VFAST Transactions on Software Engineering
http://vfast.org/journals/index.php/VTSE@ 2022, ISSN(e): 2309-3978, ISSN(p): 2411-6246
Volume 10, Number 3, July-September 2022 *pp: 51-57*

Identification of Malicious PDFs Using Convolutional Neural Networks

Umm-e-Hani Tayyab, Muhammad Zain, Faiza Babar Khan, Dr. Muhammad Hanif Durad

CIPMA Lab, DCIS, Pakistan Institute of Engineering and Applied Sciences, Pakistan
Corresponding author email:ummehani_19@pieas.edu.pk

ABSTRACT

With multiple possible carriers of malware, one of the most targeted file formats by malware writers is PDF format due to its inherent shortcomings as well as the limitations of PDF readers. The PDF-based attack is one of the most common attacks among document-based attacks. Some of the most luring features of PDF files include their widespread use which has replaced Word documents quite dominantly, secondly the ease of crafting a malicious PDF, and above all its capability of containing javascript. Initially, researchers focused on identifying malicious PDFs by comparing the structural changes between benign and malicious PDFs. Malware writers started using the evasion techniques to hide the malware so that structural changes can be hidden. Researchers started exploiting the benefits of Artificial Intelligence to combat these evasion techniques. We developed a convolutional neural network, trained to classify PDFs as benign and malicious using the byte level information, on a dataset of approximately 21,000 files. Our model can detect malicious PDFs without the human intervention and manual feature extraction. Moreover, our model detects malicious PDFs with 97% accuracy.

KEYWORDS

Malicious PDFs, Machine Learning, Convolutional Neural Network

JOURNAL INFO

HISTORY: Received: August 10, 2022 Accepted: September 25, 2022 Published: September 30, 2022

INTRODUCTION

Aggressive use of the internet has made cyberspace the most vulnerable area of technology. These vulnerabilities areexploited by evil components through different forms of malware. With multiple possible carriers of malware, oneof the most targeted file formats by malware writers in PDF format due to its inherent shortcomings as well as the limitations of PDF readers. According to the malware statistics report, 2022 of Comparitech [1], 35% of the new malware was detected being spread via PDF and Office files. The PDF-based attack is one of the most common attacks among document-based attacks [2]. Some of the most luring features of PDF files include their widespread use which has replaced MS Word documents quite dominantly, secondly the ease of crafting a malicious PDF, and above all its capability of containing javascript [3]. Structure of PDF files contain series of dictionary objects followed by compressed stream of data. These dictionary objects are connected to each other and can contain simpler objects as well. These objects define the actions taken by the pdf files. These objects main contain pages, fonts, images, embeddings etc. the flexibility of dictionary objects pose security threat to pdf files. Embedding of malicious javascript code into any dictionary object of pdf file can help attacker to exploit the weakness of pdf reader. Similarly, any malicious file can be embedded in pdf through these dictionary objects. Moreover, a pdf may contain a malicious ActionScript code which makes it a malicious pdf [15], [16].

Initially, researchers focused on identifying malicious PDFs by comparing the structural changes between benign and malicious PDFs [4] [5] [6] [7]. These methods are based on static features and are quite fast and robust. One of the drawbacks of these methods is the possibility of evasion of these techniques once when malware writers get aware of the distinguishing static features. Therefore, researchers shifted their research paradigm to the dynamic features based on javascript [8]. These methods also got failed due to obfuscation techniques adapted by malware writers. To combat all the evasion techniques, researchers started exploiting the benefits of Artificial Intelligence. Researchers started to pay attention to machine learning algorithms first and then moved towards neural networks to leave behind the efforts of feature engineering in machine learning [3]. Conventional machine learning algorithms need to be dealt by a person who is well versed in domain knowledge. Feature engineering is the phase where suitable feature needs to be selected for further pre-processing. Most of the times due to the curse of dimensionality, researchers try to focus over the subset of features [14]. Using specific features may hinder the learning process in case if suitable features are not focused on. Our major idea was to utilize all the features found in the metadata of pdfs. We tried to achieve our objective through converting the pdf files into images.

