Journal of
Sensor and Actuator Networks

MDPI

Article

Intrusion Detection System Using Feature Extraction with Machine Learning Algorithms in IoT

Dhiaa Musleh [1], Meera Alotaibi [1], Fahd Alhaidari [2], Atta Rahman [1,*] and Rami M. Mohammad [3]

[1] Department of Computer Science, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, P.O. Box 1982, Dammam 31441, Saudi Arabia
[2] SAUDI ARAMCO Cybersecurity Chair, Department of Networks and Communications, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, P.O. Box 1982, Dammam 31441, Saudi Arabia
[3] SAUDI ARAMCO Cybersecurity Chair, Department of Computer Information Systems, College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, P.O. Box 1982, Dammam 31441, Saudi Arabia
* Correspondence: aaurrahman@iau.edu.sa

Abstract: With the continuous increase in Internet of Things (IoT) device usage, more interest has been shown in internet security, specifically focusing on protecting these vulnerable devices from malicious traffic. Such threats are difficult to distinguish, so an advanced intrusion detection system (IDS) is becoming necessary. Machine learning (ML) is one of the promising techniques as a smart IDS in different areas, including IoT. However, the input to ML models should be extracted from the IoT environment by feature extraction models, which play a significant role in the detection rate and accuracy. Therefore, this research aims to introduce a study on ML-based IDS in IoT, considering different feature extraction algorithms with several ML models. This study evaluated several feature extractors, including image filters and transfer learning models, such as VGG-16 and DenseNet. Additionally, several machine learning algorithms, including random forest, K-nearest neighbors, SVM, and different stacked models were assessed considering all the explored feature extraction algorithms. The study presented a detailed evaluation of all combined models using the IEEE Dataport dataset. Results showed that VGG-16 combined with stacking resulted in the highest accuracy of 98.3%.

Keywords: intrusion detection system; Internet of Things; feature extractors; machine learning

Citation: Musleh, D.; Alotaibi, M.; Alhaidari, F.; Rahman, A.; Mohammad, R.M. Intrusion Detection System Using Feature Extraction with Machine Learning Algorithms in IoT. *J. Sens. Actuator Netw.* **2023**, *12*, 29. https://doi.org/10.3390/jsan12020029

Academic Editors: Mohamed Amine Ferrag, Leandros Maglaras and Mohamed Benbouzid

Received: 6 March 2023
Revised: 21 March 2023
Accepted: 27 March 2023
Published: 29 March 2023

1. Introduction

The Internet of Things has recently been one of the most important research topics. The IoT is a new technological paradigm defined as a global network of connected electronic devices. It aims to improve daily life by automating normal daily operations in all aspects of life without human intervention. The number of devices connected to IoT has been raised significantly, and an increase in attacks against IoT devices has accompanied this growth. Security concerns about the impact of these attacks on connected devices have naturally increased. In addition to the sensitivity of the information available on IoT devices, it was necessary to find solutions to detect and respond to these attacks [1].

Because of its weaknesses, the Internet of Things is vulnerable to assaults and security threats [2–11]. Researchers attempted to categorize attacks, vulnerabilities, and security concerns on the Internet of Things so that researchers could more easily identify answers. For example, according to the layers of the IoT architecture, the researchers categorized the vulnerabilities, and physical security hardening is lacking. Unconfident data storage and transfer, shortage of clarity and device management, botnets, insecure passcodes, ecosystem interfaces, and AI-based assaults have all been concerns for devices on the IoT. While some academics emphasized IoT's vulnerabilities and security risks, others did not.

Among these issues, the researchers pointed out that because IoT employs traditional network architecture, it inherits its flaws [6]. In addition, the rise in terminal devices (end nodes) with limited processing capabilities was one of the most significant and powerful vulnerabilities exploited by attackers [7]. End node manufacturers design them to work without paying attention to security concerns. As a result, these devices must be monitored and managed to protect networks from threats and reach a higher degree of IoT security [2]. The Open Web Application Security Project (OWASP) produced a comprehensive record of IoT attack areas and locations in IoT systems as a section of its Internet of Things plan. In addition, it recorded applications where alleged unauthorized actions could be encountered [12]. Following is a summary of the IoT attack areas:

a. Devices are probably the principal mechanism from which attacks could be launched, such as memory, firmware, physical interface, web interface, or network services. In addition, weak settings, out-of-date components, and insufficient update procedures are critical parts of any device attackers could utilize.
b. Communication channels could also be points of attack in IoT systems. Usually, protocols contain security failings that impact the whole system, such as denial of service (DoS) attacks and spoofing.
c. Applications and software, generally, could be hacked due to a shortcoming in web applications or software systems. This approach is usually utilized to steal user passwords or spread malware files or programs.

Intrusion detection systems (IDSs) are essential security techniques to conserve network security, and they are installed at a fatal location in the network [12,13]. Traditional systems contain source and preliminary processing of data and a decision-making technique. This process contains the collection of raw data from host or network traffic. By analyzing the network data traffic, an intrusion detection system can classify the network behavior as normal or abnormal [12] and then process the features passed by the decision-making method to recognize threats [13]. Three main ways to detect intrusions are signature-based IDS, anomaly-based IDS, and a hybrid of signature- and anomaly-based IDS [13]. Dynamic anomaly-based network detection systems are flexible and superior to static signature-based network intrusion systems because the former can detect new attacks [14]. They use artificial intelligence (AI) algorithms that are made of both machine learning (ML) and deep learning (DL) architectures. On the other hand, IDSs detect signatures and patterns and then match them with the predefined signature of misuses, which could be worthless with unknown attacks [13]. The three significant categories of intrusion detection systems are host intrusion detection systems (HIDSs), network intrusion detection systems (NIDSs), and network node intrusion detection systems (NNIDSs) [13]. The HIDS is installed on the entire network of machines and other parts of the physical and virtual networks and protocols. The NIDS protects vulnerable network parts where the attack opportunities are high. IDSs consider network or host-based methods to recognize and distract attacks. These methods search for attack signatures with patterns that indicate malignant action or suspicious activity. Based on where an IDS is searching for the pattern, either in network traffic or log files, it is classified as network- or host-based [15].

Machine learning methods are extensively used to build network intrusion detection systems because of their capability to grasp new intrusions [16]. To develop accurate algorithms that can cluster, classify, and predict, it is vital to utilize considerable-size data sets using supervised machine learning techniques such as SVM and naïve Bayes. In addition, decision trees demonstrate their simplicity, rapid adaptability, and accuracy. In addition, neural networks have been widely used to characterize anomaly and misuse patterns [12,16]. Accuracy and interpretability are essential factors of artificial intelligence models. To achieve accuracy and interpretability, machine learning and deep learning techniques must be considered. For example, black-box algorithms provide higher accuracy, while white-box algorithms provide feature engineering [14].

Significance of the Study and Contributions

Threats are overwhelmingly increasing in several fields, such as IoT, online banking, industries, and healthcare. Further, IoT usage has been widely accepted due to its success in wearables, smart homes, and smart cities around the world. Unfortunately, IoT devices work on public networks with bounded computing power and limited storage and bandwidth. As a result, they are more vulnerable to assaults than other end-point devices. Several techniques have been proposed in the literature, yet there is great room for improvement, especially when it comes to intrusion detection.

To overcome the issues, the contributions of the undergoing study are as follows:

1- A comprehensive review of literature on the applications of machine learning and deep learning models in intrusion detection using numerical and image-based datasets.
2- Dataset preprocessing and balancing with SMOTE technique.
3- Feature extraction using various approaches with stacked machine learning models such as kNN, sequential minimal optimization (SMO), and random forest to distinguish between malicious and normal network traffic patterns.
4- Experiment for the validation of the proposed models.

The experimental results on the IEEE Dataport image dataset reveal that the proposed techniques are promising in terms of accuracy.

The rest of the article is sectioned as follows: A comprehensive review of the related work is presented in Section 2. Section 3 describes the methodological steps, Section 4 provides the experimental results, and Section 5 concludes the paper.

2. Related Work

This section discusses the work done in the field of primary machine learning techniques used in IoT traffic.

Rose et al. [17] generated a dataset and developed a model to detect and investigate the possibilities of utilizing network profiling and machine learning to protect IoT against cyber-attacks. The authors suggested anomaly-based intrusion detection system profiles and monitoring all networked devices constantly and aggressively to identify IoT device tampering attempts and suspicious network transactions. They evaluated the suggested methodology's performance using regular and malicious network traffic on the Cyber-Trust testbed. The experimental findings reveal that the suggested anomaly detection system produces good results, with a 98.35% accuracy and 98.35% false-positive alerts.

Ali et al. [18] present a general machine learning strategy for identifying IoT devices and evaluating the trained models against four publicly available datasets. NFStream extracted 85 attributes from packet capture (.pcap) files to better identify IoT devices in the network using machine learning models. The authors used the information gain approach to choose 20 characteristics and trained six machine learning models in the tests. In the training phase, the authors achieved high accuracy, reaching 99% for IoT device identification using random forest and naïve Bayes classifiers.

