Remote Access Trojans Detection Using Convolutional and Transformer-based Deep Learning Techniques

Iustin FLOROIU[1], Miruna FLOROIU[2], Alexandru-Constantin NIGA[2], Daniela TIMISICA[1,3]
[1]National Institute for Research and Development in Informatics - ICI Bucharest
[2]Independent Researcher
[3]University of Science and Technologies Politehnica of Bucharest
iustin.floroiu@ici.ro, floroiumiruna1224@gmail.com, alex.nigactin@gmail.com, daniela.timisica@ici.ro

Abstract: In recent years, the cybersecurity landscape has been marked by an increased focus on malware attacks, specifically on Trojan-type malware attack that poses a significant threat to Windows and Linux operating systems. This threat underscores the necessity for proactive prevention measures. The present study employs intelligent algorithms to detect malware, with a specific focus on identifying Remote Access Trojans (RATs). RATs are chosen for examination due to their persistent evolution, their increased number of attacks in the last period and, also, their discreet operation within executable files. The dataset for the algorithms is created by computing texture images over a large number of executable files infected with RATs. Various machine learning models, including VGG-16, VGG-19, and ResNet50, are explored for their effectiveness in identifying malicious programs. Additionally, this research delves into transformer architectures, such as the Vision Transformer (ViT), particularly in image classification tasks. Emphasizing the importance of integrating advanced machine learning techniques into cybersecurity efforts, this study aims to strengthen defense mechanisms against evolving cyber threats. Ongoing research is directed towards refining model performance and validating findings across diverse malware datasets.
Keywords: Remote Access Trojans (RATs),Deep Learning, Malware detection, Computer vision, Transformer architectures.

INTRODUCTION

In recent years, the latest developments in the cybersecurity world have been dominated by reports of malware attacks. In February 2024, The AV-TEST Institute conducted a comprehensive analysis showing that, particularly in the case of Windows and Linux operating systems, Trojan-type malware stands out as the most common threat.

This implies a persistent evolution of cyber threats, emphasising the importance of robust security measures and proactive defense strategies in protecting systems.

https://doi.org/10.54851/v6i1y202405

The large difference regarding the number of several recently discovered malware on Linux and Windows operating systems can be observed in the diagrams from Figure 1.

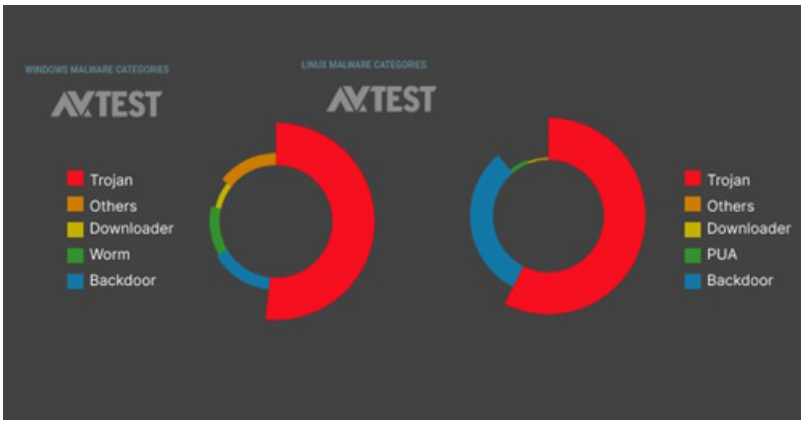

Figure 1. *Newly discovered Windows and Linux Malware and Potentially Unwanted Applications (PUA) according to the type of threat(AV-TEST Institute, 2024)*

The motivation behind this work is related to the need for faster methods of detecting malware, based on the principle that, the longer an attack lasts, the more damage it produces and the more resources it steals from the victim. The cost of cybercrime in 2015 was estimated at $3 trillion, but is now projected to grow to over $10.5 trillion in 2025. More than 87% of businesses globally consider cybersecurity as their top threat to financial well-being, ranking it higher than concerns about economic downturns and skill shortages (Brooks, 2023).

The objectives of this study rely on the implementation of intelligent models that are capable to accurately detect malicious programs and differentiate well between Remote Access Trojan (RAT) malware and clean software. Another important objective of this study is to compare the implemented models, in terms of performance and behavior.

The following sections describe the theoretical notions covering both cybersecurity and artificial intelligence and thoroughly discuss the methods implemented for image transformations, other data preprocessing, and hyperparameterization of the intelligent models.

THEORETICAL NOTIONS

About Remote Access Trojans (RATs)

Malicious software, commonly referred to as malware, constitutes a category of software specifically crafted to compromise and exploit computer systems or users, without the users' knowledge. The term „malware" is derived from the combination of"malicious" and „software" highlighting its nefarious intent. Various types of malware exist, each differentiated by its behaviour, scope or objectives. Categories include adware, rootkits, and bots, among others. However, this article will concentrate on the Trojan issue, with a specific emphasis on Remote Access Trojans (RATs) (Namanya et al., 2018).

