Machine Learning for DDoS Attack Classification Using Hive Plots

Pablo Rivas*, Casimer DeCusatis, Matthew Oakley, Alex Antaki, Nicholas Blaskey
Department of Computer Science
Marist College
Poughkeepsie, NY, USA
*Pablo.Rivas@Marist.edu

Steven LaFalce†, Stephen Stone
IBM Systems
IBM Corporation
Poughkeepsie, NY, USA
†slafalce@us.ibm.com

***Abstract*—Cyberattacks have been on the increase as computing power and data storage have become more accessible. The use of recent advances in machine learning across different fields has increased the potential adoption of new algorithms in solving important technological problems. In this paper we describe a novel application of machine learning for the detection and classification of distributed denial of service (DDoS) cybersecurity attacks. Attack pattern training data is obtained from honeypots which we created to impersonate various APIs on a cloud computing network. Attack characteristics including source IP address, country of origin, and time of attack are collected from our honeypots and visualized using a three-axis hive plot. We then implemented and trained a non-probabilistic binary linear attack pattern classifier. A support vector machine and a convolutional neural network were trained using a supervised learning model with labeled data sets. Experimental results suggest that our models can detect DDoS attacks with high accuracy rates.**

***Index Terms*—support vector machines, convolutional neural networks, cybersecurity, DDoS attacks, honeypots, hive plots**

I. Introduction

Our society's overwhelming reliance on an increasingly networked, distributed, asynchronous cyberspace has exposed many security vulnerabilities [1]–[3]. One of the most serious and fastest growing problems is the use of massive botnets for malware delivery and distributed denial of service (DDoS) attacks. Within the past year, the average DDoS attack size has increased over 540%, with the maximum attack size exceeding a terabit/second [4]. This is large enough to destabilize the electric power grid of California [5], or disable internet access for the eastern U.S. seaboard [6]. Similar incidents are expected to drive losses in excess of $2 Trillion by 2020 [7]. Thus, there is a significant need for faster, automated intrusion detection, visualization, and response to botnet attacks. This research problem may be addressed by using machine learning, i.e. training cyberdefense systems to recognize botnet attacks earlier by profiling network traffic behaviors. The potential of machine learning is well established, with the U.S. government issuing an executive order for leadership in artificial intelligence and investing $2 Billion in this field [8] in addition to investments at Google, Facebook, and Microsoft. However, the use of machine learning for intrusion detection and prevention of both external and insider threats lags behind other use cases and has yet to realize its full potential.

To cite a few recent examples, in 2017, botnets disabled major hospitals forcing ambulances to be re-routed and impacting surgical equipment [9]. In late 2016, the Krebs security blog was hit with a DDoS attack twice as large as the prior world record [10], comprised of a massive IoT botnet. Forensic analysis determined that this botnet has capabilities that have never been seen before [10]. In late 2016, the Mirai botnet launched another record setting 1.2 Tbit/second attack, impacting millions of users on Twitter, Amazon, Spotify, Netflix, Tumblr, and Reddit [11]. Such attacks are expected to get worse, since the IoT is expected to surpass 20 billion devices by 2020 [12] and it is currently possible for unskilled attackers to rent botnets with about 500,000 connected devices [9]–[12].

In our research we recognize that the future of cybersecurity is not about machine learning replacing humans, but rather playing to the strengths of both [13]. While machine learning is superior at classifying big data sets based on key features, human security analysts continue to play a valuable role in interpreting cyber-defense data. Visualization is the subject of ongoing cybersecurity research, since it plays a critical role in situational awareness and has the potential to extract more actionable intelligence from large, complex data sets [14]. This research builds on the results of an NSF Award SecureCloud (#1541384) by continuing to pioneer the use of hive plots in cybersecurity. Many of the hive plot properties that make them useful in computational genomics [15] are also well suited to cybersecurity analytics (for example, hive plots do not introduce artifacts into the data sets).

In this paper, we present a novel machine learning application for detection and classification of DDoS attacks. We have implemented, trained, and tested a non-probabilistic binary linear attack pattern classifier. A support vector machine (SVM) using scikit-learn and a convolutional neural network (CNN) using TensorFlow were trained using supervised learning with a labeled dataset. Experimental results show that this approach can identify DDoS attacks early and with high accuracy. Section II explains the honeypots design while Section III addresses the visualization paradigm of hive plots. The