El-Sayed et al. [19] examined and compared seven different supervised learning algorithms with various difficulty levels to pick the best one. The seven algorithms were separated into two groups: The category of CNN classifiers included two-layer CNN, four-layer CNN, VGG16 and logistic regression, support vector machine, and K-nearest neighbors, and the category of ordinary classifiers included logistic regression, support vector machine, and K-nearest neighbors. Experimental findings reveal that the SVM algorithm obtains the maximum performance of 94% on MobileNetv2 features because of its rapid and steady training performance with fewer resources compared with other models. Le K-H et al. [20] present IMIDS, an intelligent intrusion detection system (IDS) for IoT devices. IMIDS's core is a lightweight convolutional neural network model that can categorize numerous cyber threats and surpasses its competitors with an average F-measure of 97.22%. Furthermore, after being further educated by the data supplied by the assault data generator, IMIDS's detection performance significantly increased. These findings show that IMIDS may be used as an IDS in IoT.

Joo et al. [21] proposed a deep learning-based IoT intrusion detection system. The categorization was performed with a CNN; the best score was 86.2%. Second, machine learning classifiers were employed for the hybrid technique instead of ultimately linked layers from the vanilla CNN, which delivered roughly 87% with the additional tree classifier. Finally, the Xception model was merged with the bidirectional GRU, yielding the best accuracy at 95.6%. For quicker identification and classification of new malware, Bendiab et al. [22] propose a unique IoT technique that analyzes malware traffic based on DL and visual representation (zero-day malware). The suggested technique detects fraudulent network traffic at the package level, lowering detection time and optimistic outcomes thanks to the deployed deep learning. To test the efficacy of the proposed technique, the authors created a dataset of 1000 .pcap files of benign and virus traffic obtained from several network traffic sources. The Residual Neural Network (ResNet50) trial findings are quite encouraging, with a detection rate of 94.50% for malware traffic.

Six machine learning (ML) approaches were tested for their ability to identify MQTT-based attacks [23]. Packet-based, unidirectional, and bidirectional flow characteristics were evaluated at three abstraction levels. An MQTT simulated dataset was created and used for the training and assessment operations. The experimental findings showed that the suggested ML models were sufficient for the IDS needs of MQTT-based networks. Furthermore, the findings highlight the significance of employing flow-based characteristics to distinguish MQTT-based attacks from innocuous traffic, whereas packet-based features are sufficient for typical networking assaults. The results reveal that the model has the highest accuracy of 99.04%. Sapre et al. [24] employed the KDDCup99 and the NSLKDD, two widely used intrusion detection datasets, in their study. Their major objective was to thoroughly compare both datasets by analyzing the performance of multiple machine learning (ML) classifiers trained on them using a more extensive range of classification criteria than prior studies. Because the classifiers trained on the KDDCup99 dataset were 20.18% less accurate on average, the authors concluded that the NSL-KDD dataset is of better quality than the KDDCup99 dataset. This is because classifiers trained on the KDD-Cup99 dataset were biased toward redundancy, allowing them to attain a higher accuracy of 96.83%. Liu et al. [25] looked at assaults that might affect sensors and networks in IoT scenarios using the NSL-KDD dataset. Moreover, the authors investigated eleven machine learning techniques and provided the findings to identify the introduced assaults. They showed that tree-based approaches and ensemble methods surpass the other machine learning methods evaluated through numerical analysis. With 97% accuracy, 90.5% Matthews correlation coefficient (MCC), and 99.6% area under the curve (AUC), XGBoost is the best of the supervised algorithms. Furthermore, the expectation-maximization (EM) technique, which is an unsupervised approach, performs exceptionally well in identifying assaults in the NSL-KDD dataset and beats the naïve Bayes classifier by 22.0% in terms of accuracy.

To distinguish benign from malicious nodes, Amouri et al. [26] used a methodology that consists of two stages: in the first stage, the data are collected by dedicated sniffers (DSs), and then the CCI is generated and is regularly sent to the super node (SN). After that, in the second stage, the SN processes a linear regression method on the collected CCIs from different DSs to distinguish benign from malicious nodes. Using two mobility models, namely random waypoint (RWP) and Gauss Markov, the detection characterization is shown for several extreme cases in the network (GM). The black hole and distributed denial of service (DDoS) assaults are two harmful activities utilized at work. Nodes with high-velocity situations showed detection rates of over 98%, while nodes with low-velocity scenarios showed detection rates of approximately 90%. Fenanir et al. [27] created a lightweight intrusion detection system (IDS) using two machine learning techniques: the filter-based method was used to pick features due to its cheap computational cost. A comparison of logistic regression (LR), naïve Bayes (NB), decision tree (DT), random forest (RF), k-nearest neighbor (KNN), support vector machine (SVM), and multilayer perceptron yielded the feature classification approach to the system (MLP). Finally, the DT method was chosen for the system due to its excellent performance across various datasets. The

study's outcomes might help choose the optimum feature selection approach for machine learning; the data suggest that the best results are 98% accuracy.

Islam et al. [28] pointed out numerous types of IoT threats and discussed shallow IDSs in the IoT environment (such as decision tree (DT), random forest (RF), and support vector machine (SVM)), as well as DL (deep neural network (DNN), deep belief network (DBN), long short-term memory (LSTM), stacked LSTM, and bidirectional LSTM (Bi-LSTM))-based IDSs. The models' execution was assessed using five standard datasets: NSL-KDD, IoTDevNet, DS2OS, IoTID20, and the IoT Botnet dataset. The performance of shallow/deep machine learning-based IDSs was evaluated using several performance indicators such as accuracy, precision, recall, and F1-score. According to the research, a machine learning IDS surpasses shallow machine learning in detecting IoT threats; the most remarkable outcome of the studies is the accuracy of 98.79%. Using characteristics from the UNSW-NB15 dataset, Ahmad et al. [29] suggest feature clusters regarding its flow, Message Queuing Telemetry Transport (MQTT), and Transmission Control Protocol (TCP). Overfitting, the curse of dimensionality, and an unbalanced dataset are no longer issues. The proposed method used supervised machine learning (ML) methods such as random forest (RF), support vector machine, and artificial neural networks on the clusters. The model reaches 98.67% and 97.37% accuracy using RF in binary and multiclass classification. Utilizing RF on flow and MQTT features, TCP features, and top features from both clusters, classification accuracies of 96.96%, 91.4%, and 97.54% were obtained using cluster-based approaches. A two-stage hybrid technique was proposed by Saba et al. in [30]. To increase the accuracy of the suggested system, the genetic algorithm (GA) is first used to pick acceptable characteristics. The support vector machine (SVM), ensemble classifier, decision tree, and other well-known machine learning (ML) algorithms are then used. Using the NSL-KDD database, they attained a 99.8% accuracy using 10-fold cross-validation. Based on a hybrid convolutional neural network model, Smys et al. [31] suggested an intrusion detection system for IoT networks that can identify many forms of assaults. The proposed paradigm may be used in a variety of IoT scenarios. The proposed study is validated and compared to machine learning and deep learning models. The suggested hybrid model is more sensitive to threats in the IoT network, with a 98.6% accuracy rate. Papafotikas et al. [32] propose a digital system incorporating a machine learning (ML)-based clustering method for identifying suspected activities while using current supply characteristic dissipation. The K-means clustering algorithm accompanied by supervised training is used in this prototype system. This research demonstrated the successful identification of suspicious activity in intelligent IoT devices. Similarly, a study in [33] proposed an IDS approach using a fused machine learning model. Three datasets, namely KDD, CUP-99, and NetML-2020, were fused under a novel-built machine learning-based architecture. The trained model was promising in terms of accuracy of 95.18%.

Further, several researchers in the literature have comprehensively surveyed and emphasized the significance of machine learning and deep learning models in the IDSs involving IoT networks [34–37], especially in conjunction with cloud computing, namely the Cloud of Things security aspect [38]. This is mainly because it involves several intermediate public networks and stakeholders, making it more vulnerable to attacks. Table 1 summarizes related work approaches, including the techniques used, dataset type, and the respective study's advantages and disadvantages.

Table 1. Summary of the literature review.

