their mark in critical infrastructure security. Systems like Claroty utilize AI and ML algorithms to monitor industrial networks for anomalies and

threats. They learn the operational behaviors of industrial assets and trigger alerts when deviations occur.

The adoption of ML-based IDS solutions brings several benefits, including:

Enhanced Detection: ML algorithms can identify subtle patterns and anomalies that may go unnoticed by traditional rule-based systems. Adaptability: These systems can adapt to changing attack techniques and evolving threat landscapes. Reduced False Positives: By learning from data, ML algorithms can reduce false positives by distinguishing between genuine and anomalous activities. Automation: ML-based IDS can automate threat detection and response, alleviating the burden on security teams.

11. RESULTS AND ANALYSIS

The BoT-IoT Dataset

In the Cyber Range Lab of the Australina Center for Cyber Security (ACCS), the tshark program was used to create the raw network packets (Pcap files) of the BoT-IoT dataset. These packets include both regular and aberrant traffic. Using the Ostinato tool and Node-red (for non-IoT and IoT, respectively), simulated network traffic was produced. The source files for the dataset are offered in a variety of formats, including the generated argus files, the original PCAP files, and finally the CSV file. To aid in the classification procedure, the files were divided into assault categories and subcategories.

Only 0.8% and 0.2% of the enormous database used is used for training and testing purposes, respectively. The estimate of several metrics is shown in Figure 13 to Figure 22 below.

Accuracy: It displays the chosen classifiers' accuracy percentage attributes. It is evident from the accuracy inquiry report that the constructed model has provided an optimal performance %. The instance-based classifier is the one that has a greater accuracy rate.

Precision: The constructed model's performance quality, which has made accurate predictions. It has been misclassified if the 100% position value is stated by 1 or if the 100% percentage depreciation value obtained is smaller. The chosen classifiers' position levels vary from 0.969914 to 0.971816 for various classifiers.

Recall: The ROC values and the recall value are comparable as well. For an ideal model, a 10% false negative classification is permitted. It adheres to the equation listed below. The investigative report's recall span for several classifiers ranges from 0.965254 to 0.969218.

F1-Score: The F-Measure value is similar to the Recall values. The false negative classification of 10% is allowed for an optimum model. It follows the below mentioned equation. The span for F-Measure in the investigational report is 0.968047 to 0.969688 for different classifiers.

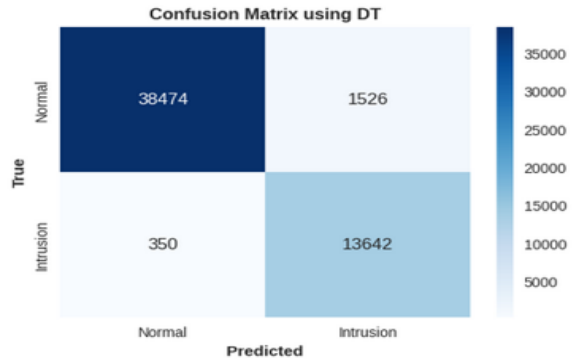

Figure 13: Confusion Matrix for Decision Tree Classifier

The confusion matrix for Decision Tree classifier, as illustrated in Figure 13, shows that the MSE of the classifier is 3.48%, the accuracy of the classifier is 96.53%, the precision value is 96.73%, the recall is 96.53%, and the F1-score is 96.57%.

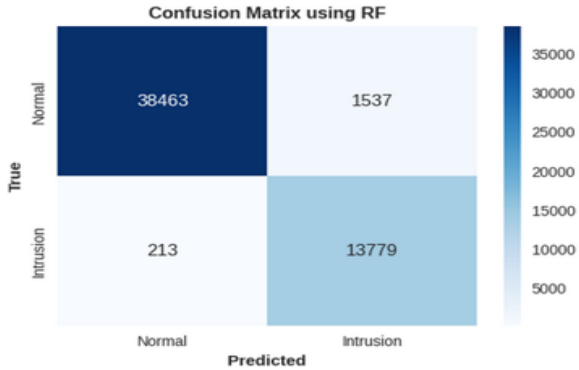

Figure 14: Confusion Matrix for Random Forest Classifier

The confusion matrix for RF classifier, as illustrated in Figure 14, shows that the MSE of the classifier is 3.24%, the accuracy of the classifier is 96.73%, the precision value is 96.99%, the recall is 96.77%, and the F1-score is 96.80%.

