Computers & Security 130 (2023) 103265

Contents lists available at ScienceDirect

Computers & Security

journal homepage: www.elsevier.com/locate/cose

Improving ransomware detection based on portable executable header using xception convolutional neural network

Caio C. Moreira[a,*], Davi C. Moreira[a], Claudomiro de S. de Sales Jr.[b]

[a] Electrical Engineering Graduate Program, Federal University of Pará, Belém, PA, 66075-110, Brazil
[b] Laboratory of Computational Intelligence and Operational Research, Federal University of Pará, Belém, PA, 66075-110, Brazil

ARTICLE INFO

Article history:
Received 3 August 2022
Revised 2 February 2023
Accepted 16 April 2023
Available online 18 April 2023

Keywords:
Ransomware
Static analysis
PE header
Xception
Convolutional neural network
Image classification

ABSTRACT

All malware are harmful to computer systems; however, crypto-ransomware specifically leads to irreparable data loss and causes substantial economic prejudice. Ransomware attacks increased significantly during the COVID-19 pandemic, and because of its high profitability, this growth will likely persist. To respond to these attacks, we apply static analysis to detect ransomware by converting Portable Executable (PE) header files into color images in a sequential vector pattern and classifying these via Xception Convolutional Neural Network (CNN) model without transfer learning, which we call Xception ColSeq. This approach simplifies feature extraction, reduces processing load, and is more resilient against evasion techniques and ransomware evolution. The proposed method was evaluated using two datasets. The first contains 1000 ransomware and 1000 benign applications, on which the model achieved an accuracy of 93.73%, precision of 92.95%, recall of 94.64%, and F-measure of 93.75%. The second dataset, which we created and have made available, contains 1023 ransomware, grouped in 25 still active and relevant families, and 1134 benign applications, on which the proposed method achieved an accuracy of 98.20%, precision of 97.50%, recall of 98.76%, and F-measure of 98.12%. Furthermore, we refined a testing methodology for a particular case of zero-day ransomware attacks detection—the detection of new ransomware families—by adding an adequate amount of randomly selected benign applications to the test set, providing representative evaluation performance metrics. These results represent an improvement over the performance of the current methods reported in the literature. Our advantageous approach can be applied as a technique for ransomware detection to protect computer systems from cyber threats.

1. Introduction

Although malware attacks are harmful to computer systems by definition, one of the most destructive forms of malware is ransomware. This malicious software is used for digital extortion, which in most cases blocks access to user files through encryption until the victim pays a ransom in cryptocurrencies to obtain the valid decryption key. This key is necessary for file recovery, but receiving the key from the thief is not guaranteed even after paying the ransom (Beaman et al., 2021; Kapoor et al., 2022).

This scenario paints an alarming picture. This cybersecurity threat is considered one of the ten worst threats amid the COVID-19 pandemic owing to the significant increase in attacks on companies and government institutions, as working from home necessitated workers to connect remotely to corporate networks (Ahmad, 2020; Khan et al., 2020). Healthcare organizations were primary victims of digital attacks during the pandemic. This sector is one of the most vulnerable to cyber-attacks as it expands and moves to digitally-enabled healthcare services, allowing digital criminals to exploit the weaknesses and security vulnerabilities exposed by these rapid shifts (Pranggono and Arabo, 2021; Thamer and Alubady, 2021). Therefore, the increase in these attacks has led users, organizations, and governments to protect and create backups of their critical data. However, owing to ransomware's highly profitable nature, these attacks constantly evolve to bypass current protection mechanisms and improve their encryption process (Kolodenker et al., 2017). Following the emergence of ransomware, Ransomware-as-a-Service (RaaS) has become a franchise available on darknet marketplaces, producing customized ransomware with the same code base and enabling ambitious cyber criminals and malicious non-technical users to participate in this unethical industry (Meland et al., 2020). Consequently, ransomware has been developed to such a dangerous degree that it has established a cooperative mechanism between criminals, both computer-savvy and non-skilled. The increase in this type of malware will likely persist in the cyber world for the coming years.

[*] Corresponding author.
E-mail address: caiomoreira@ufpa.br (C.C. Moreira).

https://doi.org/10.1016/j.cose.2023.103265
0167-4048/© 2023 Elsevier Ltd. All rights reserved.

In general, defenses against known ransomware are the most effective and commonly used by protection software. Their methods rely comparisons based on signatures that already exist in a database. Unfortunately, in previously unknown attacks (zero-day), current methods have difficulty detecting these and yield high rates of false positives and negatives (Kok et al., 2022). Most of these approaches use Machine Learning (ML) and Deep Learning (DL) methods (Kok et al., 2019; Moussaileb et al., 2021).

As a specific type of malware, ransomware is structurally different from benign files, with its typical subroutines involving obtaining system information, mapping the victim environment to locate user files, invoking an Application Programming Interface (API) for file encryption, and connecting to a Command-and-Control (C2C) server (Hampton et al., 2018; Hull et al., 2019; Kapoor et al., 2022). There are two main types of analysis for ransomware detection: dynamic and static. In dynamic analysis, the program under investigation is run in an isolated environment to avoid possible damage to the system, and its behavior is monitored during execution to determine whether the sample is ransomware. The advantage of this type of analysis is that it makes evading techniques more difficult. Moreover, it allows researchers to analyze and understand how these dangerous programs work in the Operating System (OS). However, this approach is typically slow and less safe because it requires an isolated environment to actually execute the program (Kok et al., 2022).

For static analysis, researchers disassemble suspicious software and extract information regarding the structure and content of the sample without running it. The major advantage of this approach is its ability to identify harmful instructions before the program executes. Typically, this type of analysis is fast and safe. Following the static analysis, many features such as PE[1] header information, hash values, API calls, a control flow graph (CFG), byte sequence N-grams, opcode frequencies, and string signatures can be extracted. Malware detection software is designed using these types of features (Ucci et al., 2019). Because ransomware is typically installed through phishing emails and malicious websites, its static analysis can be automated to search for Indicators of Compromise (IOC). Security solutions can be configured to scan for IOC on PE headers, file hashes, network traffic, process indicators, and even file names. When a potential IOC is spotted, it can be flagged and investigated for further analysis (Preuveneers and Joosen, 2021; Verma et al., 2018). The goal of static analysis is to detect ransomware before it has the opportunity to encrypt any files or cause any damage. However, this approach is more susceptible to being deceived by obfuscation, polymorphism, encryption, and anti-disassembler techniques (Oz et al., 2022).

These two types of analyses have their advantages and disadvantages and can be combined in a hybrid approach allowing researchers to benefit from the advantages of both strategies (Ferrante et al., 2018; Oz et al., 2022). However, our research aims to improve a determined method of static analysis using PE header information, which shows the structure of a file and provides useful content about the nature of the executable to distinguish between malware and goodware (a benign application).

According to El-Shafai et al. (2021), visual images of various malware families differ in layout and form. A malware family is a collection of malware samples with the same code base. Consequently, each malware family has its own visual properties and similarities that are different from other malware families. Ransomware is, by definition, a particular malware family with subfamilies.

Motivated by this visual similarity, our proposed approach extracts raw structure information from PE headers, summarizes it into images, and identifies patterns to differentiate ransomware from goodware using the Xception CNN model, which can be applied as a technique for anti-virus protection systems. The key advantages of this approach over other static analysis methods are that it does not require a significant understanding of the PE header structure and does not need to distinguish between variable-length fields, making feature extraction a quick and straightforward process. In addition, visualization analysis is more resilient against sophisticated malware evolution over time and anti-malware evasion techniques (Hemalatha et al., 2021).

We also refine and apply a testing methodology for detecting new ransomware families, a particular case of zero-day attack detection where no samples from a given test family are seen during model training. This type of test is crucial given the rapid proliferation of new families coming from RaaS and such detection is possible by considering that new ransomware families often exhibit similar characteristics in the core part of the attacks (Hassan, 2019; Zhang et al., 2019).

For these purposes, two datasets were used by extracting the header from PE files: the first dataset contains 1000 ransomware and 1000 goodware samples and, the second dataset, which we created and have made available, contains 1023 ransomware, grouped in 25 still active and relevant families, and 1134 goodware samples.

The remainder of this paper is organized as follows. Section 2 presents related work, their main results, advantages, and limitations. Section 3 details the materials and methods used in this study. In Section 4, we present our results and then discuss and compare them with related works. Finally, Section 5 provides the overall conclusions obtained through the study and suggests future directions for research.

2. Related works

Advances in general malware detection based on the header of PE files were made in recent works. The typical proposal of this type of research involves extracting essential features from the malware's header and classifying them using methods such as ML (Aggarwal et al., 2022; Maleki et al., 2019; Rezaei and Hamze, 2020; Rezaei et al., 2021) and DL (Moti et al., 2019, 2021; Wen and Chow, 2021; Noever and Noever, 2021) algorithms. The results of these studies were promising as they achieved good accuracy levels. However, these studies did not focus on a specific type of malware, which can lead to less accurate investigations. Common malware typically attacks the system, compromising its functionality and creating vulnerabilities for intrusion and monitoring activities. In contrast, the focus of ransomware is the information in the users' files, the most valuable assets for people and organizations because of their intangible value. Thus, specialized solutions for ransomware detection are critical owing to the great danger of file loss.

Poudyal et al. (2019) proposed a PEFile analysis technique for studying ransomware using header information with section portions of the binary executable file. Their research verified that acquiring valuable information about the behavior and characteristics of ransomware is possible with the features obtained with static analysis, including packer status, compile date and time, cryptographic functions, and Dynamic-link Library (DLL). Their work can help develop ransomware detection systems, although a shortcoming is that it only investigates one ransomware family: Locky.

Vidyarthi et al. (2019) investigated the specific properties of ransomware, other malware, and benign executables based on the discriminant characteristics of the PE file. In the static analysis,

[1] File format for executables, object code, DLLs, and others used in versions of Windows OS. The name "Portable Executable" refers to the format being not architecture-specific.

the PE file was disassembled to extract metadata from the header fields. They identified 60 static properties to enable classification and spotted specific ransomware properties, including the presence of packer, entropy of the file, text and data sections, presence of common strings, command for registry key modification, and DLL used for network communication. The classification was performed using Random Forest (RF), Decision Tree (DT) (J-48), and Naive Bayes (NB), where RF performed best. Their paper complemented contemporary anti-malware techniques; however, a significant understanding of the header structure is required to extract the features.

Manavi and Hamzeh (2021) built a Long Short Term Memory (LSTM) network to process the sequence of bytes that constructs the header to distinguish the samples from ransomware and goodware. They used a dataset with 1000 ransomware and 1000 goodware samples to evaluate their proposed method. They suggested a strategy that used two LSTM units and one Dense layer to train the best model with the fewest layers possible while avoiding overfitting. They realized an accuracy of 93.25%, precision of 93.33%, recall of 93.25%, and F-measure of 93.24%. The same authors presented a graph embedding-based static analysis method for detecting ransomware in Manavi and Hamzeh (2022a). This approach uses PE headers to form a graph, which is then mapped in an eigenspace using the Power Iteration method. This mapping transforms an executable file into a feature vector used to train a RF classifier. One of the datasets they worked on was the same as in the authors' previous research (Manavi and Hamzeh, 2021), in which they achieved an accuracy of 93.30%, precision of 93.48%, recall of 93.30%, and F-measure of 93.28%.