Ref.	Year	Dataset Type	Algorithms	Key Results	Advantages	Disadvantages
[17]	2021	Image dataset	A novel IDS in the ML framework	98.35%	A new approach to detect malware and intrusion using image processing for IoT	Need resources and computing time to achieve results and to understand what each image means
[18]	2021	Numerical dataset	Random forest and naïve Bayes algorithms in the ML model	99%	Using different types of machine learning to understand the problem	Too expensive in terms of resources and computing time, and the authors did not propose any base model for the future
[19]	2021	Image dataset	Two-layer CNN, four-layer CNN, VGG16, CNN classifiers, SVM, and K-NN in the ML model	94%	Using a hybrid technique to extract the best features from the images using VGG and CNN	Too heavy classification network and no improvement in key results
[20]	2021	Numerical dataset	Novel convolutional neural network in ML model	97.22%	A faster IDS	Despite the fast performance, the model is light with the heavy traffic amount
[21]	2021	Image dataset	Convolutional neural network (CNN)	95.6%	Connected layers and faster learning	The model uses heavy resources and without achieving higher accuracy
[22]	2021	Image dataset	Residual Neural Network (ResNet50)	94.50%	The model used a hybrid approach to detect intrusions	The authors used a small amount of the data to achieve the findings without testing on the larger networks
[23]	2020	Numerical dataset	LR, NB, k-NN, SVM, DT, and RF	99.04%	Using different ML algorithms	The model was focused on flow-based detection, not packet-based detection
[24]	2019	Numerical dataset	ANN, SVM, NBC, and random forest	91.5%	The authors used different dataset types	The model was trained separately on the dataset, and without finding a model that is trained on different intrusion features
[25]	2020	Numerical dataset	Tree-based XGBoost, MCC, AUC, ME, and naïve Bayes classifier	97%	The model used an XGBoost algorithm	The model was applied to a couple of scenarios without generating a model that could handle several scenarios
[26]	2020	Numerical dataset	Tree-based; among the supervised algorithms, XGBoost ranks first, followed by Matthew's correlation coefficient (MCC), area under the curve (AUC), expectation-maximization (EM) algorithm, naïve Bayes classifier	90%	The model used different approaches to tackle the IDS problems	The model was focused on DDOS and DOS attacks.
[27]	2019	Numerical dataset	LR, NB, DT, RF, KNN, SVM, and MLP	98%	Applied different machine learning algorithms	More feature selection processes are needed to achieve better findings
[28]	2021	Numerical dataset	DT, RF, SVM, (DNN), deep belief network (DBN), long short-term memory (LSTM), stacked LSTM, bidirectional LSTM (Bi-LSTM))	98.2%	Applied different machine learning algorithms using different datasets	The proposed model did not recommend specific datasets or algorithms to be used as a base model
[29]	2021	Numerical dataset	RF, SVM, and ANN	96.96%	Applied different machine learning algorithms	More feature selection processes are needed to achieve better findings
[30]	2021	Numerical dataset	GA, SVM, ensemble classifier, and DT	99.8%	Applied hyperparameter and K-fold	The model proposed a multiclass without presenting the features or the class features

Table 1. *Cont.*

Ref.	Year	Dataset Type	Algorithms	Key Results	Advantages	Disadvantages
[31]	2020	Numerical dataset	Different ML algorithms in hybrid convolutional neural network module	98%	The model takes advantage of deep learning feature extraction	The model was applied to a couple of scenarios without generating a model that could handle several scenarios
[32]	2019	Numerical dataset	K-means algorithms	-	Using clustering to tackle the problem	The model still needs to be trained from supervised algorithms to achieve findings
[33]	2023	Numerical dataset	Fused machine learning	95.18%	Used machine learning fusion with three datasets (KDD, CUP-99, NetML-2020)	Accuracy can be further fine-tuned

3. Methodology

This section includes the methodology, an overview of the dataset, data preprocessing, and a brief description of the algorithms and techniques used for feature extractions. The parameters used to evaluate the models are then presented. The objective of the modeling presented in this work is to distinguish regular traffic from malicious traffic. Therefore, several models are developed on the obtained dataset, in image format, and compared. Figure 1 illustrates an example of a stacked model with multiple feature extractors. Firstly, the data are pre-processed to improve model accuracy as further discussed in the subsequent section. Different feature extractors are then applied to extract relevant features; individual and multiple feature extractors are used to facilitate this. Finally, different machine learning algorithms are trained to classify the traffic.

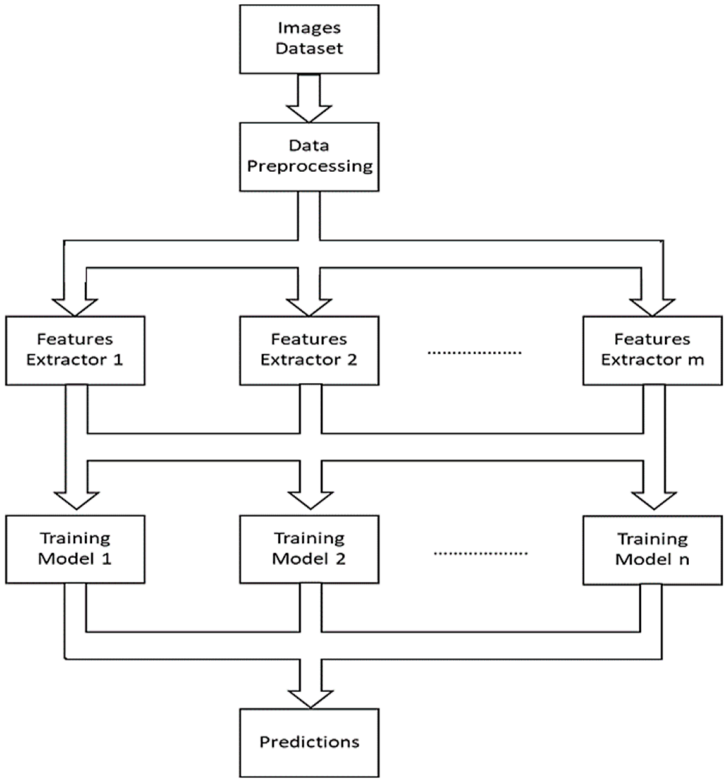

Figure 1. An Example of a Stacked Model.

3.1. Dataset Description

The dataset utilized in this research was obtained from IEEE Dataport [39]. The dataset contains more than 800 samples of normal and malicious traffic in binary visualization format for model training. It is a benchmark dataset for intrusion detection systems in the

image format. Due to the rich visual features, it has more significance than a numerical dataset. Furthermore, additional data are also provided in image format, generated from the five attack scenarios presented in [39].

Figure 2 provides examples of normal and malicious traffic packages in image format. It is clear from the examples provided that two images of normal or malicious traffic can be significantly different and, in certain instances, maybe like the other category. Therefore, by adopting machine learning techniques, there is an opportunity to differentiate between the two categories with high accuracy.

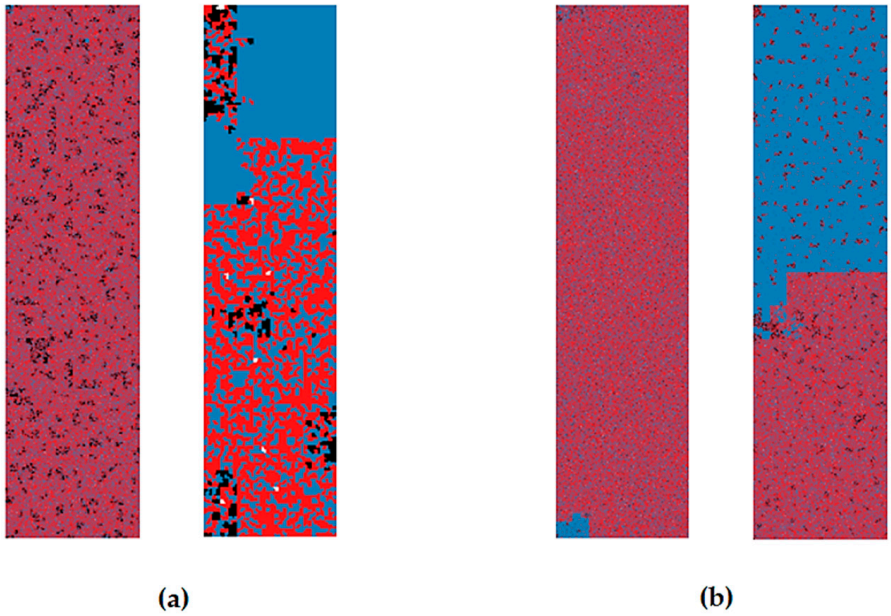

Figure 2. Two images for normal traffic (**a**) and two for malicious traffic (**b**).

3.2. Data Preprocessing

Data preprocessing is a crucial element before applying any machine learning model. It is required to address any inconsistency, errors, or noise in the data [40]. The model's performance can be significantly impacted if the data are poorly preprocessed. Data preprocessing consists of several steps: cleaning, transformation, and feature selection. For image preprocessing, filters are deployed to denoise the dataset. Several filters have been developed and are widely used. In this research, the performance of the models utilizing different filters is considered as discussed in the following sections.

3.2.1. Synthetic Minority Oversampling Technique Filter

Once the dataset is cleaned, it is essential to ensure that it is balanced. Imbalanced datasets might have a significant impact on the performance of the overall model. One way to balance the dataset is to reduce the number of instances of all classes to match the number of instances of the class with the lowest number of instances. However, this will reduce the number of training data, affecting the training process. Another approach is using the synthetic minority oversampling technique (SMOTE) approach to handle the imbalanced dataset [41].

