Analysing the Influence of Activation Functions in CNN Models for Effective Malware Classification

Siddhi Kasliwal
Department of Computer Science
Christ University, Bengaluru -58
Karnataka, India

***Abstract*—With the advancement of information technology, malware has become a persistent cyber security concern that targets computer systems, smart devices, and wide networks. Due to flaws in performance accuracy, analysis type, and malware classification methodologies that miss unsuspected malware attacks, malware classification has thus always been a significant concern and a challenging subject.Using the Malimg dataset, which has 9349 samples from 25 different families, we are classifying malware using a deep learning algorithm called a convolution neural network and evaluating the accuracy using a number of activation functions in this study. The proposed CNN model for malware classification achieves a high accuracy rate without the need for complex feature engineering. The model achieved the highest accuracy of 96.93% when using the Rectified Linear Unit (ReLU) activation functionm whereas Leaky Relu gives accuracy of 96.76%, Pre relu gives 96.36%, ELU gives 95.72% and tanh gives accuracy of 95.58%.**

***Index Terms*—Malware classification, CNN, Deep Learning, Activation function.**

I. Introduction

One of the few universally understood technical terminology is ”computer virus,” which even young children can understand.No matter their age or socioeconomic status, everyone immediately associates that term with something terrible.It frequently involves damage to the technology that we all rely on, whether it be a laptop, smartphone, app, or gaming system, illustrating how extensive computers and technology have grown in our daily lives. Malware includes things like computer viruses. The term ”malware,” also referred to as ”malicious software,” refers to any program or piece of code created with the goal of harming a computer, network, or server.A type of intrusive software is malware, harms and destroys networks, servers, host systems, and computer systems. It serves as a blanket word for any forms of malicious software that are created with the express purpose of harming or abusing any programmable system, network, or service.

Malware threats include viruses, worms, adware, spyware, trojan viruses, and ransomware.

- The CNN method and the detection of malware in new technologies are the key goals of this research project.
- The dataset was split into two sets by the author; the first set, which contains 70% of the data, is used to train the dataset, and the second set, which contains 30% of the data, is used to test the trained CNN model.
- Without utilizing any complex feature engineering, our deep learning model achieves a 96% accuracy rate.

II. Literature Review

To effectively find packed, unpacked malware, a novel ensemble CNN built architecture is suggested in the paper. Image-based Malware Classification Using Ensemble of CNNs is the term we have given to this technique. Our basic premise is that distinct CNNs produce distinct semantic portrayal of image based on their deeper architectures; as a result, a set of CNN architectures enables the extraction of features with higher quality than conventional techniques. The outcome shows over 99% accuracy for malware that has been unpacked and over 98% accuracy for malware that has been packed. IMCEC is adaptable, useful, and effective because it only requires 1.18 on average to describe new malware selected[1]

The gated recurrent unit (GRU) language algorithim and the long short-term memory (LSTM) language algorithm are only two of the cutting-edge malware categorization models we provide in this study. Also we recommend utilizing an attention mechanism alike to form the machine translation composition as a replacement for constructing the document representation from neural features, in along with the temporal max pooling method used in. Eventually, we suggest a novel single-stage malware classifier built on the basis of a convolutional neural network (CNN) at the character level. Outcome represents that LSTM with temporal max pooling and logistic regression gives a 31.3% increase in the true positive rate over the finest organization at a false positive rate of 1%.[2]

The pre-trained MobileNetV2 and ResNet-50 algorithm, a working CNN algoritm, and two transfer learning models were the three models used in this investigation. There was also used a dataset containing 51880 samples, 10 unique sample families, 9 sub-classes of malware images, 1 class of benign file images. The outcome represents that the method is 99% accurate at classifying malware families. We evaluate the resilience of our models using the Malimg dataset, and the findings reveal that our models are 100% accurate at detecting malware that has been obfuscated.[3]

In this study, we introduce a latest CNN based malware classification algorithm that employs binary malware files rather than disassembled malware files and concentrates on malware family classification, which does not call for disassembly. Our approach connects two modalities, ”malware images” and

”structural entropies,” that are changed and retrieved from binary files. Our model is validated using three well-known datasets from the Kaggle Microsoft Malware Classification, Malimg, and BODMAS datasets. The innovative outcome show’s, without the time-consuming disassembly, our model classifies malware families more precisely than earlier approaches.[4]