We focused over designing a model which saves us the laborious task of manual feature engineering, and which can be used to develop a real time system for detecting

malicious PDFs. In summary our contributions are listed below:

- Presents a model to detect malicious PDFs, which can be incorporated in real time system of malware detection in PDF files.
- Addresses the problem of manual feature engineering of PDF files by converting the files into images to make use of all features.
- Our model achieves higher accuracy than the baseline methods which use subset of structural features for detecting malicious PDFs.

The organization of our paper is shown in Figure1. Which

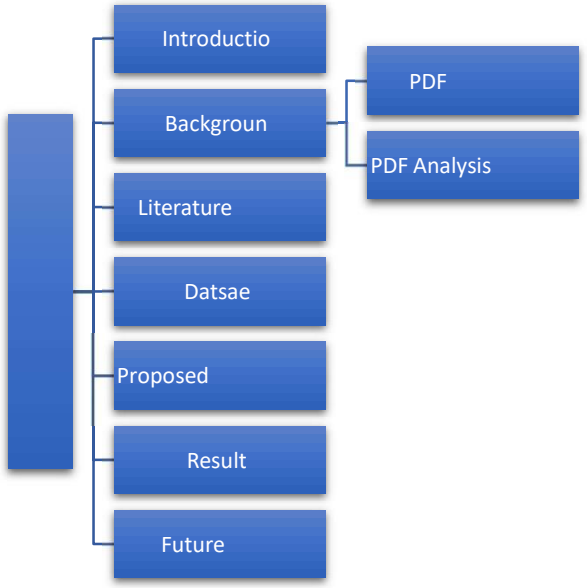

Figure 1. Paper Organization

contains section I introduces the problem and its domain. Section II sheds light on the structure of pdf files and methods of analyzing PDF files. Section III divides the literature into 3 categories: signature-based work to detect malware in PDF files, machine learning-based research to detect malware in PDFs, and neural network-based algorithms proposed for malware detection in PDFs. Section IV gives details of the dataset used in the research work. Section V proposes the neural network-based methodology. Section VI shows the results achieved as the result of experimentation. And Section

VII highlights the future work that can be carried out in this domain.

BACKGROUND

To find out the shortcomings of any file format that can facilitate the malware, we should at first have a clear pictureof the file structure.

1. *PDF Structure*

PDF is a portable document file format that can include many items like text, images, web links, other files, etc. [9]Figure 2 shows the basic structure of a PDF file.

A. *Header:*

This part of the PDF file specifies the version number used by the document. Currently, PDF files are using the version number of the format 1.N where N ranges from 0-7

B. *Body*

This section contains all the data that is shown to the user such as text streams, images, multimedia elements, etc.

C. *xref Table*

It is also known as a cross reference table which maintains the reference to all the objects present in the document. Each object corresponds to an entry in the table which is at a maximum of 20 bytes. The first 10 bytes of each entry represent the object's offset from the start of the document followed by a separator and then another number which represents the generation number of the object. It is followed by another separator after which either there is "f" or "n" to show whether the object is in use or not.

D. *Trailer*

This part of the document contains the specifications required by the reader. These specifications describe the methodology for accessing the objects by locating them through cross reference table. Therefore, all PDF readers should start reading the PDF file from the end of the file.

2. *PDF Analysis Methods*

On a classical note, if we dig into the history of malware detection, we come across 3 classical approaches [9]:

A. *Signature Based:*

This method corresponds to manual analysis of a malicious PDF for finding out a signature that can be a patternfrom the byte code of a file.

B. *Static Analysis*

As the name implies, it is a method in which indicators are found statically without executing the file. It refers to thepresence of some potentially harmful keywords like /Javascript, /Open-Action, or /GoTo. This method is not appropriate anymore because of some of the javascript that can be embedded without using /the Javascript tag.