978-1-7281-3885-5/19/$31.00 ©2019 IEEE

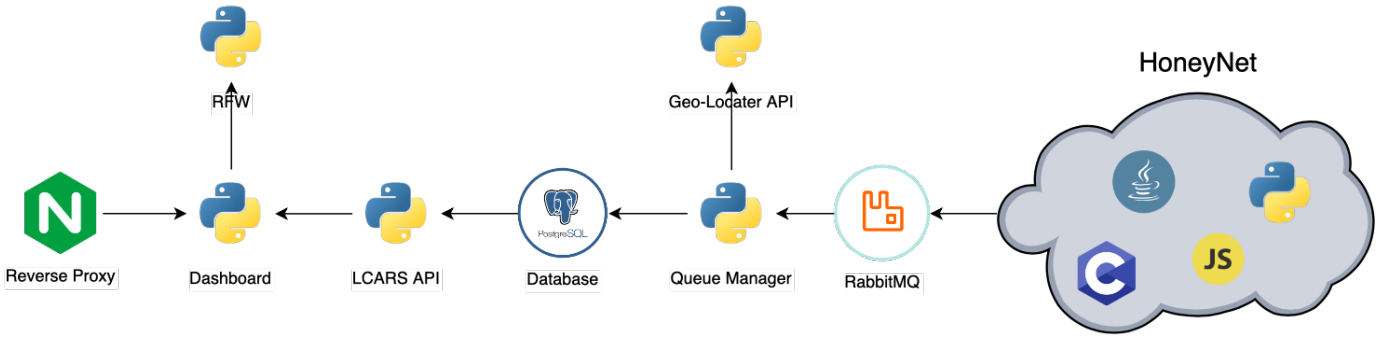


Fig. 1. The LCARS system architecture for storing attack data. The HoneyNet attracts attacks and feeds the data to a database to log all activity for further analysis. The architecture uses Python and Java-based languages as well as the open-source secure and robust database management system, PostgreSQL.

machine learning methodology and the dataset are explained in Section IV. Section V shows and discusses the results of our experiments, while Section VI draws our conclusions.

II. Honeypot Design

The effectiveness of honeypots in cyber-defense is evident from significant recent interest, including The Honeypot Project (a 501(c)3 nonprofit group) [16] and various projects in the 2017-2018 Google Summer of Code [17]. Honeypots are a form of cyber-deception that augments traditional perimeter-based defenses by decoying attackers from production environments. According to a survey of well-known honeypots [18], as attackers fingerprint existing honeypots, there is a continuous need to develop new honeypots capable of deceiving attackers long enough to collect reliable data. Honeypots offer many benefits, including the ability to capture new or unknown behavior such as zero-day attacks or insider threats and to dramatically reduce false positives. This research builds on our prior work, including honeypots for graph database G-Star (supported under NSF CAREER Award IIS-1149372) [19]. With such funding, Marist College has been developing its own set of honeypots to capture valuable attack data to study and analyze to better predict and prevent future attacks [20].

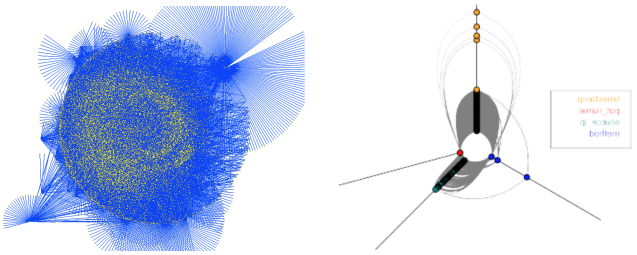
Peitho, named after a Greek god, is a high interaction honeypot designed to act and look like a REST API and attract potential attackers. Peitho then feeds 15 points of information to the LCARS system to provide easy access and analysis of the data. Generally, Peitho attracts DOS attacks and DDOS attacks. Fig. 1 depicts how the LCARS system collects, process, stores, and allows for accessibility of attack data.

III. Attack Pattern Visualization with Hive Plots

Attack pattern visualization plays a critical role in situational awareness and has the potential to extract more actionable intelligence from complex data sets. Hive plots were introduced in biological research [14], [15], [21], [22] as a scalable, computationally straightforward approach to creating informative, quantitative, and easily comparable graphs. Since then, hive plots have found applications in bioinformatics, such as rendering gene co-location networks for bacteria [21]. While they have been applied to a few other areas [22], hive plots have not yet been applied to cybersecurity visualization, including commercial applications such as IBM QRadar [23] and Cisco Tetration [24].

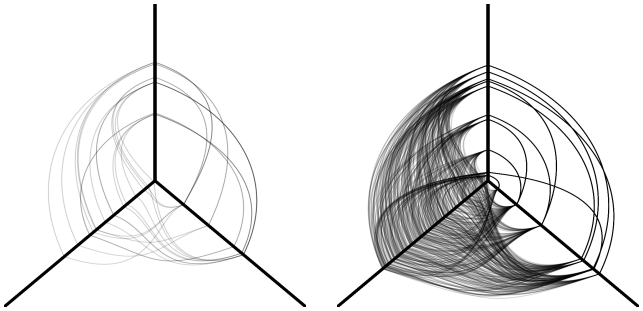
We implemented cybersecurity hive plots during a prior NSF Award #1541384 (SecureCloud), and used hive plots as input to our machine learning algorithm. Many properties which make them useful in computational genomics are well suited to cybersecurity analytics. For example, longtail distributions [25] characterize the statistics of both cybersecurity attack classifiers and many types of cancer mutations [15]. Bioinformatics is concerned with how changes in genome sequence organization and regulation give rise to phenotype differences (i.e. whether a disease state like cancer results from genome manipulation). Similarly, our cybersecurity machine learning algorithms are concerned with questions such as whether a sudden increase in network activity is the result of a botnet DDoS attack or something more benign. In this way, machine learning might help solve the currently intractable problem of botnet attack detection by finding non-obvious data patterns.

