VOL. 19, NO. 4, FEBRUARY 2024 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

ENHANCING RANSOMWARE DETECTION USING SIAMESE NETWORK

Muhammad Khamis Dauda, Iftikhar Ahmad and Madini O. Alassafi
Department of Information Technology, Faculty of Computing and Information Technology
King Abdulaziz University (KAU), Jeddah Saudia Arabia
E-Mail: mdauda0002@stu.kau.edu.sa

ABSTRACT

Organizations in the current digital era are exposed to a variety of cybersecurity threats that can often result in financial losses and harm to their reputation. Among these threats, ransomware attacks can cause significant damage. Attackers are constantly improving their techniques to bypass security channels, which makes it challenging to monitor and detect the patterns of attacks. Consequently, there is a growing inclination towards employing state-of-the-art techniques to identify and defend during ransomware attacks. Deep learning is a proven technique that can be employed to learn from large complex patterns. However, large datasets are required in the training of deep learning models which is a challenging task. Few-shot learning (FSL) overcomes this limitation by using less data. In this research work, a Siamese network design is developed by incorporating the architectural principles of AlexNet and features of the VGG configuration. The employed methodology enables us to evaluate the inherent resemblances and disparities in the data. This novel methodology demonstrated exceptional performance, with an average accuracy of 97% when compared to various effects and learning rates. The results of the presented study demonstrate the capacity to greatly enhance cybersecurity by providing a scalable and effective approach for detecting ransomware.

Keywords: ransomware detection, siamese networks, AlexNet, VGG, metric learning model.

Manuscript Received 19 March 2024; Revised 29 March 2024; Published 15 April 2024

1. INTRODUCTION

In 2022, there was a concerning surge in ransomware assaults inside the digital security domain. According to studies, over 70% of organizations experienced cyber intrusions, marking a significant increase when compared to the previous five years statistics [1]. This phenomenon not only underscores a substantial economic strain on enterprises but also engenders apprehensions over prospective security breaches. This serves as a reminder of the need to implement more efficient strategies to mitigate and identify potential risks to information systems.

Numerous strategies and approaches have been investigated by researchers and cybersecurity specialists to enhance the capabilities of ransomware detection. The methodologies encompass dynamic, behavioral, and static approaches that employ machine learning, deep learning, and hybrid models. While machine learning is effective in classifying tasks with lower complexity and deep learning is effective in finding patterns and behavior, it is important to note that these models may exhibit underfitting and limited generalization when faced with little data. Hence, within this particular context, several learning approaches are examined in the subsequent sections.

Few-shot learning is a nascent methodology that emphasizes the capacity of models to acquire and leverage knowledge from a limited amount of data, sometimes referred to as "shots." This approach addresses the challenges associated with acquiring knowledge from limited data and incorporates methodologies such as transfer learning and meta-learning [2].

Transfer Learning is a subfield within the realm of deep learning that leverages the acquired knowledge from addressing a particular problem to address another problem that is closely related. This strategy is beneficial in situations where there is a scarcity or high cost of data available for the specific problem at hand. As an illustration, a model that has undergone training using an extensive dataset of photos can acquire characteristics, including edges, forms, and textures, which can subsequently be utilized in diverse image identification endeavors. By implementing this approach, the training procedure can be expedited, leading to enhanced model performance in situations where the availability of target data is limited.

Meta-learning, sometimes known as "learning to learn", is a widely recognized technique. The primary aim of this endeavor is to enhance the capacity of machine learning models to acquire knowledge rapidly and effectively through experiential learning [3]. This is accomplished by training the model on a diverse range of activities, enabling it to effectively apply its knowledge to novel and unfamiliar tasks using limited data.

The primary objective of metric learning is to acquire knowledge about the similarities between pairs of inputs within a dual network, often a Siamese network, by utilizing a distance function [4].

