International Journal of Engineering in Computer Science

E-ISSN: 2663-3590
P-ISSN: 2663-3582
www.computersciencejournals.com/ijecs
IJECS 2024; 6(2): 115-119
Received: 08-06-2024
Accepted: 25-07-2024

Sanjeevini
Assistant Professor, Malla Reddy Engineering College for Women (Autonomous Institution), Hyderabad, Telangana, India

D Sweeya
Student, Malla Reddy Engineering College for Women (Autonomous Institution), Hyderabad, Telangana, India

O Ranitha
Student, Malla Reddy Engineering College for Women (Autonomous Institution), Hyderabad, Telangana, India

Sayad Seema
Student, Malla Reddy Engineering College for Women (Autonomous Institution), Hyderabad, Telangana, India

Corresponding Author:
Sanjeevini
Assistant Professor, Malla Reddy Engineering College for Women (Autonomous Institution), Hyderabad, Telangana, India

Intelligent malware detection with deep learning efficacy

Sanjeevini, D Sweeya, O Ranitha and Sayad Seema

DOI: https://doi.org/10.33545/26633582.2024.v6.i2b.133

Abstract
Malicious software, sometimes known as malware, is a major concern for government agencies, corporations, and individuals concerned about online security in the modern day. Existing malware detection systems are tedious and inaccurate when it comes to detecting new viruses due to their reliance on static and dynamic evaluations of malware signatures and behavior patterns. Modern malware uses evasive tactics like metamorphosis and polymorphism to quickly change its characteristics and create a plethora of varieties. Given that most newly discovered viruses are really just updated copies of older malware, machine learning algorithms (MLAs) have become an integral part of modern malware research. This can only be achieved with extensive use of features, feature learning, and feature representation. The use of advanced MLAs, such as deep learning, allows for the complete elimination of the feature engineering stage. Even though there have been some recent studies in this area, the training data still tends to bias the algorithms' performance. We need to remove prejudice and objectively evaluate these methodologies to build new, improved strategies for efficient zero-day malware detection. To fill this void in the literature, this work investigates the use of deep learning architectures and standard MLAs for malware classification, detection, and categorization utilizing public and private datasets. The train and test segments of the experimental analysis use public and private datasets that are not discontinuous from one another but obtained across various periods. We also provide a novel approach to picture processing that is well-suited to ML and deep learning frameworks. A comprehensive experimental investigation of several methods found that deep learning architectures outperformed traditional MLAs. Taken together, our findings point to a hybrid deep learning architecture that is both scalable and successful at visual inspection for real-time malware detection. Hybrid approaches based on visualization and deep learning, whether static or dynamic, offer a new and enhanced way to successfully identify zero-day malware in big data settings.

Keywords: Malware detection, deep learning, machine learning algorithms (MLAs)

Introduction
The fast development of technology in this digital age of Industry 4.0 has impacted both the day-to-day operations of companies and individuals. Thanks to the IoT and its applications, the idea of the information society as we know it today has emerged. Problems with security, however, make it difficult to reap the benefits of this industrial revolution, as cybercriminals target both individual computers and networks in an effort to steal sensitive information for financial benefit or to disrupt system operations. These cybercriminals pose a significant risk to systems by employing harmful software, also known as malware. Malicious software is software designed to damage the operating system (OS). Malware goes by many names depending on its function and actions; some examples include adware, spyware, virus, worm, trojan, rootkit, backdoor, ransomware, and command and control (C&C) bot. Malware detection and prevention is a constantly changing issue in the realm of cyber security. Malware writers become better at avoiding detection as researchers create new methods. These days, we use both static and dynamic analysis of request data to spot cyberattacks. The signature-based static analysis compares the contents of newly-received packets with known attack signatures in order to determine whether or not the packets are malicious. Although dynamic analysis is a time-consuming method, it may detect viruses and attacks by utilizing dynamic program execution. Machine learning algorithms like Support Vector Machine, Random Forest, Decision Tree, Naïve Bayes, Logistic Regression, KNearest Neighbours, and Deep Learning Algorithms like CNN and LSTM (Long Short Term Memory) are being used by the author to address this issue and improve the accuracy of malware attack

detection, both old and new. Results from CNN and LSTM outperform all other algorithms.

