CSE 543: Information Assurance and Security

Group 14 Course Project Report

Using Machine Learning to detect malware attacks in IoT systems

Group Members and Organization:

ASU ID	NAME	EMAIL ID	ROLE
1225585231	Amogh Manoj Joshi	ajoshi83@asu.edu	Leader
1225215745	Priyadarshini Venkatesan	pvenka44@asu.edu	Deputy Leader
1225497572	Vignan Varma Chekuri	vchekur1@asu.edu	Member
1224843919	Venkata Karthik Reddy Peddireddy	vpeddir2@asu.edu	Member
1224033577	Siva Priya Bollineni	sbollin5@asu.edu	Member
1223996449	Anusha Akuthota	aakuthot@asu.edu	Member
1224450630	Sarika Naidu Chirki	schiriki@asu.edu	Member
1222265655	Ramya Thota	rthota3@asu.edu	Member

List of Contents

1.1 Motivation and Background

The idea of the Internet of Things (IoT) has been around for a while, but the word "IoT" wasn't coined until the early 2000s, when the technology started to receive mainstream acceptance. From then on, there has been an increased use of the Internet of Things (IoT) devices in daily life which has led to an unprecedented rise in connected devices and an exceptional increase in the amount of data generated by the devices. The data input varies from sensor data to location data that are collected by IoT devices.

Software known as malware aimed at harming or disrupting computer networks, devices, and systems. Since then, malware has evolved into a wide variety of forms, including viruses, trojans, worms, and ransomware.

Malware attacks are among the major security risks that IoT devices encounter since they possess the potential to do a great deal of harm, including stealing sensitive data, interrupting necessary services, and even taking down entire systems.

As the IoT has grown in popularity, so have the threats posed by malware. Attackers have begun to target IoT devices with malware, taking advantage of their often-limited security features and vulnerabilities. These attacks can result in theft of personal information, destruction or manipulation of data, and even physical harm.

The first known malware attack on an IoT device occurred in 2013, when researchers discovered that a botnet called "Carna" had infected over 420,000 internet-connected devices, including routers, webcams, and printers. The botnet was able to infect the devices by exploiting known vulnerabilities in their software or by using default usernames and passwords.

To address this rising threat, researchers have used machine learning (ML) and deep learning (DL) techniques to detect and categorize malware in IoT devices. ML and DL have significant advantages in analyzing IoT data streams in real-time as they can find patterns and abnormalities in huge datasets which can prevent or reduce the impact of malware attacks that traditional signature-based approaches cannot. In order to add an extra degree of security, they can also be used in conjunction with other security tools like firewalls and antivirus programs.

To conclude, security researchers and businesses are utilizing ML and DL approaches to create more sophisticated and efficient malware detection systems. They aid in reducing false positives, increasing malware detection speed and accuracy, and ultimately enhancing system and network security.

1.2 Goal and scope of the project

The goal of our project is to detect Malware in IoT devices using Machine Learning methodologies. There are vast amounts of data, and machine learning and deep learning algorithms can swiftly evaluate these massive data sets to discover malware. The goal of the study is to provide a thorough assessment of the most famous and novel approaches present in recent times for identifying malware activity on IoT devices, which include SVM, K Nearest Neighbors, Naives Bayes, Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks and many more novel based deep learning techniques. Analyzing and implementing on the data processing, feature engineering, model selection and the various other techniques for evaluation of the model performance. The methods and results of the papers under discussion emphasize the top methods for dealing with malware detection to ensure that using IoT devices is always done in a secure manner.

Scope of the project:

1. Analyzing the presence of different malware types that infects the IoT devices.
2. Investigating the associated subtypes of malwares that are identified during static, dynamic and hybrid analysis.
3. Explore how various machine learning models perform at detecting malicious behavior.
4. Explore the common techniques employed by professionals to identify and prevent malware detection.Research on the novel approaches and trade-offs made while selecting the different machine learning models.
5. Investigate the machine learning based approaches specifically catering to resource constrained edge devices.
6. Understand the impact of ensembling different machine learning approaches on the accuracy of malware detection.
7. Recognize potential future study while summarizing the significance of the current state of machine learning for detecting malicious IoT.
8. Explore the advantages of combining machine learning and blockchain technology for IoT malware detection
9. Conclude on the advantages and limitations of all the research machine learning methods.

2. Summary of accomplishments of the project

The term "Internet of Things" (IoT) refers to a broad category of smart household appliances, wearables, medical equipment, and industrial systems that all require internet connectivity to operate. Because IoT devices are frequently interconnected with other devices and networks, it is crucial to identify malware in these systems as soon as possible because, once present, it can rapidly spread to other systems and devices. As more and more commonplace devices are connected to the internet, malware in IoT (Internet of Things) devices is becoming a bigger worry. Maintaining the security and integrity of IoT systems and making sure that these devices can be used safely and successfully depend on the early detection of malware in IoT devices.

As an efficient way to identify new and evolving malware threats that conventional signature-based techniques may miss, Machine Learning is increasingly used for malware detection in IoT. Large-scale data analysis, pattern and anomaly detection, and learning from prior assaults are all capabilities of machine learning algorithms which makes it one of the most efficient techniques for malware detection in IoT devices.

The major accomplishments of this project are summarized as follows:

- Explored traditional ML algorithms for malware detection and gained knowledge on performing static analysis and dynamic analysis using these algorithms.

- Understood the importance of early detection of malware in IOT with techniques like Gaussian Naive Bayes, K-NN and Random Forest.

- Analyzed malware detection using ML on resource constrained devices with scaling of machine learning for real-time malware detection in single IoT devices with weak processors.

- Explored malware detection with byte sequence of executable files using decision trees, random forests, and support vector machines.

- Explored multiple state of the art approaches of deep learning such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks, Ensemble of Deep Learning Techniques for IOT malware detection.

- Understood Lightweight IoT Malware Detection Solution Using CNN Classification, DCNN with malware visualization, Dynamic analysis for IoT malware by using DAIMD scheme that classified IoT malware using the CNN in a cloud environment.

- Understood effectiveness of channel Boosted CNN for malware detection and Deep Eigenspace Learning using CNNs for Malware Detection.

- Explored RNN with Long short-term memory (LSTM) to detect and analyze malware behavior, Deep Convolutional Neural Network with LSTM for healthcare IoT and smartphone malware detection.

- Understood Ensemble ML approaches (CNN + LSTM) for malware detection and Multi Dimensional DL approaches CNN and LSTM for IOT Malware detection and classification.

- Analyzed the benefits of ensembling Stacked AutoEncoders (SAEs) and the traditional ML algorithms for IoT malware detection

- Analyzed how combining machine learning with block chain technology or deep learning can be advantageous to malware detection in IOT.

3. Individual accomplishments

Member	Accomplishments
Amogh Manoj Joshi (Leader)	● Explored Hybrid Approaches in Deep Learning for Malware Detection in IoT across techniques like CNN, RNN, LSTM etc. ● Further explored variants of CNNs and LSTMs like Channel-Boosted CNN, Bi-LSTM etc. ● Also researched on combining Machine Learning and Blockchain Technology for IoT Malware Detection (a unique perspective for the project theme) ● Validated the impact of the research papers chosen by the group members during weekly study sessions ● Conducted weekly meetings to track the individual progress of the team members and understand the group's learnings and took meeting notes of every meet ● Proofread all the weekly reports and notified the changes if any ● Organized the flow of contents for the final report
Priyadarshini Venkatesan (Deputy Leader)	● Comprehended in depth about the honeypot architecture for defending DDoS attacks ● Acquired knowledge on various ML/DL algorithms such as GNB, Decision tree, Random Forest, AdaBoost, CNN for detecting malware in IoT devices. ● Explored on dynamic analysis for IoT malware detection by (DAIMD) scheme by using the CNN model in a cloud environment. ● Proofreading and alignment of the final report ● Coordinated with every member of the group for updates on their task progress. ● Assisted the leader in assigning tasks to members of the group. ● Coordinated with the leader on submission and evaluation of weekly reports.
Vignan Varma Chekuri	● Conducted in-depth research on malware detection using machine learning in IOT devices ● Analyzed the benefits and drawbacks of using machine learning and deep learning for detecting IoT malware. ● Assisted in creating comprehensive weekly reports that documented the team's progress and identified potential roadblocks. ● Contributed and in-depth understanding of conducting research, analyzing data, and writing technical reports.
Venkata Karthik Reddy Peddireddy	● Acquired knowledge on different categories of malware and developed an understanding of their creation and functionality. ● Have a deeper understanding about the application of Deep Eigenspace Learning, graph embedding, and eigenspace techniques for detecting and identifying malware in IoT and IoBT. ● Gained a profound comprehension of Deep Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks in the context of malware detection. ● Prepared gantt charts for the weekly reports. Contributing to weekly discussions and reports actively.

Siva Priya Bollineni	● Learned about the importance of using a hybrid approach that combines both static and dynamic analysis for detecting and categorizing malware. ● Learned about preprocessing data for deep learning models, using DRNN architecture to capture temporal dependencies in data, and evaluating model performance using various metrics. ● Got deep understanding in data preprocessing, feature engineering, and model selection for malware detection and evaluation techniques to optimize the performance of machine learning models for malware detection. ● Contributions for Gantt charts were created in weekly reports. ● Actively participated in meetings and cultivated skills such as effective collaboration, ideation for strategic planning, accountability.
Anusha Akuthota	● Acquired knowledge of how machine learning and deep learning algorithms are used to detect IoT malware ● Learnt lightweight CNN Classification malware detection for IoT devices using a 5G network ● Attained knowledge of detecting IoT malware by system calls using the RNN algorithm ● Gained an understanding of utilizing Behavioral Traffic Analysis to Detect and Identify IoT Malware Effectively with Multi-task DL technique-LSTM ● Collaborated with the team for preparing reports and participated in team meetings.
Sarika Naidu Chirki	● Learned the most used machine learning algorithms used for malware detection which include SVM, K Nearest Neighbors, Naive Bayes. ● Understanding the IoT dataset - the various devices, the configs used, the preprocessing of the data sets, tuning parameters. ● Learned new approaches for detecting malware using approaches like static analysis, dynamic analysis, and understanding byte, Robust - NN and C4N. ● Contributing to weekly discussions and reports and actively engaged in group discussions.
Ramya Thota	● Learned about ensemble machine learning and its techniques Adaboost and SGDC for malware detection. ● Able to gain understanding of techniques like malware visualization, DCNN and how both of these can be combined to detect malware effectively in Industrial IOT devices. ● Learned that behavior based deep learning framework gave optimal performance in malware detection where behaviors and Stack encoders were used. ● Actively engaged in group discussions and collaborated with my teammates.

4. Introduction to IoT

"Internet of Things" is referred to as "IoT." It describes a system of actual objects, including cars, appliances, and other household items, that are linked to the internet and are capable of exchanging data. By integrating sensors, software, and network connectivity into these devices, IoT technology enables them to exchange data and communicate with other devices. They can be as basic as sensors or as sophisticated as complex systems that communicate with the real world.

The ability to gather and analyze enormous volumes of data, which offers insights and enables new services and applications, is one of the primary advantages of IoT. Across a wide range of businesses, IoT offers a number of advantages, including increased efficiency and productivity through task automation, cost savings through predictive maintenance, and improved customer experiences through tailored recommendations. Additionally, it enhances safety and security by identifying potential risks and warning users to take appropriate action. Large volumes of data may be gathered and analyzed by IoT devices, offering insightful information that can be used to make better decisions about supply chains and logistics, among other things. IoT has the potential to revolutionize industries and enhance our daily lives by delivering better experiences, higher efficiency, cost savings, safety and security, and enhanced decision-making abilities.

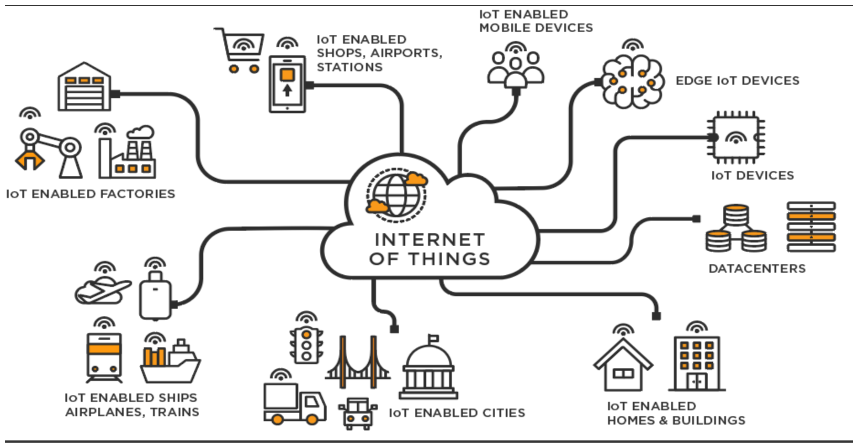

IoT ENABLED SHOPS, AIRPORTS, STATIONS
IoT ENABLED MOBILE DEVICES
EDGE IoT DEVICES
IoT DEVICES
DATACENTERS
IoT ENABLED FACTORIES
INTERNET OF THINGS
IoT ENABLED SHIPS AIRPLANES, TRAINS
IoT ENABLED CITIES
IoT ENABLED HOMES & BUILDINGS

Fig. 1. Functionalities of IOT

However, the IoT development brings about a number of serious problems, particularly in the areas of security and privacy. IoT devices can gather sensitive information about people and businesses and are frequently subject to hackers. In order to ensure that IoT devices are secure and protect user privacy, it is crucial for manufacturers and consumers to take the appropriate action.

5. Vulnerabilities in IOT

IoT vulnerabilities are weak points or security defects in IoT networks or devices that can be used by attackers to compromise the confidentiality, integrity, or availability of the system. These faults in the system's architecture, its implementation, or its configuration may lead to security breaches, unauthorized access, and other malicious behaviors. These IoT device vulnerabilities are taken as advantage by cybercriminals who then use them to target businesses and end users. Cybercriminals can advance further into corporate networks using the original breach of a vulnerable device. An intruder tries to take advantage of a machine's weakness before increasing their level of access. Cybercriminals initiate massive cyberattacks like distributed denial-of-service (DDoS) attacks using botnets, which are large networks of devices like routers. Botnets are collections of compromised devices controlled by a command-and-control (C&C) computer. For instance, the Mirai malware brought down a number of important websites and services in 2016, including gaming services. Using a botnet code that was disseminated into the open for other hackers to exploit, Mirai targeted insecure devices.

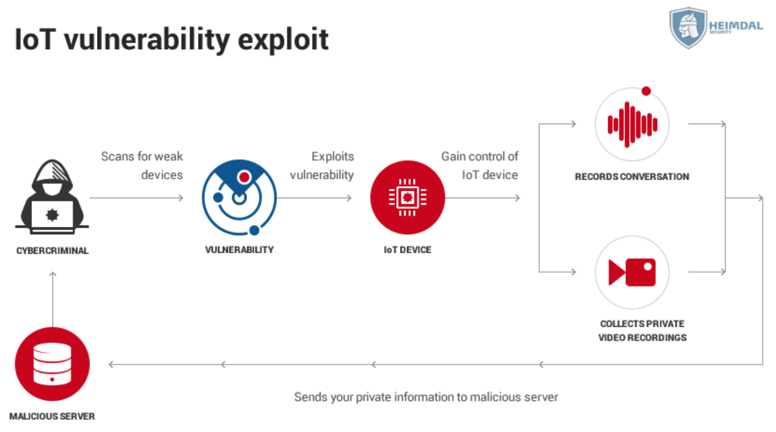

IoT vulnerability exploit
HEIMDAL
Scans for weak devices
Exploits vulnerability
Gain control of IoT device
RECORDS CONVERSATION
CYBERCRIMINAL
VULNERABILITY
IoT DEVICE
COLLECTS PRIVATE VIDEO RECORDINGS
MALICIOUS SERVER
Sends your private information to malicious server

Fig. 2. Vulnerabilities in IOT

In connected appliances, digital assistants, wearables, health trackers, and other devices, the IoT is rapidly permeating the house. Vulnerabilities in IoT services can open up new doors for other devices linked to home networks, like laptops and computers. Hackers may also be able to access business networks if these devices are used to work from home or as part of a bring-your-own-device (BYOD) policy. In order to gain entry to internal networks, attackers can target IoT devices with known flaws. They can then initiate attacks to steal information from networks and devices connected to private or public networks, such as Domain Name System (DNS) rebinding attacks.

10. What is Malware?

Malware, an abbreviation for "malicious software" refers to a program or code that is transmitted via a network and is intended to infect, exploit, steal, or perform any other action desired by an attacker. Due to the numerous types of malware, there are various methods of infecting computer systems. Despite the differences in functionality and type, malware typically pursues one of the following goals:

- To enable attackers to remotely control a compromised machine.
- To dispatch spam messages to unsuspecting recipients from the compromised machine.
- To probe the local network of the compromised user and pilfer sensitive data.

