Deep learning based automated image translation for analysis and detection of malwares in real time systems

Dr.Manikandan N
Manikandan.n@vit.ac.in

Dr.Ruby D
ruby.d@vit.ac.in

Chebrolu Sai Prasanna Abhinav
abhinavchebrolu@gmail.com

School of Computing Science and Engineering

VIT, Vellore

In this high-tech era, malicious software creates a threat to the protection of information every day. The malware invest risks to the safety of every movement in this technologically advanced world. The malware frequently employed by the hackers are nothing but prior established structures utilized to achieve accountability as required. The computer-based diversified assignments have been influenced by in-depth learning. These comprehensive learning procedures not only resulted in high-speed advancement in the challenge but also excelled in the skills applied manually for most of the procedures. Numerous well-known methods of learning about machines are different from the detailed learning devices prepared with the learning aspects rather than particular assignments of clear procedures. The computers apprehend the assignments in the system efficiently rather than a man does. We introduce an innovative pattern established on prevention from malware intervention and structure with a department depending upon the procedure of Deep Learning to augment efficiency. A thorough analogous examination of innovation proves that our prospective and extensive analysis helps to excel the old model of learning procedures of mechanics. The purpose of innovation will differentiate the kinds of malware and unoffensive files favourably. In our experiment, the recognition and categorization are done with an accurate justifiable time slot, that is, 0.25 seconds.

1. Introduction

Recognizing the harmful malicious software and eradicating the threats caused by the malicious software by applying suitable solutions is the foremost function of a cyber safety officer. this paper deals with the different mechanics and ideas which are applicable to reduce the threats to the software.

1.1 Cyber safety

Inhibiting illegal entry, utilization of resources, exposure, segmentation, alteration, examination, documentation or else abolishing the details is the fundamental liability of cyber safety. The data can be either substantial or else computerized. The word data is not a restricted one. It can be anything such as personal information, our account on the social platform, our mobile phones, our biological identifications and so on. Hence, Cyber safety encompasses a vast arena such as internet social platforms, digital forensics, mobile computing, morse coding and so on. The three main goals of Cyber Safety, generally known as CIA, are Confidentiality, integrity, and availability.

Confidentiality

Securing the data from hackers, procedures and unofficial persons. To make it clear, if one maintains a password for his Facebook and by chance if anyone happens to note it down, then the particular password of the respective Facebook account is no longer personal, as it has been intruded.

Integrity

Maintaining the complete data precisely is known as integrity. To elaborate, the data can no longer be rearranged unofficially. For instance, if a person's service is terminated, his complete details in all departments should be revised till the day he worked to provide his current position. So that the full details regarding the employee are exhaustive and precise. Moreover, only approved personnel can edit the staff details.

Availability

Making the saved data accessible whenever required is called availability. To elucidate, if an employer's leave account is needed to verify, whether he has availed the allotted days of leave or excessive, then it requires a cooperative account from various departments such as network operation, department operations, incident response, and policy management. This accessibility to getting details can be hindered by refusing to accept the intervention of malicious software.

The key to cyber safety is Information Assurance, that is, the Conservation of CIA of information, providing the guarantee that no details will be adjusted on the ground of any demanding situation. The list of situations is not restricted to natural calamities, failure of computer programs, etc., but more than that.

In this way, there is an advancement in the domain of cyber safety has advanced in the last few years. This field has opened out a vast area of expertization together with protection to network and associated groundwork, saving the applications and collection of data, examining the safety, information system auditing, preparation of the progress of business etc.

1.2 Intrusion Detection System (IDS)

An unofficial intrusion is identified and on such occasions, the intrusion is warned by the Intrusion Detection System. This computer program watches a web or process for malicious enterprise or violation of guidelines.

Every illegal undertaking or transgression is generally communicated to the chief or intrinsically gathered employing a security information and event management (SIEM) scheme. This scheme combines the data secured from heterogenous sources and extends caution-reducing methods to discriminate real wrongful ventures from fake warnings.

Undoubtedly, the intrusion detection system controls the network for unauthorized intervention, even then it becomes unavoidable to shun completely from fake warnings. As such, the management should improve their IDS device in the initial installation itself. It implies that the advanced IDS should be able to identify the fake warnings from the usual ones.

Network Intrusion Detection System (NIDS)

The warning note can be sent to the administrator after finding out whether there is any unusual activity. As a trial, a NIDS can be set up on the netmask where fireproofing is positioned to perceive if anybody is attempting to break the firewall.

Host Intrusion Detection System (HIDS)

The input and output files alone are observed by the HIDS and by chance any intrusive is identified would warm the administrator. It at once captures a snapshot of current system files and checks it with the former one. The moment any malpractice is detected in the scientific system files, immediately a warning notice is directed to the administrator to undergo investigation. A model of HIDS management can be perceived on important operative devices, which are scheduled to remain unalterable in their layout.

Protocol-Based Intrusion Detection System (PIDS)

PIDS involves a structure or medium which is as per the rule, fixed at the apex of a server, managing as well as elucidating the rules between the operator and the server. To be clear, it attempts to save the wet server by way of observing the HTTRS rule scheme systematically and approving the HTTP protocol.

As HTTP is a clear text and so before allowing intrusion easily into its web presentation it should be fixed between the device and the HTTP.

Application Protocol-based Intrusion Detection System (APIDS)

APIDS, a structure or media, is usually fixed in between a set of servers. It diagnoses any unofficial hacking utilizing continuous observation and elucidation of the activity based on the application-specific protocols. To illustrate, the structure would observe the SQL rule and explain it to the router as it executes with the computerized information in the web server.

Hybrid Intrusion Detection System (HIDS)

HIDS is created by merging multi methods of the intrusion detection system. Here the functionary or database system is integrated with web details to enhance the total prospect of the system structure. This system is more constructive when juxtaposed with other structures of intrusion systems. The prelude is a sample of Hybrid IDS.

1.3 Detection methods of IDS

Signature-based Method

Signature-based IDS recognizes the interferences on the grounds of the particular device, for instance, the number of bytes or number of I's or number of O'S in the method of transfer. It even functions to identify the malicious intervention by employing the malicious instruction program which is fed already. Signatures are the identified patterns in the IDS.

As the signature instruction is present already in the system it is not difficult for the IDS to spot the intrusion. On the other hand, it becomes hard for IDS to find out the new intrusions as their pattern is new to the system.

Anamoly based method

As the signature-based method fails to identify the new intrusions which grew at a fast pace, the Anamoly-based method was imported. In this approach, the learning of machines is used which results in developing a secured venture model and any intrusion is recognized with the existing program and announced as malicious possessing an advanced universal feature while juxtaposing with the signature-based device as this program can be equipped as per the applications and hardware structure.

The two important guards of a network are firewall and IDS but there is a difference between firewall and IDS. The firewall observes the intrusions superficially alone as such to stop them from doing so. The firewalls can confine the intrusions between networks to forbid attack by malware but if the attack occurs within the network, then it is not able to alert. Whereas an IDS identifies the unauthorized intrusion, the moment it occurs and alarms immediately.

1.4 Malwares and its types

Malware is unauthorized programs to secure data unofficially without the knowledge of the user.

Malicious software consists of computer viruses, worms, Trojan horses, ransomware, spyware and other malicious programs.

Viruses

A virus is a harmful feasible cryptograph linked with the other workable file. When an affected file is opened on various computers it spreads. Viruses can be non-dangerous but alter or remove data when the system is affected. The virus is activated an opening a file. When a program is affected by a virus then in no time it would spread to other files too on the system.

Worms

Worms reproduce on their own on the structure connecting themselves to various files and seeking avenues within the computers, to illustrate, computer network which uses a general depositing for files. The network's speed is gradually decreased by these worms. Normally a virus is activated by a hot program whereas a worm needs no such programs, it can operate by itself. The worms develop over the network immediately after it infects a host.

Spyware

Spyware is employed by a third party to misappropriate data from a computer. The spyware gathers details and sends them to unauthorized persons.

Trojan Horse

This Trojan horse is a kind of harmful code or software that seems to appear as authorized but can possess the computer. That is, it carries out illegal operations in the disguise of a legal operation such as playing online games. There is a difference between Trojan and Virus, as Trojan cannot reproduce and activate on its own but connects itself to functional files like images and audio files.

Logic Bombs

A logic bomb is malware, a fraction of code that is covertly infused onto a computer network operating system or software application. Once it is infused it lies like sleeper cells till a particular condition evolves. The moment it gets the feasible condition, it gets activated and harms a system by damaging data, deleting files or emptying the hard driver. Currently, computer safety experts found out that logic bombs destroy the hardware parts in a computer terminal or server inclusive of cooling fans, disc drivers and current supply.

Ransomware

Ransomware encroaches the complete control of the system as well as the information where the victim is not able to access the data unless he pays a demand. The ransomware enters data unless he pays a demand. The ransomware enters data in a computer with a program which is not known to the user. If the user wants to get back his data, he has to pay the demanded amount to the culprit. The user can get access to his data only after he pays the ransom amount.

Back doors

The back doors find an indirect way to access a system rather than using an authorized key. The main aim of a backdoor is to allow cyber thieves to enter a system in future even if the company has mended the previous defenselessness of the system.

Rootkit

A rootkit redesigns the OS to enter a system unofficially. It helps hackers to enter any system from a far-off distance. Many of the rootkits enjoy the benefit of software defenceless conditions to redesign system files.