Related Work

"Cybercrime Cost Assessment "
R. Anderson *et al*., 2013 [1]
The first comprehensive analysis of cybercrime's monetary impact is detailed in this chapter. In light of doubts that earlier research had exaggerated the severity of the issue, the United Kingdom Ministry of Defence requested that the first workshop paper be written. We detail the known and unknown direct, indirect, and defense costs of each major type of cybercrime to the United Kingdom and the rest of the globe. We make a point of differentiating between long-standing crimes that have become "cyber" due to their online nature (like welfare and tax fraud), transitional crimes whose methods have evolved significantly due to the shift online (like credit card fraud), new crimes that have emerged because of the Internet, and what we could refer to as platform crimes, like providing botnets that enable other crimes instead of directly stealing from victims. The annual direct costs to the average person of long-standing crimes like tax and welfare fraud are in the hundreds of pounds/euros/dollars, of transitional frauds a few pounds/euros/dollars, and of new computer crimes in the tens of pence/cents. New and transitional crimes, on the other hand, incur much greater indirect and defense expenses. They may be about the same as the offenders' earnings for the former, and significantly higher for the latter. One third of 2010's spam came from a botnet that made $2.7 million for its owners, while anti-spam efforts throughout the globe likely cost more than a billion dollars. Cybercriminals, like terrorists or metal thieves, do disproportionate harm to society by their actions, and our efforts to combat cybercrime are woefully inadequate. There are a number of well-known reasons for this, including the fact that cybercrimes are on a global scale and have significant externalities, in contrast to more conventional crimes like burglary and vehicle theft, which occur on a local level and have their own set of equilibrium points that have developed over time. Concerning the more immediate issue of what should be done, our data indicates that we ought to allocate less funds towards cybercrime prevention measures (firewalls, antivirus software, etc.) and more towards cybercrime response, i.e., the mundane task of apprehending and prosecuting cybercriminals.

"Malicious singleton file detection on a large scale"
B. Li, K. Roundy, C. Gates and Y. Vorobeychik, 2017 [2]
Within a year, we found that 94% of the billions of program binary files that showed up on 100 million computers were actually on one machine. With an 80:1 ratio of benign to harmful singleton files, polymorphism is a result of a variety of causes, one of which is malware polymorphism. Because there are so many harmless singletons, it is difficult to accurately detect the small number of dangerous ones. We report on a comprehensive analysis of the distribution, features, and attributes of singleton files, both good and bad. We utilize the findings from this study to construct a classifier that relies solely on static features. It successfully identifies 92% of the remaining malicious singletons with a false positive rate of 1.4%, even though the majority of these files use obfuscation and packing techniques, which we do not try to de-obfuscate. At last, we show that our classifier can withstand significant types of automated evasion assaults.

" Towards deciphering malicious code by use of API call extraction ",
M. Alazab, S. Venkataraman and P. Watters, 2010 [3]
Malware developers increasingly employ code obfuscation techniques, such as packers, to remain undetected by antivirus software. Malware may evade detection methods by employing evasion strategies like metamorphism and polymorphism. Consequently, the extraction of payloads concealed within packaged executables is a monumental challenge for security researchers and the antivirus industry. For virus identification, it is usual practice to analyze API calls and employ static unpacking or manual unpacking using certain software tools. Reverse obfuscation relies on these properties, but extracting them from unpacked executables is a laborious process that calls for expert-level understanding of low-level programming languages like assembly and kernel. In this research, we offer an automated approach to discovering fraudulent API request characteristics through feature extraction and analysis. Finding file birthmarks using API call features and similar methods has been studied, however there is little work on features in malcodes. To fill this void, we look for ways to automatically detect and categorize API function calls according to the concealed harmful intent in any compressed software. In order to identify six primary types of suspicious behavior in API call characteristics, this study follows a four-step technique for building an automated system.