Malware can manifest in diverse forms, each with its own unique characteristics and operates differently to achieve its malicious objectives. Below is a list of different types of malware and an overview of how they work.

- **Adware:** This type of malware may be deemed legitimate by some, but certain variants illegally infiltrate computer systems and significantly disrupt users.
- **Botnets:** Botnets, short for "robot networks," consist of infected computers under the control of a single attacking entity, utilizing command-and-control servers. Botnets are extremely versatile and adaptable, capable of maintaining resilience through redundant servers and by using infected computers to relay traffic.
- **Cryptojacking:** Cybercriminals engage in malicious cryptomining, a process that employs computing power to authenticate blockchain network transactions and obtain cryptocurrency by installing software on both personal and corporate computers, laptops, and mobile devices. Some well-known cryptojacking malwares include Coinhive, JSEcoin, and Crypto-Loot.
- **Ransomware:** Ransomware is a type of malware that encrypts a victim's files and demands a ransom payment, usually in cryptocurrency, in exchange for the decryption key.
- **Spyware:** Spyware is a type of malware that is designed to spy on a victim's computer activities and collect sensitive information. Spyware can be used to monitor keystrokes, track web browsing habits, and steal passwords and other personal data.
- **Trojans:** Trojans are a type of malware that are disguised as legitimate software programs, tricking victims into installing them on their computers.
- **Worms:** Worms are designed to spread from one computer to another, typically exploiting vulnerabilities in network protocols or operating systems. Worms are self-replicating and can quickly spread across networks, causing significant damage to infected systems.

11. Why is early detection (zero day detection) of malware important?

Malware is dangerous because it can cause significant damage to computer systems, networks, and data. Organizations must quickly identify malware, especially zero-day malware, in order to safeguard their people, data, and systems from possible harm from cyberattacks. Malware can be spread through a number of different channels, including email attachments, software downloads, and malicious websites.

Organizations are especially concerned about zero-day attacks because they take advantage of previously unidentified software or hardware flaws, making it difficult to identify and stop them. Because there is frequently no known fix or patch to stop zero-day attacks, it is essential to discover them quickly and take appropriate action.

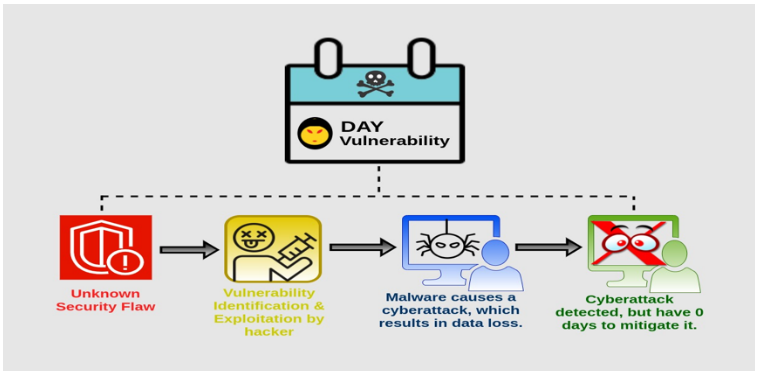

DAY
Vulnerability
Unknown Security Flaw
Vulnerability Identification & Exploitation by hacker
Malware causes a cyberattack, which results in data loss.
Cyberattack detected, but have 0 days to mitigate it.

Fig. 3. Real detection of zero-day vulnerability

The following justifies why it's crucial for enterprises to utilize early malware detection, especially zero-day detection:

- Damage Prevention: Organizations can lessen or completely avoid the harm that malicious software causes by preventing its infection. This is because it makes it possible for security personnel to quickly recognize and address risks before the virus spreads and seriously harms the system or network.
- Meeting Compliance Requirements: Many firms are subject to regulatory compliance obligations, which call for them to have systems in place for speedy malware detection and reaction.
- Preventing Malware Spread: By enabling businesses to isolate the compromised system and contain the threat, early malware detection can help avoid this from happening. For businesses with sizable networks and lots of endpoints, this is especially crucial.
- Protection of Important Data: Information can be safeguarded against theft and compromise by early identification of such viruses. This is crucial for businesses handling sensitive data, such healthcare providers, financial institutions, and governmental bodies.

12. A Summary of Machine Learning Algorithms

Machine learning is a division of artificial intelligence that utilizes statistical models and algorithms to enable computers to learn from data without explicit programming. The primary objective of machine learning is to analyze the data, so the program can recognize patterns and connections and utilize them to make predictions or decisions regarding new data. There are several categories of machine learning, such as supervised, unsupervised, and reinforcement learning. In supervised learning, the algorithm is trained on labeled data where each data point has an output associated with it. In unsupervised learning, the algorithm is trained on unlabeled data and must identify patterns and relationships on its own. In reinforcement learning, the algorithm learns through trial and error, receiving feedback in the form of rewards or punishments for its actions. Below are few ML algorithms used for classification tasks:

1. **Logistic Regression**: A statistical method to predict the probability of an event occurring, based on a set of independent variables.
2. **Decision Trees**: A method to visualize and analyze decision-making processes by creating a tree-like model of decisions and their possible consequences.
3. **Random Forest**: An ensemble learning method that combines multiple decision trees to improve prediction accuracy and reduce overfitting.
4. **K-Nearest Neighbors**: A classification algorithm that assigns a class label to a new data point based on the class labels of its k-nearest neighbors in the training data.
5. **Support Vector Machines**: A supervised learning algorithm that finds the optimal hyperplane that separates data into different classes.
6. **Naive Bayes**: A probabilistic algorithm that predicts the likelihood of a new data point belonging to a particular class based on the probability of its features.
7. **Neural Networks**: A collection of interconnected nodes that work together to learn patterns and relationships in data.
8. **Clustering**: A group of unsupervised learning algorithms that identify groups of similar data points based on their similarity to each other.

8.1 Famous Malware Datasets

Some of the well-known IoT malware datasets together with their types, number of classes, and sample count:

1) IoT-23: (Type: Multi-Class Classification)

Researchers at the University of New South Wales in Australia generated this dataset, which includes 23 different varieties of IoT malware. There are 12,000 samples in it that were gathered from a variety of sources, including honeypots, malware repositories, and public IoT devices. Together with metadata, the collection contains information about the source and destination IP addresses, ports, and protocol types.

2) Stratosphere IoT: (Type: Binary classification)

Researchers at the University of California, Santa Barbara developed this dataset, which includes 10 various kinds of IoT malware. This dataset consists of 3,332 samples gathered from various resources. Raw network traffic such as the number of bytes, packets, flows, and features that were extracted are both included in the dataset.

3) IoTPot: (**Type:** Binary classification)

The University of Twente in the Netherlands dataset observed and accumulated this dataset which includes one type of IoT malware, Mirai. It contains 2,250 samples that were gathered from honeypots that resembled weak IoT devices. Raw network traffic makes up most of this dataset.

4) CIC-IDS2018: (Type: Multi-class classification)

This dataset, which was created by researchers at the Canadian Institute for Cybersecurity, covers many kinds of network traffic, including IoT traffic. With 2.8 million samples in total, it contains 16 different kinds of malware as well as benign traffic. Raw network traffic and features that were extracted are both included in the dataset.

TABLE 1. Malware Datasets

Serial Number	Dataset Name	Classification Type	Number of IoT malware varieties	Number of Samples
1	IoT-23	Multi-Class	23	12000
2	Stratosphere IoT	Binary	10	3332
3	IoTPot	Binary	1	2250
4	CIC-IDS2018	Multi-Class	16	2.8 million
5	Gafgyt and Mirai	Binary	2	12633

9. MACHINE LEARNING BASED APPROACHES

9.1 Exploring Traditional ML Algorithms for IoT Malware Detection

In traditional machine learning algorithms, models are trained on previous data to classify or make predictions on new data. Some of the frequently used techniques are decision trees, support vector machines (SVM), random forest, and naive Bayes. Decision trees can identify important characteristics for malware detection, while SVM is excellent at classifying malicious and non-malicious samples. Random forest can manage complicated data and determine significant features, and naive Bayes is practical in handling large amounts of data and classification tasks [1].

By recognizing key features that differentiate malicious and benign files, traditional machine learning algorithms can accurately detect malware. These techniques can rapidly process vast amounts of data, contain malware threats, and adjust to new and emerging threats based on past experiences. Moreover, their ability to scale makes them appropriate for enterprise-level malware detection systems, allowing organizations to monitor and identify potential threats across multiple endpoints simultaneously. Due to their advantages, and integrating with different approaches for analysis of the data make the traditional machine learning algorithms a crucial tool in protecting critical information and systems against cyber threats. The following sections explore more based on these.

9.1.1 Static and Dynamic Analysis using Standard Machine Learning Algorithms

This report outlines a proposed method for identifying and categorizing IoT malware using a hybrid malware analysis system that combines both static and dynamic analysis techniques. To implement the proposed approach discussed by Madan et al. [2], a dataset consisting of approximately 1010 malware samples was used to train a machine learning model. Static analysis involves examining the malicious code without running the program, utilizing various techniques to determine its maliciousness, such as file name, MD5 hashes, string features, opcode sequence, control flow, and recognition by antivirus detection tools. On the other hand, dynamic analysis involves executing the malware sample in a controlled sandbox environment to observe its behavior during execution and gain insights into its functionality. This method is different from static analysis in that it focuses on behavior analysis. Hybrid analysis combines the characteristics of both static and dynamic analysis. It leverages static analysis techniques to understand the malware code before execution and dynamic analysis techniques to observe the real behavior of the malware during execution. Reverse engineering techniques like disassembly and decompiling are utilized in static analysis to comprehend how the malware operates, while dynamic analysis extracts features by monitoring malware activities like file system changes, API calls, network activity, process changes, etc. Abusnaina et al. [38] also discussed the static and dynamic analysis to understand malware's capabilities, behavior, and intent for building detectors and designing defenses. To develop a machine learning model for detecting and classifying malware, crisp features were extracted from the malware binaries. The features were categorized based on static and dynamic analysis, opcode sequence analysis, API call graph features, AV engine report, and network traffic anomalies from pcap data. This ensured a comprehensive set of features to train the machine learning model for accurate detection and classification of malware.

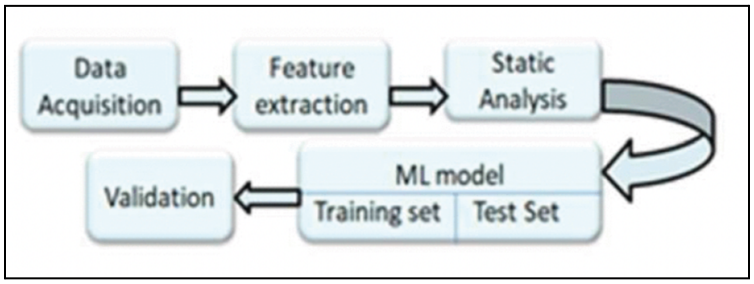


Fig. 4. Proposed malware analysis approach

Experiments were conducted to train a detection and classification model using selected features from static and dynamic analysis of malware binaries. Two feature sets were employed, namely, opcode sequence and API call graph. The model was evaluated using separate test and validation datasets, and the Random Forest algorithm emerged as the best performing algorithm with accuracy rates of 97.14% and 93.3% for opcode sequence and API call graph features, respectively. The study's findings may prove useful to researchers and practitioners in the field of cybersecurity seeking to develop more effective techniques for malware detection and classification. The results for using the Opcode Sequence and Call Graph for machine learning models development to predict potential security threats are shown below:

Algorithm	**Opcode Feature**		**API Call graph**	
	Accuracy	**F-Measure**	**Accuracy**	**F-Measure**
Decision Tree	0.917	0.965	0.9512	0.9612
Random Forest	0.933	0.977	0.9714	0.977
Naive Bayes	0.892	0.950	0.9651	0.9842

Table 2. Malware Detection Using Opcode Feature and API Call graph Results

Some other popular ensemble learning algorithms for binary classification and regression issues include AdaBoost [5] (short for Adaptive Boosting). The algorithm creates a single powerful classifier by fusing several weak classifiers together. AdaBoost's main principle is to prioritize the training examples that the previous, weak classifiers failed to correctly categorize by raising their weights in the following rounds. As a result, the program can adjust to the data and get better with each round. By instructing weak classifiers on a collection of features taken from malware samples, AdaBoost has been used to detect malware effectively. Opcode n-grams, API call sequences, and system call sequences are a few common characteristics used for malware detection. Studies have demonstrated the efficacy of AdaBoost in detecting malware, with some studies showing detection rates of over 99%.

A well-liked machine learning method called Stochastic Gradient Descent Classifier (SGDC) is used for binary classification issues like malware detection. It is a different type of linear support vector machine (SVM) classification that employs stochastic gradient descent optimization to determine the ideal model weights. The central concept of SGDC is to change the weights and bias incrementally rather than all at once using a random subset of the training data. Because of this, the algorithm is more effective and scalable for big databases.

Malware detection using SGDC [5] has been demonstrated to be effective, with some studies showing detection rates of over 95%. Similarly, some other papers also highlight the need of merging Machine Learning with IoT for its sustainability to malicious attacks. Dartel et al. [22] also discusses the challenges associated with the integration of smart data and machine learning, including the management of large amounts of data and the need for skilled personnel. The paper concludes by highlighting the potential of smart data and machine learning in the development of intelligent systems and calls for continued research and development in this area. Overall, the paper provides an overview of the benefits and challenges of integrating smart data and machine learning for the development of intelligent systems, and proposes solutions to overcome these challenges.

9.1.2 Early Detection of IoT Malware using Machine Learning

Early malware detection is critical for avoiding or reducing harm to computer systems, networks, and data, it allows quick action to be taken for prevention of further malware. Here, we will discuss a paper which outlines a method for detecting the malware from network activity.

Kumar et al. [4] proposed an approach called EDIMA to categorize network traffic flows as either benign or malignant - these are the two types of gateway traffic. To improve speed and lower memory requirements, the classification of IoT access gateway traffic is done at the aggregate level rather than at the device level. Malicious traffic is defined as gateway traffic that contains malware-induced scanning packets from one of three malware categories, as opposed to benign traffic, which is defined as regular gateway traffic without any malware-induced scanning packets. Training data samples made up of packet captures from both classes are created to categorize gateway traffic. While creating benign traffic is simple, malicious traffic's gateway traffic at set session intervals.

The access gateway's ML classifier module receives samples of incoming traffic and categorizes them using an ML model that was trained by the ML model constructor module. Using feature vectors and class labels acquired from the Packet Traffic Feature Database, the ML model constructor module trains the ML model. The optional sub-sampling module saves computing effort by only sending a portion of incoming IoT packet traffic, and the policy module specifies the actions to be taken when malicious traffic is discovered. The online feature database is regularly updated with the feature vectors generated from raw traffic data samples and their accompanying class labels.

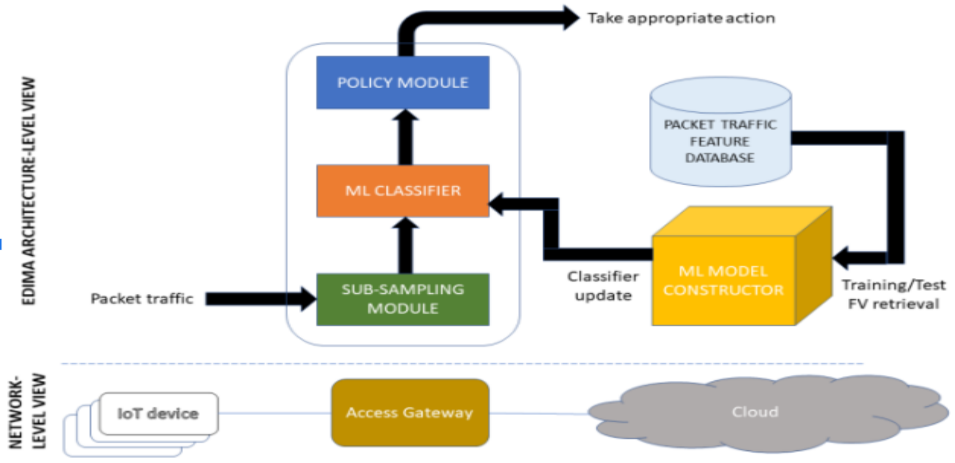


Fig. 5. EDIMA Architecture by Kumar at el. [4]

Below we discuss some Machine learning Algorithms that are used in many papers for comparing the accuracy. Paper by Nakhodchi et al. [26] compare the algorithms: KNN, Random Forest and Gaussian Naive Bayes and SVM. The same algorithms are also implemented in [23], [27], [33] and [40]. The evaluation metrics used for this is accuracy, precision, recall,false positive rate (FPR). The result in Kumar et al. [4] suggested that K-NN has better accuracy than Random Forest classifier and Gaussian Naive Bayes with K-NN accuracy being 94.44%.

