Image-based PDF Malware Detection Using Pre-trained Deep Neural Networks

Tyler Nichols
Department of Computer Science and Engineering
Texas A&M University
College Station, TX
tnichols280@tamu.edu

Jack Zemlanicky
Department of Computer Science
Quinnipiac University
Hamden, CT
jzemlanicky@quinnipiac.edu

Zhirui Luo
Department of Computer Science and Engineering
New Mexico Institute of Mining and Technology
Socorro, NM
zhirui.luo@student.nmt.edu

Qingqing Li
Department of Computer and Information Sciences
Towson University
Towson, MD
qingqingli@towson.edu

Jun Zheng
Department of Computer Science and Engineering
New Mexico Institute of Mining and Technology
Socorro, NM
jun.zheng@nmt.edu

2024 12th International Symposium on Digital Forensics and Security (ISDFS) | 979-8-3503-3036-6/24/$31.00 ©2024 IEEE | DOI: 10.1109/ISDFS60797.2024.10527343

***Abstract*—PDF is a popular document file format with a flexible file structure that can embed diverse types of content, including images and JavaScript code. However, these features make it a favored vehicle for malware attackers. In this paper, we propose an image-based PDF malware detection method that utilizes pre-trained deep neural networks (DNNs). Specifically, we convert PDF files into fixed-size grayscale images using an image visualization technique. These images are then fed into pre-trained DNN models to classify them as benign or malicious. We investigated four classical pre-trained DNN models in our study. We evaluated the performance of the proposed method using the publicly available Contagio PDF malware dataset. Our results demonstrate that MobileNetv3 achieves the best detection performance with an accuracy of 0.9969 and exhibits low computational complexity, making it a promising solution for image-based PDF malware detection.**

***Index Terms*—PDF malware, deep learning, pre-trained deep neural networks, image visualization**

I. Introduction

Adobe's PDF (Portable Document Format) is a universally recognized file format developed to prioritize flexibility and user-friendliness. Documents in this format are accessible across a wide range of modern computing devices, irrespective of their hardware, software, or operating system. PDF files support diverse content types, such as images and text, and offer convenient features like hyperlinks for easy navigation. They can be swiftly generated, shared, downloaded, and viewed, with some PDFs allowing users to directly modify their content during viewing.

The PDF format's widespread adoption across various professional fields underscores its inherent flexibility, yet this very versatility has also rendered it a favored vehicle for malicious activities, including malware dissemination. Malicious exploitation of PDF files manifests through diverse tactics, ranging from embedding executable code that triggers upon opening to leveraging third-party programs to clandestinely introduce malware before distribution [1]. Furthermore, malware propagated via PDFs often operates covertly, seamlessly blending into the file's background to evade detection, thereby exhibiting minimal suspicious behavior in appearance and functionality [2], [3].

Current PDF malware detection methods can be divided into four main categories: keyword-based, tree-based, code-based, and learning-based [1]. Keyword-based methods look for suspicious keywords in indirect objects to identify malicious PDF files. Tree-based methods utilize the interconnections between objects to construct a tree structure for a PDF file. The trees for malicious files usually end with objects containing suspicious actions. Code-based methods detect malicious PDF files by analyzing embedded suspicious script code. Learning-based methods utilize different machine learning algorithms to build PDF malware detectors, which typically involve a feature extraction step to extract discriminative features used as input for learning algorithms. For example, Lux0R [4] uses features extracted from embedded JavaScript code to detect malicious PDF files. Hidost [5] extracts features from the logical structure of a PDF file to classify it as benign or malicious.

Image-based malware detection has become popular recently, involving the conversion of binary files into grayscale images first, followed by the classification of the images as malicious or benign [6], [7]. Corum et al. [8] proposed a robust image-based PDF malware detection method that employs manually extracted keypoint descriptors and texture features from the images to classify PDF files as benign or malicious. With the recent development of deep learning, deep neural network (DNN) models such as convolutional neural networks (CNNs) have been applied for image-based malware detection

979-8-3503-3036-6/24/$31.00 ©2024 IEEE

[9]–[12]. Unlike traditional methods, DNNs can automatically learn discriminative features from images for classification.