SMOTE is a technique widely used to oversample the minority class. The oversampling is performed by generating more synthetic examples of the minority class throughout the length of the line segments connecting some/all the minority class nearest neighbors. Figure 3 illustrates how the newly generated samples $w_1 \ldots w_4$ are generated between the existing data under the minority class $y_1 \ldots y_4$ [41].

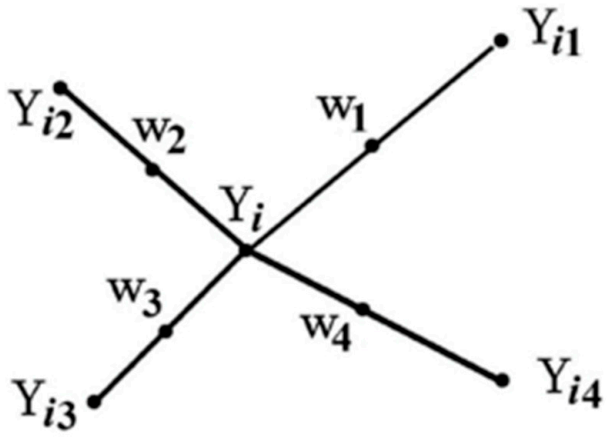

Figure 3. Oversampling technique used in SMOTE.

The SMOTE algorithm is employed using the following steps:

- For each sample, find the k-nearest neighbors.
- Then, select a random sample from the k-nearest neighbor.
- Then define the new samples as original samples plus the difference between the nearest neighbor multiplied by a random number between 0 and 1 (new sample = original samples + difference × random (0–1)).
- Add the newly generated samples to the minority.

3.2.2. Feature Extraction

Feature extraction in image processing reduces data dimensions while obtaining the relevant information from the original data to improve the classification model accuracy and maximize the recognition rate. Feature extraction is performed by extracting relevant data, characterizing classes, and storing them in feature vectors to be inputted into the machine learning algorithm [42].

Transfer learning models are pre-trained models on a vast dataset image dataset and utilized as a feature extraction method, allowing the transfer of pre-gained knowledge. For a small dataset, training a model from scratch will result in low performance due to overfitting. Several pre-trained models based on convolutional neural network (CNN) architectures were developed to resolve this issue, such as VGG-16, VGG-19, DenseNet, and multilayer perceptron (MLP). These pre-trained models can be fine-tuned and used as feature extractors [42].

Moreover, Visual Geometry Group (VGG) models can extract features from images. Two VGG models were developed at Oxford based on two CNNs with 16 and 19 layers, widely known as VGG-16 and VGG-19. These CNN models accept input of 224 by 224 pixels in RGP format. The first layer consists of 64 neurons, and the number of neurons increases by a factor of 2, reaching 512 neurons at the last layers [43].

DenseNet is a CNN pre-trained model like the Visual Geometry Group (VGG) models. However, it requires fewer parameters to remove unnecessary feature maps due to feature reuse. In DenseNet, as shown in Figure 4, all layers are connected, not only adjacent layers, as in other CNN architectures. This allows features to be mapped to other layers without the need for replications, reducing the number of parameters. Moreover, connecting all layers resolves the vanishing-gradient problem, resulting in higher performance [44].

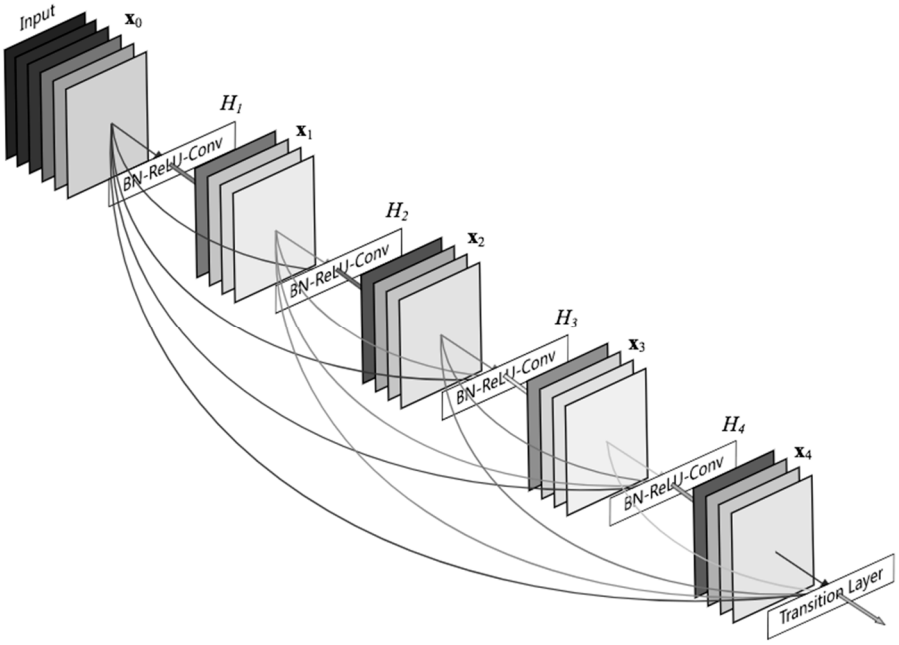

Figure 4. DenseNet Architecture.

3.2.3. Image Filter

Image filters are also widely used to map image features into feature space for input to the training model while ensuring that the features are sufficiently descriptive of the class. In this work, we use two filters: auto-color correlogram filter and fuzzy color and texture histogram (FcTH) Filter. The auto-color correlogram filter, unlike color histograms, which only describe the color distribution of an image, expresses how spatial correlation among colors varies with distance. However, the absence of spatial information might lead to false predictions. It is difficult for a histogram to distinguish the difference between both images since they have similar color contexts. However, a correlogram will distinguish the difference clearly due to the spatial information [45]. The fuzzy color and texture histogram (FcTH) filter aims to map the visual features of an image to feature space while ensuring that the features are sufficiently descriptive of the class. Like the auto-color correlogram filter, the FcTH filter uses and combines color and texture information of images. A fuzzy system produces a fuzzy linking histogram which forms several pins representing different image colors [46]. FcTH consists of three fuzzy units. The first fuzzy unit produces a hue saturation value (HSV) color space in 10 bins. The second fuzzy unit expands the 10 bins to 24 bins and then to 192 bins in the third unit. Then the 192-bin histogram is mapped into eight regions in the interval 0–7 using the Gustafson–Kessel fuzzy classifier [47].

4. Experimental Results

In this section, the developed models are presented, analyzed, and compared based on the following evaluation metrics: accuracy, precision, recall, and F1-score, given in Equations (1)–(4), respectively [48–52].

$$Accuracy = \frac{TruePositive + TrueNegative}{TruePositive + TrueNegative + FalsePostive + FalseNegative} \quad (1)$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} \quad (2)$$

$$Percision = \frac{TruePositive}{TruePositive + FalsePositive} \quad (3)$$

$$F - measure = \frac{2 * Precision * Recall}{Precision + Recall} \quad (4)$$

4.1. Auto-Color Correlogram Filters

Several models were tested using different filters, individual algorithms, and stacked models to obtain the most accurate results. The first four models were developed using the auto-correlogram filter. Three models were based on individual learning algorithms, namely KNN, SMO, and random forest, and a stacked model of both KNN and SMO was also developed, as shown in Figure 5.

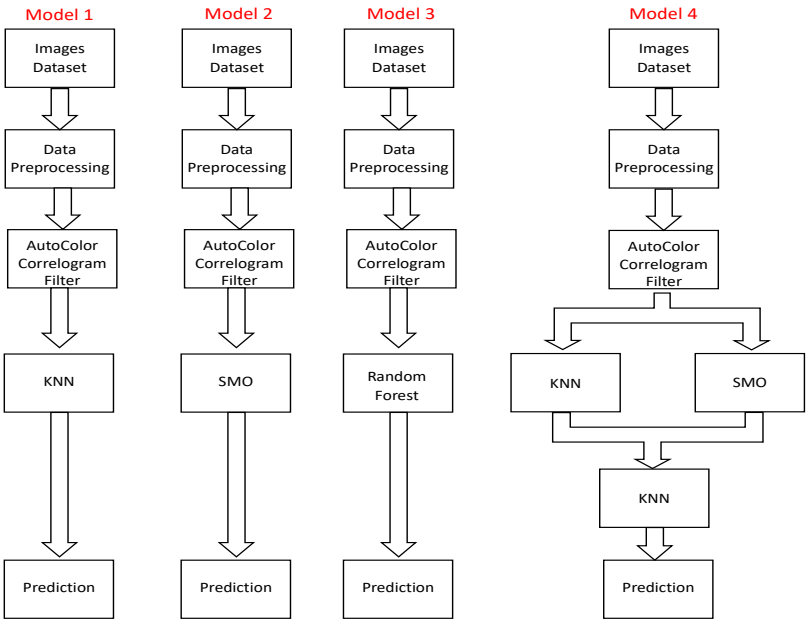

Figure 5. Four Models with Auto-Color Correlogram Filter.