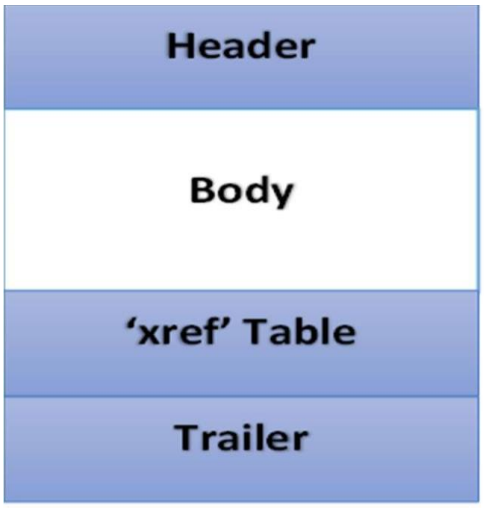

Figure 2 PDF Structure

C. *Dynamic Analysis*

This is a method in which files are executed in a sandboxed environment and during execution, features are extracted which represent the malicious behavior. These features can be some API calls or some network communication or any registry change etc.

LITERATURE REVIEW

Literature work found on malware detection in PDFs can be categorized into 3 categories as shown in Figure 3.

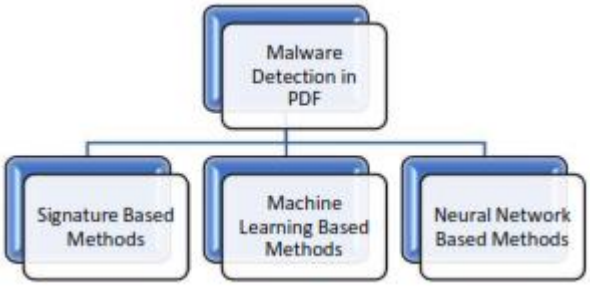

Figure 3 Malware Detection Methods in PDFs

Cross and Munson [10] performed both static and dynamic analysis of the PDF files and extracted a list of features that can be used for the detection of harmful actions such as: launching a program or running code embedded in PDF using javascript tag, and features describing the format of PDF, such as xref table or pages count. For detection,they used a very small and imbalanced dataset of 2677 benign and 89 malicious samples. They trained the decision tree on 5-fold cross-validation and detected 60% of malware with a precision of 80%. This work can be improved by increasing the amount of dataset which might ideally improve the result so that tested model can be incorporated in real time system of detecting malicious PDFs.

Stavrou and Smutz [4] performed the detection of malicious PDFs using features based on metadata and document structure. Structural features are based on string matching e.g /Font, /JS, etc., the count of these strings, and the average length of the streams. Metadata include higher level information e.g incremental updates of PDF document, and mismatch in the unique identifier of PDF. A training set of 10,000 files and a testing set of 100,000 files are used with 202 features selected manually through string extraction and they built a random forest classifier. The classifier was trained in 92 sec with an error rate of 19%. Since they used structural features to train their model, therefore, their model might not be able to detect malicious PDF if malware writer has used any evasion technique to hide the malware. Moreover 19% error rate is too high for a system to be implemented in a real time environment.

In addition to feature extraction through static analysis, Tzermiias et al. [8] analyzed the PDF to extract the objects and structure of the document and then extracted the JavaScript code for detecting the presence of shellcode. Out of 197 malicious, 176 are classified as malicious. Although their approach is very robust to obfuscation but relies only on JavaScript code and could not manage to detect malware exhibiting other techniques e.g /launch and /URWEtags.

Corum et al. [11] performed PDF malware detection by using image processing and visualization techniques to extract features. The PDF is converted to grayscale images using byte plot and Markov plot techniques. Key point descriptors and texture features are used to train Random Forest, K- nearest neighbor, and Decision Tree. Contagio dataset is used with 9000 benign and around 12000 malicious files. 10-fold cross validation is applied, and the best model classifies with an F1 score of 99.24%.