In this study, we explored a PDF malware detection method based on deep learning and image visualization. Specifically, we investigated the use of pre-trained DNN models for detecting PDF malware. These models were pre-trained on a large-scale image dataset, such as the ImageNet dataset [13], and can then be utilized for a broad range of other image classification tasks. Four classical pre-trained DNN models, MobileNet [14], ResNet [15], SqueezeNet [16], and VGG [17], were included in our study. We employed a publicly available PDF malware dataset for performance evaluation.

The rest of this paper is organized as follows. Section II introduces the background information about the PDF file format. The proposed method that utilizes pre-trained DNN models and image visualization for PDF malware detection is presented in Section III. Section IV describes the performance evaluation experiments and results. Finally, Section V concludes this paper.

II. PDF File Format

A PDF file comprises four main components: the header, body, cross-reference table, and trailer, as shown in Fig. 1. The header, typically brief, includes essential details like a unique format header and the PDF file version. The body contains all user-visible content such as images, text, streams, and other elements intended for viewer interaction. The cross-reference table, a collection of bytes, maps the location of each object within the PDF body, facilitating rendering by PDF readers. The trailer, located at the end of the file, provides instructions for software to access the cross-reference table, aiding in the orderly rendering of document objects. PDF readers follow a sequential bottom-up reading process, beginning with the trailer, accessing the cross-reference table, organizing the body's objects accordingly, and concluding with header-specific information.

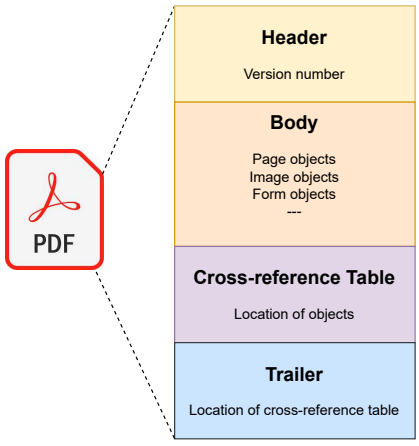

Fig. 1. PDF file structure

Malware attacks through PDF files frequently exploit components that do not impact the visual rendering of the final document, such as the header and body. For instance, the virus might embed itself within the file's body without being referenced in the cross-reference table, thereby remaining hidden from the viewer's detection. Alternatively, the malware could conceal itself within the file's header, activating only after the document is fully rendered and presented to the viewer.

III. Proposed PDF Malware Detection Method

The proposed method involves two steps for PDF malware detection, as shown in Fig. 2. First, a PDF file, whether benign or malicious, is converted into a fixed-size grayscale image. Then, the image is fed into a pre-trained DNN model to be classified as benign or malicious.

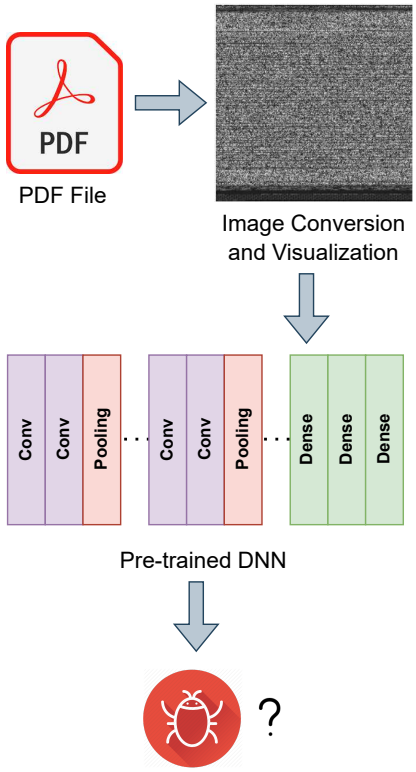

Fig. 2. Proposed image-based PDF malware detection method

A. Image Conversion and Visualization

A simple and effective method to convert any file to a grayscale image for malware detection is the byte plot proposed in [6]. To create a byte plot image, the file is read as a sequential byte stream, and each byte is converted into a pixel with a value ranging from 0 to 255. The width of the generated image is determined by the file size. For example, if the file size is less than 10 kB, the image width is set to 32, while it is set to 1,024 if the file size is larger than 1,000 kB.

