JCBI
Journal of Computing & Biomedical Informatics
ISSN: 2710 - 1606
Volume 07 Issue 01
2024

Research Article
https://doi.org/10.56979/701/2024

Innovative Machine Learning Techniques for Malware Detection

Aqsa Ijaz[1], Ammar Ahmad Khan[2*], Muhammad Arslan[3], Ashir Tanzil[4], Alina Javed[5], Muhammad Asad Ullah Khalid[3], and Shouzab Khan[6]

[1]Department of Computer Science, University of Lahore, Sargodha, 40100, Pakistan.
[2]Department of Computer Science, NAMAL University, Mianwali, 42250, Punjab, Pakistan.
[3]Faculty of Computer Science, Lahore Garrison University, Lahore, 54000, Pakistan.
[4]Department of Computer Science, Abasyn University Islamabad Campus, Islamabad, 44000, Pakistan.
[5]Department of Computer and Software Engineering, Gomal University, Dera Ismail Khan, 29220, Khyber Pakhtunkhwa, Pakistan.
[6]Department of Computer Science, University of Alabama at Birmingham, Birmingham AL 35294, USA.
[*]Corresponding Author: Ammar Ahmad Khan. Email: ammar.ahmad@namal.edu.pk

Received: March 09, 2024 **Accepted:** May 19, 2024 **Published:** June 01, 2024

Abstract: Malware hazards are becoming more perplexing with time, new types of malware are entering cyberspace and triggering millions of devices day by day. People could not restrain in this century to refrain from not using smart devices, and adopting technology, as this world is shifting into a smart world, and due to the COVID19 wave, more numbers of devices and systems were being adopted by the people. In viewing the need of the society and to save the cyber world we have to step into this war against cybercrimes and play our role to save this world by making such models that are efficient and effective against malware. Therefore, accordingly, machine learning techniques have become the main point for cybersecurity as they are most suitable for handling modern malware attacks. Moreover, machine algorithms can generalize and distinguish cyber threats to a great extent. We applied an ensemble model in which we have used different machine learning algorithms such as KNN, SVM, and LR, as first stage classifiers and voting classifiers as meta-learner classifiers to identify the complex and modern malware. We have applied hard voting in our ensemble model. We also discuss and evaluate the performance of every algorithm applied in the model. KNN shows the best results overall. The ensemble model provides us the best result than any individual used model. The output of testing proves that our proposed method is highly efficient and adaptive and gives better results than many other present techniques. We gain 99.7 % accuracy with F-score 99%. The running time of the model is also less. So this proposed detecting malware model could be easily implemented in smart IoT devices as well.

Keywords: Malware Detection; Ensemble Model; Ensemble Learning; Machine Learning.

1. Introduction

Network traffic has been emerged due to the tremendous hype of IoT devices, by this various countermeasures arose such as privacy and security of users. [1]. Sensors and electronic devices communicate with the help of the internet to facilitate our living standards, and this rising technology is called the Internet of Things (IoT) [2]. In this advanced era of technology, the internet of things (IoT) technology is rapidly increasing [3], with the passage of time every second, devices are becoming a part of this growing technology. Although by adapting this technology our life becomes more at ease and we don't have to focus on minor tasks. All fields are adapting this technology rapidly, whether it's a person's wearable watches, cars, homes, industries, etc. This technology is now becoming a part of our daily routine. Observing the past analytics of adapting IoT technology, it assumes that the total number of connected devices with IoT will go high in the near future [4]. The global number IoT devices is estimated to nearly double, growing from 15.9 billion in 2023 to over 32.1 billion by 2030 [84]

As more devices are becoming part of IoT technology more traffic should be generated and due to this our security and privacy delimiters should become vulnerable to the hacker world [1]. In the year of 2013, almost one billion user accounts had been hacked by hackers [5]. One hundred and forty five million accounts of eBay had been under attack by the hackers in 2014 and this is not the end but the beginning of the attacks it continues to increase as the years passes in 2017 the personal pieces of information of one hundred and forty-three million customers of Equifax has been leaked. the same year a toy manufacturing industry whose worth is five billion had been attacked, [6] compromised eight hundred and twenty thousand customers' accounts and more than two million voice recordings had been leaked from some of them they also demand Ransom.

As this is the era of technology everything whether it is about education shopping, banking, communication, seat reservation, entertainment, all is now possible with technology. But somewhere in the heart of users, there is a little bit of hesitation to adopt this new technology, because of weak privacy constraints and attacks that are taking place recently. [7], [8]. Security threats to breach data are the most concern of security experts, data is everything whether it's a control given by a remote to a car or a projector screen or a simple discussion between two parties [9][10], [11]. Data privacy attack leaks your personal information [12]. It classifies into two types active and passive privacy attacks ((ADPA) (PDPA)) [13],[14].

Many researchers have entered this field and gave various cyber security systems and proposed different solutions on how we could secure our private data from unauthorized users and malicious attacks, trying to access the network. IoT cyber security becomes a main worry for the IoT World [3]. In 2017 Distributed denial of service attack (DDoS) cripples the infrastructure of IoT [3], [4]. On October 21st, 2016 the biggest DDOS attack Mirai launched, which cost huge destruction. The impact is devastating as it disabled some of the major services like Twitter, Amazon, and Netflix. It targets the Domain name service (DNS) servers. Attackers infected various IoT devices such as DVR recorders and IP cameras using this malware. When the source code of the attack leaked in 2017, hackers altered it to make various other IoT malware [15].

The current situation of cybercrime activities is full of different events where somewhere we saw breaches in security flaws, massive data breaches, leaks of personal information, and demanding ransomware. Day after day we go through many news about cyber-attacks even the well-known celebrities and political leaders are not safe from these attacks. Students' computers are lockdown by hackers and they demand payment to unlock the system. Hospitals are even not safe from cyber-attacks [16]. There are flaws in billions of microchips. There are many challenges for privacy and security in IoT devices as the trend of using IoT devices is growing. One of the biggest causes for the drastic increase in using IoT devices is due to covid 19 [17], [18].

Kaspersky Lab report 0f 2020 depicts that in 2016 malware samples for IoT is 3216 and in 2018 it increased to 121588 [88]. So as a result privacy and security measures in this complex IoT environment have become a major challenge that has to be carefully monitored.

As IoT gadgets are becoming a part of our daily life and are used frequently, malicious attacks called malware attacks are highly taking place such as Trojan horses, adware, spyware, rootkits, worms, viruses, botnets, ransomware [19].

People are embracing technology eagerly with their open arms. According to the International Telecommunication Union (ITU) report, In 2023, 5.4 billion (67% worlds population) user are using the internet [90]

In our research work, we detect malware in devices through ML techniques. Malware detection is a method to detect the occurrence of the malware in the device or to figure out whether the script so-called program was malignant or benign (contains a virus or not) so the file could be removed and protection holds [20].

Machin learning and artificial intelligence (AI) methods are very efficient techniques and are adopted broadly by the analyst to spot the IoT network cyber-attacks [21], [22]. As we know malware could damage a machine to a great extent. It could cause the appliance to slow its performance or crash down. Due to malware the disk space could be restricted, increased the usage of the internet, extensions in the browsers and pop-up ads, etc. [23], [24], [19]. ML techniques gave accurate results so it gains the eye of various fields, as the output is reliable and accurate [3].

People keep using their devices without the acknowledgment that someone is peeking at or stealing their important data files, passwords, credit card information, and gaining access to their system. So it's a major concern that the first detection technique should be improved and then moves to prevention concerns. If we are unable to detect the malware, how we could prevent it. Many different models exist to detect the malware but with time malware affecting methods are being improved so we should keep on improving and researching the detection techniques to protect the cyber world.

Many system models are in the run to come into existence to detect the malware so that the cyber world will become more secure and protective from the activities that harm and sabotage the system. The malware identification detects the harmful and malicious code so that further actions could be taken so that the system which contains malware got into consideration [19].

We worked on the technique that detects malware that is now more complex in nature to be detected and you could say modernized, we could tackle malware before it further harms. We used a supervised ensemble learner to detect malware. In supervised learning, data is passed through with labels. The foremost goal of the supervised learning model is to calculate the result when new data is given. By supervised learning, we could perceive whether we are gaining the correct outcome or not, or in other words, we could say we will get direct feedback.[25]–[27]

So by proposing our ensemble model, we contribute our part to making the cyber world especially IoT world secure and trustworthy, enhancing knowledge, and building a path so researchers take further steps to improve security parameters.

We proposed classification and statistical approaches to detect malware through advanced techniques. Therefore, the main steps of our analysis are as follows:

- Reduce the dimensionality and extract the desired features from the collected data using PCA.
- Classify the data using supervised ensemble model.

Our research objective is to grow a model that can distinguish the latest malware and protect the cyber world, especially IoT, which is our future. We aim to build a model that is both effective and efficient. Our task is to detect the malware accurately, as possible, and save the systems from the serve disaster. So our proposed model will be applied to all the systems which are vulnerable to the hacker world. Especially Education departments, Hospital departments, and IoT devices.

The rest of the research paper is planned as follows: Section 2 analyses the previous work done, associated to our research. Section 3 is about the methodology, we described that how we design our model and perform our research. Section 4 includes our results, we analyze our work and its outcome and also gave comments on it. The last section 5 in which we wrote the conclusion and future work.