Srndic and Laskov [7] used the PDF structure based information to retrieve the tree like structure in the form of paths. Each path constitutes a complete path from the root to the leaf node. Paths found more than 1000 times are considered as features for SVM and DT machine learning models. The dataset contains 660,000 files, which include 120,000 malicious files from VirusTotal. The overall classification results show that SVM performs slightly better than DT with a detection accuracy of 99.8% with an FPR of 0.01%. One of the drawback of this work is that it cannot handle evasion techniques.

With the growth of data and innovation in obfuscation techniques, researchers shifted their research paradigm to neural networks. The major targeted file format

for malware detection using CNN was Portable Executable. Kolosnjaji et. al[12] using CNN in combination with Long short Term Memory for classifying malware types. Huang et. al [13] focused over API calls for classifying malware in PE file format. Deep Neural Networks are capable to deal with large amounts of data for producing reliable results secondly, the process of feature engineering gets automated.

Jeong et al. in [2] worked on detecting malicious streams found in PDF by using CNN on byte code. They claimed a detection rate of 97% with 99.7% precision. Model described by Joeng et. al [2] doesn't outperform conventional machine learning algorithm with a great margin but on the other hand takes a lot of time to operate in comparison to conventional machine learning based algorithms.

Saxe and Berlin [11] used manually extracted features rather than following the automatic extraction of features. They used 4 layer perceptron model with a 95% correct detection rate of maliciousfiles with 0.1% FPR.

PROPOSED METHODOLOGY

CNN (Convolutional Neural Network) is the most used algorithm in the domain of Deep Learning. This algorithm has advantage of automatic feature selection without the involvement of any domain expert over its predecessor algorithms. CNN architecture normally consists of convolutional layers, pooling layers and fully connected layers [17].

We designed 2 convolutional neural network models that take byte-level information from complete PDF files. The byte-level information is first preprocessed that is the size of the input vector is fixed. We took 200KB of each file which is passed to the embedding layer which transforms the input to 16 dimensional (embedding size) vectors of 200KB. These vectors are input for the rest of the model. The graphical representation of both the models is depicted in Figure 5 and Figure 6.

The model_1 in its simplest form comprises a single convolutional 1D layer, then a global max pooling layer which down samples the feature maps of the convolutional layer to the size of 128. This is followed by 2 fully connected layers of weights 128x128 and 128x1 respectively. To convert all the input byte-level information of files to vectors an embedding layer is applied. A dropout of 0.25 is used after the fully connected layers. Then it is followed by a sigmoid gate which classifies the input file.

