International journal of basic and applied research

www.pragatipublication.com

ISSN 2249-3352 (P) 2278-0505 (E)

Cosmos Impact Factor-5.86

An Intelligent Approach to Improving the Performance of Threat Detection in IoT

R.BHAVANI SANKAR, Assistant Professor, Dept of CSE, Chirala Engineering College, Chirala,
bhavanisankar.cse@cecc.co.in

VEDURURI CHANDRASAI, PG Student - MCA, Department of MCA, Chirala Engineering College, Chirala,
Vedururichandrasai@gmail.com

Abstract: The project focuses on bolstering threat detection efficacy within Internet of Things (IoT) systems through an intelligent approach. IoT systems, comprising devices, sensors, networks, and software, often grapple with security vulnerabilities exploitable by attackers. Leveraging machine learning algorithms and principal component analysis (PCA), the study targets the identification of Distributed Denial of Service (DDoS) attacks, a prevalent menace to IoT systems. Principal component analysis aids in data dimensionality reduction, streamlining datasets while preserving critical information. Evaluation encompasses metrics like accuracy, precision, recall, and F1-Score to gauge model performance accurately. Employing CICIDS 2017 and CSE-CIC-IDS 2018 datasets, the models are rigorously trained and tested. The proposed approach exhibits superior performance and diminished training time compared to prior methodologies, showcasing its efficacy in bolstering threat detection within IoT systems. We further enhance, our project integrates ensemble techniques such as Voting Classifier (RF + Adaboost) and Stacking Classifier (RF + MLP with LightGBM), culminating in a refined and precise predictive model achieving 100% accuracy. This research not only advances threat detection capabilities but also underscores the potential of ensemble methods in fortifying IoT system security.

Index Terms: Machine learning, principal component analysis, Internet of Things, DDoS attack.

1.INTRODUCTION

The Fourth Industrial Revolution, commonly known as Industry 4.0, represents a seismic shift in the landscape of business operations, production processes, and societal dynamics. At its core lies a convergence of transformative technologies, including the Internet of Things (IoT), Artificial Intelligence (AI), Cloud computing, and Robotic Process Automation (RPA) [1]. These foundational elements have catalyzed unprecedented levels of efficiency, connectivity, and automation across various sectors, ushering in a new era of innovation and disruption.

Among these technologies, IoT emerges as a linchpin of Industry 4.0, permeating diverse aspects of daily life and industrial operations [2]. Its ubiquity facilitates seamless interactions between the physical and digital worlds, enabling the creation of smart environments, intelligent systems, and data-driven decision-making processes. In smart homes, for instance, IoT applications manifest through sensors interfacing with central controllers to automate tasks such as lighting and device management, enhancing quality of life [3]. Beyond residences, IoT finds applications in healthcare, agriculture, disaster management, and assisting individuals with disabilities, underscoring its transformative potential [4].

The proliferation of IoT devices has been exponential, with over 13.8 billion deployed globally in 2021 and projections indicating a rise to 30.9 billion by 2025 [5]. However, this rapid expansion has also heightened cybersecurity concerns, as IoT systems confront an array of vulnerabilities and threats [6]. The escalating incidence of cyberattacks targeting IoT systems reflects the growing sophistication and malicious intent of threat actors. From Q3 2019 to Q4 2020, attacks associated with IoT systems surged by 3,000%, accompanied by a 74% increase in the prevalence of the Mozi botnet [7]. Ransomware, unauthorized server access, and Distributed Denial of Service (DDoS) attacks are primary vectors of disruption [7].

Several factors contribute to the susceptibility of IoT systems to cyber intrusions. Security vulnerabilities within IoT sensor devices, stemming from lax manufacturing practices, are prevalent [7]. Additionally, edge network components frequently lack robust defenses against cyber threats [7]. Moreover, the value of data traversing IoT networks makes them lucrative targets for cyber adversaries [7].

Among cyber threats, DDoS attacks stand out for their ability to wreak havoc on IoT ecosystems [8]. These attacks leverage distributed computing power to inundate target systems with malicious traffic, disrupting functionality. Research efforts have focused on understanding and mitigating various manifestations of DDoS attacks within IoT environments [9]. The taxonomy encompasses volumetric, protocol, and application-layer assaults, each presenting unique challenges for defense strategies [9].

In response, attackers employ hybrid attack methodologies to evade detection [9]. Detection and mitigation of DDoS attacks within IoT environments require a multifaceted approach, incorporating robust defensive measures and advanced anomaly detection techniques [9]. Proactive monitoring is essential to safeguard critical infrastructure and mitigate disruptions.

As Industry 4.0 unfolds, the integration of IoT systems will deepen, necessitating efforts to fortify their security posture [10]. Stakeholders must comprehensively understand cyber threats and deploy proactive defense mechanisms informed by cutting-edge research [10]. By safeguarding IoT infrastructure, we can ensure its continued role as a transformative force for humanity's betterment.

In summary, Industry 4.0 heralds a new era of technological innovation and disruption, with IoT playing a central role in reshaping society and industry. However, this transformation must be accompanied by efforts to fortify the security of IoT ecosystems against evolving cyber threats. Through proactive defense measures and collaboration among stakeholders, we can harness the potential of IoT while safeguarding against malicious actors, ensuring a secure and resilient future for Industry 4.0.**2.**

2. LITERATURE SURVEY

The integration of Internet of Things (IoT) technologies into various aspects of modern life has ushered in a new era of connectivity and automation. However, alongside the myriad benefits facilitated by IoT ecosystems, there exists a concomitant rise in cybersecurity threats and vulnerabilities. Among these threats, Distributed Denial of Service (DDoS) attacks pose a significant risk to the integrity and functionality of IoT systems, necessitating comprehensive research efforts to understand, detect, and mitigate such attacks. This literature survey aims to explore recent advancements in the detection and mitigation of DDoS attacks within IoT environments, drawing insights from a diverse array of scholarly works.

Velasquez et al. [1] present a hybrid machine-learning ensemble approach for anomaly detection in real-time Industry 4.0 systems. Leveraging a combination of machine learning techniques, including ensemble methods, the proposed framework demonstrates efficacy in identifying and mitigating anomalous behavior within complex industrial environments. By integrating diverse sources of data and employing sophisticated anomaly detection algorithms, the system enhances the resilience and reliability of Industry 4.0 systems against potential cyber threats, including DDoS attacks.