Trojans, in essence, are programs that appear legitimate, but hide malicious code. Typically presented as executables, trojans discreetly install harmful programs alongside harmless programs, thereby compromising the integrity of the infected system. In the realm of Trojans, a noteworthy subtype deserves attention – Remote Access Trojans (RATs).

In contrast with conventional Trojans, RATs have the capability to provide unauthorized access and control over a system through remote connections, without the host's knowledge (Yamada et al., 2015). This article will focus on the complexity of RATs aiming to identify files infected with them by relying on the texture derived from the transformation of executables into images. This family of malware often operates discreetly, being frequently found in executable files, due to the files' capability to seamlessly integrate with legitimate software. Additionally, executable files are easily distributable and can be attached to emails, embedded in malicious websites or shared through other methods. This convenience makes them a suitable path for RATs to infiltrate diverse systems, taking advantage of the privileges granted during the installation process (Kara &Aydos, 2019).

As can be seen in the AV-TEST statistics below (Figure 2), executables files represent the newly discovered threats for the Windows operating system, that use this type of programs most frequently.

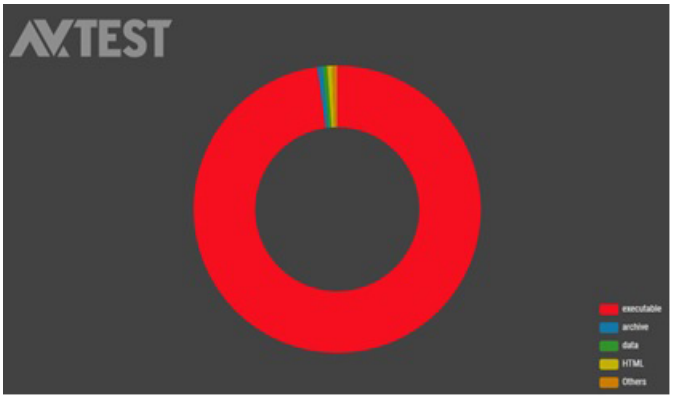

Figure 2. *Newly discovered Windows Malware and PUA are categorized according to their file type (AV-TEST Institute, 2024)*

Among the encountered RAT variants, specific ones have been identified – Gh0stRAT, Remcos RAT, and NanoCore – due to their diverse and numerous sources available on the internet. Gh0stRAT is a well-known remote access tool that has been utilized in various cyber espionage campaigns. Notorious for its stealthy operation, Gh0stRAT allows remote attackers to gain unauthorized access to targeted systems, enabling them to execute commands, exfiltrate data and maintain persistent control(Samuel et al., 2017). Remcos RAT is a remote administration tool designed to provide attackers with comprehensive control over infected systems. Known for its versatility, Remcos enables a range of malicious activities, including keylogging, file manipulation, and screen capturing as mentioned by Valeros & Garcia (2020). NanoCore is a compact, yet harmful RAT that has gained notoriety for its effectiveness in facilitating unauthorized remote access. Recognized for its modular design, NanoCore allows threat actors to customize its capabilities based on their specific objectives. This adaptability makes it a formidable tool for executing various malicious activities, such as data theft and system manipulation(Valeros & Garcia, 2020).

For all of the above there are a lot of variants which suffered changes compared to the original trojan version over the years, but have the same behaviour in terms of attacking the host, as mentioned also by Boinapally et al. (2017). For example, one of the changes made to the new types of GhostRat refers to the packing structure

of the RAT. There are different ways that a Gh0st RAT can be packed, for example standalone, included in other files or included in self-extracted archives (Norman, 2012). Although there will always be some minor changes, each RAT has a peculiarity according to which it can be classified. The distinguishing property of Gh0st is the „magic word" of network communication (Figure 3).– the RAT's name encoded, as can be observed below (Norman, 2012).

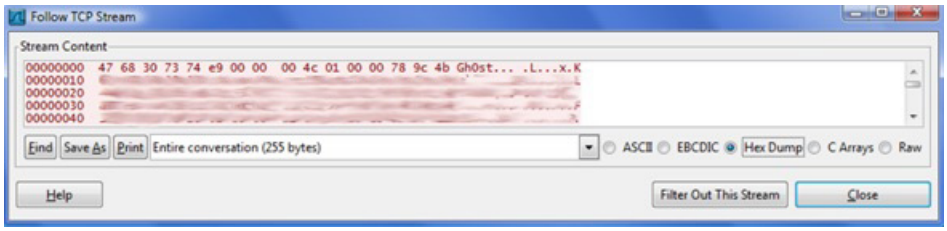

Figure 3. *The fields of a network packet containing the "magic word" identifier ('Gh0st') (Norman, 2012)*

With each move made by the defense team, the attack team or the hackers develops and learns new methods to exploit vulnerabilities in a system. Thus, as defense strengthens, hackers will create more aggressive malware to exploit the new defense systems and their vulnerabilities.