Keyloggers

The job of a keylogger is to document whatever the operator types in his computer to procure the important details like passwords and other information and direct them to the keylogging program.

1.5 Deep Learning

Deep learning or in-depth learning is nothing but a method of learning about machines which prepares the computers to execute what man learns naturally, that is, mastering through the model. The important technical knowledge behind the cars driven without drivers, empowering such cars to identify traffic signs, or else to differentiate a pedestrian from a street lamp. Technical science is the pivot of controlling the devices like cellular phones, televisions, hands-free headphones, and so on through voice. This technological advancement is made possible by deep learning. Nowadays learners are driven mostly towards a deep learning approach for positive outcomes. It is because of deep learning everything which was not possible before is made possible and accomplished.

The complete stereotype computer learns to carry out grouping jobs straight away from the images, text or audio.

In-depth learning can accomplish excellence, and perfection sometimes even surpassing manual efficiency. The prototype systems are given training by administering a bigger group of data with tags and a circuit of neurons that has manifold coatings.

1.6 Convolution neural network (CNN)

CNN is a sort of profound neural network, frequently used to examine visual imagery, and is involved in deep learning. That is to say, when we assume the neural network, we imagine a composition of two linear functions, whereas in actuality it is not the case with a convolution network. A specific strategy is employed in convolution. In other words, a convolution in mathematics is saying about the exercise of two functions that brings out a third function which demonstrates the way how the configuration of one is altered by the other.

The main idea is that the function of the convolution network is to lessen the picture into a shape that makes the process free from troubles exclusively sustaining the aspects which are important to get a good prognosis.

1.7 A magic to picture transformation in representations

Each hexadecimal digit [2,3] constitutes four binary digits, and the main use of a hexadecimal script is an amiable description of binary-coded values in calculating as well as technological devices. One hexadecimal digit represents a nibble, which is half of an octet or byte (8 bits). In particular, the values of byte start from 0 to ‘255 (decimal), on the other hand, it can easily be demonstrated as two hexadecimal digits in the span of 00 to FF. The computer memory labels are usually represented by hexadecimal. There are many benefits to hexadecimal values.

The job of copying and pasting from the image editor is made easier by the Hex values. Generally, as it has only 6-digit numbers it becomes more compact and stored in memory effortlessly, for example., #fffvsrgb (255, 255, 255). Few keys are needed to convey the same number. Moreover, Hex is to a great extent network-friendly.

2. Review of Literature

Several analyses have been undergone in the field of information and cyber safety. The applicable research works of our area of research are quoted here to highlight the gap in the analytical areas.

Alzaylace, M. K., Yerima, S. Y., & Sezer, S. (2020) A structure of in-depth learning to identify hostile Android Applications utilizing dynamic study employing input generation protocol. More than 30,00 applications are used on real devices and experiments were conducted. Besides, many types of research were undertaken to juxtapose the recognizing function and cryptograph exposure of the active input prompting approach with the normally utilised displaced method using the deep learning structure. The current study discloses that DC-Droid can perform up to the 97.8 identification range (with active and fixed aspects).

Kim t. Kang, B., Rho, M., Sezer, S., & Im, E.G. (2018) The figure of malicious software is on aggressive growth. As android gadgets are exceptionally well-received, they are aimed at by this malicious software. The current paper suggests a new structure for android malware identification. This said structure employs diversified sorts of aspects to indicate the characteristics of android applications from diverse features, and the traits are modified by utilizing the existence-based or similarity-based innovative procedure for productive aspect depiction on malware identification. In addition, a multi-modal deep learning procedure is outlined that can be utilized as malware deep learning to be employed in android malware detection.

Rathore, H., Agarwal, S., Sahay, S. K., & Sewak, M. (2018, December) The modern researches prove that of late, analysts and anti-virus bodies have begun to administer learning of machines and in-depth learning procedures for analyzing and identifying malware. The authors have employed opcode frequency as a prominent attributed vector and exercised individual learning along with guided learning for malware categorization. The target of the current work is on recognizing malware with (1) different machine learning algorithms and (2) in-depth learning specimens. The outcomes prove that random forest outperforms Deep Neural Network with opcode frequency as a highlight.

Cakir B., & Dogdu, E. (2018, March) Of late several methods of learning about machines are employed to identify the malicious software attack. But now for better accomplishment deep learning is utilized. These deep learning paradigms are proven to be functioning efficiently in the examination of lengthy arrangements of system calls. In the present paper, that is, the schematic deep-learning-based aspect withdrawal method (word 2 sec) is utilized for constituting any specified malware based on its opcodes. The gradient Boosting program is employed for the categorization function. The k-fold cross-validation is used to substantiate the program functioning without forgoing an authentic break. Assessment outcomes exhibit up to 96% precision including a less number of selected information.

K. He and D. –S. Kim (2019) K. He et al examine the proficiency of Convolutional Neural Networks (CNN) Versus superfluous API injection. They have devised a malware identification program that changes malware files into pictorial representations and categorizes the pictorial characterization with CNN. The CNN is applied with spatial pyramid pooling layers (SPP) to manage different sizes of input. They have calculated the function of their system on the two data, that is, Unvarying and hostile, with redundant API injected. The outcomes prove that neutral SPP application is inappropriate owing to RAM limitation and grayscale imaging is beneficial against superfluous API injection.

Kim, J. Y., Bu, S. J., & Cho, S. B. (2017, November) Kim et. Al have transmitted hostile networks (TGAN) for categorisation and identification of the zero-day attack. As the GAN is not well constructed in tutorial procedure practising GAN prior with an autoencoder system is suggested. They have studied the detector and the function pattern of malware utilizing the t-SNE algorithm. The suggested prototype acquires the best function juxtaposed with the traditional program of learning about machines.

Venkatraman, S., Alazab, M., & Vinayakumar, (2019) Even though a good deal of malware identification programs are possible, in this data-demanding world to meet the scale and intricacy novel procedures are needed. The authors have suggested a new and comprehensive hybrid in-depth learning and envisioning method for the successful recognition of malware. The present paper has dual objectives:

1. To introduce the utilization of image-based techniques for recognizing the sceptical performance of systems and

2. To suggest and analyze the implementation of hybrid image-based methods with deep learning designs for the successful recognition of malicious software.

Yuxin, D., & Siyi, Z. (2019) Yuxin et. Al place in proximity the function of DBNs alongside the three baseline malware detection prototypes, which make use of support vector machines, decision trees and the K-nearest –neighbour process as classifiers. The studies illustrate that the DIBN paradigm offers more perfect identification rather than the baseline model. When extra anonymous data are utilized for DBN before coaching, the DBN function exceeds the other detection programs.

Xiao, F., Lin, Z., Sun, Y., & Ma., Y. (2019) Safety is not the problem of one device. A lot of intrusions which threatened the conventional computers in the IoT domain may threaten the other IoT devices. The present study suggests a method to safeguard IoT devices from being intruded on by a local computer.

Sewak, M., Sahay, S. K., Rathore, H. (2018, June) Sewak et. Al examined and differentiated one of the Deep Learning Architecture known as Deep Neural Network (DNN) with the standard Random Forest (RF) the programs for learning of machines to recognize the malicious software. They have analyzed the function of standard RF and DNN together with 2, 4 & 7 layers construction with four dissimilar aspects group, and concluded that whatever may be the aspects inputs, the standard RF perfection surpasses the DNN.

Ye, Y., Chen, L., Hou, S., Hardy, W., & Li, X. (2018) Ye, Y. et al conducted exhaustive research on an actual and substantial file collection from Comodo Cloud Security Center to differentiate the numerous malware detection methods. The reassuring study outcomes illustrate that the author's suggested deep learning design can develop the total performance in identifying malware additionally concerning the conventional in-depth learning methods, deep learning procedures with similar structures and other anti-malware detectors. The suggested diversified deep learning structure too can be promptly implemented on other malware spotting functions.

Kim, J. Y., Bu, S. J., & Cho, S. B. (2018) Kim, J. Y. et al suggests a new procedure known as transferred deep-convolutional generative adversarial network (TDC gain), which creates duplicate malware and assimilates to discriminate the duplicate malware from the authentic ones. The data created from an indiscriminate allocation seems to be alike but is not similar to the authentic data: it comprises adapted aspects in contrast to the authentic data. TDC gain accomplishes 95.74% categorizing perfection which proves to be better than that of other prototypes and escalates the learning potential.

Yeo, M., Koo, T., Yoon, Y., Hwang, T., Ryu, J., Song, J., & Park, C. (2018, January) Yeo, M., et al have suggested a better powerful as well perfect malware recognition, as it utilizes 35 various aspects drawn out from file course, rather port number and protocols. Stratosphere IPS project data were utilized for assessment, whereupon nine various public malware files and standard condition files in an uncorrupted framework were transformed into flow data with Netmate and the 35 aspects were drawn out from the flow data. CNN, multi-layer perception (MLP), support vector machine (SVM), and random forest (RF) were tried for categorization, which manifested less than 85% perfection, exactness and revocation for the entire types employing CNN and RF.

Kan, Z., Wamg, Xu, G., Guo, Y., & Chen, X. (2018, July) The conventional methods of identifying malicious software lack success to satisfy the demands of recognizing polymorphic and fresh specimens. The available recognition methods developed on network functions are greater, nonetheless use considerably more time in both the aspects, removal and tutoring. The authors of the current paper suggest a weightless PC malware recognition structure that is built on a profound convolutional neural network (CNN).