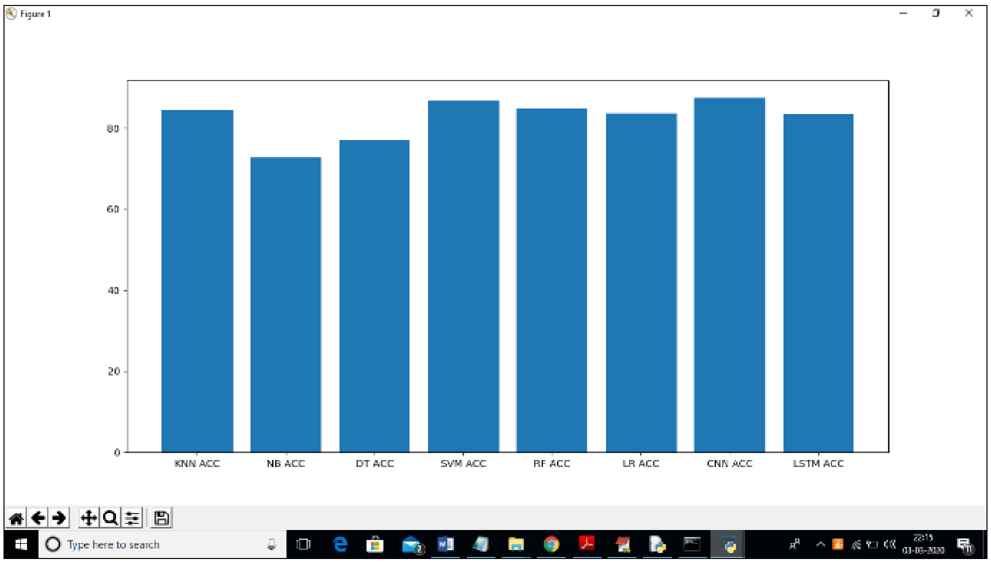
"Exploring the patterns and interconnections of vulnerability disclosure using big data in cybersecurity"
M. Tang, M. Alazab and Y. Luo, 2017 [4]
Existing and new threat actors are increasingly targeting modern organizations' complex Big Data platforms. In order to take advantage of security holes, more complex and targeted attacks will be developed. All contemporary organizations, no matter how big or little, must prioritize efficient vulnerability management in light of the rising tide of cybercrime and events caused by these weaknesses. But the sheer number of security holes found on their networks is too much for most companies to handle. In addition, in reality, vulnerability management is often more reactive. Organizations may get valuable insights and take a more proactive approach to managing cyber risks with the use of rigorous statistical models that simulate the expected volume and dependence of vulnerability disclosures. This new capacity has been made possible by our suggested innovative and rigorous architecture, which takes advantage of the extensive and complicated historical vulnerability data. We begun a significant investigation on handling persistent volatilities in the data and further uncovering multivariate dependent structure among diverse susceptibility hazards by utilizing our sound framework. While most research has focused on univariate time series, here we take a broader look at multivariate cases in an effort to understand the fascinating connections between them. We have demonstrated that a composite model may successfully capture and maintain long-term reliance between various vulnerability and exploit disclosures through our comprehensive empirical experiments utilizing real-world vulnerability data. More research on the stochastic view of

vulnerability propagation is needed to provide more precise metrics for improved cyber risk management generally, and this publication opens the door for that.