2. Previous Malware Detection Techniques and Their Limitations

Traditionally, to detect malware, signature-based methods were used but the problem was applicability and scalability become limited by this method [28]. Another kind of technique used is static code analysis, but the problem was code obfuscation and before analysis, we should have to unpack and decrypt the files [29]. So dynamic code was proposed in which we do not need to unzip or decrypt the exe.file in the virtual environment, which is resource consuming and time-intensive [30]. Although these two methods did not identify specific kinds of malware that behave well camouflaged and which do not satisfy trigger conditions [31].

In 1980 Anderson presented the primary interruption identification framework (IDS) (Mukherjee and Sharma, 2012). In 1987 Denning (Denning, 1987) examined a model for interruption location in view of ongoing recognition. Their proposed models can recognize infiltrations, break-ins, diversions, and other PC based interruptions that hurt the PC framework. Their proposed new model depended on the speculation that implies any dubious way of behaving could be identified by examining review records, and likewise conceivable by observing client conduct we could likewise identify strange assaults in an organization [3].

[66] Present a framework called Haze Registering based Security (Concentration) that sees and recognizes malware digital assaults and safeguards IoT gadgets from digital assaults. The Spotlight framework depended on a virtual confidential organization (VPN) so that secured and solid correspondence can happen between IoT gadgets (Tian, Luo, Qiu, Du, and Guizani, 2020). The VPN server is safeguarded by the authentication reaction and protect the IoT gadgets from DDoS assaults. They

likewise executed their proposed framework in mist figuring and acquired compelling result. Their framework can identify assaults with less data transfer capacity and inside a brief time frame [3]. Nonetheless, AI and Man-made consciousness (man-made intelligence) procedures are more solid and powerful strategies and are utilized broadly in the IoT climate, [21], [22], [32]. ML techniques have become very popular and adopted by almost every field due to their valid results [3].

Now many researchers put their potential into advanced ML methods and by using these advanced techniques of ML they have taken out more information from the malware datasets [31].

Table 1. Existing solutions to threats, using Machine Learning Algorithms

Reference	Threat	Algorithm	Dataset	Accuracy
[7]	MiTM	Best of 20	Private	(Precision) 98.5%
[33]	MiTM	QL,DQ	Private	-
[34]	MiTM	Softmax	Private	-
[35]	Data Privacy	SGD	-	95%
[36]	Data Privacy	LR	NSL-KDD	-
[37]	Data Privacy	OMPE	Realworld	-
[38]	Data Privacy	SVM	Realworld	94%
[39]	Data Privacy	HBD, NB,DT	-	-
[40]	Anomaly	CNN	Sino Weibo	91.34%

Recurrent neural networks (RNN) have been used by scholars since it became popular for the classification and detection of malware [41], [42], [43]. In RNN API sequence is called by the program that is used as an input the system predicts whether the program is malware or benign. The intimacy of this system was that it processes the sequence chain in only the forward direction although some sequential patterns are in the backward direction so it could not detect that type of malware.

So a solution to this was encountered by introducing bidirectional RNN [44]. It analyses and learns the patterns from both directions. In this, they implement an additional backward RNN that processes backward sequence, but the problem is computational cost accrued with low efficiency. The probability was accustomed to computing the concatenation of the hidden states from both aspects.

Kong and Yan [45] proposed a new system that applies discriminated distance matrices learning. The system was based on a function call graph that extracts fine-grained features between two samples of malware. This method helps to cluster the malware which belongs to the same family while which belongs to others keeping them in another cluster by a minute distance. The issue that weakens this system is that the correctness depends upon the extracted fine-gained features.

Mohaisen and Alrawi [46] submitted a system that detects malware on a large scale. It was an automated system AMAL that analyzed and classifies malware. It was constituted of two subsystems AutoMal and MaLabel. Automal gathers malicious low granularity pattern artifacts, while MaLabel creates representative features through artifacts. MaLabel uses various learning algorithms to classify the malware, the learning algorithms include K-nearest neighbor, decision tree, and SVM. The issue is that in the system we have to run samples of malware in a virtualized environment that causes overhead for the system.

Another technique to detect malware was through the behavior of the malware. It detects the malware as they behave. The downfall of this technique was the analyst become confused by first encrypting and decrypting the malicious code. This also required an emulated environment to observe the behavior of the code the malicious file contained. So to set up the emulated environment takes too much time. [47]

Albeit the above-examined strategies recommended by scientists are extremely valuable and advantageous in identifying and grouping malware, it is fundamental to choose the most useful list of capabilities that contains substantial in-line for the Bot IoT assault recognition in the IoT climate. Different lead steps ought to be remembered while picking highlight determination procedures, for example, following traffic, follow unique traffic, subset age, to produce highlights from following traffic subset assessment, from the following stage assess the created highlights set, and the expert takes a few choices

to pick the compelling technique for include choice. The subset assessment then, at that point, gives the indisputable choice and validates the list of capabilities [48].

Many techniques to identify malware are present. One is the signature-based identification process which has been implemented in various anti-malware software, but the programmers that wrote malware code used new techniques that could not be detective. Hostile to malware programming shields Web clients from malware assaults, and it utilizes a mark-based way to deal with recognize known dangers. This strategy is delayed as it disentangles machine code into a string. It also does not alarm the system of new threats. Heuristic-based technique becomes more popular when it comes out but it has some errors and is also time-consuming. We need intelligent detection techniques and analysis software as the malware writer writes such codes that can easily bypass these detection methods [49], [50], [51], [52]. To identify the new viruses and malware, the traditional approach which is a signature-based approach is not acceptable as it forces a user to update the anti-virus database so the correct level is maintained. The delay in updating by the user side or delay from the company side to the new malware may result in a huge loss that could not be replaceable. Heuristic algorithms are designed to detect unknown malware but are too time-consuming and error rates are too high [3].

Table 2. The downfall of some previous techniques applied to detect malware

Ref.	Techniques	Drawbacks
[28]	Signature-based methods	Applicability and scalability become limited by this method. This method is slow as it decodes machine code into a string
[29]	Static code analysis	Code obfuscation and before analysis, we should have to unpack and decrypt the files
[30]	Dynamic code analysis	Did not identify specific kinds of malware that behave well camouflaged and which do not satisfy trigger conditions
[45]	Discriminated distance matrices learning.	The correctness depends upon the extracted fine-gained features
[46]	Automated system AMAL	We have to run samples of malware in a virtualized environment that causes overhead for the system
[47]	Through the behaviour of the malware.	Have to setup emulated environment to run code so too much time is consumed
[3].	Heuristic-based technique	It has some errors and is also time-consuming
[53], [54],[55]	Optimization-based method	The perturbation is not optimal

[56]	Iteration least likely class method	The performance is affected by the number of iterations.
[57]	DeepFool	Not given the guarantee that the generated adversarial samples are good enough
[58]	Jacobian based saliency map approach	Target DNNs must be a feed-forward network. The computation complexity is high when processing high-dimensional data.

Modern technology and researchers aim to develop such programs that detect malware so the action should be taken instantly. To protect the users' data on their own devices much new work from the researchers has been based on ML algorithms [16], [7], [36], [33], [59], [60], [61]

Although the old methods could not tackle new malware efficiently so we have to move on to new techniques like machine learning methods that are giving us the best outcomes.

3. Materials and Methods

We used an ensemble model to try our best to solve the problem. We choose those algorithms which could efficiently work for us to provide the best results in distinguishing the malware.

3.1. Dataset Details

3.1.1. Data Set 1: Avgsig

The secondary data is collected from https://www.kaggle.com/rquintino/malware-avsigversion-threats. This dictionary contains 20000 entries. One for each different credible AvSigVersion. Antivirus software evaluates themselves bypassing this data set, so 95% of users frequently update them making this a reliable timestamp for each dataset observation. It contains numerical data. This dataset contains multi-viruses. Labeled data, benign or malicious. It contains up-to-date viruses that are surrounding in our cyberspace. [100]

3.1.2. Data Set 2: Windows PE file

This dataset was also obtained from the Kaggle website. It contains 77 different features. This is labeled data set discriminating benign and malicious code extracted from Windows Portable Executable (PE) file. It contains 19,611 samples extracted from different malware repositories [111]. Data set 2 is used for evaluation purposes.

3.2. Algorithms Applied

3.2.1. K-Nearest neighbor (KNN)

KNN, K-Nearest neighbor is the most effectual and simplest technique. In some cases, it is the non-parametric classification method. If we want to classify our data set e.g D we have to retrieve its nearest neighbors so there are supposed to be the neighbors of D. To classify, the voting is done, which has the majority votes in the data set that value is decided. Whereas to begin with KNN we want to choose a value of K, and the results of the classification are very much likely to depend upon the chosen value. [62].

So we could say that the KNN algorithm is based on the value of k. The best way to choose the value of k is to select various values and select which gave the most appropriate result.

If K-NN is used for classification the outer result depends upon:

- In classification, KNN Object becomes the member of that class that gains the most vote from its neighbor.
- K nearest neighbor whereas K is a positive integer. If k=1 then the object is allocated to the class of that single nearest neighbor.

3.2.2. Support Vector Machine (SVM)

SVM is counted in the category of supervised learning model it is used to observe data for classification as well as for regression problems. It is a very vigorous and strong prediction model built on a statically learning approach. Training examples are a map with much gap between the different categories in the space and when test data is given it falls into that predicted space. It could perform linear as well as non-linear classification.[63], [64]

3.2.3. Logistic Regression (LR)

Logistic regression despite the name it's a classification model. It is very effective when comes to binary classification. It models the probability of a discrete outcome when we gave an input value. It is a very effective classification method, especially for cyber security for the detection of attacks. It is an analytical technique.[65]–[67]

3.2.4. Ensemble Model

In an ensemble model, different models are created that predict the final outcome. The ensemble model aggregates the outcomes of different used models and gives the final prediction [3].