Ni, S., Qian, Q., & Zhang, R. (2018) At this instant, one of the severe dangers to cyber safety is malicious software. The present paper suggests a malware grouping program which utilizes fixed traits known as MCSC (Malware Classification using Sin Hash and CNN) that transforms the deconstructed malware cyphers into grey representation depending upon Sim Hash and afterwards recognizes their groups through the convolutional neural network. In the course of this procedure, a few programs like multi-hash, important

block option and bilinear interpolation are utilized to enhance the function. The study outcomes present that MCSC is most successful for malware category division, yet for those irregularly allotted specimens. The precision of grouping at its best is 99.260% and on average 98.862% on a malicious software dataset of around 10,805 specimens which is more than that of the other correlated programs. Besides MCSC, it simply requires 1.41s to identify a fresh specimen, which can satisfy the necessities in the majority of the functional applications.

Abdelsalam, M., Krishnan, R.,Huang, Y., & sandhu, R. (2018, July) CNN (Convulutional Neural Network) is a deep learning method. At first, a standard 2d CNN is utilized by providing tutoring on metadata on hand for every single procedure in the virtual machine (VM) acquired with the help of the hypervisor. By utilizing a new 3dCNN the accuracy of the CNN classifier is improved (where input is an accumulation of specimens on a regular interval), which facilitates considerably to declare the mixed-up specimens at the time of data collection and tutoring. Abdelsalam et al's analyses were done on the gathered data employing utilizing different malware (especially Trojans & Rootkits) on VMs. They have chosen the malware blindly for their study.

Al-Dujaili, A., Huang, A., Hemberg, E., & O'Reilly, U. M. (2018, May) A significant additional issue of malicious software is such that hostile specimens must be created to conserve their malicious performance. The authors of the current paper initiated procedures competent in creating procedures that applicably preserved hostile malware specimens in the binary field. They have included the hostile specimens in the tutoring prototypes that are strong by utilizing saddle-point formulation. The authors have endeavoured to find these methods because of the influence of the like approaches for the discrete such as binary, domain which exemplifies the malware.

Aslan, O. A., & Samet, R. (2020) For unidentified as well as intricate malware, behaviour-based, model-checking based & cloud-based methods work efficiently. Besides deep learning-based, mobile devices–based and IoT-based methods too arise to locate some parts of recognized and unrecognized malicious software. Nonetheless, there is no single method which can recognize all malicious software randomly. This indicates that developing a productive approach to identifying malicious software is certainly a demanding mission and also there is an enormous opening for new research and approaches. The authors of the current paper establish a comprehensive analysis of the methods of identifying malicious software and the new recognition approaches which utilize these methods.

Mahdavifar, S., & Ghorbani, A. A. (2019) The present paper suggests a common DL structure for internet safety and elucidates the four main components comprised of it. Then the papers associated with it are condensed and studied in terms of the core area, methodology, relevancy of the prototype and the lacking aspects. Lastly, findings and further research are outlined.

Vinay Kumar, R., Soman, K.P., Poornachandran, P., Alazab, m., & Jolfaei, A. (2019) The scheme scrutinizes the names of the domain and classifies them by employing statistical aspects, which are outlined clearly through deep learning programs. The scheme is analyzed and adopted in their laboratory. The study outcomes indicate the benefit of the suggested structure and prove that the recommended approach possesses prominent efficiency and lesser fake-assured grades. The recommended structure is an

the uncomplicated structure that has baser masterable instructions in contrary to other character-based, short-text grouping paradigms.

Vinay Kumar, r., Alzab, M., soman, K. P., Poornachandran, P., Al-Nemart, A.,& Vwenkatraman, S. (2019) On account of the powerful character of the malicious software along with constantly modifying the intrusion approaches, the openly employable malware datasets are to be modified methodically and tested. This paper investigated a deep neural network (DNN), a kind of deep learning design to improve an adaptable and powerful IDS to identify and group unexpected and doubtful malware. The constant modification in network conduct and speedy development of intrusions assess different datasets which have been created during the past years using fixed and effective methods. The present study helps to recognize the best program which can function successfully in recognizing future malicious software intrusions.

HaddadPajouh, H., Dehghantanha, A., Khatami, R., & choo, K.K. R. (2018) Haddad Pajouh et al's study tries to put forth that there is no method prevalent which utilizes deep learning for the identification of IoT malware. Thus, this paper recommends using RNN to identify IoT malware by scrutinizing IOT applications OpCodes.

Table 1. Various approaches and restrictions

S.No	Approaches / Program used	Restrictions
1.	The identifiers of malicious software utilize two inputs. First to identify the hostility of the package and the next is the package itself to be examined. The methods for identifying malware can be classified into two groups: anemology-based detection signature-based detection	The hostile approaches produce fake alerts of intrusions and evaluate the real malicious attack as natural.
2.	Employ naturally established approaches to analyze a computer program which is not strong enough to be modified by the intruder. The effective recognition of malware challenges to compete with the deception originated by the hackers to escape from the antivirus detection software. Approaches utilized: Modelling assembling, Pruning, Predictive, adjustment	Since this approach employs only nominal integration, the exactness of the points got in the highly adjusted way did not enhance remarkably.
3.	The structure recommended employs a malware grouping program that depends on fixed aspects called MCSC (Malware Classification Using SimHash and CNN). Schemes graded malware codes into SimHash-	The practice of side-by-side computers such as GPUs may accelerate the grouping procedure.

	based images and find out on-computer-sensitive groups utilizing CNN.	
4.	This paper analyses MobileMalware detection through Opcode investigation and recommends the optimistic machine learning classifier approach. Many machine learning approaches are chosen depending upon the preceding studies and their function is scrutinized to enhance TRR and FPR.	The size of the specimen for undergoing the study is very small to establish that SVM would be the best Machine Learning technique – only 1000 Malignant and 500 benign specimens were adopted.
5.	Assess old MLAs and deep learning resources for a system which is uncertain for computer, categorization and grouping employing several public and private data sets. It eliminates the entire site prejudice using test analysis by detecting public and private data from training and model testing.	It recommends new image processing methods with suitable MLA specifications to bring about a successful model for zero-day malware recognition.

3. System model

A convolutional neural network or CNN is supplementary to the traditional feed network (TFN), which is largely in the field of image processing. In this endeavour, the CNN network is developed on a convolution. ID layer, which associates the ID layer with a completely combined layer. CNN network can comprise a simply multi-layer of convolution ID, which has a connected layer and ID layer.