Manavi and Hamzeh (2022b) proposed a static analysis method for detecting ransomware using CNN in which the main steps were to extract the header from the executable files, build a gray-scale image in a zigzag pattern using the extracted headers, train their own CNN model based on the images, and test the model with unseen samples. A dataset they worked on was the same as in the authors' previous research (Manavi and Hamzeh, 2021). They proposed a CNN sequential model with 12 layers deep, including batch normalization, convolution, maxpooling, dropout, flatten, and dense, with an input shape of $32 \times 32 \times 1$, training for 100 epochs to achieve a 93.33% accuracy, 93.40% precision, 93.33% recall, and 93.34% F-measure. The advantage of this method is that using raw PE header does not need a robust understanding of its structure, and no distinction between variable-length fields is required. Nevertheless, the choices for the generated images, particularly the zigzag vector pattern and gray-scale, are suboptimal options for classification in this domain, as demonstrated in our results.

A limitation of Manavi and Hamzeh (2021, 2022a, 2022b) is that they did not test the detection of new ransomware families. This could be accomplished if, during sample collecting, they categorized each ransomware under a common family name, which are typically provided by vendors of protection systems.

Sgandurra et al. (2016) proposed the EldeRan, an ML approach to classifying ransomware based on their behavior using Regularized Logistic Regression (RLR) as the classifier algorithm. They built and shared a dataset containing 582 ransomware, categorized as 11 families, and 942 goodware samples with 30,967 features, representing several Windows events, which is one of the few datasets available to the public (Fernando et al., 2020). Sgandurra et al. (2016) also tested the detection of new ransomware families by isolating one family for testing and training with the remaining ransomware and goodware samples. They achieved a weighted accuracy average of 93.30%. However, their testing methodology does not evaluate all outcome possibilities because it tests with only positive samples, thereby reducing the range of applicable evaluation metrics.

Zahoora et al. (2022) explored the capabilities of Zero-shot Learning (ZSL) in a two-stage process to propose a new deep contractive autoencoder-based attribute learning (DCAE-ZSL) technique and inference stage method based on heterogeneous voting ensemble (DCAE-ZSL-HVE). Their research focused on detecting zero-day ransomware attacks, particularly on detecting new ransomware families. They used the same dataset as Sgandurra et al. (2016), but improved the approach for detecting new ransomware families by including goodware in the test set, providing representative evaluation metrics; however, they evaluated only four of the 11 certain families. In addition, the four families were evaluated in the same test set, i.e., the training set contained seven families and the test set the four remaining families. Their findings could have been enhanced by maximizing the number of families in the training set and isolating one family for testing.

Although previous research has yielded promising results, more comprehensive static analysis models of PE headers and a refined testing methodology for detecting new ransomware families may produce more accurate and dependable results. Table 1 summarizes the advantages and limitations of each related work and compares them with this study. As demonstrated by our test methodology and results, our study fills the gaps in related studies while maintaining their advantages. This study can improve the development of detection solutions for a specific type of malware, complementing contemporary anti-malware techniques by analyzing static PE header information without requiring a prior comprehensive understanding of its structure or a distinction between variable-length fields. Our study uses better choices for the generated images in this classification domain, such as the vector pattern and color palette. Furthermore, we create and share a new and updated dataset with the categorization of the most relevant and still active 25 ransomware families according to protection system vendors. In addition, we refine a testing methodology for detecting new ransomware families and test its performance. Finally, our work achieves the best performance results compared with those in related work.

3. Materials and methods

This section describes the methodology, datasets, tools, evaluation metrics, experiment environment, parameters, and hyperparameters used in this study. Fig. 1 illustrates the high-level workflow of the proposed static analysis method. First, we extract raw header information from the executable sample (step 1), then convert it from a one-dimensional vector to a two-dimensional square matrix following a sequential pattern (step 2). The resulting matrix is converted to an image using a continued color palette (step 3). Finally, the color image generated in a sequential pattern is fed to the trained Xception CNN model, called Xception ColSeq (step 4), which classifies whether it is ransomware or goodware (step 5).

3.1. Data description

The complete header of a PE file includes the MS-DOS Header, MS-DOS Stub, PE Signature, COFF File Header, Optional Headers, and Section Headers[2], which contain metadata at different levels, showing the structure of that file and providing meaningful information about the executable binary. However, some fields are not required, such as Section Headers, or have no fixed size, such as ImageBase, SizeOfStackReserve, SizeOfStackCommit, SizeOfHeapReserve, and SizeOfHeapCommit from Optional Header Windows-Specific Fields. Each extracted header has a length of 1024 bytes. Each byte is converted to a decimal value from 0 to 255 and then normalized from 0 to 1.

[2] https://learn.microsoft.com/en-us/windows/win32/debug/pe-format.

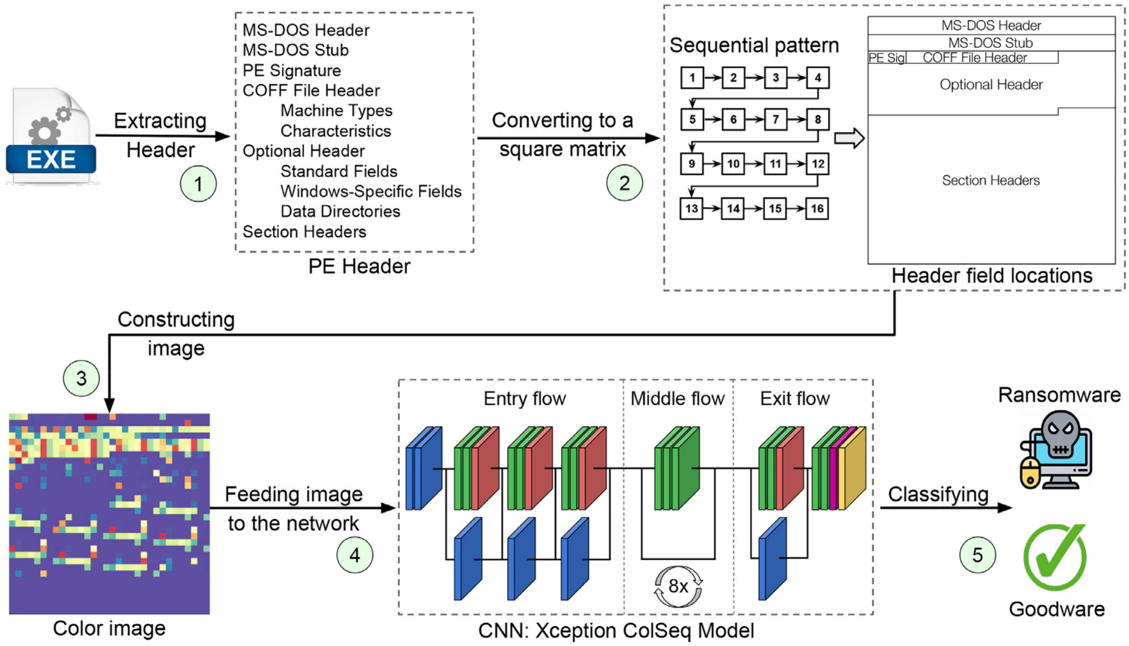


Fig. 1. High-level workflow of the proposed static analysis method.

We evaluated the proposed method using two datasets. First, we used a dataset provided directly by Manavi and Hamzeh (2022b), which contains 1000 Windows OS trusted applications downloaded from various online software repositories and 1000 applications containing ransomware randomly selected from the VirusShare[3] database. The ransomware samples include different families, such as Cerber, Locky, TorrentLocker, and TeslaCrypt. However, these families are uncategorized. The study (Manavi and Hamzeh, 2022b) provides a complete description of the dataset. This dataset is referred to as Dataset 1.

We created the second dataset by collecting 2157 binary executable samples comprising 1134 legitimate software and 1023 ransomware, grouped into 25 ransomware families. Recent reports from seven global cybersecurity vendors list a vast diversity of ransomware families still active and dangerous (see Table A.1 in Appendix A); however, the information from these reports does not always converge. Therefore, we used two criteria for selecting the most updated and relevant families for our dataset: 1) first observation occurred from 2020 *or* 2) appeared in at least two of the seven reports. Table 2 lists the family names, its assigned family ID, the first year of observation, number of times reported, number of samples collected, and percentage distribution; information in bold indicates compliance with the criterion. All ransomware samples were downloaded from the VirusShare and Hybrid-Analysis[4] databases. These files are executable binaries from the Windows OS with minimum and maximum sizes of 14.8 kB and 12.2 MB, respectively.

Ransomware creators can name their family themselves or the name can be assigned by the cybersecurity vendors' research community. Determining whether a sample is ransomware and which family it belongs to is not easy, as different antivirus vendors may categorize the sample as undetected, ransomware, or another type of malware. In addition, antivirus vendors may not specify the ransomware family or use different names for the same family, such as REvil and Revilcrypt or REvil and Sodinokibi; or worse, they may categorize them as different families.

The VirusTotal website provides a score based on the analysis rate of security vendors, and for each sample, a list of all vendors that have analyzed it is presented; for example, "Microsoft: Ransom:MSIL/Revilcrypt.PAA!MTB" or "AhnLab-V3:Ransomware/Win.Revilcrypt.C5045035." Hence, we used three criteria to categorize ransomware into families based on vendor engine detection: 1) at least 45 flagged as malicious, 2) at least 15 flagged as ransomware, *and* 3) at least ten nominations for the same family. This set of criteria was balanced to ensure a correct categorization of families and obtain sufficient samples for the tests.

To diversify non-malicious executable files, we collected samples from the PortableApps[5] and Softonic[6] databases and Windows 10 standard system, as well as by scouring the web for no-install software. All of these were scanned using ESET NOD32 Antivirus[7] version 15.1.12.0 to ensure their safety.

We then extracted PE headers from ransomware and goodware samples and converted the data into a Comma-Separated Values (CSV) file with the sample ID, filename, target class, family ID, and 1024 numerical features ranging from 0 to 255. The filenames of malicious samples is their SHA-256 hash. Fernando et al. (2020) noted that not sharing datasets in the area is a common practice, as none of the authors of the other papers in the survey have made their datasets available to the public. Therefore, we have made our dataset publicly available in CSV format at Mendeley Data[8]. This dataset is referred to as Dataset 2.

3.2. Proposed method

The main goal is to identify representative characteristics in the resultant images and differentiate ransomware from goodware. In this sense, our proposal is based on the deep CNN model, which

[3] https://www.virusshare.com.

[4] https://www.hybrid-analysis.com/.

[5] https://portableapps.com/.

[6] https://en.softonic.com/.

[7] https://www.eset.com/.

[8] https://data.mendeley.com/datasets/p3v94dft2y.

Table 1
Advantages and limitations of related studies.

