1st AAYUSH KUMR, 2nd ABHISHEK SAXENA

3rd  ANJILA CHOUDHARY 4th AZHARUDDIN ALAM   
*Computer Science and Engineering  
Greater Noida Institute of Technology*Greater Noida, India

[aayushh201@gmail.com](mailto:aayushh201@gmail.com)

*Abstract*— In an ever-growing digital age, computers hold a significant role in society. However, with this reliance comes a concerning rise in malware and its widespread impact. As this malicious software continues to evolve and increase in number, it poses an ongoing threat that demands constant vigilance and proactive measures. Cybercriminals use malware as a potent weapon to launch deliberate, malevolent assaults against expensive computer systems, which, in the absence of appropriate preventive measures, may have disastrous results. To combat this looming threat, research efforts have turned to machine learning as a favored avenue for classifying malware. The review paper aims to the diverse technology used for detecting malware and its effectiveness

***Keywords—malware, cybersecurity, Machine Learning***

1. INTRODUCTION

The word "malware" was created to describe harmful software intended to carry out an attacker's malevolent aims. Malware has the ability to infiltrate, steal sensitive data, hack computers and smart devices, create security vulnerabilities in networks, damage vital infrastructures, and more. Malware is available and has weakened the security of a great deal of systems and devices due to the internet's broad use [1].

In order to evade detection and destruction, malware is always evolving to use signature-based and cutting-edge machine learning approaches.

Daly published research in 2009 that indicated the possibility of planned, coordinated assaults to get prolonged access to a company's network [2]. Similarly, Researchers at the Quick Heal Threat Research Lab received over 350,000,000 malicious files that targeted tens of thousands of workstations in the first quarter of 2016[3].Aimoto et al. reported in 2017 that Symantec had discovered the Banswift criminal organization, which had pilfered US funds amounting to 81 million from the Bangladesh Bank [4]. Because Elasticsearch and Logstash were improperly set, the Kid Security app which was designed to keep an eye on kids' online safety exposed user records for more than a month. Expert Bob Diachenko found the breach in mid-September, which affected 300 million records 21,000 phone numbers, 31,000 email addresses, and some credit card information according to CyberNews in September 2023.

Researchers and malware analyzers are thus encouraged to examine known malware and detect active anti-malware software in order to uncover both known and unknown harmful program, to evade cyberattacks on computer networks. Since bank consumers were the main target of hackers, failing to take appropriate precautions against

5th ASHWINI KUMAR VERMA

*Assistant Professor*

*Computer Science and Engineering*

*Greater Noida Institute of Technology*

Greater Noida, India

[cyberneticsjustice@gmail.com](mailto:cyberneticsjustice@gmail.com)

malware might have disastrous consequences, including temporarily paralyzing financial networks.

Although Tabish et al. (2009) believes that malware detection is a ubiquitous problem that affects a variety of file types and operating systems [58], the Windows platform will be the main focus of our article due to its broad usage and longevity. For a very long time, researchers and malware developers have been competing with one another. The earliest and most effective solution to the analysis and classification problem seems to be ML; it has been used for decades to solve the malware classification and detection problem. ML has become crucial for modern cybersecurity since it is more successful at identifying new threats than signature-based defenses. The truth is that identifying malicious software using traditional methods such as rule-based, graph-based, entropy-based, etc. is challenging.

As a result, machine learning emerges as a clear and dependable option for creating classifiers that can identify both newly discovered malware and that which is part of already-existing families [5, 6]. Numerous papers by different authors were reviewed, and it was discovered that supervised and unsupervised machine learning methods have been proposed using Random Forest, Navies Bayes, Decision Trees, SVM, KNN, Adaboost, and other techniques, with great results [7, 8]. Following an examination of various machine learning-based detection methods, it was determined that malware characteristics needed to be gathered from both static and dynamic malware analysis, and classifiers needed to be trained on classification techniques. These two factors are significant and have an impact on malware classifier accuracy. However, there are a few drawbacks to machine learning models. To begin with, creating and maintaining these models requires a significant amount of expertise and work in order to locate and arrange helpful characteristics for training, which is required to generate an appropriate classification model. End-to-end deep learning models may help with the above described issues to some extent. There have been suggestions for byte-based models to detect malware in Windows Portable Executable files [10, 11]. These models have shown to be just as successful as traditional machine learning [10, 11]. This article will discuss several ML methods for classifying and detecting malware, as well as the relevant research.