To build the primary stage classifiers, first, the preparation information is seen in numerous ways, then, at that point, by coordinating the consequences of all the main stage classifiers another classifier as a group classifier is made and in the last stage, the model figures out how to coordinate the forecasts from the principal stage classifiers.

The stacking approach has been applied which has two phases. In the principal stage, models are prepared on the dataset, and the forecast of each model is put away. We could say another informational collection is framed. In the subsequent stage, the dataset is applied to create the eventual outcome, with the utilization of a meta-learning calculation. [68], [19]

First stage models are also known as base models and the final stage classifier is referred to as a meta-learner. Both are first trained on the training dataset. [19]

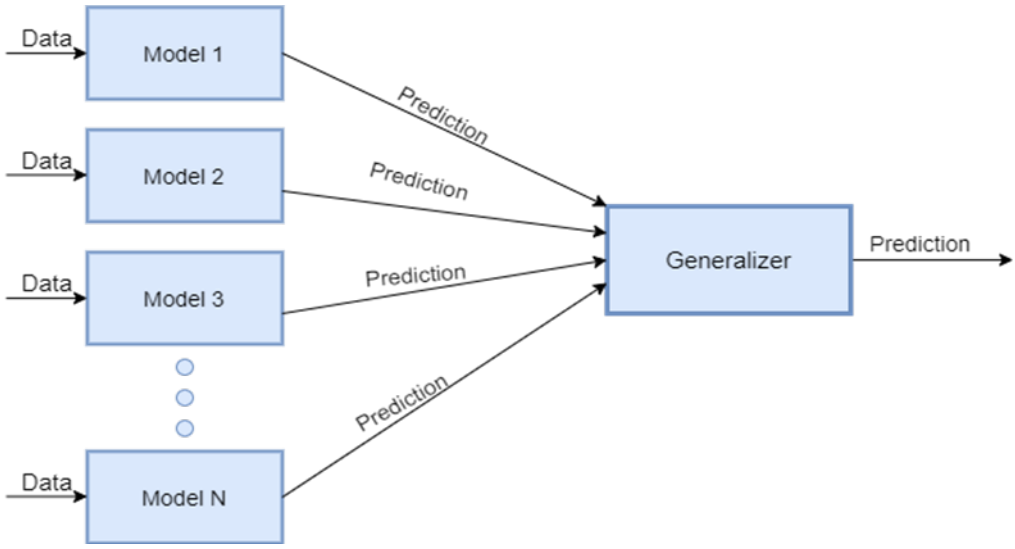

Figure 1. Ensemble Model

3.2.4.1. Stages of the ensemble model

There are three stages of the ensemble learner algorithm

1) To create an ensemble

Choose various base classifiers

Choose the final-stage classifier

2) To Train an Ensemble

By training the dataset train each first-stage classifier

K-fold cross-validation, performed on base models and predictions are recorded

The predictions of the first stage classifiers are then combined to form a feature matrix.

Train the Meta learner based on new data (predictions and features). The ensemble model integrates both the base classifiers and Meta learner to have better predictions on new (unknown) data.

Test the New Data

First-stage classifiers predictions are stored.

Give predictions of the first-stage classifier as an input to the final-stage classifier to have a final result.

3.2.4.2. Ensemble Learner Algorithm (Stacking approach)

```
Input: Training data D ={xi, yi}m i=1
Output: Ensemble classifier E
BEGIN
Step1: learn base-level classifiers
for t=1 to T do
  learn et based on D
 end for
 Step2: Construct new data set for predictions
 for i=1 to m do
 De ={x'i,yi}, where x'i = {e1(xi),...,eT(xi)}
 end for
 Step 3: Learn a meta-classifier
 learn E based on De
 return E
 END
```

Figure 2. Ensemble Algorithm

3.3. The Research Design

Following are the steps we took to proceed with our research.

1. At first, we select features and reduce the dimensionality using an auto-encoder from the data set.
2. Then we classified our data using supervised learning algorithms.
3. We used the ensemble model as our final stage classifier.
4. Then we test our model by comparing the results from previous techniques used.
5. Finally, we evaluate our model based on results, performance, and efficiency.

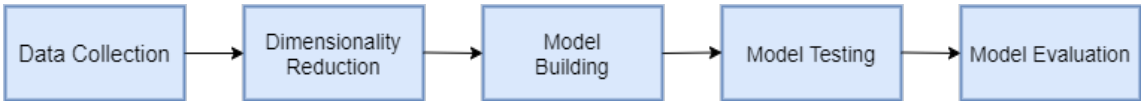

Figure 3. Flow Chart of Research Methodology

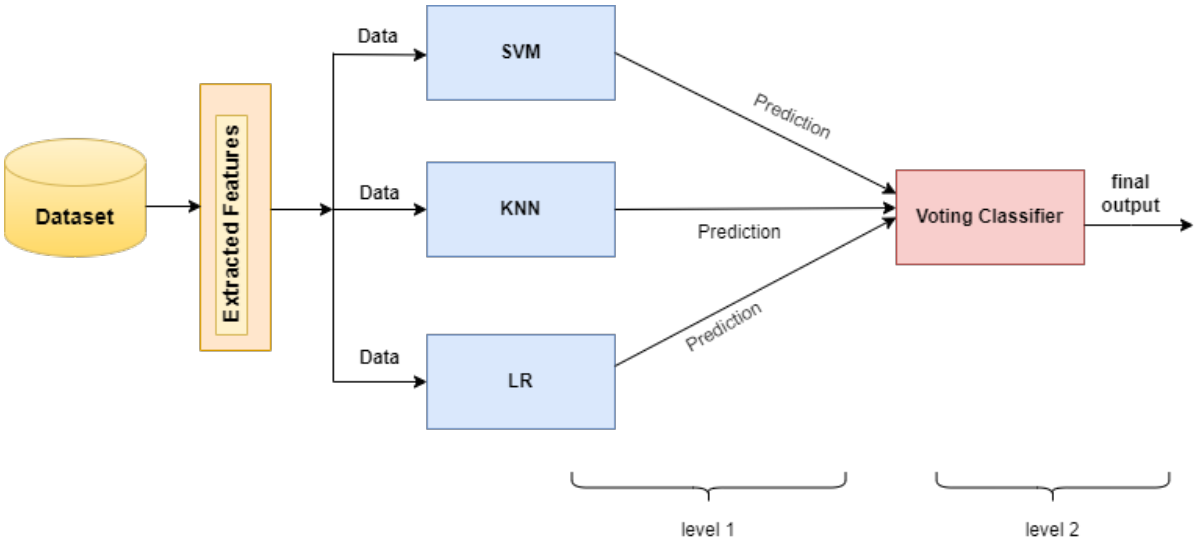

Figure 4. Flowchart of Proposed Methodology

This is a supervised ensemble model. We used this approach because supervised machine learning algorithms are the best in detecting complex malware that is not easily detectable [3]. Machine learning

algorithms detect malware with greater efficiency as compared to other techniques. The ensemble model gives us the best results overall.

3.3.1. A Voting Classifier

A voting classifier gave results on aggregating the outcomes of the base models or first stage models.[69]–[71] The final output of the first stage classifier is weak, it combined their results on the bases of voting of each base model and gives the final results into a voting classifier. [72], [73]

3.3.2. *Hard Voting Classifier*

We applied hard voting which gives us the outcome by aggregating the prediction of every class or every first stage classifier and gives the final prediction that has more votes.

3.3.3. Feature Extraction

The dataset is high dimensional when it contains a large number of attributes. If the dataset is of high dimensional the training and running time of the machine learning model took more time. It creates an overhead for the system and also create complexity. So it is prime to extract only important features and exclude the unimportant featurers.[74]

Our objective is simplify the dataset with vital, to the point and relevant features. This results in dropping the dimensionality of the raw dataset and the outcome could be seen in varience, overfitting and most of all reducing the overall computaytional cost of the model.

3.4. Steps We Performed In Our Research

3.4.1. STEP 1: Feature Extracting

We first extract the features from the dataset by using PCA so that the processing of data takes less time.We then saved the extracted feature dataset into a file so when needed we directly import it without processing the dataset again.

3.4.2. STEP 2: Training

Then we split our data into two half, training and testing datasets. Then train the chosen first stage classifiers (SVM, KNN, LR) by using the training dataset and then train the meta learner (voting classifier) using the results of first stage classifiers.

3.4.3. STEP 3: Testing

After training our model we passes the test data through the ensemble model to test the model. Wheather it giving us the expected results or not.

3.4.4. STEP 4: Evaluating

Then we take the results and evaluate the model by using the parameters and by comparing the generated results of our model with previous proposed model.

4. Results and Evaluations

Performance and evaluation will be measured, based on:

1. Precision
2. F-score
3. Accuracy
4. Error rate
5. Comparison with the previous model result

The critical measures are the amount of precisely perceived malware known as TP called genuine positive and the code that has been wrongly particular is known as FP called a bogus positive. Following are the limits that we will use for the presentation evaluation of our model.