Table 2 below shows these four models' accuracy, precision, recall, and F1-score. As shown in the table, KNN with k = 1 has the highest accuracy of 97.5% with a precision of 98.0% when data are split 90% for training and 10% for testing. When the training set is smaller, 70% to 30%, the random forest has the highest accuracy of 92.2% with a precision of 94.0%.

Table 2. Comparison of Four models with Auto-Color Correlogram Filter.

Data Split	Metric	KNN Neighbors (K)		Random Forest	SMO	Stacking (KNN + SMO), Meta (KNN) with Neighbors (K)	
Cross-validation 10	Accuracy	K = 1	93.0%	93.3%	84.8%	K = 1	91.5%
	Precision		91.4%	92.0%	76.6%		89.1%
	Recall		92.5%	92.7%	93.6%		91.7%
	F1-Score		0.92	0.92	0.84		0.90
70:30	Accuracy	K = 1	92.0%	92.2%	82.7%	K = 1	89.1%
	Precision		92.9%	94.0%	75.4%		94.2%
	Recall		88.8%	88.2%	91.3%		80.7%
	F1-Score		0.91	0.91	0.83		0.87
90:10	Accuracy	K = 1	97.5%	96.6%	84.9%	K = 1	95.0%
	Precision		98.0%	94.4%	75.0%		91.1%
	Recall		96.2%	98.1%	98.1%		98.1%
	F1-Score		0.97	0.96	0.85		0.94

4.2. Auto-Color Correlogram and FcTH Filters

An additional five models were developed using auto-correlogram and FcTH filters. Three models were based on individual learning algorithms, like the first four models mentioned above, in addition to two stacked models, as shown in Figure 6.

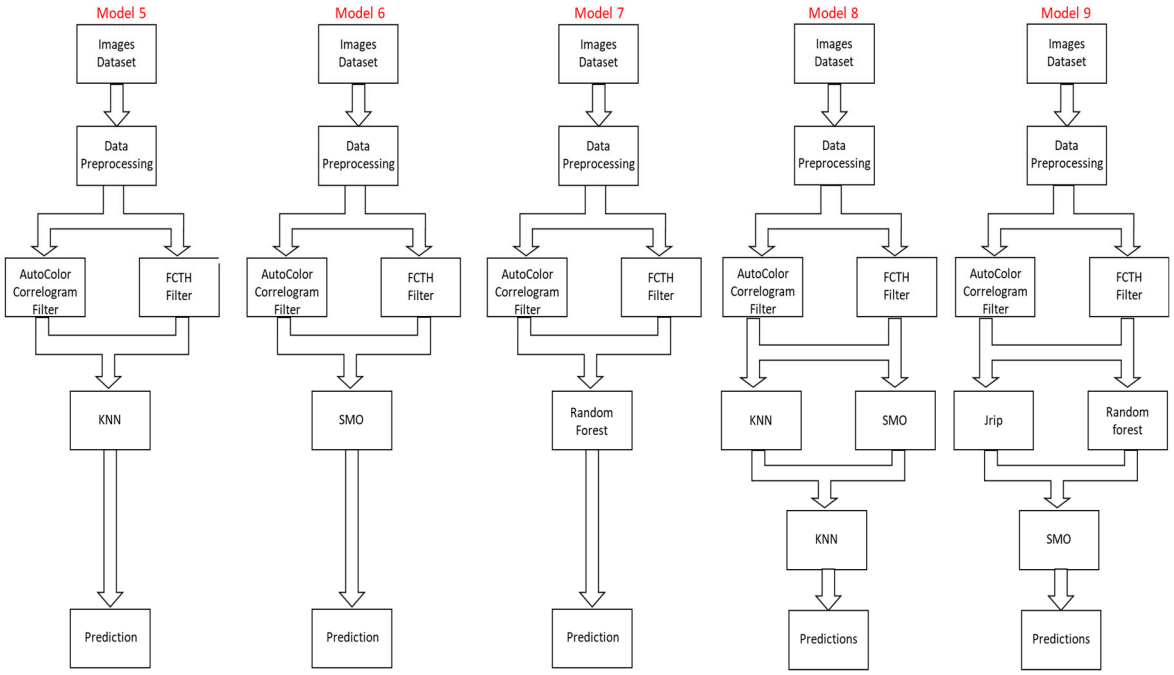

Figure 6. Five Models with Auto-Color Correlogram and FcTH Filters.

Table 3 presents the accuracy of these five models. As shown in the table, both random forest and the stacked model with Jrip and random forest as base-level classifiers and SMO as meta classifiers have the highest accuracy of 97.5% and precision of 96.2% when data are split 90% for training and 10% for testing. However, the RF model has a higher recall of 100% compared to the stacked model, with 98.1%. When the training set is smaller, 70% to 30%, the random forest has the highest accuracy of 95%.

Table 3. Comparison of Five models with Auto-Color Correlogram and FcTH Filters.

Data Split	Metric	KNN Neighbors (K)		Random Forest	SMO	Stacking (KNN + SMO), Meta (KNN) with Neighbors (K)		Stacking (Jrip, RF), Meta (SMO)
cross-validation 10	Accuracy	K = 1	94.6%	94.7%	92.0	K = 1	93.7%	95.0%
	Precision		94.0%	93.5%	86.9%		93.7%	93.9%
	Recall		93.6%	94.4%	96.1%		91.7%	94.6%
	F1-Score		0.94	0.94	0.91		0.93	0.90
70:30	Accuracy	K = 1	94.0%	95.0%	90.8%	K = 1	93.6%	94.4%
	Precision		92.6%	93.3%	85.2%		93.7%	94.3%
	Recall		93.4%	96.0%	96.3%		91.9%	93.2%
	F1-Score		0.93	0.95	0.90		0.93	0.94
90:10	Accuracy	K = 1	96.6%	97.5%	92.4%	K = 1	96.0%	97.5%
	Precision		96.2%	94.5%	87.7%		96.1%	96.2%
	Recall		96.2%	100	96.2%		94.2%	98.1%
	F1-Score		0.96	0.98	0.92		0.95	0.97

4.3. DenseNet Transfer Model

Four additional models were developed using the DenseNet transfer model. Three of these models were based on individual learning algorithms, like the first four models mentioned above, in addition to a stacked model, KNN and SMO, and KNN as the meta classifier, as shown in Figure 7.

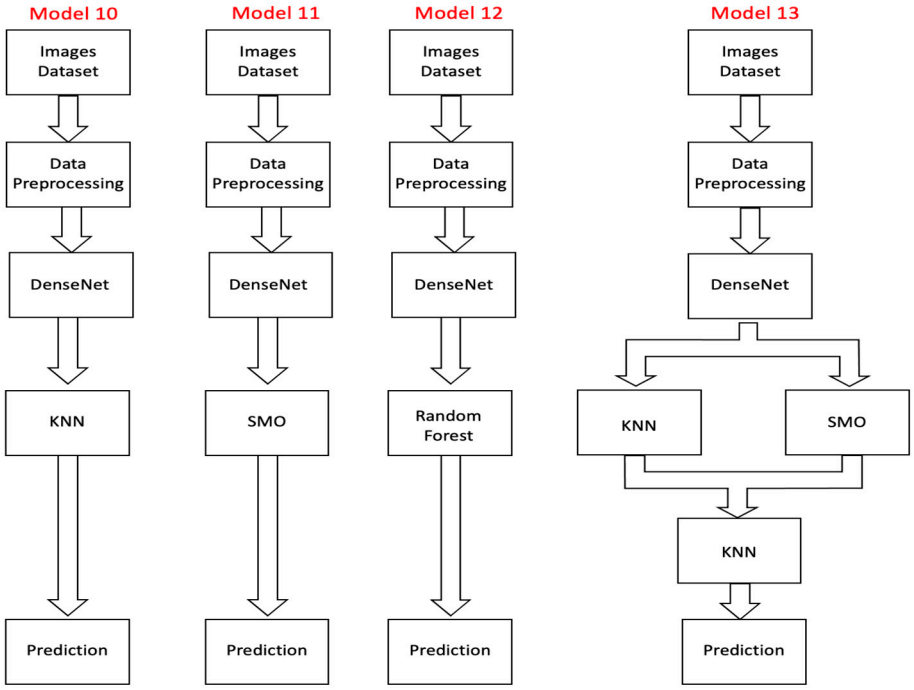

Figure 7. Four Models with DenseNet Transfer Model.

Table 4 below shows the accuracy of these four models. As shown in the table, the random forest model has the highest accuracy of 96.6% with a precision of 92.9% when data are split 90% for training and 10% for testing. When the training set is smaller, 70% to 30%, KNN, with a k value is 5, has the highest accuracy of 94.4% with a precision of 91.7%.