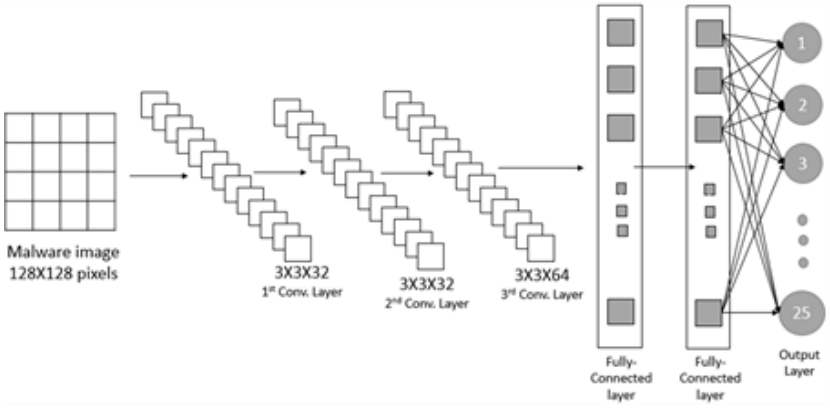
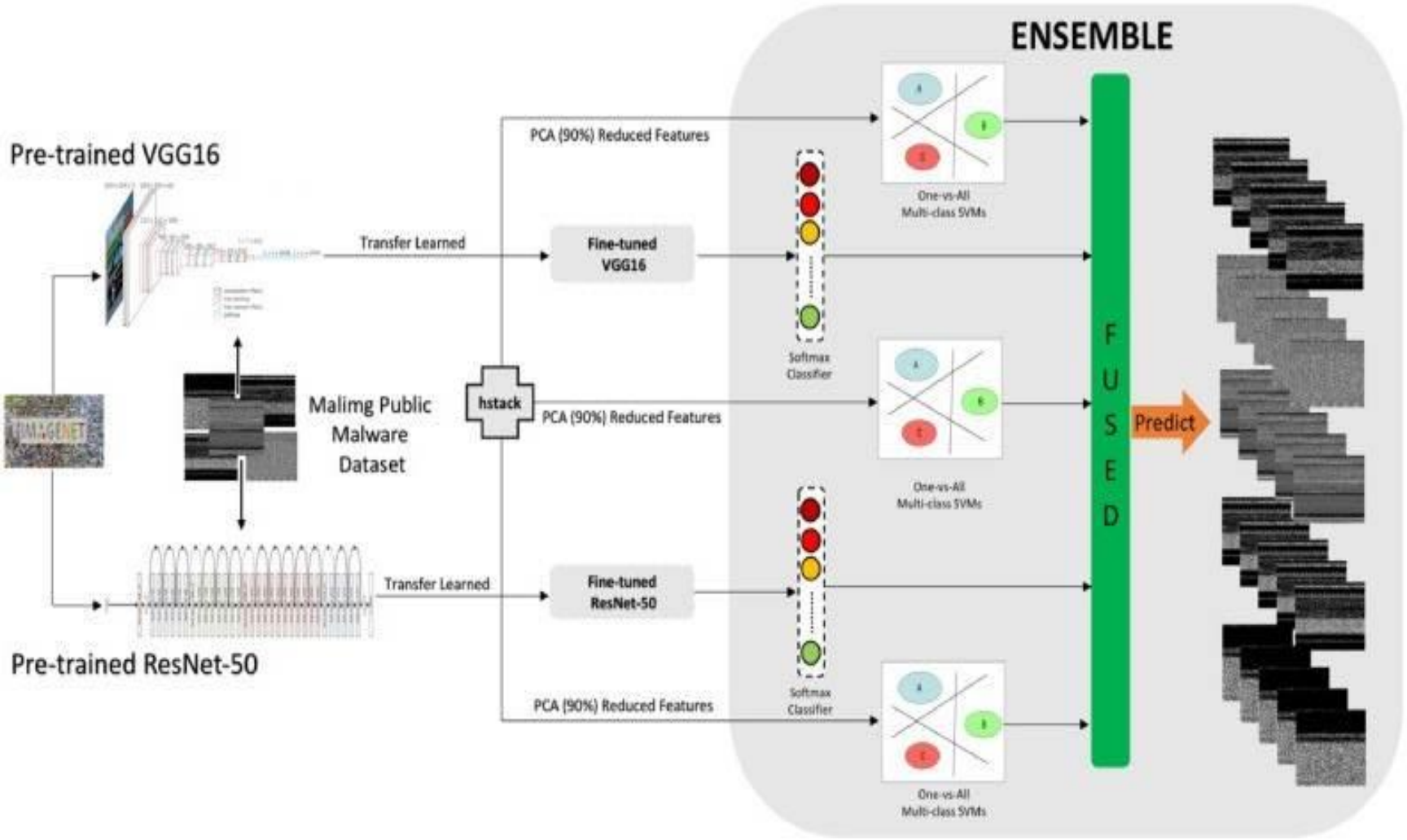
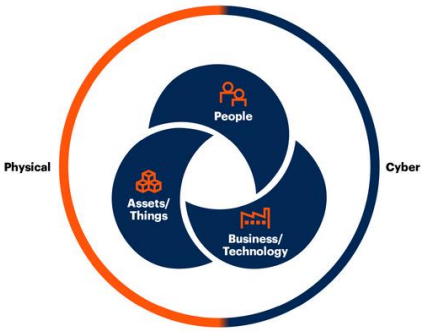
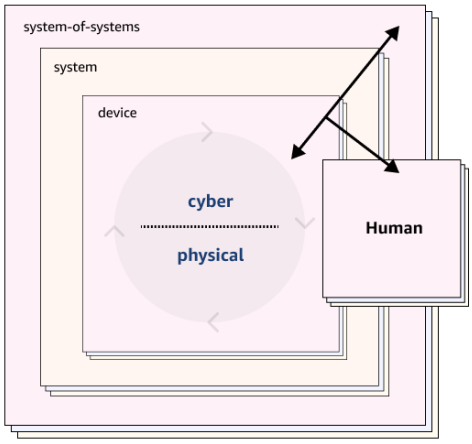
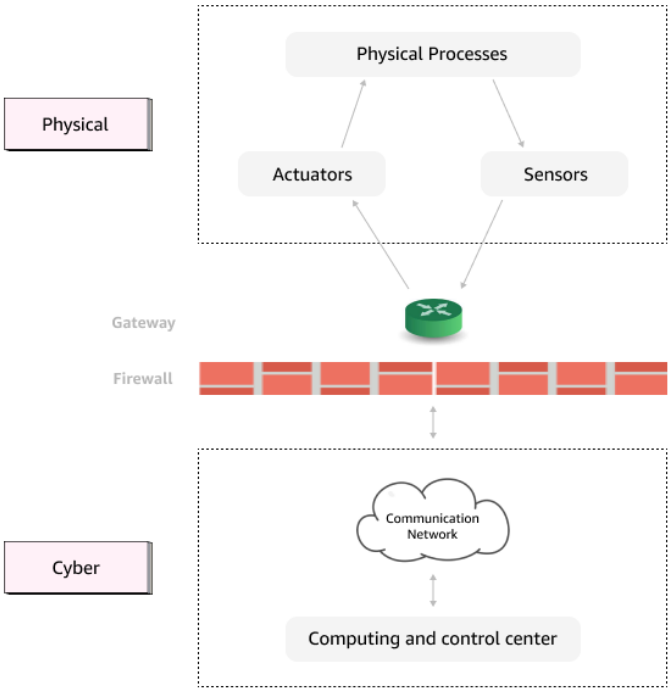


Fig. 1 CNN-based image detection system


ENSEMBLE
Pre-trained VGG16
Transfer Learned
Fine-tuned VGG16
PCA (90%) Reduced Features
Softmax Classifier
One-vs-All Multi-class SVMs
Malimg Public Malware Dataset
hstack
PCA (90%) Reduced Features
One-vs-All Multi-class SVMs
F U S E D
Predict
Transfer Learned
Fine-tuned ResNet-50
Softmax Classifier
PCA (90%) Reduced Features
One-vs-All Multi-class SVMs
Pre-trained ResNet-50


People
Physical
Cyber
Assets/
Things
Business/
Technology


system-of-systems
system
device
cyber
physical
Human


Physical
Physical Processes
Actuators
Sensors
Gateway
Firewall
Cyber
Communication
Network
Computing and control center

TCP Protocol

TCP (Transmission Control Protocol) is one of the Internet protocol suite's primary protocols. It is located between the Application and Network Layers and is used to provide dependable delivery services. It is a connection-oriented communication protocol that aids in the exchange of messages between devices on a network.

Train the model

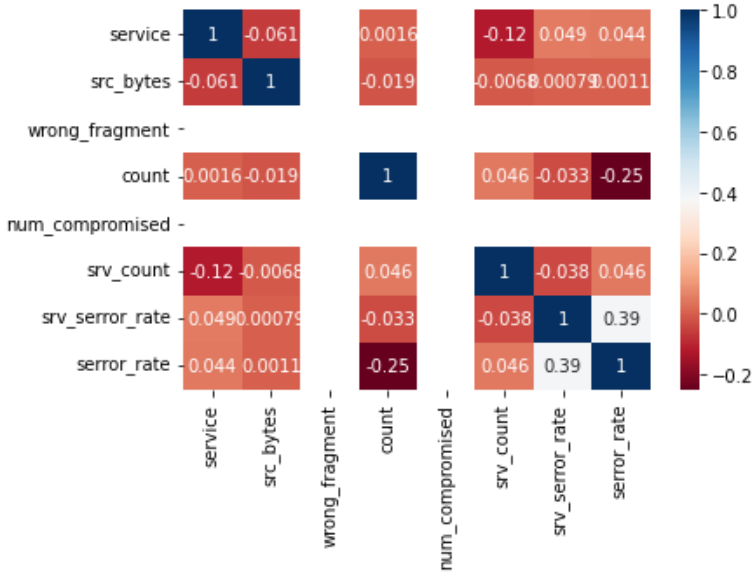
```
PS C:\Users\Sajna\OneDrive\Desktop\TECH\ISAA Project\ISSA-main\code> python train.py tcp_syn 0
Data preprocessing done.
The model has been fit.
Save the fitted model?(y/n):
y
PS C:\Users\Sajna\OneDrive\Desktop\TECH\ISAA Project\ISSA-main\code> python train.py tcp_syn 1
Data preprocessing done.
The model has been fit.
Save the fitted model?(y/n):
y
PS C:\Users\Sajna\OneDrive\Desktop\TECH\ISAA Project\ISSA-main\code>
```

Test The Model

We may use the model to identify the anomaly once it has been stored. You must verify that you input the correct number of arguments when testing the class. . This will test the specified parameters on the ICMP pre-trained model.

