Utilizing Computer Vision Algorithms to Detect and Classify Cyberattacks in IoT Environments in Real-Time

Mikhail Gromov, David Arnold, and Jafar Saniie

Embedded Computing and Signal Processing (ECASP) Research Laboratory (http://ecasp.ece.iit.edu/)
Department of Electrical and Computer Engineering
Illinois Institute of Technology, Chicago IL, U.S.A.

***Abstract*– Computer vision has proven itself capable of accurately detecting and classifying objects within images. This also works in cases where images are used as a way of representing data, without being actual photographs. In cybersecurity, computer vision is rarely used, however it has been used to detect botnets successfully. We applied computer vision to determine how well it would be able to detect and classify a large number of attacks and determined that it would be able to run at a decent rate on a Jetson Nano. This was accomplished by training a convolutional neural network using data publicly available in the IoT-23 database, which contains packet captures of IoT devices with and without different malware infections. The neural network was evaluated on an RTX 3050 and a Jetson Nano to see if it could be used in IoT.**

I. Introduction

IoT devices and networks are often designed in ways which attempt to minimize their costs; however, such a design often contains many security flaws. These flaws can be exploited by using a variety of cybersecurity attacks. Such attacks can often result in a third person getting full or partial access to these devices. This can allow the attacker to read and modify data that the devices should be analyzing, or to utilize the resources of these devices to launch further attacks. Ultimately, this can result in entire IoT systems being rendered useless, or high costs due to these systems being used to perform other attacks, which can utilize the network and processing capabilities of these systems.

While it would be best to prevent these attacks by designing IoT devices and networks to be more secure from the start, that would significantly raise the design cost and complexity. An alternative approach would involve making it so that a system is able to detect these cyberattacks as they occur, and respond to them appropriately, such as disabling traffic going to or from a command-and-control server or disabling devices that are known to be infected with a botnet.

Computer vision and deep learning are machine learning algorithms often used for detecting objects in images, however they can also be repurposed for cybersecurity applications. For example, they can be used to detect malware present in files [6, 7]. Another use case is detecting Mirai botnet infections on an IoT network in real-time [10]. Since computer vision has proven to be successful at detecting these threats, we will be using it to detect and classify when a variety of different cyberattacks are being used on a IoT device, however the same approach should work for an IoT network as well.

For our implementation, we will utilize the IoT-23 dataset, which contains pcap (packet capture) files, and extract the same features that were used by the N-BaIoT dataset in the Mirai botnet detection [10] and Kitsune [9] projects. Then, after feature extraction is complete, a similar image generation process as the Mirai botnet detection project will be utilized to generate images from the dataset, which can be used to train a neural network to detect the cyberattack that was performed during these packet captures. This neural network will be able to generate an output corresponding to either a benign dataset, or the specific cyberattack that is being performed. In case a cyberattack is unknown, there is a chance that it may appear as benign data, however due to the behavior of neural networks, it will most likely be detected as the most similar attack that is present in the dataset and labels. After training, this neural network will be run on a Jetson Nano, such that it is able to capture packets going to multiple IoT devices, as shown in Figure 1.

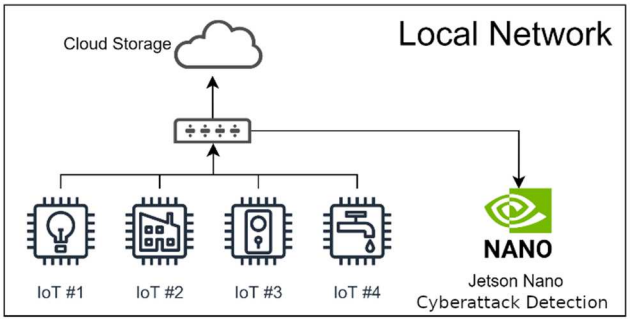

Fig. 1. Proposed IoT Cyberattack Detection System. Multiple IoT Devices are Monitored by One Jetson Nano, which can detect cyberattacks in Real-Time.