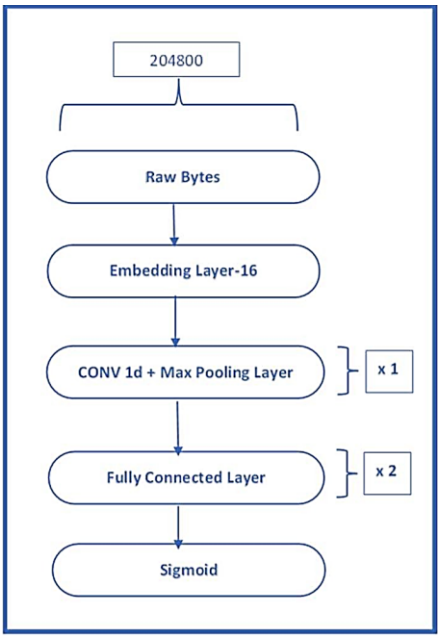

Figure 5 Model 1

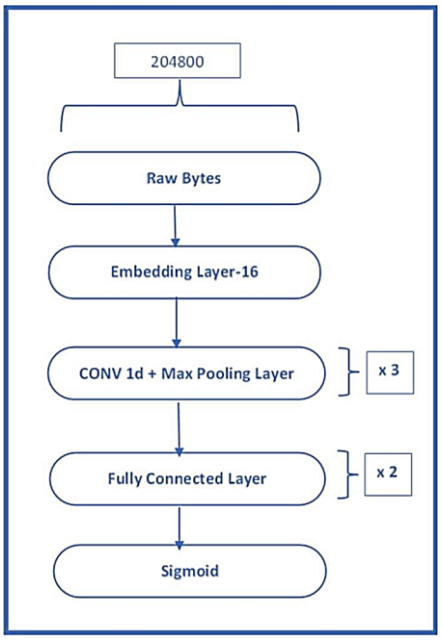

Figure 6 Model 2

Model_2 has 3 convolutional layers. Each layer is followed by a ReLU layer, max pooling layer, and batch normalization. The batch normalization is done so that each convolutional layer learns independently by normalizing the outputs of the preceding layers. These layers are followed by 2 fully connected layers and finally, the output passes through a sigmoid gate. The dropout used for this model is the same as 0.25 after the fully connected layer.

DATASET

For training, any classical machine learning or deep learning model authentic and large size dataset is required. Creation of the dataset manually was not possible because the available open-source tools can only create files that are of the same nature with very few changes. And files cannot be created one by one as it takes a huge amount of time. Considering all these facts, we chose to collect files from online freely available sources. The main source for malicious data is virusSign, Contagio dump, VirusShare, and VirusTotal. These sources provide limited data for research and development. The details of files collected from these sources are represented below in figure 4.

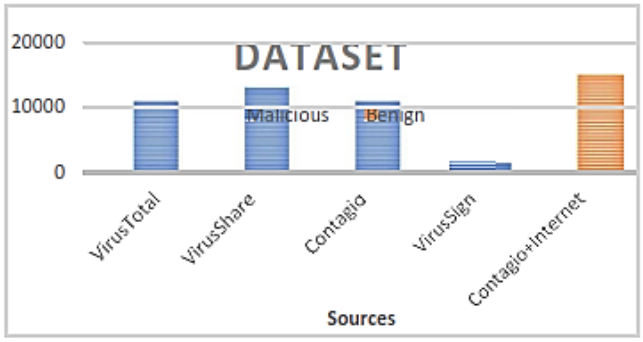

Figure 4 Dataset

We performed model training on two datasets, dataset_1 comprises of contagio dumps dataset. And it contains files generated in 2011. Dataset_2 is prepared using files from VirusTotal, VirusSign, and contagio/Internet, which were generated in the years 2017, 2018, and 2019. These files are more complex than the files in dataset_1 as they were generated in the year 2011 and contain malicious objects other than javascript codes as well. Table 1 shows the distribution of samples in datasets.

To make the data accessible for model training, the data should be prepared accordingly and for supervised learning, it should be labeled for each benign and malicious file. We created CSV files containing fields or hash, label, and first_seen for each file. We randomly set the time interval for the first_seen but in the future, this can be used for detection of newer or zero-day attacks given the dataset is evenly distributed for the last 3 or 4 years (both benign and malicious).

We utilized the approach of random splitting, and our dataset is divided as:

- Train_test_split ratio is 0.3, which means the training set contains 70% of files and the test set contains 30% of total files.
- Valid_test_split ratio is 0.2, means the validation set contains 20% of test set files. And remaining 80% of the total test set is used for testing.

Table 1 Dataset Distribution

Dataset	Training Samples	Validation Samples	Testing Samples
Datset_1	13985	1203	4789
Dataset_2	15068	1294	5145

EXPERIMENTS AND RESULTS

Jeong et. al [2] made use of privately collected PDF files and labelled them malicious on the basis of the presence of javascript codes and their results are shown in Table 2. Similarly, results claimed by Laskov et. al [7] are also obtained focusing over pdf files having maliciousness due to javascript code only. Whereas the dataset we collected comprises of pdf files containing maliciousness other than javascript also. Since the training data that we used to train our models contains different sources of malicious exes and jpeg files embedded in pdf files, therefore, our model will be capable of detecting malicious pdf files in real time more robustly. We conducted two different experiments using the above mentioned models.