Because the size of the byte plot image generated by the method of [6] varies with the file length, while pre-trained DNNs require fixed-size inputs, we adopted a modified approach to convert PDF files of various sizes into fixed-size

grayscale images. Considering the size of a PDF file is N bytes, the file is converted into an image with dimensions of $n \times n$ first, where n is equal to $\lceil \sqrt{N} \rceil$. If N is less than $n \times n$, the file is padded with 0s. After the conversion, all images are resized to a fixed size of $k \times k$. In our study, k is set to 256. Fig. 3 shows an example of a 256×256 image generated from a PDF file.

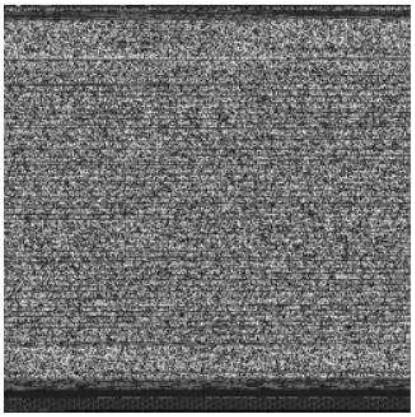

Fig. 3. An example of a 256×256 image generated from a PDF file

B. Pre-trained DNNs

In our study, we investigated four pre-trained DNNs: MobileNetv3, ResNet101, SqueezeNet1.1, and VGG19, all of which were pre-trained with the ImageNet dataset and have been widely used for various image classification tasks [18]–[21]. We transferred the knowledge of these pre-trained models to our PDF malware detection problem through fine-tuning.

- **MobileNetv3**: MobileNetV3 is the third version of MobileNet [14]. The primary objective of MobileNet is to develop a DNN architecture tailored for deployment on mobile devices. Unlike its predecessors, MobileNetV3 was developed using the neural architecture search (NAS) technique for optimized architecture design. The architecture of MobileNetV3 is illustrated in Fig. 4. MobileNetV3 incorporates several advanced deep learning techniques, such as lightweight depthwise separable convolution operations, linear bottleneck layers, inverted residual blocks, and the h-swish activation function.
- **ResNet101**: ResNets (Residual Networks) are a set of CNN models designed with residual learning [15]. A ResNet is formed by stacking residual blocks on top of each other. Each residual block contains two or three Conv layers with a skip connection connected directly from the input to the output. ResNet models are named based on the number of weighted layers in the model. We adopted ResNet101 in our study, which has an architecture shown in Fig. 5. There are a total of 33 residual blocks in ResNet101, each containing a 1×1 Conv layer, followed by a 3×3 Conv layer, and another 1×1 Conv layer, as shown in Fig. 6.