9.1.3 Using Machine Learning in Honeypot Architecture in DDoS Detection System

A DDOS (Distributed Denial of Service) detection system is a security technique designed to detect and mitigate DDoS attacks on computer networks. DDoS attacks overwhelms a network with a massive volume of traffic or requests, forcing the system to crash or become unavailable.

A honeypot architecture is a security mechanism designed to detect and deflect unauthorized access to computer systems. Moreover, it is a trap which is designed to lure attackers into revealing their tactics and techniques. This architecture typically includes several components, from a decoy system or network, from a monitoring system to track and analyze activity, and a response system to take action against attackers. Machine learning (ML) can improvise the honeypot architecture by allowing in advanced and proactive threat identification and response. Machine learning algorithms are trained to study and detect patterns of activity that suggest a potential attack, such as anomalous network traffic or suspicious behavior. They assist in detecting threats faster and more accurately, allowing for a speedier reaction.

Honeypots are a solution to address these security issues by simulating vulnerabilities that can be exploited. They are built to draw in attackers and collect data such IP addresses, MAC addresses, port numbers, device types, and malware executables and commands. By analyzing and comprehending attacker techniques with the help of this data, security measures can be strengthened in order to thwart potential attacks in the future. Honeypots are an efficient tool for identifying different malware and its variants and have been used for many years in computer security. They can be categorized according to the amount of contact they permit with attackers, which can range from little to a lot, and their intended use, which can be either production or research. While production honeypots are used to protect against risks, research honeypots are used to acquire information and spot prospective hazards. The author Matin et al. [28] also proposes the honeypot architecture to detect malware by using support vector machines and Decision Tree algorithms for their higher accuracy and efficiency. A honeypot architecture is used to gather information on attempts to install malware as part of the suggested solution for identifying and foreseeing security threats to IoT devices.

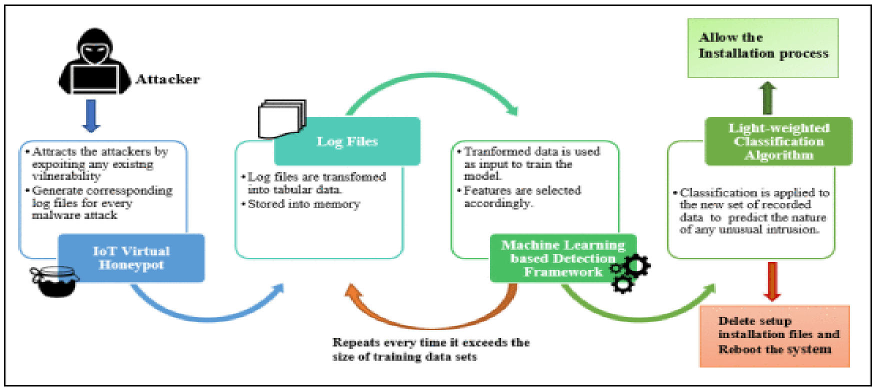

Attacker
•Attracts the attackers by expoiting any existing vilnerability
•Generate corressponding log files for every malware attack
IoT Virtual Honeypot
Log Files
•Log files are transfomed into tabular data.
•Stored into memory
•Tranformed data is used as input to train the model.
•Features are selected accordingly.
Machine Learning based Detection Framework
Light-weighted Classification Algorithm
•Classification is applied to the new set of recorded data to predict the nature of any unusual intrusion.
Allow the Installation process
Delete setup installation files and Reboot the system
Repeats every time it exceeds the size of training data sets

Fig. 6. The honeypot process.

Next, using the appropriate unsupervised learning algorithms, such as clustering, anomaly detection, and neural networks, the machine learning model trained on this data may automate the process of identification and prediction of incoming threats. The model uses unsupervised learning to categorize unknown malware family variants and does not require human input to establish rules or give labels. Malware detection can be thought of as a classification or clustering problem, and depending on the type of data provided, either approach can be employed.

In the initial stages, it is essential to attract attackers to exploit vulnerabilities in IoT devices. Vishwakarma et.al [34] suggest using IoT honeypots, which are systems or devices created to mimic an exploitable IoT device and draw attackers, to do this. According to how much an attacker interacts with the IoT device, the authors describe three different forms of honeypots: high, medium, and low engagement. Due to resource limitations, a medium interaction honeypot is selected because it is not practical to set up a high interaction honeypot for IoT devices. He provides information about IoTPOT, Telnet IoT honeypots, HoneyThing, Dionaea, ZigBee honeypots, Multi-purpose IoT honeypots, and ThingPot in terms of malware detection. Last but not least, suggests choosing ThingPot, which can simulate the entire IoT ecosystem and all supported application layer protocols, including MQTT, XMPP, AMQP, CoAP, UPnP, and HTTP REST.

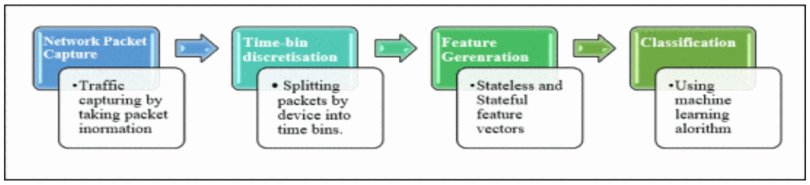

Network Packet Capture
•Traffic capturing by taking packet inormation
Time-bin discretisation
• Splitting packets by device into time bins.
Feature Gerenration
•Stateless and Stateful feature vectors
Classification
•Using machine learning alorithm

Fig. 7. The machine learning-based detection framework's workflow.

Machine learning algorithms, which include numerous processes like data gathering, feature extraction, and binary classification, can be used to study the behavior of IoT traffic. Packet length, inter-packet intervals, and protocol are characteristics derived from IoT-specific network behavior. For detecting attacks, classifiers like random forests, K-nearest neighbors, and neural networks are very useful. Anomaly detection entails capturing traffic from IoT devices, grouping packets by device and time, and then extracting stateless and stateful information for each packet. Stateful features gather aggregate flow data while stateless features are lightweight. Then, to distinguish between regular traffic and DDoS traffic, binary classification is carried out using various methods. Deep learning classifiers operate better because they can use additional data produced by real-world deployments.

A mix of virtualization, machine learning techniques, and honeypots can be used to create a real-time machine learning detection framework for Internet of Things devices. Classifiers can be deployed on the router level due to IoT limitations, whereas virtual boxes can be utilized to put the detection framework on every IoT device in a network. IoT network simulators can create enough IoT traffic to train the machine learning model effectively. However, if a chosen honeypot is employed, there is no need for extra simulators because the honeypot can produce the necessary traffic. Bash scripts on Linux and machine learning tools can be used to change log files into the format necessary for input to the machine learning model. The usage of honeypots ensures the logging of newly appearing malware traits, which may be used by the ML-based detection framework to efficiently train their classifiers. Overall, this method can offer a real-time solution for identifying and combating malware attacks on IoT devices.

9.1.4 Malware Detection from Byte Sequence of Executable Files using Machine Learning

Another approach to detection of malware is from byte sequences at the entry point of ELF files, the malware for linux based systems attacks using these ELF files. A binary file made up of sequences of bytes containing data and information about commands that can be processed by a machine to do certain tasks that is in the form of an executable file. Since this approach is for linux based systems - which is a part of Unix we use binary format. The Byte sequences, also discussed byte-code in Peter et al. [32] obtained from executable files.

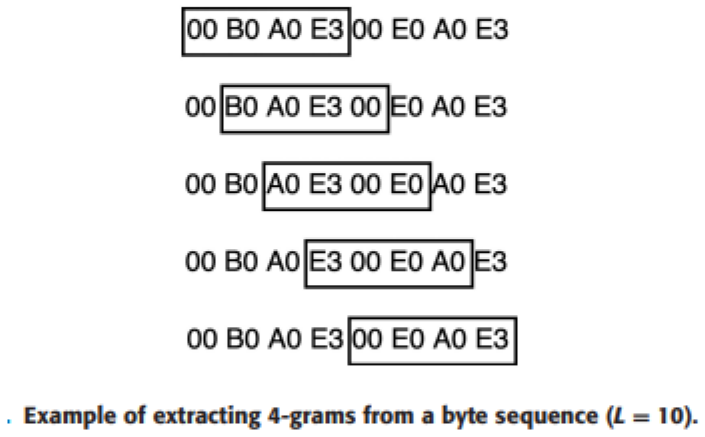


Fig. 8. Example of Byte Sequence from Wan et al.. [24]

Wan et al. [24] proposed classification and detection of malware from a byte sequence from executable files. are taken as the input features for machine learning algorithms for detection and classification. In this approach, when a downloaded program is found to be malicious, a security warning is raised to notify the user. All of the classifiers employed in the experiment worked remarkably well, with an accuracy rate of over 99%, according to the evaluation results. This shows that across all IoT platforms, the byte sequences at the entry point include essential information for differentiating between malware and benign. The classification methods that are used for testing the data are SVM, KNN, NB and MLP.

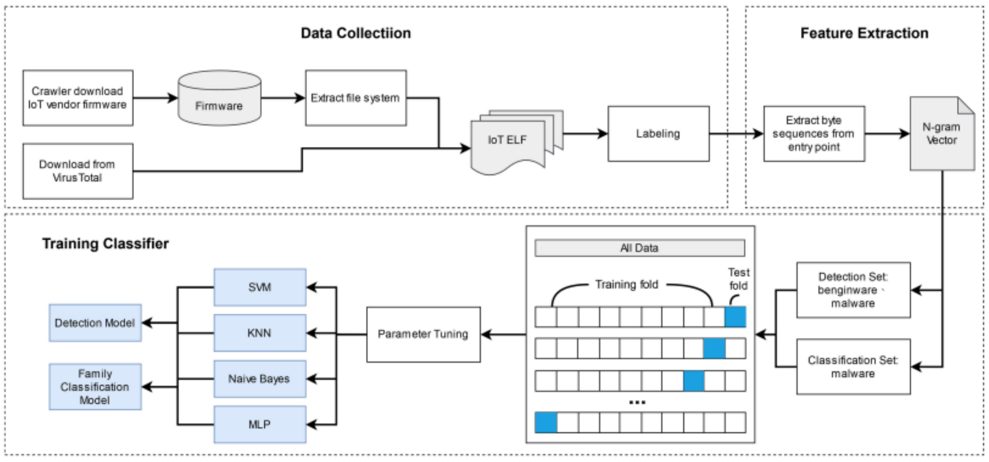


Fig. 9. Architecture of paper proposed by Wan et al. [24]

The Support Vector Machine (SVM) algorithm, which required somewhat more training time than other techniques, had the greatest accuracy rate of 99.96% with training time as 3,466 seconds and testing time the least among other models with value 0.0006. To produce predictions, SVM just requires a bias parameter and a vector in the D-dimensional space, where D is the dimension of the feature vectors.

9.1.5 Using Machine Learning for Malware Detection on Resource Constrained Devices

There are several reasons why resource-constrained devices are increasing these days. One major factor is the proliferation of the Internet of Things (IoT), which has led to an explosion of connected devices that require low-power, low-cost, and small form factors. Another reason is the increasing demand for mobile devices such as smartphones and tablets, which require longer battery life and higher portability. Additionally, the growth of edge computing and cloud computing has created a need for devices that can perform computational tasks locally, reducing the need for data transmission and reducing latency. Finally, advances in technology have made it possible to produce smaller and more energy-efficient components, enabling the creation of devices that can operate with limited resources.

9.1.5.1 Scaling Machine Learning in single IoT devices with weak processors

The use of edge-efficient devices is increasing, but such devices are also prone to malware. Machine learning (ML) algorithms have shown promising results in detecting malware on IoT devices. Due to the resource constraints of these devices, it is not possible to run heavy machine learning models on them. Moreover, due to the weak processors of IoT devices, detection of malware on a single IoT node has not been possible. ABC et al. [22] investigates the possibilities of using ML algorithms to detect malware on a single IoT device in real-time. The study utilized the LOLIN32 board, which has the ESP32 chip with 512kB of RAM and a 240MHz dual-core processor. These boards are resource-constrained devices designed for lightweight applications. Despite their limitations, they are capable of performing a wide range of tasks, making them highly portable and ideal for use in applications where space is limited.

The research conducted by Achary et al. [6] demonstrates the potential of using ML algorithms for near real-time network traffic anomaly detection on a single IoT device, which can improve the security of IoT networks. The study utilizes Decision Tree classifiers, which highlight the effectiveness of ML in detecting malware on IoT devices with weak processors. The research's implications include reducing the number of infected devices, increasing consumer confidence in IoT devices, and accelerating the growth of the IoT industry. However, the research's limitations include the use of a single ML algorithm and a single IoT device, which limits the generalizability and scalability of the results. Further research is needed to investigate the effectiveness of other algorithms and the scalability of the proposed solution on different types of IoT devices.

In summary, the research paper showcases the potential of using ML algorithms for real-time malware detection on a single IoT device, contributing to the improvement of IoT security. The results provide insights into the possibility of implementing malware detection mechanisms on devices with weak processors. However, further research is required to investigate the effectiveness of other algorithms and the scalability of the proposed solution on different types of IoT devices.

9.1.5.2 Using ML for reduced data storage

Reducing data storage requirements can improve efficiency, lower costs, and enhance security. Machine learning algorithms can compress data, select important features, and detect anomalies, reducing the amount of data needed for analysis and storage. Nakahara et al. [21] proposed an architecture that detects malware traffic in IoT devices using summarized statistical data of packets instead of whole packet information. The architecture comprises three modules: traffic monitoring, traffic analysis, and traffic detection, and uses machine learning algorithms of Isolation Forest and K-means clustering to detect malware traffic. The authors evaluated the proposed architecture, which achieved high accuracy in detecting malware traffic, reduced data storage size by over 90%, and analyzed the number of IoT devices with low computational resources.

The proposed architecture's effectiveness reduces the storage space taken up by data and can analyze the number of IoT devices with low computational resources. The authors used machine learning algorithms of Isolation Forest and K-means clustering to detect malware traffic and showed high accuracy in detecting malware traffic in the evaluation results. The proposed architecture can be implemented in various domains, including healthcare, industrial control, and smart homes, to enhance security and prevent infections.

In conclusion, reducing data storage is desirable due to its potential to improve efficiency, lower costs, and enhance security. The architecture proposed by Nakahara et al. [21] uses machine learning algorithms of Isolation Forest and K-means clustering to detect malware traffic with high accuracy, reducing data storage size by over 90%, and analyzing the number of IoT devices with low computational resources. The proposed architecture can be applied in various domains to enhance security and prevent infections.

10. What is Deep Learning? Why is it better than Traditional Machine Learning?

10.1 What is Deep Learning?

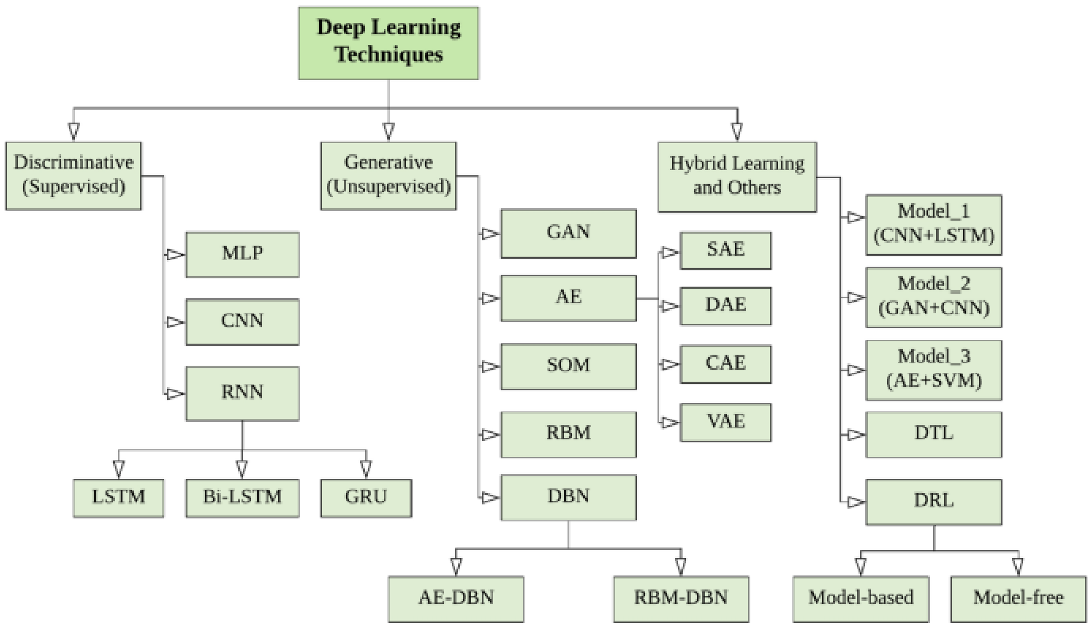


Fig. 10. Flow chart summarizing the techniques used in Deep Learning

Deep Learning is a subfield of Machine Learning that uses artificial neural networks to learn and solve complex problems. The reason it is called "deep" is because it involves the use of neural networks with many layers, which are capable of learning representations of data at different levels of abstraction. Fig. 1 briefly summarizes all the techniques that Deep Learning encompasses. Other techniques used in Deep Learning include Deep Belief Networks (DBNs), Autoencoders, and Reinforcement Learning.