The remainder of this paper will first discuss the similar works that were done in the field. Then, we will go over the IoT-23 dataset. Afterwards, we will discuss our data processing and implementation in detail, and a final simulation will also be performed to measure the Jetson Nano's utilization and speed performance. Finally, we will discuss the results of this study, as well as the future impacts it may have on the IoT field.

II. Terminology

This paper looks at how to protect IoT systems from cyberattack. IoT generally refers to devices that are designed to smartly interact with a user and other devices, while utilizing little processing power. These devices often use the internet to connect with other devices or a database server. Due to this, IoT devices often tend to use the bare minimum-security consideration and are quite the attractive target for cyberattacks. A cyberattack is considered an attempt by an

unauthorized user to gain control of a computer system for their own benefit, against the user's interests and without the user's knowledge. When data is sent over the internet in IoT or any other application, it is grouped in packets, which can be captured by a packet capture program. These raw packets are recorded in a pcap file in plaintext, with the packet metadata being plainly readable by anyone. This packet metadata includes a timestamp, the length of the packet, source and destination IPs, mac addresses, and ports, as well as the protocol being used for communication, and the entire data within the packet. This allows a network administrator or program to monitor incoming and outgoing data, and to investigate if something suspicious is detected.

III. Related Works

The first work that was similar to our research is detecting malware embedded in files using computer vision [6, 7]. This work visualized the data present in the files as an image, with each byte of the data being represented by the image. After generating an image from a file, this image is processed using machine vision algorithms, and a binary output is generated which determines whether the file contains malware or doesn't. It is also possible to utilize the same algorithm to get a more specific result on the type of malware that is contained in a file. This would require more processing power than the binary classification, especially with the large list of known malwares.

Another work that was performed in a similar direction was done in the Kitsune paper [9], which utilized a variety of autoencoders to detect a botnet. While this would theoretically be able to detect any cyberattack efficiently, since the autoencoder would be able to detect when a new type of data started to appear in the network, it would require the autoencoder to be trained for each usage application individually. Additionally, it would not be able to distinguish between different cyberattacks. With it often being the case that different cyberattacks have different mitigation procedures, it would not be useful if someone wants to be able to mitigate the cyberattacks after one is detected. Also, in some cases, it may be possible for the benign data to have some similarities to malicious data, which would prevent some cyberattacks from being detected by this approach. While autoencoders would be a more efficient type of neural network compared to computer vision algorithms, the need for individual training would reflect negatively on the cost and complexity of the IoT system design, since each designed system would have to have its own autoencoder trained on every possible benign use case.

In our previous work, we used computer vision to detect the Mirai botnet in real-time using computer vision [10]. While we only focused on being able to detect the Mirai botnet, we were also able to detect another botnet, the Gafgyt botnet, and used a similar approach to the one used in this paper, with only two classification groups. We were able to distinguish between the botnet being present in the network and benign data that had similar features to cyberattacks. This was done with a high overall accuracy and performance. While the dataset used for that work only included data about the Mirai and Gafgyt botnet, the IoT-23 dataset used in this work will include more cyberattacks, so we should be able to train the neural network to be more precise in detecting and classifying a cyberattack, which can open up ways to automatically mitigate the attack while it is in progress before it is able to cause significant damage.

IV. Dataset Description

The IoT-23 dataset contains packets harvested from 23 IoT scenarios. These scenarios include a variety of different attacks, like a Mirai, Torii, and Okiru botnet, as well as Trojans and various other attacks. They also include 3 benign scenarios. In the data present, there are two types of files. The first type is pcap files, which contain the packets captured from the device. The other file type is a labeled Zeek flow, with each connection being cataloged and summarized. For this research, we will be focusing on the pcap files, as those are the files containing data that can be analyzed in real-time. The dataset is unbalanced since it has 20 malicious packet captures and 3 benign packet captures. Additionally, it is quite unbalanced, as the largest two classes, the Mirai and benign data, have 10x and 3x as much data as the other respectively. The dataset includes a large amount of packet data, captured over the course of several hours to 24 hours for each pcap file. To speed up the machine learning process, only the first 100,000 packets of each pcap will be analyzed. Additionally, the initial 2,000 packets will be discarded since the IoT-23 dataset also contains some data prior to the malware infection.