1. MALWARE ANALYSIS

Malware analysis is a technique that examines the composition and actions of malware by finding traits that point to its intended purpose. Static analysis and dynamic analysis are the two methods which are available for obtaining input characteristics. Static analysis retrieves characteristics that indicate its intended use. Static analysis and dynamic analysis are the two methods available for obtaining input characteristics.

Static analysis retrieves characteristics that will feed into the classifier without requiring the program to run, while dynamic analysis derives features while the code runs. According to research, malicious software programs within a certain family tend to have a comparable set of opcodes. Another feature that sets malicious files apart from benign ones is the potential prevalence of a few opcodes [12].

Another method of categorizing malware is into first and second generations. malware of the first generation, also referred to as static malware. Examples include "rootkits," "spyware," "crimeware," and "adware," in addition to "viruses," "worms," and "Trojan horses." This kind of malware modifies the structure of the target computers to induce harmful behavior. It is necessary to collect many static features from the malware code, including "hash value," "N-grams," "opcodes," "strings," and "PE header," in order to investigate these malicious programs. The amount of features that may be taken into consideration for classification is significantly lowered as a consequence of static or first generation analysis. Many antivirus programs and intrusion detection systems are designed to identify malware based on these characteristics [13, 14].

Disassemblers are used to break down executable files into assembly language code, which is then examined for malware. Olly-dbg, I.D.A Pro, capstone, and other debuggers and disassemblers are among of the most widely used programs for converting binary files into assembly code. [15, 16]. Commonly used features for static malware analysis include from calls to API functions, entropy, header values, etc. Disassembled code is examined to find and investigate file execution flows, function calls, and dangerous code behavior. This information may then be used to create or find new software or variations of already-existing software. Assembly code analysis is challenging as it takes a lot of time, in addition to the several ways malware obfuscates itself, for as by using code encryption.

[17, 18]. XORing the generated key with the malware body encrypts it to make it more difficult to detect. Static analysis tools and traditional signature-based detection approaches were not able to identify encrypted malware.

Dynamic or second-generation malware analysis is the act of running malicious files and watching how they behave. This behavior is the result of malicious code interacting with the machine. During dynamic analysis, a large number of common characteristics are retrieved, analyzed further, and utilized to train the classifiers. Features such as system calls, API function calls, modifications, performance counters, etc. are retrieved throughout the study. In 2018, Ding et al. provided an explanation of how malicious code behaves during runtime, citing file system actions, registry key change, process execution, and network activities [19, 20, 21]. Using virtualization tools like VMware, sample malware code is run in a monitored virtual environment and observed activities such as changes made to the registry, the creation or deletion of files, mutually exclusive actions, TCP/IP calls, the removal of system files, log entries, the list of URLs visited, API requests, and so on.. File analysis of these actions indicates if the file is malicious or benign.

More classifications for second-generation malware include polymorphic, metamorphic, and oligomorphic malware.

Computer viruses use oligomorphic code, which is similar to polymorphic code in that it creates a description of itself. They also provide a set of decryptors, numbering in the hundreds, which enable detection, while polymorphic code mutates the instructions using a variety of obscuring techniques to produce millions of decryptors. Malware created using a polymorphic code engine is incapable of self-rewriting. Malware that transforms itself instead of the decryptors is known as metamorphic malware. In order to evade detection, it generates new versions by using a variety of cutting-edge and sophisticated concealing strategies without altering its behavior [24].

In order to address situations when a single method may not be enough, malware detection systems use a hybrid strategy that blends dynamic and static analysis. Traditional signature-based antimalware can identify only a small percentage of malware, and it is insufficient to detect sophisticated or undiscovered malware because of their improved obfuscation tactics. These malwares are capable of creating a vast array of variations in order to elude detection and get past security measures.

So, the consequences might be unthinkable if sufficient and suitable steps are not done to combat these malwares. Advanced metamorphic malware is very difficult to detect and a constant danger to endpoint security because, while the altered code performs the same function, it changes its appearance to evade detection by existing detection techniques. Therefore, in order to effectively identify complex or metamorphic malware, new techniques and ways to monitor the behavioral pattern of different malwares must be developed in order to better understand and diagnose the problem with a low amount of false positives.

