ENHANCING ADVERSARIAL ROBUSTNESS IN MACHINE LEARNING-BASED MALWARE DETECTION VIA ACTIVATION FUNCTION DESIGN

Chi Duc Luu[1], *Truong Son Pham*[1,*]

Abstract

In recent years, machine learning (ML) has significantly enhanced the efficiency of malware detection systems. Despite achieving high performance, these models now face a growing threat from adversarial attacks. Adversarial malware samples can be intricately crafted to deceive detection models, resulting in misclassifications of malicious programs, thereby allowing them to bypass security systems. Various techniques have been developed to generate adversarial malware specifically designed to evade different ML-based detection systems. This threat underscores the urgent need for solutions that enhance the resilience of malware detection models against adversarial attacks. The paper evaluates and proposes an empirical cost-efficient adversarial defense strategy recommendation via activation function design, that does not require computationally intensive methods such as adversarial training, while boosting the inherent resilience of ML-based malware detection models against black-box attacks. Results show that specific combinations, in particular Rectified Linear Unit (ReLU) and Tanh, can significantly boost robustness without additional training or inference setup. This work provides an empirical design aspect for building intrinsically robust ML-based malware detection systems.

Index terms

Adversarial attacks; adversarial defense; malware detection; activation function engineering; machine learning.

1. Introduction

The rapid evolution of cyber threats has driven the need for more sophisticated and effective malware detection systems. In response, ML has emerged as a transformative force, significantly improving the accuracy and efficiency of modern malware detection frameworks. By analyzing large datasets, ML models can uncover intricate patterns and

[1]Institute of Information and Communication Technology, Le Quy Don Technical University
[*]Corresponding author, email: son.pham@lqdtu.edu.vn
DOI:10.56651/lqdtu.jst.v14.n02.1110.ict

identify potential malware binaries with greater effectiveness than traditional signature-based approaches.

Despite these advancements, adversarial attacks have emerged as a formidable challenge to ML-based malware detectors. Adversarial malware samples are crafted to deceive models by introducing perturbations to executable files, effectively bypassing detection [1]–[3]. Such attacks exploit vulnerabilities in the decision boundaries of classifiers, raising concerns about the robustness of current systems. Studies have shown that adversarial techniques such as byte padding, sections manipulation, and Generative Adversarial Network (GAN)-based malware can significantly reduce detection accuracy, exposing the limitations of ML-based detection models.

Therefore, the need to develop robust malware detection systems, capable of withstanding adversarial attacks, is an urgent research area. While there have been efforts to mitigate this problem, most approaches mainly focus on two categories: adversarial training [4], [5], or ensemble models [6], [7]. Despite the effectiveness of these strategies, they face the challenges of hard to find samples or higher resources costs. Moreover, some synthetic adversarial samples do not retain the functionality of malware, which cannot authentically reflect real-world scenarios. On the other hand, the inherent resilience of malware detection model based on architectural design has not been widely explored. Several studies in the domain of computer vision have suggested that activation functions can have an effect on the accuracy and robustness of ML models [8], [9]. However, this research gap has not been addressed in the field of malware detection.

This research investigates the inherent adaptability of ML-based malware detection models in the face of black-box adversarial attack from a design standpoint. In particular, the hypothesis is that different choices of activation functions can have a profound effect on the innate robustness of ML-based malware detection models. Therefore, the research aims to evaluate the different configurations of activation functions in such models and propose recommendations for an empirical adversarial defense strategy based on activation function engineering. By enhancing robustness at an architecture level, the study contributes by proposing realistic empirical adversarial defense strategy to enhancing the resilience of malware detection systems. Specifically, the contributions of this paper are as follows:

- Proposing an empirical adversarial defense strategy recommendation for ML-based malware detection model through activation function design.
- Conducting extensive experiments to evaluate the proposed approach on two different state-of-the-art models with various designs against two black-box adversarial attacks.
- Results show that the robustness can be significantly enhanced with this strategy while mitigating the need for rare samples acquisition or re-training. From there, empirical recommendations can be made for designing more inherently robust models in real-world scenario.

The paper is organized into the following 7 sections. Section 2 discusses related research in malware detection and adversarial defenses. Section 3 explains the background knowledge. Section 4 describes the motivation of the methodology and Section 5 details experiments setup. The results are evaluated in Section 6. Finally, the paper discusses the conclusion and future directions in Section 7.

2. Related works

While there have been many attempts to address malware detection in other domains such as computer vision [10], the focus on defending the vulnerability of malware detection models against adversarial attacks has been relatively scarce. One popular defense method is adversarial training, in which adversarial samples are used during the training phase [11]–[13]. While these techniques have been able to boost the robustness of ML-based malware models positively, they rely on the procurement and generation of adversarial malware, which might not be available in real-world scenarios. Zhang *et al.* proposed a higher level of adversarial training with GAN generation and filtering of adversarial malware to narrow down the most evasive samples for training [4]. However, GAN-based perturbations do not guarantee an accurate reflection of Portable Executable (PE) malware, making it unideal for realistic situations.

A novel method with non-deterministic architecture was proposed by Wang *et al.* where features are randomly nullified in both the training and testing phases of the model [14]. This approach was able to boost the robustness of the model while maintaining accuracy. However, this work mainly focused on audit log-based features rather than detection of binary files. Another method called randomized smoothing was proposed by Gilbert *et al.* [15]. Through introducing noise for randomizing inputs, a slightly perturbed version or binary files are generated. This boosts the resilience of malware detection models through blurring the decision boundaries, effectively boosting the flexibility of the model during attack. However, randomized smoothing is only effective in certain cases since malware attacks perturb very specific parts of an executable rather than randomly.

Ensemble methods have also been employed as a defense technique for adversarial attacks. Chen *et al.* proposed an ensemble consisting of an adversarial sample detection model and an anomaly detection model [16]. This method implements multiple detection methods to defend against both specific and agnostic adversarial attacks. However, the approach works under the assumption of abundant adversarial samples. Ensila, an ensemble adversarial dynamic behavior detection model, was proposed in [17]. Utilizing dynamic API call sequences, the approach was able to show resilience without the need for re-training. While effective, ensemble methods and dynamic analysis methods can be costly in terms of resources.