Works	Advantages	Limitations
Aggarwal et al. (2022); Maleki et al. (2019); Moti et al. (2021, 2019); Noever and Noever (2021); Rezaei and Hamze (2020); Rezaei et al. (2021); Wen and Chow (2021)	Can improve solutions for malware detection systems.	Does not focus on a specific type of malware.
Poudyal et al. (2019)	Can improve solutions for malware detection systems.	Analyzes only one ransomware family.
Vidyarthi et al. (2019)	Complements contemporary anti-malware techniques.	Needs significant understanding of PE header structure.
Manavi and Hamzeh (2021, 2022a)	Needs no significant understanding of PE header structure. Requires no distinction between variable-length fields. Achieves good performance.	Did not test the detection of new ransomware families.
Manavi and Hamzeh (2022b)	Needs no significant understanding of PE header structure. Requires no distinction between variable-length fields. Achieves good performance.	Uses suboptimal options for the vector pattern and color palette. Did not test the detection of new ransomware families.
Sgandurra et al. (2016)	Creates and shares a feature dataset. Categorizes 11 ransomware families. Tests a detection of new ransomware families.	Uses an incomplete testing methodology for detecting new ransomware families.
Zahoora et al. (2022)	Tests the detection of new ransomware families. Improves the testing methodology for detecting new ransomware families.	Did not test all the families available. Did not maximize the number of families in the training set.
This study	Can improve solutions for malware detection systems. Focus on a specific type of malware. Complements contemporary anti-malware techniques. Needs no significant understanding of PE header structure. Requires no distinction between variable-length fields. Uses better options for the vector pattern and color palette. Creates and shares a updated feature dataset. Categorizes and analyzes 25 ransomware families. Refines the testing methodology for detecting new ransomware families. Tests the detection of new ransomware families in all available. Maximizes the number of families in the training set. Achieves the best performance among related works.	

has become a state-of-the-art method for solving various computer vision tasks such as object tracking, pose estimation, text recognition, visual saliency detection, human action interpretation, scene labeling, and image classification (Li et al., 2021; Malik and Shapiai, 2022). CNNs can automatically learn different feature representation levels with sufficient training data. Their deep structure can extract local and global features in images and use them in the categorization phase. Unlike traditional ML-based methods, a CNN can automatically learn feature representations from bytecode images (Ding et al., 2020). Hence, an advantage of this method is that it allows using the raw PE header. That is, neither advanced knowledge of the header structure to extract features nor a distinction between variable-length fields is required.

We use an already defined CNN model: Xception, which is 71 layers deep and has a special CNN architecture based entirely on depthwise separable convolution layers, that is, a depthwise convolution followed by a pointwise convolution. This architecture is based on the assumption that the mapping of cross-channel and spatial correlations in the feature maps of the CNN can be completely decoupled. More details of the Xception architecture can be found in Chollet (2017), which also demonstrates to be a solid DL model for ImageNet, a large benchmark image dataset used in recognition challenges for evaluating algorithms on object detection and image classification at a large scale.

The Xception model is more powerful with fewer overfitting problems than popular CNN models such as VGG16 (Lo et al., 2019), and it outperforms the technically mature models ResNet50, InceptionV3, and DenseNet121 in accuracy classification (Chen et al., 2021; Shaheed et al., 2022). Recent studies have used this model as a feature extractor (Shaik and Cherukuri, 2021; Sharma and Kumar, 2022) and image classifier for medical diagnoses (Shaheed et al., 2022; Yadavendra and Chand, 2020), as well as in computer vision tasks such as scene classification (Chen et al., 2020) and roughness detection (Chen et al., 2021).

We use the Keras API[9] to build the Xception architecture, preprocess images, and evaluate the samples. This API provides pre-built models that can be used in three methods. The first and most common method is transfer learning, which uses a pretrained model by importing its weights and training only the last few layers to adapt to the required domain. The second method uses a pretrained partial model, freezing some layers, and marking them as untrainable such that only the other layers have their weights updated during the training phase. Finally, we can use the pre-built model and train it from scratch with random initialized weights. He et al. (2019) compared the results of the approaches and determined that, although a model trained from scratch requires a longer training time, it exhibits a robust performance and may surpass the pretrained model in accuracy; therefore, the paradigm of using pretrained models should be reconsidered. Our research used a pre-built Xception model and trained it from scratch.

In our proposed method, various parameters can affect the model's performance, including the vector sequence pattern, color palette, and image size; consequently, we experimented with four distinct patterns: sequential, zigzag, spiral, and diagonal zigzag. Fig. 2 shows a sample of ransomware represented as each of the four experimental patterns. A sample has 1024 numerical features and can thus be converted into a 32 × 32 square matrix following the defined pattern. The key idea of the experiment is to analyze which pattern provides better continuity of the feature vector to enhance the model's results.

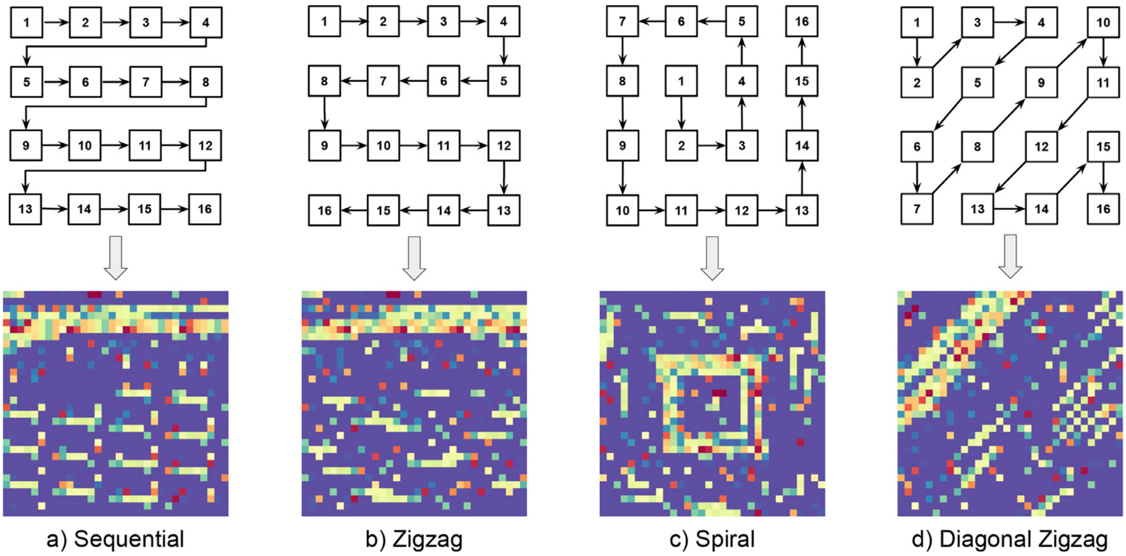
The authors of Naeem et al. (2020) examined classification in different color scales and demonstrated that color images outperformed gray-scale images for malware classification. According to Gupta et al. (2021), resizing and cropping original resolution images results in information loss before feature extraction. Therefore, because 256 values are possible for each feature, we used the continued spectral reverse color palette provided by the Seaborn API[10] to create images with a size of 256 × 256 pixels. This is the closest size to Xception's default input shape (299 × 299) that

[9] https://keras.io.

[10] https://seaborn.pydata.org/.

Table 2
25 ransomware families collected according to the defined criteria.

Family	ID	1st Observation	# Reported	# Samples	%
Babuk (Babyk)	2	**2021**	1	44	4.30
BlackMatter	3	**2021**	1	45	4.40
Avaddon	1	**2020**	1	49	4.79
Conti	4	**2020**	**4**	48	4.69
DarkSide	5	**2020**	**2**	50	4.89
Exorcist	8	**2020**	1	19	1.86
Makop (Oled)	11	**2020**	1	35	3.42
MountLocker	13	**2020**	1	14	1.37
Nefilim (Nephilim)	14	**2020**	1	39	3.81
Thanos (Prometeus)	23	**2020**	1	35	3.42
WastedLocker	24	**2020**	2	40	3.91
DoppelPaymer	7	2019	**2**	23	2.25
LockBit (ABCD)	10	2019	**2**	48	4.69
Maze (ChaCha)	12	2019	**4**	50	4.89
NetWalker (MailTo, Koko)	15	2019	**3**	50	4.89
Phobos	16	2019	**2**	50	4.89
Pysa (Mespinoza)	17	2019	**2**	38	3.71
Ragnarok (RagnarLocker)	18	2019	**2**	43	4.20
REvil (Sodinokibi)	20	2019	**6**	49	4.79
Stop (Djvu)	22	2019	**2**	48	4.69
Zeppelin (Buran, VegaLocker)	25	2019	**2**	49	4.79
GandCrab	9	2018	**2**	50	4.89
Ryuk	21	2018	**3**	48	4.69
RansomeXX (Defray777, Target 777)	19	2017	**2**	13	1.27
Dharma (CrySIS, Wadhrama)	6	2016	**2**	46	4.50
Total				1023	100



a) Sequential

b) Zigzag

c) Spiral

d) Diagonal Zigzag

Fig. 2. Representations of the same ransomware sample in the four experimental sequence patterns.

maintains an integer (64) amount of pixels for each byte represented, ensuring that all pixels of a point have the same color.

The spectral reverse color bar and six sequential pattern image samples used as inputs to our static analysis model are shown in Fig. 3: from a) to c) ransomware, and from d) to f) goodware. This color scheme has good and uniform diversity because it uses the three primary colors and smoothly transitions between blue and yellow, and then yellow and red, generating secondary and tertiary colors. Additionally, no direct transition exists between red and blue, avoiding confusion caused by variations in purple. This variety of colors provides valuable information to the resulting image.

Table 3 summarizes the parameters of our experimental Xception model. Any parameters not mentioned were set to the default values provided by the Keras API. The input shape and target size have the same pixel shape as the image sample we constructed. This avoids resizing the samples, reducing computational costs, and avoiding information loss before feature extraction. Because our classification domain is binary, we used only one class as the output with a sigmoid activation function. We also used the Adamax optimizer, a variant of Adam based on the infinity norm (Kingma and Ba, 2017), with a learning rate of 0.001. For training, we used batch sizes of 32 and 35 epochs. In addition, we used the early stopping technique to avoid overfitting, which stops training before convergence and shortens training time. We monitored the training accuracy to maximize it such that, if accuracy did not improve within a range of 0.1% for ten epochs, the training was stopped and testing commenced.

To effectively evaluate our model's performance, we compare it with two other CNN models: InceptionResNetV2, which is a

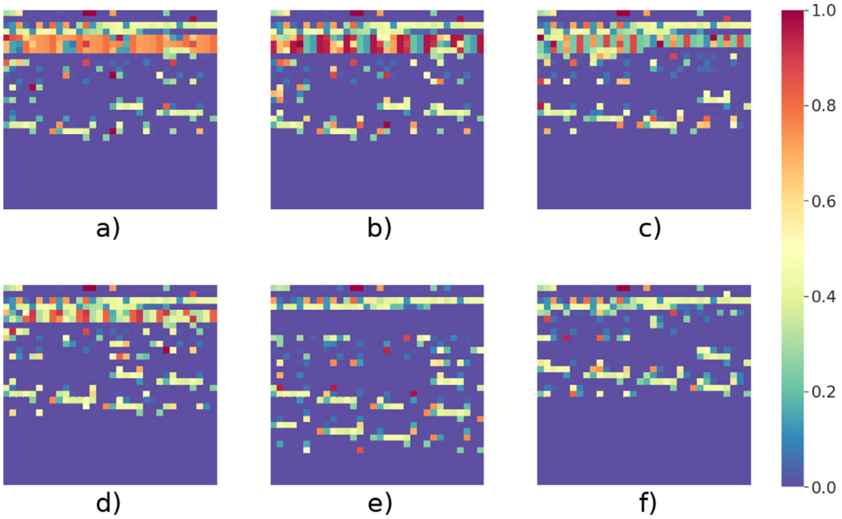


Fig. 3. Six image samples: ransomware from a) to c) and goodware from d) to f).

Table 3
Parameters used in the Xception CNN model.