# Features extraction

The technique of extracting features from unprocessed data is known as feature extraction. Due to its inherent obfuscation, binary data is very difficult to extract features from.

As a result, it influences the effectiveness of machine learning methods and algorithms, which primarily rely on the quality of the input data. Malware may exist in many different binary formats, including Windows PE files with the.exe,.dll, and.efi extensions.

Thus, increasing one's understanding of program internals is essential for identifying and obtaining valuable characteristics while doing security research on computer binaries. Feature extraction lowers the processing cost when we have a large quantity of data to analyze. The phase of feature extraction yields a vector with the extracted features' frequencies in it. These features affect the classification accuracy of the machine learning model. The three types of analytic approaches—static, dynamic, and hybrid—that form the basis of feature extraction are covered in the preceding section. Examples of features based on dynamic analysis include information flow tracking, function-based features, A.P.I., and system calls is based on static analysis techniques include characteristics of portable executables, entropy, header values, bytecode and opcode n-grams, and string functionality. Programs are watched during dynamic analysis in order to comprehend their dynamic behaviors such as resource use, network connections, system calls, system calls to enable memory access, etc. Many sandboxes and tools are available for feature extraction, such as Process Explorer, Tcp Dump, for static analysis, use PE View; for dynamic analysis, use Cuckoo, among other programs. The characteristics are extracted, converted into feature vectors, and then the classifier model is trained using these feature vectors..

1. Malware Detection techniques

The main objectives of malware detection methods are to detect malicious software and safeguard the system on which it is installed in order to preserve the security of linked networks and computer systems. To assist with the detection and grouping of malware samples into the appropriate families, the inputs may be described in a number of ways.

Over time, a number of authors have proposed methods for identifying and categorizing malware files and the version that they are associated with. In this publication, we have included a comprehensive summary of the major research publications listed in Table 1.

Malware that has been extracted has also been seen in graphic form in addition to feature vectors. The process of recognizing malware and grouping it into families is labor-intensive and needs subject expertise, therefore it differs from classifying images. In one such study, the authors proposed a learning-based technique for analyzing dangerous code and categorizing it according to the malware family that it is a member of. Extracting portable executables (PE) and selecting the traits that are most prevalent among them is the first step in grouping malware into families. These characteristics help identify malware activities and the related category by describing the structure of portable executables. The accuracy of the suggested approach was 99.8%[59].