10.2 The rise of Deep Learning and why is it better than Traditional Machine Learning?

The recent years have witnessed an incredible surge in the amount of attention which Deep Learning has garnered from the research community. Examining the volume of academic articles on the subject is one technique to gauge the growth of deep learning. The number of papers containing "deep learning" in their title or abstract jumped from just 20 in 2010 to over 25,000 in 2019, according to a Scopus assessment of the scientific literature database. The chart in Fig. 2 summarizes the rise of Deep Learning as compared to Traditional ML algorithms through a timeframe from 2000 to 2021.

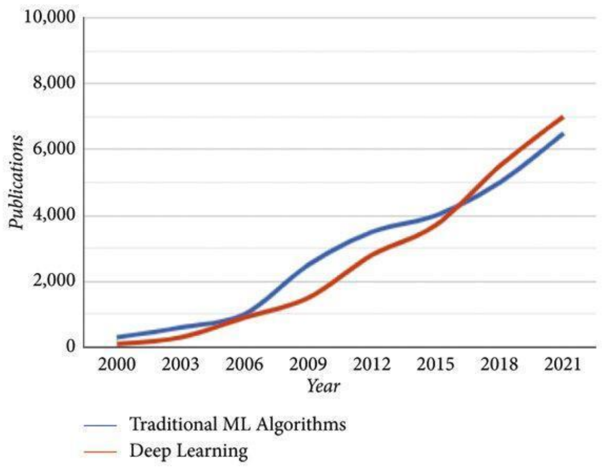


Fig. 11. The rise of Deep Learning as compared to Traditional ML algorithms

Deep Learning algorithms have demonstrated astounding performance on a variety of tasks, including autonomous driving, audio and picture recognition etc. Although traditional machine learning models are frequently straightforward and easy to understand, they fail to represent complicated relationships and necessitate substantial feature engineering.

Automatic feature extraction from data is one of Deep Learning's key advantages over conventional machine learning. When working with complicated, multidimensional data like voice or images, this is especially helpful. Manual feature engineering is often needed for traditional machine learning models, which can be laborious and error-prone. Specifically talking about IoT Malware detection, it is a crucial cybersecurity duty, locating dangerous software that might inadvertently damage a computer system. The static characteristics of the software, such as the file size, file type, or order of the instructions in the code, are frequently used in traditional Machine Learning algorithms for malware identification. These traits might not be enough to detect new, trickier varieties of malware, though. Deep Learning models for malware detection, in contrast, have the ability to automatically learn more complicated aspects from the data, such as patterns in the binary code or how the code communicates with the operating system. This can result in the more precise and consistent identification of novel malware types that conventional Machine Learning approaches might not be able to recognize.

Finally, compared to conventional Machine Learning models, Deep Learning models can swiftly adapt to new malware variants. Malware detection methods must be able to swiftly adapt to new malware types due to the continuously changing threat landscape. The following sections will highlight the novel and important Deep Learning based approaches used for IoT malware detection.

11. Deep Learning for IoT Malware Detection

11.1 Using Convolutional Neural Networks (CNNs) for IoT Malware Detection

CNN classification is a sort of deep learning technique that is extensively used in image and video analysis. It entails running data through multiple layers of convolutional and pooling procedures, which can learn key features and patterns in the input data automatically. CNN classification can be beneficial in detecting lightweight IoT (Internet of Things) malware since it can discover patterns in data that may be reflective of malicious behavior.

A CNN model can learn to recognize common patterns and features indicative of malicious behavior by training it on a large dataset of known IoT malware samples. The model can then immediately categorize new IoT devices or traffic as benign or potentially malicious based on these learned patterns when they are encountered. The below fig depicts an architecture of a typical CNN classification. It is possible to adapt and modify the CNN classification model's architecture for particular malware and IoT device varieties. In order to optimize the network's performance, the model must be trained and fine-tuned by being fed labeled data and having the weights of each layer updated.

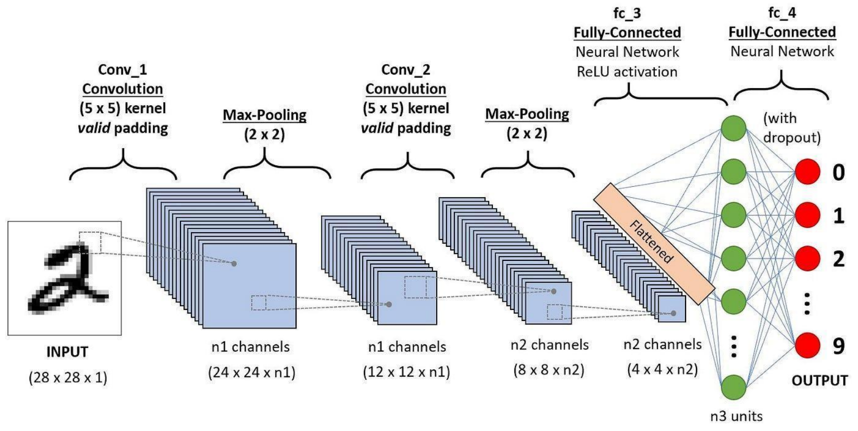


Fig. 12. A typical CNN for a classification task

CNN classification can be a powerful method for detecting IoT malware because it can learn to recognize patterns and attributes that humans may find difficult or time-consuming to discover and CNNs are more reliable than other machine learning algorithms because they are less vulnerable to noise and other sorts of interference.

11.1.1 Lightweight IoT Malware Detection Solution Using CNNs

Training the model requires a lot of resources, once it is trained, it becomes lightweight and is simple to install on IoT networks, contrary to the widespread belief that CNNs or machine learning techniques in general require a lot of resources to function. The researchers from ZaZa et al. [8] aim to create a lightweight convolutional neural network (CNN)-based malware detection system for IoT networks using (5G) and to optimize a CNN model to produce an ideal design that can aid in a novel method of detecting malware.

The proposed method is utilizing a central node to categorize files sent over an IoT network as either goodware or malware, with the categorization indicated. The experiment aimed to create a classifier that could distinguish between malware and useful software in IoT devices. The model was trained using a simple CNN with three categories of malware attacks: Gafgyt, Mirai and goodware. Over 5000 samples of IoT malware and 1000 samples of goodware datasets were used in the experiment. Two representations of the binary data are used: grayscale images and entropy-cluster images utilizing the Hilbert curve. The Hilbert curve is a remarkable construct in many aspects, but the fact that makes it valuable in the field of computer science is that it has good clustering properties.

According to the amount of processing power needed, the model can be trained either on the edge or in the cloud. The experiment's findings revealed a 98.4% accuracy rate for identifying malware and legitimate software. This model can be utilized in a variety of IoT applications that rely on 5G connectivity, such as agriculture and drone surveillance systems, and it can be optimized for inference on resource-constrained devices, such as microcontrollers and low-power IoT devices. To react to new malware threats, the proposed system can be improved by combining anomaly detection techniques and dynamic model retraining.

11.1.2 Using Deep CNNs for IoT Malware Detection

A class of neural networks called Deep Convolutional Neural Networks (DCNNs) is especially good at processing images and videos. They are made up of numerous convolutional, pooling, and fully connected layers that can autonomously learn to represent visual data in hierarchical fashion. When a DCNN is being trained, an input image is fed into the network and propagated through the layers to create an outcome prediction. The network then propagates the difference between the predicted output and the actual label in order to modify the weights and biases of the neurons.

The ability of DCNNs to automatically learn features from the unprocessed binary data, without the need for manually-engineered feature design, is one of their benefits for malware detection. However, the network's design and hyperparameters, as well as the model's performance and interpretability, are all influenced by the caliber and applicability of the training data. For each unique malware detection issue, the DCNN model must be meticulously created and assessed. Malware detection using DCNNs with malware visualization [13] has been demonstrated to be effective, with some studies showing detection rates of over 99%. The network design and the caliber and applicability of the preprocessed data, however, are crucial elements that can affect the model's functionality and interpretability. Based on the unique issue at hand and the available resources, it is crucial to choose the network architecture and data representation with care.

11.1.3 Classify IoT malware using CNN in cloud environment with DAIMD scheme:

Jeon et al. [18] proposed a technique called Dynamic Analysis for IoT Malware Detection for finding malware in Internet of Things (IoT) devices through DAIMD scheme. DAIMD is an acronym that stands for "Dynamic Analysis-based Intelligent Malware Detection." It is a cloud-based system that analyzes malware risks to Internet of Things (IoT) devices with low hardware resources. Its major function is to detect malware by employing unique analysis and detection techniques to construct a Convolutional Neural Network (CNN) model. DAIMD collects a variety of operations from memory, network, processes, system calls, and virtual file systems to spot malicious behavior on embedded Linux-based Internet of Things devices. These actions produce behavior images, which are then utilized to detect and categorize IoT malware using a convolution neural network (CNN) model called ZFNet. DAIMD is a useful strategy, according to the study's findings. As DAIMD's approach is an all in one functions of debugging, feature extraction, feature pre-processing, feature selection, and feature classification.
Creating a dataset of IoT malware and benign files is the initial stage in developing the model. The dataset includes 401 benign files and 1,000 new and variant samples of IoT malware. Initially, Debugging is carried out using Interactive Disassembler (IDA) Pro analysis tools during the study of the execution files. In the feature extraction step, debugging-analyzed file behaviors and internal structures are used to extract signatures, which are characteristics of execution files. Memory, network, system call, virtual file system (VFS), and process are the features that DAIMD extracts as signatures of files and stores as a.csv files in Excel.

While, in the feature preprocessing phase, the feature data is compressed and converted into an image type to represent the enormous amount of feature data that has been collected from IoT malware and benign files. Before transforming the extracted feature data into a picture, the behavior feature metadata stage processes feature data that reflect execution file behaviors. Strings in the behavior feature data are turned into vectors during the digitization phase of the visualization process, and then RGB channels are produced based on the vectors. DAIMD uses ZFNet, a CNN model, to automatically identify and train representative features of behaviors in order to detect IoT malware by combining the feature selection and classification phases into a single phase. An RGB image created through feature preprocessing serves as the input datum for the CNN model, from which behavior features are identified and calculated in a matrix known as a feature map. By using the max pooling technique to minimize the resulting feature map dimensions, it performs a feature selection phase where it only extracts behavior feature values. The CNN model's last step involves training representative behavior features through classification to perform classification and determine if the dynamically examined execution file is malicious or benign.

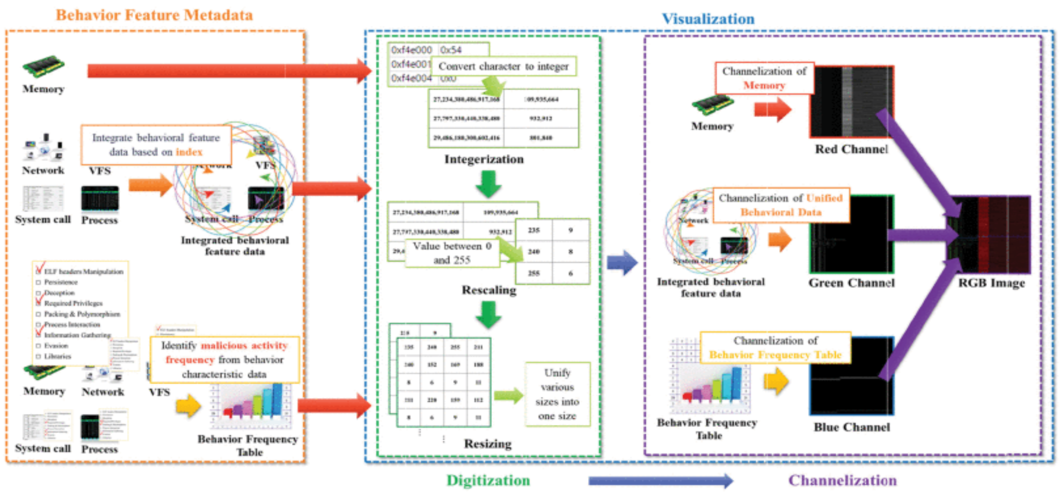

Behavior Feature Metadata
Memory
Network
VFS
System call
Process
Integrate behavioral feature data based on index
Integrated behavioral feature data
Identify malicious activity frequency from behavior characteristic data
Behavior Frequency Table
Visualization
Convert character to integer
Integerization
Value between 0 and 255
Rescaling
Unify various sizes into one size
Resizing
Channelization of Memory
Red Channel
Channelization of Unified Behavioral Data
Green Channel
Channelization of Behavior Frequency Table
Blue Channel
RGB Image
Digitization
Channelization

Fig. 14. Feature preprocessing phase of DAIMD.

Using feature data of behaviors according to different vector sizes and the ZFNet model, the authors trained the DAIMD model. The DAIMD model demonstrated the highest level of precision (99.28%) and the lowest likelihood of misclassifying malware as benign (0.63%) among the compared approaches in the paper. Therefore, it can be concluded that the DAIMD model is accurate in identifying variant malware that poses a threat to IoT devices.

11.1.4 Channel Boosted CNN for Malware Detection

A deep learning model known as a Channel Boosted Convolutional Neural Network (CNN) is frequently employed for image identification and classification applications. There are several ways in which this variant is different from its predecessor: standard CNNs. Convolutional layers in a channel-boosted CNN architecture are given more channels to boost the network's speed. The additional channels function as filters to improve the network's capacity to draw out useful characteristics from the input image. A Squeeze-and-Excitation (SE) module, a method for boosting particular channels based on their relevance, can be added to the Channel Boosted CNN architecture to further improve it.

Asam et al. [20] proposed a novel architecture using channel boosted and squeezed CNN for IoT malware detection and named it "IoT Malware Detection Architecture (iDMA)". The proposed iDMA architecture consists of various feature learning schemes through its modular blocks like (1) edge exploration and smoothing, (2) multipath dilated convolutional operations, and (3) channel squeezing and boosting in CNN to learn a diverse set of features. By implementing smoothing techniques in the split-transform-merge (STM) block, edge learns the local structural variations within malware classes. The global structure of malware patterns is identified using the multi-path dilated convolutional operation. Concurrently, channel squeezing and merging assists in controlling complexity and obtaining a variety of feature maps.

The proposed malware detection architecture, iMDA, better explored textural variation in the malware images by systematically using region and boundary information through the Avg and Max-pooling operations. Channel split-transform-merge technique helps to extract the features at different granularity. Incorporating the concepts mentioned earlier in CNNs improved the performance of the proposed architecture over the existing CNN models. The part (A) of Fig. 15. explains the proposed iDMA framework while the part (B) illustrates the layout of split-transform-merge (STM) block. The superior performance of the proposed iDMA architecture is verified as it achieves a remarkable accuracy of 98% and an F1 score of 94%.

Overall, the Channel Boosted CNN architecture is a powerful tool for image recognition tasks, and its effectiveness can be further improved by incorporating additional features such as the SE module. In the context of the proposed IoT malware detection architecture, the Channel Boosted CNN is designed to leverage its enhanced power to detect patterns associated with malware in IoT network traffic data.