Table 4. Comparison of Four Models with DenseNet Transfer Model.

Data Split	Metric	KNN Neighbors (K)		Random Forest	SMO	Stacking (KNN + SMO), Meta (KNN) with Neighbors (K)	
cross-validation 10	Accuracy	K = 3	94.6%	94.1%	94.1%	K = 3	94.2%
	Precision		91.7%	91.0%	92.1%		92.1%
	Recall		96.1%	95.9%	94.4%		94.8%
	F1-Score		0.94	0.93	0.93		0.93
70:30	Accuracy	K = 5	94.4%	93.6%	93.9%	K = 5	93.9%
	Precision		91.7%	91.1%	93.7%		91.6%
	Recall		96.3%	95.0%	92.5%		95.0%
	F1-Score		0.94	0.93	0.93		0.933
90:10	Accuracy	K = 5	95.0%	96.6%	93.3%	K = 5	92.4%
	Precision		89.7%	92.9%	92.3%		89.1%
	Recall		100	100	92.3%		94.2%
	F1-Score		0.95	0.96	0.92		0.92

4.4. VGG16 Transfer Model

Then four additional models were developed using VGG-16 as the transfer model. These four models are like the DenseNet models regarding training algorithms, as shown in Figure 8.

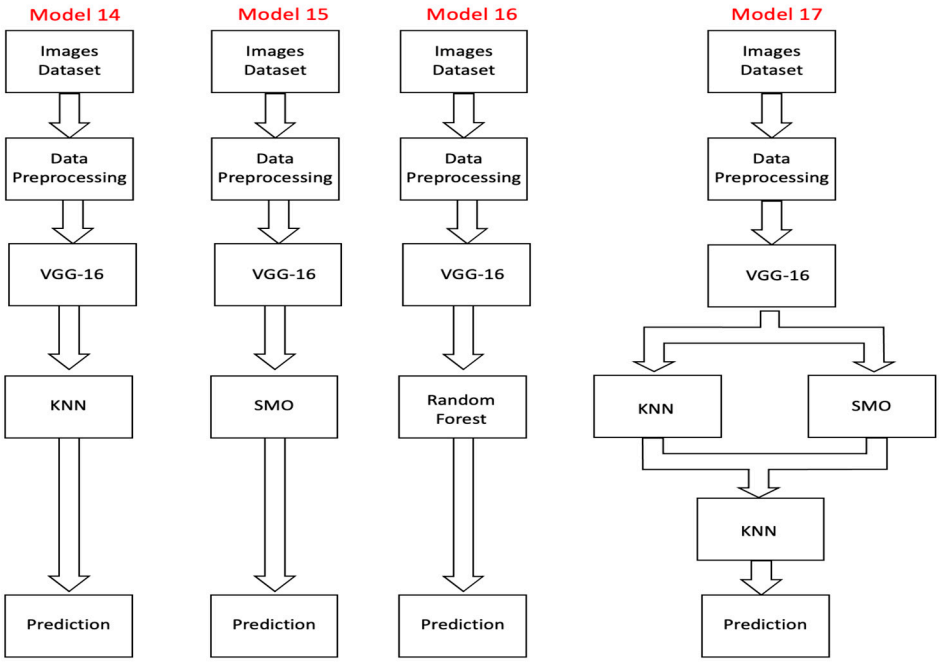

Figure 8. Four Models with VGG-16 Transfer Model.

Table 5 shows the accuracy of these four models. The table shows that the stacked model has the highest accuracy of 98.3% with a precision of 96.3% when data are split 90% for training and 10% for testing. When the training set is smaller, 70% to 30%, KNN, with a k value of 5, has the highest accuracy of 95.8% with a precision of 95.1%.

Table 5. Comparison of Four Models with VGG-16 Transfer Model.

Data Split	Metric	KNN Neighbors (K)		Random Forest	SMO	Stacking (KNN + SMO), Meta (KNN) with Neighbors (K)	
cross-validation 10	Accuracy	K = 3	94.2%	94.1%	93.1%	K = 3	93.7%
	Precision		91.3%	91.0%	90.8%		92.4%
	Recall		95.8%	95.8%	93.6%		93.2%
	F1-Score		0.94	0.93	0.92		0.93
70:30	Accuracy	K = 5	95.8%	94.4%	94.7%	K = 5	94.7%
	Precision		95.1%	92.7%	92.8%		93.3%
	Recall		95.7%	95.0%	95.7%		95.0%
	F1-Score		0.95	0.94	0.94		0.94
90:10	Accuracy	K = 5	97.5%	97.5%	94.7%	K = 5	98.3%
	Precision		94.5%	94.5%	94.5%		96.3%
	Recall		100%	100%	100%		100%
	F1-Score		97.2%	0.97	0.97		0.98

4.5. Comparison of Different Feature Extractors

This section will evaluate the proposed models based on the feature extractors. For cross-validation of 10, auto-color correlogram and FcTH filters resulted in the highest accuracy of 94.7% when combined with random forest, with a precision of 93.5%, as shown in Table 6.

Table 6. Comparison of Different Feature Extractors for Cross-Validation of 10.

Data Split	Metric	KNN Neighbors (K)		Random Forest	SMO	Stacking (KNN + SMO), Meta (KNN) with Neighbors (K)		Stacking (Jrip, RF), Meta (SMO)
Auto-Color Correlogram Filter	Accuracy	K = 1	93.0%	93.3%	84.8%	K = 1	91.5%	Not appliable (NA)
	Precision		91.4%	92.0%	76.6%		89.1%	NA
	Recall		92.5%	92.7%	93.6%		91.7%	NA
	F1-Score		0.92	0.92	0.84		0.90	NA
Auto-Color and FcTH	Accuracy	K = 1	94.6%	94.7%	92.0	K = 1	93.7%	95.0%
	Precision		94.0%	93.5%	86.9%		93.7%	93.9%
	Recall		93.6%	94.4%	96.1%		91.7%	94.6%
	F1-Score		0.94	0.94	0.91		0.93	0.90
DenseNet	Accuracy	K = 3	94.6%	94.1%	94.1%	K = 3	94.2%	NA
	Precision		91.7%	91.0%	92.1%		92.1%	NA
	Recall		96.1%	95.9%	94.4%		94.8%	NA
	F1-Score		0.94	0.93	0.93		0.93	NA
VGG-16	Accuracy	K = 3	94.2%	94.1%	93.1%	K = 5	93.7%	NA
	Precision		91.3%	91.0%	90.8%		92.4%	NA
	Recall		95.8%	95.8%	93.6%		93.2%	NA
	F1-Score		0.94	0.93	0.92		0.93	NA

For a 70% to 30% split, VGG-16 has resulted in the highest accuracy, when combined with KNN, of 95.8% with a precision of 95.1%, as shown in Table 7. VGG-16 has also resulted in the highest accuracy for a 90% to 10% data split. VGG-16 has resulted in an accuracy of 98.3% and precision of 96.3% when combined with the stacked model, as shown in Table 8.

Table 7. Comparison of Different Feature Extractors for 70–30% Data Split.

Data Split	Metric	KNN Neighbors (K)		Random Forest	SMO	Stacking (KNN + SMO), Meta (KNN) with Neighbors		Stacking (Jrip, RF), Meta (SMO)
Auto-Color Correlogram Filter	Accuracy	K = 1	92.0%	92.2%	82.7%	K = 1	89.1%	NA
	Precision		92.9%	94.0%	75.4%		94.2%	NA
	Recall		88.8%	88.2%	91.3%		80.7%	NA
	F1-Score		0.91	0.91	0.83		0.87	NA
Auto-Color and FcTH	Accuracy	K = 1	94.0%	95.0%	90.8%	K = 1	93.6%	94.4%
	Precision		92.6%	93.3%	85.2%		93.7%	94.3%
	Recall		93.4%	96.0%	96.3%		91.9%	93.2%
	F1-Score		0.93	0.95	0.90		0.93	0.94
DenseNet	Accuracy	K = 3	94.4%	93.6%	93.9%	K = 3	93.9%	NA
	Precision		91.7%	91.1%	93.7%		91.6%	NA
	Recall		96.3%	95.0%	92.5%		95.0%	NA
	F1-Score		0.94	0.93	0.93		0.93	NA
VGG-16	Accuracy	K = 3	95.8%	94.4%	94.7%	K = 5	94.7%	NA
	Precision		95.1%	92.7%	92.8%		93.3%	NA
	Recall		95.7%	95.0%	95.7%		95.0%	NA
	F1-Score		0.95	0.94	0.94		0.94	NA

Table 8. Comparison of Different Feature Extractors for 90–10% Data Split.