Hive plots define a linear layout for nodes, grouping them by type and arranging them along radial axes based on emergent properties of the data. Edges are drawn as curved links (Bezier curves) showing the rational relationship between nodes. In contrast to other types of graphs such as “hairball” force layouts [14], as shown in Fig. 2 (a) do not assign intrinsically meaningful positions to nodes, a hive plot explicitly encodes information in the node position. As shown by Fig. 2 (b), hive plots better reveal underlying structure and communicate interdependencies and aggregate relationships. A hive plot does not obscure the relationship between raw data and the graph, making it well suited to graphic attack pattern recognition. Since hive plots are based on emergent network properties, we can more readily identify and compare attack patterns, analyze the homogeneity and diversity of features in various attacks, and extrapolate time evolution to predict developing attacks. Hive plots excel at managing visual complexity arising from large numbers of edges, and exposing trends and outliers in the data set.



(a) (b)

Fig. 2. *Left:* Example of conventional "hairball" force graph of cyberattack data. *Right:* Example of the same data visualized as a three-axis hive plot mapping. Note that these graphs are shown for topological comparison only.



(a) (b)

Fig. 3. *Left:* Hive plot that exhibits a pattern of normal traffic. Notice that there are several source countries (top axis) that show incoming traffic to mostly different IPs (right axis) during mostly different time periods (left axis). *Right:* Hive plot that shows a DDoS attack pattern. The top axis shows traffic coming from many different countries using a few source IPs (right axis) over distributed periods of times (left axis). However, notice that the darker color elucidates the amount of traffic generated.

IV. Machine Learning Approach

A. Hive Plots Dataset

The hive plot dataset was originally constructed through simulating attacks on the honeypot Peitho. We made a series of HTTP requests to the honeypot Peitho in a given time period. Randomness was introduced in the following two places to better model real world attacks. First, randomness was used in selecting how many proxies would be involved in simulating the attacks. And second, randomness was also used in choosing how many request would be made from each proxy to the honeypot. Then, after the attack was done simulating, a request would be made to the LCARS API to get the attack data from the database and this would be displayed using a hive plot. A similar method but with different tuned parameters was used to generate the normal traffic section of the dataset.

The attacks were simulated using Python. This allows us to automatically simulate an attack and record the honeypot-like responses and also to directly simulate the responses at random for a faster simulation.

The hive plots we used in this research have three axes. The first axis (left) describes the time elapsed since the start of the attack, closeness from the center indicates closeness to the beginning of time snapshot in which the data has been collected. The second axis (right) is the source IP address. The third axis (top) is the country corresponding to source IP address. The lines have a 50% transparency so that many overlapping lines are displayed with a stronger and solid black color. Fig. 3 (a) shows an example of normal traffic, while Fig. 3 (b) depicts a DDoS attack.

One interesting aspect to the data we thought to explore was to see the time progression of the simulated attacks. To simulate this we made the hive plots save at regular time intervals in order to test how well the classifiers could do at measuring real world attacks. This question of how fast and how well our classifiers can determine the probability of an attack going on is very useful. It could potentially allow stopping of cybersecurity attacks both before a human could possibly recognize them and before they cause any harm.

This sequential dataset is shown in Fig. 4. The sequence in the dataset is divided into eight different time steps, t_n, for $n = 0, 1, \ldots, 7$. The sequences intentionally start with blank plots at t_0, to be used as a control group to establish if the machine learning algorithms are biased toward a specific type of class. Ideally, the models should have accuracy of 50% (random chance) at t_0 if they are not biased. In Fig. 4, the top row exemplifies a DDoS attack while the bottom row corresponds to normal traffic. All samples were acquired using the same procedure as explained in Section IV.A.

B. SVM as a Binary Classifier

The initial version of our SVM training algorithm is a non-probabilistic binary linear classifier based on the `sklearn.svm` model [26], intended to perform early identification of DDoS attacks with a confidence score.

Formally, if we define a hive plot image as a square matrix of size $m \times m = d$; then it is possible to flatten such image as a d-dimensional vector $\mathbf{x} \in \mathbb{R}^d$. For an SVM, we define a positive constant $C > 0$ describing the trade off between the training error and define a penalizing term on the parameters of an SVM as $||\mathbf{w}||_2^2$ promoting sparser solutions on $\mathbf{w}$. If we further let variable ξ_i be a non-negative slack variable to account for possible unfeasible solutions, then we can define an SVM as a predictor over input data, $\mathbf{x}$, with the objective

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \frac{1}{2}||\mathbf{w}||_2^2 + C\sum_{i=1}^{N}\xi_i$$
$$\text{s.t. } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i$$

with $\boldsymbol{\xi} \geq \mathbf{0}$, for all $i = 1, 2, \ldots, N$, and where $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$ defines our data set with N images of hive plots having y as the target class. Specifically in our case, the target class is binary: $y \in \{-1, +1\}$. The positive class is a DDoS attack, and the negative class is normal traffic. With this we trained an SVM to model our classification problem using the large hive plot training data sets for various attack patterns.