Type**"python test.py tcp_syn -1.5 1.0 2.0 1.0 1.0"** and **"python test.py tcp_syn -1.5 1.0 2.0 30.0 1.0"** in the CMDAs an input, each protocol now has a separate parameter length. While the ICMP protocol requires 5 float values, other protocols may need other inputs.

```
PS C:\Users\Sajna\OneDrive\Desktop\TECH\ISAA Project\ISSA-main\code> python test.py tcp_syn -1.5 1.0 2.0 1.0 1.0
[0]
PS C:\Users\Sajna\OneDrive\Desktop\TECH\ISAA Project\ISSA-main\code> python test.py tcp_syn -1.5 1.0 2.0 30.0 1.0
[0]
```




```
136558 : service       -1.5
count          1.0
srv_count      1.0
src_bytes      0.0
serror_rate    1.0
Name: 136558, dtype: float64
136559 : service       -1.5
count          1.0
srv_count      2.0
src_bytes      0.0
serror_rate    1.0
Name: 136559, dtype: float64
136560 : service       -1.5
count          1.0
srv_count      2.0
src_bytes      0.0
serror_rate    1.0
Name: 136560, dtype: float64
136561 : service       -1.5
count          1.0
srv_count      2.0
src_bytes      0.0
serror_rate    1.0
Name: 136561, dtype: float64
136562 : service       -1.5
count          1.0
srv_count      3.0
src_bytes      0.0
serror_rate    1.0
Name: 136562, dtype: float64
136563 : service       -1.5
count          1.0
srv_count      2.0
src_bytes      0.0
serror_rate    1.0
Name: 136563, dtype: float64
136564 : service       -1.5
count          1.0
```

TCP Attack Plotting

```
Accuracy of the model is:  99.98171512159443
Confusion Matrix:
 [[   2    1]
 [   0 5466]]
Report:
              precision    recall  f1-score   support

           0       1.00      0.67      0.80         3
           1       1.00      1.00      1.00      5466

    accuracy                           1.00      5469
   macro avg       1.00      0.83      0.90      5469
weighted avg       1.00      1.00      1.00      5469


==============***==============
Accuracy of the model is:  99.94514536478333
Confusion Matrix:
 [[   0    3]
 [   0 5466]]
Report:
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         3
           1       1.00      1.00      1.00      5466

    accuracy                           1.00      5469
   macro avg       0.50      0.50      0.50      5469
weighted avg       1.00      1.00      1.00      5469


==============***==============
Accuracy of the model is:  99.98171512159443
Confusion Matrix:
 [[   2    1]
 [   0 5466]]
Report:
              precision    recall  f1-score   support

           0       1.00      0.67      0.80         3
           1       1.00      1.00      1.00      5466

    accuracy                           1.00      5469
   macro avg       1.00      0.83      0.90      5469
weighted avg       1.00      1.00      1.00      5469


==============***==============
```

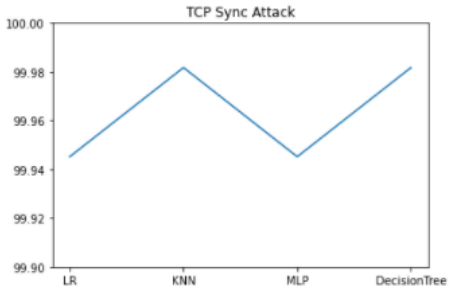


Fig: TCP Attack Accuracy and sync attack graph

UDP Protocol

A Transport Layer protocol is User Datagram Protocol (UDP). UDP is a component of the Internet Protocol suite, sometimes known as the UDP/IP suite. It is an unstable and connectionless protocol, as opposed to TCP. As a result, there is no requirement to connect before data transfer.

Although Transmission Control Protocol (TCP) is the primary transport layer protocol used with the majority of Internet services; it provides guaranteed delivery, dependability, and much more, all of these services come at an additional cost in terms of overhead and delay. UDP enters the picture here. UDP is required for real-time applications such as computer gaming, phone or video communication, and live conference. Because great throughput is required, UDP allows packets to be discarded rather than processing delayed ones. In UDP, there is no error checking.

Train the model

```
PS C:\Users\Sajna\OneDrive\Desktop\TECH\ISAA Project\ISSA-main\code> python train.py udp 0
Data preprocessing done.
The model has been fit.
Save the fitted model?(y/n):
y
PS C:\Users\Sajna\OneDrive\Desktop\TECH\ISAA Project\ISSA-main\code> python train.py udp 1
Data preprocessing done.
The model has been fit.
Save the fitted model?(y/n):
y
PS C:\Users\Sajna\OneDrive\Desktop\TECH\ISAA Project\ISSA-main\code>
```

Test The Model

We may use the model to identify the anomaly once it has been stored. You must verify that you input the correct number of arguments when testing the class. . This will test the specified parameters on the ICMP pre-trained model.

Type "**python test.py UDP 146.0 0.0 105.0 254.0 2.0**" and"**python test.py UDP 45.0 -0.3 45.0 236.0 2.0**" in the CMDAs an input, each protocol now has a separate parameter length. while the ICMP protocol requires 5 float values, other protocols may need other inputs.

```
PS C:\Users\Sajna\OneDrive\Desktop\TECH\ISAA Project\ISSA-main\code> python test.py udp 146.0 0.0 105.0 254.0 2.0
[1]
PS C:\Users\Sajna\OneDrive\Desktop\TECH\ISAA Project\ISSA-main\code> python test.py udp 45.0 -0.3 45.0 236.0 2.0
[1]
```

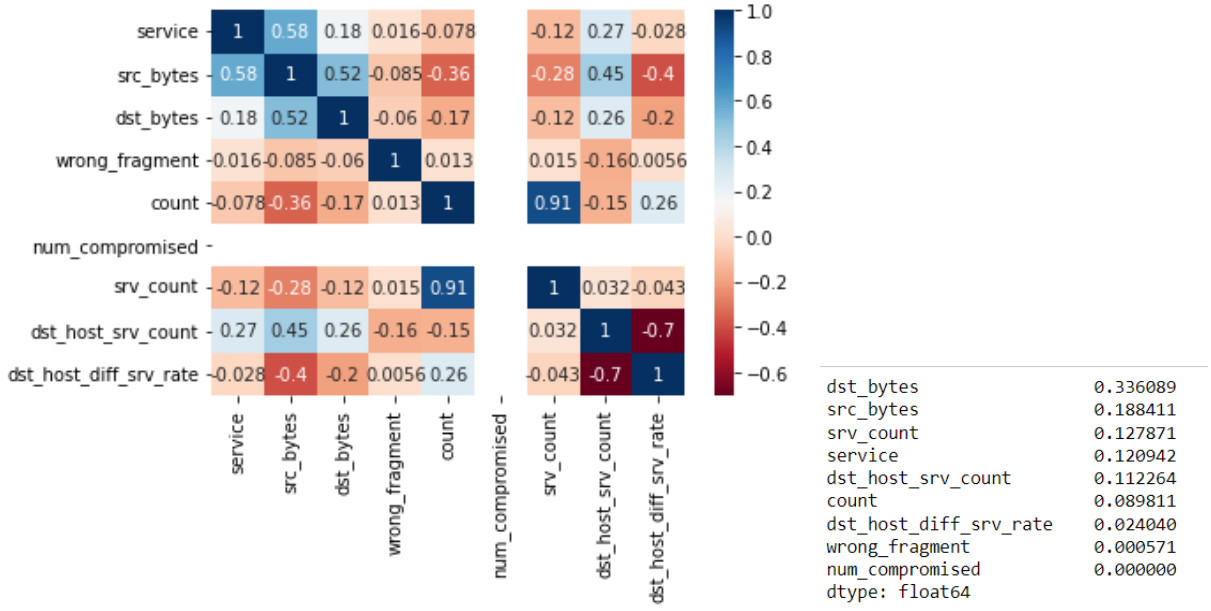


Fig: UDP Attack Plotting and their features

```
Accuracy of the model is:  71.13968293596305
Confusion Matrix:
 [[2787 2070]
 [ 242 2912]]
Report:
              precision    recall  f1-score   support

           0       0.92      0.57      0.71      4857
           1       0.58      0.92      0.72      3154

    accuracy                           0.71      8011
   macro avg       0.75      0.75      0.71      8011
weighted avg       0.79      0.71      0.71      8011
```

```
==============***===============
Accuracy of the model is:  74.54749719136188
Confusion Matrix:
 [[3862  995]
 [1044 2110]]
Report:
              precision    recall  f1-score   support

           0       0.79      0.80      0.79      4857
           1       0.68      0.67      0.67      3154

    accuracy                           0.75      8011
   macro avg       0.73      0.73      0.73      8011
weighted avg       0.74      0.75      0.75      8011
```

```
==============***===============
Accuracy of the model is:  73.93583822244413
Confusion Matrix:
 [[3560 1297]
 [ 791 2363]]
Report:
              precision    recall  f1-score   support

           0       0.82      0.73      0.77      4857
           1       0.65      0.75      0.69      3154

    accuracy                           0.74      8011
   macro avg       0.73      0.74      0.73      8011
weighted avg       0.75      0.74      0.74      8011
```

```
==============***===============
Accuracy of the model is:  77.18137560853826
Confusion Matrix:
 [[3831 1026]
 [ 802 2352]]
Report:
              precision    recall  f1-score   support

           0       0.83      0.79      0.81      4857
           1       0.70      0.75      0.72      3154

    accuracy                           0.77      8011
   macro avg       0.76      0.77      0.76      8011
weighted avg       0.78      0.77      0.77      8011


==============***===============
```

Fig: UDP Attack Accuracies

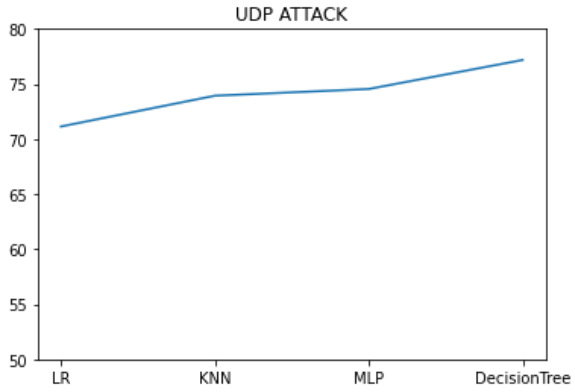


Fig: UDP graph based on Accuracy

ICMP Protocol

Because IP lacks a built-in method for transmitting error and control messages. Error control is provided through the Internet Control Message Protocol (ICMP). It is used for error reporting and management inquiries. It is a supporting protocol used by network devices such as routers to convey error messages and operational information, such as the requested service being unavailable or a host or router being unable to be accessed.