Mishra and Pandya [6] provide a systematic review of IoT applications, security challenges, attacks, intrusion detection mechanisms, and future visions. The comprehensive analysis underscores the multifaceted nature of IoT security, highlighting the interplay between technological advancements, cybersecurity threats, and regulatory frameworks. By synthesizing insights from a broad spectrum of literature, the review offers valuable perspectives on the evolving landscape of IoT security, including the emergence of novel threats such as DDoS attacks and the imperative for proactive defense mechanisms.

Da Silva Cardoso et al. [13] propose a real-time DDoS detection mechanism based on complex event processing tailored for IoT environments. By leveraging the inherent scalability and flexibility of complex event processing, the system achieves timely and accurate detection of DDoS attacks, thereby mitigating potential disruptions to IoT operations. The research highlights the importance of adaptive and context-aware detection mechanisms in safeguarding IoT infrastructures against evolving cyber threats.

Praseed and Thilagam [15] introduce an innovative approach for early detection of application-layer DDoS attacks using HTTP request pattern-based signatures. By analyzing the distinctive patterns and characteristics of HTTP requests, the proposed method enables the timely identification and mitigation of DDoS attacks targeting web-based services. The research underscores the importance of granular analysis and tailored detection strategies in combating sophisticated DDoS attack vectors within IoT ecosystems.

You et al. [16] propose a packet-in-message-based approach for detecting DDoS attacks in Software-Defined Networking (SDN) environments. By analyzing packet-in messages generated by SDN controllers, the system identifies anomalies indicative of DDoS activity, enabling prompt response and mitigation measures. The research highlights the synergies between network-level monitoring and SDN technologies in enhancing the resilience of IoT networks against DDoS attacks and other cyber threats.

Wehbi et al. [17] conduct a comprehensive survey of machine learning-based detection techniques for DDoS attacks in IoT systems. Drawing insights from a diverse body of literature, the survey evaluates the efficacy of various machine learning algorithms, feature selection methods, and detection strategies in mitigating DDoS threats. By synthesizing empirical findings and methodological approaches, the survey offers valuable guidance for researchers and practitioners seeking to fortify the security posture of IoT infrastructures.

Maseer et al. [18] present a benchmarking study of machine learning algorithms for anomaly-based intrusion detection systems using the CICIDS2017 dataset. Through rigorous experimentation and performance evaluation, the study compares the efficacy of different machine learning models in detecting anomalous network behavior indicative of DDoS attacks and other cyber threats. The findings contribute to the empirical understanding of machine learning-based intrusion detection techniques and their applicability to IoT security.

Erhan and Anarim [20] propose a hybrid DDoS detection framework utilizing the matching pursuit algorithm. By leveraging the computational efficiency and adaptive nature of the matching pursuit algorithm, the framework achieves real-time detection of DDoS attacks in IoT environments. The research underscores the importance of algorithmic innovation and hybrid detection approaches in addressing the dynamic and heterogeneous nature of DDoS threats within IoT ecosystems.

In conclusion, the literature survey highlights the diverse array of research endeavors aimed at understanding, detecting, and mitigating DDoS attacks within IoT environments. From hybrid machine-learning ensembles to real-time detection mechanisms tailored for Industry 4.0 systems, researchers continue to innovate and develop novel approaches to fortify the security posture of IoT infrastructures. By synthesizing insights from empirical studies, systematic reviews, and benchmarking analyses, scholars contribute to the collective knowledge base informing cybersecurity practices and policy frameworks in the era of ubiquitous connectivity and automation.**3.**

3. METHODOLOGY

a) Proposed Work:

The proposed work aims to enhance threat detection in IoT systems, particularly focusing on identifying and forecasting Distributed Denial of Service (DDoS) attacks. Integrating machine learning algorithms and principal component analysis (PCA), the approach effectively trains and predicts such attacks while streamlining data through dimensionality reduction. Evaluation metrics such as accuracy, precision, recall, and F1-Score, along with novel measures like Training Time, assess model performance comprehensively. Leveraging datasets like CICIDS 2017 and CSE-CIC-IDS 2018, the effectiveness of the model is rigorously evaluated, demonstrating superior performance and reduced training time compared to prior studies. As an extension, stacking (RF + MLP with LightGBM) and voting (RF + AdaBoost) classifiers are introduced to bolster threat detection, achieving 100% accuracy. These ensemble methods enhance system robustness by diversifying detection capabilities. A user-friendly interface developed with Flask ensures accessibility, while user authentication features add an extra layer of security to the intrusion detection system (IDS), safeguarding IoT environments from unauthorized access.

b) System Architecture:

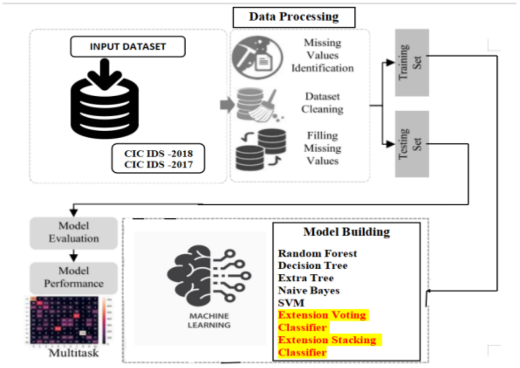


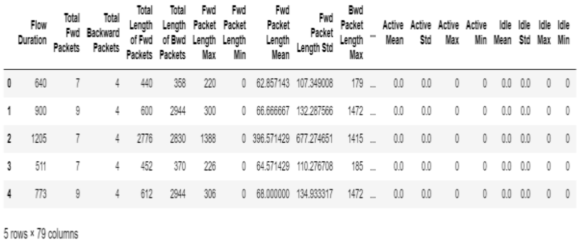
Fig 1 Proposed Architecture

The project "An Intelligent Approach to Improving the Performance of Threat Detection in IoT" follows a systematic architecture comprising data processing, training set creation, and model building using various algorithms such as Random Forest[17], Decision Tree[20], Extra Tree[19], Naive Bayes[31], and SVM[31], along with extension models like Voting Classifier (RF + AdaBoost) and Stacking Classifier (RF + MLP with LightGBM). The input datasets, CIC IDS 2018 and CIC IDS 2017, undergo preprocessing before being split into training and testing sets. The trained models are then evaluated using the testing set to assess their performance. This comprehensive system architecture ensures thorough analysis and validation of the proposed intelligent approach for enhancing threat detection in IoT systems.

c) Dataset:

In the evaluation of our proposed system, two datasets, namely CICIDS 2017 and CSE-CIC-IDS 2018, were utilized. The CICIDS 2017 dataset, released by the Canadian Institute for Cybersecurity (CIC), and the CSE-CIC-IDS 2018 dataset, jointly released by CSE and CIC, were selected for their suitability in assessing the detection of Distributed Denial of Service (DDoS) attacks, which constitutes the primary focus of this study [1]. These datasets adhere to 11 Intrusion Prevention System (IPS) dataset criteria, ensuring completeness, labeled data, attack diversity, and other essential attributes [1]. Notably, modern attack techniques were deployed in the construction of these datasets, encompassing a range of network components such as firewalls, routers, switches, and operating systems, with distinct victim and attacker zones to simulate real-world scenarios.