The structure of this article is being highlighted further. Section 2 covers a thorough examination of the existing methods for identifying ransomware, provided through a comprehensive literature review followed by a gap analysis. In section 3, we comprehensively elaborate the analysis of the accessible dataset, and the method of data preprocessing, encompassing the underlying transformation logic. The novel model is proposed with a blend of techniques, offering a thorough exposition of its developmental procedure. The implementation of technical attributes and factors have also been taken into

account, along with the conversion of the conceptual framework into a functional model. The section further shows the findings of the performance evaluation of the proposed model that are presented in the testing and validation sections. Subsequently, section 4 encompasses the discussion on results that display significant insights through the analysis of the model and its outcomes. Moreover, it facilitates a comparative evaluation of its performance. Finally, we conclude the study in section 5 by effectively summarising the significant contributions and implications of this study. We offer valuable perspectives on the wider importance of the work and suggest possible areas for future investigation, so adding to the continuing scholarly conversation on the subject.

2. LITERATURE REVIEW

To learn and understand patterns, there are three common ways in machine learning namely, supervised learning, semi-supervised learning, and unsupervised learning. Supervised learning models learn from labeled samples and attempt to predict unlabeled samples. These models are evaluated based on their ability to accurately predict the output and input and are widely used for classification and regression tasks [5].

Semi-supervised learning, as the name implies, is used when labeled data is expensive or difficult to obtain. In this approach, a small amount of labeled data is used to help categorize a large amount of unlabeled data [6]. Unlike the first two models, unsupervised learning is used to capture the intrinsic structure and underlying common patterns in a given dataset. This method groups data based on a particular property, without the use of labeled data [7].

With the increasing frequency of cyber security attacks, there is a growing demand for intelligent models to mitigate malicious program proliferation. To address this, several artificial intelligence models have been proposed, including machine learning, deep learning, and hybrid models. In cyber-attack prevention and mitigation, machine learning models are widely adopted for their effectiveness and promising results [8]. Parkar and Bilimoria [9] highlighted the trends of effective machine-learning models and techniques in detecting ransomware. Sharma *et al*., [10] discussed the limitations of supervised learning, which heavily relies on anti-virus vendors to provide explicit labels. They proposed a clustering-based unsupervised machine learning approach that addresses the mislabeling of targets and detecting unknowns.

Another approach proposed by Khammas [11], uses frequent pattern mining to extract features from the raw byte. They claim an increase in the detection speed. Hwang *et al*., [12] used random forest and a model called Markov to produce a pipelining model on Windows API calls to capture the behavior of the ransomware.

Yilmaz *et al*. [13] conducted a study on a single variant of ransomware that causes a screen splash in a controlled environment with 538 participants. The aim was to measure the adaptability of best practices. Faghihi and Zulkernine [14] studied ransomware related to smartphones and proposed decrypting tools capable of recovering lost data without compromising privacy.

Sreelaja N. K. [15], proposed a signature-based model that uses fuzzy hashing and ClamAV methods to detect ransomware. However, the effectiveness of this technique is limited when dealing with novel attacks. For the dynamic approach, Ramesh and Menen [16] proposed a method to monitor changes in resource utilization, persistence, and lateral movement using a finite state machine learning model to identify ransomware attacks.

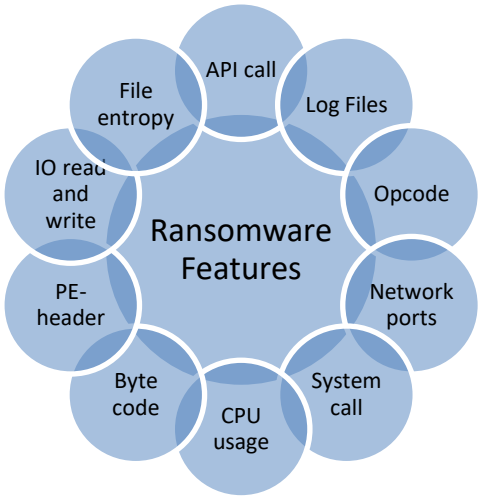


Figure-1. Ransomware features utilized in the existing techniques.