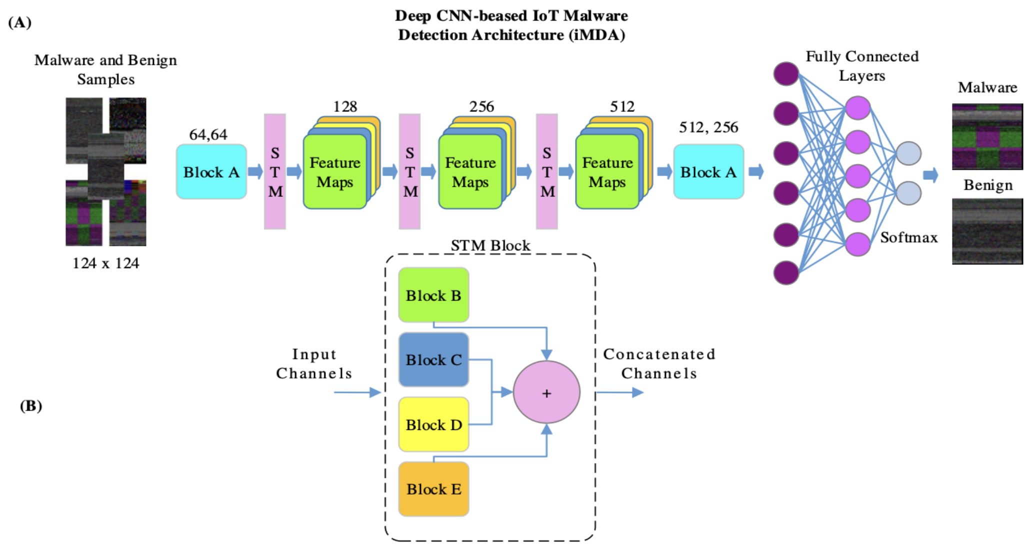


Fig. 15. The iDMA framework proposed by Asam et al. [20]

11.1.5 Deep Eigenspace Learning using CNN for Malware Detection

Azmoodeh et al. [9] proposed a novel approach to detect malware in Internet of Things (IoT) and Internet of Battlefield Things (IoBT) devices using Deep EigenSpace Learning. This method transforms the adjacency matrix of an OpCode graph into a vector space by leveraging eigenvectors and eigenvalues, enabling the representation of complex and non-linear data patterns. The resulting feature vectors are then input into a deep neural network for effective classification of malicious and benign applications. This approach has been shown to be highly effective in detecting malware and has the potential to improve the security of military IoT systems.

To generate the OpCode sequences required for this approach, the authors used a dataset consisting of 1078 benign samples and 128 malware samples, which were disassembled to produce their OpCode sequences. However, attackers may insert Junk code into their malware as an anti-forensic measure to evade detection by security mechanisms. To combat this issue, the authors employed the Class-Wise Information Gain (CIG) technique for class-aware feature selection, which extracts the most relevant features (OpCodes) from each sample and eliminates unimportant ones.

The authors found that a significant number of malware features are represented by 2-gram sequences, while benign application features are largely composed of 1-gram sequences. CIG values were calculated for all 1-gram and 2-gram sequence features, and the top 82 features were selected. These features were then converted into a graph representation, with each OpCode feature represented as a node/vertex, and the edge values calculated based on the distance between the OpCodes in the OpCode sequence. The distance between OpCodes was calculated based on their occurrence in the OpCode sequence. By using distance-based edge value calculation, the algorithm is less likely to be fooled by junk code because it looks at the overall structure of the OpCode sequence rather than just the immediate occurrence of specific OpCodes. This makes the algorithm more robust against junk code insertion attacks.

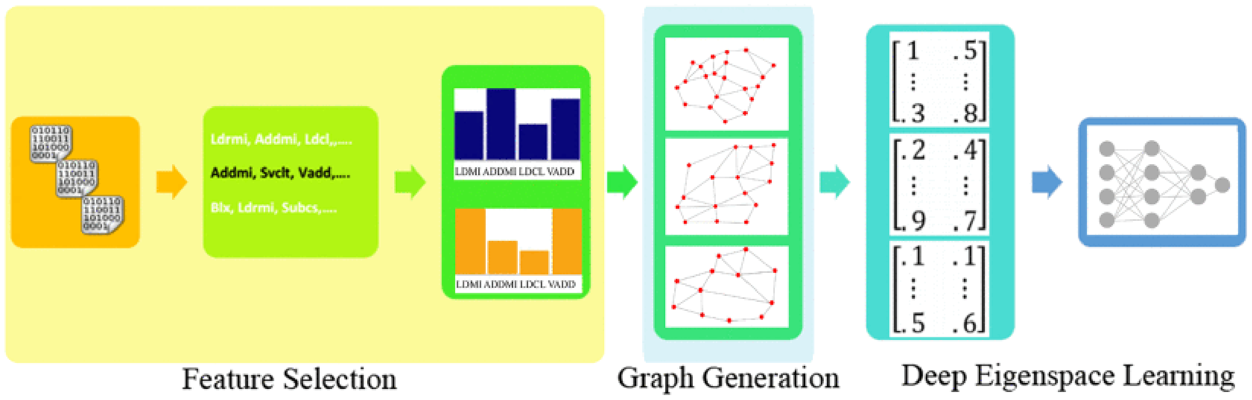


Fig. 16. Malware Detection with Deep Eigenspace Learning[9]

Graph embedding was used to convert the selected features into a small dimensional vector space. Eigenvalues and eigenvectors are utilized to transform the adjacency matrix of a graph into a vector space during graph embedding. [9] encompasses a more comprehensive explanation of how eigenvalues and eigenvectors are computed from the OpCode's adjacency matrix. The low-dimensional vector representations obtained were then fed to a CNN which achieved a malware detection rate of 99.68%. The approach proved to be robust against junk code insertion attacks, and the use of graph embedding and eigenvectors/eigenvalues allowed for analysis of the structure of the dataset's cumulative affinity matrix and improved classification performance.

Thus, the use of CNNs for malware detection has demonstrated exceptional accuracy in identifying malicious software instances. The advantage of using CNNs is their ability to effectively learn complex and non-linear patterns in data, making them well-suited for the detection of malware. Other deep learning techniques such as recurrent neural networks (RNNs) have also been used for similar tasks, and their effectiveness will be discussed in the following sections.

11.2 Using Recurrent Neural Networks (RNNs) for IoT Malware Detection

RNN (Recurrent Neural Network) is a form of deep learning algorithm used to evaluate sequential data, such as system call traces, natural language processing, speech recognition, and time series analysis. RNNs contain a feedback loop that allows them to handle sequential data while also remembering previous inputs, making them ideal for applications with temporal dependencies. RNNs and CNNs, both are forms of neural networks that can be used to identify IoT malware. However, they differ in regards to their architecture and the types of data that they are best suited to analyze.

CNNs are designed to process spatial data, such as images or audio signals, where the location of the data points is crucial, whereas RNNs are designed to process sequential data, where the order of the data points is significant. RNNs are better suited to processing time-series data, where the order of the data points is crucial, like network traffic or sensor readings. RNNs are useful for detecting IoT malware because they can identify the temporal dependencies between data points. When processing spatial data, such as images or audio signals, where the location of the data points is crucial, CNNs are better suited.

Both RNNs and CNNs are effective tools for identifying IoT malware, but they are made for various kinds of data and have unique advantages and disadvantages. The kind of neural network you deploy will rely on the data's nature and the particular requirements of the application. This section explores the application of RNNs for IoT malware detection.

11.2.1 Detecting IoT malware by System Calls using Baseline RNNs

Detecting IoT malware by monitoring system calls is an another way to improve IoT Security. The proposed approach from study, Shobana et al. [11] of the system, consists of three main components: dataset generation, data preprocessing, and RNN and the overall architecture is shown in Figure 17.
In the dataset generation phase, the IoT malware samples were collected from IoTPoT, a honeypot designed to capture telnet-based attacks for IoT devices. Benign samples were collected from the system files in Ubuntu system directories. The system calls for both malware and benign samples were generated using the strace tool, and the arguments and return values were removed to create a system call database consisting of 200 malware and 200 benign files.

In the data preprocessing phase, n-gram and TFIDF techniques were used to extract the most significant features from the system calls. The N-gram algorithm was used to extract meaningful patterns from the input attributes, with n set to 10. TFIDF was utilized to assign weights to each word depending on its overall number of occurrences. The weight value was likewise set to 10. Following these changes, the input was fed into RNN as a sparse matrix.

In the RNN phase, a one-layer RNN model with four input neurons, four hidden neurons, and one output neuron was used. The model was trained using system calls, and new system calls were classified during the testing phase. The hyperparameters of the RNN model were optimized through experiments, and the final values were as follows: five epochs, one hidden layer with a ReLU activation function, and an output layer with a sigmoid activation function. The optimizer used was Adam, and the loss function was binary cross-entropy.

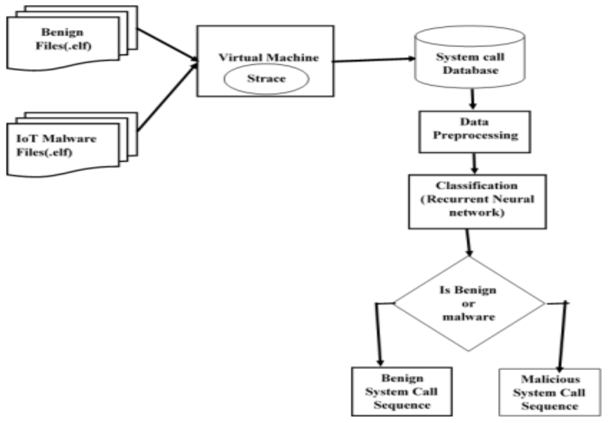


Fig. 17. Architecture of the model

The data used in this model consists of IoT malware samples and benign system files. The samples of IoT malware were obtained through IoTPoT, a honeypot designed to detect telnet-based attacks on IoT devices. The benign samples were obtained from system files on an Ubuntu system. The resulting system call database generated using strace tool includes 200 malware and 200 benign files. The data in this model consisted of processed system calls from IoT malware and benign system files. The study used a Recurrent Neural Network technique to distinguish IoT malware and benign files based on the frequency of system calls. The model's performance was evaluated using accuracy as the metric. Using only five epochs and a single hidden layer, the model achieved great accuracy and low loss, making it suitable for lightweight IoT devices.

RNNs can be a valuable tool for identifying malware in IoT utilizing system calls because they can examine temporal correlations and patterns in data that other machine learning algorithms may find difficult to capture. This work can be further enhanced by incorporating other effective deep learning approaches such as LSTM and Bi-LSTM.

11.2.2 Using Baseline LSTMs for IoT Malware Detection

LSTM (Long Short-Term Memory) is a form of recurrent neural network (RNN) architecture developed to overcome the constraints of ordinary RNNs in dealing with long-term dependencies in sequential data. They're ideal for studying time series data like IoT network traffic. Since they can selectively recall and forget information over long periods of time, LSTM networks are particularly well suited for sequence modeling and prediction tasks. The procedure involves simultaneously training a single LSTM model on several tasks, such as malware detection, device identification, and attack attribution.

Overall, LSTMs are a more developed and complex version of RNN that are better suited for managing extended sequences and selectively remembering or forgetting information. In many tasks, especially those with long-term dependencies, they have been found to perform better than conventional RNNs.

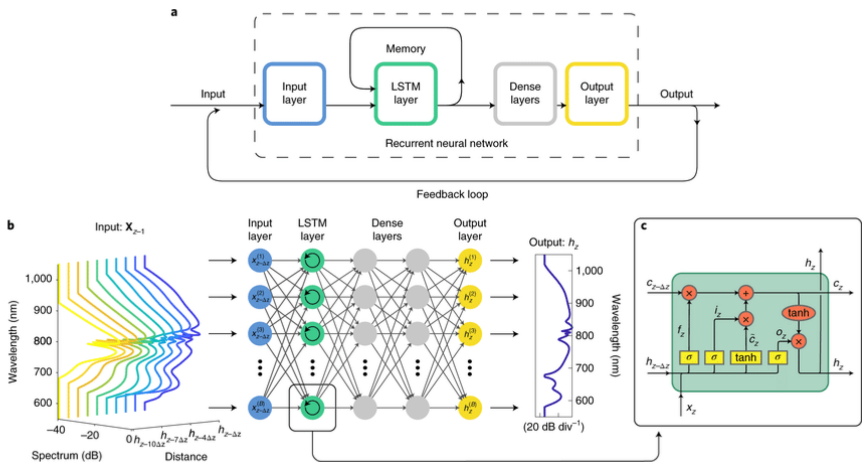


Fig. 18. An overview of LSTMs and their functioning pipeline

Fig. 18. above briefly explains the functioning of LSTMs for a sample sequence prediction task. Part (a) of the figure explains the architecture of an LSTM, part (b) explains the pipeline/flow of LSTM's training and inference process while part (c) illustrates an individual LSTM cell.

This section describes another approach with the early discussed LSTM models for detecting the IoT Malware. The initial step in the IoT malware hunting approach is to gather samples of both malware and benign ware specific to IoT devices to establish a dataset. The benign files were obtained from Linux Debian package repositories for applications that worked with Raspberry Pi II, while the malware samples were acquired from the VirusTotal Threat Intelligence platform.

The dataset is subsequently employed to extract OpCodes, which are low-level instructions utilized by the malware to execute malicious operations. A batch script was used to extract the OpCodes for each sample, which were then pruned to obtain a sequential list of OpCodes. The approach focused on ARM processors, with Cortex-A having the largest instruction set. A feature vector file is generated for each sample based on these OpCodes. The feature selection process for the dataset involved creating a word dictionary of unique OpCodes and generating a feature vector for each sample using various metrics such as binary encoding, TF-IDF, and count of occurrences. The values were sorted based on Information Gain (IG) to identify the most important features [10]. Additionally, techniques like word embedding and PCA were employed to address issues like the curse of dimensionality and sparsity in the dataset. The most frequently occurring OpCodes in both the malware and benign samples were identified and compared.

To detect IoT malware based on sequences of OpCodes, the researchers developed a deep learning model using Long Short Term Memory (LSTM), which is a Recurrent Neural Network (RNN) structure. They used TensorFlow as the backend structure and Scikit-learn for model evaluation. The LSTM structure consists of four neural networks and can learn dependencies between data. They also utilized bidirectional neural networks (BNN) to update neuron weights. The authors used LSTM architecture for malware detection and evaluated the performance of different LSTM configurations using 10-fold cross-validation. They tested the models on unseen malware samples with different OpCode distributions and found that the second LSTM configuration with two layers had the best performance, achieving an average accuracy of 97%. The authors also examined the effect of window sizes on classification performance and found that the optimum window size depends on the LSTM configuration used. Overall, the second LSTM configuration outperformed other approaches in classifying new malware samples with a classification accuracy of 94%.

Deep learning techniques such as LSTM and RNN have shown great potential in detecting malware based on OpCodes sequences. Integrating this approach into existing security systems can provide an additional layer of protection for IoT devices, helping to prevent cyber attacks and improve overall security.

11.2.3 Effective Multi-Task Deep Learning for IoT Malware Detection Using LSTMs

The method the authors utilized in Ali et al. [17] is a multi-task approach using LSTM architecture, which employs three gating units, namely the input unit, forget unit, and output unit, which are used to update, maintain, and delete the status of the LSTM cell to learn temporal correlations between historical values of each time-series feature and multivariate time-series. The proposed model learns two correlated tasks at the same time: malware classification and the type of IoT device classification. The proposed framework for multitask classification is presented in Figure 6. The dataset is initially separated into three modalities. Before the training and test set, a variety of preprocessing steps is carried out. Each modality is normalized, and the class unbalanced technique is used to ensure model stability. Finally, the model is trained on the chosen features and tested on the testing set.

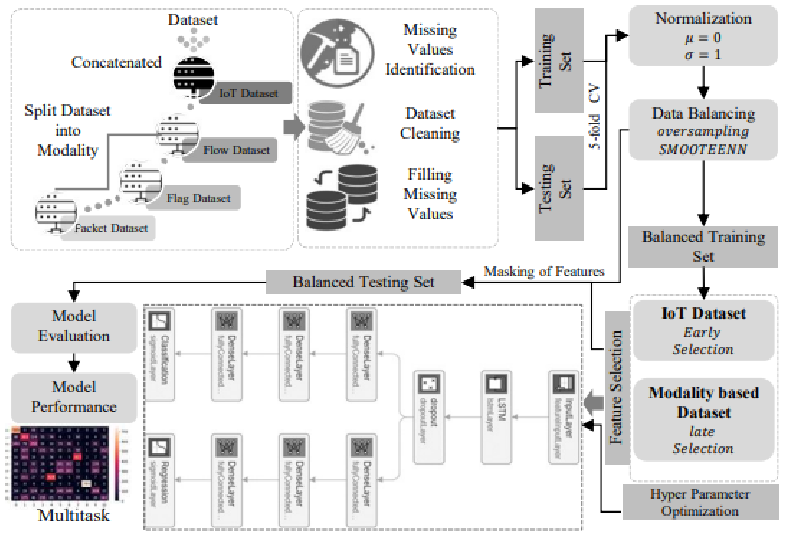


Fig. 19. The framework is presented for the multitask classification.

The multitasking LSTM model is evaluated based on its performance across various modalities and tasks (Binary and multitask classification). The performance of each task across modalities is evaluated based on accuracy. The first set of experiments used class imbalance and entire features in each modality, with the flow modality performing the best in both tasks. The second set of experiments examined the role of feature selection in training the model with the class imbalance and selected features in each modality. The flow modality still performed the best in both tasks, with feature selection leading to higher accuracy but lower stability. Finally, the multitask model achieved higher accuracy than the single-task model, with the flag-related dataset performing the best in both tasks. The combination had the best testing accuracies of 92.63%, 88.45%, and 95.83%, for task 1, task 2, and multitask classification, respectively.