C. Convolutional Neural Network

Another approach we explored is a convolutional neural network (CNN). A CNN was originally conceived to solve image-based classification problems and has recently gained

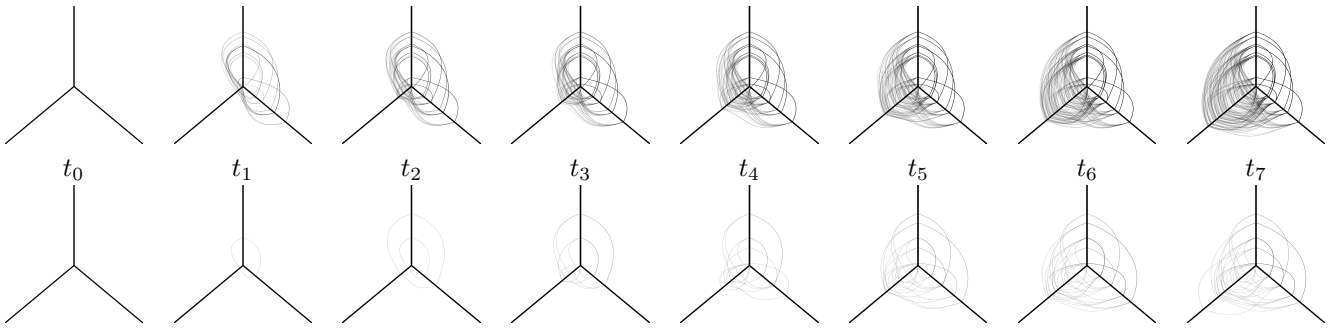


Fig. 4. Time sequence of traffic patterns as they develop in eight different time steps, t_n, for $n = 0, 1, \ldots, 7$. *Top row:* DDoS attack. *Bottom row:* normal traffic. Note that both sets start with a blank plot at t_0, which is used as a control group to determine if the models are biased toward a specific class.

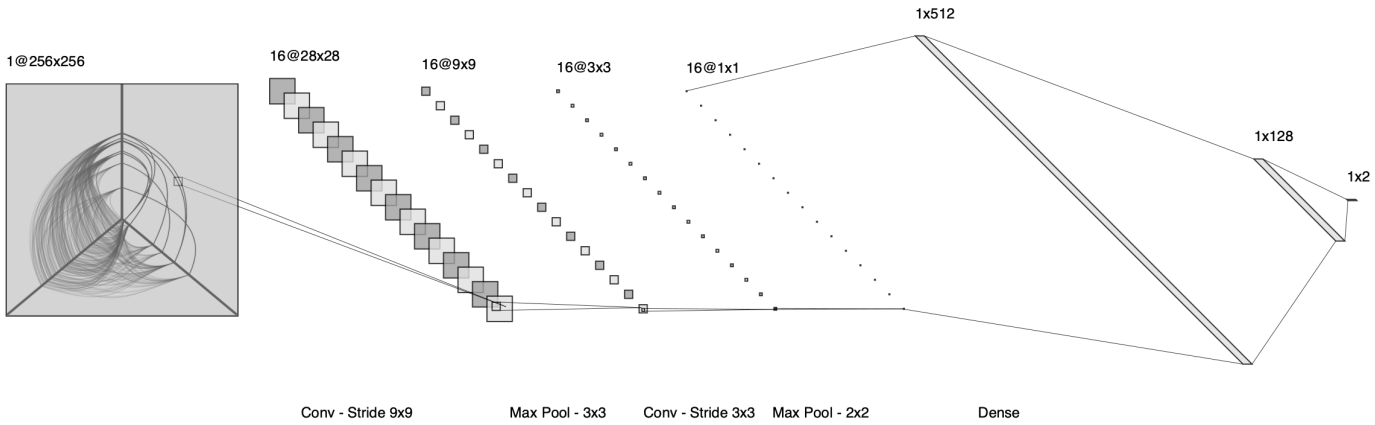


Fig. 5. Proposed convolutional neural network architecture. The model reduces dimensionality quickly to ensure fast response times upon deployment.

a lot of attention due to the potential applications to other problems in machine learning [27], even achieving better performance than a human being in image classification tasks [28].

In general, a CNN determines filters (or matrices) that will be convolved with an input image. There are usually many filters to be determined using a gradient descent technique. The result of the convolution is usually followed by a pooling operation aimed to reduce the dimensionality of the results by making a summary of each by taking the maximum, for example, of a region in the results. This is followed, usually, by an activation function such as a rectified linear unit (ReLU) [29]. This process is repeated for different numbers of filters at different scales to build a network.

Our proposed network is shown in Fig. 5. The CNN begins with 16 convolutional filters of size 9×9 and stride of 9×9. This is followed by max pooling of size 3×3 and ReLUs. The next convolutional layer has 16 filters of size 3×3 and stride 3×3 followed by max pooling of size 2×2 followed by ReLUs. This is then connected to a three-layered dense neural network with 512, 128, and 2 neural units in each layer.

The last layer with two neurons is designed to observe the neuron with the maximum activation out of the two and assign the correct classification to the neuron with the largest stimulus. The activation function in the last layer is, therefore, a sigmoid. The other two layers have ReLU activations followed by a 20% dropout rate on each. A dropout strategy is well-known to prevent overfitting in neural networks [30].

The difference between the input to the CNN and the SVM is that in the SVM the image is flattened to a d-dimensional vector, while in the CNN, the input is left as an $m \times m$ image, thus, the input vector is a grayscale matrix $\mathbf{x} \in \mathbb{R}^{m \times m}$.