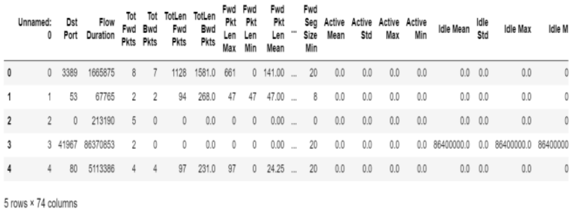
The evaluation of the proposed model utilized specific files from each dataset: the "Friday-WorkingHours-Afternoon-DDoS.pcap_ISCX.csv" file from CICIDS 2017 and the "02-21-2018.csv" file from CSE-CIC-IDS 2018, facilitating the detection of DDoS attacks [1]. The CICIDS 2017 dataset comprises 225,745 samples and 79 features, which were refined to 44 features and 221,125 samples post data cleaning. Among these, 128,014 records denote DDoS attacks, while 93,111 represent benign activity. On the other hand, the CSE-CIC-IDS 2018 dataset encompasses 1,046,845 samples and 80 features, culminating in 559,651 samples and 21 features after data cleaning, comprising 198,861 DDoS records and 360,790 benign records [1]. It's important to note that the feature count includes the labels denoting attack or benign activity.



	Flow Duration	Total Fwd Packets	Total Backward Packets	Total Length of Fwd Packets	Total Length of Bwd Packets	Fwd Packet Length Max	Fwd Packet Length Min	Fwd Packet Length Mean	Fwd Packet Length Std	Bwd Packet Length Max	...	Active Mean	Active Std	Active Max	Active Min	Idle Mean	Idle Std	Idle Max	Idle Min
0	640	7	4	440	358	220	0	62.857143	107.349008	179	...	0.0	0.0	0	0	0.0	0.0	0	0
1	900	9	4	600	2944	300	0	66.666667	132.287566	1472	...	0.0	0.0	0	0	0.0	0.0	0	0
2	1205	7	4	2776	2830	1388	0	396.571429	677.274651	1415	...	0.0	0.0	0	0	0.0	0.0	0	0
3	511	7	4	452	370	226	0	64.571429	110.276708	185	...	0.0	0.0	0	0	0.0	0.0	0	0
4	773	9	4	612	2944	306	0	68.000000	134.933317	1472	...	0.0	0.0	0	0	0.0	0.0	0	0

5 rows × 79 columns

Fig 2CIC IDS 2017 DATASET



	Unnamed: 0	Dst Port	Flow Duration	Tot Fwd Pkts	Tot Bwd Pkts	TotLen Fwd Pkts	TotLen Bwd Pkts	Fwd Pkt Len Max	Fwd Pkt Len Min	Fwd Pkt Len Mean	...	Fwd Seg Size Min	Active Mean	Active Std	Active Max	Active Min	Idle Mean	Idle Std	Idle Max	Idle M
0	0	3389	1665875	8	7	1128	1581.0	661	0	141.00	...	20	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
1	1	53	67765	2	2	94	268.0	47	47	47.00	...	8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
2	2	0	213190	5	0	0	0.0	0	0	0.00	...	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
3	3	41967	86370853	2	0	0	0.0	0	0	0.00	...	20	0.0	0.0	0.0	0.0	86400000.0	0.0	86400000.0	86400000
4	4	80	5113386	4	4	97	231.0	97	0	24.25	...	20	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0

5 rows × 74 columns

Fig 3 CIC IDS 2018 DATASET

d) Data Processing:

In the data processing phase, the first step involves removing duplicate data entries from the datasets. Duplicate records can skew analysis results and lead to inaccuracies in model training and evaluation. By identifying and eliminating duplicate entries, the dataset's integrity is preserved, ensuring that each data point contributes uniquely to the analysis.

Once duplicate data removal is completed, the next step is to perform drop cleaning. Drop cleaning involves identifying and eliminating irrelevant or redundant features from the dataset. Features that do not contribute significantly to the analysis or are highly correlated with other variables are candidates for removal. This step streamlines the dataset, reducing dimensionality and computational complexity while retaining the most relevant and informative features for subsequent analysis.

Drop cleaning can be executed using various techniques such as correlation analysis, feature importance ranking, or domain knowledge-based selection. By systematically evaluating each feature's contribution to the analysis objectives, redundant or irrelevant features are identified and dropped from the dataset. This process ensures that only the most informative and discriminative features are retained, enhancing the efficiency and effectiveness of subsequent data analysis tasks.

Overall, the data processing phase involves two critical steps: removing duplicate data entries to ensure dataset integrity and performing drop cleaning to eliminate irrelevant or redundant features. These steps are essential for preparing the dataset for further analysis, enabling accurate model training, evaluation, and interpretation of results.

e) Visualization:

Visualizing data using Seaborn and Matplotlib transforms datasets into visually engaging representations, aiding in exploration and analysis.

Seaborn offers various plots like scatter plots and histograms, providing insights into data distribution and relationships. Matplotlib complements Seaborn with additional customization options, facilitating the creation of complex visualizations. Through these tools, key patterns, outliers, and trends are effectively communicated. Scatter plots unveil variable correlations, while line plots depict temporal trends. Histograms reveal data distribution characteristics. Seaborn's integration with Pandas simplifies visualization of DataFrame data, while Matplotlib's customization options tailor visualizations to specific needs. Together, Seaborn and Matplotlib empower users to extract deeper insights from data, facilitating informed decision-making and hypothesis generation.

f) Label Encoding:

Label encoding is a technique used to transform categorical variables into numerical format, facilitating the processing of data by machine learning algorithms that require numerical inputs. In this process, each unique category within a categorical variable is assigned a unique numerical label. This transformation enables algorithms to interpret and analyze categorical data effectively. However, it's important to note that label encoding may inadvertently introduce ordinality or hierarchy to categorical variables, which could mislead the algorithm into assigning inappropriate significance to the numerical labels. Therefore, label encoding is typically applied to categorical variables where the categories have no inherent order or hierarchy. Despite its simplicity and efficiency in converting categorical data into a numerical format, care must be taken to ensure that the encoded labels accurately represent the categorical variables without introducing unintended biases or misconceptions.