In this study, we try to improvise the presentation of the MLP mixer by using an autoencoder (AE). In many applications, dimensionality reduction to eliminate noise and identify key input data components typically uses AE. We suggest a light ensemble model exploiting this advantage of AE by integrating a customizer MLP-mixer and Autoencoder to augment charaterstics retrieved from the MLP-mixer with encoder-decoder architecture of autoencoder. [5]

The development of recently suggested Deep Learning-based malware detection models is the subject of the study. It gives a thorough examination of newest Deep Learning-based malware observation procedure. Detection techniques for malware on mobile devices (including iOS and Android), Windows, IoT, advanced persistent threats, and ransomware are also closely explored. Additionally, new malware is researched.[6]

This paper offers a thorough examination and new perspectives on the application of AutoML for both static and online virus detection. We correlate an AutoML approach against 6 current state-of-the-art CNNs.Our experimental findings demonstrate that static and online malware detection models based on AutoML outperform hand-crafted models or cutting-edge models described in the literature in terms of performance.[7]

We introduce a Deep Learning based CNN algorithm to excute malware classification on Portable Executable (PE) binary files in this paper using fusion feature technique. Using fusion feature sets, the suggested algorithm successfully detects malicious or benign files with an accuracy 97%, according to our investigation of its performance.The suggested approach performed similarly on two malware datasets that had never before been observed and is reliable and generalizable. To comprehend the characteristics of the datasets, numerous visualization techniques are used along with the visualization of the embedding features of CNN algorithm.[8]

The trial consequence demonstrate that Random Forest outperformed competition with a 99.44 percent accuracy level for malware detection. With this, it is possible to create a desktop application for the Windows platform that can be customized to check for malware.[9]

In this study, we suggest a machine learning based paradigm that may be explained for hardware-assisted malware detection. We demonstrate theoretically that the ability of our suggested technique to give a clarification of categorization findings can handle the challenge of transparency. Then, we show how correctly leveraging hardware presentation counters and an embedded trace buffer can cause the exact localization of malicious behavior. Our platform has successfully passed rigorous testing using a various of real world malware datasets, demonstrating its ability to deliver precise and understandable malware detection findings with verifiable assurances. [10]

This approach aims to accomplish high malware detection rates and significantly give to the study and work being done to protect the development of mobile information. It is based on the Support Vector Machines, K-nearest neighbors, and Naive Bayes machine learning techniques, which are all well-known.[11]

The two objectives of the study are to find out the use of hybrid image based approaches paired with deep learning model for an productive malware classification and to explain the usage of image-based techniques for identifying systems that exhibit suspicious behavior. Using large malware datasets that are both publicly available and privately acquired, our proposed hybrid method is scaled, demonstrating the exceptional accuracy of our malware classifiers. [14]

III. Proposed Work

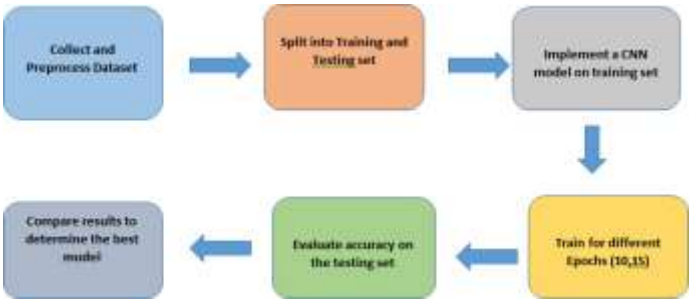

Figure 1: Steps of getting result

The proposed work focuses on utilizing a CNN model for malware classification. The Malimg dataset, includes 9,349 samples from 25 malware families, is used for training and testing the model. The dataset represents malware binaries converted into grayscale images, where each 8-bit vector is treated as a pixel value. The CNN algorithm is trained on 70% of the data and evaluated on the rest 30%, as the steps shown in fig 1. The presentation of the model is compared using different activation functions, namely ReLU, Leaky ReLU, PreReLU, ELU, and tanh. The highest accuracy of 96.93% is achieved with the ReLU activation function, while other activation functions also yield high accuracies ranging from 95.58% to 96.76%. This has been found out using Python Programming language.

A. Dataset

The Malimg dataset’s central concept is the process by which malware binaries that are converted into the grayscale images and added to the Malimg collection. Each of the 8-bit vectors that make up the malware binary is handled as a pixel value in a grayscale image.

The dataset contains 9,349 malware pictures that represent 25 different malware families mentioned in table 1. It is noteworthy that the Skintrim.N malware family contains least samples, while the Allaple.A malware family has the most samples .