About Machine Learning Techniques

Artificial Intelligence revolves around the implementation of techniques that can be used implicitlyto obtain a desired output. While machine learning techniques are easy to use and computationally undemanding, deep learning models based on the design of human neural networks started have begun to emerge and develop fast. While more challenging from a computational perspective, these models obtain better results in a variety of tasks, including forecasting, classification and natural language processing. From the first implemented neural networks, one of the examples being the multilayer perceptron, to the more advanced and complex models, such as convolutional neural networks, the range of models and the domain usage have exponentially scaled (Floroiu et al., 2023; Oke, 2008).

Neural networks of higher complexity encompass multiple concealed layers. In specific scenarios, such as image-oriented applications, convolutional neural networks come into play. Convolution involves merging two signals to generate a fresh signal, relying on a kernel that outlines the window or the shape of the signal, as well as the signal undergoing reshaping. In the realm of frequency, convolution transforms into multiplication. Amid convolutional networks, certain concealed layers are designated as convolutional layers, which remodel visual data by using a chosen window magnitude, thus involving intricate matrix operations. Convolutional neural networks exhibit spatial invariance. Convolutional layers accept input in the form of tensors with four dimensions, encompassing parameters like length, width, height, and the count of data channels, especially significant in visual data contexts. Convolutional neural networks use convolutional layers that help them extract features locally, which are vital in the process of image understanding, classification and generation (Li et al., 2021). While deep models develop mathematical problems, especally regarding the gradient of the network, techniques that utilize residual connections were developed to solve these problems, called residual neural networks. Most of the deep learning models that are in demand are computationally challenging due to the high number of parameters used (Thorpe &van Gennip, 2018).

Due to the great development of the technologies implemented, transformer-based architectures were developed. These models

aid with the development of an internal model of the data given as input, which confides the algorithms with the possibility of forming partial generalisations about it. Transformer-based models use a mechanism called attention, which allows the models to take into consideration input data more effectively than traditional deep learning methods. While remaining computationally challenging, these models can offer enhanced results. In the field of computer vision and image-based processes, transformers can be implemented in a simpler way, which has been proven, during thisstudy, to be more effective with regards to image classification than the techniques previously discussed.

The Vanilla Transformer (Figure 4) is a type of neural network architecture designed for various natural language processing tasks. It employs a unique self-attention mechanism, avoiding the sequential nature of traditional recurrent neural networks. It consists of an encoder and a decoder. The encoder processes input sequences by calculating self-attention values for each word, capturing their relationships across the entire sequence. This enables the model to consider all words simultaneously and understand the context without relying on sequential processing. The decoder generates output sequences by also utilizing self-attention, but with an added step that attends to the output of the encoder, enhancing context and information exchange. During training, the model learns to optimize attention weights and generate coherent outputs. Positional encodings are added to the input embeddings, enabling the model to differentiate word positions, as self-attention alone lacks inherent position information. The architecture of the Vanilla Transformer allows for parallelization during training, thus speeding up computation. The attention mechanism provides a powerful tool for capturing context, making it particularly effective for tasks like translation, text generation, and language understanding. The Vanilla Transformer made breakthroughs in capturing global context and parallelization.

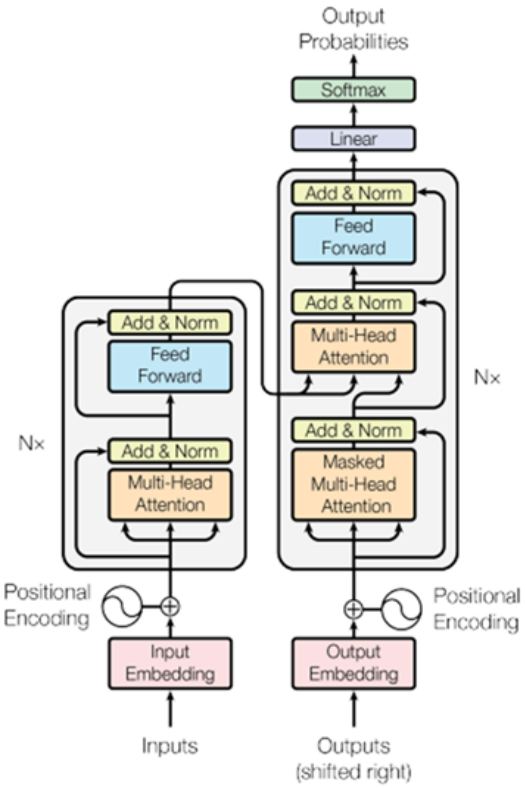

Figure 4. *The architecture of the Vanilla Transformer (Vaswani et al., 2017)*

The MultiHead Attention module (Figure 5), a pivotal element in the architecture of the Vanilla Transformer, introduced by Vaswani et al. (2017), holds a vital role in apprehending the connections amid diverse words (or tokens) within a sequence. This functionality empowers the Transformer to excel across various undertakings like machine translation, and text generation. This module processes three inputs: the input sequence (comprising word embeddings or features), commonly termed the query, alongside two parallel sequences named the key and the value, respectively, as depicted in Figure 4. While these sequences are fundamentally identical, their application varies in subsequent steps. Each of these inputs (query, key, and value) undergoes a distinct linear transformation through learned weight matrices. This transformation projects the inputs into distinct “subspaces” or „representations” that encapsulate varying facets of the connections between words. The core innovation within the MultiHead Attention module

is the integration of multiple "heads". Each head embodies a distinct learned attention pattern, effectively implying that the model acquires several strategies for assessing relationships within the sequence. For every head, the module computes attention scores by gauging the compatibility (similarity) between the query and the key vectors. Then, these scores are employed to assign weight to corresponding value vectors.