Segment	Parameter	Value
Xception	Input shape	(256, 256, 3)
	Weights	None
	Classes	1
	Classifier activation	Sigmoid
	Loss	Binary crossentropy
	Optimizer	Adamax
	Learning rate	0.001
Training	Epochs	35
	Batch size	32
	Class mode	Binary
	Target size	(256, 256)
Early Stopping	Monitor	Accuracy
	Mode	Max
	Patience	10
	Minimum delta	0.001

contemporary model to Xception and an improvement of InceptionV3 (Szegedy et al., 2016), and EfficientNetV2S, a recent and evolved model that has a faster training speed, better parameter efficiency, and superior performance than previous models on ImageNet dataset (Tan and Le, 2021). Moreover, we compare it with six previously evaluated ML algorithms for ransomware detection in (Bae et al., 2020; Moreira et al., 2022): NB, K-Nearest Neighbors (KNN), Logistic Regression (LR), RF, Stochastic Gradient Descent (SGD), and Support Vector Machine (SVM). These ML techniques represent the most commonly used categories of supervised learning, including Bayesian, regression, ensemble, and instance-based learning. They are widely applied for binary classification, and each algorithm presents a different internal mechanism and data-handling method, as well as their advantages and disadvantages, aiding us in comprehensively exploring the nature of the data (Bae et al., 2020; Ray, 2019).

The performance of modern DL and ML methods is highly dependent on their hyperparameter adjustments, which aid in maximizing the outcomes without overfitting, underfitting, or creating a high variance. To compare the models more fairly, each method was tuned on hyperparameters. Table B.1 in Appendix B lists the final hyperparameters (tuned and default) used for each method. We initially followed the original studies on InceptionResNetV2 (Szegedy et al., 2016) and EfficientNetV2S (Tan and Le, 2021) to adjust their hyperparameters; however, to adapt the models for the experiment domain, we tuned the optimizer searching for better results. For the ML algorithms, we tuned the hyperparameters frequently adjusted in the literature (Paper, 2020; van Rijn and Hutter, 2018; Singh and Singh, 2022). We used the Scikit-learn GridSearchCV method with $K = 10$ and accuracy as a scoring strategy to evaluate the performance of the cross-validated model on the test set.

3.3. Evaluation metrics

For the effective evaluation of our method, we used well-known ML metrics such as accuracy, precision, recall, and F-measure in a binary classification context. A combination of these metrics provides a better interpretation of model performance. Eqs. (1) to (4) present the mathematical expressions of the evaluation metrics. These metrics can be interpreted as relations based on four values: true positive (TP): number of ransomware samples correctly classified as ransomware, true negative (TN): number of goodware samples correctly classified as goodware, false positive (FP): number of goodware samples incorrectly classified as ransomware, and false negative (FN): number of ransomware samples incorrectly classified as goodware.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F - measure = 2 \times \left(\frac{Precision \times Recall}{Precision + Recall} \right) \tag{4}$$

For every classification, we performed an average of ten iterations of stratified K-fold Cross-validation (CV) with $K = 10$. This study presents the results as 0.95 Confidence Interval (CI). When presenting comparative results, the bold numbers indicate the best results.

3.4. Interpretation method

As mentioned, CNNs excel in various computer vision tasks. However, the black-box nature of deep networks makes interpreting how they draw conclusions difficult. Several of the techniques proposed aim to attribute importance values to input features to

Table 4
Results of the four experimental sequence patterns. Metric (%) ± 0.95 CI (%).

Metric	Sequential	Zigzag	Spiral	Diagonal Zigzag
Accuracy	**93.73 ±0.23**	92.96 ± 0.23	92.42 ± 0.23	93.08 ± 0.22
Precision	**92.95 ±0.33**	92.52 ± 0.33	92.07 ± 0.33	92.57 ± 0.34
Recall	**94.64 ±0.34**	93.46 ± 0.35	92.81 ± 0.34	93.67 ± 0.34
F-Measure	**93.75 ±0.23**	92.98 ± 0.22	92.44 ± 0.22	93.12 ± 0.22

then highlight the models' decision-making process. A type of interpretation method widely used is based on the Class Activation Map (CAM), which is typically generated from the final convolution layer of a CNN model to highlight the crucial input features on which the deep model primarily relies (Li et al., 2022; Rao et al., 2022).

We used a recent interpretation method called LayerCAM to perform the Xception decision-making to show the distinguishability of the saliency features of the PE header in the detection of new families. It produced images of the discriminative regions used for classifying ransomware and goodware based on weights connecting features to the classes. LayerCAM can generate trustworthy CAMs from the final convolution layer and from shallow layers, where it can obtain coarse spatial locations and fine-grained object details. In addition, it is simple to apply to pre-built CNN-based image classifiers, making it more versatile and practical (Jiang et al., 2021).

The LayerCAM technique was applied with the tf-keras-vis[11] python package, a visualization toolkit for debugging Keras' models. The main parameters and functions used were penultimate_layer = -1, model_modifier = ReplaceToLinear(), and score = BinaryScore(1.0 or 0.0) for predicting the ransomware or goodware classes, respectively.

4. Results and discussion

This section presents the comparative results between this study and related work based on the evaluation metrics defined in Section 3.1. We present the results of the four pattern sequences previously explained and then compare our best results with related work and various DL and ML algorithms using Datasets 1 and 2. Furthermore, this section shows the performance of our approach for detecting new ransomware families. We used a Google Colab environment equipped with a two-core Intel(R) Xeon(R) CPU @2.20GHz, eight-core Tensor Processing Units (TPU), 32 GB of RAM, and 225 GB HDD.

4.1. Model evaluation using dataset 1

The setup presented in Table 3 was applied to Dataset 1 with the four sequence patterns we experimented with (see Fig. 2), and Table 4 shows the results.

Although the other sequence patterns provided fair results, the sequential pattern obtained the best performance for all evaluation metrics. For the best result, the minimum and maximum number of training epochs was 19 and 32, respectively, and the average number of epochs for the entire training process was 22.65. With this setup, our proposed method (Xception ColSeq) achieved an accuracy of 93.73%, precision of 92.95%, recall of 94.64%, and F-measure of 93.75%.

Table 5 lists the comparative results of the Xception ColSeq method to related works, two CNN models, and six ML algorithms using Dataset 1. Xception ColSeq surpassed all models in accuracy, recall, and F-measure, but the related works had precisions above ours. The highlight was our recall performance of 94.64%, which emphasizes the model's ability to predict ransomware correctly. This metric indicates a low false-negative rate, which means that ransomware is likely not classified as goodware, which is the worst-case scenario. InceptionResNetV2 performed similarly to our work. EfficientNetV2S, RF, and SVM achieved results of ≈92%. KNN, LR, and SGD performed with results between ≈88% and ≈91%. Evidently, NB had the worst result: between ≈63% and ≈86%.

4.2. Model evaluation using dataset 2

To demonstrate the efficiency of our method, we apply it to Dataset 2 with the same parameters, hyperparameters, and early stopping technique. For comparative effects, we apply gray-scale images in a zigzag pattern with 256 × 256 pixels to Xception (Xception GrayZZ[12]). We also reproduce the model of Manavi and Hamzeh (2022b) using gray-scale images in zigzag patterns (Baseline Manavi and Hamzeh, 2022b GrayZZ) and spectral reverse color images in a sequential pattern (Baseline Manavi and Hamzeh, 2022b ColSeq) with 32 × 32 pixels. In addition, we compare the results with the CNN and ML models previously compared. Until now, we have been unconcerned with the training time because the experiment environments were different, although it is a relevant variable depending on the application domain of a model. Table 6 presents the comparative results of the specified models and average training time for each fold using Dataset 2. Xception ColSeq achieved an accuracy of 98.20%, precision of 97.50%, recall of 98.76%, and F-measure of 98.12% with an average training time of 123.38 s. For this result, the minimum and maximum number of training epochs was 14 and 20, respectively, and the average number of epochs for the entire training process was 16.26.

Xception ColSeq outperformed the other models in accuracy, precision, recall, and F-measure. Our approach using spectral reverse color images in sequential patterns outperforms the gray-scale images in zigzag patterns by ≈1% in all metrics, whether comparing Xception ColSeq with Xception GrayZZ or Baseline (Manavi and Hamzeh, 2022b) ColSeq with Baseline (Manavi and Hamzeh, 2022b) GrayZZ. In addition, the training time of Xception ColSeq was shorter than that of the Xception GrayZZ model owing to fewer epochs to reach the early stop criterion. The baselines (Manavi and Hamzeh, 2022b) ColSeq and Manavi and Hamzeh (2022b) GrayZZ methods achieved general performance of ≈97% and ≈96%, respectively, with similar training times, as they used a fixed number of epochs. We verified that InceptionResNetV2 had similar results to ours, although its training time was approximately three times greater than ours. EfficientNetV2S achieved a general performance of ≈97%, but its training time was the longest. The KNN, LR, RF, SGD, and SVM algorithms had a performance between ≈94% and ≈97% and training times between 2.22 and 9 s. NB spent the least time training but had the worst performance. Note that the better overall performance obtained by the models using Datasets 2 compared to Dataset 1, which may be because of the adequate criteria for obtaining correct ransomware samples and greater variety of benign software types that better generalize the models to differentiate goodware from ransomware.

In a training cost/benefit ratio, the typical ML algorithms could be the best models because their training times are considerably lower than that of the CNN models. However, performance metrics are more important in cybersecurity, as an undetected attack can cause severe damage; hence, the model that achieves the highest evaluation metrics is the best. Furthermore, after training our model, the prediction of a submitted 32-size batch requires

[11] https://keisen.github.io/tf-keras-vis-docs/.

[12] The Xception model only accepts three channels as the input shape.

Table 5
Comparing our results with those of related work, two CNN models, and six ML algorithms using Dataset 1: Metric (%) ± 0.95 CI (%).

Model	Accuracy	Precision	Recall	F-measure
Xception ColSeq	**93.73** ± 0.23	92.95 ± 0.33	**94.64** ± 0.34	**93.75** ± 0.23
Manavi and Hamzeh (2021)	93.25	93.33	93.25	93.24
Manavi and Hamzeh (2022a)	93.30	**93.48**	93.30	93.28
Manavi and Hamzeh (2022b)	93.33	93.40	93.33	93.34
InceptionResNetV2	93.60 ± 0.25	92.70 ± 0.36	94.63 ± 0.37	93.62 ± 0.25
EfficientNetV2S	92.11 ± 0.47	91.38 ± 0.49	92.99 ± 0.90	92.12 ± 0.52
NB	68.79 ± 0.45	63.74 ± 0.36	86.72 ± 0.47	73.43 ± 0.32
KNN	89.64 ± 0.28	89.34 ± 0.40	89.99 ± 0.41	89.62 ± 0.28
LR	89.89 ± 0.28	89.33 ± 0.37	90.57 ± 0.40	89.90 ± 0.28
RF	92.49 ± 0.25	92.58 ± 0.36	92.37 ± 0.33	92.44 ± 0.25
SGD	88.36 ± 0.33	88.98 ± 0.55	87.81 ± 0.78	88.18 ± 0.37
SVM	91.69 ± 0.25	90.96 ± 0.33	92.56 ± 0.37	91.72 ± 0.25

Table 6
Comparing our results with those of our baseline-related work, two CNN models, and six ML algorithms using Dataset 2: Metric (%) ± 0.95 CI (%).