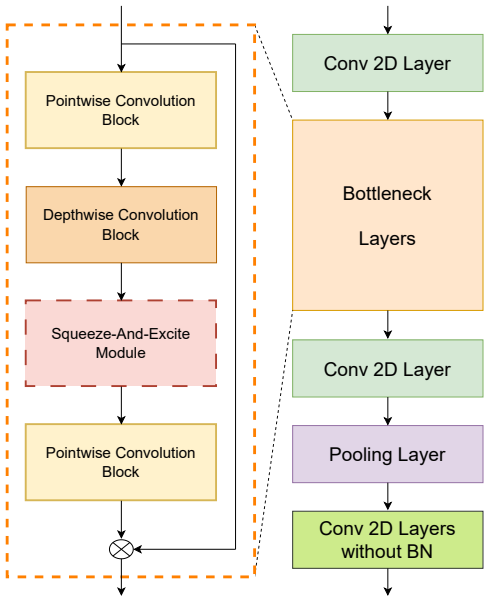

Fig. 4. MobileNetv3 architecture

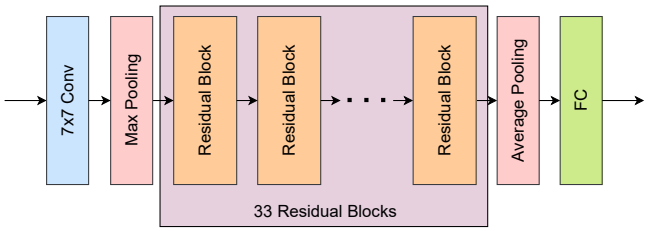

Fig. 5. ResNet101 architecture

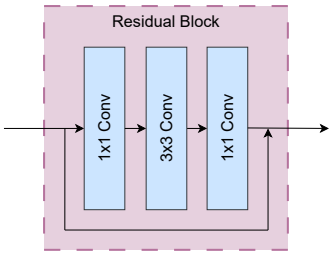

Fig. 6. Residual block for ResNet

- **SqueezeNet1.1**: SqueezeNet is a DNN architecture developed in [16] to achieve the same level of performance as AlexNet on the ImageNet dataset with a much smaller model size. The architecture of SqueezeNet1.1 is illustrated in Fig. 7. conv1 is a 3×3 convolution layer with a stride of 2. poo1, pool3, and poo5 are 3x3 max-pooling layers with a stride of 2. There are eight fire modules in the architecture, which are the building blocks designed to achieve the goal of maintaining mode performance while having fewer parameters. As can be seen from Fig. 7, each fire module consists of a 1x1 squeeze convolution layer and an expand layer with a mix of 1x1 and 3x3

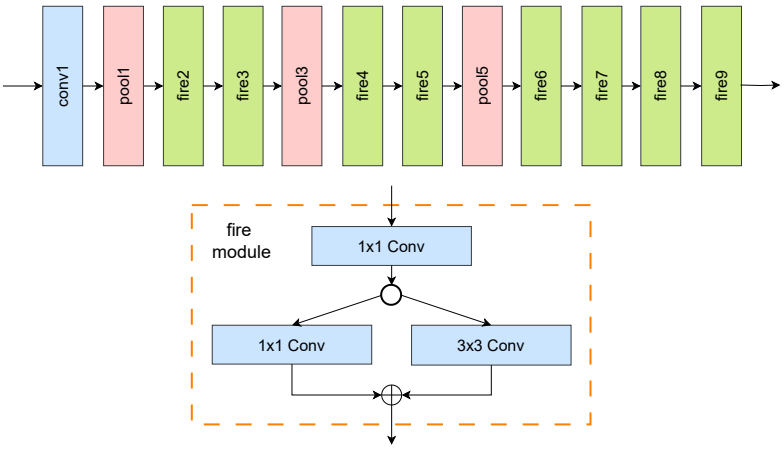

Fig. 7. SqueezeNet architecture

convolution filters.

- **VGG19**: VGG, also known as VGGNet, is a CNN architecture proposed in [17] that improves model performance by increasing model depth. The architecture of VGG has inspired the development of other pre-trained DNN models, such as ResNet. VGG19 is the VGG architecture that contains 19 weight layers, as illustrated in Fig. 8. The 16 convolution layers in VGG19 are divided into four groups, with two in each of the first two groups and four in each of the last two groups. A max-pooling operation is applied at the end of each group of convolution operations.