The findings can be enhanced in the future by using more datasets obtained from other sources to extend the suggested multitask model, hence making the model resilient in the malware classification and detection problem. In addition, incorporating another module or task into the proposed model, such as recognizing the device based on IoT traffic, increases the protection and mitigation procedures.

While LSTMs are effective in modeling sequential data, they suffer from the inability to capture long-term dependencies due to the vanishing gradient problem. Bidirectional LSTMs (Bi-LSTMs) have been developed to address this issue by allowing information to flow both forward and backward through the network, thereby increasing the amount of information available for the prediction at each time step. In the context of malware detection, the bidirectional nature of Bi-LSTM is particularly useful for analyzing the temporal relationships between different sequences of opcodes.

11.2.4 Convolutional Recurrent Neural Networks with Bi-LSTM and MalDozer

Currently, there are three main types of Android malware detection techniques: static, dynamic, and hybrid detection. Machine learning-based Android malware detection is an emerging approach that involves dataset construction, feature engineering, model training, and model evaluation. However, the high cost of feature engineering in machine learning-based detection methods has led to a need for end-to-end automated feature engineering methods. We examine two resampling/preprocessing methods and two deep learning models proposed by Ren et al. [37] for end-to-end Android malware detection.

The resampling process involves converting variable-length classes.dex files of APKs into fixed-size sequences using resampling techniques. Two resampling methods: Nearest Neighbor Interpolation (NNI) and Average-pooling (AVGPOOL), are compared for downsampling and upsampling the input sequences. Nearest Neighbor Interpolation (NNI) is an image scaling technique that uses the nearest pixel to approximate the value of a new pixel, while AVGPOOL is a downsampling technique that takes the average of neighboring pixels to reduce image size.

Ren et al. [37] proposed two deep learning models, DexCNN and DexCRNN, for detecting Android malware. The DexCNN model is based on the TextCNN and Malconv models and uses fixed-size input sequences and an 8-dimensional vector to map each input byte. The One-dimensional Convolution layer extracts feature maps from the input sequence, and the Global Max-pooling layer reduces all feature maps to a fixed-size vector. The Softmax layer outputs binary classification results. DexCNN has no size limitation as it uses a pre-processed fixed vector as the input of the Convolution layer.

DexCRNN is a neural network architecture that combines CNNs and RNNs for malware detection in Android apps. It uses RNN to process sequence data but addresses the short-term memory problem by using LSTMs and GRU (Gated Recurrent Unit). It also tries bi-directional LSTM and GRU to account for bi-directional dependencies. The Recurrent layer allows the model to consider not only the previous state but also the future state when making predictions. Bi-directional models such as BiLSTM and BiGRU take this a step further by allowing the model to consider both the past and future states simultaneously.

DexCNN achieves a validation accuracy of 93.33%, while DexCRNN_BiLSTM was able to converge to the highest accuracy in all models, with a validation accuracy of 95.33%. The proposed methods are suitable for Android IoT devices due to no limitations on input file size, low resource consumption, and no manual feature engineering required. The methods could be expanded to cover additional malware detection tasks and incorporate more files as input to the model.

MalDozer is another Android malware detection framework that uses sequence mining through neural networks [16]. MalDozer uses word embedding techniques such as word2vec and GloVe to represent Android API method calls as dense vectors with semantic meaning. MalDozer's neural network is inspired by a neural network architecture used for sentence classification in NLP. The choice of using an NLP model for Android malware detection is due to the similarities between the sequence of Android API method calls and sentences in natural language. The efficiency and ability to run on resource-constrained devices make this design ideal for MalDozer.

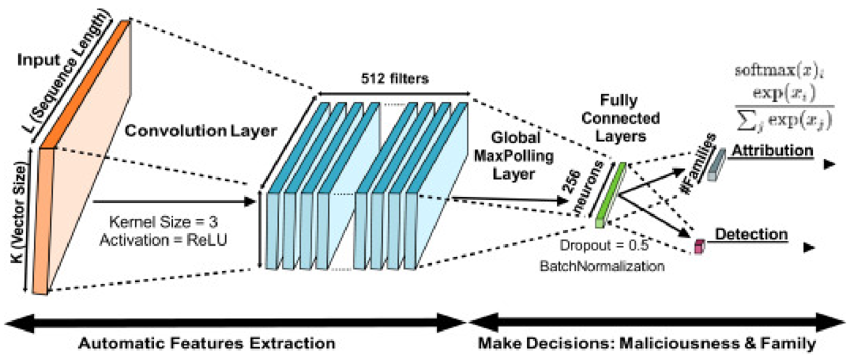


Fig. 20. Framework of MalDozer proposed in [16]

Maldozer performs excellently well in attributing malware to its family accurately, even with only a few samples. It is demonstrated that MalDozer is resilient to API evolution, which is crucial in the rapidly evolving mobile malware landscape MalDozer's ability to automatically extract and learn malicious and benign features during training makes it an attractive choice for detecting unknown malware, and its family attribution feature is useful in identifying specific malicious threats, such as ransomware, and taking appropriate precautions.

11.3 Ensemble of Deep Learning Techniques for IoT Malware Detection

11.3.1 Ensemble DL approaches (CNN + LSTM) for malware detection

Ensemble deep learning approaches, such as combining Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) models, have proven to be effective in detecting malware in Internet of Things (IoT) devices. In this section, we will briefly cover the benefits of using ensemble machine learning approaches for malware detection in IoT, as well as how CNN and LSTM models work together to achieve high levels of accuracy.

Machine learning (ML) models have been employed in the past to detect malware in IoT devices. However, some ensemble ML approaches have been proposed because single models can detect malware only with a limited degree of accuracy. As we have seen in the earlier sections, CNNs are capable of extracting information from an input signal or image and categorizing it into a specific class. CNNs can spot patterns in a file's binary code that point to the existence of malware when used for malware detection. Similarly, we also learnt earlier that Long Short-Term Memory (LSTM) models are another type of machine learning model that are particularly useful in detecting sequences and patterns over time. In the context of malware detection, LSTM models can analyze the sequence of operations in a program and identify patterns that indicate the presence of malware.

Ensemble methods can combine the predictions of multiple models and use a voting system to determine the final prediction. This approach can reduce the risk of false positives and false negatives, leading to more accurate malware detection. When integrated, CNN and LSTM models may identify malware in IoT devices with high levels of accuracy. The LSTM model can examine the flow of operations in a program and spot patterns that point to the existence of malware, whereas the CNN model can analyze the binary code of a file and extract characteristics that point to the presence of malware.

Riaz et al. [7] proposes a novel approach based on this same hybrid approach by using CNN and LSTM for malware detection in IoT. To accurately detect malware, the suggested framework is trained on a dataset comprising both benign and malicious traffic. The suggested approach's experimental findings, which demonstrate its effectiveness and high accuracy in detecting malware in IoT devices, are also presented in the study. The proposed CNN-LSTM approach achieves a remarkable accuracy of 99.8% and outperforms all the related works in the domain.

11.3.2 Deep Convolutional Neural Network with LSTM for Malware Detection

The demand for better detection systems is increasing to combat malware, particularly in healthcare. Customization of Android through third-party apps and ROMs can create security holes, and there are still undiscovered bugs in the official OS. To address this, Amin et al. [29] proposed an approach where a deep learning-based feature detector to classify Android and IoT samples as either benign or malicious. To accomplish this, first samples of Android and IoT malware from various sources are gathered. The samples were then decompiled, and opcodes were extracted from them. The feature detector takes input as the input data is the byte code extracted from APKs, and it consists of several layers that process this data to generate a final output that can be used for classification. The first step in the feature detection process is to convert the byte code into one-hot vectors, which are then used as input to the Chars2Vec model. This model generates embeddings for the byte code, which are vector representations of the input data. Embeddings are effective in reducing the dimensionality of data, thereby facilitating easier processing of large datasets.

The feature detector has multiple layers, including two convolutional layers with different filters, a pooling layer to reduce input size and prevent overfitting, additional convolutional layers for high-level feature learning, a dropout layer to regularize the network, and a final fully connected layer that reduces the final vector to two classes: malicious or benign, with a Softmax layer to output probabilities. The features learned by the detector were then classified using both a fully connected neural network and an LSTM network. The Softmax network provided a probabilistic classification of the sample, while the LSTM network was used to capture long-term dependencies in the sample's behavior.

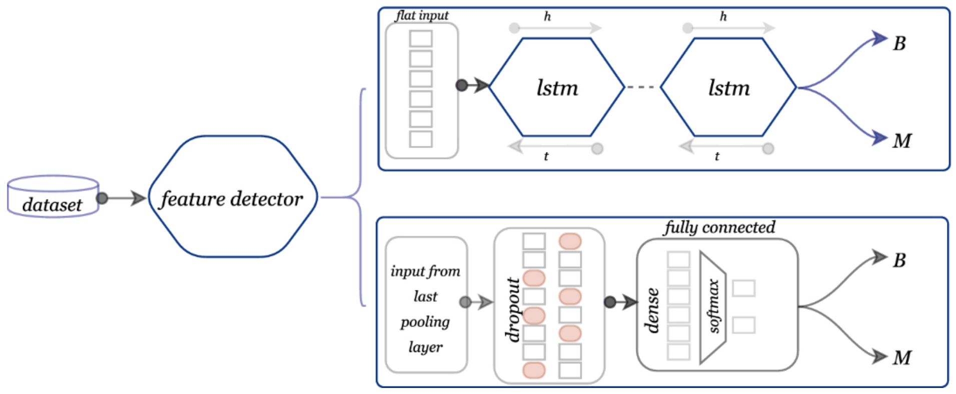


Fig. 21. Classification Flow with LSTM and FCN networks

The proposed method in [29] uses a Recurrent LSTM architecture to classify the APKs. Recurrent layers of Long Short-Term Memory (LSTM) networks are a type of neural network architecture that address the vanishing gradient problem in traditional recurrent neural networks (RNNs). LSTM networks control the flow of input information using gates, which are collections of mathematical functions and activations such as sigmoid, and perform point-wise multiplications on the input data. The gates are responsible for determining which pieces of information to remember or forget, and the final output is obtained by adding the output of the remember gate to the output of the LSTM's computation. The LSTM model takes input from embeddings, and employs a SpatialDropout1D before passing through two LSTM layers with 100 cell units. Finally, the output layer gives a binary-class classification for benign or malicious samples, with a Binary_Crossentropy loss function employed.

11.3.3 Multi-level DL framework for IoT Malware Classification using CNN and LSTM

Similar to the previous study on CNN and LSTM models, a different approach to malware classification is the multi-dimensional deep learning (DL) malware classification method was proposed to evaluate the effectiveness of CNN and LSTM deep learning algorithms in classifying IoT malware based on their byte-based image representation, which automatically learns static features from multiple representations of malware binaries and improves classification accuracy by combining multiple modalities. This approach can detect and attribute executable malware binaries to known IoT malware families. It utilizes static malware analysis techniques to create a multi-level classifier for IoT malware family attribution. By combining learned characteristics of malware from various data representations, this approach achieves higher classification performance than single modality DL algorithms. The multi-level DL framework for IoT malware classification is composed of three key components namely the image-based component, the string-based component, and the feature fusion and classification component.

The string-based and image-based components extract or learn relevant features from various representations of the malware, while the final component is responsible for combining the learned features into a shared representation, which is then used to generate the final classification result. The proposed framework by Dib et al. [30] for IoT malware classification uses a multi-level model that combines both image-based and string-based approaches. Su et al. [15] proposed that the use of a CNN-based method is generally viewed as efficient because it doesn't require the storage of training data for classification. The classification process using a CNN is quite straightforward and involves only basic mathematical operations such as summation and activation.

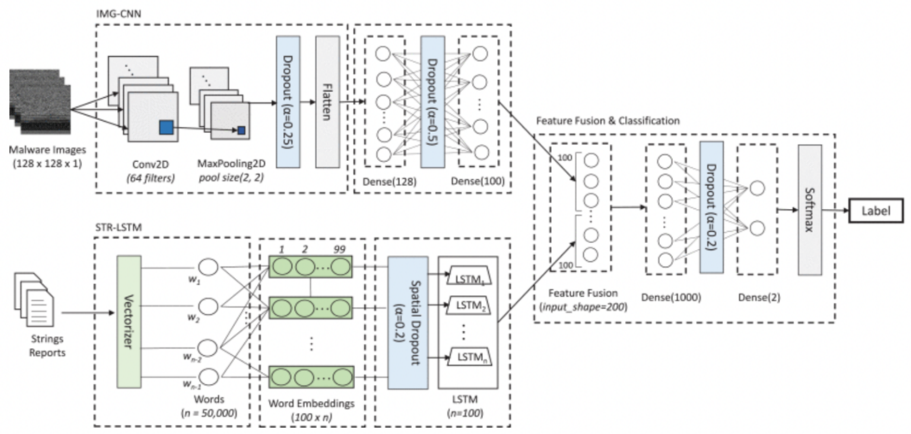


Fig. 22 . Multilevel Deep Learning Malware Classification System by Dib et al. [30]

The image-based part of the framework converts malware binary files into grayscale images, with each byte of the binary file being arranged into a 2D array that forms the grayscale image. Xiao et al. [39] uses an approach that is effective in classifying malware samples using CNN and LSTM neural networks, with hyper-parameter tuning being performed to select the optimal architecture [30]. The string-based component uses reverse-engineering techniques to extract useful strings from the binary code. The Linux strings utility is used to extract strings with three or more bytes, which can provide information about the malware and its functionalities. The most common 50,000 words are tokenized, and embeddings encode their meaning. This approach is effective for analyzing IoT malware in the wild, but not suitable for obfuscated binaries. The fusion component combines features for a shared representation.

The CNN model was found to perform slightly better than the LSTM model, achieving an accuracy and macro F1-score of 97.2%. This multilevel model that combines both image-based and string-based approaches, provided a robust and precise classification model for IoT malware. The results showed that the proposed multilevel model outperforms the deep learning implementation of the image and string-based models. Furthermore, the performance of the feature-engineering approaches was found to be lower than that of the proposed multi-level DL approach, which achieved significantly high accuracy and F1-score that exceeded 99.5%.

In conclusion, ensemble machine learning techniques have been successful in identifying malware in IoT devices, such as integrating CNN and LSTM models. By pooling the predictions of various models, ensemble approaches can further increase the accuracy of malware detection which is crucial to ensuring the security of these devices given the growing threat of malware assaults on IoT devices.

12. Combining Machine Learning & Deep Learning for IoT Malware Detection

Integration of machine learning (ML) and deep learning (DL) approaches enhance accuracy and performance across a wide range of applications. As contrast to employing a single model or algorithm, ensemble approaches integrate numerous models or algorithms to provide superior results. Deep learning (DL) is a powerful technique for feature extraction because it can automatically learn and extract complex, hierarchical features from raw data. In contrast, traditional feature extraction methods in machine learning (ML) often require manual feature engineering, which can be time-consuming and may not capture all relevant features.

Convolutional Neural Networks (CNNs) are a type of deep learning algorithm that are intended to automatically learn features from the input data. In this paper, image-based data visualization are used to discuss several methods for categorizing IoT malware. Using packet sniffing software like Wireshark or tcpdump. EI- Sayed et al.[3] justifies that PCAP files are important tools for finding unwanted behavior, each of the PCAP files was converted to an RGB image using the online binary visualization application.

The first model under examination was a Two-Layer CNN, which featured two convolutional layers, a dropout of 20%, a fully connected, max-pooling layer, and three completely connected layers. The second model was a Four-Layer CNN with two fully connected layers and four convolutional layers, followed by batch normalization and max-pooling layers. Two previously trained models, VGG16 and ResNet50, which were initially trained on the ImageNet dataset, were also put to the test by the authors. Also, they experimented with a method in which features from a pre-trained model using ImageNet weights were retrieved and then put through a straightforward classifier like Logistic Regression, Support Vector Machine (SVM), or K-Nearest Neighbors (KNN). To demonstrate the difference between using a sophisticated model over a lightweight one, the authors chose ResNet50 and MobileNetv2 as pre-trained models for feature extraction.

In this study, four CNN models—two-layer, four-layer, VGG16, and ResNet50—are compared in terms of image size, complexity, and prediction accuracy. According to the results, VGG16 has the highest accuracy, but it necessitates a high image resolution and extensive training. In contrast, ResNet50 achieves the second-highest accuracy while using lower image resolution and more layers, which adds to its complexity. Whereas the Four-Layer CNN has good accuracy with low complexity, the Two-Layer CNN has the least accuracy with a moderate level of complexity.