Our first experiment was to check the performance of simple convnet by giving input of variable sizes to check the effect of zero padding. We gave dataset_1 to model 1 with sizes of 200KB, 100 KB, and 50 KB. With all the input sizes model 1 nearly performed the same way. Results are shown in Table 3. In the second experiment, we gave dataset_2 to both the models to check which model performed better. Results of this experiment show that simple convnet performed better over dataset_2.

Table 2 Experimental results of Base Paper [2]

Input Size (kB)	Model	Precision (%) Malicious / Benign	Recall (%) Malicious / benign
200	Conv+Conv+Pool+FC	99.78/100.0	97.27/92.62

Table 3 Experiment 1 Results

Dataset	Input Size (kB)	Model	Accuracy (%)	False Positive Rate	Precision (%) Malicious / Benign	Recall (%) Malicious / benign
Dataset_1	200	Model_1	99.68	0.01	99.84/98.40	99.04/99.81
Dataset_1	100	Model_1	99.14	0.0	99.07/99.20	99.04/99.23
Dataset_1	50	Model_1	99.09	0.01	98.21/99.53	99.44/98.51

Table 4 Results of Experiment 2

Dataset	Input Size (kB)	Model	Accuracy (%)	False Positive Rate	Precision (%) Malicious / Benign	Recall (%) Malicious / benign
Dataset_2	200	Model_1	97.05	0.03	96.23/97.87	97.88/96.21
Dataset_2	200	Model_2	90.92	0.14	86.73/96.09	96.47/85.43

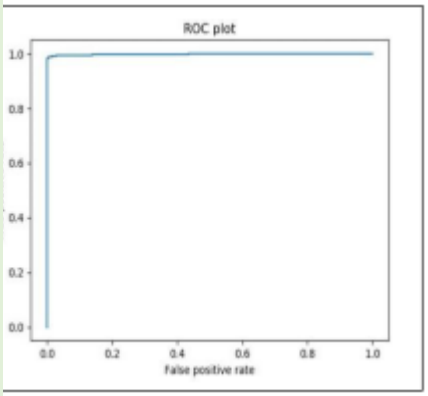

Figure 7 ROC for model_1 trained on dataset_1

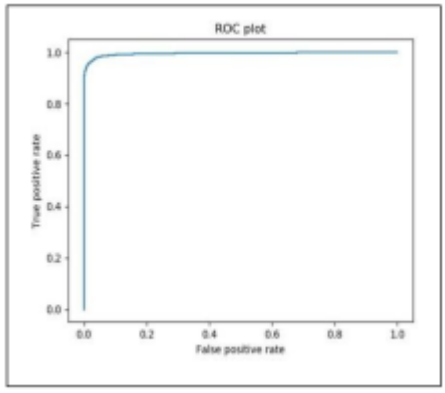

Figure 8 ROC model_1 trained on dataset_2

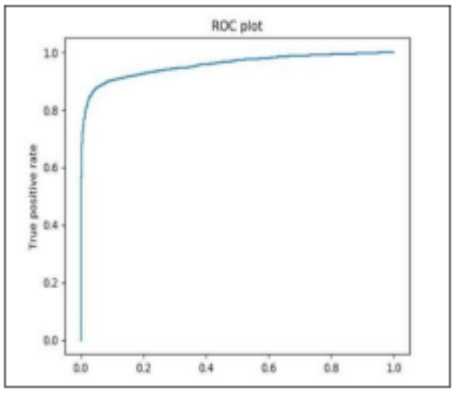

Figure 9 ROC for model_2 trained on dataset_2

We developed a convolutional neural network, trained to classify PDFs as benign and malicious using the byte level information, on a dataset of approximately 21,000 files. Our model can detect malicious PDFs without the need for static and dynamic signature identification or manual feature extraction. The Model works best with an accuracy of 97% to detect malicious files that are created after model training. This approach also works best against mimicry and reverse mimicry attacks which evade keywords and tag-based detection models as it is not based on keywords and tags. It can help researchers and malware companies to save time and detect zero-day attacks.