3. Background

While the mentioned studies have provided valuable insights into adversarial defense strategies for malware detection, they primarily rely on resource-intensive methods such as adversarial training and ensemble learning. However, limited attention has been given to architectural factors that may inherently influence robustness. To provide the necessary context for the experimental methodology, this section details the foundational components of this research. The paper first introduces the two target state-of-the-art ML-based malware detection models, MalConv and SorelNet, which have achieved high performance for malware detection based on static features without the need for intensive dynamic analysis [18], [19]. However, static features are also easier to modify and bypass, which makes the above models vulnerable to adversarial attacks. Moreover, the two models represent different approaches to ML-based malware detection which enables a more comprehensive view at evaluation. Then, a description of the black-box adversarial attack framework used for experimentation in evaluating the hypothesis is provided. Finally, the section will review the mathematical definitions and properties of the different activation functions that form the basis of the approach.

3.1. MalConv model

For static malware detection, a ML approach called MalConv [20] was proposed that only relies on raw bytes. The model was a pioneering model in malware detection for facilitating a method of classification that does not require feature engineering or domain knowledge. MalConv utilizes a convolutional neural network (CNN) architecture that processes the raw bytes of a PE file as a one-dimensional sequence. This end-to-end design enables the model to automatically learn the special hidden patterns to discern between benign programs and malware.

MalConv begins with a byte embedding layer where each byte is mapped to an 8-dimensional learnable vector. The embedded sequence is then passed to two parallel convolutional layers for feature extraction, whose inputs are activated through ReLU and Sigmoid in the original paper. The outputs are combined via element-wise multiplication. Temporal max-pooling is then conducted so that model can recognize malicious patterns with location invariance. Finally, the resulting vector is passed through two dense layers with ReLU and Sigmoid activation consecutively to output a confidence score ranging between 0 and 1. As such, in the case of the MalConv model, the ReLU activation can be replaced with smoother and more expressive functions for robustness, and the Sigmoid activation can also be mapped to a function that has output interpretable as probabilities, such as the Tanh activation function. The model architecture of MalConv that was utilized is implemented in [21], and is detailed in Figure 1.

3.2. EMBER features-based model

EMBER is originally a malware dataset consisting of static features from millions of PE files [22], which are extracted by utilizing the LIEF project [23], a library

designed to parse and manipulate Executable and Linkable Format, PE, Mach-O formats. The EMBER features are designed with eight groups of raw features consisting of general file information, header information, imported functions, exported functions, section information, byte histogram, byte-entropy histogram, string information, and one additional group of data directories. The final feature vector consists of 2381 raw features.

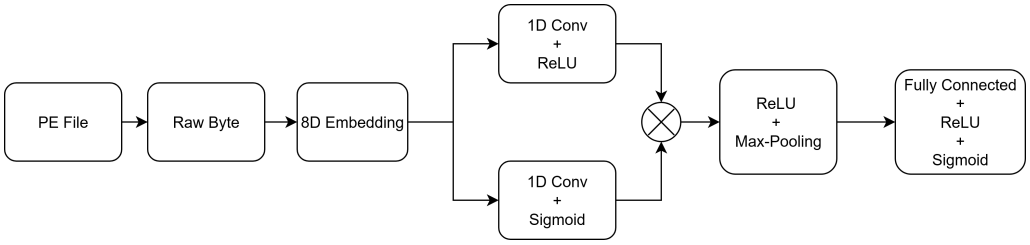

Fig. 1. Model architecture of MalConv.

While the EMBER features set have been used extensively as benchmark, a Deep Neural Network for this approach, called SorelNet, was implemented in the secml-malware framework [21]. This is a smaller model based on the auxiliary loss approach proposed by Sophos in [24]. SorelNet is designed to be a multi-head model with capability for both malware detection and behavior tag labeling. However, the research specifically utilized the malware detection head for binary classification only. As a result, the model functions as a simple feed-forward Deep Neural Network with layers transforming the original EMBER vector of 2381 dimensions to 512, 512, 128 with layernorm, ELU activation, and dropout. The final output layer for malware classification is done with a Sigmoid activation function. In this case, similar to the MalConv model, the hidden layers activations and the output activation can be replaced with different functions. The architecture of the SorelNet with only malware detection head is shown in Figure 2.

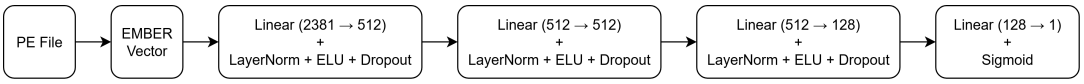

Fig. 2. Model architecture of SorelNet with malware detection head.

3.3. Black-box adversarial attack

A novel functionality-preserving black-box adversarial attack framework was introduced by Demetrio *et al.* in [25]. The framework is able to attack on ML malware detection models by only observing outputs from queries to create adversarial malware without accessing internal weights or structures. This effectively simulates a realistic attack in a black-box setting where only the query of the target model is accessible. The attacks utilize a genetic algorithm to iteratively search for specific modifications that cause misclassifications while maintaining functionality of the original malware.

Two specific attacks were implemented, which are the padding attack and section injection attack. For each attack, contents are extracted from normal programs to act as mutation payloads to simulate benign features optimally. In the padding attack, the additional contents are injected at the end of the file, effectively modifying the overall binary without hindering execution flow. The section injection attack is a more intrusive technique where entirely new sections are injected into the PE file with additional entries being created inside the section table and file contents being shifted to accommodate the change. However, the file functionality is also preserved by keeping alignment with the PE file specifications. The attack flow is visualized in Figure 3.

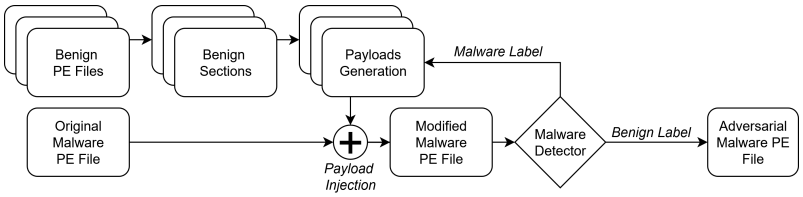

Fig. 3. Black-box adversarial attack framework flow.