IV. Experiments and Results

We utilized the Contagio PDF malware dataset [22] to assess the performance of the proposed method, which has been widely used in learning-based PDF malware detection research [4], [5], [8]. This dataset comprises 10,980 malicious and 9,000 benign PDF files. For our experiments, we employed 10-fold cross-validation. Within each fold, 20% of the training set served as validation data. We retained the structure of the classification head for each pre-trained model, adjusting the size of the output layer to two classes: benign or malicious. Additionally, we modified the number of channels in the input layer of each model from 3 to 1, given that the inputs are grayscale images. In addition to the four pre-trained DNN models, we implemented the CNN model proposed in [10] as a reference method. This method converts the byte stream of a PDF file into task-specific embedding using an embedded layer before the convolution and max-pooling layers. All models underwent training for 100 epochs. Subsequently, the architecture yielding the best validation performance for each model was selected for testing. We used accuracy as the performance metric for our experiments, which is calculated as the ratio of correctly classified samples to the total number of samples in the testing set.

Table I presents the results of our performance evaluation experiments, showcasing the detection accuracy aggregated

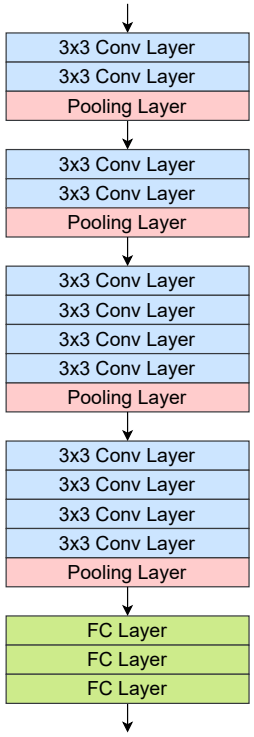

Fig. 8. VGG19 architecture

over the 10 folds. It is evident that all four pre-trained DNN models outperform the reference CNN model due to their deeper architectures. Among these models, MobileNetV3 and ResNet101 achieve significantly better performance compared to SqueezeNet1.1 and VGG19. Specifically, MobileNetV3 stands out as the best-performing model, with a detection accuracy of 0.9969. Table II shows the computational complexity of the four pre-trained DNN models in terms of the number of parameters (**Params**) and the number of floating point operations (**FLOPs**). MobileNetV3 and SqueezeNet1.1 have significantly lower computational complexity compared

to ResNet101 and VGG19, as they were designed to be deployed on resource-limited devices like mobile phones. It can be seen that MobileNetV3 has the lowest FLOPs among the four pre-trained models due to the use of lightweight depthwise separable convolutions and the NAS approach for architecture optimization. Considering both performance and computational complexity, MobileNetV3 clearly emerges as a promising solution for image-based PDF malware detection.

TABLE I
PERFORMANCE OF PRE-TRAINED DNN MODELS AND THE REFERENCE CNN MODEL

Model	Accuracy
CNN [10]	0.9495
MobileNetv3	**0.9969**
ResNet101	0.9959
SqueezeNet1.1	0.9800
VGG19	0.9532

TABLE II
COMPUTATIONAL COMPLEXITY OF PRE-TRAINED DNN MODELS (B= 10^9, M= 10^6)

Model	Params	FLOPs
MobileNetv3	4.2 M	566.7 M
ResNet101	42.5 M	20.2 B
SqueezeNet1.1	0.72 M	665.3 M
VGG19	139.6 M	51.1 B

V. CONCLUSIONS

PDF has become a primary target for malware attacks due to its flexible file structure and the capability of embedding a variety of content types. In this paper, we present an image-based PDF malware detection method that utilizes pre-trained DNN models. We employed an image visualization technique to convert PDF files into fixed-size grayscale images, which serve as the input for deep learning models. Unlike traditional methods that rely on manually extracted features for PDF malware detection, deep learning models can automatically extract discriminative features from input images for classification.

In this study, we specifically investigated the use of four classical pre-trained DNN models for image-based PDF malware detection: MobileNetv3, ResNet101, SqueezeNet1.1, and VGG19. We evaluated the performance of the proposed method using the popular Contagio PDF malware dataset. Our results indicate that all four pre-trained DNN models outperform a reference CNN model proposed for PDF malware detection, demonstrating the validity of using pre-trained DNN models for this task. Among the four pre-trained models, MobileNetv3 achieves the highest detection accuracy and exhibits low computational complexity. The next step in our work is to develop an improved MobileNetv3 model to further enhance detection performance.