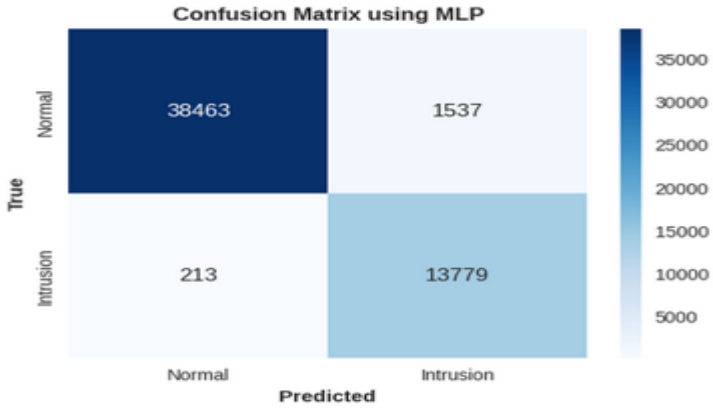

Figure 15: Confusion Matrix for Multi-layer Perceptron Classifier

The confusion matrix for MLP classifier, as illustrated in Figure 15, shows that the MSE of the classifier is 3.24%, the accuracy of the classifier is 96.77%, the precision value is 96.99%, the recall is 96.76%, and the F1-score is 96.80%.

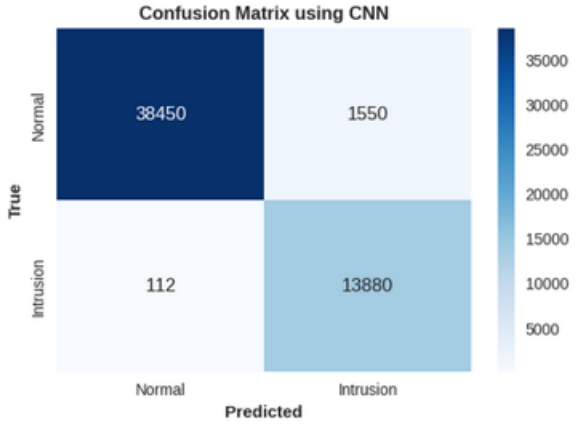

Figure 16: Confusion Matrix for 1D Convolutional Neural Networks Classifier

The confusion matrix for 1D-CNN classifier, as illustrated in Figure 16, shows that the MSE of the classifier is 3.07%, the accuracy of the classifier is 96.92%, the precision value is 97.18%, the recall is 96.92%, and the F1-score is 96.97%.

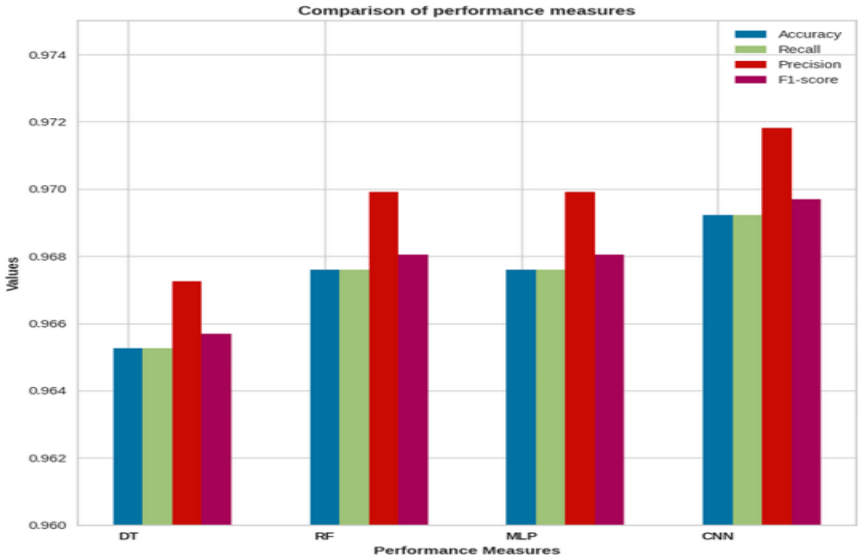

Figure 17: Comparison of performance measures

Table 1 gives the performance characteristics of the selected base classifier. The MSE of the classifiers ranges 030782 to 032412, the accuracy of the classifiers ranges from 0.965254 to 0.969218, the recall of the classifiers ranges from 0.965254 to 0.969218, the precision of the classifiers ranges from 0.969914 to 0.971816, and the F1-score of the classifiers ranges from 0.968047 to 0.969688 for Decision Tree, Random Forest, MLP, 1D CNN the instant base

classifier respectively. Accuracy is predictions done by the built model in a true positive way.