V. Experimental Results

We conducted two experiments. The first deals with testing the accuracy of both machine learning methods in distinguishing from DDoS attacks and normal traffic. The second one aims to determine at which time step can the models begin distinguishing between from both classes.

A. Fixed Snapshot Classification

To report performance we performed cross validation [31] on six folds to record accurate generalization capabilities. The performance of both SVM and CNN is shown in Fig. 6 and Fig. 7, respectively. We measured performance using the metric known as the area under the receiving operating characteristic (ROC) curve (AUC). The ROC measures the true positive rate (TPR) and the false positive rate (FPR), which measure the models ability to discriminate among binary classes and exhibit any biases if they are present. Therefore, the AUC is traditionally a better metric when accuracies are

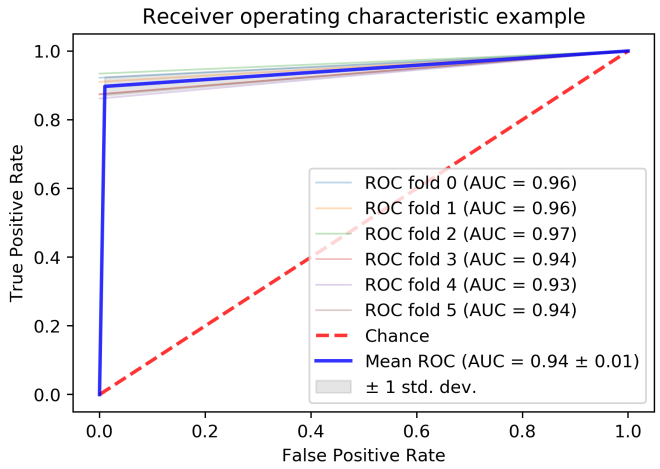


Fig. 6. Using six-fold cross validation, an SVM produced AUCs of around 0.94 in the average case.

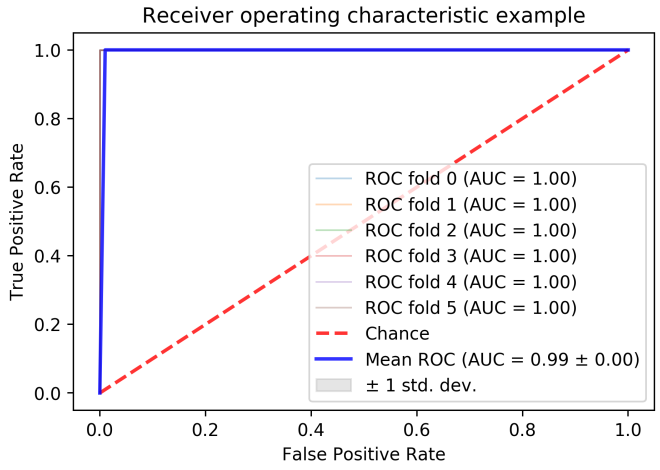


Fig. 7. Using six-fold cross validation, an CNN produced AUCs of around 0.99 in the average case.

very high. The cross-validated accuracy rate on the test set using an SVM was 0.95 ± 0.01, while the AUC was 0.94 ± 0.01.

In comparison, the CNN achieves an accuracy rate of 1.0 ± 0.0 and an AUC of 0.99 ± 0.0, as shown in Fig. 7. Since the performance of the CNN was superior to the SVM, the following remarks are in order. The SVM is known to have $\mathcal{O}(d^2)$ complexity and super-linear behavior at best on N, raising efforts on making the training of SVMs faster [32]. The fastest solution to our problem, given its dimensions (256×256) and number of samples (2000) is to use a linear SVM [33]. This limits the SVM by not taking advantage of the kernel-trick [34], which gives a clear advantage to SVMs to perform better in a kernel-induced high-dimensional hyper-space. This limitation may influence the lower performance of an SVM under the CNN.

As a summary of performance, we present the cross-validated metrics shown in Table I. In terms of accuracy and recall, i.e., the rate of correct classification on relevant samples, the SVM under-performs the CNN.

TABLE I
CROSS-VALIDATED PERFORMANCE METRICS

ML Model	*Accuracy*	*Precision*	*Recall*
SVM	0.947 ± 0.015	1.0 ± 0.0	0.896 ± 0.030
CNN	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0

TABLE II
CROSS-VALIDATED PERFORMANCE METRICS FOR A CONVOLUTIONAL NEURAL NETWORK TRAINED OVER A SEQUENTIAL DATASET

t	Accuracy	Precision	Recall	AUC
t_0	0.506 ± 0.01	0.086 ± 0.19	0.167 ± 0.37	0.500 ± 0.00
t_1	0.952 ± 0.02	0.999 ± 0.00	0.905 ± 0.04	0.952 ± 0.02
t_2	0.990 ± 0.01	0.997 ± 0.00	0.982 ± 0.02	0.990 ± 0.01
t_3	0.993 ± 0.00	0.998 ± 0.00	0.987 ± 0.01	0.993 ± 0.00
t_4	0.995 ± 0.00	0.997 ± 0.00	0.992 ± 0.01	0.995 ± 0.00
t_5	0.996 ± 0.00	0.999 ± 0.00	0.994 ± 0.01	0.996 ± 0.00
t_6	0.998 ± 0.00	0.999 ± 0.00	0.998 ± 0.00	0.998 ± 0.00
t_7	0.998 ± 0.00	0.995 ± 0.01	1.000 ± 0.00	0.997 ± 0.00