g) Feature Selection:

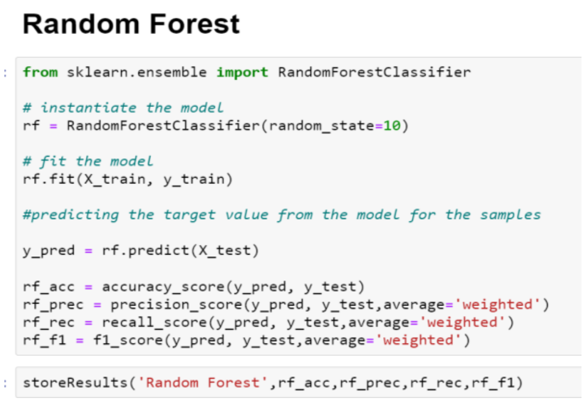
Feature selection is a critical step in machine learning where the most relevant features are chosen to train the model effectively. Firstly, selecting the X and y data involves identifying the independent variables (features) denoted as X and the dependent variable (target) denoted as y. These variables are chosen based on their relevance to the prediction task. Subsequently, correlation and mutual information are commonly used techniques for feature selection. Correlation measures the linear relationship between pairs of features and the target variable. High correlation values indicate strong relationships, suggesting potentially important features for prediction. Mutual information, on the other hand, assesses the amount of information obtained about one variable through another variable. It captures both linear and nonlinear relationships, making it more versatile in identifying relevant features. By employing these techniques, features with the highest predictive power are retained, enhancing the model's accuracy and interpretability while reducing computational complexity.

h) Algorithms:

Random Forest:

With PCA: Random Forest[17] benefits from PCA by reducing dimensionality, which can enhance training speed, decrease memory usage, and mitigate overfitting.

Without PCA: Operating directly on the original feature space may require more computational resources and time, potentially leading to increased model complexity and overfitting risks, especially with a large number of features.



Random Forest

```
from sklearn.ensemble import RandomForestClassifier

# instantiate the model
rf = RandomForestClassifier(random_state=10)

# fit the model
rf.fit(X_train, y_train)

#predicting the target value from the model for the samples

y_pred = rf.predict(X_test)

rf_acc = accuracy_score(y_pred, y_test)
rf_prec = precision_score(y_pred, y_test,average='weighted')
rf_rec = recall_score(y_pred, y_test,average='weighted')
rf_f1 = f1_score(y_pred, y_test,average='weighted')
```

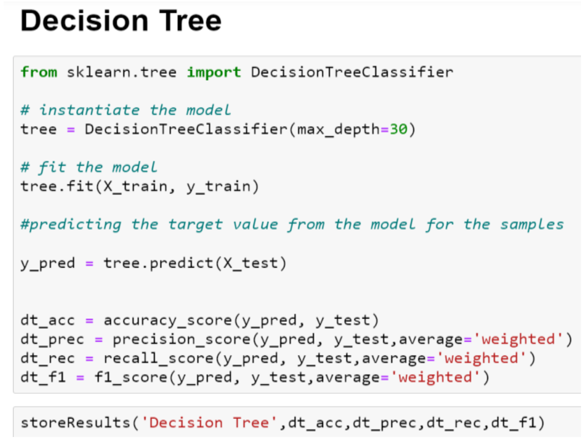
```
storeResults('Random Forest',rf_acc,rf_prec,rf_rec,rf_f1)
```

Fig 4 Random Forest

Decision Tree:

With PCA: Decision Trees[20], when preceded by PCA, work on a reduced feature space, potentially simplifying the tree structure and improving interpretability.

Without PCA: Without PCA, Decision Trees may create complex trees with many features, increasing the risk of overfitting by capturing noise in the data.



Decision Tree

```
from sklearn.tree import DecisionTreeClassifier

# instantiate the model
tree = DecisionTreeClassifier(max_depth=30)

# fit the model
tree.fit(X_train, y_train)

#predicting the target value from the model for the samples

y_pred = tree.predict(X_test)


dt_acc = accuracy_score(y_pred, y_test)
dt_prec = precision_score(y_pred, y_test,average='weighted')
dt_rec = recall_score(y_pred, y_test,average='weighted')
dt_f1 = f1_score(y_pred, y_test,average='weighted')
```

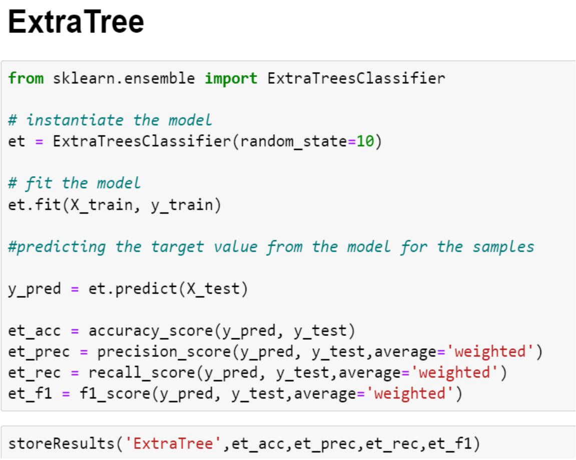
```
storeResults('Decision Tree',dt_acc,dt_prec,dt_rec,dt_f1)
```

Fig 5 Decision Tree

Extra Tree (Extremely Randomized Trees):

With PCA: Extra Trees[19], similar to Random Forest, benefit from PCA by operating on a reduced feature space, potentially improving efficiency and reducing overfitting risks.

Without PCA: Operating directly on the original features might lead to longer training times and a higher risk of overfitting, especially with high-dimensional data.