3.4. Activation functions

Activation functions introduce non-linearity into neural networks, enabling ML models to learn complex relationships from input data. The choice of activation function can have a profound effect on the learning process as well as expressiveness in neural networks [26]. Therefore, this transformation of input data through each layer of a ML model is one of the main factors for driving the model's final decision. ReLU is a commonly used activation function, denoted by the following equation:

$$\text{ReLU}(x) = max(0, x) \tag{1}$$

For ReLU, with input x, all negative values are mapped to zero while positive values remain unchanged, helping to introduce non-linearity while being computationally efficient. Sigmoid Linear Unit (SiLU) function is another alternative, providing smoother transformations and improving certain models, detailed by the following equation:

$$\text{SiLU}(x) = x * \sigma(x) \tag{2}$$

where, $\sigma(x)$ is the logistic sigmoid. Mish is a self regularized, non-monotonic, and smooth activation function introduced in [27], which has shown to improve upon other standard activation functions even in challenging datasets. The function is as follows:

$$\text{Mish}(x) = x * \text{Tanh}(\text{Softplus}(x)) \tag{3}$$

where, $\text{Softplus}(x) = \log(1 + e^x)$, which is another activation function that is a smooth approximation of ReLU and can be used as a replacement. Tanh, or Hyperbolic Tangent, is an activation function that maps the input to a value in the range of -1 and 1. This can be defined as:

$$\text{Tanh}(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{4}$$

While Tanh is not a probability function, its range mapping characteristic can be used for binary classification interpretation by applying a threshold at 0. Sigmoid, or logistic sigmoid is a widely used for binary probability output, which is as follows:

$$\text{Sigmoid}(x) = \frac{1}{1 + e^{-x}} \tag{5}$$

Sigmoid transform input x into the range between 0 and 1, effectively mapping the model's output into a probability estimate, useful for binary predictions. For this research, all activation functions were implemented using the PyTorch library. A visualized comparison of activation functions are displayed in Figure 4.

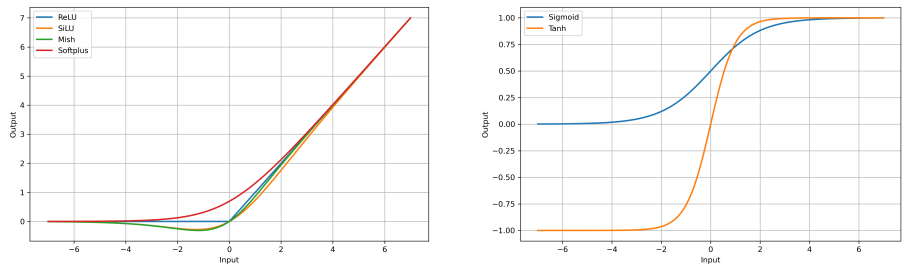

Fig. 4. Comparisons of activation functions.

4. Methodology

For the research, the aim is to enhance the inherent robustness of ML-based malware detection models through designing different activation function combinations and evaluate them in adversarial condition. The paper investigates alternative configurations for both the hidden transformation layers and the output prediction probability layer. Of the different factors that account for a neural network's decision boundary, activation function is one of the key elements influencing the decision boundaries of models through introducing non-linearity. Adversarial attacks work by exploiting the boundaries in these trained models. By introducing perturbations, the samples can be shifted in decision space, causing misclassifications if they are moved in the attacker's intended direction [28]. Many black-box adversarial

malware functions by conducting modifications that emulates features resembling benign programs. As a result, the modified malware moves closer in decision space to fool the ML model [29].

Based on that notion, the paper hypothesizes that activation functions can create varying degrees of robustness due to distinct decision boundaries resulting from the different transformation of data. From a theoretical standpoint, the vulnerability of a malware detection model towards adversarial attacks has strong relations with the curvature and margin of its decision boundaries. By altering the non-linear mappings of the hidden layers and the final prediction layer, the decision regions can be made to be different shapes, which in turn can influence how easily a modified sample can cross from one decision to another.

Activation functions then can be seen as one crucial element in building a inherently robust model. Even small perturbations can transform and accumulate through the model layers, potentially propagating into significant shift in the final probability output [30]. Smoother activation functions for neural network layers, such as Mish or SiLU, can potentially alleviate this by producing more gradual transitions between layers. On the other hand, smoother output functions, such as Tanh, can create a wider probability mapping range which lessen the effect that perturbations have on the final confidence score. As a result, perturbations have less of an effect on the transformed output values and the decisions at the extreme margins can be smoother, increasing robustness.

Therefore, designing the appropriate activation functions can enhance the inherent capability of the detection model even under adversarial conditions. This is ideal for real-world scenarios as modified malware are often rare and not easily provided. The inherent robustness promoted by careful activation function selection may reduce the need for extensive adversarial training or costly data augmentation, both of which often require large collections of crafted adversarial samples that may not exist in practice. Moreover, methods such as ensemble models or distillation also introduce large consumption of resources. Instead, by leveraging activation function choices that naturally improve a model's resistance to adversarial perturbations, the aim is to achieve improved security and generalization with minimal overhead.

5. Experiments

To fully evaluate the effectiveness of this approach, the research conducted comparison experiments on the MalConv model and the SorelNet model described in Section 3. The models comprehensively covers two different approaches of byte-based detection and EMBER features-based detection. For each model, variants were created, with different combinations of activation functions for the hidden layers and the final output layer. For nonlinear mappings of the hidden layers, ReLU, SiLU, Mish, and Softplus are selected as activation functions. For the final confidence output layer, Sigmoid and Tanh was chosen. The range for these layers are also adjusted to [0, 1] and [-1, 1] accordingly, with corresponding threshold of 0.5 and 0.