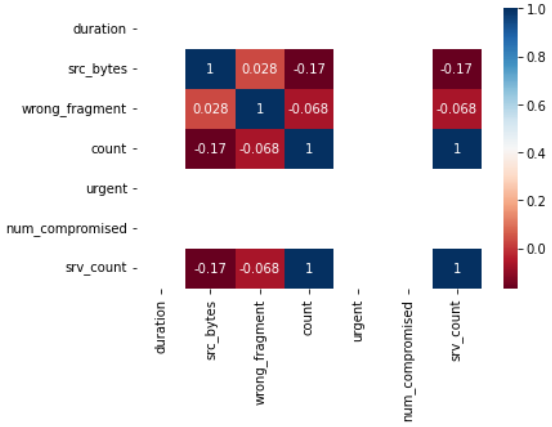


Fig: ICMP Protocol Plotting

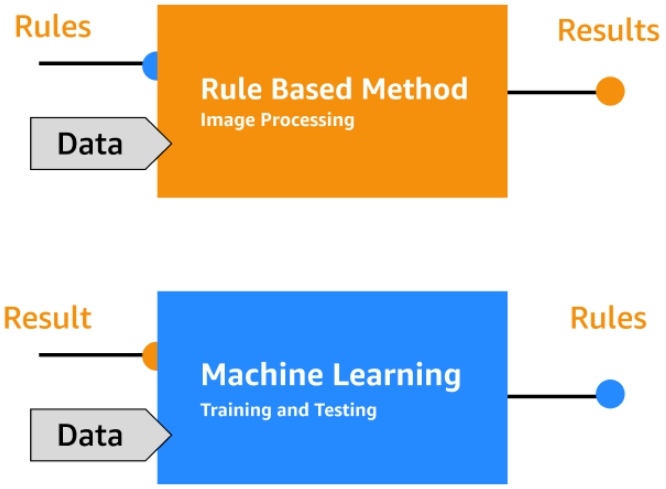
```
PS C:\Users\Sajna\OneDrive\Desktop\TECH\ISAA Project\ISSA-main\code> python train.py icmp 0
Data preprocessing done.
The model has been fit.
Save the fitted model?(y/n):
y
PS C:\Users\Sajna\OneDrive\Desktop\TECH\ISAA Project\ISSA-main\code> python train.py icmp 1
Data preprocessing done.
The model has been fit.
Save the fitted model?(y/n):
y
PS C:\Users\Sajna\OneDrive\Desktop\TECH\ISAA Project\ISSA-main\code>
```

We may use the model to identify the anomaly once it has been stored. You must verify that you input the correct number of arguments when testing the class. This will test the specified parameters on the ICMP pre-trained model.

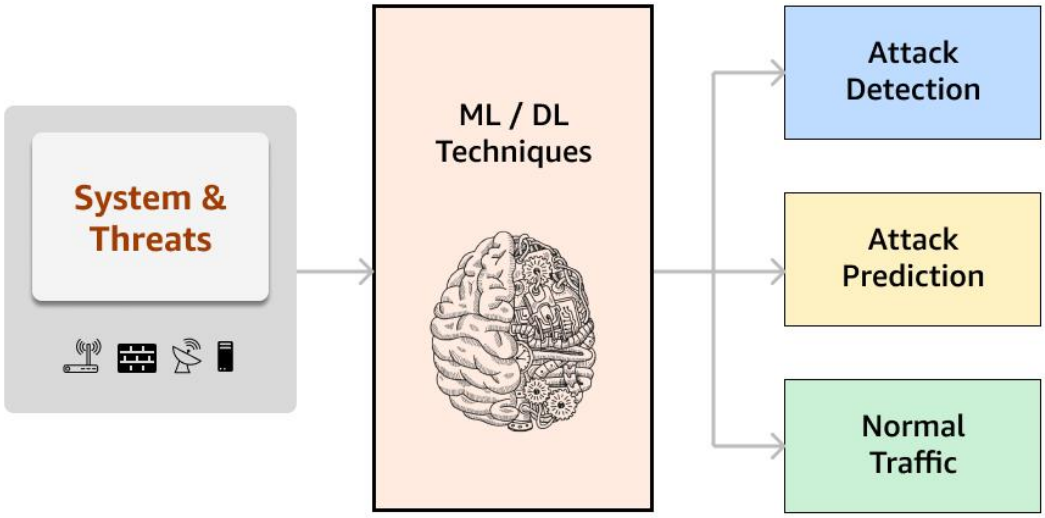
Type **"python test.py ICMP 0.0 0.0 30.0 0.0 1.0 0.0 0.0"** *and* **"python test.py ICMP -0.1 30.0 0.0 2.0 0.0 0.0 2.0"** *in the CMD*. As an input, each protocol now has a separate parameter length. While the ICMP protocol requires 7 float values, other protocols may need other inputs.

```
PS C:\Users\Sajna\OneDrive\Desktop\TECH\ISAA Project\ISSA-main\code> python test.py icmp 0.0 0.0 30.0 0.0 1.0 0.0 0.0
[1]
PS C:\Users\Sajna\OneDrive\Desktop\TECH\ISAA Project\ISSA-main\code> python test.py icmp -0.1 30.0 0.0 2.0 0.0 0.0 2.0
[0]
```

In the ID Convolutional layer, the filtrate slides over ID sequences and removal of total aspects. The components which have been removed from each filter are afterwards classified into novel aspects also called feature maps. The count of filters and sketches are chosen as per the tuning method of the upper ranges. This utilizes the indirect unlock function, ReLU for every single aspect. The size of the positive components is decreased by utilizing an ID layer involving the substantial, meagre or medium association.



Machine Learning entails teaching a machine to perform something (in this case, image processing) by supplying a collection of training data. Machine Learning has models/architectures, loss functions, and a variety of methodologies that may be utilised to discover which provides better image processing.



There is a lot of terminology and technical jargon surrounding AI that is frequently used interchangeably and can be confusing, especially for individuals who do not have a technical background. The essential words defined below are intended to help the average reader comprehend some of the terminology surrounding AI and the debate in this text. This list is neither exhaustive nor technologically exhaustive.

Artificial neural networks (ANNs) are machine learning frameworks that seek to emulate the learning patterns of natural biological neural networks. Biological neural networks function in the sense that dendrites receive inputs that are considered to be presented in the human brain's linked neurones. They generate an output signal based on these inputs, which are sent by an axon to another neurone. We will attempt to replicate this process using Artificial Neural Networks (ANN), which shall be referred to as neural networks from here on. Deep learning is built on neural networks. It is a subset of machine learning that is responsible for some of today's most interesting technical developments!

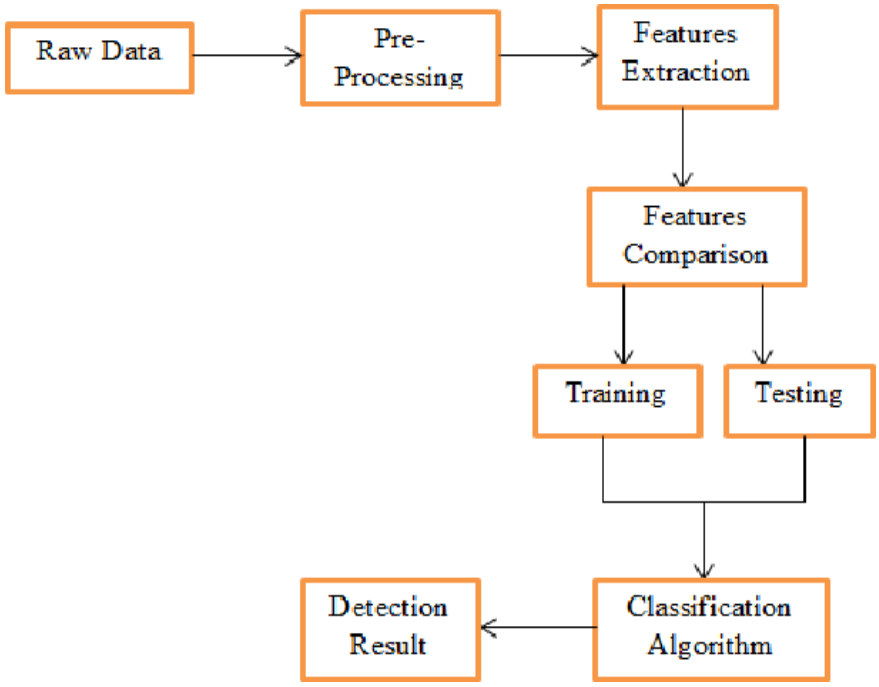
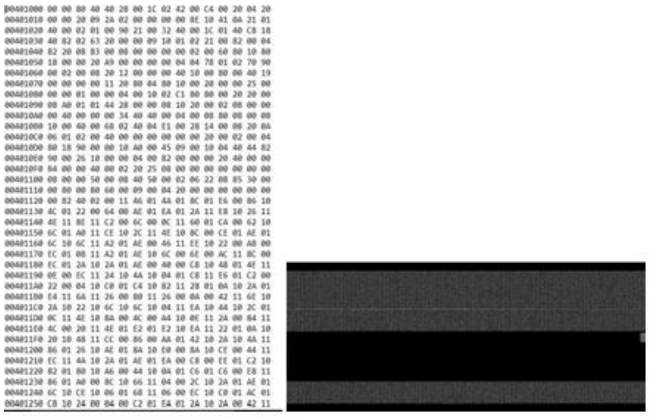


Fig. 3 System flow diagram

The distribution Analysis graph was created from the basic document file. The information **utilized in this scheme has been collected from Kaggle.** It is an enormous document file with a size of 1.3 GB comprising 8404 analyses and 935 approved samples -90% and 10% respectively. The x-axis exhibits the various sorts of malicious software existing in the document file since the y-axis exhibits the number of malware of every kind. After scrutinizing this graph, it can be concluded that Allaple. A is typically identified in the dataset.