On the basis of feature extractor, classifier, and prediction accuracy, the paper also examines three feature selection approaches, namely LR, SVM, and KNN. With the maximum accuracy over MobileNetv2 as a feature, SVM classifier outperforms LR and KNN classifiers. KNN classifier performs better with less features but can't attain greater accuracy than 87% with the fewest features. Table demonstrates that the SVM classifier outperforms MobileNetv2 as a feature extractor in terms of accuracy. Overall, the study sheds light on the trade-offs between picture resolution, complexity, and accuracy for various CNN models.

Table 3. Comparison of the Performance of LR, SVM & KNN Classifiers.

Metric	ResNet			MobileNet		
	LR	SVM	KNN	LR	SVM	KNN
A	88%	80%	80%	93%	94%	87%
P	87%	79%	79%	93%	93%	87%
R	87%	80%	80%	93%	94%	87%
F1	87%	80%	79%	93%	93%	87%

The study comes to the conclusion that SVM-based Ordinary Classifiers, as opposed to CNN models, are more suited for IoT devices with constrained resources. Therefore, from the above table it demonstrates that SVM outperforms alternative models in terms of training stability, training speed, resource requirements, and processing complexity. Furthermore author Sharma et al. [14] has discussed in depth about static, dynamic and hybrid analysis on the LR, SVM and KNN algorithms.

12.1. Novel Approach using Robust-NN and C4N followed by SVM

Taheri et al. [23] proposed a model, in their paper, they discussed two methods for classifying the malware: Robust-NN and a combination of convolutional neural networks. Followed by 1- nearest neighbors (C4N). In Robust-NN correcting the label of perturbed samples, the Robust-Nearest Neighbor method (Robust-NN) seeks to increase the accuracy of machine learning models. The program first determines the average distance to each sample's nearest neighbor by doing several random sample selections. The sample is added to the updated training set if the label of the closest neighbor matches the one on the chosen sample. If not, the search interval's whole sample population whose label matches that of the selected sample is added to the rectified training set. The machine learning model is then trained using the revised training set to increase accuracy.This uses euclidean distance to calculate the distance that is present in the samples.

While in 1-nearest neighbors (C4N) ,the technique combines the 1-nearest neighbor (1-NN) and convolutional neural network (CNN) classification algorithms to train a second CNN classification algorithm that can avoid misclassifying adversarial samples. To train a specific sample and modify the weights used in the CNN algorithm, the procedure begins by determining the total of the loss functions for both the 1-NN and CNN algorithms.

Then, the maximum output of the functions for 1-NN and CNN, and the anticipated label is taken into account as the associated sample label. The machine learning model is subsequently trained against adversarial cases using this updated training set. The objective of this strategy is to combine 1-NN and CNN to enhance the performance of the CNN classifier and prevent adversarial samples from being misclassified. The results suggested that classification accuracy of Robust-NN and C4N methods is better than those already in use. On the API features of datasets, the C4N approach outperforms Robust-NN method by averaging an accuracy of 94% to 96%.

12.2 Using Stacked AutoEncoders (SAEs) and the Traditional ML algorithms

Stacked AutoEncoders (SAEs) are Deep Learning architectures with multiple layers of autoencoders, which are essentially Artificial Neural Networks (ANNs) created to encode input data into a lower-dimensional representation and then decode it back to its original shape. In an SAE, by reconstructing the preceding layer's output, each layer is trained to acquire a different representation of the input data. The ultimate encoded version of the input data is the final layer's output. Stacked AutoEncoders (SAEs) and conventional machine learning (ML) algorithms are used in a behavior-based deep learning strategy to extract high-level features from raw data and then classify the data based on those features. SAE-DT are used to extract features from data, further using these features as an input to train a Decision Tree classifier.

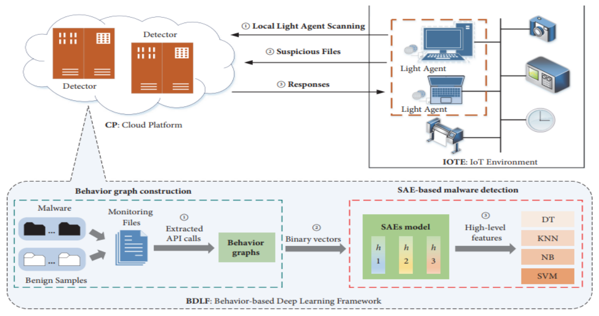
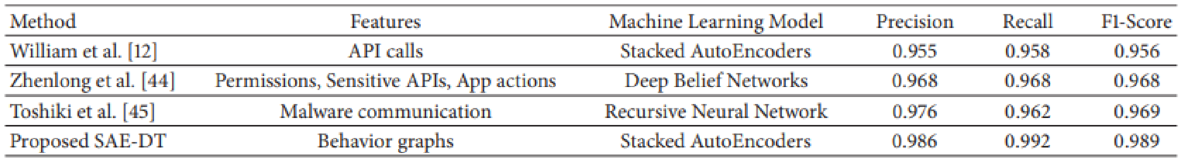


FIGURE 1: System overview.

Fig. 23. Proposed Behavior based Deep Learning Framework by Fei et al. [21]

In this paper, data normalization, noise removal, and format conversion are performed in the preparation stage of data. The preprocessed data is fed into the SAEs [21]. Principal component analysis (PCA) or correlation-based feature selection methods can be used to further process the compressed representation obtained from the SAEs to choose the most pertinent features for classification. To classify the data based on the behavior patterns discovered from the features, the chosen features are used to train the conventional ML algorithms, such as decision trees, support vector machines (SVMs), or random forests [36]. Deep learning and traditional ML complement each other well because the SAEs can capture the data's abstract and complicated features, while traditional ML algorithms can classify the data using these features according to clear and understandable rules. By using regularization methods like dropout or denoising, the SAEs can learn to extract robust features that are invariant to noise and variations in the data.

Table 4. The comparison between proposed SAE-DT approach and other methods



Method	Features	Machine Learning Model	Precision	Recall	F1-Score
William et al. [12]	API calls	Stacked AutoEncoders	0.955	0.958	0.956
Zhenlong et al. [44]	Permissions, Sensitive APIs, App actions	Deep Belief Networks	0.968	0.968	0.968
Toshiki et al. [45]	Malware communication	Recursive Neural Network	0.976	0.962	0.969
Proposed SAE-DT	Behavior graphs	Stacked AutoEncoders	0.986	0.992	0.989

By enabling parallel model training and deployment across numerous platforms, combining ML and DL models can increase scalability. Reduced overfitting and increased model consistency are two benefits of combining ML and DL models, which can enhance generalization. Combining the two methods can achieve an equilibrium between detection accuracy and interpretability. In conclusion, combining ML and DL techniques has the potential to increase accuracy, robustness, real-time detection, scalability, generalization, and interpretability for malware detection in IoT.

13. Combining Machine Learning and Blockchain Technology for IoT Malware Detection: A perspective

Machine Learning entails creating algorithms that can recognize patterns in data and learn from them to make predictions. As a result of its ability to scan enormous amounts of data and spot patterns related to malware, it has been extensively employed for malware detection. However, the centralization and vulnerability of machine learning models makes them prone to errors in spotting malware in IoT devices.

On the other hand, blockchain technology is a distributed ledger system that can offer a safe and decentralized platform for data sharing and keeping. It's been suggested as a way to improve the security of IoT devices by establishing a trustless network where information may be transferred and stored securely. Researchers have suggested a decentralized method for IoT malware detection that combines machine learning and blockchain technologies. In this method, machine learning models are built and deployed on a blockchain network where data from IoT devices is sent in order to detect malware trends. A permanent record of the detection process is subsequently created by storing the results on the blockchain.

One of the main benefits of this approach is that it can leverage the advantages of both machine learning and blockchain technology to enhance the security of IoT devices. The machine learning models can analyze large amounts of data and detect malware patterns, while the blockchain network can provide a secure and decentralized platform for storing and sharing data. Kumar et al. [12] proposes a novel multimodal approach that combines both machine learning and blockchain technology to detect malware in android setup in IoT systems. The proposed technique is implemented using a sequential approach, which includes clustering, classification, and blockchain. Machine learning automatically extracts the malware information using clustering and classification techniques and stores the information into the blockchain. Thereby, all malware information stored in the blockchain history can be communicated through the network, and therefore any latest malware can be detected effectively.

Fig. 24. Below illustrates the working of the proposed multi-modal framework combining machine learning and blockchain technology. The implementation of the clustering technique includes calculation of weights for each feature set, development of parametric study for optimization and simultaneously iterative reduction of unnecessary features having small weights. The classification algorithm is implemented to extract the various features of Android malware using naive Bayes classifier.

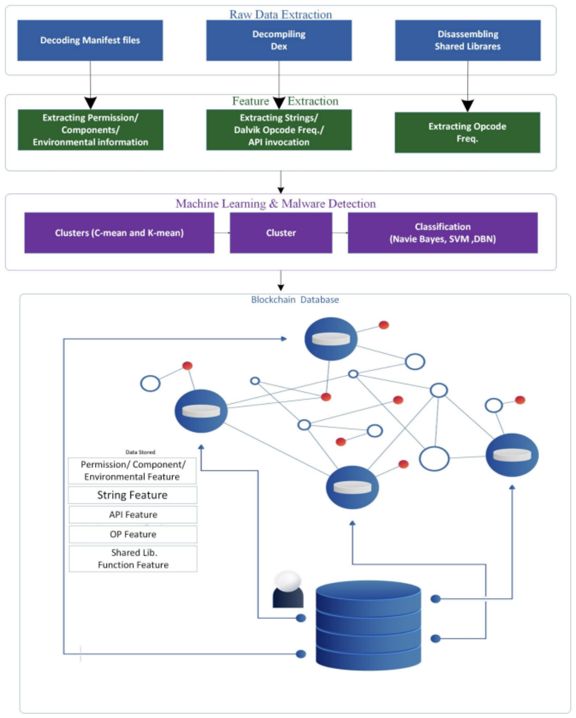

Raw Data Extraction
Decoding Manifest files
Decompiling Dex
Disassembling Shared Libraries
Feature Extraction
Extracting Permission/ Components/ Environmental information
Extracting Strings/ Dalvik Opcode Freq./ API invocation
Extracting Opcode Freq.
Machine Learning & Malware Detection
Clusters (C-mean and K-mean)
Cluster
Classification (Navie Bayes, SVM ,DBN)
Blockchain Database
Data Stored
Permission/ Component/ Environmental Feature
String Feature
API Feature
OP Feature
Shared Lib. Function Feature

Fig. 24. Architecture of the multimodal framework proposed by Kumar et al. [12]

The proposed framework uses the permissioned blockchain to store authentic information of extracted features in a distributed malware database blocks to increase the runtime detection of malware with more speed and accuracy, and further to announce malware information for all users.

Combining blockchain and machine learning for IoT malware detection is not without its difficulties, though. It might be difficult to install machine learning models on IoT devices with limited resources due to the high computational cost of training models on huge amounts of data. In addition, the high throughput demands of IoT networks may make it difficult for blockchain networks to scale. To address these challenges, researchers have proposed several approaches for optimizing machine learning models and blockchain networks for IoT malware detection. These include using lightweight machine learning models, optimizing the blockchain consensus mechanism, and using off-chain storage solutions to reduce the computational burden on IoT devices.

Overall, combining machine learning and blockchain technology for IoT malware detection is a promising approach that has the potential to enhance the security of IoT devices. However, there are still some challenges in this novel approach that need to be addressed, and further research is needed to optimize the approach for practical applications in real-world IoT networks.

14. Future steps after detecting malware in IoT systems

The Internet of Things (IoT) has transformed the way we interact with technology, with billions of devices connected to the internet, enabling them to communicate with each other and share data. However, the proliferation of IoT devices has also led to an increase in the number of cyber attacks on these devices. Once malware has been detected in an IoT system, several steps can be taken to mitigate the damage and prevent future attacks.

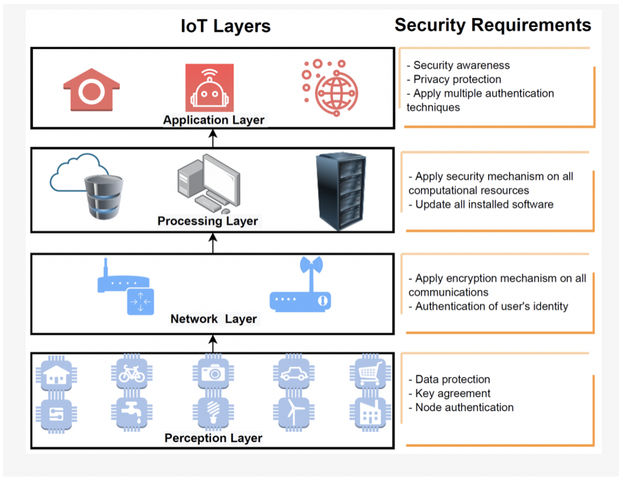

IoT Layers
Security Requirements
Application Layer
- Security awareness
- Privacy protection
- Apply multiple authentication techniques
Processing Layer
- Apply security mechanism on all computational resources
- Update all installed software
Network Layer
- Apply encryption mechanism on all communications
- Authentication of user's identity
Perception Layer
- Data protection
- Key agreement
- Node authentication

Fig . 25. Internet of Things (IoT) Architecture and Security Requirements

Step 1: Isolate the Infected Devices

The first step in responding to a malware attack on an IoT system is to isolate the infected devices. This can help prevent the malware from spreading to other devices in the network. In some cases, it may be necessary to disconnect the devices from the network altogether to prevent further damage.

Step 2: Identify the Type of Malware

Once the infected devices have been isolated, the next step is to identify the type of malware that has infected the system. There are many different types of malware that can infect IoT systems, including viruses, worms, Trojans, and botnets. Each type of malware operates differently and may require a different response.

Step 3: Determine the Extent of the Infection

After identifying the type of malware, the next step is to determine the extent of the infection. This can be done by conducting a thorough scan of the system to identify all infected devices and files. It is important to identify all infected devices, including those that may not be immediately visible, such as devices that are turned off or disconnected from the network.

Step 4: Remove the Malware

Once the infected devices have been identified, the next step is to remove the malware. This can be a complex process that may require specialized tools and expertise. In some cases, it may be necessary to wipe the device clean and reinstall the operating system and applications from scratch.

Step 5: Update Security Measures

After removing the malware, the next step is to update the security measures on the system to prevent future attacks. This can include updating software and firmware, installing security patches, and changing passwords and access controls.

Step 6: Monitor the System

Once the system has been cleaned and secured, it is important to monitor it closely for any signs of future attacks. This can be done using a combination of automated tools and manual monitoring. It is important to be vigilant and to respond quickly to any suspicious activity, such as unusual network traffic or unauthorized access attempts.

Step 7: Educate Users

Finally, it is important to educate users about the risks of malware and how to prevent future attacks. This can include training on how to recognize and avoid phishing scams, how to use strong passwords, and how to update software and firmware regularly.

15. Conclusion & Learnings

Machine learning has emerged as a powerful tool for detecting malware in IoT devices. By training machine learning algorithms on large datasets of device behavior, network traffic, and sensor data, patterns indicating the presence of malware can be identified. One of the key advantages of machine learning is its ability to adapt to new and emerging threats, making it a valuable tool in the fight against constantly evolving cyber threats. The use of machine learning (ML) for detecting malware in its early stages is effective in reducing the severity of attacks by detecting patterns in data that are indicative of malware. This approach is particularly helpful in detecting newer or modified forms of malware and allows for prompt containment of the infection and recovery from any damage caused. However, it is important to keep the ML models updated to stay current with evolving threats, and the models may generate false positives or false negatives that require careful monitoring.

Machine learning (ML) is also used to detect malware on resource-constrained devices, which is a hopeful strategy that can improve the security of such devices. ML helps in identifying malware patterns and irregularities in real-time, even on devices with limited processing power and memory. This is beneficial in preventing malware attacks from jeopardizing the devices or disrupting the data they store. Nonetheless, there are difficulties linked with implementing ML on resource-constrained devices, such as maintaining the balance between accuracy and efficiency, as well as the possibility of producing false positives or false negatives.

Integrating methods like hybrid analysis helps these ML models to provide high accuracy. The utilization of hybrid analysis for malware detection offered several advantages over traditional signature-based detection methods. By combining static and dynamic analysis techniques and incorporating machine learning algorithms, hybrid analysis can provide a more accurate and dependable detection of malware. Its comprehensive approach allows it to identify both known and unknown malware patterns and behaviors, leading to a more efficient and effective mitigation of threats.