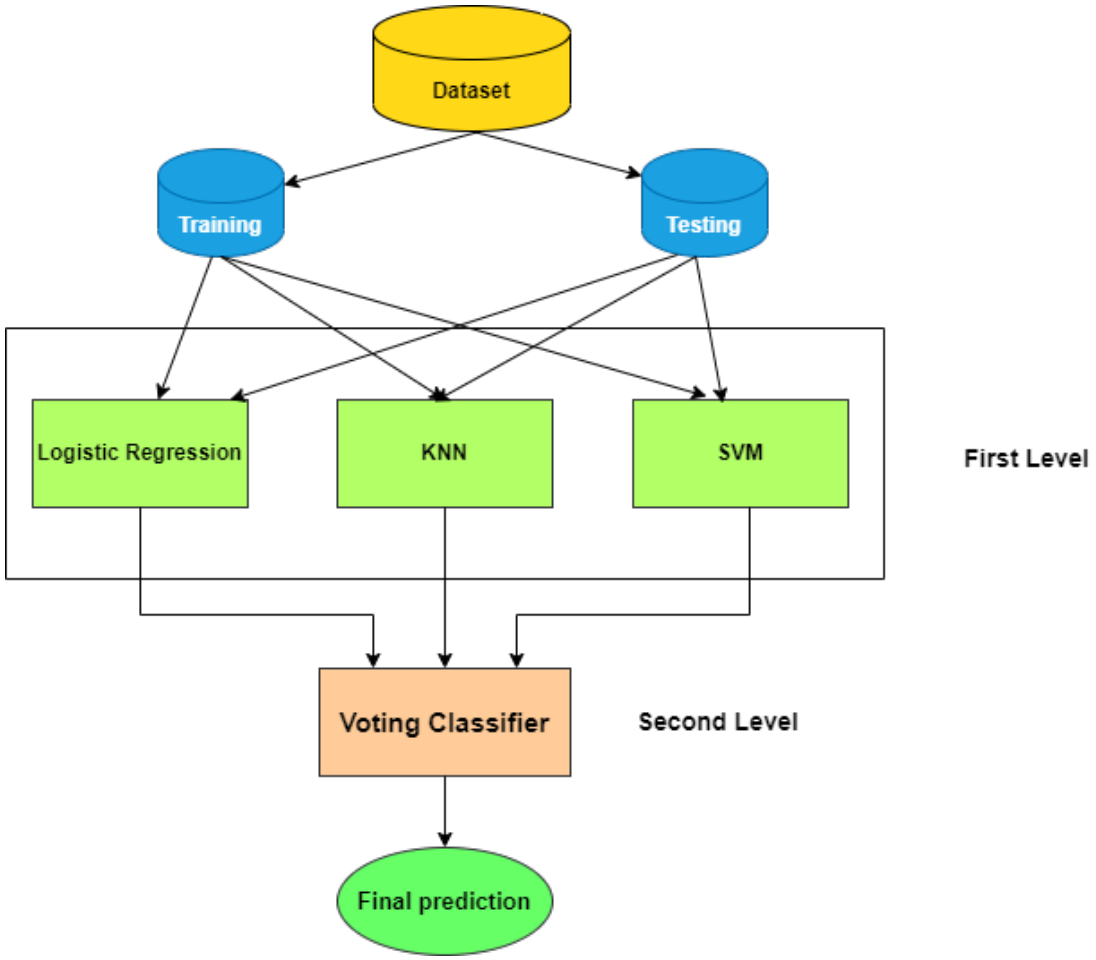

Figure 5. Flow Chart of Proposed Model

The critical measures are the amount of precisely perceived malware known as TP called genuine positive and the code that has been wrongly particular is known as FP called a bogus positive. Following are the limits that we will use for the presentation evaluation of our model.

A genuine positive outcome that is really recognized is TP.

Misleading up-sides results are wrongly recognized as FP.

False negatives: A tempered script that is undetected and set apart as not malware is FN.

True negatives: The script that is malware and it also effectively identified is TN.

True positive rate: Genuine positive rate: The true positive rate (TPR moreover called affectability) is the genuine positive likelihood that will test positive. Its condition is not set in stone as:

$$TRP = \frac{TP}{TP + FN} \quad (1)$$

True Negative Rate (TNR): The true negative rate is generally called distinction which is the probability that I certified negative will test negative. It's not set in stone as:

$$TNR = \frac{TN}{TN + FP} \quad (2)$$

False Positive Rate (FPR): The false positive rate is determined as the proportion of misleading up-sides (FP) to the absolute number of negatives (FP + TN), where FP addresses the quantity of misleading up-sides and TN addresses the quantity of genuine negatives. This rate indicates the probability of incorrectly predicting a positive result when the actual value is negative. The formula is:

$$FPR = \frac{FP}{FP + TN} \quad (3)$$

False-negative rate (FNR): The misleading negative rate, otherwise called the miss rate, is the likelihood that a true positive will be remembered fondly by the test. It is determined as the proportion of false negatives (FN) to the absolute number of up-sides (FN + TP), where FN addresses the quantity of misleading negatives and TP addresses the quantity of genuine up-sides. It's not entirely settled as:

$$FNR = \frac{Fn}{FN + TP} \quad (4)$$

Precision and Recall: Precision connotes the probability that a recognized malware is truly malware, while review shows the probability that malware is distinguished. The Precision is meant by P in the underneath given conditions. The recall is ordinarily otherwise called the genuine positive rate. It is meant by R in the beneath given conditions. We moreover give the score as an activity that merges accuracy and review into a solitary worth. Accuracy and review are on the other hand comparative with each other and in this manner understanding their variations is huge in building a capable gathering structure. They're entirely settled as:

$$Recall = \frac{True\ positives(TP's)}{True\ positives\ (TP's) + False\ negatives\ (FN's)} \quad (5)$$

$$Precision = \frac{True\ positive(Tp's)}{True\ positives\ (TP's) + False\ positives\ (FP's)} \quad (6)$$

Recall and precision are on the other hand comparative with each other and in this manner understanding their dis-equalities is huge in building a capable gathering system. TP is the amount of precisely distinguished malware. FP implies the amount of incorrectly perceived malware and FN are the dis-really missed malware, equivocal.

F Measure/F1 score: The F score, likewise, called the F1 score or F measure, is a measurement used to assess the exactness of a test. It is the weighted consonant mean of the test's recall and precision. The F score is determined utilizing the accompanying recipe:

$$F'Measure = 2 * \frac{Precission * Recall}{Precission + Recall} \quad (7)$$

At the point when you really want to find a harmony among accuracy and review and there is an inconsistent class conveyance i.e. (countless Genuine Negatives), you'll require the F1 Score.

Accuracy: Accuracy is one method for estimating a model's exhibition. Casually, it addresses the extent of right forecasts made by the model. Officially, exactness is characterized as:

Precision = number of right forecasts/Absolute number of expectations

In double grouping, exactness can likewise be communicated utilizing the quantity of genuine up-sides and genuine negatives.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (8)$$

Table 3. Performance Measure

Performance measure	SVM	KNN	LR	Voting Classifier	Total
Accuracy	98.71%	99.82%	49.66%	99.70%	100%
Error	1.29%	0.18%	50%	0.30%	100%
FPR	224	11	4087	0	20000
TPR	9709	9922	5875	9933	20000
FNR	34	25	3267	59	20000
TNR	10033	10042	6771	10008	20000
Precision	0.98	1.00	0.50	1.00	1.00
Recall	1.00	1.00	0.50	1.00	1.00
F-measure	0.99	1.00	0.66	0.99	1.00

4.1. Experimental Analysis

System Settings

Hardware and Software

We use Python with its different libraries, Ski-learn., pandas, Seaborn, matplolib, numpy, and datetime to implement our methodology and all experiments are carried out on the computer with a 64-bit Windows 10 operating system with Intel Core i5-8265U CPU 1.80 GHz with 8GB RAM.

Splitting of data: We split our information into two sections. One is for preparing and the other is for trying. It's very crucial to split the data keenly so the parameter could be well-tuned.

Training

Eighty percent (80%) of data is used for training.

Testing

Twenty percent (20%) of data is for testing

4.2. Final Results

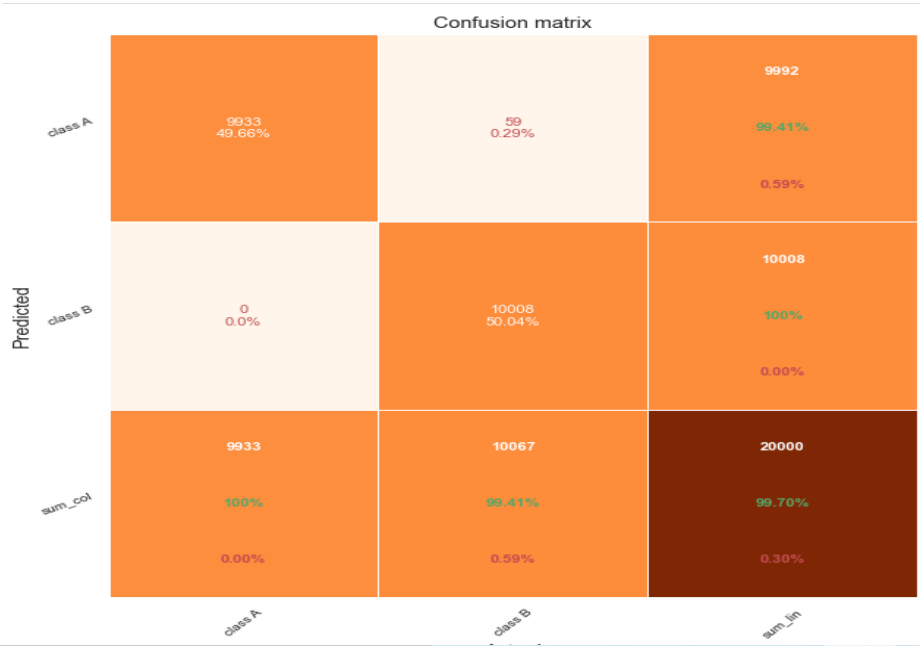

Figure 6. Ensemble Voting Classifier Confusion Matrix

	precision	recall	f1-score	support
0	1.00	0.98	0.99	9969
1	0.98	1.00	0.99	10031
accuracy			0.99	20000
macro avg	0.99	0.99	0.99	20000
weighted avg	0.99	0.99	0.99	20000

Figure 7. The Ensemble Voting classifier Summary

The results we got from the baseline classifiers of ML methods were shown in table 4.1. The confusion matrices are presented in figure 4.3. The finest results were of KNN with an accuracy of 99.82%. False-positive 0.06%. False negatives 0.12%, F1 score of 100% indicates that KNN classifies the classes well.