TABLE I: Family and the type of Malware

No.	Family/class	Type
1	Adialer.C	Dialer
2	Agent.FYI	Backdoor
3	Allaple.A	Worm
4	Allaple.L	Worm
5	Alueron.gen!J	Worm
6	Autorun.K	Worm:AutoIT
7	C2Lop.P	Torjan
8	C2Lop.gen!G	Torjan
9	Dialplatform.B	Dialer
10	Dontovo.A	Downloader
11	Fakerean	Rouge
12	Instanacess	Dialer
13	Lolyda.AA1	PWS
14	Lolyda.AA2	PWS
15	Lolyda.AA3	PWS
16	Lolyda.AT	PWS
17	Malex.gen!J	Torjan
18	Obfuscator	Downloader
19	Rbot!gen	Backdoor
20	Skintrim	Torjan
21	Swizzor.gen!1	Downloader
22	Swizzor.gen!E	Downloader
23	VB.AT	Worm
24	Wintrim.BX	DOWNLOADER
25	Yuner.A	Worm

B. Malware as Image

This study discovered that viruses become more obvious when seen as photos. Deep learning has the ability to find patterns in images. CNN models are particularly helpful for image classification as they can pull out key properties from an image by sub sampling, pooling, and calculations. As a result, they are quite good at categorizing images.

Figure 2 illustrates a technique for changing a binary PE document into a series of 8-bit vectors or hexadecimal values that can be used to turn malware binaries into graphics. An 8-bit vector (255) in Figure 1 can be represented by the integers 00000000 (zero) through 11111111. In a virus image, 8-bit vectors shows numerical values which can be translated into pixels.[13]

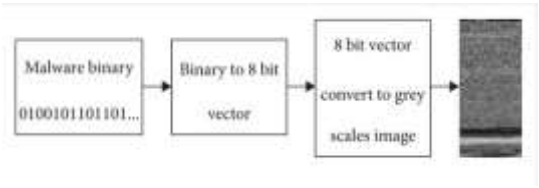

Figure 2: Converting malware binary to an image

C. Convolution Neural Network (CNN)

Convolution neural network is a deep learning algorithm which is generally used for computer vision. A CNN has various hidden layers that help in taking data from an image .The only levels of the network that actively learn parameters are the convolution and, Fully Connected layers, the remaining layers are solely in charge of carrying out a specific operation.

D. Need of an Activation function

Without an activation function, the output signal of a convolution neural network would only be a simple linear function or a polynomial of degree one. In spite it's simple and quick to solve linear equations, they can't learn or recognize complex mappings from data because their complexity is limited. An absence of activation functions in a neural network typically results in a linear regression model with poor performance and power. In addition to learning and computing a linear function, it is desirable for a neural network to be able to model complex types of data, like images, videos, audio, voice, text, etc.

E. Types of Activation function

There are various types of activation functions, in this paper we will talk about a few of them.

1) *Sigmoid function:* The sigmoid function modifies values between 0 and 1 as shown in fig3. You can define it as

$$f(x) = 1/e^{-}x. \quad (1)$$

The function's derivative is

$$f'(x) = 1 - sigmoid(x). \quad (2)$$

Additionally, because the sigmoid function is asymmetric around 0, all of the neurons' output values will have identical signs.To figure out this issue, the sigmoid function should be scaled.

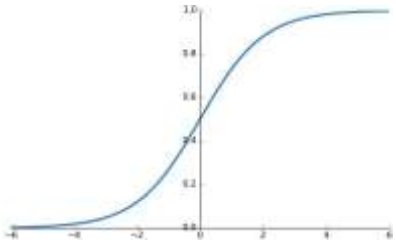

Figure 3: Sigmoid graph

2) *tanh:* It has a hyperbolic tangent function. The tanh function is symmetric at the origin, much like the sigmoid function is. As a result, the sign of the result from earlier layers that would be used as input for the following layer will differ. The formula is:

$$f(x) = 2sigmoid(2x) - 1 \quad (3)$$

According to Fig. 4, the Tanh function is continuous and differentiable, and its value range from -1 to 1. Tanh is choose over sigmoid because it is zero-centered and has gradients that are free to fluctuate in any direction.