There are various hybrid techniques for machine learning, such as the double-layered hybrid approach which combines the Naive Bayes and SVM classifiers. A related approach is developed by Wisanwanichthan and Thammawichai [17] to capture underlying patterns effectively. Another study by Mercaldo [18] explored API calls for ransomware threats using a hybrid approach that combined static and dynamic analysis. Jung and Won [19] stated that file entropy is an effective feature value that can help differentiate between ransomware and benign wares. The features of ransomware behavior that were used for detection purposes are illustrated in Figure-1.

2.1 Gap Analysis

Deep learning models often use convolutional neural networks (CNNs) as a base network to which other techniques can be added for classification. In [20], deep learning trends, meta-learning implementations, and challenges are discussed, along with possible ways to mitigate them. Mascarenhas and Agarwal [21] provide a concise review of several implementations of CNNs, ranging from AlexNet [22] and its variants, to VGG [23] and its variants, RestNet [24], and Inception models [25]. These models are often used for transfer learning due to their effectiveness in image recognition. However, the data needs to be transformed and converted to suit such models.

The idea of converting one-dimensional data into two-dimensional data to detect and classify malicious

content using convolutional neural networks has been explored in various research studies such as [26], [27], [28] and [29]. These studies used deep learning models and transfer learning techniques like freezing and fine-tuning. In this paper, the same approach is adopted to improve detection accuracy and bridge the gap between an ineffective model by providing an effective one.

Cui *et al*., [30] utilized the conversion of malicious code into grayscale and processed it using CNN images. Siamese Networks are a type of deep learning model that excels at metric learning and can learn to measure similarity between pairs of inputs. The architecture of Siamese Networks consists of two identical subnetworks that share weights and reduce bias. This architecture is incredibly useful in various applications, including differentiating data, making it a valuable tool in the presented case. Table 1 summarizes some literature on the state-of-the-art and its limitations.

Table-1. Literature review works with their limitations.

Reference	Proposed Method	Added value	Accuracy	Limitations
[11]	A simple static analysis as opposed to the complex dynamic analysis of ransomware detection	Feature extraction from raw bytes using frequent pattern mining to improve detection speed. Gain ratio technique used for feature selection.	97.74%	Static detection has several limitations, resulting in higher False Positive Rate (FPR) and False Negative Rate (FNR). This can cause a significant problem as it can negatively impact the efficiency of any algorithm. The paper lacks clarity on several aspects such as the absence of a confusion matrix or the rate of FP and FN. Additionally; it is unclear how the proposed paper tackles the challenge of an unseen dataset or attack.
[12]	A two-stage mixed ransomware detection model, Markov model, and Random Forest model	A sequential characteristic of Windows APIs by building Markov models. Statistical machine learning techniques using Windows API call. Contribute to a mixed two-stage detection method with a strong focus on controlling false negative error rates (FNR) under nominal control of false positive error rates (FPR). The dataset used is from http://virusshare.com and http://en.softonic.com.	97.3%	The accuracy needs to be improved. Using deep learning techniques, accuracy might be improved
[31]	CNN-based detection.	CNN out-performed SVM in Windows OS using: Contagio, Open Malware, VirusTotal Datasets	87.9% on Windows 59.0% of mobile	The model does not help in detecting or preventing the system directly but rather an algorithm is used to prevent the propagation of infection from one system to another. This tool may be helpful in distributed systems and connected peer-to-peer systems or the transfer of information using USB and other file transfer hardware.
[32]	Process behavior analysis using RF	RF model was proposed to detect zero-day attacks. Built the deataset.	75%	The accuracy indicates that there is still room for improvement.
[33]	Deep learning-based detector (DeepRan) Using BiLSTM-FC in an enterprise network using host logging data	A proactive detection method was proposed to detect ransomware attacks before these are fully deployed on the victim host using an experimental testbed for host log data collection from bare metal servers.	99.87%	The evaluation of accuracy is mainly employed on 2 user data. Experimentation on the public datasets is needed because it shows the true real-world attack.
[34]	Customized deep contractive autoencoder-based attribute learning (DCAEZSL) for zero-day ransomware detection	By increasing the penalty term in the proposed model's loss function, it is possible to separate known and unknown ransomware using an unsupervised approach.	92.8%	The proposed algorithm is based on one type of file extension to classify it as either a good-ware or a ransomware. Other ransomware classes and families may not be detected.
[35]	Ransomware detection and mitigation using Shannon entropy and Fuzzy hash principles	Using the experimental testbed, the proposed technique achieved a higher detection rate on the Windows operating system by using the VirusTotal and Virus Share APIs	95%	Not a proactive measure, the user is notified after the encryption happens. A proactive approach is deemed necessary.
[36]	A domain-specific ransomware analysis in a Wireshark packet analysis tool.	Using a lightweight approach to analyze, identify, and track dynamic ransomware behavior during runtime to uncover ransomware, and network threats and detect infection using minimum effort.	-	It is a manual process that needs a lot of effort, expertise, and understanding of the Wireshark tool. No machine learning is used.