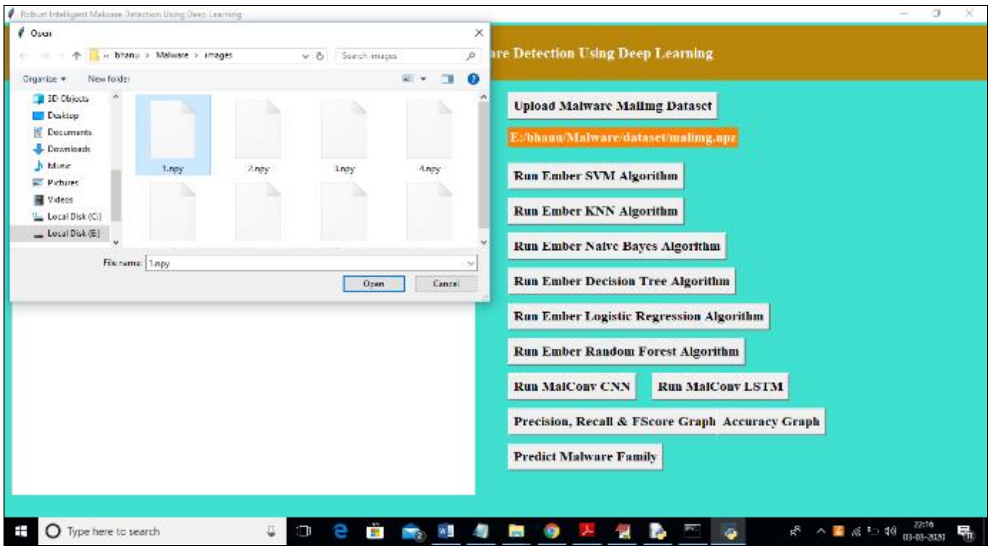
"API call signatures-based supervised learning methods for zero-day malware detection"
M. Alazab, S. Venkatraman, P. Watters and M. Alazab, 2011 [5]
Code obfuscation techniques allow the creation of zero-day or unknown malware by altering the parent code to generate clones with different signatures but identical functionality. There is a significant gap in the literature about the detection of zero-day malware and the methods currently available. In this work, we present and assess a new approach to detecting and categorizing zero-day malware efficiently and accurately using the frequency of Windows API calls. We do this by combining several data mining approaches. The methodology used to train the classifiers is described in this paper. The performance results of the data mining algorithms used in the study are analyzed using a fully automated tool that was developed for this research. The experimental investigations and evaluations are also detailed. Our experimental results allow us to compare and contrast the two data mining techniques and draw conclusions about whether one is better at identifying zero-day malware.
This research makes use of a data mining system that learns by analyzing the behavior of both benign and malicious algorithms in massive datasets. We have tested and assessed the efficacy of several strong classifiers, including the Naïve Bayes (NB) Algorithm, the k− Nearest Neighbor (kNN) Algorithm, the Sequential Minimal Optimization (SMO) Algorithm with four distinct kernels (SMO-Normalized PolyKernel, SMO-PolyKernel, SMO-Puk, and SMO-Radial Basis Function (RBF)), the Backpropagation Neural Networks Algorithm, and the J48 decision tree.

Methodology
1. **Upload Malware MalImg Dataset:** The Attack dataset may be uploaded to the application using this module.
2. **Preprocess & Split Dataset:** Processing techniques including as shuffling, normalization, and test/train split are all within the purview of this module.
3. **Run Ember SVM algorithm:** Here we utilize the SVM approach to determine the model's prediction accuracy, FSCORE, precision, and recall after reading the malware dataset and creating a train and test model.
4. **Run Ember KNN Algorithm:** By utilizing this module, we can receive malware datasets, create test and train models, then use the Ember KNN algorithm to determine the accuracy, precision, recall, and FSCORE of the predictions.
5. **Run Naïve Bayes:** By utilizing this module, we can import malware datasets, create test and train models, and use the Naïve Bayes algorithm to determine the accuracy, precision, and recall of its predictions.
6. **Run Decision Tree:** We read the malware dataset, create a train and test model, and then apply the Decision Tree method to determine the accuracy, precision, recall, and FSCORE of the model's predictions.
7. **Run Logistic Regression:** In order to determine the accuracy, precision, recall, and FSCORE of the model's predictions, we read the malware dataset using this module, create a train and test model, and then apply the LR method.
8. **Run MalConv CNN:** using this module we get its performance details.
9. **Comparison Graph:** Algorithm names are shown on the x-axis of the graph, while other metrics, including accuracy, are shown on the y-axis using various colored bars.
10. **Detect Attack from Test Data:** Choose the "Test Data" file and upload it in this section.