Figure 17 shows the performance characteristics of the selected base classifier. It has been shown that the performance of 1D-CNN is better than other classifiers.

Table 1: Comparison of performance parameters

Model	MSE	Accuracy	Recall	Precision	F1-score
DT	0.034746	0.965254	0.965254	0.967249	0.965696
RF	0.032412	0.967588	0.967588	0.969914	0.968047
MLP	0.032412	0.967588	0.967588	0.969914	0.968047
CNN	**0.030782**	**0.969218**	**0.969218**	**0.971816**	**0.969688**

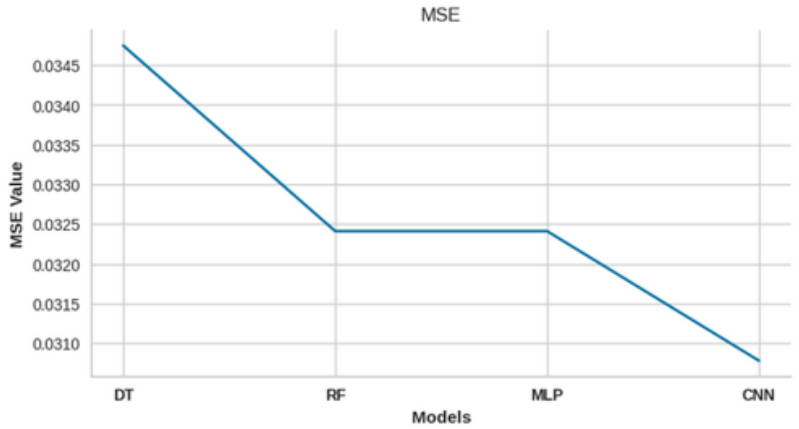

Figure 18: MSE of different models

Above figure 18 shows decrement in MSE over different models. The least MSE found for CNN model.

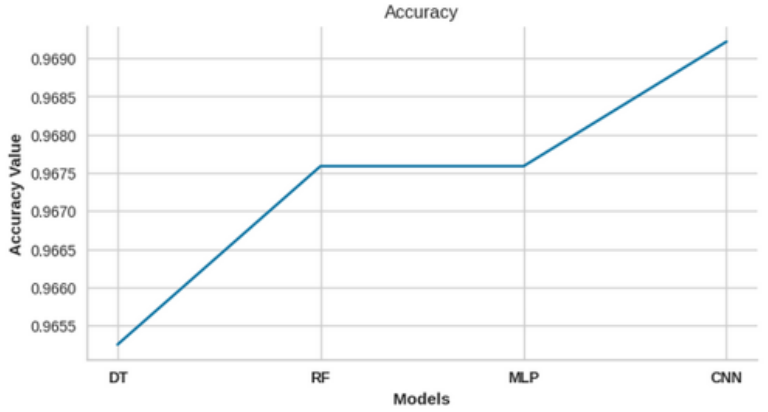

Figure 19: Accuracy of different models

Above figure 19 shows improvement of accuracy level of CNN over other techniques. The maximum accuracy found for CNN model.

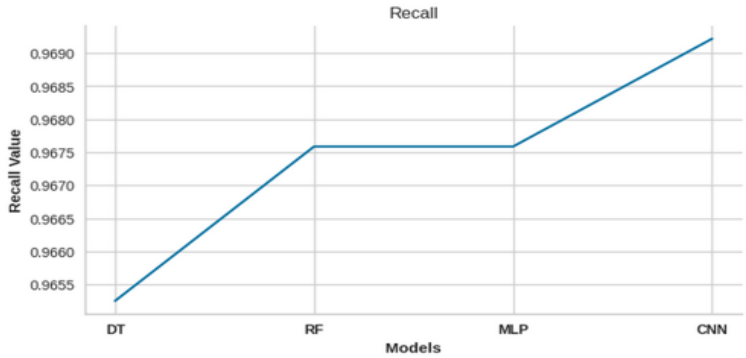

Figure 20: Recall of different models

Above figure 20 shows improvement of recall level of CNN over other techniques. The maximum recall found for CNN model.