An Extensive review of the literature depicts the implementation of behavioral and dynamic features that are effective in thwarting ransomware attacks. However, there is a significant scope for refinement in creating accurate and efficient models. To address this, we have derived inspiration from the concepts presented in [26], [27], [28] and [29].

The proposed solution aims to extract and transform 1D data into 2D data and employ a Siamese net to identify and differentiate distinct behaviors, enhancing the performance and accuracy of ransomware detection.

3. METHODOLOGY

This section explains the selected dataset, the materials, the selected model, and the methods of training, testing, and validation steps applied to achieve the objectives of this work. Figure-2 illustrates the methodology steps.

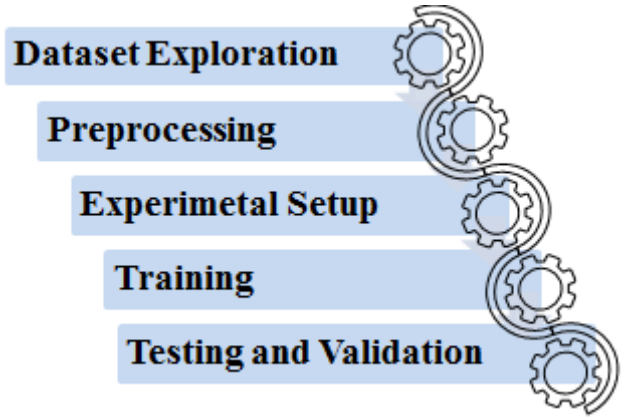


Figure-2. Proposed methodology.

3.1 Dataset Exploration

The dataset is a very crucial part of the presented analysis. The quality of data influences the model performance and results. We explored two malware repositories namely, 1) Virus Share and 2) Virus Total. These repositories offer unique insights and opportunities to researchers and security personnel in tackling cyber threats. Virus Share has an extensive collection of malware families, enabling researchers to gain a better understanding of imaging cyber threats. On the contrary, Virus Total is a collaborative platform, as shown in Figure-2. It fosters proactive threat intelligence gathering and combines the collective expertise of security professionals, researchers, and independent contributors worldwide, providing a comprehensive view of potential threats by consolidating metadata and analysis results from multiple antivirus engines and other security tools.

Although these repositories can be helpful in threat intelligence and cybersecurity analysis, it is important to note that these may not be suitable for analyzing ransomware. Ransomware is a specific type of malware that requires specialized techniques and tools for analysis. Therefore, we have to look into other public repositories to search for a ransomware dataset that meets the requirements.

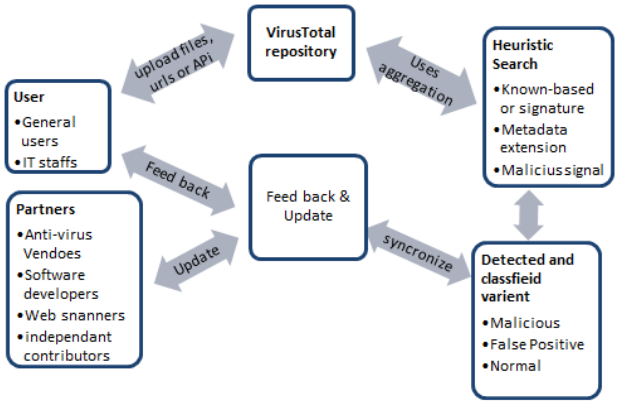


Figure-3. Workflow of virus total repository.