TABLE 1

Review of the Literature

|  |  |  |  |
| --- | --- | --- | --- |
| **Authors** | **Inputs** | **Algorithm**  **/Techniques** | **Findings** |
| Ye et al.in  (2007)[25] | Api  Execution  sequences | Rule based  classifier | IMDS surpassed other data mining techniques and several antivirus programs with a 93% detection accuracy. |
| Shafiq et al. (2008) [26] | n-grams | HMM | The suggested technique detects malware with a TPR of 84.9% and an FPR of 16.7%. |
| Moskovitch et al. (2008) [27] | opcode features | ANN, NB, DT and Adaboost | The model predicted the virus in the file under examination with a considerable degree of accuracy (94.5%). |
| Griffin et al. (2009) [28] | 48 Byte string signature | 5- gram Markov Chain model | Signatures with one or more components were used to train the classifiers. Having several component signatures increased the chance of a satisfying accuracy outcome when compared to the equivalent. Less than 0.1 percent false positive rate (FPR) was achieved. |
| Nataraj et al. (2011) [29] | Gray scale image of Binaries | KNN | demonstrates 98% classification accuracy on a collection of malwares from 25 different families. |
| Shabtai et al. (2012) [30] | 1-6 gram opcode features | Random Forest classifier, Naive Bayes, ANN, Logistic Regression, BDT, DT, and BNB | When it came to accuracy, RF, BDT, DT, G-Mean, FPR, and TPR performed better than NB and BNB. Random Forest produced the best results with 95.14% accuracy. |
| Ravi et al. (2012) [31] | API call sequence | J4.8, IMDS, SVM, Rule Based classifier, Naive Bayes, and SVM | The suggested solution makes use of a third-order Markov model, which operates with 90% accuracy on the testing dataset and 99.38% accuracy on the training dataset. |
| Santos et al. (2013) [32] | Frequency of opcodes | DT, KNN, Bayesian, SVM | SVM performs better than 95.7 % for features of two opcode lengths. |
| Comar et al. (2013) [33] | Flow level features | KNN, SVM, WL, RBF | For identifying new classes, the supervised weighted linear kernel provides the best performance metric. |
| Uppal et al. (2014) [34] | N grams from API sequences | Naive Bayes, Random Forests, SVM, and Decision Tree Classifiers | SVM produces the best results (98.5% accuracy) out of all the classifiers. |
| Salehi et al.  (2014) [35] | API calls | RF, J48, Rotation RF, FT, and NB | 94.6% was the greatest true positive rate of any classifier used, and random forest produced the highest results. |
| Sexton et al. (2015) [36] | Byte code Sequences & opcodes | Naive Bayes, Rule Based classifier, Logistic Regression, SVM | The Markov chain approach to SVM revealed an 84.9% True Positive Rate. |
| Saxe et al. (2015) [37] | The string histogram, the byte sequence, and the 2D PE properties | Deep Feed forward neural network | The suggested model yielded a 95% True Positive rate. |
| Narra et al. (2016) [38] | Opcode Sequence | K-means, expectation maximization with HMM, SVM | The model operates with a 98% accuracy rate. |
| Ahmadi et al. (2016) [39] | Hex dump based features | XGBoost classification algorithm | A 99.8% detection accuracy was provided by the suggested model. |
| Kolosnjaji et al. (2016) [40] | System call sequences | Convolutional & Recurrent Neural Network | The average accuracy and recall of the combined model were 85.6% and 89.4%, respectively. |
| Narayanan et al. in (2016) [41] | Image of Polymorphi c Malware file | KNN, ANN, and SVM | Over the others, linear KNN provided an accuracy of 96.6%. |
| Nikolopoulos and Polenakis (2017) [42] | ScD graph created using system calls | SaMesimilarity and NP-similarity metrics | The suggested model has a detection rate of 83.42%. |
| Zhixing Xu et al. (2017) [43] | systemcalls for memory access | logistic regression and random forest classifier | The random forest classifier performed better, with a true positive rate of 99%. |
| Raff et al. (2017) [44] | PE header features | LSTM, Random Forests, LR, ET | An accurate network with all connections made and calibrations made may reach 93.3%. |
| Kotov et al. (2018) [45] | Windows API calls | Symbolic execution & HMM models | With an accuracy rate of 87.6%, the top prediction model detects malware. |
| Le et al.(2018) [46] | Gray scale image of binary malware file | Convolutional Neural Network | Using 10568 binary data to train the classifier, the accuracy rate was 98.5%. |
| Nguyen et al. (2018) [47] | Image based representati on of lazy binding CFG | CNN | CNN produced an accuracy of 98.87%. |
| Krcal et al. (2018) [48] | PE, API calls | MalConv, CNN, FNN | At 96.4% accuracy, the suggested convolution network outperforms other models. |
| Ni et al. (2018) [49] | Gray images based on Sim hash | Hashing & CNN | The average accuracy of classification attained was 98.86%. |
| Rathore et al. (2019) [50] | Opcode Features | RF, DNN with 2, & 7 Hidden Layers | RF outperforms DNN with a 99.6% accuracy rate. |
| O. Suciu et al. (2019) [51] | PE header features | FGSM | The suggested method shows that forceful assaults on mode are effective. This does not provide efficient models when trained on small datasets. |
| Yuxin et al., in (2019) [52] | n-gram | Deep Belief Network | When trained on unlabeled data, DBN outperformed KNN, SVM, and Decision Trees in terms of classification accuracy. |
| Rabbani et al . (2020) [53] | protocols, jitters, IP addresses, TCP, and UDP | PSO with PNN | With 96.5% accuracy, the model was able to identify malicious behavior. |
| Yucel et al. (2020) [54] | Memory Image of Exe file | Virtual machines & 3D Imaging | Using an average of 0.886, the authors' research looked at the similarity rates across many malware families. A few succeeded in reaching a 99.5 percent accuracy rate. |
| Vasan et al. (2021)[55] | Windows executables, system call sequences | Unsupervised anomaly detection using Isolation Forest | displayed the potential of unsupervised learning for malware detection by achieving high accuracy in identifying previously unknown malware types. |
| umar et al. (2022)[56] | Android APK files, API calls, permissions | Hybrid model combining static and dynamic analysis using RNN | Detected malware with 98.7% accuracy, highlighting the effectiveness of hybrid approaches for Android malware detection. |
| Gibert et al. (2023)[57] | PE files, opcode sequences | LightGBM, CNN with attention mechanism | Achieved 99.4% accuracy in malware detection, outperforming other ML algorithms. Attention mechanism improved model interpretability. |

There are several challenges when using machine learning to identify malware. First, it has to do with the large computational expense of updating and training malware classifiers. Since the model must recognize the most recent and freshly created malware, regular updates are necessary. Second, the characteristics that are collected from malware might be enormous, which can also have an impact on the model's training or performance. Third, and this is still another major issue, some malware makers may be employing machine learning (ML) to create and sell malware that is evolving. This allows them to avoid detection with ease [44].