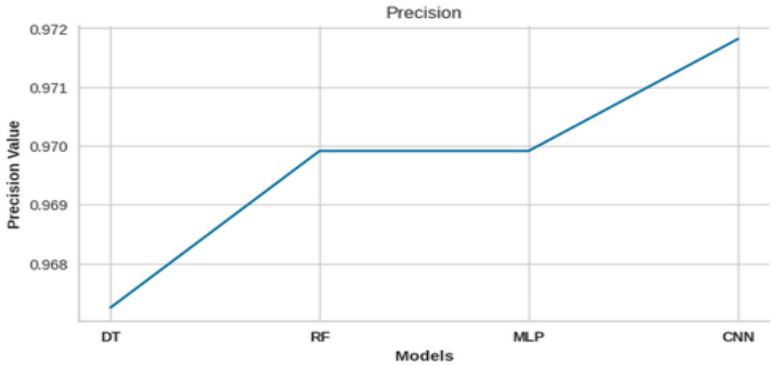

Figure 21: Precision of different models

Above figure 21 shows improvement of precision level of CNN over other techniques. The maximum precision found for CNN model.

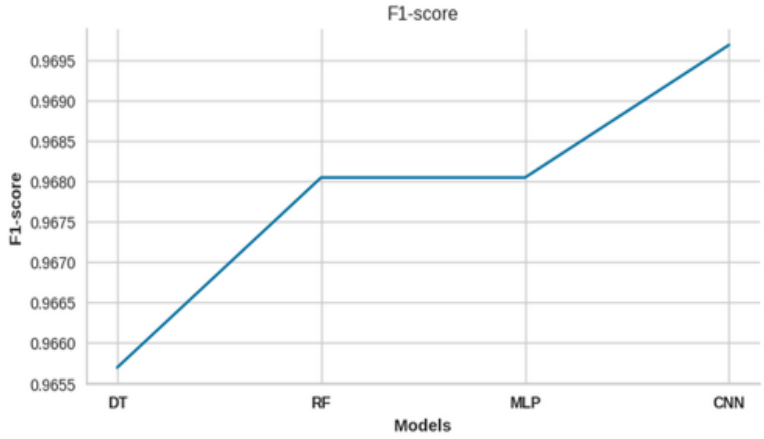

Figure 22: F1-score of different models

Above figure 22 shows improvement of F1-score level of CNN over other techniques. The maximum F1-score found for CNN model.

12. FUTURE TRENDS AND DEVELOPMENTS

The future of Intrusion Detection Systems (IDS) is tightly intertwined with the advancements in Artificial Intelligence (AI) and Machine Learning (ML). As the cybersecurity landscape becomes increasingly complex, the capabilities of AI and ML are poised to play a pivotal role in revolutionizing how organizations detect and respond to cyber threats. Here are some future trends and developments in AI and ML for IDS:

1) Enhanced Anomaly Detection: AI and ML will continue to refine anomaly detection capabilities. Deep learning techniques, such as neural networks and deep autoencoders, will enable IDS to better distinguish subtle and complex deviations from normal behavior, even in high-dimensional data.

2) Unsupervised Learning Dominance: Unsupervised learning approaches will gain prominence due to their ability to detect novel and emerging threats. Clustering algorithms and generative models will become essential for identifying previously unseen attack patterns.

3) Behavioral Analysis and UEBA: User and Entity Behavior Analytics (UEBA) will leverage AI and ML to monitor and analyze user activities in real-time. These systems will move beyond just identifying anomalies to understanding the context behind user behaviors, enabling more accurate threat detection.

4) Adaptive Security and Continuous Learning: AI- powered IDS will learn from every new piece of data, adapting to changes in network behavior and threat landscape. Continuous learning will enable these systems to evolve and provide up-to-date protection against dynamic attacks.