3.2 The Ransap Dataset

Based on the idea by Arabo *et al.*, [32] where byte read and write, memory usage, API calls, and read & write count provide important features and contain significant insight for ransomware analysis, we found a public dataset called Ransap (Ransomware Access Pattern dataset) [37]. It is an open-access dataset that focuses on capturing the dynamic behavior of ransomware, specifically the read and write operations, which are crucial for understanding the precise difference between ransomware and goodware behavior, to facilitate the training and evaluating of machine learning models to combat the cyber threat. The dataset contains seven prominent ransomware specimens, including those offered as a service formally known as RaaS, along with 5 goodware samples. Table-2 enlists Ransap dataset features and Table 3 displays a comparison for the explored dataset and respective domain.

Table-2. Ransap dataset features.

The Read Operation	• Time in seconds • Time in nanoseconds • Logical block address (LBA) • Size of the operation (in bytes)
The Write Operation	• Time in seconds • Time in nanoseconds • Size of the operation (in bytes) • Entropy 1 (every block or 4096 bytes) • Entropy 2 (sum of eight entropy values for every 256)

Table-3. Comparison table of selected dataset and other popular respository.

	Virus Share	Virus Total	RanSap
Availability	Yes	Yes	Yes
Accessibility	Close	Close	Open
Ransomware Only	No	No	Yes
Type of threat	Malware	Intrusion detection	Ransomware
Primary Purposed	Research	General purpose	Research

3.3 Preprocessing

Data conversion

We initiated this phase by extracting the read and write access from the dataset which is a crucial step in crafting an effective model. This was identified as an effective feature and key indicator that encapsulates the true behavior and actions of both benign and ransomware entities.

However, raw data rarely aligns perfectly with the proposed model's architecture. To bridge this gap, the extracted features underwent a metamorphosis (Figure-4 illustrates), transitioning into an image format. This transformation prepared the data for the Siamese network's image-based processing pipeline.

Figure-4. Data preprocessing steps.

Converting to bytecode is crucial in terms of precision, and integrity of the entries, keeping such in the bytecode ensures capturing the exact behavior of each application and also reduces the need for a huge number of entries in a human-readable format (see Figure-5).

Algorithm 1: Conversion of Read and Write Entries to Bytecode

Input: Filename of the read and write csv file
Output: Bytecode representation of the read and write data

Initialize an empty list `data` to store the CSV entries.
Open the CSV file specified by `filename` for reading.
For each row in the CSV file:
Read the row.
Append the row to the `data` list.
Close the CSV file.
Process the data in `data'
For each row in `data:`
For each entry in the row:
If the entry is numeric:
convert it to an integer.
Else:
keep the entry as is.
Encode the `data` into bytecode.
Serialize `data` into a binary format using a suitable serialization method.
Save the generated bytecode.
End.

The serialization (bytecode) depends on domain requirements, which can include interoperability, and the complexity of the converted data, therefore in this case we used Pickle to serialize the data into a binary format. Algorithms 1 and 2 elaborate the steps of conversion from dataset to images as shown in Table-4.