However, logistic Regression shows less likely results than the other base models the reason behind, this might be that the features of every class are very strongly correlated with one another or the unfair splitting of data in the testing and training set might cause LR to perform poorly. LR gave us 49.77% accuracy and F-measure 66%. SVM performed well with an accuracy of 98.71. F-score 99%.

Although the ensemble model that we ought to be our final output gives us 99.7% accuracy and F-Score 99% with a False Positive rate of 0%.

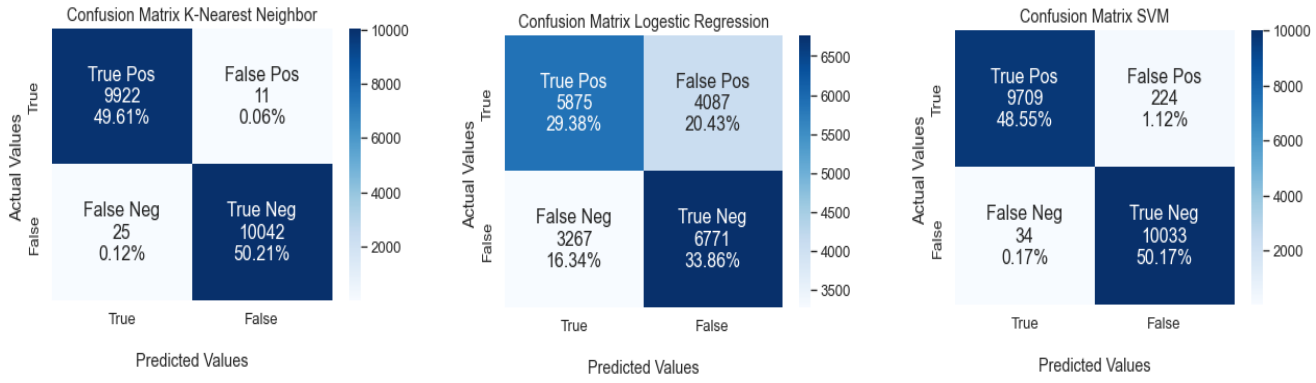

Figure 8. Confusion matrixes of baseline machine learning methods

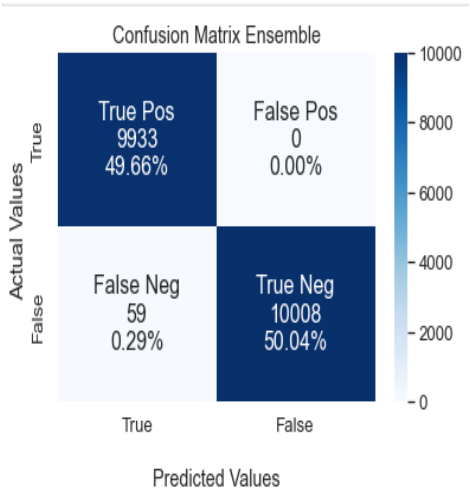

Figure 9. Confusion matrix of Ensemble model

	precision	recall	f1-score	support
0	1.00	1.00	1.00	9933
1	1.00	1.00	1.00	10067
accuracy			1.00	20000
macro avg	1.00	1.00	1.00	20000
weighted avg	1.00	1.00	1.00	20000

Figure 10. KNN Performance Summary

	precision	recall	f1-score	support
0	0.64	0.59	0.62	9962
1	0.62	0.67	0.65	10038
accuracy			0.63	20000
macro avg	0.63	0.63	0.63	20000
weighted avg	0.63	0.63	0.63	20000

Figure 11. LR Performance Summary

	precision	recall	f1-score	support
0	1.00	0.98	0.99	9969
1	0.98	1.00	0.99	10031
accuracy			0.99	20000
macro avg	0.99	0.99	0.99	20000
weighted avg	0.99	0.99	0.99	20000

Figure 12. SVM Performance Summary

5. Comparison with Previous Work

We compared our results with other researchers' malware detection techniques. Note that the data set is not the same but for [19], the dataset and system settings were the same and we achieved a higher performance with an accuracy of 99.8%.

Table 4. Comparison with other malware detection techniques

References	Benign	Malware	Accuracy	Precision	Recall	F-score
[75]	-	4850	98.14	n/a	n/a	n/a
[76]	3000	3000	96.92	96.75	97.23	96.99
[77]	4596	4596	97.23	98.69	98.69	98.69
[78]	5065	426	93.4	93.5	93.4	93.2
[79]	1000	1000	96	n/a	95	n/a
[80]	-	18,831	99.03	n/a	n/a	n/a

[81]	-	7826	99.8	n/a	n/a	n/a
[82]	1,000,020	1,011,766	99.41	n/a	n/a	89.02
[19]	5012	14,598	98.6	96.3	99	n/a
This paper	5012	14,598	99.8	99.7	99.2	98.2

Time complexity: Time complexity of the ensemble model is 3.52 sec approximately

```
stop = timeit.default_timer()

print('Time: ', stop - start)
```

```
Time:  232.0227732
```

Figure 13. Time Complexity

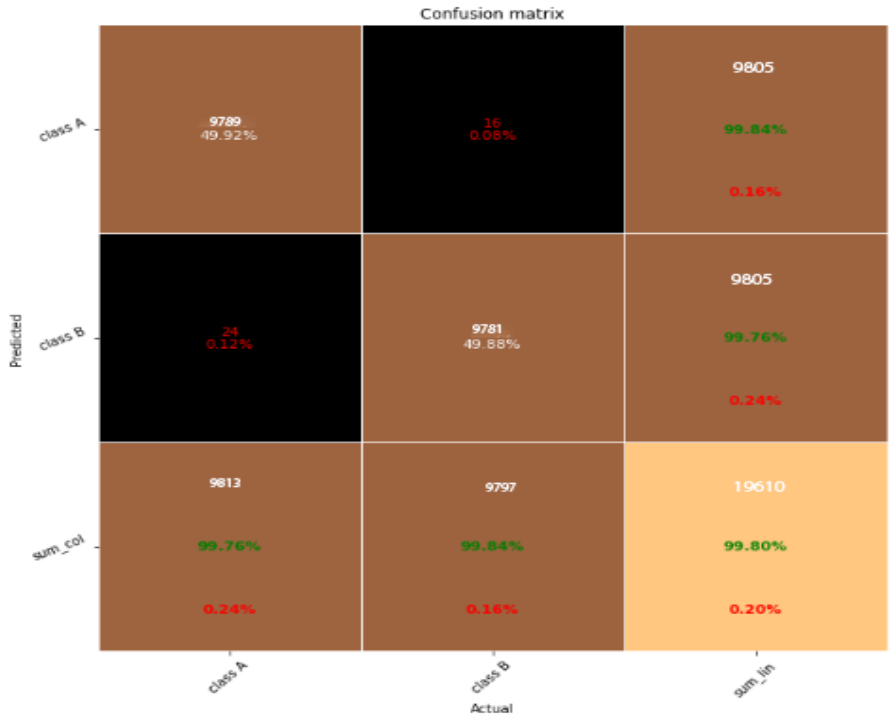

Figure 14. Confusion Matrix for Comparison

6. Results Discussion

So viewing the outcome of our model it shows very good results by taking less running time as well. It gave an accuracy of 99.7% which is a great outcome. If we talk about other parameters such as true negative, false negative, false positive, and true positive we got over all great results with error rate 0.30% overall. F-score is 99 %.

By analyzing the results of each algorithm used in our model, KNN gives us the best results overall with the accuracy of 99.82% and f-score 100%. And we observe logistic regression perform poorly as compared to others with just 49.77% of accuracy and F-score 66%.

This might be that the features of every class are very strongly correlated with one another or the imbalanced splitting of data in the testing and training set. Our running time is also efficient.

The comparison results are also good. We compared our model using the same dataset and same system requirements which they have applied and it gives us the accuracy of 99.8% .Whereas their model gave 98.6 % accuracy on the same dataset.

7. Conclusion and Future Work

We presented a successful and vigorous strategy for a supervised ensemble model. Using voting classifier as our Meta learner classifier. We applied hard voting here. The algorithms we applied in our ensemble model are KNN, SVM, and Logistic Regression. Altogether our model turn out to be successful in distinguished complex malware. The accuracy result is overall 99.7% with a running time of 3.52 sec.

In the future, we will adopt deep learning algorithms, and adopt unsupervised algorithms that could also distinguish unknown malware, and also time could be saved by the analyst to label the data first. We are thinking of better improving our vulnerabilities and protecting our privacy by blocking the malware on time and alerting the user.