Model	Accuracy	Precision	Recall	F-measure	Avg. training time (s)
Xception ColSeq	**98.20** ± 0.11	**97.50** ± 0.19	**98.76** ± 0.12	**98.12** ± 0.11	123.38
Xception GrayZZ	97.14 ± 0.14	96.34 ± 0.22	97.71 ± 0.17	97.01 ± 0.14	134.82
Baseline Manavi and Hamzeh (2022b) ColSeq	96.97 ± 0.18	96.14 ± 0.29	97.67 ± 0.24	96.89 ± 0.18	106.42
Baseline Manavi and Hamzeh (2022b) GrayZZ	95.93 ± 0.17	95.09 ± 0.29	96.46 ± 0.26	95.74 ± 0.18	106.73
InceptionResNetV2	98.01 ± 0.10	97.12 ± 0.22	**98.76** ± 0.14	97.92 ± 0.11	370.11
EfficientNetV2S	97.59 ± 0.15	97.05 ± 0.21	97.92 ± 0.22	97.47 ± 0.16	410.50
NB	77.36 ± 0.39	81.21 ± 0.49	68.21 ± 0.77	73.98 ± 0.52	**0.67**
KNN	95.67 ± 0.16	95.14 ± 0.26	95.83 ± 0.26	95.46 ± 0.17	4.73
LR	94.74 ± 0.20	94.49 ± 0.29	94.47 ± 0.31	94.45 ± 0.21	2.22
RF	96.51 ± 0.18	95.62 ± 0.27	97.13 ± 0.22	96.35 ± 0.16	9.00
SGD	94.31 ± 0.21	93.96 ± 0.35	94.14 ± 0.31	94.01 ± 0.21	6.60
SVM	96.48 ± 0.15	95.92 ± 0.23	96.73 ± 0.21	96.31 ± 0.16	7.12

≈0.72 s, that is, only 0.0225 s per sample. Therefore, our results demonstrate that the proposed model is viable as a ransomware detection technique.

4.2.1. Detection of new families

A particular case of zero-day ransomware attack detection is the identification of new ransomware families. This ability is crucial for any protection system. Sgandurra et al. (2016) evaluated the detection of each family of their dataset by separating all samples of one family to test while using all remaining ransomware and goodware to train the model. However, this testing methodology reduces the range of applicable evaluation metrics because a test set where all samples are positive implies the non-occurrence of FPs and TNs during the test, transforming accuracy into recall (Eq. (5)); furthermore, it implies a precision of 100% if at least one sample is inferred as ransomware, i.e., a TP (Eq. (6)). In addition, a hypothetical biased model that infers ransomware for any case implies inferring only TPs and no FNs. Therefore, this hypothetical model achieves 100% in accuracy/recall (Eq. (7)) and, consequently, an F-measure of 100% (Eq. (8)) without representing the model's correct performance.

Test set with only positive samples $\Rightarrow$ $FP = 0$ and $TN = 0$;

$$\Rightarrow \quad Accuracy = Recall = \frac{TP}{TP + FN} \tag{5}$$

$$\Rightarrow \quad Precision = \frac{TP}{TP} = 1 \tag{6}$$

Hypothetical biased model that infers only ransomware $\Rightarrow$ $FN = 0$;

$$\Rightarrow \quad Accuracy = Recall = \frac{TP}{TP} = 1 \tag{7}$$

$$\Rightarrow \quad F - measure = 2 \times \left(\frac{Precision \times Recall}{Precision + Recall} \right) = 1 \tag{8}$$

We refined this testing methodology of new ransomware family detection by adding goodware samples selected randomly to the test set with the same amount of samples present in the family evaluated, balancing the test set, providing representative metrics of the model's performance, and preventing biased models from achieving inaccurate results. Some families have less than 15 samples; consequently, we complement the test set with up to 30 samples to continue using the Z-score for calculating the CI (Aityan, 2022). The evaluation process is repeated ten times for each family to obtain the mean results, which corresponds to our model's complete performance in detecting new ransomware families. This analysis is only possible with Dataset 2, as each ransomware sample was labeled as one of the 25 families.

Table 7 presents our results in accuracy, precision, and F-measure metrics. Our weighted accuracy average shows that our model can correctly classify samples in 93.84% of cases, in which 17 out of 25 families achieved above 95% of correct classification. However, Makop, MountLocker, and Thanos were the three worst results with accuracies less than 75%. The weighted precision average of 97.50% indicates a low number of goodware incorrectly classified as ransomware. All families achieved over 92% in precision. The F-measure can be interpreted as the general performance of the model, in which we performed a weighted average of 92.27%, and 20 out of the 25 families achieved above 90%, demonstrating the good efficiency of our method. Nonetheless, Makop, MountLocker, and Thanos had poor general performance achieving 63.71%, 59.43%, and 37.82%, respectively.

We present recall separately in Fig. 4 because this metric is the most crucial for this type of evaluation. Our results show that five out of the 25 ransomware families were 100% correctly classified, and 18 out of the 25 had results above 90%. The PE headers of these families exhibit characteristics similar to the other families; thus, during training, the model can learn from features that highlight its differences with goodware. While Ragnarok and Nefilim families achieved a recall of ≈86%, Zeppelin and Ryuk achieved

Table 7
Metric (%) ± 0.95 CI (%) of new ransomware families detection.

Family	# Samples (Rans.-Good.)	Accuracy	Precision	F-measure
Avaddon	49-49	98.98 ± 0.26	98.06 ± 0.48	99.01 ± 0.25
Babuk	44-44	99.55 ± 0.16	99.12 ± 0.30	99.55 ± 0.15
Blackmatter	45-45	99.00 ± 0.16	98.06 ± 0.31	99.02 ± 0.16
Conti	48-48	95.73 ± 0.24	98.28 ± 0.32	95.60 ± 0.25
Darkside	50-50	98.50 ± 0.13	98.23 ± 0.27	98.51 ± 0.13
Dharma	46-46	97.61 ± 0.31	97.26 ± 0.54	97.64 ± 0.30
Doppelpaymer	23-23	96.52 ± 0.98	98.78 ± 0.75	96.31 ± 1.10
Exorcist	19-19	98.42 ± 0.77	98.94 ± 0.67	98.39 ± 0.79
Gandcrab	50-50	97.40 ± 0.22	97.80 ± 0.32	97.39 ± 0.22
Lockbit	48-48	99.06 ± 0.15	98.18 ± 0.28	99.08 ± 0.14
Makop	35-35	72.86 ± 1.08	94.74 ± 1.33	63.72 ± 1.75
Maze	50-50	96.10 ± 0.25	96.64 ± 0.40	96.07 ± 0.26
Mountlocker	14-16	73.67 ± 2.64	98.57 ± 1.23	59.43 ± 5.57
Nefilim	39-39	91.67 ± 0.94	98.76 ± 0.45	90.83 ± 1.09
Netwalker	50-50	98.60 ± 0.20	97.86 ± 0.32	98.62 ± 0.20
Phobos	50-50	97.70 ± 0.22	97.27 ± 0.38	97.72 ± 0.21
Pysa	38-38	98.82 ± 0.16	97.71 ± 0.30	98.83 ± 0.16
Ragnarok	43-43	92.79 ± 0.67	97.24 ± 0.45	92.33 ± 0.74
RansomeXX	13-17	97.33 ± 0.89	98.46 ± 1.10	96.86 ± 1.04
Revil	49-49	96.84 ± 0.32	98.16 ± 0.37	96.78 ± 0.34
Ryuk	48-48	88.02 ± 0.28	97.95 ± 0.38	86.62 ± 0.34
Stop	48-48	93.65 ± 0.27	97.09 ± 0.28	93.40 ± 0.29
Thanos	35-35	61.86 ± 1.77	92.58 ± 2.14	37.82 ± 4.23
Wastedlocker	40-40	97.13 ± 0.51	96.87 ± 0.60	97.11 ± 0.52
Zeppelin	49-49	87.65 ± 1.04	96.05 ± 0.54	85.94 ± 1.48
Weighted Avg.		93.84 ± 0.47	97.50 ± 0.51	92.27 ± 0.66

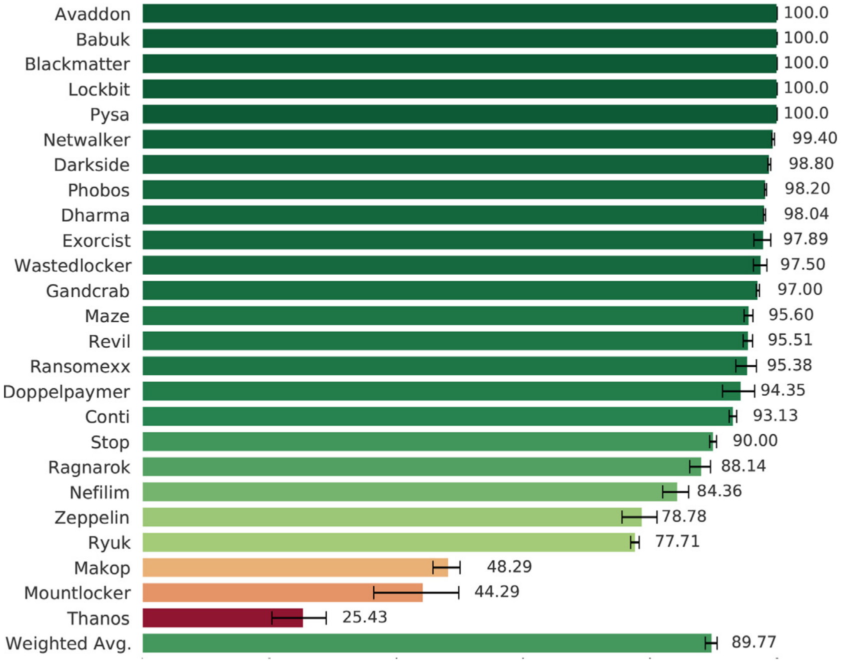


Fig. 4. Recall (%) ± 0.95 CI (%) of new ransomware families detection.

≈78%. This result shows that some PE headers from these families have properties similar to goodware and uncommon to ransomware. Our results for the Makop, Mountlocker, and Thanos families unveil that most characteristics in their PE header are indistinguishable from goodware in this zero-day attack detection test; consequently, their recalls are below 50%. Finally, we achieved a weighted recall average of 89.77%, indicating a low FN rate. Notably, the newer Babuk and BlackMatter (2021) families had an 100% detection performance. In contrast, the three worst families are from 2020.