For the research, the DikeDataset was utilized, which consists of labeled benign and malicious PE files [31]. The dataset contains readily accessible raw binaries which allow for different feature extraction methods, enabling more comprehensive evaluation across the chosen state-of-the-art models. Each model variant was trained on a training dataset of binaries which consists of 1586 samples, with 803 unmodified malicious samples and 783 benign samples. The same settings and training data were used for each model. The training for MalConv was done with learning rate set to 0.0001, batch size set to 64, and 20 epochs. A mean squared error loss and Adam optimizer was used. For the SorelNet, training was done with the learning rate of 0.001, the batch size of 128, and 20 epochs. With the trained model, evaluation is then conducted on an unperturbed test set, which has 396 samples, with 197 malicious samples and 199 benign samples. The paper will refer to this scenario as the original scenario.

In order to generate adversarial samples, the paper utilized the secml-malware framework which includes many implementations of adversarial malware attacks [21]. For the MalConv model, padding and section injection attacks were conducted. For the SorelNet model, due to the EMBER features' natural resilience to padding attack, only section injection attack was conducted. The trained models on unmodified malware and benign programs are then used as targets for the attack model. The attacks are performed with a set limit of 200 queries to the targeted model. Using the same test set as the original scenario, the attacks extract sections from the benign binaries and create payloads to modify every malware programs into adversarial malware. As a result, a set of adversarial malware binaries are generated specifically for each ML model. The final test set for the adversarial scenario consists of 396 samples, with 197 adversarial samples and the same 197 benign samples as the original scenario.

For each test scenario, the standard metrics of Accuracy, Precision, Recall, and F1 score are all calculated for a comprehensive view at performance. The evaluation flow for each model variant is detailed in Figure 5.

6. Results and discussion

6.1. Experimental results

Table 1 shows the impact of different activation function designs on robustness of the MalConv model when under padding attack. While all variants suffer performance degradation when testing with the adversarial test set, there are definite differences in inherent performance and robustness. Comparing the drop in metrics, Mish with Sigmoid and ReLU with Tanh are the combinations with more resilience. Given the attack settings, Mish and Sigmoid is the best and only suffers a 0.159 drop in Accuracy, 0.315 drop in Recall. On the other hand, the combinations of ReLU and Sigmoid, Softplus and Tanh are very vulnerable to attacks as they perform noticeably worse with most adversarial malware bypassing the models, resulting in high false negatives.

For section injection attack on the MalConv model, a similar pattern can be seen in Table 2. Overall, section injection is a stronger attack due to a higher degree of

modification to the PE file. Mish and Sigmoid along with ReLU and Tanh are still the best performing combinations. The worst performance is shown in same model variant as before. Most adversarial malware samples successfully bypass the trained model with ReLU and Sigmoid or with Softplus and Tanh activation functions, resulting in less than 10% for recall values.

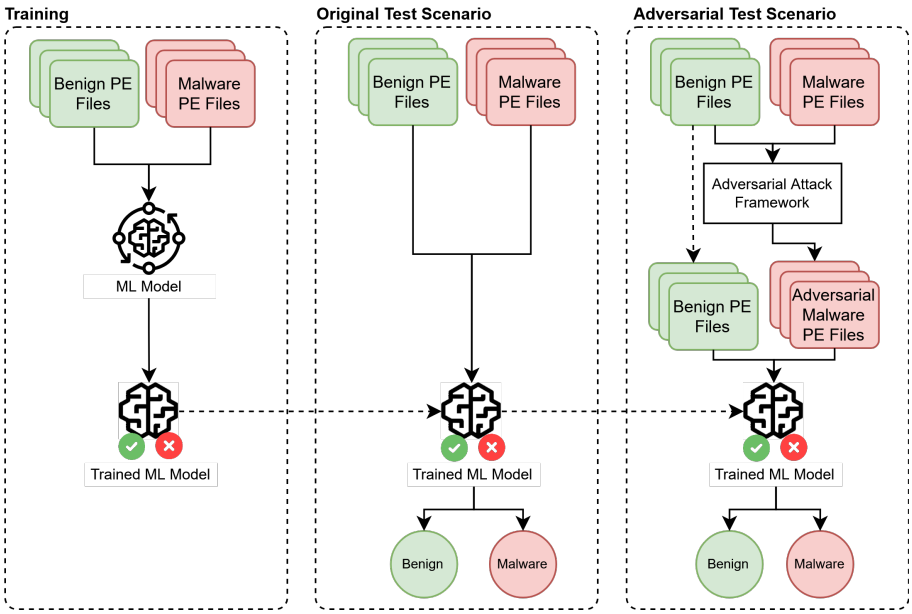

Fig. 5. Experimental evaluation flow.

Table 1. Performance of MalConv with padding attack

Activation functions	Accuracy			Precision			Recall			F1		
	Orig.	Adv.	Drop	Orig.	Adv.	Drop	Orig.	Adv.	Drop	Orig.	Adv.	Drop
ReLU and Sigmoid	0.982	0.599	0.383	0.980	0.918	0.061	0.985	0.228	0.756	0.982	0.366	0.616
SiLU and Sigmoid	0.977	0.676	0.301	0.990	0.973	0.016	0.964	0.371	0.594	0.977	0.537	0.440
Mish and Sigmoid	**0.969**	**0.81**	**0.159**	**0.984**	**0.977**	**0.008**	**0.954**	**0.64**	**0.315**	**0.969**	**0.773**	**0.196**
Softplus and Sigmoid	0.982	0.635	0.347	0.995	0.982	0.012	0.970	0.284	0.685	0.982	0.441	0.541
ReLU and Tanh	**0.974**	**0.766**	**0.208**	**0.984**	**0.973**	**0.011**	**0.964**	**0.553**	**0.411**	**0.974**	**0.706**	**0.269**
SiLU and Tanh	0.969	0.722	0.247	0.984	0.968	0.016	0.954	0.467	0.487	0.969	0.630	0.339
Mish and Tanh	0.949	0.604	0.344	0.984	0.939	0.045	0.914	0.234	0.680	0.947	0.374	0.573
Softplus and Tanh	0.972	0.584	0.388	0.979	0.907	0.072	0.964	0.198	0.766	0.972	0.325	0.647

Scenarios with SorelNet shows an inherent robustness of the EMBER-based features approach as overall performance drops are less extreme in Table 3. In this case, it is evident that Softplus and Tanh is a common vulnerable combination as confidence suffer a biggeer drop under adversarial attack, with recall being only 0.65. For this approach, ReLU and Tanh demonstrates the best performance, with only minimal metrics degradation, showing only a drop of 0.031 in Accuracy and a drop of 0.061 in Recall. However, unlike MalConv, rather than Mish and Sigmoid, SiLU and Tanh is the combination that has the second best resilience, with competitive values compared to ReLU and Tanh.