B. Classification of Sequential Data

The next experiment was to study the classification ability on a larger dataset containing eight times more images than in the previous experiment, for a total of $16,000$ images. We can think of the dataset in the previous experiment as only evaluating the time snapshot consisting of the final image at time t_7, and this new dataset consists of the time snapshots leading to it, that is, $t_0, t_1, \ldots, t_7$. These sequences (samples shown in Fig. 4) are meant to show the capability of early detection of attacks.

Beginning with the CNN, we found that the CNN is able to recognize a DDoS attack as early as t_1. Table II shows a summary of the cross validated performance metrics.

The model is able to recognize attacks as early as t_1 with an accuracy beyond random chance. And at t_2 and beyond, the model achieves accuracies of 99% with low variance. If we recall that t_0 is a blank plot (Fig. 4) meant to be used as a control group, it has to be true that the accuracy at t_0 has to be random chance, i.e., 50% accuracy.

The SVM on the other hand produced the results shown in Table III. The SVM begins reducing variance after t_2 and is very stable at recognizing attacks at a 96% accuracy rate and greater at t_5. The performance drop can be due to the increase in the number of samples that can be learned and can be ambiguous, e.g., t_0 and t_1 can be easily causing bias toward specific classes. This can be evident by the recall score, which is relatively lower than desirable.

C. What are the ML models learning?

Clearly, using a linear SVM has the disadvantage of not using kernel-induced high-dimensional spaces for classification; however, on the other hand, the solution that it learns from the data is completely transparent and has a direct meaning that is not possible to obtain if we do not use a linear SVM. This means that a linear SVM can be used to determine feature

TABLE III
CROSS-VALIDATED PERFORMANCE METRICS FOR A LINEAR SUPPORT VECTOR MACHINE TRAINED OVER A SEQUENTIAL DATASET

t	**Accuracy**	**Precision**	**Recall**	**AUC**
t_0	0.500 ± 0.03	0.000 ± 0.00	0.000 ± 0.00	0.500 ± 0.00
t_1	0.683 ± 0.02	1.000 ± 0.00	0.366 ± 0.04	0.683 ± 0.02
t_2	0.860 ± 0.02	1.000 ± 0.00	0.720 ± 0.04	0.860 ± 0.02
t_3	0.916 ± 0.01	1.000 ± 0.00	0.832 ± 0.02	0.916 ± 0.01
t_4	0.943 ± 0.01	1.000 ± 0.00	0.887 ± 0.02	0.943 ± 0.01
t_5	0.958 ± 0.01	1.000 ± 0.00	0.917 ± 0.01	0.958 ± 0.01
t_6	0.971 ± 0.01	1.000 ± 0.00	0.941 ± 0.01	0.971 ± 0.01
t_7	0.976 ± 0.01	1.000 ± 0.00	0.953 ± 0.02	0.977 ± 0.01

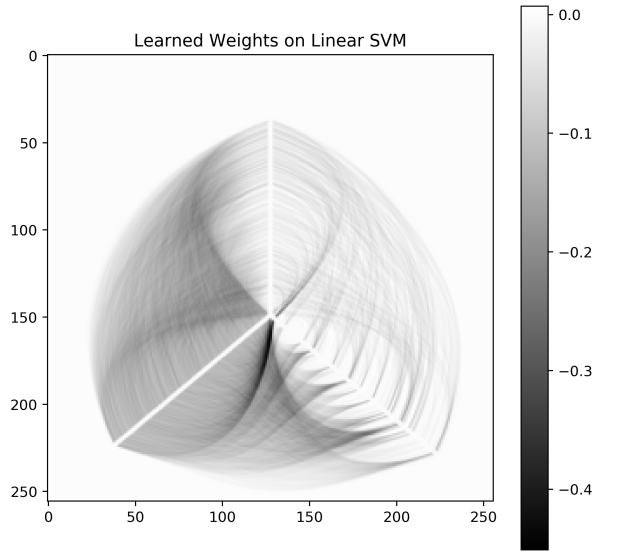


Fig. 8. Learned weights of the Linear SVM represented in two dimensions.

importance. And in our case, we can use the learned solution to display what the SVM believes are the relevant regions of the input images. Fig. 8 shows these learned weights.

Something that is intuitive is that the axis (the actual lines) of the hive plots are irrelevant to determining if there is an attack. This is confirmed by observing that the learned weights in the SVM are completely ignoring the axis and observing at particular curves in the input images.

The CNN on the other hand learns two-dimensional filters in the early layers of the network that are difficult to interpret. These learned filters are shown in Fig. 9.

The filters that a CNN learns through stochastic gradient descent are filters that usually exploit characteristic shapes present or absent in the training images. For example, filters 1, 4, 6, 8, 12, and 13 seem to be learning diagonal patterns and curves so that the network pays attention to these patterns. The other filters are rather abstract and might not have a clear interpretation. Further research will explore the attention mechanism that the CNN model learns.