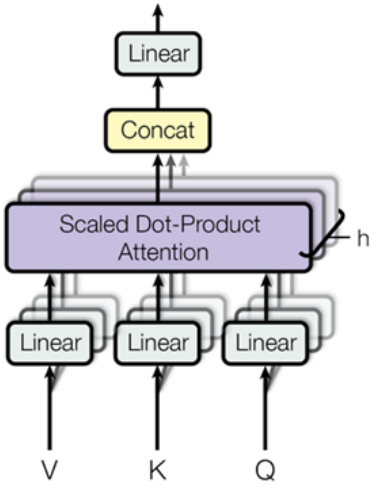

Figure 5. *The block architecture of the MultiHead Attention module (Vaswani et al., 2017)*

Vision Transformer (ViT) is a cutting-edge neural network architecture designed for image classification tasks, challenging traditional convolutional neural networks (CNNs). Unlike CNNs, ViT adopts the architecture of Vanilla Transformer, originally designed for natural language processing. ViT approaches image processing by reshaping 2D image data into sequences of flattened patches, treating them like tokens in a language. Then, these patches are passed through an initial linear embedding layer, transforming them into higher-dimensional embeddings that retain spatial information. ViT relies on the self-attention mechanism, enabling each patch to capture relationships with all other patches, mimicking how words interact in a language. Positional embeddings are also added to maintain the spatial arrangement of patches. ViT can outperform traditional CNNs on various image classification benchmarks, as its self-attention mechanism captures long-range dependencies and enables efficient parallelization, avoiding the limitations of sequential processing. However, ViT may require larger datasets, due to its need for diverse patch-level training data (Figure 6) (Zhou et al., 2021).

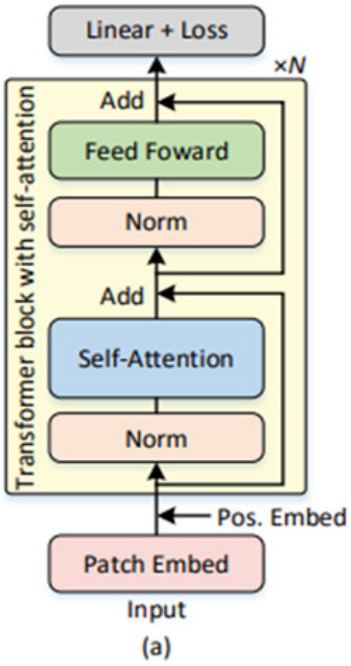

Figure 6. *The block architecture of the standard ViT model (Zhou et al., 2021)*

PRACTICAL IMPLEMENTATION

Dataset Description

The database used for training, validation and testing of the models consists of images sourced from publicly available archives on Kaggle, as well as of images processed in a custom manner from the malware files, following the acquisition of executable files of the RATs type. To create images,PE and .EXE extension files were downloaded from various sources, files known to be infected by RAT-type trojans, filtering three types of RATs: Gh0stRAT, Remcos RAT and NanoCore Rat. In addition to these classes, a "benign" category that includes binaries with clean installers was introduced.

The dataset contains 125.004 .PNG grayscale images retrieved from 17 malware families and also benign software. From the dataset, only the benign ones, namely Gh0stRAT, Remcos AND NanoCore labels were chosen, since the purpose of the paper is RAT classification.

As it can be seen in Figure 7, the dataset is highly uneven with regards to label distributions, which is why different executable files were customly designed as images, to enhance the dataset, for training, validation and testing alike.