Table 2. Performance of MalConv with section injection attack

Activation functions	Accuracy			Precision			Recall			F1		
	Orig.	Adv.	Drop	Orig.	Adv.	Drop	Orig.	Adv.	Drop	Orig.	Adv.	Drop
ReLU and Sigmoid	0.982	0.530	0.452	0.980	0.818	0.162	0.985	0.091	0.893	0.982	0.164	0.818
SiLU and Sigmoid	0.977	0.584	0.393	0.990	0.949	0.041	0.964	0.188	0.777	0.977	0.314	0.663
Mish and Sigmoid	**0.969**	**0.684**	**0.285**	**0.984**	**0.963**	**0.022**	**0.954**	**0.391**	**0.563**	**0.969**	**0.556**	**0.413**
Softplus and Sigmoid	0.982	0.548	0.434	0.995	0.957	0.038	0.970	0.112	0.858	0.982	0.200	0.782
ReLU and Tanh	**0.974**	**0.640**	**0.334**	**0.984**	**0.952**	**0.032**	**0.964**	**0.305**	**0.660**	**0.974**	**0.462**	**0.513**
SiLU and Tanh	0.969	0.602	0.368	0.984	0.938	0.047	0.954	0.228	0.726	0.969	0.367	0.602
Mish and Tanh	0.949	0.535	0.414	0.984	0.864	0.120	0.914	0.096	0.817	0.947	0.174	0.774
Softplus and Tanh	0.972	0.519	0.452	0.979	0.778	0.202	0.964	0.071	0.893	0.972	0.130	0.842

Table 3. Performance of SorelNet with section injection attack

Activation functions	Accuracy			Precision			Recall			F1		
	Orig.	Adv.	Drop	Orig.	Adv.	Drop	Orig.	Adv.	Drop	Orig.	Adv.	Drop
ReLU and Sigmoid	0.987	0.915	0.072	0.980	0.977	0.003	0.995	0.853	0.142	0.987	0.911	0.077
SiLU and Sigmoid	0.990	0.848	0.141	0.990	0.986	0.004	0.990	0.711	0.279	0.990	0.826	0.164
Mish and Sigmoid	0.985	0.902	0.082	0.975	0.970	0.005	0.995	0.832	0.162	0.985	0.896	0.089
Softplus and Sigmoid	0.987	0.833	0.154	0.990	0.985	0.005	0.985	0.680	0.305	0.987	0.805	0.182
ReLU and Tanh	**0.987**	**0.956**	**0.031**	**0.990**	**0.989**	**0.001**	**0.985**	**0.924**	**0.061**	**0.987**	**0.955**	**0.032**
SiLU and Tanh	**0.987**	**0.933**	**0.054**	**0.990**	**0.989**	**0.001**	**0.985**	**0.878**	**0.107**	**0.987**	**0.930**	**0.057**
Mish and Tanh	0.987	0.920	0.067	0.990	0.988	0.002	0.985	0.853	0.132	0.987	0.916	0.072
Softplus and Tanh	0.990	0.817	0.172	0.990	0.985	0.005	0.990	0.650	0.340	0.990	0.783	0.207

Generally, it can be seen that while adversarial attacks degrade all performances, varying degrees of robustness are shown throughout different scenarios. For all models and all attacks, it is evident that Softplus as hidden layers activation has the worst resilience and degrades even further when combined with Tanh as confidence output activation. Contrasty, even though ReLU has been evaluated as a vulnerable function, when used in conjunction with Tanh as output activation, displays considerable robustness, achieving highest or second highest in all adversarial scenarios, while sacrificing little original performance. This can be perhaps attributed to the harmonious combination of the two functions to create an effective decision boundary that can withstand perturbed samples without amplifying in the wrong class direction. A balance is created where the sharp distinctive nature of internal representations from ReLU is kept, ensuring discriminative ability, while the larger range of Tanh function helps soften the shifts in output caused by adversarial perturbations.

For the MalConv model, the smoother function of Mish increased the robustness of model the most when used with Sigmoid. However, it is worth noting that the performance on the original unmodified set is also lower compared to other variants. This suggests that smoother function can produce a higher generalization capability but sacrifices sensitivity for discrimination.

6.2. Visualization and interpretability

To better understand how samples are transformed through activations inside each model, the latent representations with PCA across layers in four scenarios were

visualized. For both the MalConv model and SorelNet model, the two combinations of ReLU and Tanh, Softplus and Tanh were chosen as representative of the resilient and vulnerable variants. Figure 6 and Figure 7 display the transformations of the fused convolutional output, the global-pooled output, and the dense hidden encoding of the MalConv model. Through each subsequent transformation, the samples are grouped more distinctly into tight clusters, separating benign and original malware programs. However, it can be seen that, in the weaker variant of Softplus and Tanh, adversarial samples are more prominently pushed to the benign cluster, with low confidence. A more gradual shift is shown with ReLU and Tanh, having adversarial malware samples situated towards the middle of the two distinct class clusters. The confidence scores also have a smoother transition between samples, demonstrating a more robust decision boundary.