3) *Relu function:* The non-linear activation function known as ReLU, or rectified linear unit, is often employed in neural networks. The ReLU function has the benefit of preventing simultaneous activation of all neurons. This suggests that a neuron won't stop functioning until its linear transformation's output is 0. It could be mathematically stated as f(x) = max(0,x). ReLU is more effective than other processes since

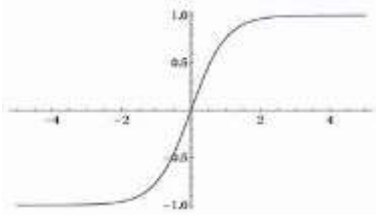

Figure 4: tanh graph

just few neurons are excited at once as opposed to all of them.

Figure 5: Relu graph

4) *Leaky ReLU function:* Leaky ReLU is a modified kind of the ReLU function, where the value of the function is determined as an exceedingly small linear component of x rather than zero for negative values of x as seen in fig. 6. It could be stated mathematically as follows:

$$f(x) = 0.01x, x < 0 f(x) = x, x >= 0 \quad (4)$$

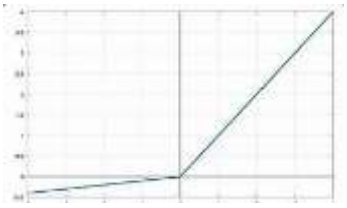

Figure 6: Leaky ReLU function

5) *Parameterized ReLU function:* A generalization of the conventional rectified unit with a slope for negative values is the parametric rectified linear unit, or PReLU. The issue of the gradient of ReLU being 0 for negative values of x is fixed by adding a new parameter to the negative component of the function, as shown in fig 7. It is written as follows:

$$f(y) = y, y >= 0, and f(y) = ay, y < 0. \quad (5)$$

6) *Exponential Linear Unit:* The exponential linear unit, or ELU, is a variant of the rectified linear unit. ELU adds the slope parameter for x's negative values as shown in fig 8. The negative values are defined using a log curve. [12]

$$f(x) = x, x >= 0 f(x) = a(e^x - 1), x < 0 \quad (6)$$

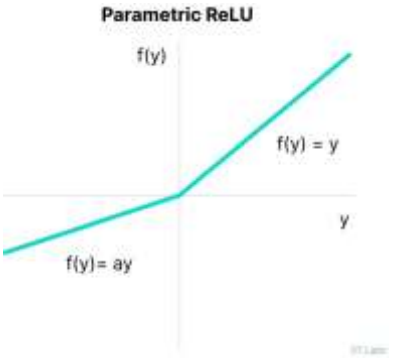

Figure 7: Parameterized ReLU function

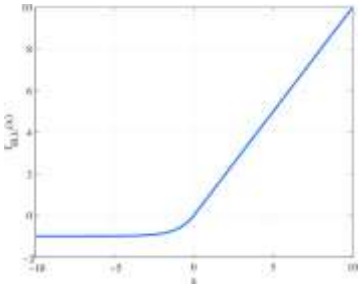

Figure 8: Exponential Linear Unit

F. CHOOSING THE RIGHT ACTIVATION FUNCTION

The activation function is among the most important considerations.. There is no general guideline for choosing any activation function, but the decision is task-specific, or based on the context. The advantages and drawbacks of various activation functions vary depending on the kind of the system we are designing.

Example

1) A combination of sigmoid functions produces better outcomes for classification issues.
2) Sigmoid and tanh functions are avoided because of the vanishing gradient problem, which is when the gradient becomes zero.
3) We can apply the leaky ReLU function if our network contains any dead neurons.

The CNN model's final layer requires the application of a different activation function known as Softmax activation. By computing the exponents of every output and then normalizing every number by the sum of those exponents so that the whole output vector adds up to one, the SoftMax function translates logit value into probabilities.

If this activation function is not used at the output layer the accuracy is decreased, for example by taking the malimg dataset we tried sigmoid, tanh, Relu activation and got the accuracy of 0.3105, 0.0128 and 0.0114 respectively, which was very less.

There is another important factor which plays a role in increasing or decreasing the accuracy which is the dropout layer.

The Dropout layer is a mask that excludes certain neurons from contributing to the following layer while maintaining the functionality of all other neurons. In order to get the highest level of accuracy, we tested a dropout layer in this paper.

IV. Discussion

Implementation of CNN Algorithm

- Convolutional Layer(CL) : 30 filters, (3 * 3) kernel size
- Max Pooling Layer(MPL) : (2 * 2) pool size
- CL : 15 filters, (3 * 3) kernel size
- MPL : (2 * 2) pool size
- DropOut Layer(DL) : Dropping of neurons
- Dense/Fully Connected Layer(FCL) : 128 neurons, activation function
- DL : Dropping of neurons.
- FCL Layer : 50 neurons, activation function
- FCL : numclass neurons, Softmax activation function.