To locate the model's most valuable regions of the PE header images, we applied the LayerCAM method using the same testing methodology, but only for one run of the best five and worst three families according to the recall metric. Each image contains the original input overlaid by a relevant areas heatmap, providing interpretability support for understanding the decision of the Xception ColSeq model. As colors transition from blue to yellow to red, a redder color indicates that more weight is assigned to that header field. Note that the model makes decisions based on the data contained in the decision regions and not on the location of

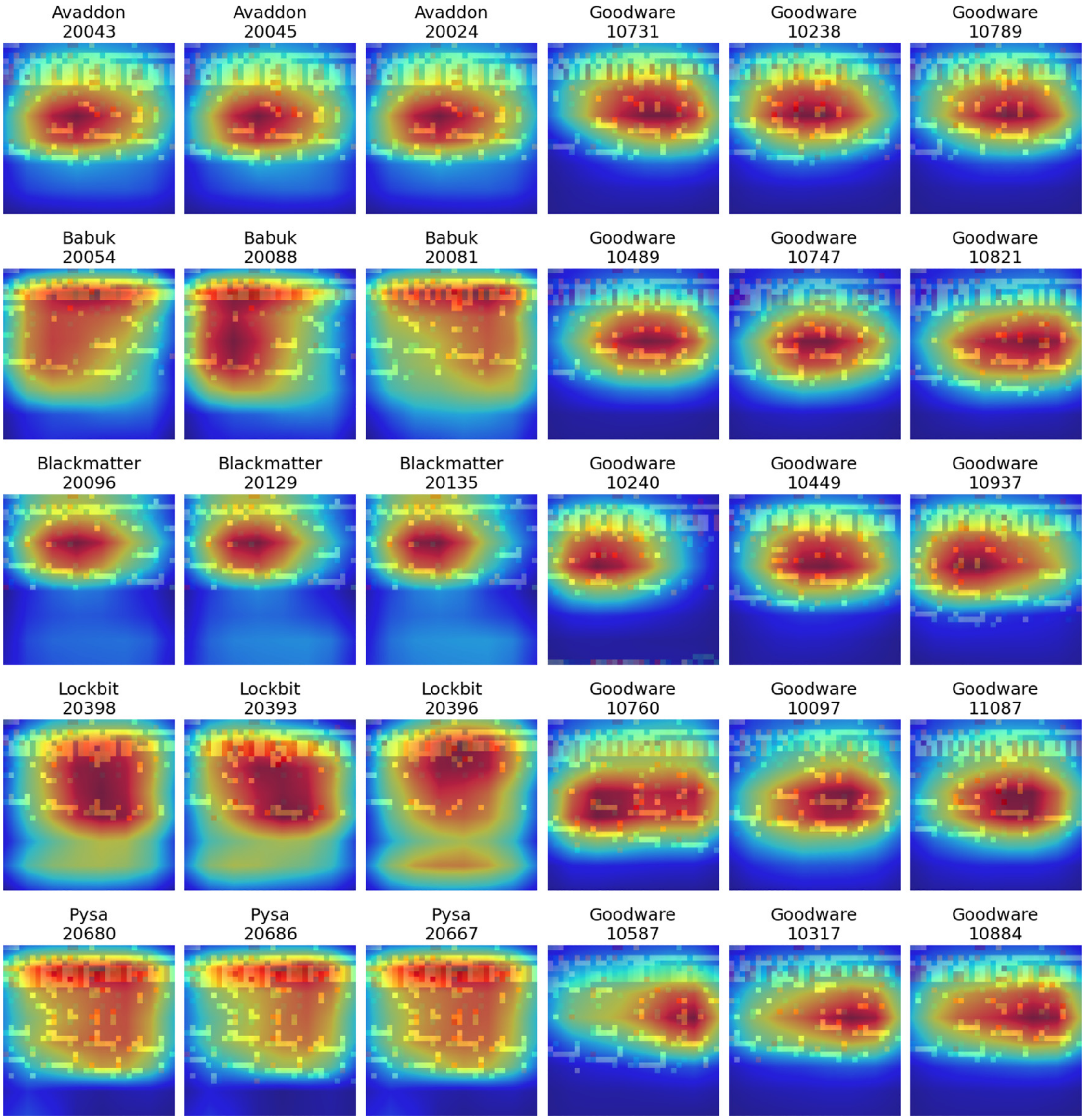


Fig. 5. Discriminative regions used to classify the best five families and goodware in terms of recall.

these zones. Therefore, the values in the decision regions make the model distinguish between classes.

Fig. 5 illustrates the discriminating detection regions of the best five families, with three ransomware and three goodware samples randomly chosen from the test set (each row) and their respective IDs. Common highlighted areas of these families range from the Optional Header, such as the Windows-Specific and Data Directories fields, to the third Section Header. These regions significantly contributed to the CNN's decision-making. Note that while the relevance zones of Avaddon and Blackmatter are concentrated, the remaining families are sparse. Goodware decision regions are more uniform and have a common red area from the middle of the Optional Header to the seventh Section Header.

To understand the results of the worst-performing families, we randomly selected and analyzed three samples predicted as TP, FP, FN, and TN for each family. Fig. 6 shows the results of the Makop family and goodware. Ransomware samples predicted as TP exhibit common, well-defined, and sparse decision areas. The only goodware sample predicted as FP shows a diffuse decision region in yellow with a more concentrated red zone on the left side of the image. The ransomware samples incorrectly classified as goodware (FN) show uniform decision zones, similar to the goodware predicted correctly (TN).

Fig. 7 presents discriminative regions used to classify the Mountlocker family and goodware. The samples predicted as TP exhibit widespread decision regions in the initial areas of the header. Sample 20519 indicates an additional lighter decision zone at the bottom of the image, i.e., the last Section Headers. Samples 20510 and 20518, predicted as FN, exhibit local areas in the center part of the image, whereas sample 20513 exhibits two concentration areas on the sides. The TN samples exhibit similar decision areas in the middle of the PE header. No FP sample was predicted.

Fig. 8 illustrates the discriminating detection regions for the Thanos family and goodware. Note that the six Thanos samples present data in all Section Headers, i.e., from the middle to the bottom of the image. This characteristic is strongly present in Thanos, unlike with the other families studied. The results imply that Xception Colseq did not learn these traits since the decision zones

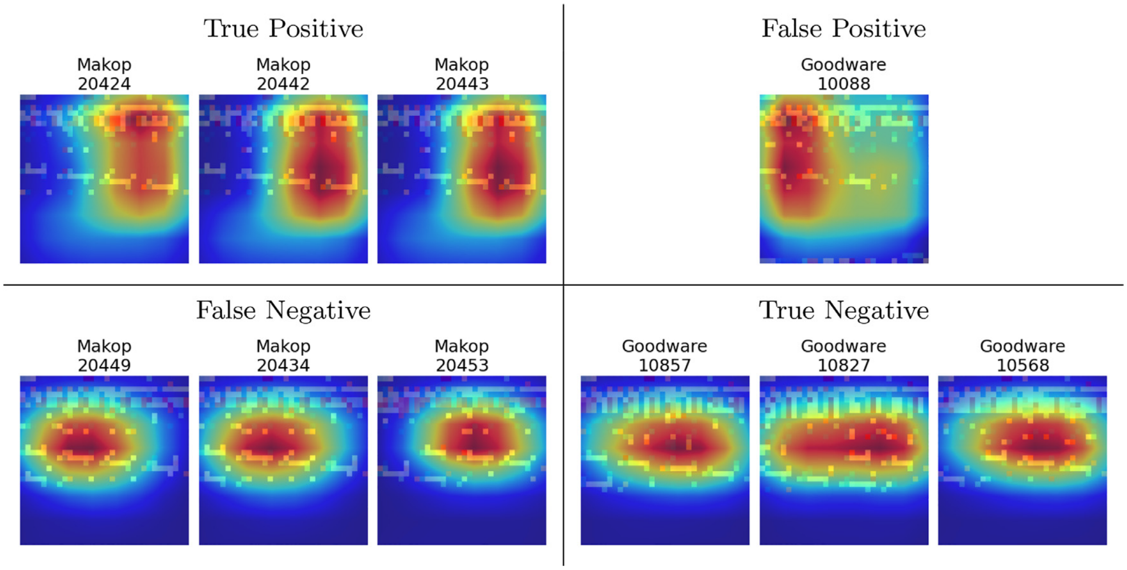


Fig. 6. Discriminative regions used to classify the Makop family and goodware. Samples were predicted as True Positive, False Positive, False Negative, and True Negative.

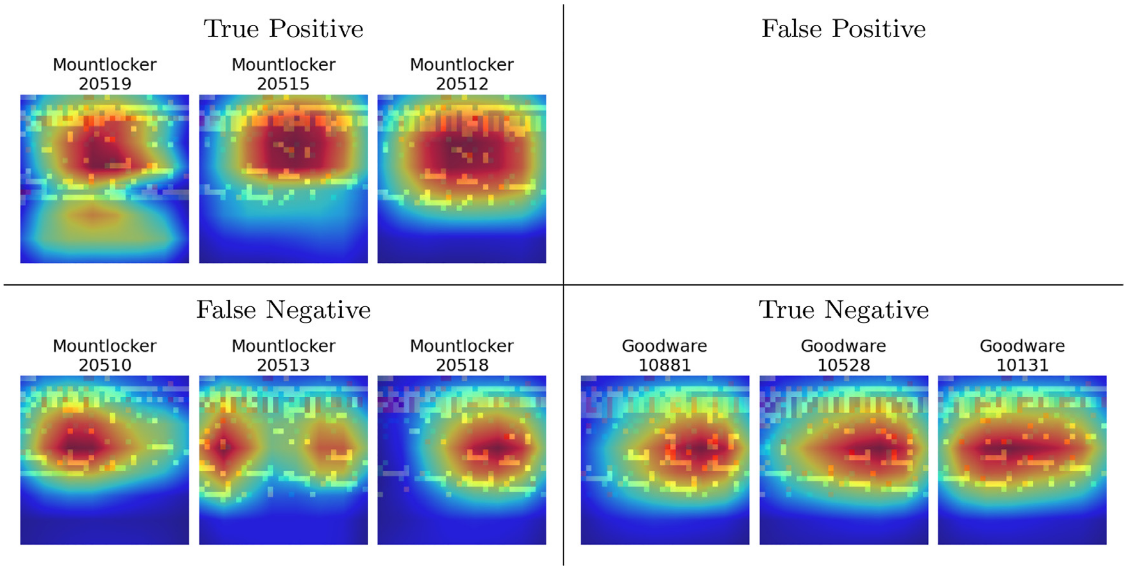


Fig. 7. Discriminative regions used to classify the Mountlocker family and goodware. Samples were predicted as True Positive, False Positive, False Negative, and True Negative.

show that the model disregarded them while making decisions. The samples predicted as TP exhibit a broad yellow area in the top half of the header, with a concentrated red zone in the Optional Header. The only goodware sample incorrectly predicted as ransomware (FP) exhibits a concentrated decision area in the Optional Header on the right side of the image. The correctly classified goodware samples (TN) exhibit similar decision regions to those of incorrectly predicted ransomware (FN).

A mutual observation across the correctly classified samples from the eight families is the decision regions containing the Optional Header and first Section Headers. The decision zones within each ransomware family are similar but still varied. Correctly predicted goodware samples exhibit decision areas that are more uniform, concentrated, and with few variations between samples. These common regions consider the Optional Header and the first Section Headers. Therefore, these two parts of the PE header are the most significant for any sample classification.

5. Conclusions

Cyber-attack techniques and anti-malware software are in permanent competition. The cyber threat of ransomware is rapidly growing and poses a severe problem that may prevent or limit users from accessing their system, data, or both until the ransom is paid. The continuous advancement of inventive and effective security techniques is essential to counteract these threats. Solutions based on DL algorithms and CNN models have demonstrated excellent performances in accurately detecting ransomware. Therefore, the notable contribution of this study is the improvement of a static analysis method for ransomware detection based on information extracted from PE headers files, which is converted into color images in a sequential vector pattern and classified using the Xception CNN without transfer learning.