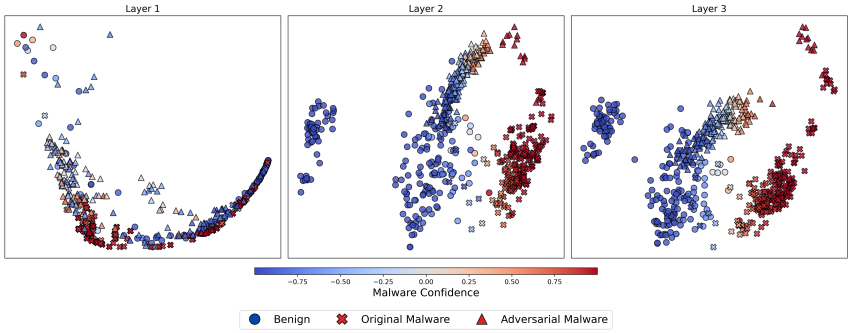

Fig. 6. Latent representation across activation layers in MalConv - ReLU and Tanh.

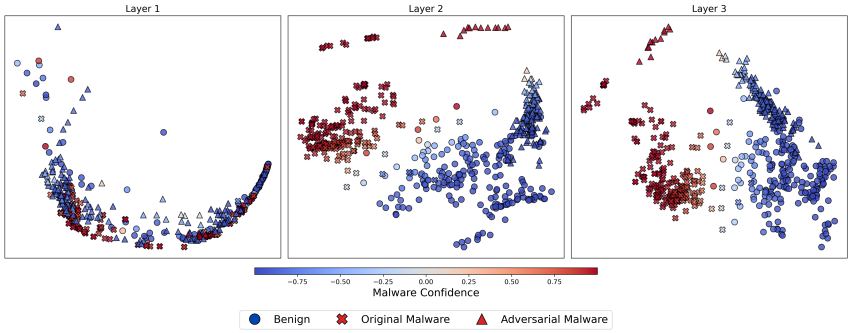

Fig. 7. Latent representation across activation layers in MalConv - Softplus and Tanh.

For the SorelNet model, Figure 8 and Figure 9 visualize the representations of model's output at each layer after activation. As the EMBER features-based approach has a higher resilience as shown through lower performance drops, the clusters are also more discerning in both variants, showing high separation between the two classes. Unlike the MalConv model, the robust ReLU and Tanh variant displays a strong ability to discriminate adversarial malware. Even though the adversarial samples still have a tendency to move towards the middle of the decision space, they are distributed closer towards the malicious cluster, with a distinct classes gap. The

Softplus and Tanh variant has a smoother transition in distribution supported by the bigger drops in metrics value as adversarial samples lie more uncertain at the intersection of the two main clusters.

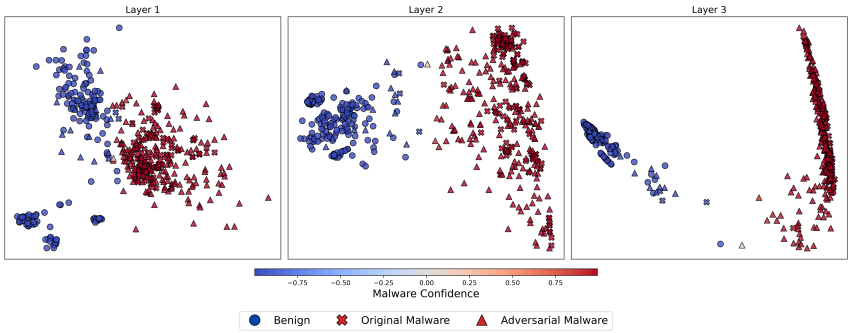

Fig. 8. Latent representation across activation layers in SorelNet - ReLU and Tanh.

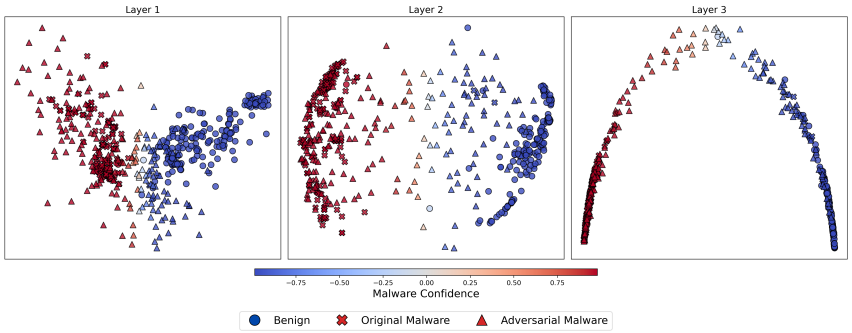

Fig. 9. Latent representation across activation layers in SorelNet - Softplus and Tanh.

7. Conclusion and future work

While alternative approaches for adversarial defense in ML-based malware detection has been explored in various works, the design of activation function is relatively scarce in this domain. Empirical results demonstrate that different activation functions can improve the inherent robustness of malware detection models without the need for adversarial samples or retraining. In particular, ReLU and Tanh, while not achieving the best performance in every cases, showed consistent robustness despite different models and attack scenarios. This finding can serve as a good strategy for designing a resilient ML-based malware detection model. Moreover, EMBER features-based approach was also proven to be more robust under adversarial attacks compared to the CNN byte-based MalConv model.

However, it is important to recognize that even though ReLU and Tanh or Mish and Sigmoid demonstrated better robustness, their original performance is slightly lower compared to some of other vulnerable variants. Despite that, it is believed that with

proper hyperparameters tuning, the performance of these variants can be increased while keeping robustness. There are also some other limitations to consider. First, there are differences for the most robust combination of activation functions between models. In particular, supposedly smoother or more generalizable activation functions did not perform the best when combined together but only increased model's inherent robustness slightly, such as in the case of Mish and Tanh. Deeper investigation into how each activation function interact with each other is needed. Second, the sample pool for the research was limited, facilitating the need for a more comprehensive dataset. Different types of malware and how they are modified for evasion should be examined for a deeper understanding based on domain knowledge. Third, there are other models and adversarial attack types that should be considered.