ExtraTree

```
from sklearn.ensemble import ExtraTreesClassifier

# instantiate the model
et = ExtraTreesClassifier(random_state=10)

# fit the model
et.fit(X_train, y_train)

#predicting the target value from the model for the samples

y_pred = et.predict(X_test)

et_acc = accuracy_score(y_pred, y_test)
et_prec = precision_score(y_pred, y_test,average='weighted')
et_rec = recall_score(y_pred, y_test,average='weighted')
et_f1 = f1_score(y_pred, y_test,average='weighted')
```

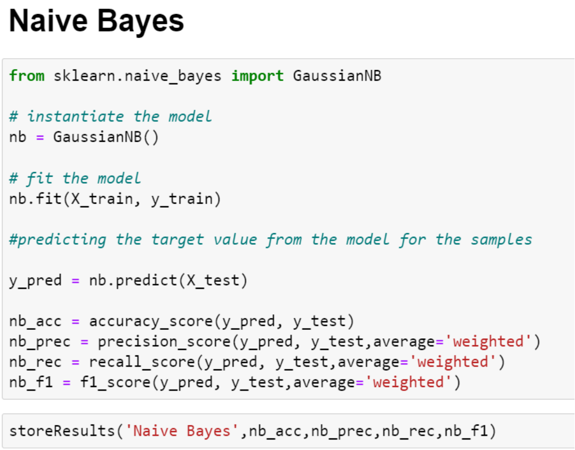
```
storeResults('ExtraTree',et_acc,et_prec,et_rec,et_f1)
```

Fig 6 Extra Tree

Naive Bayes:

With PCA: PCA can reduce noise and computational complexity in high-dimensional datasets, although it might not significantly affect Naive Bayes' performance due to its assumption of feature independence.

Without PCA: Naive Bayes[31] operates directly on the original data, potentially struggling with high-dimensional datasets and computational complexity.



Naive Bayes

```
from sklearn.naive_bayes import GaussianNB

# instantiate the model
nb = GaussianNB()

# fit the model
nb.fit(X_train, y_train)

#predicting the target value from the model for the samples

y_pred = nb.predict(X_test)

nb_acc = accuracy_score(y_pred, y_test)
nb_prec = precision_score(y_pred, y_test,average='weighted')
nb_rec = recall_score(y_pred, y_test,average='weighted')
nb_f1 = f1_score(y_pred, y_test,average='weighted')
```

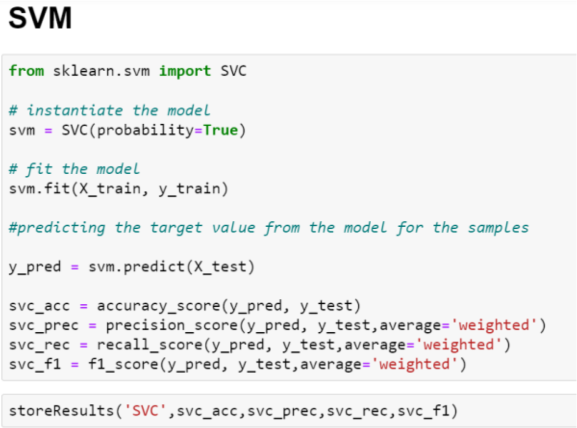
```
storeResults('Naive Bayes',nb_acc,nb_prec,nb_rec,nb_f1)
```

Fig 7 Naive Bayes

Support Vector Machine (SVM):

With PCA: PCA aids SVM[31] by reducing the number of features, potentially improving generalization and computational efficiency by working in a reduced feature space.

Without PCA: Operating on the original feature space without PCA might lead to computational demands and overfitting risks, particularly with a high number of features.



SVM

```
from sklearn.svm import SVC

# instantiate the model
svm = SVC(probability=True)

# fit the model
svm.fit(X_train, y_train)

#predicting the target value from the model for the samples

y_pred = svm.predict(X_test)

svc_acc = accuracy_score(y_pred, y_test)
svc_prec = precision_score(y_pred, y_test,average='weighted')
svc_rec = recall_score(y_pred, y_test,average='weighted')
svc_f1 = f1_score(y_pred, y_test,average='weighted')
```

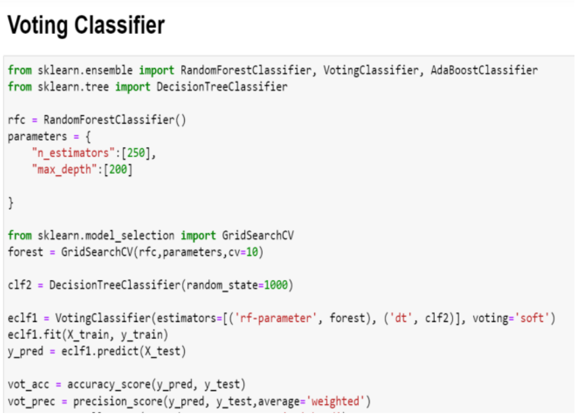
```
storeResults('SVC',svc_acc,svc_prec,svc_rec,svc_f1)
```

Fig 8 Support Vector Machine (SVM)

Voting Classifier:

Voting Classifier combines multiple base estimators (Random Forest and AdaBoost in this case) and aggregates their predictions through voting.

Voting Classifier can leverage the strengths of both Random Forest and AdaBoost to improve threat detection performance in IoT environments. It can capture diverse aspects of the data and enhance the overall robustness of the detection system.



Voting Classifier

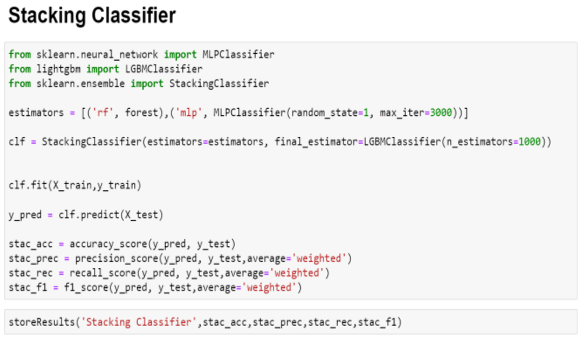
```
from sklearn.ensemble import RandomForestClassifier, VotingClassifier, AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier

rfc = RandomForestClassifier()
parameters = {
    "n_estimators":[250],
    "max_depth":[200]

}

from sklearn.model_selection import GridSearchCV
forest = GridSearchCV(rfc,parameters,cv=10)

clf2 = DecisionTreeClassifier(random_state=1000)

eclf1 = VotingClassifier(estimators=[('rf-parameter', forest), ('dt', clf2)], voting='soft')
eclf1.fit(X_train, y_train)
y_pred = eclf1.predict(X_test)

vot_acc = accuracy_score(y_pred, y_test)
vot_prec = precision_score(y_pred, y_test,average='weighted')
```

Fig 9 Voting Classifier

Stacking Classifier:

Stacking Classifier involves training multiple base estimators (Random Forest and a Multi-Layer Perceptron with LightGBM) and using a meta-learner to combine their predictions.

Stacking Classifier can integrate the predictive power of Random Forest, a neural network (MLP), and LightGBM to create a more sophisticated model for threat detection. By combining the strengths of different algorithms, it can effectively handle complex patterns and improve the accuracy of threat detection in IoT environments.