References

1. N. Waheed, X. He, M. Ikram, M. Usman, S. S. Hashmi, and M. Usman, "Security and Privacy in IoT Using Machine Learning and Blockchain: Threats and Countermeasures," ACM Comput. Surv., vol. 53, no. 6, 2021, doi: 10.1145/3417987.
2. S. Kumar, P. Tiwari, and M. Zymbler, "Internet of Things is a revolutionary approach for future technology enhancement: a review," J. Big Data, vol. 6, no. 1, 2019, doi: 10.1186/s40537-019-0268-2.
3. M. Shafiq, Z. Tian, A. K. Bashir, X. Du, and M. Guizani, "CorrAUC: A Malicious Bot-IoT Traffic Detection Method in IoT Network Using Machine-Learning Techniques," IEEE Internet Things J., vol. 8, no. 5, pp. 3242–3254, 2021, doi: 10.1109/JIOT.2020.3002255.
4. Y. N. Soe, Y. Feng, P. I. Santosa, R. Hartanto, and K. Sakurai, Implementing Lightweight IoT-IDS on Raspberry Pi Using Correlation-Based Feature Selection and Its Performance Evaluation, vol. 926. Springer International Publishing, 2020. doi: 10.1007/978-3-030-15032-7_39.
5. S. Jajodia, V. S. Subrahmanian, V. Swarup, and C. Wang, Cyber deception: Building the scientific foundation. 2016. doi: 10.1007/978-3-319-32699-3.
6. D. E. Kouicem, A. Bouabdallah, and H. Lakhlef, "Internet of things security: A top-down survey," Comput. Networks, vol. 141, pp. 199–221, 2018, doi: 10.1016/j.comnet.2018.03.012.
7. H. Aksu, A. S. Uluagac, and E. S. Bentley, "Identification of Wearable Devices with Bluetooth," IEEE Trans. Sustain. Comput., vol. 6, no. 2, pp. 221–230, 2018, doi: 10.1109/tsusc.2018.2808455.
8. K. Fan, Y. Ren, Y. Wang, H. Li, and Y. Yang, "Blockchain-based efficient privacy preserving and data sharing scheme of content-centric network in 5G," IET Commun., vol. 12, no. 5, pp. 527–532, 2018, doi: 10.1049/iet-com.2017.0619.
9. K. K. F. Yuen, "Towards a Cybersecurity Investment Assessment method using Primitive Cognitive Network Process," 1st Int. Conf. Artif. Intell. Inf. Commun. ICAIIC 2019, pp. 68–71, 2019, doi: 10.1109/ICAIIC.2019.8668842.
10. P. Prabhu and K. N. Manjunath, Secured image transmission in medical imaging applications—A survey, vol. 31. Springer International Publishing, 2019. doi: 10.1007/978-3-030-04061-1_12.
11. M. Brewczyńska, S. Dunn, and A. Elijahu, "Data Privacy Laws Response to Ransomware Attacks: A Multi-Jurisdictional Analysis," pp. 281–305, 2019, doi: 10.1007/978-94-6265-279-8_15.
12. C. Tselios, I. Politis, and S. Kotsopoulos, "Enhancing SDN security for iot-related deployments through blockchain," 2017 IEEE Conf. Netw. Funct. Virtualization Softw. Defin. Networks, NFV-SDN 2017, vol. 2017-Janua, pp. 303–308, 2017, doi: 10.1109/NFV-SDN.2017.8169860.
13. M. Al-Rubaie and J. M. Chang, "Privacy-Preserving Machine Learning: Threats and Solutions," IEEE Secur. Priv., vol. 17, no. 2, pp. 49–58, 2019, doi: 10.1109/MSEC.2018.2888775.
14. J. Weng, J. Weng, J. Zhang, M. Li, Y. Zhang, and W. Luo, "DeepChain: Auditable and Privacy-Preserving Deep Learning with Blockchain-based Incentive," IEEE Trans. Dependable Secur. Comput., vol. PP, no. 8, pp. 1–1, 2019, doi: 10.1109/tdsc.2019.2952332.
15. A. Kumar and T. J. Lim, "EDIMA: Early Detection of IoT Malware Network Activity Using Machine Learning Techniques," IEEE 5th World Forum Internet Things, WF-IoT 2019 - Conf. Proc., pp. 289–294, 2019, doi: 10.1109/WF-IoT.2019.8767194.
16. W. N. Price and I. G. Cohen, "Privacy in the age of medical big data," Nat. Med., vol. 25, no. 1, pp. 37–43, 2019, doi: 10.1038/s41591-018-0272-7.
17. C. K. Williams C, Chaturvedi R, "No TitleWilliams C, Chaturvedi R, Chakravarthy K Cybersecurity Risks in a Pandemic J Med Internet Res 2020;22(9):e23692 URL: https://www.jmir.org/2020/9/e23692 DOI: 10.2196/23692".
18. S. Hakak, W. Z. Khan, M. Imran, K. K. R. Choo, and M. Shoaib, "Have You Been a Victim of COVID-19-Related Cyber Incidents? Survey, Taxonomy, and Mitigation Strategies," IEEE Access, vol. 8, pp. 124134–124144, 2020, doi: 10.1109/ACCESS.2020.3006172.

19. N. A. Azeez, O. E. Odufuwa, S. Misra, J. Oluranti, and R. Damaševičius, "Windows PE Malware Detection Using Ensemble Learning," Informatics, vol. 8, no. 1, p. 10, 2021, doi: 10.3390/informatics8010010.
20. M. Fan, T. Liu, J. Liu, X. Luo, L. Yu, and X. Guan, Android malware detection: a survey, vol. 50, no. 8. Springer International Publishing, 2020. doi: 10.1360/SSI-2019-0149.
21. D. Ventura, D. Casado-Mansilla, J. López-de-Armentia, P. Garaizar, D. López-de-Ipiña, and V. Catania, "ARIIMA: A real IoT implementation of a machine-learning architecture for reducing energy consumption," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 8867, pp. 444–451, 2014, doi: 10.1007/978-3-319-13102-3_72.
22. R. Xue, L. Wang, and J. Chen, "Using the IOT to construct ubiquitous learning environment," 2011 2nd Int. Conf. Mech. Autom. Control Eng. MACE 2011 - Proc., pp. 7878–7880, 2011, doi: 10.1109/MACE.2011.5988881.
23. D. Gibert, C. Mateu, and J. Planes, "The rise of machine learning for detection and classification of malware: Research developments, trends and challenges," J. Netw. Comput. Appl., vol. 153, p. 102526, 2020, doi: 10.1016/j.jnca.2019.102526.
24. M. Christodorescu and S. Jha, "Testing Malware Detectors ∗ Categories and Subject Descriptors," Security, vol. 29, no. 4, p. 34, 2004.
25. V. Nasteski, "An overview of the supervised machine learning methods," Horizons.B, vol. 4, no. December 2017, pp. 51–62, 2017, doi: 10.20544/horizons.b.04.1.17.p05.
26. D. Opitz and R. Maclin, "—◉ ¤Ÿ ®¡¤¥ 0¤©¤§ # "©¤£¦ Q¡ P¢ ¤£ 0¢," vol. 1, 1999.
27. M. Cord and S. J. Delany, "Supervised learning," Springer Tracts Adv. Robot., vol. 61, pp. 7–13, 2010, doi: 10.1007/978-3-642-11210-2_2.
28. L. Nataraj, S. Karthikeyan, G. Jacob, and B. S. Manjunath, "Malware images: Visualization and automatic classification," ACM Int. Conf. Proceeding Ser., 2011, doi: 10.1145/2016904.2016908.
29. E. Gandotra, D. Bansal, and S. Sofat, "Malware Analysis and Classification: A Survey," J. Inf. Secur., vol. 05, no. 02, pp. 56–64, 2014, doi: 10.4236/jis.2014.52006.
30. A. Makandar and A. Patrot, "Overview of Malware Analysis and Detection," Int. J. Comput. Appl., no. Nckite, pp. 975–8887, 2015.
31. X. Liu et al., A Novel Method for Malware Detection on ML-based Visualization Technique. 2019. doi: 10.1016/j.cose.2019.101682.
32. M. A. Alsheikh, S. Lin, D. Niyato, and H. P. Tan, "Machine learning in wireless sensor networks: Algorithms, strategies, and applications," IEEE Commun. Surv. Tutorials, vol. 16, no. 4, pp. 1996–2018, 2014, doi: 10.1109/COMST.2014.2320099.
33. L. Xiao, Y. Li, G. Han, G. Liu, and W. Zhuang, "PHY-Layer Spoofing Detection with Reinforcement Learning in Wireless Networks," IEEE Trans. Veh. Technol., vol. 65, no. 12, pp. 10037–10047, 2016, doi: 10.1109/TVT.2016.2524258.
34. L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu, "IoT Security Techniques Based on Machine Learning," pp. 1–20, 2018.
35. X. Ma, J. Ma, H. Li, Q. Jiang, and S. Gao, "PDLM: Privacy-Preserving Deep Learning Model on Cloud with Multiple Keys," IEEE Trans. Serv. Comput., vol. 14, no. 4, pp. 1251–1263, 2021, doi: 10.1109/TSC.2018.2868750.
36. T. Zhang and Q. Zhu, "Distributed Privacy-Preserving Collaborative Intrusion Detection Systems for VANETs," IEEE Trans. Signal Inf. Process. over Networks, vol. 4, no. 1, pp. 148–161, 2018, doi: 10.1109/TSIPN.2018.2801622.
37. Q. Jing, A. V. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the Internet of Things: perspectives and challenges," Wirel. Networks, vol. 20, no. 8, pp. 2481–2501, 2014, doi: 10.1007/s11276-014-0761-7.
38. H. Zhu, X. Liu, R. Lu, and H. Li, "Efficient and Privacy-Preserving Online Medical Prediagnosis Framework Using Nonlinear SVM," IEEE J. Biomed. Heal. Informatics, vol. 21, no. 3, pp. 838–850, 2017, doi: 10.1109/JBHI.2016.2548248.
39. X. Sun, P. Zhang, J. K. Liu, J. Yu, and W. Xie, "Private Machine Learning Classification Based on Fully Homomorphic

Encryption," IEEE Trans. Emerg. Top. Comput., vol. 8, no. 2, pp. 352–364, 2020, doi: 10.1109/TETC.2018.2794611.