This study observed that a different combination of dropout layer with activation function and epoch results gives how much accuracy.

In below graphs Red line represents Accuracy when epoch is 15 and Blue line represents accuracy when epoch is 10.

For example, Figure 9 depicts the accuracy which we got

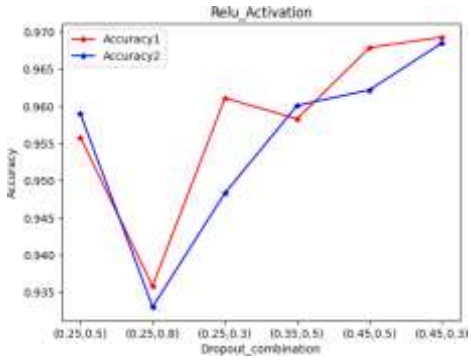

Figure 9: Accuracy with epoch 10 and 15 vs Dropout combination

through CNN model using Relu Activation function and got maximum accuracy of 96.93% when epoch is 15.

Figure 10 depicts the accuracy which we got through CNN model using Leaky Relu Activation function and got maximum accuracy of 96.36% when epoch is 10.

Figure 11 depicts the accuracy which we got through CNN model using Parametric linear unit Activation function and got maximum accuracy of 96.76% when epoch is 10.

Figure 12 depicts the accuracy which we got through CNN model using Exponential Relu Activation function and got maximum accuracy of 95.72% when epoch is 15.

Figure 13 depicts the accuracy which we got through CNN model using Exponential Relu Activation function and got maximum accuracy of 95.58% when epoch is 10.

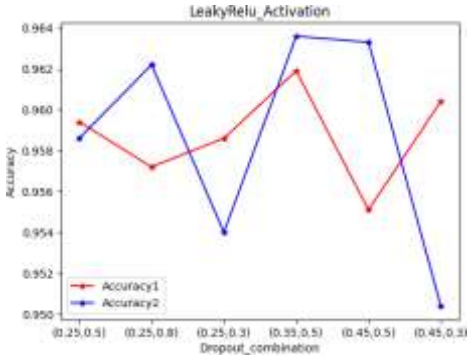

Figure 10: Accuracy with epoch 10 and 15 vs Dropout combination

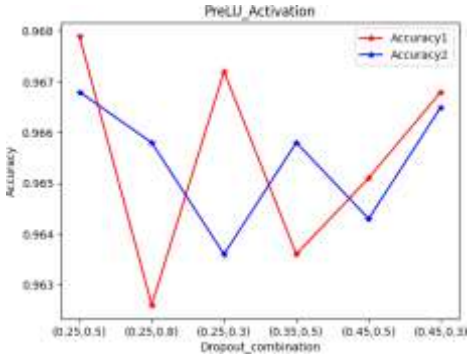

Figure 11: Accuracy with epoch 10 and 15 vs Dropout combination

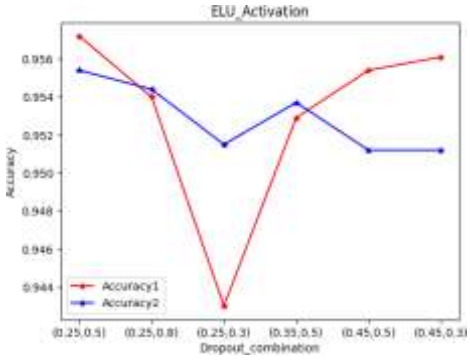

Figure 12: Accuracy with epoch 10 and 15 vs Dropout combination

A. *Comparing the accuracy of different Activation functions.*

Got the maximum accuracy with the Relu Activation function as shown in Figure 14 of 96.93%. This indicates that ReLU was most effective in capturing the relevant features and patterns in the malware images, leading to accurate classification. The Leaky ReLU activation function closely followed ReLU with an accuracy of 96.76%. PreReLU, ELU, and tanh also yielded competitive accuracies of 96.36%, 95.72%, and 95.58%, respectively. These results suggest that these