We evaluated the effectiveness of our proposed Xception-based ransomware detection method on two feature datasets and ana-

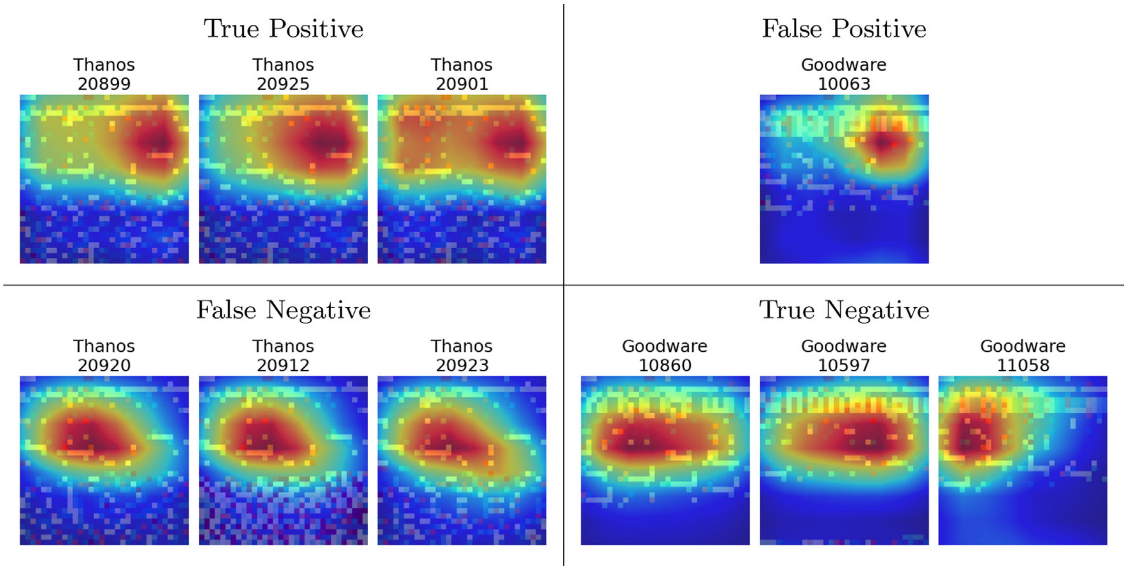


Fig. 8. Discriminative regions used to classify the Thanos family and goodware. Samples were predicted as True Positive, False Positive, False Negative, and True Negative.

lyzed its superiority over competing models. We created one of these datasets according to the criteria for categorizing 25 relevant ransomware families and have made it available at Mendeley Data.

The experiments demonstrated that our approach successfully discriminates ransomware from goodware, outperforming other CNN models, typical ML algorithms, and related works with training and testing times viable for practical implementation. In addition, this approach requires neither disassembly nor code execution for classification and is more resilient against anti-ransomware evasion techniques and ransomware evolution over time. Furthermore, it can be applied alone or integrated into a model based on ensemble learning that considers other features. Consequently, our proposed method is advantageous and may help developers of ransomware detection systems build more robust and dependable solutions.

We also improved a particular testing methodology for zero-day attacks detection—the detection of new ransomware families—by adding an adequate number of goodware to the test set, balancing the test set and providing representative metrics of the model's performance that improves the reliability of model testing. In this test, our model demonstrated an efficiency in correctly identifying new ransomware: achieving above 90% in 18 of the 25 families and a recall of 100% in five of the analyzed families. We then applied a visual interpretation method to the best five and worst three families and determined significant PE header detection regions for model decision-making. This investigation demonstrated that the most decisive fields are the Optional Header and the first Section Headers.

For future work, researchers can improve ransomware detection by applying our proposed method to other parts of the PE file or to features extracted during its execution, particularly if meaning is maintained in the information flow. Furthermore, different detection techniques can explore our dataset and investigate ransomware families' characteristics and similarities.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Caio C. Moreira: Conceptualization, Methodology, Software, Investigation, Writing – original draft, Visualization. **Davi C. Moreira:** Validation, Writing – review & editing. **Claudomiro de S. de Sales Jr.:** Validation, Writing – review & editing, Supervision.

Data availability

Mendeley Data: https://data.mendeley.com/datasets/p3v94dft2y.

Acknowledgments

The authors thank the support provided by their host institution and the Brazilian funding agency CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior).

Appendix A. Cybersecurty Reports

Table A1
Seven ransomware recent reports from global cybersecurity vendors.

Vendor	Report Title	Date
Cyber Security Networks	Ransomware: Through the Lens of Threat and Vulnerability Management	Fev-2022
EmsiSoft	Ransomware statictis for 2021: Year in summary	Jan-2022
Sophos	Sophos 2022 Threat Report: Interrelated threats target an interdependent world	Nov-2021
McAfee	Advanced Threat Research Report	Oct-2021
VirusTotal	Ransomware in a Global Context	Oct-2021
Palo Alto Networks	2021 Unit 42 Ransomware Threat Report: Understand trends and tactics to bolster defenses	Apr-2021
Group-IB	Ransomware Uncovered 2020-2021	Mar-2021

Appendix B. Hyperparameters

Table B1
Hyperparameters used for each classifier. Tuned in **‘bold’** and default in ‘regular’ text.

Model/Classifier	Hyperparameters and values
InceptionResNetV2	**optimizer=Adamax, learning_rate=0.001**, beta_1=0.9, beta_2=0.999, epsilon=1e-07
EfficientNetV2S	**optimizer=RMSprop, learning_rate=0.005**, rho=0.9, momentum=0.0, epsilon=1e-07, centered=False
NB	**alpha=0.3, fit_prior=True**, binarize=0.0, class_prior=None
KNN	**metric=’manhattan’, n_neighbors=6, weights=’distance’**, algorithm=’auto’, leaf_size=30, p=2, metric_params=None, n_jobs=None
LR	**C=0.9, max_iter=50, penalty=’l2’, solver=’lbfgs’**, dual=False, tol=0.0001, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, multi_class=’auto’, verbose=0, warm_start=False, n_jobs=None, l1_ratio=None
RF	**criterion=’gini’, max_depth=300, n_estimators=100**, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=’auto’, max_leaf_nodes=None, min_impurity_decrease=0.0, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False, class_weight=None, ccp_alpha=0.0, max_samples=None
SGD	**alpha=0.001, loss=’hinge’, penalty=’elasticnet’, tol=0.0001**, l1_ratio=0.15, fit_intercept=True, max_iter=1000, shuffle=True, verbose=0, epsilon=0.1, n_jobs=None, random_state=None, learning_rate=’optimal’, eta0=0.0, power_t=0.5, early_stopping=False, validation_fraction=0.1, n_iter_no_change=5, class_weight=None, warm_start=False, average=False
SVM	**C=0.1, gamma=0.1, kernel=’poly’**, degree=3, coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size=200, class_weight=None, verbose=False, max_iter=- 1, decision_function_shape=’ovr’, break_ties=False, random_state=None

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.cose.2023.103265

References

Aggarwal, N., Aggarwal, P., Gupta, R., 2022. Static malware analysis using PE header files API. In: 2022 6th International Conference on Computing Methodologies and Communication (ICCMC), pp. 159–162. doi:10.1109/ICCMC53470.2022.9753899.

Ahmad, T., 2020. Corona virus (covid-19) pandemic and work from home: challenges of cybercrimes and cybersecurity. SSRN Electron. J. 4. doi:10.2139/ssrn.3568830.

Aityan, S.K., 2022. Confidence Intervals. Springer International Publishing, Cham, pp. 233–277. doi:10.1007/978-3-030-76857-7_13. Ch. 13

Bae, S.I., Lee, G.B., Im, E.G., 2020. Ransomware detection using machine learning algorithms. Concurr. Comput. Pract. Exp. 32 (18). doi:10.1002/cpe.5422. E5422

Beaman, C., Barkworth, A., Akande, T.D., Hakak, S., Khan, M.K., 2021. Ransomware: recent advances, analysis, challenges and future research directions. Comput. Secur. 111, 102490. doi:10.1016/j.cose.2021.102490.

Chen, H., Yang, Y., Zhang, S., 2020. Learning robust scene classification model with data augmentation based on xception. J. Phys. Conf. Ser. 1575 (1), 012009. doi:10.1088/1742-6596/1575/1/012009.

Chen, Y., Yi, H., Liao, C., Huang, P., Chen, Q., 2021. Visual measurement of milling surface roughness based on xception model with convolutional neural network. Measurement 186, 110217. doi:10.1016/j.measurement.2021.110217.

Chollet, F., 2017. Xception: deep learning with depthwise separable convolutions. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1800–1807. doi:10.1109/CVPR.2017.195.

Ding, Y., Zhang, X., Hu, J., Xu, W., 2020. Android malware detection method based on bytecode image. J. Ambient Intell. Humaniz. Comput. 2020, 1–10. doi:10.1007/S12652-020-02196-4.

El-Shafai, W., Almomani, I., AlKhayer, A., 2021. Visualized malware multi-classification framework using fine-tuned cnn-based transfer learning models. Appl. Sci. 11 (14), 6446. doi:10.3390/app11146446.

Fernando, D.W., Komninos, N., Chen, T., 2020. A study on the evolution of ransomware detection using machine learning and deep learning techniques. IoT 1 (2), 551–604. doi:10.3390/iot1020030.

Ferrante, A., Malek, M., Martinelli, F., Mercaldo, F., Milosevic, J., 2018. Extinguishing ransomware - a hybrid approach to android ransomware detection. Springer International Publishing, Cham, pp. 242–258. doi:10.1007/978-3-319-75650-9_16.

Gupta, S., Dileep, A.D., Thenkanidiyoor, V., 2021. Recognition of varying size scene images using semantic analysis of deep activation maps. Mach. Vis. Appl. 32 (2), 52. doi:10.1007/s00138-021-01168-8.

Hampton, N., Baig, Z., Zeadally, S., 2018. Ransomware behavioural analysis on windows platforms. J. Inf. Secur. Appl. 40, 44–51. doi:10.1016/j.jisa.2018.02.008.

Hassan, N.A., 2019. Ransomware Families, Apress. Berkeley, CA doi:10.1007/978-1-4842-4255-1_3.

He, K., Girshick, R., Dollar, P., 2019. Rethinking Imagenet Pre-training, in: Proceedings of the IEEE international conference on computer vision, vol. 2019-October. Institute of Electrical and Electronics Engineers Inc., Seoul, Korea, pp. 4917–4926. doi:10.1109/ICCV.2019.00502.

Hemalatha, J., Roseline, S.A., Geetha, S., Kadry, S., Damaševičius, R., 2021. An efficient densenet-based deep learning model for malware detection. Entropy 23 (3), 344. doi:10.3390/e23030344.

Hull, G., John, H., Arief, B., 2019. Ransomware deployment methods and analysis: views from a predictive model and human responses. Crime Sci. 8, 2. doi:10.1186/s40163-019-0097-9.

Jiang, P.T., Zhang, C.B., Hou, Q., Cheng, M.M., Wei, Y., 2021. Layercam: exploring hierarchical class activation maps for localization. IEEE Trans. Image Process. 30, 5875–5888. doi:10.1109/TIP.2021.3089943.

Kapoor, A., Gupta, A., Gupta, R., Tanwar, S., Sharma, G., Davidson, I.E., 2022. Ransomware detection, avoidance, and mitigation scheme: a review and future directions. Sustainability 14 (1), 8. doi:10.3390/su14010008.

Khan, N.A., Brohi, S.N., Zaman, N., 2020. Ten deadly cyber security threats amid covid-19 pandemic 5. doi:10.36227/TechRxiv.12278792.V1.

Kok, S., Abdullah, A., Jhanjhi, N., 2022. Early detection of crypto-ransomware using pre-encryption detection algorithm. J. King Saud Univ. Comput. Inf. Sci. 34 (5), 1984–1999. doi:10.1016/j.jksuci.2020.06.012.