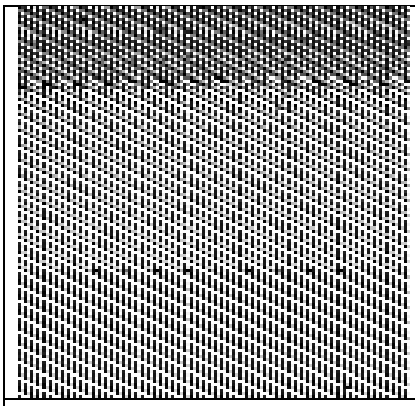
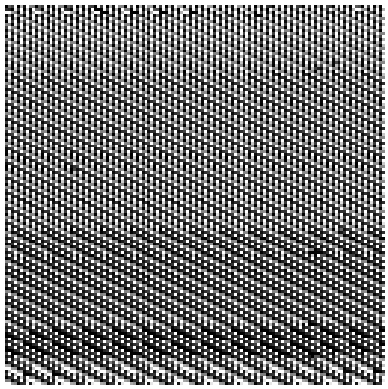
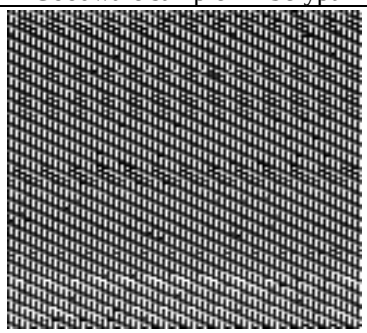
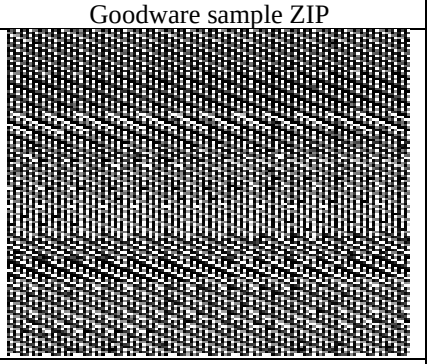
Algorithm 2: Create Image from Bytecode

Input: **Bytecode File**
Output: **Colored Image File**
Deserialize the Bytecode:
Open the file containing the serialized data in binary read mode.
Use a deserialization method to reconstruct the original data from the bytecode.
Determine Image Dimensions:
Determine the height of the image from the number of rows in the deserialized data.
Determine the width of the image from the number of elements in the first row of the deserialized data.
Initialize the Image size:
Create a new image with the determined width and height, or using a pre-predefine size.
Populate the Image with pixeled values:
Iterate over each row and each element within that row in the deserialized data.
For each element:
Determine the pixel value based on the element.
in the image to its value.

Save the Image:
End

The algorithm depicted in Figure-6 illustrates the procedure of transforming the data into an image format for the proposed model. It specifically emphasizes the creation of a colored image by associating data elements

with RGB color values ranging from 0 to 255. This concept was initially introduced in references [17], [18], [19]. The critical step lies in defining the data mapping to RGB values. Table-4 presents sample images of ransomware and benign software after the conversion process.

	
Goodware sample AEScrypt	Goodware sample ZIP
	
Ransomware sample Cerber	Ransomware sample wannacry

3.4 Experimental Setup

Because of the limitations of the size of the dataset and computational resources, we chose the AlexNet [22] architecture, due to its proven efficiency. It is notable for its streamlined structure with fewer layers and varying filter sizes in its convolutional stages. This decision was based on the need for a model that can deliver good performance while being computationally efficient. AlexNet's design inherently requires less computational power as compared to the more demanding VGG [21] models, which use a consistent 3x3 filter approach across much deeper layers, resulting in a substantial increase in the number of parameters to train.

The proposed model was customized for the research-specific requirements while leveraging the strengths of both architectures. We combined the architectural principles of AlexNet with aspects of the VGG setup, using a simplified convolutional layer design. This hybrid model includes convolutional layers with varying filter numbers, 16, 32, and then 16 again, but maintains a constant kernel size of 3x3, resembling VGG's consistency in filter dimensions. It also incorporates 2x2 max pooling to effectively reduce spatial dimensions. This strategic implementation is to achieve a careful balance between computational efficiency and to ensure that the model is both practical and efficient for the task.

Figure-4 Illustrates the proposed model architecture, showing the use of multiple convolutional layers with varying numbers of filters, followed by max pooling.

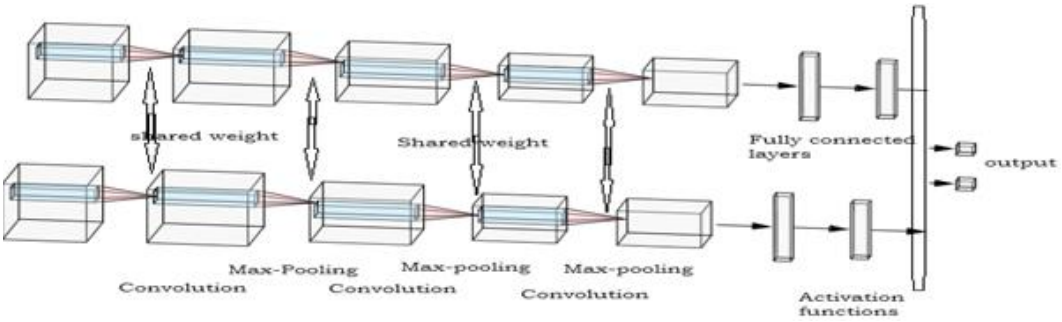


Figure-5. The architecture of the proposed model.