V. Methodology

To process the pcap files into a more usable form, feature extraction was performed. During feature extraction, five lists of objects were generated, with each object containing metadata about a packet when grouped by a certain type of connection. For each packet, an object in every list was selected or created such that it had matching parameters. The first list used the source ip to select the object, with the second and third list both looking at the source and destination IP. A fourth list was created to look at connections from a source ip and mac address, and a fifth list looked at sockets, considered connections between a source and destination using the IP and port. A moving average was used for each object, which would allow us to add new data as it came in efficiently, and output values of the weight, mean, and standard deviation. The core data point used was the length, however the third list kept track of the jitter between packets. Bidirectional statistics were also extracted, including the magnitude, which is the square root of the sum of squares of the mean for both directions in a connection. This also included the radius, which was the square root of the sum of the variances in a similar case. For efficient feature extraction, the number of objects in each list was limited, with older objects being dropped from the list when new objects would need to be created. The extracted features were written in a csv file, which we further processed into images. While many features were extracted, only twelve of them were stored in the images generated. These included all the 1-dimensional statistics for each source, and source mac pair. For the host-host group, the moving average and 2-dimensional parameters were used for the length, with just the moving average of the jitter being considered. The final two data points looked at the socket's moving average and standard deviation.

After the 12 selected features for the first 100,000 packets were extracted and stored in a csv file, they needed to be put into images. To accomplish this, the first few thousand data points were ignored to give the malware time to initialize. Then, after that, the remaining data points were grouped such that they occupied 2x2 pixel areas in a 20x20 rgb image. Four such examples are shown below. The two images on the left are from two different benign packet captures from two different IoT devices. The two images in the center left are from two packets captures of Mirai infected IoT devices. Next, the two images on the center right are from a Gafgyt packet capture. The two images on the right are from a Trojan packet capture. From the images, it can be seen that the features in the packet captures of Trojans and Mirai are completely different. However, in the case of Mirai and Gafgyt, both of which are botnets, there are less identifying features which is due to the similarity in the attacks, but there are still some differences present. The two benign images are completely different, however the images for each device are similar in appearance. A convolutional neural network should be able to select identifying features that would be able to distinguish between malicious and benign data, however it may confuse some of the malicious classes which behave similarly.

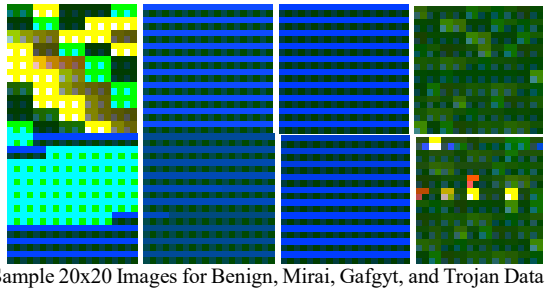

Fig. 2. Sample 20x20 Images for Benign, Mirai, Gafgyt, and Trojan Data (Left to Right)

The images were than randomly assigned to the training, validation and testing sets with a probabilities of 0.5, 0.3, and 0.1. For the Mirai data, a random dropout of 80% of the images was used to make the final dataset more balanced. The neural network used to classify the cyberattack used a 2x2 convolution layer with a stride of 2 and 12 channels, followed by a 2x2 max-pooling layer to transform the image into a 5x5x12 image. Afterwards, a second convolution layer was used with a stride of 1 and 16 channels, followed by another 2x2 max-pooling layer. At this point, the images were 2x2x16, and were flattened into 64. All of the previous layers had a ReLU activation functions. After flattening the data, it goes through three dense layers with sizes 128, 64, and 12. The 12 outputs would each correspond to a cyberattack, excluding the first one that corresponded to no cyberattack.