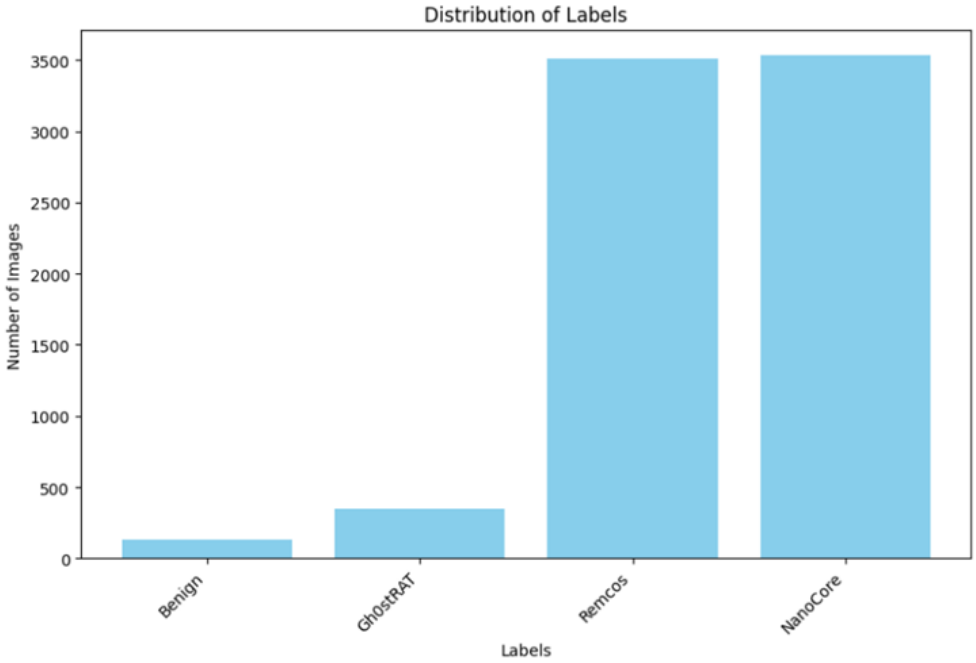

Figure 7. *The distribution of images based on the labels in the train dataset*

After gathering the files, the processing phase involved modelling the data segments for easy representation in the RGB image format. The content of executables was converted into bit-level vectors, organized into arrays, and transformed into RGB pictures. Consequently, each pixel was depicted as a tuple of colour pixels, each having a value within the range 0 to 255.

As observed in the results, the output images are similar to texture images, facilitating the detection of obfuscated malicious code by tracking areas of interest. Keeping in mind that the attack will evolve as the defense does, there will be more variants of RATs that can be identified by their image's area of interest, even if the obfuscation method will improve.

As noted by Omar (2022), there are various approaches for classifying malware stored in executable formats: static malware analysis, dynamic malware analysis, and hybrid malware analysis – a combination of the former two. The most powerful method is the hybrid analysis, taking advantage of both the static and the dynamic approach. Given the rapid advances with regards to deep learning models, the proposed solution revolves around the automated detection of malware instead of traditional methods and analysis techniques. The reason for this choice can be explained through the high recognition speed rate of the artificial intelligence models, which allows for faster detection. Besides the time motivation, another reason for choosing this approach is the detection of newly discovered malware. Using this technique, RATs that have similar behavior can be identified, providing the traceability of the model proposed in the present analysis to Trojans developed in the future.

An example of an image resulting from an executable process can be observed below (Figure 8):

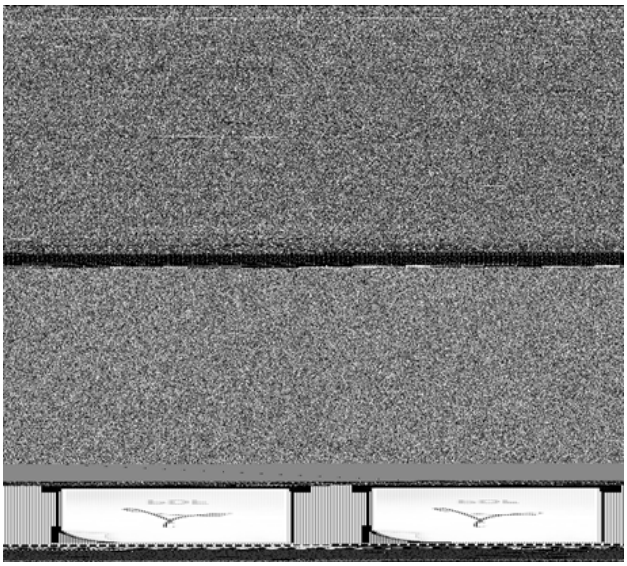

Figure 8. *Texture image based on NanoCore trojan-infected executable file*

Model Description

The Deep Learning models chosen to be implemented are VGG-16, VGG-19 and ResNET50. The choice was based on the models' performance in image classification.

VGG-16 is a convolutional neural network (CNN) architecture designed for image classification tasks. It was proposed by the Visual Geometry Group (VGG) at the University of Oxford. The number "16" present in its name refers to the total number of layers, namely 13 convolutional layers and 3 fully connected layers. The 13 convolutional layers are stacked on top of each other, with max-pooling layers in between, followed by 3 fully connected layers. The convolutional layers use small receptive fields, of 3x3, and a stride of 1. Padding is used to maintain the spatial resolution, while max-pooling is used to reduce spatial dimensions. Thus, the output feature maps have the same size as the input. Rectified Linear Unit (ReLU) is used as the activation function throughout the network, while Softmax is used in the output layer, for classification tasks. VGG-16 has a large number of parameters, around 138 million, which makes it computationally expensive to train and deploy, compared to smaller neural networks.