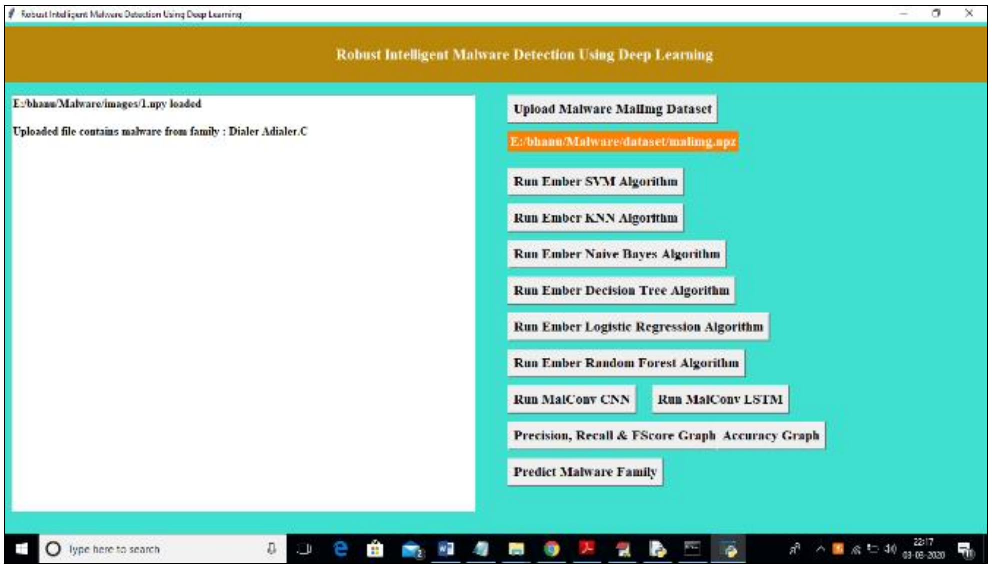
Results and Discussion
Now click on accuracy button to get accuracy graph



Now click on ‘Predict Malware Family’ button and upload binary file to get or predict class of malware



In above graph I am uploading one binary file called 1.npy and below is the malware prediction of that file


Robust Intelligent Malware Detection Using Deep Learning
Upload Malware Mallmg Dataset
E:/bhanu/Malware/dataset/malimg.npz
Run Ember SVM Algorithm
Run Ember KNN Algorithm
Run Ember Naive Bayes Algorithm
Run Ember Decision Tree Algorithm
Run Ember Logistic Regression Algorithm
Run Ember Random Forest Algorithm
Run MalConv CNN
Run MalConv LSTM
Precision, Recall & FScore Graph
Accuracy Graph
Predict Malware Family
Type here to search

```
E:/bhanu/Malware/images/1.npy loaded

Uploaded file contains malware from family : Dialer Adialer.C
```

In above screen we can see uploaded test file contains 'Dialer Adialer. C' malware attack. Similarly u can upload other files and predict class.

Conclusion

A highly scalable system for detecting, classifying, and categorizing zero-day malwares was built and assessed in this research using deep learning architectures based on static analysis, dynamic analysis, and image processing approaches, in comparison to standard machine learning algorithms (MLAs). After collecting malware from end user hosts, this framework uses a two-stage procedure to analyze it using deep learning. To begin, malware was categorized using a method that included static and dynamic analysis. The second step was classifying malwares according to their associated characteristics using image processing techniques. Results from this study's experimental analyses, which used several versions of the models on publicly accessible benchmark datasets as well as privately acquired datasets, showed that deep learning based techniques performed better than standard MLAs. The created framework can analyze a huge amount of malware in real-time, and it can be expanded to analyze an even bigger amount of malware by adding more layers to the current structures. The next step for researchers is to look at these variants with additional characteristics that may be included in the current data.