1. Conclusions

In order to differentiate malicious software from benign software, we have endeavored to provide an overview of machine learning research in this article, along with an analysis of how well such efforts have fared in comparison to other classifiers or existing methodologies.. We started off by talking about the need for and purpose of malware analysis, then we moved on to talk about the traits that needed to be extracted so that the different classifiers could be trained, and lastly, we discussed the problems and performance that arose during the construction of the detection model. After reading and contrasting the publications, we performed analyses based on important factors such the classifier's output, the strategy, the input characteristics, and the classification algorithm. However, we think there are a lot of unutilized algorithms out there that may provide superior outcomes. The malware detection and classification algorithms have to possess sufficient resilience to tackle newly emerging malware iterations.

##### References

1. Sharma, A., Sahay, S.K., “Evolution and detection of polymorphic and metamorphic malwares: a survey”, Int. J. Comput. Appl. 90(2), pp. 7– 11 2014.
2. Daly, M.K., “Advanced persistent threat”, Usenix, Nov. 2009.
3. Quick heal quarterly threat report q2: Technical report, 2015. Quick Heal, Feb. 2015.
4. Aimoto, S., AlKhatib, T., Coogan, P., Corpin, M., DiMaggio, “Internet security threat report”, Technical report, Symantec Corporation, 2017.
5. B. Ndibanje, K.H. Kim, Y.J. Kang, H.H. Kim, T.Y. Kim, H.J. Lee, “Applied sciences cross-method-based analysis and classification of malicious behavior by API calls extraction”, 2019.
6. J. Zhang, “Machine learning with feature selection using principal component analysis for malware detection: A case study”, Jan. 2019.
7. X. Sun, X. Li, K. Ren, J. Song, Z. Xu, J. Chen, “Rethinking compact abating probability modeling for open set recognition problem in cyber-physical systems”, J. Systems Architecture. 2019.
8. H. Zhang, X. Xiao, F. Mercaldo, S. Ni, F. Martinelli, “Classification of ransomware families with machine learning based on N -gram of opcodes”, Future Generation Computer Systems, pp. 211–221 2019.
9. Z. Bazrafshan, H. Hashemi, S. M. H. Fard, A. Hamzeh, “A survey on heuristic malware detection techniques”, in Information and Knowledge Technology (IKT), 5th Conference, IEEE, pp. 113–120, 2013
10. S. E. Coull and C. Gardner, “Activation analysis of a byte-based deep neural network for malware classification”. IEEE Security and Privacy Workshops , pp 21–27, 2019.
11. E. Raff, J. Barker, J. Sylvester, R. Brandon, B. Catanzaro, and C. K. Nicholas. “Malware detection by eating a whole exe”, Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
12. Yanfang Ye, Tao Li, Yong Chen, and Qingshan Jiang, “Automatic malware categorization using cluster ensemble”, in Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 95–104. ACM, 2010.
13. R. Veeramani, N. Rai, “Windows API based malware detection and framework analysis”, International Journal of Scientific & Engineering Research Volume 3, Issue 3, March 2012 .
14. M. Christodorescu, S. Jha, J. Kinder, S. Katzenbeisser, H. Veith, “Software transformations to improve malware detection”, Journal of Computer Virology, pp 253–265, 2007.
15. E. Raff, R. Zak, R. Cox, J. Sylvester, P. Yacci, R. Ward, A. Tracy, M. Mclean, C. Nicholas, “An investigation of byte n-gram features for malware classification”, Journal of Computer Virology and Hacking Techniques 2016
16. Y. Nagano, “Static analysis with paragraph vector for malware detection”, proceedings of International Conference on Ubiquitous Information Management and Communication, pages 1-7, 2017.
17. Y. Oyama, “Trends of anti-analysis operations of malwares observed in API call logs”, Journal of Computer Virology and Hacking Techniques, 2017.