VGG-19 is an extension of the VGG-16 architecture, also proposed by the Visual Geometry Group (VGG) at the University of Oxford. The number "19" present in its name refers to the total number of layers, namely 16 convolutional layers and 3 fully connected layers. Similar to VGG-16, VGG-19 stacks convolutional layers with max-pooling layers in between them, followed by 3 fully connected layers. It uses 3x3 receptive fields with a stride of 1 for convolutional layers, maintains spatial resolution with padding, and employs ReLU activation throughout the network. Softmax is used in the output layer for classification tasks. With approximately 144 million parameters, VGG-19 is computationally expensive, but, at the same time, very effective for image classification tasks.

ResNet50 is a convolutional neural network architecture introduced by Microsoft Research. The number "50" present in its name refers to the total number of layers, including residual blocks. Unlike traditional architectures likeVGG, ResNet introduces residual connections, which help address the gradient problems encountered in very deep networks. ResNet50 consists of 50 layers, including convolutional layers, pooling layers, and fully connected layers. However, the core building blocks of ResNet are

residual. These residual blocks contain shortcut connections, also known as skip connections, that enable the network to learn residual mappings instead of directly learning the desired underlying mapping. This is achieved by adding the input of a layer to its output, effectively bypassing the layer. The presence of these residual connections allows ResNet to effectively train much deeper networks without the vanishing gradient problem. As a result, ResNet50 can maintain high performance, even with its increased depth. The residual connections help in propagating gradients through the network during training, which facilitates the learning process and enables the training of very deep neural networks. In addition to the use of residual connections, ResNet50 also employs other techniques common in convolutional neural networks, such as batch normalization and ReLU activation functions. These components collectively contribute to the success of ResNet50 in various computer vision tasks, including image classification and object detection. With around 23.5 million parameters, ResNet50 strikes a balance between model complexity and computational efficiency, making it widely used in both research and practical applications.

Due to the performance observed in other research papers on these architectures, the ResNet50 model is expected to perform better than VGG-16 and VGG-19 models.

Regarding fine-tuning, a learning rate of 0.00001 was chosen, alongside categorical cross-entropy as the loss function and the Adam optimizer. The dataset was divided into training (80% of data), validation (15% of total data) and testing (15% of total data). Due to the large amount of data used for training the model, a batch size of 32 is suitable, given the fact that a smaller batch size is less consuming from a computational perspective and that it allows for more randomness about the learning process of the models. The learning rate was set to reduce on the plateau, even though it was not the case for either of the three models implemented. Given the high complexity and the dimension of the dataset, 50 epochs were chosen for the simultaneously implemented training and validation processes.

This study desires to also draw a performance and functionality comparison between the implemented methods. For this sole purpose, the fine-tuning parameters were chosen to be the same in the implementation of all the models, including the ViT method. However, for the ViT architecture, 12 patches per image were implemented. The purpose of this choice revolves around the current data that was used and its texture-based aspect, which necessitates locality in its analysis, thus needing bigger patches for each image.

Results

Using the VGG-16 model, an accuracy of 69.81% was obtained during testing. The The dataset was divided into training (80% of data), validation (15% of total data) and testing (15% of total data) (Figure 9).

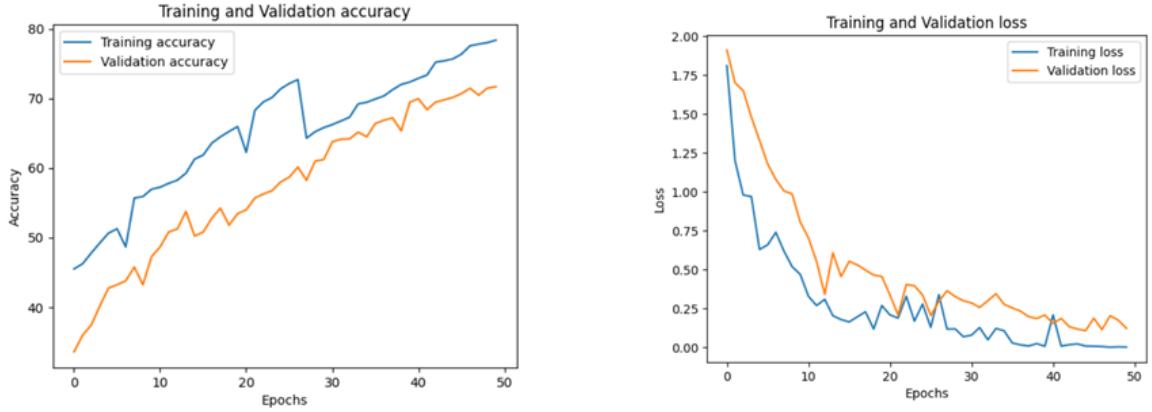

Figure 9. *The accuracy and loss for both training and validation of the VGG-16 model*

For the VGG-19 model, an accuracy of 73.47% was obtained during testing and the graphs of the accuracy and loss for both training and validation are shown below (Figure 10):