Subtle or complex malware that exhibit behaviors similar to normal device behavior may be difficult to detect using machine learning algorithms. Additionally, factors such as the quality of the training dataset and the complexity of the model used can affect the accuracy of machine learning algorithms. Deep learning, such as deep neural networks, can help address the limitations of machine learning for IoT malware detection by learning more complex representations of data than traditional machine learning algorithms. This allows deep learning algorithms to detect more subtle and complex patterns in IoT device data, leading to improved accuracy in malware detection. Furthermore, deep learning algorithms can analyze multiple sources of data simultaneously, making it easier to identify sophisticated attacks that involve multiple stages or vectors of attack.

Some of the Deep learning (DL) approaches offered several advantages for malware detection over traditional methods. One such approach is using Long Short-Term Memory (LSTM) for malware detection, that analyzes large volumes of data and identifies complex patterns, resulting in higher accuracy. LSTM also adapts to new types of malware by learning from new data and reducing false positives by analyzing program behavior. It also detects zero-day attacks and automates the detection process, saving time and resources for cybersecurity teams. These benefits make LSTM a valuable tool in the fight against constantly evolving malware threats.

Another DL approach is a multi-level deep learning framework for IoT malware classification, that classifies malware at various levels, improving accuracy and detail, and enabling the identification and isolation of specific malware types. Deep learning algorithms facilitate automatic feature extraction, making classification more efficient and precise, especially for complex and evolving IoT malware. Additionally, the framework's adaptability to changes in the IoT environment can increase its effectiveness in detecting and classifying emerging threats. Overall, the use of a multi-level DL framework is a promising approach that can enhance IoT malware classification accuracy and detail, reduce manual feature engineering, and improve adaptability to changes in the IoT environment.

Ensemble approaches combining 2-3 Machine Learning algorithms have the potential to detect malware with high degree of robustness as compared to individual algorithms since single models can usually detect malwares with only a limited level of accuracy. Ensembling different models is like combining the best of each world, which generally results in a better performance. We also learnt that as compared to a standard CNN, a channel-boosted CNN has the potential to capture more intricate patterns which increases its precision in image recognition tasks. A method for boosting particular channels based on their relevance called Squeeze-and-Excitation (SE) module, can be added to the Channel Boosted CNN architecture to further improve it. Lastly, we also discussed some novel approaches combining two different fields: machine learning and blockchain technology for malware detection, which proved to be a promising approach with ML being used for feature extraction while blockchain for storing important information regarding the extracted features for future use.

Deep learning convolutional neural networks (CNNs) also play a crucial role in identifying malware in IoT devices as they can detect characteristics in network traffic that are linked to malware, such as strange data flows and questionable connections. CNNs are highly effective and adaptable, making them suitable for detecting malware in real-time and in a variety of data sources. By using CNNs, the security of IoT devices can be improved, and they can be safeguarded against potential cyber threats. Despite the advantages of deep learning, there are also drawbacks. Their deployment in resource-constrained IoT devices faces significant challenges due to high computational requirements and limited memory and power. Additionally, deep learning models can be challenging to interpret, making it difficult to understand how the model arrived at a particular decision.

In conclusion, machine learning and deep learning are valuable tools for detecting malware in IoT devices. Machine learning is well-suited for identifying known and emerging threats, while deep learning can detect more subtle and complex malware. However, both approaches have their limitations and require careful consideration and expertise to use effectively. With the continued growth of IoT devices and evolving cyber threats, the development of advanced machine learning and deep learning techniques will be essential for securing these devices against emerging threats.

The future of machine learning and deep learning for malware detection in IoT devices is promising, as the number of IoT devices continues to grow rapidly and the threat of cyber attacks becomes increasingly prevalent. Here are some potential future scopes:

- Improved accuracy: Machine learning and deep learning algorithms will continue to improve in accuracy, allowing them to detect more sophisticated and subtle malware in IoT devices. This can help prevent attacks that may have otherwise gone undetected.
- Real-time detection: As the volume of IoT data continues to increase, there will be a growing need for real-time detection of malware. Machine learning and deep learning algorithms can be trained to analyze data streams in real-time, allowing for rapid detection and response to threats.
- Anomaly detection: Machine learning and deep learning algorithms can be used to identify anomalous behavior in IoT devices, which may indicate the presence of malware. This approach can be particularly effective for detecting zero-day attacks, which are not yet known to security experts.
- Combining multiple techniques: Machine learning and deep learning can be combined with other techniques, such as signature-based detection, to improve the accuracy of malware detection in IoT devices.

- Explainable AI: As machine learning and deep learning models become more complex, there will be a growing need for models that are transparent and explainable. Explainable AI techniques can help security experts understand how a model arrived at a particular decision, which can be important for identifying and mitigating false positives.
- Resource Optimization: Deep learning models have shown great promise in detecting malware in IoT devices, but their deployment in resource-limited environments poses significant challenges, including high computational demands and limited memory and power. Therefore, there is a need for future research to concentrate on developing efficient deep learning models and optimizing their deployment in resource-constrained IoT environments.

Overall, the future of machine learning and deep learning for malware detection in IoT devices is exciting, with potential applications in a wide range of industries, from healthcare to manufacturing to transportation. As the field continues to advance, it is likely that we will see more sophisticated and accurate methods for detecting and responding to malware in IoT devices.

REFERENCES

1. F. Hussain, R. Hussain, S. A. Hassan and E. Hossain, "Machine Learning in IoT Security: Current Solutions and Future Challenges," in IEEE Communications Surveys & Tutorials, vol. 22, no. 3, pp. 1686-1721, thirdquarter 2020, doi: 10.1109/COMST.2020.2986444.

2. S. Madan and M. Singh, "Classification of IOT-Malware using Machine Learning," 2021 International Conference on Technological Advancements and Innovations (ICTAI), Tashkent, Uzbekistan, 2021, pp. 599-605, doi: 10.1109/ICTAI53825.2021.9673185.

3. R. El-Sayed, A. El-Ghamry, T. Gaber and A. E. Hassanien, "Zero-Day Malware Classification Using Deep Features with Support Vector Machines," 2021 Tenth International Conference on Intelligent Computing and Information Systems (ICICIS), Cairo, Egypt, 2021, pp. 311-317, doi: 10.1109/ICICIS52592.2021.9694256.

4. A. Kumar and T. J. Lim, "EDIMA: Early Detection of IoT Malware Network Activity Using Machine Learning Techniques," 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), Limerick, Ireland, 2019, pp. 289-294, doi: 10.1109/WF-IoT.2019.8767194.

5. Santhadevi D, Janet B, “IoT Malware Detection using Machine Learning Ensemble Algorithms”, International Journal of Advanced Science and Technology (IJAST), vol. 29, no. 10s, pp. 8006-8016, Jun. 2020.

6. Achary, Rathnakar, and Chetan J. Shelke. "Malware Attack Detection on IoT Devices Using Machine Learning." In Smart Data Intelligence: Proceedings of ICSMDI 2022, pp. 11-22. Singapore: Springer Nature Singapore, 2022.

7. S. Riaz et al., “Malware Detection in Internet of Things (IoT) Devices Using Deep Learning,” Sensors, vol. 22, no. 23, p. 9305, Nov. 2022, doi: 10.3390/s22239305.

8. A. M. N. Zaza, S. K. Kharroub and K. Abualsaud, "Lightweight IoT Malware Detection Solution Using CNN Classification," 2020 IEEE 3rd 5G World Forum (5GWF), Bangalore, India, 2020, pp. 212-217, doi: 10.1109/5GWF49715.2020.9221100.

9. A. Azmoodeh, A. Dehghantanha and K. -K. R. Choo, "Robust Malware Detection for Internet of (Battlefield) Things Devices Using Deep Eigenspace Learning," in IEEE Transactions on Sustainable Computing, vol. 4, no. 1, pp. 88-95, 1 Jan.-March 2019, doi: 10.1109/TSUSC.2018.2809665.

10. Hamed HaddadPajouh, Ali Dehghantanha, Raouf Khayami, Kim-Kwang Raymond Choo,A deep Recurrent Neural Network based approach for Internet of Things malware threat hunting,Future Generation Computer Systems,Volume 85,2018,Pages 88-96,ISSN 0167-739X,https://doi.org/10.1016/j.future.2018.03.007.

11. M. Shobana and S. Poonkuzhali, "A novel approach to detect IoT malware by system calls using Deep learning techniques," 2020 International Conference on Innovative Trends in Information Technology (ICITIIT), Kottayam, India, 2020, pp. 1-5, doi: 10.1109/ICITIIT49094.2020.9071531.

12.Kumar, Rajesh & Zhang, Xiaosong & Wang, Wen & Khan, Riaz & Kumar, Jay & Sharif, Abubakar. (2019). A Multimodal Malware Detection Technique for Android IoT Devices Using Various Features. IEEE Access. PP. 1-1. 10.1109/ACCESS.2019.2916886.

13. Hamad Naeem, Farhan Ullah, Muhammad Rashid Naeem, Shehzad Khalid, Danish Vasan, Sohail Jabbar, Saqib Saeed,Malware detection in industrial internet of things based on hybrid image visualization and deep learning model,Ad Hoc Networks,Volume 105,2020,102154,ISSN 1570-8705,https://doi.org/10.1016/j.adhoc.2020.102154.

14. S. Sharma and S. Bharti, "Malware Analysis using Ensemble Techniques: A Machine Learning Approach," 2021 International Conference on Artificial Intelligence and Machine Vision (AIMV), Gandhinagar, India, 2021, pp. 1-5, doi: 10.1109/AIMV53313.2021.9670949.

15. J. Su, D. V. Vasconcellos, S. Prasad, D. Sgandurra, Y. Feng and K. Sakurai, "Lightweight Classification of IoT Malware Based on Image Recognition," 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), Tokyo, Japan, 2018, pp. 664-669, doi: 10.1109/COMPSAC.2018.10315.

16. ElMoataz Billah Karbab, Mourad Debbabi, Abdelouahid Derhab, Djedjiga Mouheb, MalDozer: Automatic framework for android malware detection using deep learning, Digital Investigation, Volume 24, Supplement, 2018, Pages S48-S59, ISSN 1742-2876, doi: https://doi.org/10.1016/j.diin.2018.01.007.

17. S. Ali, O. Abusabha, F. Ali, M. Imran and T. ABUHMED, "Effective Multitask Deep Learning for IoT Malware Detection and Identification Using Behavioral Traffic Analysis," in IEEE Transactions on Network and Service Management, 2022, doi: 10.1109/TNSM.2022.3200741.

18. J. Jeon, J. H. Park and Y. -S. Jeong, "Dynamic Analysis for IoT Malware Detection With Convolution Neural Network Model," in IEEE Access, vol. 8, pp. 96899-96911, 2020, doi: 10.1109/ACCESS.2020.2995887.

19. Nakahara, Masataka & Okui, Norihiro & Kobayashi, Yasuaki & Miyake, Yutaka. (2020). Machine Learning based Malware Traffic Detection on IoT Devices using Summarized Packet Data. 78-87. 10.5220/0009345300780087.

20. Asam M, Khan SH, Akbar A, Bibi S, Jamal T, Khan A, Ghafoor U, Bhutta MR. IoT malware detection architecture using a novel channel boosted and squeezed CNN. Sci Rep. 2022 Sep 15;12(1):15498. doi: 10.1038/s41598-022-18936-9. PMID: 36109570; PMCID: PMC9477830.

21. Fei Xiao, Zhaowen Lin, Yi Sun, Yan Ma, "Malware Detection Based on Deep Learning of Behavior Graphs", *Mathematical Problems in Engineering*, vol. 2019, Article ID 8195395, 10 pages, 2019. https://doi.org/10.1155/2019/8195395

22. Dartel, B.V. (no date) *MALWARE DETECTION IN IOT DEVICES USING MACHINE LEARNING*.

23. Taheri, R., Javidan, R., & Pooranian, Z. (2021). Adversarial android malware detection for mobile multimedia applications in IoT environments. Multimedia Tools and Applications, 80(24), 16713-16729.

24. Wan, T.-L., Ban, T., Cheng, S.-M., Lee, Y.-T., Sun, B., Isawa, R., Takahashi, T., & Inoue, D. (2020). Efficient detection and classification of internet-of-things malware based on byte sequences from executable files. IEEE Access, 8, 103068-103078.

25. Abusnaina, Ahmed & Anwar, Afsah & Alshamrani, Sultan & Alabduljabbar, Abdulrahman & Jang, RhongHo & Nyang, Daehun & Mohaisen, David. (2021). ML-based IoT Malware Detection Under Adversarial Settings: A Systematic Evaluation.

26. Nakhodchi, S., Upadhyay, A., Dehghantanha, A. (2020). A Comparison Between Different Machine Learning Models for IoT Malware Detection. In: Karimipour, H., Srikantha, P., Farag, H., Wei-Kocsis, J. (eds) Security of Cyber-Physical Systems. Springer, Cham. https://doi.org/10.1007/978-3-030-45541-5_10

27. Al-Sarem M, Saeed F, Alkhammash EH, Alghamdi NS. An Aggregated Mutual Information Based Feature Selection with Machine Learning Methods for Enhancing IoT Botnet Attack Detection. Sensors (Basel). 2021 Dec 28;22(1):185. doi: 10.3390/s22010185. PMID: 35009725; PMCID: PMC8749651.

28. .I. M. M. Matin and B. Rahardjo, "Malware Detection Using Honeypot and Machine Learning," 2019 7th International Conference on Cyber and IT Service Management (CITSM), Jakarta, Indonesia, 2019, pp. 1-4, doi: 10.1109/CITSM47753.2019.8965419.

29. Amin, M., Shehwar, D., Ullah, A. et al. A deep learning system for healthcare IoT and smartphone malware detection. Neural Comput & Applic 34, 11283–11294 (2022). https://doi.org/10.1007/s00521-020-05429-x

30. M. Dib, S. Torabi, E. Bou-Harb and C. Assi, "A Multi-Dimensional Deep Learning Framework for IoT Malware Classification and Family Attribution," in IEEE Transactions on Network and Service Management, vol. 18, no. 2, pp. 1165-1177, June 2021, doi: 10.1109/TNSM.2021.3075315.

31. K. D. T. Nguyen, T. M. Tuan, S. H. Le, A. P. Viet, M. Ogawa and N. L. Minh, "Comparison of Three Deep Learning-based Approaches for IoT Malware Detection," 2018 10th International Conference on Knowledge and Systems Engineering (KSE), Ho Chi Minh City, Vietnam, 2018, pp. 382-388, doi: 10.1109/KSE.2018.8573374.

32. Peters, W., Dehghantanha, A., Parizi, R.M., Srivastava, G. (2020). A Comparison of State-of-the-Art Machine Learning Models for OpCode-Based IoT Malware Detection. In: Choo, KK., Dehghantanha, A. (eds) Handbook of Big Data Privacy. Springer, Cham. https://doi.org/10.1007/978-3-030-38557-6_6

33. M. Fahim and A. Sillitti, "Anomaly Detection, Analysis and Prediction Techniques in IoT Environment: A Systematic Literature Review," in IEEE Access, vol. 7, pp. 81664-81681, 2019, doi:

10.1109/ACCESS.2019.2921912.

34. R. Vishwakarma and A. K. Jain, "A Honeypot with Machine Learning based Detection Framework for defending IoT based Botnet DDoS Attacks," 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 2019, pp. 1019-1024, doi: 10.1109/ICOEI.2019.8862720.

35. R. Kumar and G. Geethakumari, "Temporal Dynamics and Spatial Content in IoT Malware detection," TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON), Kochi, India, 2019, pp. 1590-1595, doi: 10.1109/TENCON.2019.8929496.

36. Larsen, E., MacVittie, K., & Lilly, J. (2021). A Survey of Machine Learning Algorithms for Detecting Malware in IoT Firmware. ArXiv, abs/2111.02388.

37. Zhongru Ren, Haomin Wu, Qian Ning, Iftikhar Hussain, Bingcai Chen,End-to-end malware detection for android IoT devices using deep learning,Ad Hoc Networks,Volume 101,2020, 102098,ISSN 1570-8705, https://doi.org/10.1016/j.adhoc.2020.102098.

38. A. Abusnaina et al., "DL-FHMC: Deep Learning-Based Fine-Grained Hierarchical Learning Approach for Robust Malware Classification," in IEEE Transactions on Dependable and Secure Computing, vol. 19, no. 5, pp. 3432-3447, 1 Sept.-Oct. 2022, doi: 10.1109/TDSC.2021.3097296.

39. L. Xiao, X. Wan, X. Lu, Y. Zhang and D. Wu, "IoT Security Techniques Based on Machine Learning: How Do IoT Devices Use AI to Enhance Security?," in *IEEE Signal Processing Magazine*, vol. 35, no. 5, pp. 41-49, Sept. 2018, doi: 10.1109/MSP.2018.2825478.

40. R. Raman, "Detection of Malware Attacks in an IoT based Networks," *2022 Sixth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Dharan, Nepal, 2022, pp. 430-433, doi: 10.1109/I-SMAC55078.2022.9987253.