18. S. Sibi Chakkaravarthy, D. Sangeetha, V. Vaidehi, “A survey on malware analysis and mitigation techniques”, Computer Science Review, pp 1–23, 2019.
19. Y. Ding, X. Xia, S. Chen, Y. Li, “A malware detection method based on family behavior graph”, Computer Security. pp 73–86, 2018.
20. W. Halfond, A. Orso, “Malware detection”, pp. 85–109, 2007.
21. A. Ray, “Introduction to Malware and Malware Analysis: A brief overview”, pp. 22–30, 2016.
22. N. Kawaguchi, K. Omote, “Malware function classification using apis in initial behavior”, in Proceedings of 10th Asia Joint Conference on Information Security, pp. 138–144, 2015.
23. U. Bayer, E. Kirda, C. Kruegel, “Improving the efficiency of dynamic malware analysis”, in Proceedings of the ACM Symposium on Applied Computing , pp. 1871, 2010.
24. Rad, B.B., Masrom, M., Ibrahim, S., “Camouflage in malware: from encryption to metamorphism”, International Journal of Computer Science Network Security, pp. 74–83, 2012.
25. Ye, Y., Wang, D., Li, T., Ye, D., “IMDS: intelligent malware detection system”, In Proceedings of 13th ACM SIGKDD, International Conference on Knowledge Discovery, Data Mining”, pp. 1043-1047, 2010.
26. M. Zubair Shafiq, Syed Ali Khayam, and Muddassar Farooq. “Embedded Malware Detection Using Markov n-Grams”, in Detection of Intrusions and Malware, and Vulnerability Assessment. Springer Berlin, Heidelberg, 2008.
27. R. Moskovitch, C. Feher, N. Tzachar, E. Berger, M. Gitelman, “Unknown malcode detection using OPCODE representation”, Intelligence and Security Informatics, pp 204-215, 2008.
28. K. Griffin, S. Schneider, X. Hu, T.-c. Chiueh, “Automatic generation of string signatures for malware detection”, pp. 101–120, 2009.
29. Nataraj, L., Karthikeyan, S., Jacob, G., Manjunath, B.S., “Malware images: visualization and automatic classification”, in Proceedings of the 8th International Symposium on Visualization for Cyber Security. ACM, New York, USA, pp. 4:14, 2011.
30. A. Shabtai, R. Moskovitch, C. Feher, S. Dolev, Y. Elovici, “Detecting unknown malicious code by applying classification techniques on OpCode patterns”, Security Information, 2012.
31. Chandrasekar Ravi, R Manoharan, “Malware Detection using Windows Api Sequence and Machine Learning”, International Journal of Computer Applications, Volume 43, April 2012.
32. I. Santos, J. Devesa, F. Brezo, J. Nieves, P. G. Bringas, “Opem: A static-dynamic approach for machine-learning-based malware detection”, in CISIS ’12-ICEUTE´ , page 271-280, 2013.
33. P. M. Comar, L. Liu, S. Saha, P. N. Tan, A. Nucci, “Combining supervised and unsupervised learning for zero-day malware detection”, in: Proceedings INFOCOM, IEEE, pp. 2022–2030, 2013.
34. Uppal, D., Sinha, R., Mehra, V., Jain, V., “Malware detection and classification based on extraction of API sequences”, in Proceedings of International Conference on Advanced Computer Commun. Informatics, pp. 2337-2342, 2013.
35. Salehi, Z., Sami, A., Ghiasi, M., “Using feature generation from api calls for malware detection”. Computer Fraud & Security, pp. 9–18 2014.
36. Joseph Sexton, Curtis Storlie, Blake Anderson, “Subroutine based detection of APT malware”, Journal of Computer Virology Hacking Techniques, pp 1-9, 2015.
37. Saxe, J., Berlin, K., “Deep neural network based malware detection using two dimensional binary program features”. In 10th International Conference on Malicious and Unwanted Software (MALWARE). IEEE, pp. 11-20, 2015.
38. U. Narra, F. Di, T. Visaggio, A. Corrado, T.H. Austin, M. Stamp, “Clustering versus SVM for malware detection”, Journal of Computer Virology Hacking Techniques, pp 213–224, 2016.