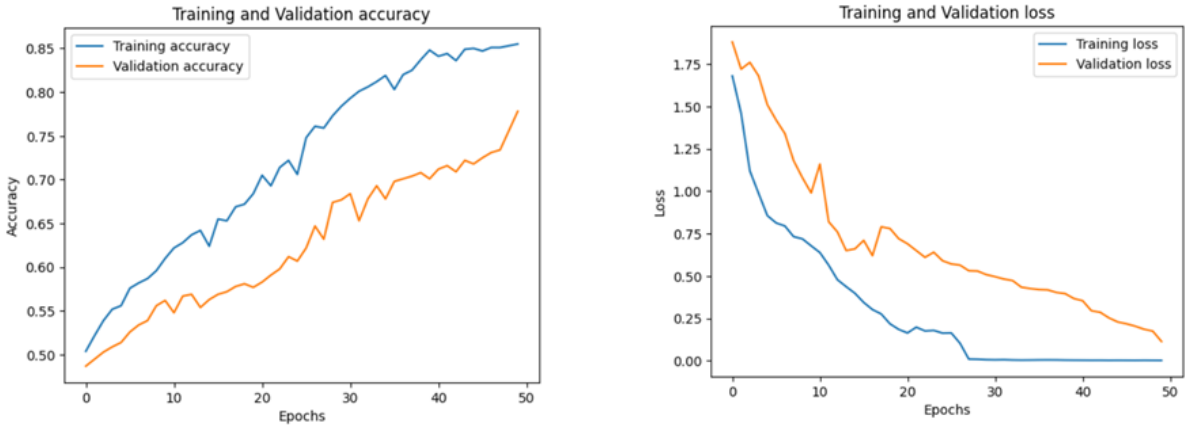

Figure 10. *The accuracy and loss for both training and validating the VGG-19 model*

For the ResNet model, the accuracy for testing peaked at 76.83 and the graphs of the accuracy and loss for both training and validation are shown below (Figure 11):

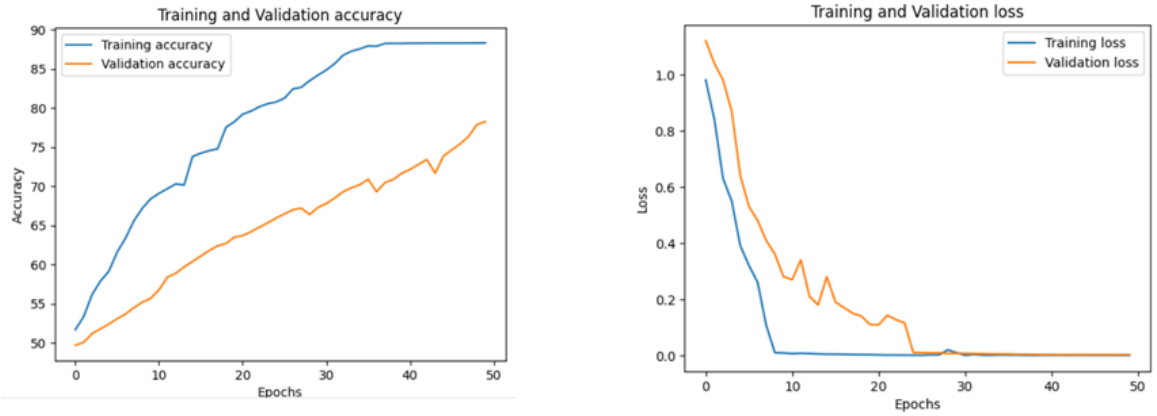

Figure 11. *The accuracy and loss for both training and validating the ResNet model*

For the Vision Transformer, an accuracy of 83.26% was obtained during testing and the graphs of the accuracy and loss for both training and validation are shown below (Figure 12):

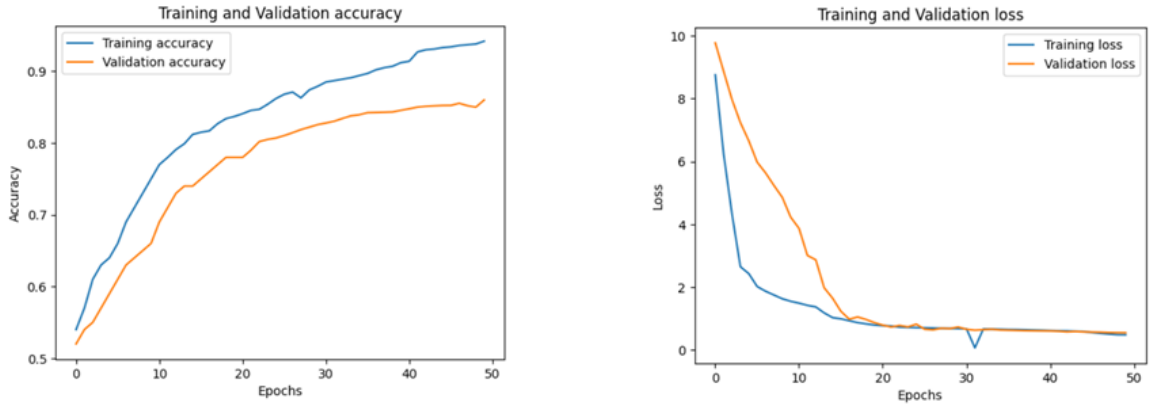

Figure 12. *The accuracy and loss for both training and validating the Vision Transformer*

CONCLUSION

In conclusion, the methodology currently implemented in this article shows that ResNte50 obtains better performance scores in terms of image classification and malware, more precisely, RAT detection, than VGG-16 and VGG-19. This can be explained by the residual network approach which is vital in residual neural networks.