A conventional CNN model was created to prepare the baseline scoring of the practice model mentioned in this paper. The model was 2 convolution layered where every layer consisted of a convolutional layer and a max pooling layer. Lastly, it was compressed and passed through a concealed thick layer and subsequently to the output layer with 25 neurons relevant to the number of classes imminent in the labels of the output.



VGG16 (also called Oxford Net) is a pattern of the convolutional neural network that got its name from the Visual Geometry Group from Oxford. VGG-16 is a 16-layer convolutional neural network. The model syncs a group of pre-trained weights to the Image net. The model benefits 92.7% precision of the test over 5 at Image Net, which is a 14 – million-images site with more than 1000 classes. the customary size of the VGG16 model is 224*244.

i. Conversion to Image

Raw data is made of the hexadecimal representation of the binary file of the Malware. First, we convert those files into PNG images as shown in fig****.

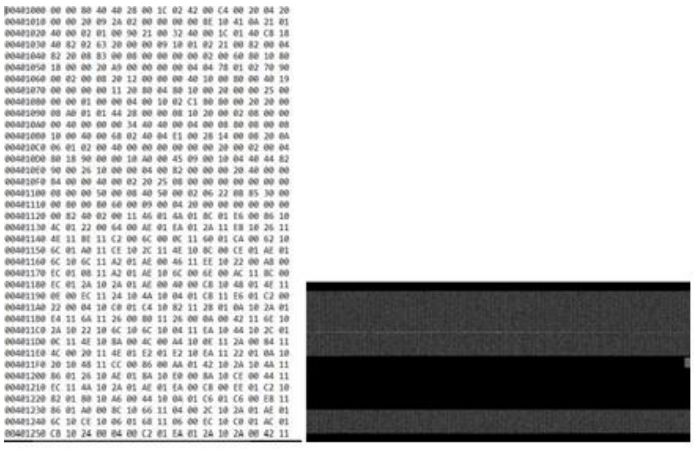


Fig file to image conversion

ii. Perform Exploratory data analysis (EDA)

Since the input data taken is more wage, accomplishing initial examinations on data to realize patterns, find anomalies, examine hypotheses and check expectations with the help of instantaneous statistics and graphical illustrations.

As a result of the EDA process, the outcome of various hypotheses are presented in figure***

Fig. General malware images from hex values

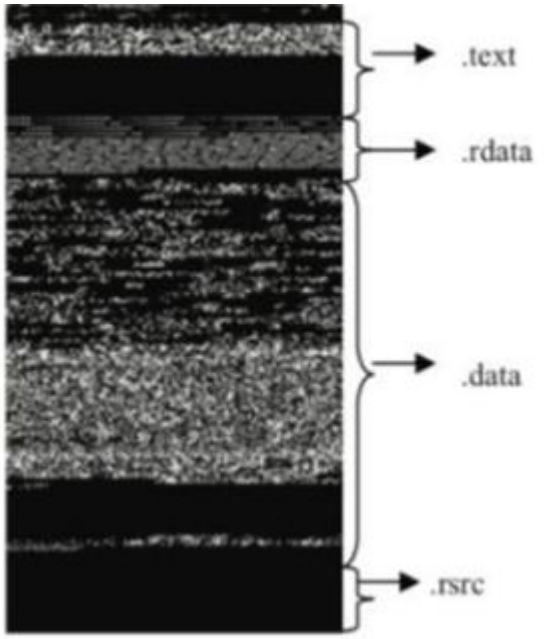


Fig. Distribution of malware information from malware images

The Distribution Analysis graph was formed from the dataset. The dataset used in this project has been taken from Kaggle. It is a huge dataset of 1.3GB in size, containing 8404 test and 935 validation samples - 90% and 10% respectively.

The x-axis shows the different types of malware present in the dataset, whereas, the y-axis shows the number of malware of each type. Analyzing this graph we can infer that Allaple.A. is most commonly found in the dataset.

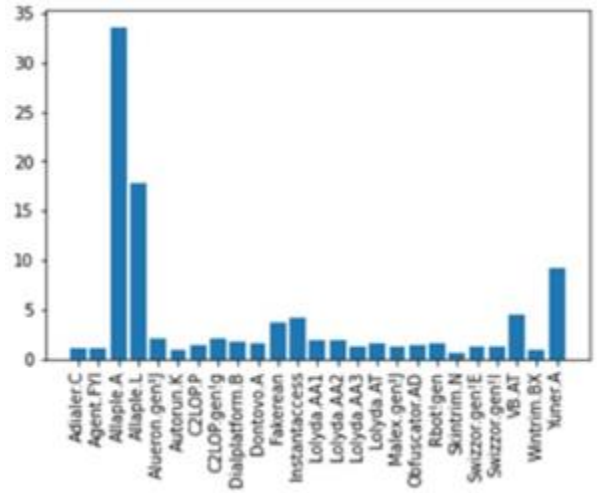


Fig. Distribution of malware images

A custom CNN model was generated to set the baseline scoring of the models to follow in this paper. The model was a 2 convolution layered model where each contained a convolutional layer and a max-pooling layer. Finally, it was flattened and passed through a hidden dense layer and ultimately to the output layer with 25 neurons about the number of classes at hand in the labels of the outputs.

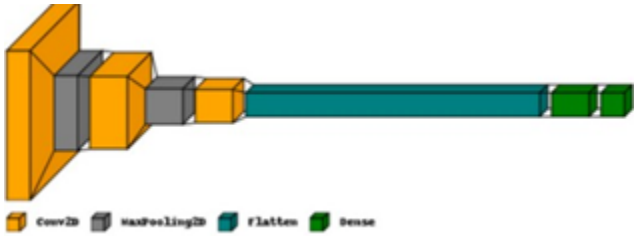


Fig. CNN network model

VGG16 (also known as OxfordNet) is a structure of the neural convolutional neural network named after the Visual Geometry Group from Oxford. VGG-16 is a 16-layer convolutional neural network. The model uploads a set of pre-trained weights to ImageNet. The model gains 92.7% test accuracy over 5 at ImageNet, which is a 14-million-

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Functional)           (None, 512)               14714688
_________________________________________________________________
flatten_1 (Flatten)          (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 25)                12825
=================================================================
Total params: 14,727,513
Trainable params: 14,727,513
Non-trainable params: 0
_________________________________________________________________
```

image site of more than 1000 classes. The default input size for the VGG16 model is 224 x 224 pixels and 3 RGB image channels. It has a 3x3 filter believing layer of layers and a large 2x2 filter layer with 2 rows *(include the parameters in running text or create a table)*

Inception-ResNet-v2 is a neural convolutional network trained with over a million images from the ImageNet website. The network is 164 layers deep and can split images into up to a thousand objects, such as a keyboard, mouse, pencil, and many other animals. As a result, the network has learned to represent the rich features of a variety of images. The network has 299-by-299 image input, and the output is a list of limited class opportunities. *(include the parameters in running text or create a table)*

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
inception_resnet_v2 (Functio (None, 1536)              54336736
_________________________________________________________________
flatten (Flatten)            (None, 1536)              0
_________________________________________________________________
dense (Dense)                (None, 25)                38425
=================================================================
Total params: 54,375,161
Trainable params: 54,314,617
Non-trainable params: 60,544
_________________________________________________________________
```

4. Results and discussion

a. Scoring metric for the models

Each of the models was inspected for the accuracy score and f1 score. The accuracy score determines the correct predictions of the model whereas the F1 scores determine the overall precision and recall for predictions by the model.

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

$$F1\ Score = \frac{TP}{TP + FP}$$

Where
TP
TN
FP
FN

Generating Adversarial Examples

$$Adversarial\ Image\ =\ Original\ Image\ +\ sign(\nabla x\ J(\theta, x, y))$$

The below heat map shows the comparison between actual names of malware and the predicted names. We can see that the diagonal is mostly made of lighter shades, indicating that our proposed model has a high accuracy of above 90%.