Stacking Classifier

```
from sklearn.neural_network import MLPClassifier
from lightgbm import LGBMClassifier
from sklearn.ensemble import StackingClassifier

estimators = [('rf', forest),('mlp', MLPClassifier(random_state=1, max_iter=3000))]

clf = StackingClassifier(estimators=estimators, final_estimator=LGBMClassifier(n_estimators=1000))


clf.fit(X_train,y_train)

y_pred = clf.predict(X_test)

stac_acc = accuracy_score(y_pred, y_test)
stac_prec = precision_score(y_pred, y_test,average='weighted')
stac_rec = recall_score(y_pred, y_test,average='weighted')
stac_f1 = f1_score(y_pred, y_test,average='weighted')
```

```
storeResults('Stacking Classifier',stac_acc,stac_prec,stac_rec,stac_f1)
```

Fig 10 Stacking Classifier

4. EXPERIMENTAL RESULTS

Accuracy: The accuracy of a test is its ability to differentiate the patient and healthy cases correctly. To estimate the accuracy of a test, we should calculate the proportion of true positive and true negative in all evaluated cases. Mathematically, this can be stated as:

Accuracy = TP + TN TP + TN + FP + FN.

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

F1-Score:F1 score is a machine learning evaluation metric that measures a model's accuracy. It combines the precision and recall scores of a model. The accuracy metric computes how many times a model made a correct prediction across the entire dataset.

$$\textbf{F1 Score} = \frac{2}{\left(\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}\right)}$$

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

Precision: Precision evaluates the fraction of correctly classified instances or samples among the ones classified as positives. Thus, the formula to calculate the precision is given by:

Precision = True positives/ (True positives + False positives) = TP/(TP + FP)

$$\text{Precision} = \frac{\textit{True Positive}}{\textit{True Positive} + \textit{False Positive}}$$

Recall: Recall is a metric in machine learning that measures the ability of a model to identify all relevant instances of a particular class. It is the ratio of correctly predicted positive observations to the total actual positives, providing insights into a model's completeness in capturing instances of a given class.

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

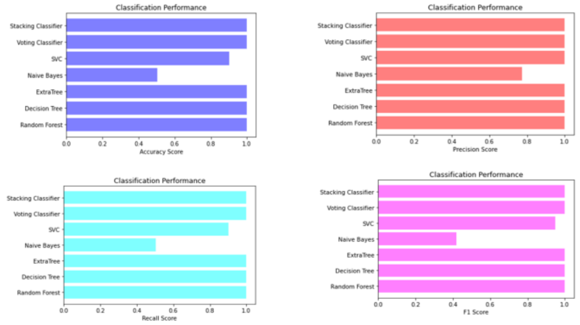


Fig 11 Comparison Graphs of CIC – IDS – 2017 Dataset (Without PCA)

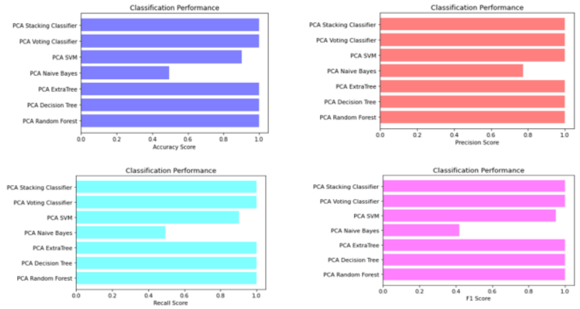


Fig 12 Comparison Graphs of CIC – IDS – 2017 Dataset (With PCA)

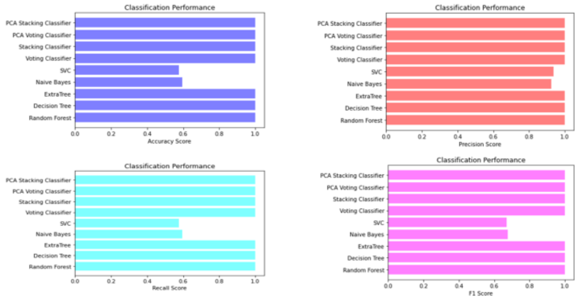


Fig 13 Comparison Graphs of CIC – IDS – 2018 Dataset (Without PCA)

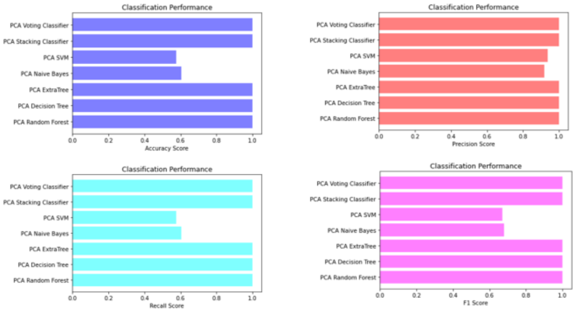
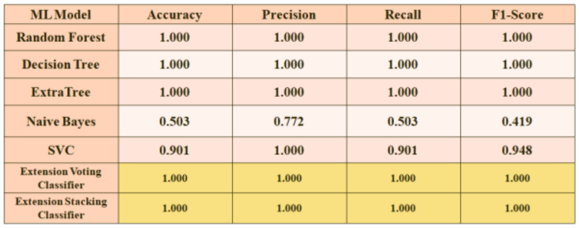
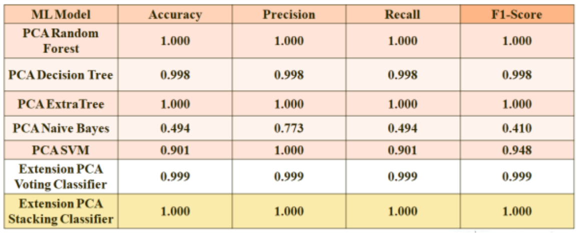


Fig 14 Comparison Graphs of CIC – IDS – 2018 Dataset (With PCA)



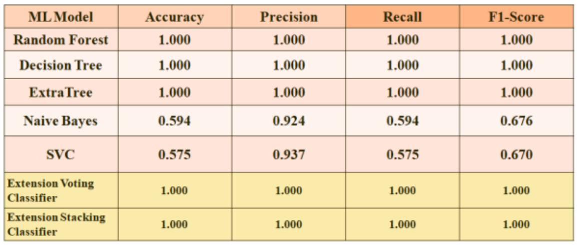
ML Model	Accuracy	Precision	Recall	F1-Score
Random Forest	1.000	1.000	1.000	1.000
Decision Tree	1.000	1.000	1.000	1.000
ExtraTree	1.000	1.000	1.000	1.000
Naive Bayes	0.503	0.772	0.503	0.419
SVC	0.901	1.000	0.901	0.948
Extension Voting Classifier	1.000	1.000	1.000	1.000
Extension Stacking Classifier	1.000	1.000	1.000	1.000