In future work, this work can be improved to detect malware families using unsupervised learning techniques. Additional deep neural network approaches can be used to enhance the detection rate.

REFERENCES

[1] https://www.comparitech.com/antivirus/malware-statistics-facts/

[2] Young-Seob Jeong, Jiyoung Woo, Ah Reum Kang, "Malware Detection on Byte Streams of PDF Files UsingConvolutional Neural Networks", *Security and Communication Networks*, vol. 2019. https://doi.org/10.1155/2019/8485365

[3] D. Liu, H. Wang and A. Stavrou, "Detecting Malicious Javascript in PDF through Document Instrumentation,"2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, 2020, pp. 100- 111, doi: 10.1109/DSN.2014.92.

[4] C. Smutz and A. Stavrou, "Malicious pdf detection using metadata and structural features," in Proceedings ofAnnual Computer Security Applications Conference (ACSAC), 2021

[5] N. Srndic and P. Laskov, "Detection of malicious pdf files based on hierarchical document structure," in NDSS,2013

[6] D. Maiorca, G. Giacinto, and I. Corona, "A pattern recognition system for malicious pdf files detection," in Proceedings of International conference on Machine Learning and Data Mining in Pattern Recognition (MLDM), 2012.

[7] P. Laskov and N. Srndi ˇ c, "Static detection of malicious ´ javascript-bearing pdf documents," in Proceedingsof Annual Computer Security Applications Conference (ACSAC), 2011.

[8] Z. Tzermias, G. Sykiotakis, M. Polychronakis, and E. P. Markatos, "Combining static and dynamic analysis forthe detection of malicious documents," in Proceedings of European Workshop on System Security (EUROSEC), 2011.

[9] M. Munson, *Deep PDF parsing to extract features for detecting embedded malware*. 2011.A. Corum, D. Jenkins, and J. Zheng, "Robust PDF malware detection with image visualization and processing techniques," in *2019 2nd International Conference on Data Intelligence and Security (ICDIS)*, 2019.

[10] J. Saxe and K. Berlin. Deep neural network based malware detection using two dimensional binary program features. In 2015 10th International Conference on Malicious and Unwanted Software (MALWARE), pages 11–20.IEEE, 2015.

[11] B. Kolosnjaji, A. Zarras, G. Webster et al. Deep learning for classification of malware system call sequences,"*Lecture Notes in Computer Science*, vol. 9992, pp. 137–149, 2016.

[12] W. Huang and J. W. Stokes, "MtNet: a multi-task neural network for dynamic malware classification," *Lecture Notes in Computer Science*, vol. 9721, pp. 399–418, 2016.

[13] M. Uddin, J. Lee, S. Rizvi, and S. Hamada, "Proposing enhanced feature engineering and a selection model for machine learning processes," *Appl. Sci. (Basel)*, vol. 8, no. 4, p. 646, 2018.

[14] D. Maiorca, G. Giacinto, and I. Corona, "A pattern recognition system for malicious PDF files detection," in *Machine Learning and Data Mining in Pattern Recognition*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2021, pp. 510–524..

[15] P. Anantharaman, S. Cheung, N. Boorman, and M. E. Locasto, "A format-aware reducer for scriptable rewriting of PDF files," in *2022 IEEE Security and Privacy Workshops (SPW)*, 2022.

[16] L. Alzubaidi *et al.*, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," *J. Big Data*, vol. 8, no. 1, p. 53, 2021.