VI. CONCLUSIONS

This paper presents a novel application of machine learning for the detection of DDoS cybersecurity attacks. Attack pattern training data is obtained from honeypots using characteristics including source IP address, country of origin, time of attack. Such data was visualized using a three-axis hive plot. Then we implemented and trained an SVM classifier using scikit-learn and a CNN using Tensorflow. Experimental results show that both an SVM and a CNN can accurately predict attacks with 95% and 100% accuracy, respectively. Furthermore, the models can detect DDoS attacks as early as in t_1 with accuracy of 95% with a CNN and in t_5 with an SVM.

In 2017, Marist began to host the IBM Z Community Cloud[1] for enterprise development [35], which continues to grow its user base. As part of this service, Marist has access to the IBM Open Data Analytics for z/OS (IzODA) platform[2], which we want to use to create a classifier for cybersecurity attack patterns. An advantage of this platform is that it has been natively developed for high capacity enterprise servers, capable of easily handling the increased volumes of data we expect to collect.

In future research, we will also explore implementing an SVM classifier based on open-source libraries available in the IzODA environment installed on an IBM Enterprise Server running z/OS at Marist [36]. Since the mainframe has high data storage and processing capacity [37] its possible to quickly explore different attack parameter sets and compare the results. When our solutions transition into a production environment, there will be additional benefits of this platform, including pervasive encryption which protects all the machine learning data and results [37]. We will also investigate the relationship between image resolution of the attack graphs and the accuracy of the ML models.

The code to reproduce our experiments can be freely accessed under an MIT license on this Google Colaboratory:

`http://marist.ai/hive-plots-code`

ACKNOWLEDGMENT

We acknowledge the support of Marist College's School of Computer Science & Mathematics, the New York State Cloud Computing & Analytics Center, and IBM Poughkeepsie, NY.

REFERENCES

[1] J. Meier, A. Mackman, M. Dunner, S. Vasireddy, R. Escamilla, and A. Murukan, *Improving web application security: threats and countermeasures*. Microsoft Corporation Washington, DC, 2003.

[2] A. Shostack, *Threat modeling: Designing for security*. John Wiley & Sons, 2014.

[3] W. House, "Trustworthy cyberspace: Strategic plan for the federal cyber security research and development program," *Report of the National Science and Technology Council, Executive Office of the President*, 2011.

[4] M. Kumar, "1.7 tbps ddos attack–memcached udp reflections set new record," accessed on 2018-04-02. [Online]. Available: https://thehackernews.com/2018/03/ddos-attack-memcached.html

[5] S. Soltan, P. Mittal, and H. V. Poor, "Blackiot: Iot botnet of high wattage devices can disrupt the power grid," in *27th {USENIX} Security Symposium ({USENIX} Security 18)*, 2018, pp. 15–32.

[6] J. Margolis, T. T. Oh, S. Jadhav, Y. H. Kim, and J. N. Kim, "An in-depth analysis of the mirai botnet," in *2017 International Conference on Software Security and Assurance (ICSSA)*. IEEE, 2017, pp. 6–12.

[1]Register for free at: `https://zcloud.marist.edu/#/register`

[2]`https://izoda.github.io/`

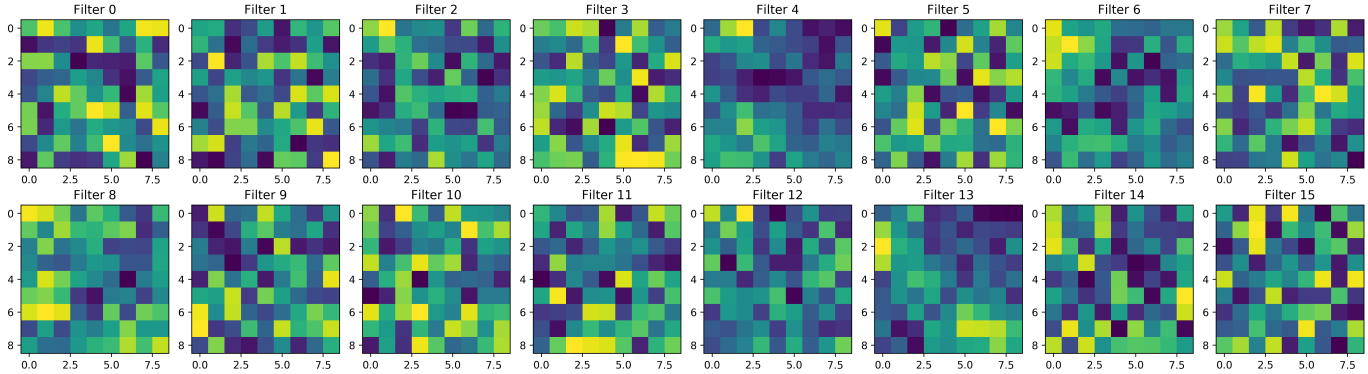


Fig. 9. Learned filters of the CNN.

[7] J. Moar, "The future of cybercrime & security: Financial and corporate threats & mitigation," *Juniper, Dec*, 2015.