Data Split	Metric	KNN Neighbors (K)		Random Forest	SMO	Stacking (KNN + SMO), Meta (KNN) with Neighbors (K)		Stacking (Jrip, RF), Meta (SMO)
Auto-Color Correlogram Filter	Accuracy	K = 1	97.5%	96.6%	84.9%	K = 1	95.0%	NA
	Precision		98.0%	94.4%	75.0%		91.1%	NA
	Recall		96.2%	98.1%	98.1%		98.1%	NA
	F1-Score		0.97	0.96	0.85		0.94	NA
Auto-Color and FcTH	Accuracy	K = 1	96.6%	97.5%	92.4%	K = 1	96.0%	97.5%
	Precision		96.2%	94.5%	87.7%		96.1%	96.2%
	Recall		96.2%	100	96.2%		94.2%	98.1%
	F1-Score		0.96	0.98	0.92		0.95	0.97
DenseNet	Accuracy	K = 3	95.0%	96.6%	93.3%	K = 3	92.4%	NA
	Precision		89.7%	92.9%	92.3%		89.1%	NA
	Recall		100%	100%	92.3%		94.2%	NA
	F1-Score		0.95	0.96	0.92		0.92	NA
VGG-16	Accuracy	K = 3	97.5%	97.5%	94.7%	K = 5	98.3%	NA
	Precision		94.5%	94.5%	94.5%		96.3%	NA
	Recall		100%	100%	100%		100	NA
	F1-Score		0.97	0.97	0.97		0.98	NA

4.6. Analysis of the Results

In this work, seventeen models have been evaluated with different feature extractors and different classification algorithms. The models were evaluated based on accuracy, precision, recall, and F1-score., with more emphasis on accuracy and precision. The objective is to have the highest accuracy with the highest precision. The objective of having the highest precision is to ensure that minimum malicious traffic is wrongly classified as normal jeopardizing network security.

The highest precision and accuracy are achieved when VGG-16 combines the stacked model, KNN and SMO, and KNN as the meta classifier with k = 3, for 90% to 10% data split. Therefore, this model was selected as the best model.

5. Conclusions

The Internet of Things is a new technological paradigm that aims to improve daily life by automating normal daily operations in all aspects of life without human intervention. With the continuous increase in Internet of Things (IoT) device use, more interest is shown in internet security, specifically focusing on protecting these vulnerable devices from malicious traffic. Such threats are difficult to distinguish, so advanced detection systems are becoming necessary.

This study aimed to develop a model with the highest performance in distinguishing malicious from normal traffic. Various feature extraction techniques and machine learning algorithms were used to achieve the study's objectives. The experiments show that feature extraction techniques are important for attaining high performance. Moreover, VGG-16 transfer proved to give the highest accuracy and precision. This study investigated the effect of individual and stacked machine learning algorithms. It also investigated the impact of the data split ratio on the execution of the models. The conducted experiments showed that the stacked model achieved the highest accuracy when combined with the VGG-16 transfer model, achieving an accuracy of 98.3%.

Author Contributions: Conceptualization, D.M.; data curation, M.A.; formal analysis, A.R.; funding acquisition, A.R.; investigation, D.M. and F.A.; methodology, D.M. and M.A.; project administration, R.M.M.; resources, A.R.; software, M.A.; supervision, D.M.; validation, M.A., F.A. and R.M.M.; writing—original draft, M.A.; writing—review and editing, F.A., A.R. and R.M.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the SAUDI ARAMCO Cybersecurity Chair at the College of Computer Science and Information Technology (CCSIT), Imam Abdulrahman Bin Faisal University (IAU), Dammam, Kingdom of Saudi Arabia.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Dramé-Maigné, S.; Laurent, M.; Castillo, L.; Ganem, H. Centralized, Distributed, and Everything in between: Reviewing Access Control Solutions for the IoT. *ACM Comput. Surv.* **2022**, *54*, 1–34. [CrossRef]
2. Granjal, J.; Monteiro, E.; Silva, J.S. Security for the internet of things: A Survey of existing protocols and open research issues. *J. Fac. Eng. Archit. Gazi Univ.* **2018**, *33*, 1247–1272. [CrossRef]
3. Drame-Maigne, S. Blockchain and Access Control: Towards a More Secure Internet of Things. Networking and Internet Architecture [cs.NI]. Ph.D. Thesis, Université Paris Saclay (COmUE), Yvette, France, 2019.
4. Gibson, A.; Thamilarasu, G. Protect Your Pacemaker: Blockchain based Authentication and Consented Authorization for Implanted Medical Devices. *Procedia Comput. Sci.* **2019**, *171*, 847–856. [CrossRef]
5. ICANN. The General Data Protection Regulation 2020 Review. May 2020. Available online: https://itp.cdn.icann.org/en/files/government-engagement-ge/ge-003-07may20-en.pdf (accessed on 5 March 2023).
6. Antonakakis, M.; April, T.; Bailey, M.; Bernhard, M.; Bursztein, E.; Cochran, J.; Zhou, Y. Understanding the Mirai Botnet. In Proceedings of the 26th USENIX Security Symposium, Vancouver, BC, Canada, 16–18 August 2017.
7. O'Sullivan, W.; Choo, K.-K.R.; Le-Khac, N.-A. *Defending IoT Devices from Malware*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 5–29. [CrossRef]
8. Wu, H.; Han, H.; Wang, X.; Sun, S. Research on Artificial Intelligence Enhancing Internet of Things Security: A Survey. *IEEE Access* **2020**, *8*, 153826–153848. [CrossRef]
9. Ferrag, M.A.; Maglaras, L. DeepCoin: A Novel Deep Learning and Blockchain-Based Energy Exchange Framework for Smart Grids. *IEEE Trans. Eng. Manag.* **2020**, *67*, 1285–1297. [CrossRef]
10. Alkadi, O.; Moustafa, N.; Turnbull, B.; Choo, K.K.R. A Deep Blockchain Framework-Enabled Collaborative Intrusion Detection for Protecting IoT and Cloud Networks. *IEEE Internet Things J.* **2021**, *8*, 9463–9472. [CrossRef]
11. HaddadPajouh, H.; Dehghantanha, A.; Khayami, R. A Deep Recurrent Neural Network Based Approach for Internet of Things Malware Threat Hunting. *Future Gener. Comput. Syst.* **2018**, *85*, 88–96. [CrossRef]
12. Belavagi, M.C.; Muniyal, B. Performance evaluation of supervised machine learning algorithms for intrusion detection. *Procedia Comput. Sci.* **2016**, *89*, 117–123. [CrossRef]
13. Ahmad, I.; Haq, Q.E.U.; Imran, M.; Alassafi, M.O.; AlGhamdi, R.A. An Efficient Network Intrusion Detection and Classification System. *Mathematics* **2022**, *10*, 530. [CrossRef]
14. Thapa, N.; Liu, Z.; Kc, D.B.; Gokaraju, B.; Roy, K. Comparison of machine learning and deep learning models for network intrusion detection systems. *Future Internet* **2020**, *12*, 167. [CrossRef]
15. Das, V.; Pathak, V.; Sharma, S.; Srikanth, M.; Kumar, T.G. Network intrusion detection system based on machine learning algorithms. *AIRCC's Int. J. Comput. Sci. Inf. Technol.* **2010**, *2*, 138–151. [CrossRef]
16. Rahman, A.; Mahmud, M.; Iqbal, T.; Saraireh, L.; Kholidy, H.; Gollapalli, M.; Musleh, D.; Alhaidari, F.; Almoqbil, D.; Ahmed, M.I.B. Network anomaly detection in 5G networks. *Math. Model. Eng. Probl.* **2022**, *9*, 397–404. [CrossRef]
17. Rose, J.R.; Swann, M.; Bendiab, G.; Shiaeles, S.; Kolokotronis, N. Intrusion Detection using Network Traffic Profiling and Machine Learning for IoT. In Proceedings of the 2021 IEEE Conference on Network Softwarization: Accelerating Network Softwarization in the Cognitive Age, NetSoft 2021, Tokyo, Japan, 28 June–2 July 2021; pp. 409–415. [CrossRef]
18. Ali, Z.; Hussain, F.; Ghazanfar, S.; Husnain, M.; Zahid, S.; Shah, G.A. A Generic Machine Learning Approach for IoT Device Identification. In Proceedings of the 2021 International Conference on Cyber Warfare and Security (ICCWS), Islamabad, Pakistan, 23–25 November 2021; pp. 118–123. [CrossRef]
19. El-Sayed, R.; El-Ghamry, A.; Gaber, T.; Hassanien, A.E. Zero-Day Malware Classification Using Deep Features with Support Vector Machines. In Proceedings of the 2021 Tenth International Conference on Intelligent Computing and Information Systems (ICICIS), Cairo, Egypt, 5–7 December 2021; pp. 311–317. [CrossRef]
20. Le, K.-H.; Nguyen, M.-H.; Tran, T.-D.; Tran, N.-D. IMIDS: An Intelligent Intrusion Detection System against Cyber Threats in IoT. *Electronics* **2022**, *11*, 524. [CrossRef]
21. Joo, H.; Choi, H.; Yun, C.; Cheon, M. Efficient Network Traffic Classification and Visualizing Abnormal Part Via Hybrid Deep Learning Approach: Xception + Bidirectional GRU. *Glob. J. Comput. Sci. Technol.* **2022**, *21*, 1–10. [CrossRef]
22. Bendiab, G.; Shiaeles, S.; Alruban, A.; Kolokotronis, N. IoT malware network traffic classification using visual representation and deep learning. In Proceedings of the 2020 IEEE Conference on Network Softwarization: Bridging the Gap Between AI and Network Softwarization, NetSoft 2020, Virtual, 29 June–3 July 2020; pp. 444–449. [CrossRef]
23. Hindy, H.; Bayne, E.; Bures, M.; Atkinson, R.; Tachtatzis, C.; Bellekens, X. Machine Learning Based IoT Intrusion Detection System: An MQTT Case Study (MQTT-IoT-IDS2020 Dataset). *Lect. Notes Netw. Syst.* **2021**, *180*, 73–84. [CrossRef]