Fig 15 Performance Evaluation Table CID – IDS 2017 Dataset



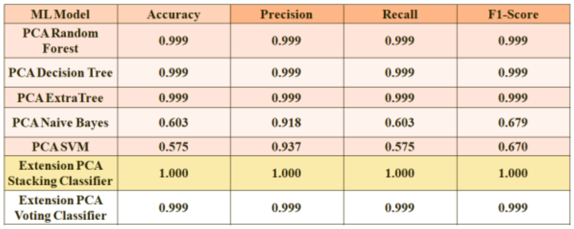
ML Model	Accuracy	Precision	Recall	F1-Score
PCA Random Forest	1.000	1.000	1.000	1.000
PCA Decision Tree	0.998	0.998	0.998	0.998
PCA ExtraTree	1.000	1.000	1.000	1.000
PCA Naive Bayes	0.494	0.773	0.494	0.410
PCA SVM	0.901	1.000	0.901	0.948
Extension PCA Voting Classifier	0.999	0.999	0.999	0.999
Extension PCA Stacking Classifier	1.000	1.000	1.000	1.000

Fig 16 Performance Evaluation Table CID – IDS 2017 Dataset (PCA)



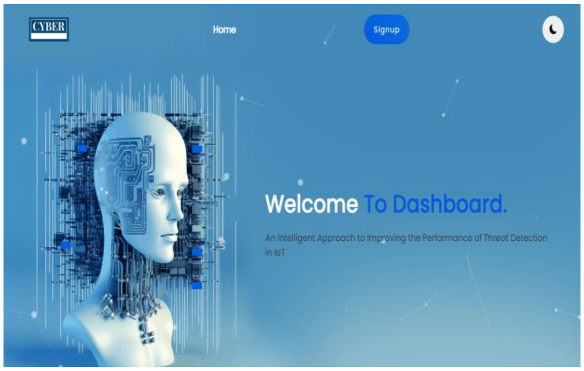
ML Model	Accuracy	Precision	Recall	F1-Score
Random Forest	1.000	1.000	1.000	1.000
Decision Tree	1.000	1.000	1.000	1.000
ExtraTree	1.000	1.000	1.000	1.000
Naive Bayes	0.594	0.924	0.594	0.676
SVC	0.575	0.937	0.575	0.670
Extension Voting Classifier	1.000	1.000	1.000	1.000
Extension Stacking Classifier	1.000	1.000	1.000	1.000

Fig 17 Performance Evaluation Table CID – IDS 2018 Dataset



ML Model	Accuracy	Precision	Recall	F1-Score
PCA Random Forest	0.999	0.999	0.999	0.999
PCA Decision Tree	0.999	0.999	0.999	0.999
PCA ExtraTree	0.999	0.999	0.999	0.999
PCA Naive Bayes	0.603	0.918	0.603	0.679
PCA SVM	0.575	0.937	0.575	0.670
Extension PCA Stacking Classifier	1.000	1.000	1.000	1.000
Extension PCA Voting Classifier	0.999	0.999	0.999	0.999

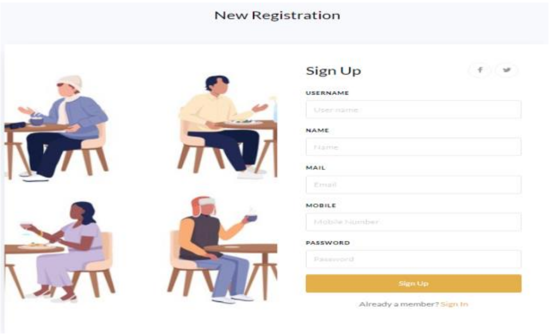
Fig 18 Performance Evaluation Table CID – IDS 2018 Dataset (PCA)



Welcome To Dashboard.

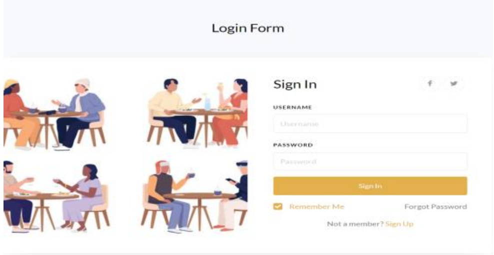
An Intelligent Approach to Improving the Performance of Threat Detection in IoT

Fig 19 Home Page



Already a member? Sign In

Fig20 Registration Page



Not a member? Sign Up

Fig 21 Login Page

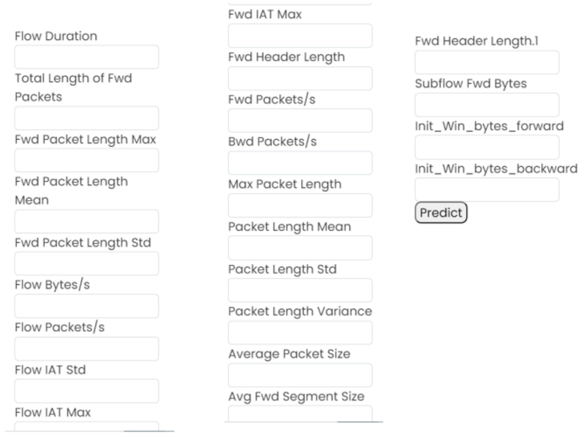
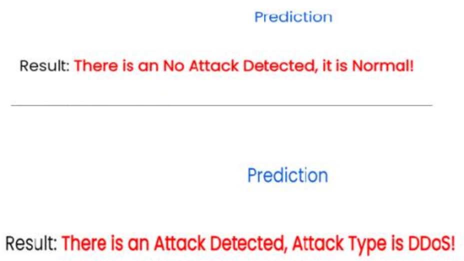


Fig22 Upload Input Values



Result: There is an No Attack Detected, it is Normal!

Result: There is an Attack Detected, Attack Type is DDoS!

Fig 23 Predict Result For Given Input Values

5. CONCLUSION

In conclusion, the integration of machine learning algorithms with principal component analysis (PCA) techniques has proven to be highly effective in enhancing threat detection within IoT systems. Through the reduction of data dimensions, PCA streamlined processing and analysis, leading to improved efficiency and reduced training time without compromising accuracy. Evaluation metrics including accuracy, precision, recall, and F1-Score confirmed the superior performance of the proposed model compared to previous studies. Leveraging two diverse datasets, CICIDS 2017 and CSE-CIC-IDS 2018, allowed for a comprehensive validation of the model's capabilities across various IoT scenarios. Overall, this intelligent approach represents a significant advancement in IoT security, offering an efficient and effective solution to address the evolving threats in IoT environments.