[8] D. M. Kocak, "State of technology report: Maritime technology in 2018," *Marine Technology Society Journal*, vol. 52, no. 5, pp. 6–16, 2018.

[9] R. Arndt, "A year after wannacry, health care organizations face mounting cyberthreats," accessed on 2018-12-18. [Online]. Available: https://www.modernhealthcare.com/article/20180622/TRANSFORMATION02/180629972/a-year-after-wannacry-healthcare-organizations-face-mounting-cyberthreats/

[10] P. Krebs, "Krebsonsecurity hit with record ddos attack," accessed on 2017-09-20. [Online]. Available: https://www.krebsonsecurity.com/2016/09/krebsonsecurity-hit-with-record-ddos

[11] B. Krebs, "Ddos attack on bank hid $900,000 cyberheist," *Krebs on Security*, 2013.

[12] R. H. Weber and E. Studer, "Cybersecurity in the internet of things: Legal aspects," *Computer Law & Security Review*, vol. 32, no. 5, pp. 715–728, 2016.

[13] S. Bergsten and P. Rivas, "Societal benefits and risks of artificial intelligence: A succinct survey," in *21st International Conference on Artificial Intelligence (ICAI 2019)*, 2019.

[14] M. Krzywinski, K. Kasaian, O. Morozova, I. Birol, S. Jones, and M. Marra, "Linear layout for visualization of networks," in *Genome Inform*, 2010.

[15] M. Krzywinski, I. Birol, S. J. Jones, and M. A. Marra, "Hive plotsrational approach to visualizing networks," *Briefings in bioinformatics*, vol. 13, no. 5, pp. 627–644, 2011.

[16] L. Spitzner, "The honeynet project: Trapping the hackers," *IEEE Security & Privacy*, vol. 99, no. 2, pp. 15–23, 2003.

[17] H. Vakaria, "Google summer of code," accessed on 2018-12-18. [Online]. Available: https://blog.nebulis.io/

[18] E. van Ommeren, M. Borrett, and M. Kuivenhoven, *Staying ahead in the cybersecurity game: what matters now*. Sogeti & IBM, 2014.

[19] V. Joseph, P. Liengtiraphan, G. Leaden, and C. DeCusatis, "A software-defined network honeypot with geolocation and analytic data collection," in *Proc. 12th Annual IEEE/ACM Information Technology Professional Conference, Trenton, NJ*, 2017.

[20] D. N. Gisolfi, M. Gutierrez, T. V. Rimaldi, C. DeCusatis, and A. G. Labouseur, "A honeynet environment for analyzing malicious actors."

[21] M. J. Pocock *et al.*, "The visualisation of ecological networks, and their use as a tool for engagement, advocacy and management," in *Advances in Ecological Research*. Elsevier, 2016, vol. 54, pp. 41–85.

[22] S. Engle and S. Whalen, "Visualizing distributed memory computations with hive plots," in *Proceedings of the Ninth International Symposium on Visualization for Cyber Security*. ACM, 2012, pp. 56–63.

[23] D. Miller *et al.*, *Security information and event management (SIEM) implementation*. McGraw-Hill,, 2011.

[24] V. Jeyakumar, O. Madani, A. ParandehGheibi, and N. Yadav, "Data driven data center network security," in *Proceedings of the 2016 ACM on International Workshop on Security And Privacy Analytics*. ACM, 2016, pp. 48–48.

[25] R. Kalyanam and B. Yang, "Try-cybsi: An extensible cybersecurity learning and demonstration platform," in *Proceedings of the 18th Annual Conference on Information Technology Education*. ACM, 2017, pp. 41–46.

[26] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[27] K. Mulligan and P. Rivas, "Dog breed identification with a neural network over learned representations from the xception cnn architecture," in *21st International Conference on Artificial Intelligence (ICAI 2019)*, 2019.

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[29] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[30] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[31] R. Kohavi *et al.*, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Ijcai*, vol. 14, no. 2. Montreal, Canada, 1995, pp. 1137–1145.

[32] P. Rivas-Perea and J. Cota-Ruiz, "An algorithm for training a large scale support vector machine for regression based on linear programming and decomposition methods," *Pattern Recognition Letters*, vol. 34, no. 4, pp. 439–451, 2013.

[33] T. Joachims, "Training linear svms in linear time," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2006, pp. 217–226.

[34] B. Schölkopf, "The kernel trick for distances," in *Advances in neural information processing systems*, 2001, pp. 301–307.

[35] Q. Wu, Z. Gao, E. Wang, H. Min, and Z. Wei, "Research on highly consumable platform for business analytics," in *2016 International Conference on Progress in Informatics and Computing (PIC)*. IEEE, 2016, pp. 649–653.

[36] K. Wei and G. Cai, "Bring intelligence to where critical transactions run–an update from machine learning for z/os."

[37] D. Wolpert, E. Behnen, L. Sigal, Y. Chan, G. E. Téllez, D. Bradley, R. Serton, R. Veerabhadraiah, W. Ansley, A. Bianchi *et al.*, "Ibm z14: Enabling physical design in 14-nm technology for high-performance, high-reliability microprocessors," *IBM Journal of Research and Development*, vol. 62, no. 2/3, pp. 10–1, 2018.