24. Sapre, S.; Ahmadi, P.; Islam, K. A Robust Comparison of the KDDCup99 and NSL-KDD IoT Network Intrusion Detection Datasets Through Various Machine Learning Algorithms. *arXiv* **2019**, arXiv:1912.13204.
25. Liu, J.; Kantarci, B.; Adams, C. Machine learning-driven intrusion detection for Contiki-NG-based IoT networks exposed to NSL-KDD dataset. In Proceedings of the WiseML 2020—2nd ACM Workshop on Wireless Security and Machine Learning, Abu Dhabi, United Arab Emirates, 28 June–2 July 2020; pp. 25–30. [CrossRef]
26. Amouri, A.; Alaparthy, V.T.; Morgera, S.D. A machine learning based intrusion detection system for mobile internet of things. *Sensors* **2020**, *20*, 461. [CrossRef] [PubMed]
27. Fenanir, S.; Semchedine, F.; Baadache, A. A machine learning-based lightweight intrusion detection system for the internet of things. *Rev. D'Intell. Artif.* **2019**, *33*, 203–211. [CrossRef]
28. Islam, N.; Farhin, F.; Sultana, I.; Kaiser, M.S.; Rahman, M.S.; Mahmud, M.; Cho, G.H. Towards Machine Learning Based Intrusion Detection in IoT Networks. *Comput. Mater. Contin.* **2021**, *69*, 1801–1821. [CrossRef]
29. Ahmad, M.; Riaz, Q.; Zeeshan, M.; Tahir, H.; Haider, S.A.; Khan, M.S. Intrusion detection in internet of things using supervised machine learning based on application and transport layer features using UNSW-NB15 data-set. *Eurasip J. Wirel. Commun. Netw.* **2021**, *2021*, 1–23. [CrossRef]
30. Saba, T.; Sadad, T.; Rehman, A.; Mehmood, Z.; Javaid, Q. Intrusion detection system through advance machine learning for the internet of things networks. *IT Prof.* **2021**, *23*, 58–64. [CrossRef]
31. Smys, S.; Basar, A.; Wang, H. Hybrid Intrusion Detection System for Internet of Things (IoT). *J. ISMAC* **2020**, *2*, 190–199. [CrossRef]
32. Papafotikas, S.; Kakarountas, A. A machine-learning clustering approach for intrusion detection to IoT devices. In Proceedings of the 2019 4th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference, SEEDA-CECNSM 2019, Piraeus, Greece, 20–22 September 2019; pp. 1–6. [CrossRef]
33. Farooq, M.S.; Abbas, S.; Rahman, A.U.; Sultan, K.; Khan, M.A.; Mosavi, A. A fused machine learning approach for intrusion detection system. *Comput. Mater. Contin.* **2023**, *74*, 2607–2623.
34. Liu, H.; Lang, B. Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey. *Appl. Sci.* **2019**, *9*, 4396. [CrossRef]
35. Verma, A.; Ranga, V. Machine Learning Based Intrusion Detection Systems for IoT Applications. *Wirel. Pers. Commun.* **2020**, *111*, 2287–2310. [CrossRef]
36. Kocher, G.; Kumar, G. Machine learning and deep learning methods for intrusion detection systems: Recent developments and challenges. *Soft Comput.* **2021**, *25*, 9731–9763. [CrossRef]
37. Aversano, L.; Bernardi, M.L.; Cimitile, M.; Pecori, R. A systematic review on Deep Learning approaches for IoT security. *Comput. Sci. Rev.* **2021**, *40*, 100389. [CrossRef]
38. Alhaidari, F.; Rahman, A.; Zagrouba, R. Cloud of Things: Architecture, applications and challenges. *J. Ambient. Intell. Human Comput.* 2020. [CrossRef]
39. Rose, J. 913 Malicious Network Traffic PCAPs and Binary Visualisation Images Dataset, IEEE Dataport. 2021. Available online: https://ieee-dataport.org/open-access/913-malicious-network-traffic-pcaps-and-binary-visualisation-images-dataset (accessed on 7 May 2022).
40. Obaid, H.S.; Dheyab, S.A.; Sabry, S.S. The impact of data pre-processing techniques and dimensionality reduction on the accuracy of machine learning. In Proceedings of the 2019 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON), Jaipur, India, 13–15 March 2019; pp. 279–283.
41. Hussein, A.S.; Li, T.; Yohannese, C.W.; Bashir, K. A-SMOTE: A new preprocessing approach for highly imbalanced datasets by improving SMOTE. *Int. J. Comput. Intell. Syst.* **2019**, *12*, 1412. [CrossRef]
42. Li, J.; Lo, W.L.; Fu, H.; Chung, H.S.H. A transfer learning method for meteorological visibility estimation based on feature fusion method. *Appl. Sci.* **2021**, *11*, 997. [CrossRef]
43. Desai, C. Image Classification Using Transfer Learning and Deep Learning. *Int. J. Eng. Comput. Sci.* **2021**, *10*, 25394–25398. [CrossRef]
44. Chauhan, T.; Palivela, H.; Tiwari, S. Optimization and Fine-Tuning of DenseNet model for classification of Covid-19 cases in Medical Imaging. *Int. J. Inf. Manag. Data Insights* **2021**, *1*, 100020. [CrossRef]
45. Huang, J.; Kumar, S.R.; Mitra, M.; Zhu, W.-J.; Zabih, R. Image indexing using color correlograms. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico, 17–19 June 1997; pp. 762–768.
46. Chatzichristofis, S.A.; Boutalis, Y.S. FCTH: Fuzzy color and texture histogram-a low level feature for accurate image retrieval. In Proceedings of the 2008 Ninth International Workshop on Image Analysis for Multimedia Interactive Services, Klagenfurt, Austria, 7–9 May 2008; pp. 191–196.
47. Jankovic, R. Classifying cultural heritage images by using decision tree classifiers in WEKA. In Proceedings of the 1st International Workshop on Visual Pattern Extraction and Recognition for Cultural Heritage Understanding Co-Located with 15th Italian Research Conference on Digital Libraries (IRCDL 2019), Pisa, Italy, 30 January 2019; pp. 119–127.
48. Alhaidari, F.; Abu-Shaib, N.; Alsafi, M.; Alharbi, H.; Alawami, M.; Aljindan, R.; Rahman, A.-U.; Zagrouba, R. ZeVigilante: Detecting Zero-Day Malware Using Machine Learning and Sandboxing Analysis Techniques. *Comput. Intell. Neurosci.* **2022**, *2022*, 1615528. [CrossRef] [PubMed]
49. Alqarni, A.; Rahman, A. Arabic Tweets-Based Sentiment Analysis to Investigate the Impact of COVID-19 in KSA: A Deep Learning Approach. *Big Data Cogn. Comput.* **2023**, *7*, 16. [CrossRef]

50. Alotaibi, A.; Rahman, A.; Alhaza, R.; Alkhalifa, W.; Alhajjaj, N.; Alharthi, A.; Abushoumi, D.; Alqahtani, M.; Alkhulaifi, D. Spam and sentiment detection in Arabic tweets using MARBERT model. *Math. Model. Eng. Probl.* **2022**, *9*, 1574–1582. [CrossRef]
51. Basheer Ahmed, M.I.; Zaghdoud, R.; Ahmed, M.S.; Sendi, R.; Alsharif, S.; Alabdulkarim, J.; Albin Saad, B.A.; Alsabt, R.; Rahman, A.; Krishnasamy, G. A Real-Time Computer Vision Based Approach to Detection and Classification of Traffic Incidents. *Big Data Cogn. Comput.* **2023**, *7*, 22. [CrossRef]
52. Alghamdi, A.S.; Rahman, A. Data Mining Approach to Predict Success of Secondary School Students: A Saudi Arabian Case Study. *Educ. Sci.* **2023**, *13*, 293. [CrossRef]