5) Explainable Al for Transparency: As Al-driven IDS become more sophisticated, the need for explainability will grow. Methods for making AI decisions transparent and interpretable will be crucial, especially for compliance and regulatory requirements.

6) Integration with Threat Intelligence: Al-driven IDS will seamlessly integrate with threat intelligence platforms, enriching their knowledge with real-time global 4 eat data. This integration will enhance the system's ability to detect and respond to emerging threats.

7) Edge and IoT Security: With the proliferation of Internet of Things (IoT) devices, AI-based IDS will be essential for securing edge networks and IoT environments. These systems will identify abnormal device behaviors and potential vulnerabilities.

8) Zero-Day Threat Detection: Machine learning models trained on historical data will detect zero-day threats by recognizing patterns that deviate from normal behaviors. This predictive capability will significantly enhance threat prevention.

9) Federated Learning for Privacy: Federated learning, where models are trained across multiple decentralized devices, will gain traction in IDS. It preserves data privacy while allowing models to be trained on a larger, more diverse dataset.

10) Al-Driven Threat Response: Al will not only detect, threats but also automate response actions. Adaptive IDS will orchestrate dynamic responses such as isolating compromised devices, altering network configurations, or blocking malicious activities.

11) Evolving Attack Simulation: Al-driven IDS will incorporate Al-generated attack simulations to continually test and validate their detection capabilities. This proactive

approach will enhance system robustness against novel attack vectors.

12) Multimodal Data Analysis: IDS will analyse diverse data sources like network traffic, logs, and user behaviors in a unified manner. Combining multiple data modalities will provide a holistic view of potential threats.

13) Collaborative Defense: Al-powered IDS will enable collaborative defense mechanisms, where information about new threats and attack techniques is shared across organizations and industries to create a united front against cyber adversaries.

13. CONCLUSION

IDSS safeguard digital environments from evolving cyberthreats. This extensive IDS study, aided by machine learning, reveals how technological advancement and data, system, and network security are linked. This talk discussed IDS foundations, types, and concerns. Rule-based to machine learning-based threat detection improved adaptability, accuracy, and efficiency. Understanding data collection, preprocessing, and feature engineering created models. These steps illustrated how data preparation helps computers detect intrusion attempts in a sea of harmless activity. Labelling training data required exact and balanced datasets. Practitioners learned how to create robust detection algorithms by distinguishing anomaly detection from signature-based detection. Cross-validation and hyperparameter fine- tuning were essential to model training and optimisation. This ensures the model learns from data and understands unknown real-world circumstances. Confusion matrix analysis, ROC curves, and AUC were utilised to assess models for adversarial tasks. This knowledge helps model tuning balance accuracy and generality. Real-world case studies revealed how machine learning-based IDS protect against more cyberthreats. Each scenario proved AI and ML systems can identify breaches and adapt to new attack paths. Finally, AI and ML will improve IDS. Enhanced anomaly identification, behavioural analysis, continuous learning, and human-machine harmony are possible. Finally, IDSs and machine learning are transforming cybersecurity. Each iteration, innovation, and advancement increases cyber threat defences, protecting data, systems, and networks in a complex digital world. AI and ML, careful planning, and training may help us overcome challenges and create a secure and trustworthy digital future.

REFERENCES

1. Adnan, A, Muhammed, A, Abd Ghani, AAA, Abdullah, A & Hakim, F2021, ‘An intrusion detection system for the internet of things based on machine learning: review and challenges. Symmetry’, vol. 13, no. 6, pp. 1-13.
2. Kasongo, SM & Sun, Y 2021, ‘A Deep Gated Recurrent Unit based model for wireless intrusion detection system. ICT Express, vol. 7, no. 1, pp. 81-87.
3. Kayode Saheed, Y, Idris Abiodun, A, Misra, S, Kristiansen Holone, M & Colomo-Palacios, R 2022, ‘A machine learning-based intrusion detection for detecting internet of things network attacks. Journal of Alexandria Engineering vol. 6, no. 12, pp. 9395-9409.
4. D. Agrawal, C. Agrawal and H. Yadav. "A Machine Learning Based Intrusion Detection Framework Using KDDCUP 99 Dataset", vol. 4. no. 6, pp. 11.
5. Lyngdoh, J, Hussain, MI, Majaw, S & Kalita, HK 2019, ‘An Intrusion detection method using artificial immune system approach’, Communications in Computer and Information Science International conference on advanced informatics for computing research, pp. 379-387.