The major finding of this work, weakness and future scope can be summarized as follows:

- Our suggested architecture for scalable malware detection is a two-stage approach. As a first step, the suggested system employs state-of-the-art deep learning to identify malware; as a second step, it sorts malware into relevant categories.
  Whether it was static or dynamic malware detection or classification based on image processing, deep learning architectures achieved better results than traditional MLAs. One research that used dynamic analysis to detect malware used deep learning architectures trained on characteristics derived from domain knowledge. To circumvent this, it is possible to capture binary file memory dumps during runtime and convert them into grayscale images.
- In the study on malware identification using image processing and deep learning, the malwares were first flattened and then converted to fixed-size pictures. Using the spatial pyramid pooling (SPP) layer in future work enables the use of pictures of any size as input.

We may increase our models' adaptability by inserting this, which learns features at varied scales, between the subsampling and fully linked layers.

- There is a significant imbalance in the malware families found in the Maling dataset. The unbalanced problem of multiclass malware families can be addressed by adopting a cost-sensitive strategy. Incorporating cost components into deep learning architectures' backpropogation learning approach is made easier by this. The primary indicator of categorization significance, which yields lower results, is the cost item.

References

1. Anderson R, Barton C, Böhme R, Clayton R, Van Eeten MJ, Levi M, *et al*. Measuring the cost of cybercrime. In: The economics of information security and privacy. Berlin, Heidelberg: Springer; c2013. p. 265-300.
2. Li B, Roundy K, Gates C, Vorobeychik Y. Large-scale identification of malicious singleton files. In: Proceedings of the Seventh ACM Conference on Data and Application Security and Privacy; c2017 Mar. p. 227-238.
3. Alazab M, Venkataraman S, Watters P. Towards understanding malware behaviour by the extraction of API calls. In: 2010 Second Cybercrime and Trustworthy Computing Workshop; c2010 Jul. p. 52-59. IEEE.
4. Tang M, Alazab M, Luo Y. Big data for cybersecurity: vulnerability disclosure trends and dependencies. IEEE Trans Big Data; c2017.
5. Alazab M, Venkatraman S, Watters P, Alazab M. Zero-day malware detection based on supervised learning algorithms of API call signatures. In: Proceedings of the Ninth Australasian Data Mining Conference. Australian Computer Society, Inc; c2011 Dec. p. 171-182.
6. Alazab M, Venkatraman S, Watters P, Alazab M, Alazab A. Cybercrime: the case of obfuscated malware. In: 7th ICGS3/4th e-Democracy Joint Conferences 2011. Springer; c2011 Jan. p. 1-8.
7. Alazab M. Profiling and classifying the behavior of malicious codes. J Syst Softw. 2015;100:91-102.
8. Huda S, Abawajy J, Alazab M, Abdollalihian M, Islam R, Yearwood J. Hybrids of support vector machine wrapper and filter based framework for malware detection. Future Gener Comput Syst. 2016;55:376-390.
9. Raff E, Sylvester J, Nicholas C. Learning the PE header, malware detection with minimal domain knowledge. In: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security; c2017 Nov. p. 121-132.
10. Rossow C, Dietrich CJ, Grier C, Kreibich C, Paxson V, Pohlmann N, *et al*. Prudent practices for designing malware experiments: status quo and outlook. In: Security and Privacy (SP), 2012 IEEE Symposium on. IEEE; c2012 May. p. 65-79.
11. Raff E, Barker J, Sylvester J, Brandon R, Catanzaro B, Nicholas C. Malware detection by eating a whole exe. arXiv preprint arXiv:1710.09435; c2017.
12. Krcál M, Švec O, Bálek M, Jasek O. Deep convolutional malware classifiers can learn from raw executables and labels only; c2018.
13. Rhode M, Burnap P, Jones K. Early-stage malware prediction using recurrent neural networks. Comput Secur. 2018;77:578-594.
14. Anderson HS, Kharkar A, Filar B, Roth P. Evading machine learning malware detection. Black Hat; c2017.
15. Verma R. Security analytics: adapting data science for security challenges. In: Proceedings of the Fourth ACM International Workshop on Security and Privacy Analytics. ACM; c2018 Mar. p. 40-41.