Therefore, the paper proposes the following future directions. First, further exploration into how samples are transformed at a theoretical level should be considered through more comprehensive experimentation with more models and adversarial attacks. Implementation with explainable ML could also potentially provide deeper insights into the model's decision-making process and improve the understanding of the mechanisms driving robustness. Second, to effectively capitalize on this approach, future research will develop adaptive strategies for selecting the best possible activation functions for both normal and adversarial conditions. Third, extensive experiments with larger and different datasets of modern malware containing diverse categories will be conducted, along with more model types, such as Transformer-based architectures. Moreover, training and inference cost assessment will also allow for a more comprehensive evaluation of the capabilities of different activation functions, which will enhance the understanding about the inherent ability of malware detection models. Additionally, there will be investigation on how different malware structural patterns relate to their evasion rates, which can provide a more comprehensive view at how models process specific malware. Finally, future research will incorporate activation function design with other adversarial defenses to further improve model's robustness. With these future directions, the aim is to advance the practical deployment of resilient, ML-based malware detection systems capable of withstanding evolving adversarial threats.

Acknowledgment

We would like to thank Le Quy Don Technical University for supporting this research. This research is funded by Le Quy Don Technical University under grant number 25.01.62.

References

[1] L. Demetrio, S. E. Coull, B. Biggio, G. Lagorio, A. Armando, and F. Roli, "Adversarial exemples: A survey and experimental evaluation of practical attacks on machine learning for windows malware detection," *ACM Transactions on Privacy and Security*, Vol. 24, No. 4, pp. 1–31, 2021. DOI: 10.1145/3473039

[2] R. L. Castro, C. Schmitt, and G. Dreo, "AIMED: Evolving malware with genetic programming to evade detection," in *18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, Rotorua, New Zealand, 2019, pp. 240–247. DOI: 10.1109/trustcom/bigdatase.2019.00040

[3] L. Demetrio, B. Biggio, G. Lagorio, F. Roli, and A. Armando, "Functionality-preserving black-box optimization of adversarial windows malware," *IEEE Transactions on Information Forensics and Security*, Vol. 16, pp. 3469–3478, 2021. DOI: 10.1109/tifs.2021.3082330

[4] Y. Zhang, H. Li, Y. Zheng, S. Yao, and J. Jiang, "Enhanced DNNs for malware classification with GAN-based adversarial training," *Journal of Computer Virology and Hacking Techniques*, Vol. 17, No. 2, pp. 153–163, 2021. DOI: 10.1007/s11416-021-00378-y

[5] L. Chen, Y. Ye, and T. Bourlai, "Adversarial machine learning in malware detection: Arms race between evasion attack and defense," in *2017 European Intelligence and Security Informatics Conference (EISIC)*, 2017, pp. 99–106. DOI: 10.1109/EISIC.2017.21

[6] S. M. Devine and N. D. Bastian, "An adversarial training based machine learning approach to malware classification under adversarial conditions," in *Hawaii International Conference on System Sciences*, 2021, pp. 1–10. DOI: 10.24251/HICSS.2021.102

[7] C. Smutz and A. Stavrou, "When a tree falls: Using diversity in ensemble classifiers to identify evasion in malware detectors," in *Network and Distributed System Security (NDSS) Symposium*, 2016. DOI: 10.14722/ndss.2016.23078

[8] M. Tavakoli, F. Agostinelli, and P. Baldi, "SPLASH: Learnable activation functions for improving accuracy and adversarial robustness," *Neural Networks*, Vol. 140, pp. 1–12, 2021. DOI: 10.1016/j.neunet.2021.02.023

[9] C. Xie, M. Tan, B. Gong, A. Yuille, and Q. V. Le, "Revisiting activation function design for improving adversarial robustness at scale," in *The Eleventh International Conference on Learning Representations*, 2023.

[10] N. Akhtar, A. Mian, N. Kardan, and M. Shah, "Advances in adversarial attacks and defenses in computer vision: A survey," *IEEE Access*, Vol. 9, pp. 155 161–155 196, 2021. DOI: 10.1109/ACCESS.2021.3127960

[11] K. Lucas, S. Pai, W. Lin, L. Bauer, M. K. Reiter, and M. Sharif, "Adversarial training for raw-binary malware classifiers," in *32nd USENIX Security Symposium (USENIX security 23)*, 2023, pp. 1163–1180.

[12] M. Imran, A. Appice, and D. Malerba, "Evaluating realistic adversarial attacks against machine learning models for windows pe malware detection," *Future Internet*, Vol. 16, No. 5, pp. 168, 2024. DOI: 10.3390/fi16050168

[13] K. Shaukat, S. Luo, and V. Varadharajan, "A novel method for improving the robustness of deep learning-based malware detectors against adversarial attacks," *Engineering Applications of Artificial Intelligence*, Vol. 116, Nov. 2022. DOI: 10.1016/j.engappai.2022.105461

[14] Q. Wang, W. Guo, K. Zhang, A. G. Ororbia, X. Xing, X. Liu, and C. L. Giles, "Adversary resistant deep neural networks with an application to malware detection," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 1145–1153. DOI: 10.1145/3097983.3098158

[15] D. Gibert, G. Zizzo, Q. Le, and J. Planes, "Adversarial robustness of deep learning-based malware detectors via (de)randomized smoothing," *IEEE Access*, Vol. 12, pp. 61 152–61 162, 2024. DOI: 10.1109/ACCESS.2024.3392391

[16] J. Chen, C. Yuan, J. Li, D. Tian, R. Ma, and X. Jia, "ELAMD: An ensemble learning framework for adversarial malware defense," *Journal of Information Security and Applications*, Vol. 75, 2023. DOI: 10.1016/j.jisa.2023.103508

[17] C. Jing, Y. Wu, and C. Cui, "Ensemble dynamic behavior detection method for adversarial malware," *Future Generation Computer Systems*, Vol. 130, pp. 193–206, 2022. DOI: 10.1016/j.future.2021.12.013

[18] K. Aryal, M. Gupta, M. Abdelsalam, and M. Saleh, "Explainability guided adversarial evasion attacks on malware detectors," in *33rd International Conference on Computer Communications and Networks (ICCCN)*, 2024, pp. 1–9. DOI: 10.1109/ICCCN61486.2024.10637577

[19] A. Brown, M. Gupta, and M. Abdelsalam, "Automated machine learning for deep learning based malware detection," *Computers & Security*, Vol. 137, 2024. DOI: 10.1016/j.cose.2023.103582