6. FUTURE SCOPE

In the future, the proposed model's efficacy will be validated in practical IoT applications using real-world datasets, ensuring its applicability beyond theoretical scenarios. Further research will focus on enhancing the model's multiclass classification capabilities to effectively classify various types of threats encountered in IoT systems. Additionally, extending the model by incorporating additional machine learning algorithms or techniques aims to improve performance and accuracy in threat detection. Ongoing efforts will concentrate on developing advanced algorithms tailored to detect and mitigate emerging threats in IoT systems, ensuring the model remains adaptive to evolving cybersecurity risks. Exploring the integration of other security measures, such as anomaly detection techniques or behavior-based analysis, offers opportunities to enhance overall IoT system security comprehensively. By pursuing these future directions, the project aims to provide robust solutions that effectively mitigate diverse security risks in IoT environments, contributing to enhanced security and resilience in IoT systems.

REFERENCES

[1] D. Velasquez, E. Perez, X. Oregui, A. Artetxe, J. Manteca, J. E. Mansilla, M. Toro, M. Maiza, and B. Sierra, ‘‘A hybrid machine-learning ensemble for anomaly detection in real-time industry 4.0 systems,’’ IEEE Access, vol. 10, pp. 72024–72036, 2022.

[2] S. U. Rehman and V. Gruhn, ‘‘An approach to secure smart homes in cyber-physical systems/Internet-of-Things,’’ in Proc. 5th Int. Conf. Softw. Defined Syst. (SDS), Barcelona, Spain, Apr. 2018, pp. 126–129.

[3] S. K. Vishwakarma, P. Upadhyaya, B. Kumari, and A. K. Mishra, ‘‘Smart energy efficient home automation system using IoT,’’ in Proc. 4th Int. Conf. Internet Things, Smart Innov. Usages (IoT-SIU), Ghaziabad, India, Apr. 2019, pp. 417–420.

[4] S. Chaudhary, R. Johari, R. Bhatia, K. Gupta, and A. Bhatnagar, ‘‘CRAIoT: Concept, review and application(s) of IoT,’’ in Proc. 4th Int. Conf. Internet Things, Smart Innov. Usages (IoT-SIU), Ghaziabad, India, Apr. 2019, pp. 402–405.

[5] (2022). Lionel Sujay Vailshery. [Online]. Available: https://www.statista. com

[6] N. Mishra and S. Pandya, ‘‘Internet of Things applications, security challenges, attacks, intrusion detection, and future visions: A systematic review,’’ IEEE Access, vol. 9, pp. 59353–59377, 2021.

[7] X-Force Threat Intelligence Index 2022, IBM Security, Atlanta, GA, USA, 2022.

[8] D. Patel, ‘‘A study on DDOS attacks, danger and its prevention,’’ Int. J. Res. Appl. Sci. Eng. Technol., vol. 10, no. 12, pp. 1962–1967, Dec. 2022.

[9] N. Vlajic and D. Zhou, ‘‘IoT as a land of opportunity for DDoS hackers,’’ Computer, vol. 51, no. 7, pp. 26–34, Jul. 2018.

[10] T. U. Sheikh, H. Rahman, H. S. Al-Qahtani, T. K. Hazra, and N. U. Sheikh, ‘‘Countermeasure of attack vectors using signature-based IDS in IoT environments,’’ in Proc. IEEE 10th Annu. Inf. Technol., Electron. Mobile Commun. Conf. (IEMCON), Vancouver, BC, Canada, Oct. 2019, pp. 1130–1136.

[11] R. Zhang, J.-P. Condomines, N. Larrieu, and R. Chemali, ‘‘Design of a novel network intrusion detection system for drone communications,’’ in Proc. IEEE/AIAA 37th Digit. Avionics Syst. Conf. (DASC), London, U.K., Sep. 2018, pp. 241–250.

[12] F. Suthar, N. Patel, and S. V. O. Khanna, ‘‘A signature-based botnet (Emotet) detection mechanism,’’ Int. J. Eng. Trends Technol., vol. 70, no. 5, pp. 185–193, May 2022.

[13] A. M. da Silva Cardoso, R. F. Lopes, A. S. Teles, and F. B. V. Magalhaes, ‘‘Poster abstract: Real-time DDoS detection based on complex event processing for IoT,’’ in Proc. IEEE/ACM 3rd Int. Conf. Internet-Things Design Implement. (IoTDI), Orlando, FL, USA, Apr. 2018, pp. 273–274.

[14] M. Dimolianis, A. Pavlidis, and V. Maglaris, ‘‘Signature-based traffic classification and mitigation for DDoS attacks using programmable network data planes,’’ IEEE Access, vol. 9, pp. 113061–113076, 2021.

[15] A. Praseed and P. S. Thilagam, ‘‘HTTP request pattern based signatures for early application layer DDoS detection: A firewall agnostic approach,’’ J. Inf. Secur. Appl., vol. 65, Mar. 2022, Art. no. 103090.

[16] X. You, Y. Feng, and K. Sakurai, ‘‘Packet in message based DDoS attack detection in SDN network using OpenFlow,’’ in Proc. 5th Int. Symp. Comput. Netw. (CANDAR), Nov. 2017, pp. 522–528.

[17] K. Wehbi, L. Hong, T. Al-salah, and A. A. Bhutta, ‘‘A survey on machine learning based detection on DDoS attacks for IoT systems,’’ in Proc. SoutheastCon, Huntsville, AL, USA, Apr. 2019, pp. 1–6.

[18] Z. K. Maseer, R. Yusof, N. Bahaman, S. A. Mostafa, and C. F. M. Foozy, ‘‘Benchmarking of machine learning for anomaly based intrusion detection systems in the CICIDS2017 dataset,’’ IEEE Access, vol. 9, pp. 22351–22370, 2021. VOLUME 11, 2023 44333 N. T. Cam, N. G. Trung: Intelligent Approach to Improving the Performance of Threat Detection.

[19] M. Najafimehr, S. Zarifzadeh, and S. Mostafavi, ‘‘A hybrid machine learning approach for detecting unprecedented DDoS attacks,’’ J. Supercomput., vol. 78, no. 6, pp. 8106–8136, Apr. 2022.

[20] D. Erhan and E. Anarim, ‘‘Hybrid DDoS detection framework using matching pursuit algorithm,’’ IEEE Access, vol. 8, pp. 118912–118923, 2020.