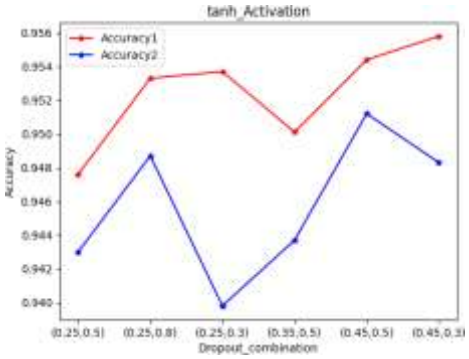

Figure 13: Accuracy with epoch 10 and 15 vs Dropout combination

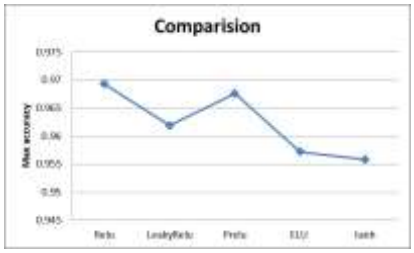

Figure 14: Comparison of Different Activation function

activation functions are capable of learning and representing the underlying characteristics of the malware images, although with slightly lower accuracy compared to ReLU and Leaky ReLU. Overall, the comparison of activation functions in our experiments highlights the importance of selecting an appropriate activation function for optimizing the performance of the CNN model in malware classification tasks.

V. Conclusion

The paper comes to the conclusion that the suggested CNN-based approach delivers accurate malware classification without significant feature engineering. Among the activated functions studied, the ReLU activation function are highly preise. The suggested method advances deep learning methodologies for efficient malware detection and categorization. The above paper can include several directions for further research and development. like, Enhanced CNN Architectures: Researchers can investigate more advanced CNN architectures, such as deeper networks or network variations like ResNet, DenseNet, or InceptionNet, to significantly increase the classification of malware's accuracy. Overall, the future scope of this paper lies in exploring advanced techniques, enhancing model performance, addressing emerging challenges, and applying the recommended strategy in practical situations to strengthen the field of malware classification and contribute to improved cybersecurity measures.

References

[1] Vasan, Danish, et al. "Image-Based malware classification using ensemble of CNN architectures (IMCEC)." Computers Security 92 (2020): 101748.

[2] Athiwaratkun, Ben, and Jack W. Stokes. "Malware classification with LSTM and GRU language models and a character-level CNN." 2017 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, 2017.

[3] Habibi, Omar, Mohammed Chemmakha, and Mohamed Lazaar. "Performance Evaluation of CNN and Pre-trained Models for Malware Classification." Arabian Journal for Science and Engineering (2023): 1-15.

[4] Kim, Jeongwoo, Joon-Young Paik, and Eun-Sun Cho. "Attention-Based Cross-Modal CNN Using Non-Disassembled Files for Malware Classification." IEEE Access 11 (2023): 22889-22903.

[5] Dao, Tuan Van, Hiroshi Sato, and Masao Kubo. "MLP-Mixer-Autoencoder: A Lightweight Ensemble Architecture for Malware Classification." Information 14.3 (2023): 167.

[6] Gopinath, M., and Sibi Chakkaravarthy Sethuraman. "A comprehensive survey on deep learning based malware detection techniques." Computer Science Review 47 (2023): 100529.

[7] Brown, Austin, Maanak Gupta, and Mahmoud Abdelsalam. "Automated Machine Learning for Deep Learning based Malware Detection." arXiv preprint arXiv:2303.01679 (2023).

[8] Chaganti, Rajasekhar, Vinayakumar Ravi, and Tuan D. Pham. "A multi-view feature fusion approach for effective malware classification using Deep Learning." Journal of Information Security and Applications 72 (2023): 103402.

[9] Hussain, Abrar, et al. "Malware detection using machine learning algorithms for windows platform." Proceedings of International Conference on Information Technology and Applications: ICITA 2021. Singapore: Springer Nature Singapore, 2022.

[10] Pan, Zhixin, Jennifer Sheldon, and Prabhat Mishra. "Hardware-assisted malware detection and localization using explainable machine learning." IEEE Transactions on Computers 71.12 (2022): 3308-3321

[11] Shatnawi, Ahmed S., Qussai Yassen, and Abdulrahman Yateem. "An android malware detection approach based on static feature analysis using machine learning algorithms." Procedia Computer Science 201 (2022): 653-658.

[12] Sharma, Sagar, Simone Sharma, and Anidhya Athaiya. "Activation functions in neural networks." Towards Data Sci 6.12 (2017): 310-316.

[13] AlGarni, Musaad Darwish, et al. "An efficient convolutional neural network with transfer learning for malware classification." Wireless Communications and Mobile Computing 2022 (2022): 1-8.

[14] Venkatraman, Sitalakshmi, Mamoun Alazab, and R. Vinayakumar. "A hybrid deep learning image-based analysis for effective malware detection." Journal of Information Security and Applications 47 (2019): 377-389.