[20] E. Raff, J. Barker, J. Sylvester, R. Brandon, B. Catanzaro, and C. Nicholas, "Malware detection by eating a whole EXE," in *The Association for the Advancement of Artificial Intelligence Workshop*, 2018. DOI: 10.48550/ARXIV.1710.09435

[21] L. Demetrio and B. Biggio, "secml-malware: A python library for adversarial robustness evaluation of windows malware classifiers," *arXiv:2104.12848*, 2021. DOI: 10.48550/arXiv.2104.12848

[22] H. S. Anderson and P. Roth, "EMBER: An open dataset for training static PE malware machine learning models," *arXiv e-prints*, 2018. DOI: 10.48550/arXiv.1804.04637

[23] R. Thomas, "LIEF - Library to instrument executable formats," Apr 2017. [Online]. Available: https://lief.quarkslab.com/

[24] E. M. Rudd, F. N. Ducau, C. Wild, K. Berlin, and R. Harang, "ALOHA: Auxiliary loss optimization for hypothesis augmentation," in *28th USENIX Security Symposium (USENIX Security 19)*, Santa Clara, USA, 2019, pp. 303–320. DOI: 10.48550/arXiv.1903.05700

[25] L. Demetrio, B. Biggio, G. Lagorio, F. Roli, and A. Armando, "Functionality-preserving black-box optimization of adversarial windows malware," *IEEE Transactions on Information Forensics and Security*, Vol. 16, pp. 3469–3478, 2021. DOI: 10.1109/TIFS.2021.3082330

[26] F. Agostinelli, M. Hoffman, P. Sadowski, and P. Baldi, "Learning activation functions to improve deep neural networks," *arXiv preprint arXiv:1412.6830*, 2014. DOI: 10.48550/arXiv.1412.6830

[27] D. Misra, "Mish: A self regularized non-monotonic activation function," *arXiv preprint arXiv:1908.08681*, 2019. DOI: 10.48550/arXiv.1908.08681

[28] W. Brendel, J. Rauber, and M. Bethge, "Decision-based adversarial attacks: Reliable attacks against black-box machine learning models," *arXiv preprint arXiv:1712.04248*, 2017. DOI: 10.48550/arXiv.1712.04248

[29] F. Wang, X. Zuo, H. Huang, and G. Chen, "ADBA: Approximation decision boundary approach for black-box adversarial attacks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39, No. 7, 2025, pp. 7628–7636. DOI: 10.1609/aaai.v39i7.32821

[30] V. Zantedeschi, M.-I. Nicolae, and A. Rawat, "Efficient defenses against adversarial attacks," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 2017, pp. 39–49. DOI: 10.1145/3128572.3140449

[31] iosifache, "Dikedataset." [Online]. Available: https://github.com/iosifache/DikeDataset

Manuscript received 19-08-2025; Accepted 19-12-2025.

■

Chi Duc Luu has received the Bachelor degree in Computer Science from National Defense Academy, Japan, and the Master degree in Forensic Computing and Cybercrime Investigation from University College Dublin, Ireland. Currently, he is a researcher at the Cybersecurity Center of Excellence, Institute of Information and Communication Technology, Le Quy Don Technical University. Current research interests: intrusion detection, malware detection. E-mail: duc.luu@lqdtu.edu.vn.

Truong Son Pham has received his Doctor of Philosophy in Computer Science from the Bundeswehr University Munich, Germany, where he also completed a Post-Doctoral position in the domain of Cybersecurity. He is currently the vice director of the Cybersecurity Center of Excellence, Le Quy Don Technical University, Vietnam. His research focuses on cybersecurity, digital forensics, malware analysis, and machine learning. He has established strong national and international collaborations and serves as a reviewer for several academic journals and conferences. E-mail: son.pham@lqdtu.edu.vn

TĂNG CƯỜNG KHẢ NĂNG CHỐNG CHỊU CỦA MÔ HÌNH HỌC MÁY CHO PHÁT HIỆN MÃ ĐỘC THÔNG QUA THIẾT KẾ HÀM KÍCH HOẠT

Lưu Chí Đức, *Phạm Trường Sơn*

Tóm tắt

Trong những năm gần đây, học máy đã nâng cao đáng kể hiệu năng của các hệ thống phát hiện mã độc. Tuy đã đạt được hiệu suất cao, các mô hình này hiện đang đối mặt với mối đe dọa ngày càng tăng từ các cuộc tấn công đối kháng. Các mẫu mã độc đối kháng có thể được tạo ra một cách tinh vi nhằm đánh lừa các mô hình phát hiện, dẫn đến việc phân loại sai các chương trình độc hại, cho phép chúng vượt qua các hệ thống bảo mật. Nhiều kỹ thuật khác nhau đã được phát triển để tạo ra mã độc đối kháng, nhằm né tránh các loại hệ thống phát hiện dựa trên học máy khác nhau. Mối đe dọa này nhấn mạnh nhu cầu cấp thiết về các giải pháp giúp tăng cường khả năng chống chịu của các mô hình phát hiện mã độc trước các cuộc tấn công đối kháng. Bài báo đánh giá và đề xuất một chiến lược phòng thủ chống tấn công đối kháng dựa trên thiết kế hàm kích hoạt, không yêu cầu các phương pháp tính toán chuyên sâu như huấn luyện đối kháng, đồng thời tăng cường khả năng chống chịu tự nhiên của các mô hình học máy cho phát hiện mã độc trước các cuộc tấn công hộp đen. Kết quả cho thấy rằng các tổ hợp hàm kích hoạt cụ thể, đặc biệt là ReLU và Tanh, có thể cải thiện đáng kể khả năng chống chịu của mô hình mà không cần bổ sung thêm tác vụ huấn luyện hoặc cấu hình suy luận. Nghiên cứu này cung cấp một góc nhìn thiết kế thực nghiệm để xây dựng các hệ thống phát hiện mã độc sử dụng học máy với khả năng chống chịu nội tại cao.

Từ khóa

Tấn công đối kháng; phòng thủ đối kháng; phát hiện mã độc; thiết kế hàm kích hoạt; học máy.