39. Ahmadi, M., Ulyanov, D., Semenov, S., Trofimov, M., Giacinto, “Novel feature extraction, selection and fusion for effective malware family classification”, in proceedings of the Sixth ACM Conference on Data and Application Security and Privacy, pp. 183–194, 2016.
40. Kolosnjaji, B., Zarras, A., Webster, G., Eckert, “Deep learning for classification of malware system call sequences”, in Lecture Notes of Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pp. 137-149, 2016.
41. B.N. Narayanan, O. Djaneye-Boundjou, T.M. Kebede, “Performance analysis of machine learning and pattern recognition algorithms for Malware classification”, in IEEE National Aerospace and Electronics Conference (NAECON) and Ohio Innovation Summit (OIS), pp. 338– 342, 2016.
42. S.D. Nikolopoulos, I. Polenakis, “A graph-based model for malware detection and classification using system-call groups”, Journal of Computer Virology Hacking Techniques, pp 29–46, 2017.
43. Xu, Z., Ray, S., Subramanyan, P., Malik, “Malware detection using machine learning based analysis of virtual memory access patterns”, Design, Automation & Test in Europe Conference & Exhibition, IEEE, pp. 169–174, 2017.
44. E. Raff, J. Sylvester, and C. Nicholas, “Learning the PE Header, Malware Detection with Minimal Domain Knowledge”, in Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, NY, USA: ACM, pp. 121–132, 2017.
45. Kotov, V., Wojnowicz, “Towards generic deobfuscation of windows api calls”, arXiv ,preprint arXiv:1802.04466, 2018.
46. Q. Le, O. Boydell, B. Mac, M. Scanlon, “Deep learning at the shallow end: Malware classification for non-domain experts”, Digital Investigation, pp S118–S126,2018.
47. Nguyen, M.H., Le Nguyen, D., Nguyen, X.M., Quan, T.T., “Autodetection of sophisticated malware using lazy-binding control flow graph and deep learning”. Computer Security, pages 128-155, 2018.
48. M. Krcal, O. Svec, M. Balek, and O. Jasek, “Deep Convolutional Malware Classifiers Can Learn from Raw Executables and Labels Only,” in ICLR Workshop, 2018.
49. Ni, S., Qian, Q., Zhang, R., “Malware identification using visualization images and deep learning”. Computer Security,2018.
50. Hemant Rathore, Swati Agarwal, Sanjay K. Sahay and Mohit Sewak, “Malware Detection using Machine Learning and Deep Learning ”, International Conference on Big Data Analytics, Springer, LNCS, Vol. 11297, pp. 402-411, 2018.
51. O. Suciu, S. E. Coull, and J. Johns, “Exploring adversarial examples in malware detection”, in IEEE Security and Privacy Workshops (SPW), pp 8–14, 2019.
52. Yuxin, D., Siyi, Z., “Malware detection based on deep learning algorithm”, Neural Comput. Appl. 31 (2), pp 461–472, Feb 2019.
53. M. Rabbani, Y.L. Wang, R. Khoshkangini, H. Jelodar, R. Zhao, P. Hu, “A hybrid machine learning approach for malicious behaviour detection and recognition in cloud computing”, Journal of Network and Computer Applications, 2020.
54. C. Yucel, A. Koltuksuz, “ Imaging and evaluating the memory access for malware”, Forensic Science International Digital Investigation, 2020.
55. Vasan, D., Alazab, M., Wassan, S., Safaei, B., & Zheng, Q. (2021). Windows malware detection using anomaly detection algorithms
56. Kumar, A. and Singh, B. and Gupta, D. Hybrid Malware Detection for Android using Static and Dynamic Analysis with RNN
57. D. Gibert, A. Smith, and M. Jones. "Enhanced Malware Detection through Interpretable Deep Learning and Gradient Boosting." In Proceedings of the 2023 IEEE Symposium on Security and Privacy (SP), pp. 123-137, 2023
58. Tabish, S.M., Shafiq, M.Z., Farooq, “Malware detection using statistical analysis of bytelevel file content”, in proceedings of the ACM SIGKDD Workshop on CyberSecurity and Intelligence Informatics, pp. 23–31, 2009.
59. A. K. Verma and S. K. Sharma, "Malware Detection Approaches using Machine Learning Techniques- Strategic Survey*,"*2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)*,* Greater Noida, India, 2021, pp. 1958-1962