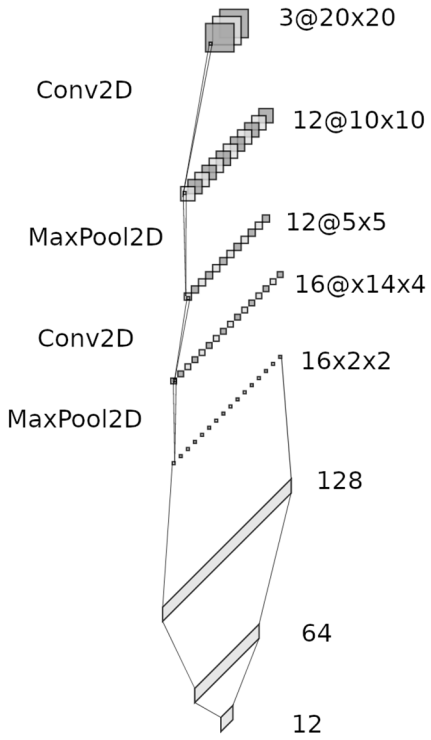

Fig. 3. Neural Network Structure

VI. Results and Impact

When training the neural network, it was found to have a training accuracy of around 70-80%. The final validation accuracy when distinguishing cyberattacks was 76.23%, with an accuracy of 99.84% when a binary classification of whether an attack was ongoing was needed. The testing accuracies was quite similar, being 76.31% and 99.68%. To further discuss the relatively low cyberattack classification accuracy, a confusion matrix was created, and is presented in Figure 4. According to the confusion matrix, it appears that most of the cyberattacks were correctly detected, however some of them were difficult to distinguish between. In this case, it was mostly different botnets that utilized similar algorithms to propagate being confused by the neural network. Looking at the images, we can observe that oftentimes they even look similar to the human eye, for example the Mirai and Gafgyt images presented in figure 2 appear to share many features, so this confusion is justifiable. In a real-world scenario, regardless of the botnet attacks being mixed up, all of them would still be mitigated in a similar method. If the propagation method differs significantly enough, then it might warrant a different approach to disrupting the botnet, but in that case it would be easier for the neural network to distinguish between them.

	Benign	Mirai	Torii	Trojan	Gafgyt	Kenjiro	Okiru	Hakai	IRCBot	Hajime	Muhstik	HideAndSeek
Benign	**0.99**	0	0	0	0	0	0	0	0	0	0	0
Mira	0.01	**0.37**	0	0.02	0.27	0.13	0.09	0	0.07	0.04	0.01	0
Torii	0.02	0	**0.95**	0.01	0	0	0	0	0	0	0.01	0
Trojan	0	0.04	0	**0.92**	0	0	0	0	0	0.04	0	0
Gafgyt	0	0	0	0	**0.99**	0.01	0	0	0	0	0	0
Kenjiro	0	0.01	0	0	0.58	**0.39**	0.02	0	0.01	0	0	0
Okiru	0	0	0	0	0	0.02	**0.8**	0	0	0.17	0	0
Hakai	0	0	0	0	0	0	0	**1**	0	0	0	0
IRCBot	0	0.02	0	0	0.35	0.02	0	0	**0.59**	0	0.02	0.01
Hajime	0	0	0	0	0	0	0.07	0	0	**0.93**	0	0
Muhstik	0	0	0	0	0	0	0	0	0	0	**0.96**	0.04
HideAn	0	0	0	0	0	0	0	0	0	0	**0**	**1**

Fig. 4. Confusion Matrix for Testing Data. The top row represents the predicted attack while the left column presents the observed attack.