6. Saheed, YK 2022, 'Performance improvement of intrusion detection system for detecting attacks on Internet of things and edge of things. In: Misra S, TKA, Piuri V, Garg L, editors, Artificial intelligence for cloud and edge computing'. Internet of things (technology, communications and computing). Cham: Springer; pp. 321-39.
7. D. M. Abdulqader, A. M. Abdulazeez and D. Q. Zeebaree, "Machine Learning Supervised Algorithms of Gene Selection: A Review, vol. 62, no. 03. pp. 13. 2020.
8. R. Vijayanand, D. Devaraj and B. Kannapiran. "A Novel Deep Learning Based Intrusion Detection System for Smart Meter Communication Net- work", 2019 IEEE International Conference on Intelligent Techniques in Control Optimization and Signal Processing (INCOS), pp. 1-3, Apr. 2019.
9. Zhong, W, Yu, N & Ai, C 2020, 'Applying big data based deep learning system to intrusion detection' , Journal of Big Data Min Anal, vol. 3, no. 3, pp. 181-195.
10. Nagalalli, G & Ravi, G 2022, 'A Novel Megabat Optimized Intelligent Intrusion Detection System In Wireless Sensor Networks', Journal of Intelligent Automation and Soft Computing, Tech Science Press, vol. 35, no. 1, pp. 475-490.
11. J. Gu and S. Lu, "An effective intrusion detection approach using SVM with naïve Bayes feature embedding", Computers Security, vol. 103, pp. 102158, Apr. 2021.
12. D. M. Manimekalai and G. Anupriya, "A Novel Intrusion Detection System using Data Mining Techniques", vol. 6, no. 6, pp. 5, 2019.
13. D. E. Baraneetharan, "Role of machine learning algorithms intrusion detection in WSNs: A survey," Journal of Information Technology and Digital World, vol. 2, no. 3, pp. 161–173, 2020.
14. F. A. Khan, A. Gumaci, A. Derhab and A. Hussain, "TSDL: A Two- Stage Deep Learning Model for Efficient Network Intrusion Detection", IEEE Access, vol. 7. pp. 30373-30385, 2019.
15. Y. Xiao, C. Xing, T. Zhang and Z. Zhao, "An Intrusion Detection Model Based on Feature Reduction and Convolutional Neural Networks", IEEE Access, vol. 7. pp. 42210-42219, 2019.
16. H. Yao, D. Fu, P. Zhang, M. Li and Y. Liu, "MSML: A Novel Multilevel Semi-Supervised Machine Learning Framework for Intrusion Detection System", IEEE Internet Things J. vol. 6, no. 2, pp. 1949-1959. Apr. 2019.
17. R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran. A. Al- Nemrat and S. Venkatraman, "Deep Learning Approach for Intelligent Intrusion Detection System", IEEE Access, vol. 7, pp. 41525-41550, 2019.
18. G. Andresini, A. Appice, N. D. Mauro, C. Loglisci and D. Malerba, "Multi-Channel Deep Feature Learning for Intrusion Detection", IEEE Access, vol. 8, pp. 53346-53359, 2020.
19. F. A. M. Bargarai, A. M. Abdulazecz, V. M. Tiryaki and D. Q. Zecbarce. "Management of Wireless Communication Systems Using Artificial Intelligence-Based Software Defined Radio", Int. J. Interact. Mob. Technol, vol. 14, no. 13, pp. 107, Aug 2020.
20. S. Ifzarne, H. Tabbaa, I. Hafidi and N. Lamghari, "Anomaly detection using machine learning techniques in wireless sensor networks," Journal of Physics: Conference Series, vol. 1743, pp. 012021, 2021.
21. W. Zhang, D. Han, K. C. Li and F. I. Massetto, "Wireless sensor network intrusion detection system based on MKELM," Soft Computing, vol. 24, no. 16, pp. 12361–12374, 2020.
22. S. Jiang, J. Zhao and X. Xu, "SLGBM: An intrusion detection mechanism for wireless sensor networks in smart environments," IEEE Access, vol. 8, pp. 169548–169558, 2020.