Kok, S.H., Abdullah, A., Jhanjhi, N.Z., Supramaniam, M., 2019. Ransomware, threat and detection techniques: areview. Int. J. Comput. Sci. Netw. Secur. 19 (2), 136–146. http://paper.ijcsns.org/07_book/201902/20190217.pdf

Kolodenker, E., Koch, W., Stringhini, G., Egele, M., 2017. Paybreak: defense against cryptographic ransomware. In: Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, ASIA CCS ’17, Association for Computing Machinery. New York, NY, USA, pp. 599–611. doi:10.1145/3052973.3053035.

Li, X., Xiong, H., Li, X., Wu, X., Zhang, X., Liu, J., Bian, J., Dou, D., 2022. Interpretable deep learning: interpretation, interpretability, trustworthiness, and beyond. Knowl. Inf. Syst. 64 (12), 3197–3234. doi:10.1007/s10115-022-01756-8.

Li, Z., Liu, F., Yang, W., Peng, S., Zhou, J., 2021. A survey of convolutional neural networks: analysis, applications, and prospects. IEEE Trans Neural Netw Learn Syst 1–21. doi:10.1109/TNNLS.2021.3084827.

Lo, W.W., Yang, X., Wang, Y., 2019. An xception convolutional neural network for malware classification with transfer learning. In: 2019 10th IFIP International Conference on New Technologies, Mobility and Security, NTMS 2019 - Proceedings and Workshop, pp. 1–5. doi:10.1109/NTMS.2019.8763852.

Maleki, N., Bateni, M., Rastegari, H., 2019. An improved method for packed malware detection using pe header and section table information. Int. J. Comput. Netw. Inf. Secur. 11 (9), 9–17. doi:10.5815/ijcnis.2019.09.02.

Malik, Z., Shapiai, M.I.B., 2022. Human action interpretation using convolutional neural network: a survey. Mach. Vis. Appl. 33 (3), 37. doi:10.1007/s00138-022-01291-0.

Manavi, F., Hamzeh, A., 2021. Static detection of ransomware using LSTM network and PE header. In: 2021 26th International Computer Conference, Computer Society of Iran (CSICC), pp. 1–5. doi:10.1109/CSICC52343.2021.9420580.

Manavi, F., Hamzeh, A., 2022a. A novel approach for ransomware detection based on pe header using graph embedding. J. Comput. Virol. Hacking Tech. doi:10.1007/s11416-021-00414-x.

Manavi, F., Hamzeh, A., 2022b. Ransomware detection based on PE header using convolutional neural networks. The ISC Int. J. Inf. Secur. 14 (2), 181–192. doi:10.22042/isecure.2021.262846.595.

Meland, P.H., Bayoumy, Y.F.F., Sindre, G., 2020. The ransomware-as-a-service economy within the darknet. Comput. Secur. 92, 101762. doi:10.1016/j.cose.2020.101762.

Moreira, C., Sales Jr, C., Moreira, D., 2022. Understanding ransomware actions through behavioral feature analysis. J. Commun. Inf. Syst. 37 (1), 61–76. doi:10.14209/jcis.2022.7.

Moti, Z., Hashemi, S., Karimipour, H., Dehghantanha, A., Jahromi, A.N., Abdi, L., Alavi, F., 2021. Generative adversarial network to detect unseen internet of things malware. Ad Hoc Netw. 122, 102591. doi:10.1016/j.adhoc.2021.102591.

Moti, Z., Hashemi, S., Namavar, A., 2019. Discovering future malware variants by generating new malware samples using generative adversarial network. In: 2019 9th International Conference on Computer and Knowledge Engineering (ICCKE), pp. 319–324. doi:10.1109/ICCKE48569.2019.8964913.

Moussaileb, R., Cuppens, N., Lanet, J.L., Bouder, H.L., 2021. A survey on windows-based ransomware taxonomy and detection mechanisms. ACM Comput. Surv. 54 (6), 117. doi:10.1145/3453153.

Naeem, H., Ullah, F., Naeem, M.R., Khalid, S., Vasan, D., Jabbar, S., Saeed, S., 2020. Malware detection in industrial internet of things based on hybrid image visualization and deep learning model. Ad Hoc Netw. 105, 102154. doi:10.1016/j.adhoc.2020.102154.

Kingma D.P., Ba J.. Adam: A method for stochastic optimization. 2017. doi:10.48550/arXiv.1412.6980

Paper, D., 2020. Scikit-learn Classifier Tuning from Complex Training Sets. Apress, Berkeley, CA, pp. 165–188. doi:10.1007/978-1-4842-5373-1_6. Ch. 6

Poudyal, S., Gupta, K.D., Sen, S., 2019. Pefile analysis: a static approach to ransomware analysis. Int. J. Forensic Comput. Sci. 14 (1), 34–39. doi:10.5769/J201901004. http://ijofcs.org/V14N1-PP004-PEFile-analysis.pdf

Pranggono, B., Arabo, A., 2021. Covid-19 pandemic cybersecurity issues. Internet Technol. Lett. 4 (2). doi:10.1002/itl2.247. E247

Preuveneers, D., Joosen, W., 2021. Sharing machine learning models as indicators of compromise for cyber threat intelligence. J. Cybersecur. Priv. 1 (1), 140–163. doi:10.3390/jcp1010008.

Rao, S., Böhle, M., Schiele, B., 2022. Towards better understanding attribution methods. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 10213–10222. doi:10.1109/CVPR52688.2022.00998.

Ray, S., 2019. A quick review of machine learning algorithms. In: 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), pp. 35–39. doi:10.1109/COMITCon.2019.8862451.

Rezaei, T., Hamze, A., 2020. An efficient approach for malware detection using PE header specifications. In: 2020 6th International Conference on Web Research (ICWR), pp. 234–239. doi:10.1109/ICWR49608.2020.9122312.

Rezaei, T., Manavi, F., Hamzeh, A., 2021. A pe header-based method for malware detection using clustering and deep embedding techniques. J. Inf. Secur. Appl. 60, 102876. doi:10.1016/j.jisa.2021.102876.

van Rijn, J.N., Hutter, F., 2018. Hyperparameter importance across datasets. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '18, Association for Computing Machinery. New York, NY, USA, pp. 2367–2376. doi:10.1145/3219819.3220058.

Oz, H., Aris, A., Levi, A., Uluagac, A.S., 2022. A survey on ransomware: evolution, taxonomy, and defense solutions. ACM Comput. Surv. 54 (11s), 238. doi:10.1145/3514229.

Shaheed, K., Mao, A., Qureshi, I., Kumar, M., Hussain, S., Ullah, I., Zhang, X., 2022. Ds-cnn: a pre-trained xception model based on depth-wise separable convolutional neural network for finger vein recognition. Expert Syst. Appl. 191, 116288. doi:10.1016/j.eswa.2021.116288.

Shaik, N.S., Cherukuri, T.K., 2021. Lesion-aware attention with neural support vector machine for retinopathy diagnosis. Mach. Vis. Appl. 32 (6), 126. doi:10.1007/s00138-021-01253-y.

Sharma, S., Kumar, S., 2022. The xception model: a potential feature extractor in breast cancer histology images classification. ICT Express 8 (1), 101–108. doi:10.1016/j.icte.2021.11.010.

Singh, J., Singh, J., 2022. Assessment of supervised machine learning algorithms using dynamic api calls for malware detection. Int. J. Comput. Appl. 44 (3), 270–277. doi:10.1080/1206212X.2020.1732641.

Sgandurra D., Muñoz González L., Mohsen R., Lupu E.C. Automated dynamic analysis of ransomware: benefits, limitations and use for detection. 2016. doi:10.48550/arXiv.1609.03020.

Szegedy C., Ioffe S., Vanhoucke V., Alemi A.. Inception-v4, inception-resnet and the impact of residual connections on learning. 2016. doi:10.48550/arXiv.1602.07261

Thamer, N., Alubady, R., 2021. A survey of ransomware attacks for healthcare systems: Risks, challenges, solutions and opportunity of research. In: 2021 1st Babylon International Conference on Information Technology and Science (BICITS), pp. 210–216. doi:10.1109/BICITS51482.2021.9509877.

Ucci, D., Aniello, L., Baldoni, R., 2019. Survey of machine learning techniques for malware analysis. Comput. Secur. 81, 123–147. doi:10.1016/j.cose.2018.11.001.

Verma, M., Kumarguru, P., Deb, S.B., Gupta, A., 2018. Analysing indicator of compromises for ransomware: Leveraging iocs with machine learning techniques. In: 2018 IEEE International Conference on Intelligence and Security Informatics (ISI), pp. 154–159. doi:10.1109/ISI.2018.8587409.

Vidyarthi, D., Kumar, C., Rakshit, S., Chansarkar, S., 2019. Static malware analysis to identify ransomware properties. Int. J. Comput. Sci. Issues 16 (3), 10–17. doi:10.5281/zenodo.3252963.

Wen, Q., Chow, K., 2021. Cnn based zero-day malware detection using small binary segments. Forensic Sci. Int. Digital Invest. 38, 301128. doi:10.1016/j.fsidi.2021.301128.

Yadavendra, Chand, S., 2020. A comparative study of breast cancer tumor classification by classical machine learning methods and deep learning method. Mach. Vis. Appl. 31 (6), 46. doi:10.1007/s00138-020-01094-1.

Zahoora, U., Rajarajan, M., Pan, Z., Khan, A., 2022. Zero-day ransomware attack detection using deep contractive autoencoder and voting based ensemble classifier. Appl. Intell. 52 (12), 13941–13960. doi:10.1007/s10489-022-03244-6.

Zhang, H., Xiao, X., Mercaldo, F., Ni, S., Martinelli, F., Sangaiah, A.K., 2019. Classification of ransomware families with machine learning based on n-gram of opcodes. Future Gener. Comput. Syst. 90, 211–221. doi:10.1016/j.future.2018.07.052.

Tan M., Le Q.V.. Efficientnetv2: Smaller models and faster training. 2021. doi:10.48550/arXiv.2104.00298

Noever D., Noever S.E.M.. Virus-mnist: A benchmark malware dataset. 2021. doi:10.48550/arXiv.2103.00602

Caio Carvalho Moreira received the M.Sc. degree in electrical engineering from the Federal University of Pará (UFPA), Belém, Pará, Brazil, in 2013. He served as a computer system analyst at the Brazilian Air Force from 2013 to 2017. He is currently a Ph.D. student in the Electrical Engineering Graduate Program at UFPA. His contemporary research addresses ML techniques applied to network and system security, particularly ransomware detection.

Davi Carvalho Moreira received the Ph.D. in electrical engineering from the Federal University of Pará (UFPA), Belém, Pará, Brazil, in 2021. Since 2005, he has been with the Electrical Center of North Brazil, Tucurui, Pará, Brazil, where he is currently with the HPP Tucurui in the Electrical Maintenance Department. He has experience in the operation of hydroelectric plants, maintenance planning, equipment design/specifications, and maintenance of generators, transformers, and gas insulated substations.

Claudomiro de Souza de Sales Júnior received the Ph.D. in electrical engineering from the Federal University of Pará (UFPA), Belém, Brazil, in 2009. Since 2010, he has been with UFPA, where he is currently with the Computer Science Department, and Electrical Engineering and Computer Science Graduate Programs. He is involved in developing new dimension reduction and data visualization techniques, applying machine learning algorithms for ransomware detection and bioinformatics, and developing new metaheuristics and variants for PSO and genetic algorithms.