ACKNOWLEDGMENT

This work is supported by the National Science Foundation under grant no. CNS-2150145.

REFERENCES

[1] D. Maiorca and B. Biggio, "Digital investigation of PDF files: Unveiling traces of embedded malware," *IEEE Security & Privacy*, vol. 17, no. 1, pp. 63–71, 2019.
[2] J. Park and H. Kim, "K-depth mimicry attack to secretly embed shellcode into PDF files," in *International Conference on Information Science and Applications*. Springer, 2017, pp. 388–395.
[3] C. Mainka, V. Mladenov, and S. Rohlmann, "Shadow attacks: Hiding and replacing content in signed PDFs," in *NDSS*, 2021.
[4] I. Corona, D. Maiorca, D. Ariu, and G. Giacinto, "Lux0r: Detection of malicious PDF-embedded javascript code through discriminant analysis of API references," in *Proceedings of the 2014 Workshop on Artificial Intelligent and Security Workshop*, 2014, pp. 47–57.
[5] N. Šrndić and P. Laskov, "Hidost: a static machine-learning-based detector of malicious files," *EURASIP Journal on Information Security*, vol. 2016, pp. 1–20, 2016.
[6] L. Nataraj, S. Karthikeyan, G. Jacob, and B. S. Manjunath, "Malware images: visualization and automatic classification," in *Proceedings of the 8th International Symposium on Visualization for Cyber Security*, 2011, pp. 1–7.
[7] K. Kancherla and S. Mukkamala, "Image visualization based malware detection," in *2013 IEEE Symposium on Computational Intelligence in Cyber Security (CICS)*. IEEE, 2013, pp. 40–44.
[8] 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)*. IEEE, 2019, pp. 108–114.
[9] S. Venkatraman, M. Alazab, and R. Vinayakumar, "A hybrid deep learning image-based analysis for effective malware detection," *Journal of Information Security and Applications*, vol. 47, pp. 377–389, 2019.
[10] R. Fettaya and Y. Mansour, "Detecting malicious PDF using CNN," *arXiv preprint arXiv:2007.12729*, 2020.
[11] D. Vasan, M. Alazab, S. Wassan, H. Naeem, B. Safaei, and Q. Zheng, "Imcfn: Image-based malware classification using fine-tuned convolutional neural network architecture," *Computer Networks*, vol. 171, p. 107138, 2020.
[12] Y. Jian, H. Kuang, C. Ren, Z. Ma, and H. Wang, "A novel framework for image-based malware detection with a deep neural network," *Computers & Security*, vol. 109, p. 102400, 2021.
[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.
[14] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1314–1324.
[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[16] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.
[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[18] Z. Luo, Q. Li, and J. Zheng, "A study of adversarial attacks and detection on deep learning-based plant disease identification," *Applied Sciences*, vol. 11, no. 4, p. 1878, 2021.
[19] S. B. R. Prasad and B. S. Chandana, "Mobilenetv3: a deep learning technique for human face expressions identification," *International Journal of Information Technology*, vol. 15, no. 6, pp. 3229–3243, 2023.
[20] J. Cui, X. Luo, Z. Wu, J. Zhou, H. Wan, X. Chen, and X. Qin, "High-precision inversion of shallow bathymetry under complex hydrographic conditions using VGG19—a case study of the Taiwan banks," *Remote Sensing*, vol. 15, no. 5, p. 1257, 2023.
[21] Q. Huang, H. Ding, and M. Effatparvar, "Breast cancer diagnosis based on hybrid SqueezeNet and improved chef-based optimizer," *Expert Systems with Applications*, vol. 237, p. 121470, 2024.
[22] Contagio, 2013. [Online]. Available: http://contagiodump.blogspot.com/2013/03/16800-clean-and-11960-malicious-files.html