40. B. Feng, Q. Fu, M. Dong, D. Guo, and Q. Li, "Multistage and Elastic Spam Detection in Mobile Social Networks through Deep Learning," IEEE Netw., vol. 32, no. 4, pp. 15–21, 2018, doi: 10.1109/MNET.2018.1700406.
41. S. Lawe and R. Wang, "Optimization of Traffic Signals Using Deep," AI 2016 Adv. Artif. Intell. AI 2016. Lect. Notes Comput. Sci., vol. 9992, pp. 403–415, 2016, doi: 10.1007/978-3-319-50127-7.
42. R. Pascanu, J. W. Stokes, H. Sanossian, M. Marinescu, and A. Thomas, "Malware classification with recurrent networks," ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc., vol. 2015-Augus, pp. 1916–1920, 2015, doi: 10.1109/ICASSP.2015.7178304.
43. S. Tobiyama, Y. Yamaguchi, H. Shimada, T. Ikuse, and T. Yagi, "Malware Detection with Deep Neural Network Using Process Behavior," Proc. - Int. Comput. Softw. Appl. Conf., vol. 2, pp. 577–582, 2016, doi: 10.1109/COMPSAC.2016.151.
44. B. Iung, "Cœur et grossesse," EMC - Trait. médecine AKOS, vol. 8, no. 2, pp. 1–4, 2013, doi: 10.1016/s1634-6939(13)59289-1.
45. D. Kong and G. Yan, "Discriminant malware distance learning on structural information for automated malware classification," Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., vol. Part F1288, pp. 1357–1365, 2013, doi: 10.1145/2487575.2488219.
46. A. Mohaisen, O. Alrawi, and M. Mohaisen, "AMAL: High-fidelity, behavior-based automated malware analysis and classification," Comput. Secur., vol. 52, pp. 251–266, 2015, doi: 10.1016/j.cose.2015.04.001.
47. O. Pluskal, "Behavioural malware detection using efficient SVM implementation," Proceeding 2015 Res. Adapt. Converg. Syst. RACS 2015, pp. 296–301, 2015, doi: 10.1145/2811411.2811516.
48. M. Wazzan, D. Algazzawi, O. Bamasaq, and A. Albeshri, "applied sciences Internet of Things Botnet Detection Approaches : Analysis and Recommendations for Future Research," 2021.
49. D. Ucci, L. Aniello, and R. Baldoni, "Survey of machine learning techniques for malware analysis," Comput. Secur., vol. 81, pp. 123–147, 2019, doi: 10.1016/j.cose.2018.11.001.
50. A. Souri and R. Hosseini, "A state-of-the-art survey of malware detection approaches using data mining techniques," Human-centric Comput. Inf. Sci., vol. 8, no. 1, 2018, doi: 10.1186/s13673-018-0125-x.
51. S. Najari, "Malware Detection Using Data Mining Techniques," Int. J. Intell. Inf. Syst., vol. 3, no. 6, p. 33, 2014, doi: 10.11648/j.ijiis.s.2014030601.16.
52. J. Gardiner and S. Nagaraja, "On the security of machine learning in malware C&C detection: A survey," ACM Comput. Surv., vol. 49, no. 3, pp. 1–38, 2016, doi: 10.1145/3003816.
53. Z. Tan, A. Jamdagni, X. He, P. Nanda, and R. P. Liu, "A system for denial-of-service attack detection based on multivariate correlation analysis," IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 2, pp. 447–456, 2014, doi: 10.1109/TPDS.2013.146.
54. N. Carlini and D. Wagner, "Towards Evaluating the Robustness of Neural Networks," Proc. - IEEE Symp. Secur. Priv., pp. 39–57, 2017, doi: 10.1109/SP.2017.49.
55. C. Szegedy et al., "Intriguing properties of neural networks," 2nd Int. Conf. Learn. Represent. ICLR 2014 - Conf. Track Proc., pp. 1–10, 2014.
56. I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc., pp. 1–11, 2015.
57. A. Fawzi and P. Frossard, "DeepFool : a simple and accurate method to fool deep neural networks," pp. 2574–2582.
58. N. Papernot, P. Mcdaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," Proc. - 2016 IEEE Eur. Symp. Secur. Privacy, EURO S P 2016, pp. 372–387, 2016, doi: 10.1109/EuroSP.2016.36.
59. L. Xiao, Y. Li, G. Liu, Q. Li, and W. Zhuang, "Spoofing detection with reinforcement learning in wireless networks,"

2015 IEEE Glob. Commun. Conf. GLOBECOM 2015, 2015, doi: 10.1109/GLOCOM.2014.7417078.

60. L. Wei, W. Luo, J. Weng, Y. Zhong, X. Zhang, and Z. Yan, "SPECIAL SECTION ON INTERNET-OF-THINGS (IOT) BIG DATA TRUST MANAGEMENT Machine Learning-Based Malicious Application Detection of Android," vol. 5, pp. 25591–25601, 2017.
61. Q. Liu, P. Li, W. Zhao, W. Cai, S. Yu, and V. C. M. Leung, "A survey on security threats and defensive techniques of machine learning: A data driven view," IEEE Access, vol. 6, pp. 12103–12117, 2018, doi: 10.1109/ACCESS.2018.2805680.
62. G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, "KNN model-based approach in classification," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 2888, no. November 2012, pp. 986–996, 2003, doi: 10.1007/978-3-540-39964-3_62.
63. T. Evgeniou and M. Pontil, "Support vector machines: Theory and applications," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 2049 LNAI, no. May, pp. 249–257, 2001, doi: 10.1007/3-540-44673-7_12.
64. B. Gaye, D. Zhang, and A. Wulamu, "Improvement of Support Vector Machine Algorithm in Big Data Background," Math. Probl. Eng., vol. 2021, 2021, doi: 10.1155/2021/5594899.
65. J. L. Alzen, L. S. Langdon, and V. K. Otero, "A logistic regression investigation of the relationship between the Learning Assistant model and failure rates in introductory STEM courses," Int. J. STEM Educ., vol. 5, no. 1, pp. 1–12, 2018, doi: 10.1186/s40594-018-0152-1.
66. C. Y. J. Peng, K. L. Lee, and G. M. Ingersoll, "An introduction to logistic regression analysis and reporting," J. Educ. Res., vol. 96, no. 1, pp. 3–14, 2002, doi: 10.1080/00220670209598786.
67. B. T. Zewude and K. M. Ashine, "Binary Logistic Regression Analysis in Assessment and Identifying Factors That Influence Students ' Academic Achievement : The Case of College of Natural and Computational," J. Educ. Pract., vol. 7, no. 25, pp. 1–6, 2016.
68. R. Maclin, "Popular Ensemble Methods : An Empirical Study Popular Ensemble Methods : An Empirical Study," J. Artif. Intell. Res., vol. 11, no. July, pp. 169–198, 2016.
69. H. Kim, H. Kim, H. Moon, and H. Ahn, "A weight-adjusted voting algorithm for ensembles of classifiers," J. Korean Stat. Soc., vol. 40, no. 4, pp. 437–449, 2011, doi: 10.1016/j.jkss.2011.03.002.
70. U. K. Kumar, M. B. S. Nikhil, and K. Sumangali, "1-vote-Naïve Bayes SMV J48.pdf," no. August, pp. 108–114, 2017.
71. E. S. M. El-Kenawy, A. Ibrahim, S. Mirjalili, M. M. Eid, and S. E. Hussein, "Novel feature selection and voting classifier algorithms for COVID-19 classification in CT images," IEEE Access, vol. 8, 2020, doi: 10.1109/ACCESS.2020.3028012.
72. M. Culp, K. Johnson, and G. Michailidis, "Ada: An R package for stochastic boosting," J. Stat. Softw., vol. 17, no. 2, pp. 1–27, 2006, doi: 10.18637/jss.v017.i02.
73. D. W. Opitz and R. F. MacLin, "An empirical evaluation of bagging and boosting for artificial neural networks," IEEE Int. Conf. Neural Networks - Conf. Proc., vol. 3, pp. 1401–1405, 1997, doi: 10.1109/ICNN.1997.613999.
74. W. K. Mutlag, S. K. Ali, Z. M. Aydam, and B. H. Taher, "Feature Extraction Methods: A Review," J. Phys. Conf. Ser., vol. 1591, no. 1, 2020, doi: 10.1088/1742-6596/1591/1/012028.
75. K. Bakour and H. M. Ünver, "VisDroid: Android malware classification based on local and global image features, bag of visual words and machine learning techniques," Neural Comput. Appl., vol. 33, no. 8, pp. 3133–3153, 2021, doi: 10.1007/s00521-020-05195-w.
76. L. Cai, Y. Li, and Z. Xiong, "JOWMDroid: Android malware detection based on feature weighting with joint optimization of weight-mapping and classifier parameters," Comput. Secur., vol. 100, p. 102086, 2021, doi: 10.1016/j.cose.2020.102086.
77. J. Chen et al., "SLAM: A Malware Detection Method Based on Sliding Local Attention Mechanism," Secur. Commun.