Despite being a simplified model, in comparison with the VGG-16 and VGG-19 models, the residual network architectures solve the gradient problem that is present in deep convolutional networks. The vision transformer architecture, even though presents higher computational complexity, obtains the best metrics based on its ability to efficiently include and determine patterns, through its attention mechanism.

REFERENCE LIST

AV-TEST Institute. (2024) *Windows Malware Categories. AV-TEST Institute. The Independent IT-Security Institute.* https://portal.av-atlas.org/malware [Accessed 1st March 2024].

Boinapally, V., Hsieh, G. &Nauer, K.S. (2017) Building a gh0st malware experimentation environment. In: *Proceedings of the 2017 International Conference on Security and Management (SAM' 17),17 - 20 July 2017, Las Vegas, USA.* University of Detroit Mercy, USA,The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp).pp. 89-95.

Brooks, C. (20 October 2023) Spooky Cyber Statistics And Trends You Need To Know. *Forbes.* https://www.forbes.com/sites/chuckbrooks/2023/10/20/spooky-cyber-statistics-and-trends-you-need-to-know/ [Accessed 1st March 2024].

Floroiu, I., Timisică, D. & Boncea, R.M. (2023) Automated diagnosis of breast cancer using deep learning [Diagnosticul automatizat al cancerului mamar utilizând algoritmi de învățare profundă]. *Revista Română de Informatică și Automatică [Romanian Journal of Information Technology and Automatic Control].* 33(3), 99-112. doi:10.33436/v33i3y202308.

Kara, İ. &Aydos, M. (2019) The ghost in the system: technical analysis of remote access trojan. *International Journal on Information Technologies & Security.* 11(1), pp.73-84.

Li, Z., Liu, F., Yang, W., Peng, S. & Zhou, J. (2021) A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE Transactions on Neural Networks and Learning Systems.* 33(12), 6999-7019. doi: 10.1109/TNNLS.2021.3084827.

Namanya, A.P., Cullen, A., Awan, I.U& Disso, J.P. (2018)The World of Malware: An Overview. In: *2018 IEEE 6th International Conference on Future Internet of Things and Cloud (FiCloud), 06-08 August 2018, Barcelona, Spain.* New Jersey, USA, Institute of Electrical and Electronics Engineers (IEEE). pp. 420-427. doi: 10.1109/FiCloud.2018.00067.

Norman, A.S.A. (2012) The many faces of Gh0st Rat. *Yumpu*. https://www.yumpu.com/en/document/read/11503105/the-many-faces-of-gh0st-rat-norman[Accessed 1st March 2024].

Oke, S.A. (2008)A literature review on artificial intelligence. *International Journal of Information and Management Sciences*. 19(4), 535-570.

Omar, M. (2022) New Approach to Malware Detection Using Optimized Convolutional Neural Network. In: *Machine Learning for Cybersecurity: Innovative Deep Learning Solutions*. Cham, Switzerland, Springer International Publishing. pp. 13-35.

Samuel, S., Graham, J. & Hinds, C. (2017)Hunting malware: An example using gh0st. In: *2017 International Conference on Computational Science and Computational Intelligence (CSCI), 14-16 December 2017, Las Vegas, NV, USA*. New Jersey, USA, Institute of Electrical and Electronics Engineers (IEEE).pp. 97-102. doi: 10.1109/CSCI.2017.16.

Thorpe, M. & van Gennip, Y. (2018) Deep limits of residual neural networks. *arXiv*. [Preprint] https://arxiv.org/abs/1810.11741 [Accessed 1st March 2024].

Valeros, V.& Garcia, S. (2020) Growth and commoditization of remote access trojans. In: *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW),07-11 September 2020,Genoa, Italy.* New Jersey, USA, Institute of Electrical and Electronics Engineers (IEEE). pp. 454-462. doi:10.1109/EuroSPW51379.2020.00067.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. & Polosukhin, I. (2017) Attention is all you need. In: *31st Conference on Neural Information Processing Systems (NIPS 2017), 4-9 December 2017, Long Beach, CA, USA*. New York, USA, Curran Associates Inc.pp. 5998–6008.

Yamada, M., Morinaga, M., Unno, Y., Torii, S. & Takenaka, M. (2015) December. RAT-based malicious activities detection on enterprise internal networks. In: *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST),14-16 December 2015,London, UK*. New Jersey, USA, Institute of Electrical and Electronics Engineers (IEEE). pp. 321-325.

Zhou, D., Kang, B., Jin, X., Yang, L., Lian, X., Jiang, Z, Hou, Q. & Feng, J. (2021) Deepvit: Towards deeper vision transformer. *arXiv*. [Preprint] https://arxiv.org/abs/2103.11886[Accessed 1st March 2024].