In terms of the performance, running the model on an RTX 3050 was able to process a 100-packet image in 720 microseconds, for a total throughput of around 140,000 packets per second. Using a batch size of 64 changes the performance to 3 milliseconds, so processing multiple batches can be used to increase the throughput. On a Jetson Nano, the single image performance decreased to one image in 6 milliseconds, for a throughput of 17,000 packets per second. Using a batch size of 64 on the Jetson Nano gives an evaluation time of 8 milliseconds. These throughputs are generally sufficient for most IoT devices, and should even be able to handle multiple devices or systems using one Jetson Nano. In cases where this is not enough, aggregating data and running a larger batch should allow 50 times more data to be processed in a similar timeframe.

VII. Conclusion

In this paper, a computer vision architecture to detect and classify multiple cyberattacks was created. This system was found to have a high attack detection accuracy, and a moderate attack classification accuracy with reasonable exceptions. It was also found to be able to run on both medium-grade GPUs, as well as a Jetson Nano, so there should be no issues utilizing it in IoT devices and edge computing.

VIII. Future Works

While the convolutional neural network performs very well, there are a few potential improvements that can be made. One such improvement would be changing the feature extraction. In the current implementation, there is simple a limited number of connections, before the last active connection is dropped. It would be more accurate to drop connections as they are closed, which may help optimize feature extraction. Additionally, the size of images and the complexity of the neural network can also be reduced to get responses faster, however that would significantly impact the accuracy of the classification.

References

[1] E. Bertino and N. Islam, "Botnets and Internet of Things Security," in Computer, vol. 50, no. 2, pp. 76-79, 2017

[2] M. Gromov, D. Arnold and J. Saniie, “Tackling Multiple Security Threats in an IoT Environment,” in *2022 IEEE International Conference on Electro Information Technology*, 2022.

[3] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher and Y. Elovici, “N-BaIoT - Network-Based Detectino of IoT Botnet Attacks Using Deep Autoencoder,” *IEEE Pervasive Computing,* vol. 17, no. 3, pp. 12-22, 2018.

[4] C. Kolias, G. Kambourakis, A. Stavrou and J. Voas, "DDoS in the IoT: Mirai and Other Botnets," in Computer, vol. 50, no. 7, pp. 80-84, 2017

[5] J. Margolis, T. T. Oh, S. Jadhav, Y. H. Kim and J. N. Kim, "An In-Depth Analysis of the Mirai Botnet," 2017 International Conference on Software Security and Assurance (ICSSA), 2017

[6] I. Baptista, S. Shiaeles and N. Kolokotronis, “A Novel Malware Detection System Based on Machine Learning and Binary Visualization,” in *2019 IEEE International Conference on Communications Workshops* , 2019.

[7] L. Barlow, G. Bendiab, S. Shiaeles and N. Savage, "A Novel Approach to Detect Phishing Attacks using Binary Visualisation and Machine Learning," 2020 IEEE World Congress on Services (SERVICES), 2020, pp. 177-182

[8] NVIDIA , “Jetson Nano Developer Kit,” [Online]. Available: https://developer.nvidia.com/embedded/jetson-nano-developer-kit.

[9] Mirsky, Y., Doitshman, T., Elovici, Y., & Shabtai, A. (2018). Kitsune: An Ensemble of Auto-encoders for Online Network Intrusion Detection. Proceedings 2018 Network and Distributed System Security Symposium.

[10] M. Gromov, D. Arnold and J. Saniie, “Edge Computing for Real Time Botnet Propagation Detection,” in *2022 IEEE Real Time Communications Conference & Expo*, 2022.

[11] Sebastian Garcia, Agustin Parmisano, & Maria Jose Erquiaga. (2020). IoT-23: A labeled dataset with malicious and benign IoT network traffic (Version 1.0.0) [Data set]. Zenodo. http://doi.org/10.5281/zenodo.4743746”

[12] A. Agniel, D. Arnold, and J. Saniie, "Image Processing for Detecting Botnet Attacks: A Novel Approach for Flexibility and Scalability", in 2022 IEEE International Conference on Real Time Communications Conference and Expo; 2022