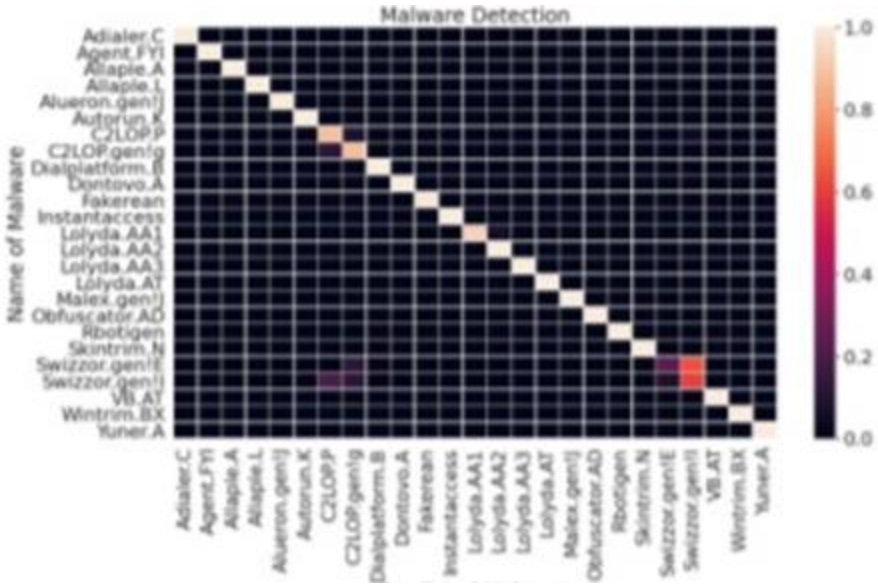


Fig. Test set analysis report

b. Comparing the Convoluted Images Processed by the Model

The CNN model generates 12 convoluted images identifying the most significant part of an image and using that to look for similarities and dissimilarities. Generated convoluted images of similar family malware pointed out a few extremely similar patterns whereas in the case of different families the generated patterns were quite dissimilar comparatively.

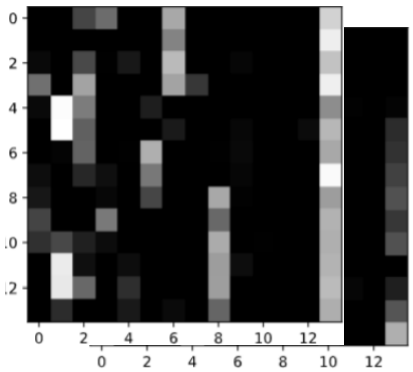


Fig. Convoluted image for malware by Yuner and Adler

To analyze the malicious characteristics we have used a seven-step methodology

1. Read the raw data containing a hexadecimal representation of the binary file.
2. Convert those files into PNG images.
3. Data Analysis to find the number of images in each class and do further preprocessing.
4. Generate standard tensor image data collections from the relevant data index.
5. Generate a CNN-based model and train on the dataset.
6. Model malware generated output in terms of the accuracy of classification using confusion matrix and weighted f1-score.
7. Generate convoluted images to analyze the similarities in the same family malware.

The reason to convert the hexadecimal code of Malware to Images was if the attacker tries to make changes to certain malware hex values converting that malware to an image and using a CNN over it will still be able to tell us about the family of malware with a certain accuracy. And the model succeeded in doing so. We changed certain lines of the bytecode by deleting certain lines, and modifying individual values, and were still able to classify the malware into its family accurately. This method is also extremely fast. This methodology can be very useful in classifying a new malware and then building a solution for it.

In the real-world test scenario, our model successfully discriminated between types of malware and a harmless file. The identification and classification were also done reasonably quick, about 0.25 seconds in our testing. Therefore, this solution could be deployed without having to worry about massive latencies in file transfer

During our literature survey, we read about the different types of models that the papers were using like VGG16, VGG19 and our accuracy of 98.46% turned out to be better than those models but the Custom CNN was not able to outperform InceptionResNet and DenseNet Neural Nets. To compare these methods we use the recorder accuracies.

```
148/148 [==============================] - 20s 136ms/step - loss: 0.0542 - acc: 0.9846
```

These were the recorded accuracies of a few other models on the same dataset from different research papers.

Clearly from the above models and their respective accuracies, our model outperforms highly dense models like Vgg 16, Vgg 19, and Xception

	Model_Name	Accuracy
0	Vgg16	0.952815
1	InceptionResNet	0.972244
2	VGG19	0.339017
3	DenseNet201	0.977002
4	Xception	0.963521

The below graph shows the comparison between accuracy and F1 score between the different types of models that we have worked on. We can infer from the graph that

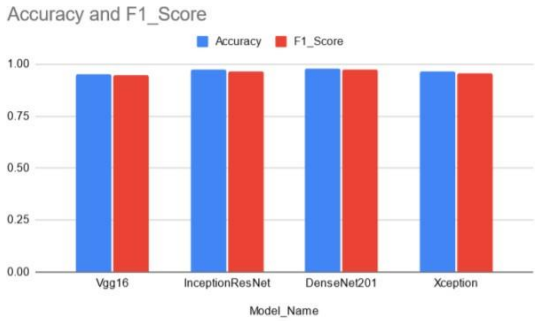


Fig 1. Accuracy and F1 score

5. Conclusion

We were able to successfully implement our proposed technique to identify a new and modified byte code into a class of available Malwares. We proposed a novel approach of first visualizing malware as an image and then training a classifier based on Deep Learning methods to maximize accuracy. The accuracy achieved using our CNN was found to be better than a few available models and the implementation method is quite fast in identifying the Malware class. The limitation we found with our proposed techniques is if the database is corrupt or not labelled accordingly, the algorithm won't be able to distinguish between clean and malicious files, so the solution will deliver unreliable results.

References

1. Alzaylaee, M. K., Yerima, S. Y., & Sezer, S. (2020). DL-Droid: Deep learning-based android malware detection using real devices. *Computers & Security*, *89*, 101663.
2. Kim, T., Kang, B., Rho, M., Sezer, S., & Im, E. G. (2018). A multimodal deep learning method for android malware detection using various features. *IEEE Transactions on Information Forensics and Security*, *14*(3), 773-788.
3. Rathore, H., Agarwal, S., Sahay, S. K., & Sewak, M. (2018, December). Malware detection using machine learning and deep learning. In *International Conference on Big Data Analytics* (pp. 402-411). Springer, Cham.

4. Cakir, B., & Dogdu, E. (2018, March). Malware classification using deep learning methods. In *Proceedings of the ACMSE 2018 Conference* (pp. 1-5).
5. K. He and D. -S. Kim, "Malware Detection with Malware Images using Deep Learning Techniques," *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, 2019, pp. 95-102, DOI: 10.1109/TrustCom/BigDataSE.2019.00022.
6. Kim, J. Y., Bu, S. J., & Cho, S. B. (2017, November). Malware detection using deep transferred generative adversarial networks. In *International Conference on Neural Information Processing* (pp. 556-564). Springer, Cham.
7. Venkatraman, S., Alazab, M., & Vinayakumar, R. (2019). A hybrid deep learning image-based analysis for effective malware detection. *Journal of Information Security and Applications*, *47*, 377-389.
8. Yuxin, D., & Siyi, Z. (2019). Malware detection is based on a deep learning algorithm. *Neural Computing and Applications*, *31*(2), 461-472.
9. Xiao, F., Lin, Z., Sun, Y., & Ma, Y. (2019). Malware detection based on deep learning of behaviour graphs. *Mathematical Problems in Engineering*, *2019*.
10. Sewak, M., Sahay, S. K., & Rathore, H. (2018, June). Comparison of deep learning and the classical machine learning algorithm for malware detection. In *2018 19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)* (pp. 293-296). IEEE.
11. Ye, Y., Chen, L., Hou, S., Hardy, W., & Li, X. (2018). DeepAM: a heterogeneous deep learning framework for intelligent malware detection. *Knowledge and Information Systems*, *54*(2), 265-285.
12. Kim, J. Y., Bu, S. J., & Cho, S. B. (2018). Zero-day malware detection using transferred generative adversarial networks based on deep autoencoders. *Information Sciences*, *460*, 83-102.
13. Yeo, M., Koo, Y., Yoon, Y., Hwang, T., Ryu, J., Song, J., & Park, C. (2018, January). Flow-based malware detection using convolutional neural network. In *2018 International Conference on Information Networking (ICOIN)* (pp. 910-913). IEEE.
14. Kan, Z., Wang, H., Xu, G., Guo, Y., & Chen, X. (2018, July). Towards lightweight deep learning-based malware detection. In *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)* (Vol. 1, pp. 600-609). IEEE.
15. Ni, S., Qian, Q., & Zhang, R. (2018). Malware identification using visualization images and deep learning. *Computers & Security*, *77*, 871-885.
16. Abdelsalam, M., Krishnan, R., Huang, Y., & Sandhu, R. (2018, July). Malware detection in cloud infrastructures using convolutional neural networks. In *2018 IEEE 11th International conference on cloud computing (CLOUD)* (pp. 162-169). IEEE.
17. Al-Dujaili, A., Huang, A., Hemberg, E., & O'Reilly, U. M. (2018, May). Adversarial deep learning for robust detection of binary encoded malware. In *2018 IEEE Security and Privacy Workshops (SPW)* (pp. 76-82). IEEE.
18. Aslan, Ö. A., & Samet, R. (2020). A comprehensive review of malware detection approaches. *IEEE Access*, *8*, 6249-6271.

19. Mahdavifar, S., & Ghorbani, A. A. (2019). Application of deep learning to cybersecurity: A survey. *Neurocomputing*, *347*, 149-176.
20. Vinayakumar, R., Soman, K. P., Poornachandran, P., Alazab, M., & Jolfaei, A. (2019). DBD: Deep learning DGA-based botnet detection. In *Deep learning applications for cyber security* (pp. 127-149). Springer, Cham.
21. Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2019). Deep learning approach for the intelligent intrusion detection system. *IEEE Access*, *7*, 41525-41550.
22. HaddadPajouh, H., Dehghantanha, A., Khayami, R., & Choo, K. K. R. (2018). A deep recurrent neural network-based approach for the internet of things malware threat hunting. *Future Generation Computer Systems*, *85*, 88-96.