Networks, vol. 2020, 2020, doi: 10.1155/2020/6724513.

78. S. I. Imtiaz, S. ur Rehman, A. R. Javed, Z. Jalil, X. Liu, and W. S. Alnumay, "DeepAMD: Detection and identification of Android malware using high-efficient Deep Artificial Neural Network," Futur. Gener. Comput. Syst., vol. 115, pp. 844–856, 2021, doi: 10.1016/j.future.2020.10.008.
79. . Jeon and J. Moon, "Malware-Detection Method with a Convolutional Recurrent Neural Network Using Opcode Sequences," Inf. Sci. (Ny)., vol. 535, pp. 1–15, 2020, doi: 10.1016/j.ins.2020.05.026.
80. A. Namavar Jahromi et al., "An improved two-hidden-layer extreme learning machine for malware hunting," Comput. Secur., vol. 89, p. 101655, 2020, doi: 10.1016/j.cose.2019.101655.
81. B. N. Narayanan and V. S. P. Davuluru, "Ensemble malware classification system using deep neural networks," Electron., vol. 9, no. 5, 2020, doi: 10.3390/electronics9050721.
82. V. P. Ranganthan, R. Dantu, A. Paul, P. Mears, and K. Morozov, "A decentralized marketplace application on the ethereum blockchain," Proc. - 4th IEEE Int. Conf. Collab. Internet Comput. CIC 2018, pp. 90–97, 2018, doi: 10.1109/CIC.2018.00023.
83. Soe, Y. N., Feng, Y., Santosa, P. I., Hartanto, R., & Sakurai, K. (2020). Implementing Lightweight IoT-IDS on Raspberry Pi Using Correlation-Based Feature Selection and Its Performance Evaluation. In Advances in Intelligent Systems and Computing (Vol. 926). Springer International Publishing. https://doi.org/10.1007/978-3-030-15032-7_39
84. Souri, A., & Hosseini, R. (2018). A state-of-the-art survey of malware detection approaches using data mining techniques. Human-Centric Computing and Information Sciences, 8(1). https://doi.org/10.1186/s13673-018-0125-x
85. Sparks, S., & Butler, J. (2005). Sherri Sparks Jamie Butler.
86. Stopel, D., Boger, Z., Moskovitch, R., Shahar, Y., & Elovici, Y. (2006). Application of artificial neural networks techniques to computer worm detection. IEEE International Conference on Neural Networks - Conference Proceedings, (January), 2362–2369. https://doi.org/10.1109/ijcnn.2006.247059
87. Sun, X., Zhang, P., Liu, J. K., Yu, J., & Xie, W. (2020). Private Machine Learning Classification Based on Fully Homomorphic Encryption. IEEE Transactions on Emerging Topics in Computing, 8(2), 352–364. https://doi.org/10.1109/TETC.2018.2794611
88. Sundeep Desai, S., Varghese, V., & Nene, M. J. (2020). Controller area network for battlefield-of-things. In Lecture Notes in Electrical Engineering (Vol. 612). https://doi.org/10.1007/978-981-15-0372-6_16
89. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2014). Intriguing properties of neural networks. 2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings, 1–10.
90. Tan, Z., Jamdagni, A., He, X., Nanda, P., & Liu, R. P. (2014). A system for denial-of-service attack detection based on multivariate correlation analysis. IEEE Transactions on Parallel and Distributed Systems, 25(2), 447–456. https://doi.org/10.1109/TPDS.2013.146
91. Tapas, N., Merlino, G., & Longo, F. (2018). Blockchain-Based IoT-cloud authorization and delegation. Proceedings - 2018 IEEE International Conference on Smart Computing, SMARTCOMP 2018, 411–416. https://doi.org/10.1109/SMARTCOMP.2018.00038
92. Tian, Z., Luo, C., Qiu, J., Du, X., & Guizani, M. (2020). A Distributed Deep Learning System for Web Attack Detection on Edge Devices. IEEE Transactions on Industrial Informatics, 16(3), 1963–1971. https://doi.org/10.1109/TII.2019.2938778
93. Tobiyama, S., Yamaguchi, Y., Shimada, H., Ikuse, T., & Yagi, T. (2016). Malware Detection with Deep Neural Network Using Process Behavior. Proceedings - International Computer Software and Applications Conference, 2, 577–582. https://doi.org/10.1109/COMPSAC.2016.151
94. Tselios, C., Politis, I., & Kotsopoulos, S. (2017). Enhancing SDN security for iot-related deployments through blockchain. 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks, NFV-SDN 2017, 2017-Janua, 303–308. https://doi.org/10.1109/NFV-SDN.2017.8169860
95. Typel, S., & Baur, G. (2003). Theory of the Trojan-Horse method. Annals of Physics, 305(2), 228–265. https://doi.org/10.1016/S0003-4916(03)00060-5
96. Ucci, D., Aniello, L., & Baldoni, R. (2019a). Survey of machine learning techniques for malware analysis. Computers and Security,

81, 123–147. https://doi.org/10.1016/j.cose.2018.11.001

97. Ucci, D., Aniello, L., & Baldoni, R. (2019b). Survey of machine learning techniques for malware analysis. Computers and Security, 81, 123–147. https://doi.org/10.1016/j.cose.2018.11.001
98. Ventura, D., Casado-Mansilla, D., López-de-Armentia, J., Garaizar, P., López-de-Ipiña, D., & Catania, V. (2014). ARIIMA: A real IoT implementation of a machine-learning architecture for reducing energy consumption. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 8867, 444–451. https://doi.org/10.1007/978-3-319-13102-3_72
99. Waheed, N., He, X., Ikram, M., Usman, M., Hashmi, S. S., & Usman, M. (2021). Security and Privacy in IoT Using Machine Learning and Blockchain: Threats and Countermeasures. ACM Computing Surveys, 53(6). https://doi.org/10.1145/3417987
100. Warf, B. (2018). Adware. The SAGE Encyclopedia of the Internet. https://doi.org/10.4135/9781473960367.n7
101. Wazzan, M., Algazzawi, D., Bamasaq, O., & Albeshri, A. (2021). applied sciences Internet of Things Botnet Detection Approaches : Analysis and Recommendations for Future Research.
102. Wei, L., Luo, W., Weng, J., Zhong, Y., Zhang, X., & Yan, Z. (2017). SPECIAL SECTION ON INTERNET-OF-THINGS (IOT) BIG DATA TRUST MANAGEMENT Machine Learning-Based Malicious Application Detection of Android. 5, 25591–25601.
103. Weng, J., Weng, J., Zhang, J., Li, M., Zhang, Y., & Luo, W. (2019). DeepChain: Auditable and Privacy-Preserving Deep Learning with Blockchain-based Incentive. IEEE Transactions on Dependable and Secure Computing, PP(8), 1–1. https://doi.org/10.1109/tdsc.2019.2952332
104. Williams C, Chaturvedi R, C. K. (n.d.). No TitleWilliams C, Chaturvedi R, Chakravarthy K Cybersecurity Risks in a Pandemic J Med Internet Res 2020;22(9):e23692 URL: https://www.jmir.org/2020/9/e23692 DOI: 10.2196/23692. Retrieved from https://www.jmir.org/2020/9/e23692, URL:%0A10.2196/23692, DOI:
105. Xiao, L., Li, Y., Han, G., Liu, G., & Zhuang, W. (2016). PHY-Layer Spoofing Detection with Reinforcement Learning in Wireless Networks. IEEE Transactions on Vehicular Technology, 65(12), 10037–10047. https://doi.org/10.1109/TVT.2016.2524258
106. Xiao, L., Li, Y., Liu, G., Li, Q., & Zhuang, W. (2015). Spoofing detection with reinforcement learning in wireless networks. 2015 IEEE Global Communications Conference, GLOBECOM 2015. https://doi.org/10.1109/GLOCOM.2014.7417078
107. Xiao, L., Wan, X., Lu, X., Zhang, Y., & Wu, D. (2018). IoT Security Techniques Based on Machine Learning. 1–20. Retrieved from http://arxiv.org/abs/1801.06275
108. Xue, R., Wang, L., & Chen, J. (2011). Using the IOT to construct ubiquitous learning environment. 2011 2nd International Conference on Mechanic Automation and Control Engineering, MACE 2011 - Proceedings, 7878–7880. https://doi.org/10.1109/MACE.2011.5988881
109. Yuen, K. K. F. (2019). Towards a Cybersecurity Investment Assessment method using Primitive Cognitive Network Process. 1st International Conference on Artificial Intelligence in Information and Communication, ICAIIC 2019, 68–71. https://doi.org/10.1109/ICAIIC.2019.8668842
110. Zewude, B. T., & Ashine, K. M. (2016). Binary Logistic Regression Analysis in Assessment and Identifying Factors That Influence Students ' Academic Achievement : The Case of College of Natural and Computational. Journal of Education and Practice, 7(25), 1–6. Retrieved from https://eric.ed.gov/?id=EJ1115855
111. Zhang, T., & Zhu, Q. (2018). Distributed Privacy-Preserving Collaborative Intrusion Detection Systems for VANETs. IEEE Transactions on Signal and Information Processing over Networks, 4(1), 148–161. https://doi.org/10.1109/TSIPN.2018.2801622
112. Zhu, H., Liu, X., Lu, R., & Li, H. (2017). Efficient and Privacy-Preserving Online Medical Prediagnosis Framework Using Nonlinear SVM. IEEE Journal of Biomedical and Health Informatics, 21(3), 838–850. https://doi.org/10.1109/JBHI.2016.2548248