It is a Siamese neural network model, designed to learn from the paired network to assess the input data's similarity and dissimilarity. The model uses a shared architecture, which is implemented and named as a base network, to process each input in the pair. This approach ensured that the same weights and computational logic apply to both inputs, which is crucial for learning a consistent representation of the inputs regardless of their order in the pair.

The transition from max-pooling layers to dense layers marks a shift from feature extraction to decision-making processes, where features are projected and viewed in 2d space to measure the similarities.

A Euclidean distance function is used to quantify the similarity or dissimilarity of the resultant vectors, images were classified based on their proximity to either the ransomware or benign range. Equation 1 shows the Euclidean distance function for computing the distance between classes in the feature space.

$$d(r,g) = \sqrt{\sum_{i=1}^{n}(r_i - g_i)^2} \quad (1)$$

Where:

d(r,g) is the Euclidean distance between goodware b and ransomware r.
r_i,...r_n and g_i… g_(n) are projected features points in features space.

Specifically, if an image's Euclidean distance aligns closely with the ransomware range, it is classified as ransomware; conversely, if it aligns with the benign range, it is categorized as benign. Then contrastive function, which penalizes the model for inaccurate predictions based on the Euclidean distance is used as the loss function, as shown in equation 2.

$$L(y,d) = (1-y)\frac{1}{2}(d)^2 + (y)\frac{1}{2}max(0, m-d)^2 \quad (2)$$

Where:
$L(y,d)$ is the contrastive loss for the pair.
y is the euclidean distance value computed.
m is the margin, a value between the none similar pair
$(1-y)\frac{1}{2}(d)^2$ for the similar pair to be closer as much as possible.
$(y)\frac{1}{2}\max(0, m-d)^2$ for the none similar pair to be as far as possible.

3.5 Training

The training begins with two input layers, input a and input b which intend to be designed to receive paired inputs. These inputs are both processed by an instance of the base network. The base network, as described in the implementation section, is dual arm network that ensures the sharing of weight across both arms, allowing the model to learn the pattern and update its loss in a recurve manner.

The sequential layering of convolutional layers and pooling layers enables a step-by-step learning process, understanding the hierarchical features from basic edges, and textures, to intricate patterns. Moreover, network training spans 100 epochs with an optimal learning rate of 0.0005 facilitated by the Adam optimizer, resulting in the commendable convergence of model parameters. This optimization strategy produced a model characterized by 417,410 trainable parameters, a testament to its streamlined and effective design. Despite the relatively modest parameter count, the network displayed significant training accuracy and training loss improvements, as shown in Figures 5 and 6. This highlights the network's capability to achieve high-level performance metrics, positioning it as a promising tool for distinguishing between benign and malicious software entities.

Figure-6. Training and validation loss.

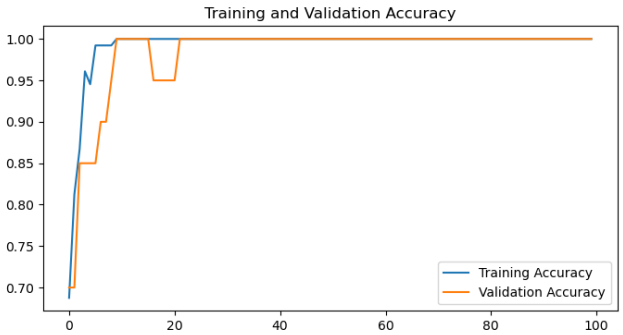


Figure-7. Training and validation accuracy.

Table-4. Proposed model parameter settings.

Total Number of Samples	**130 (50 good ware and 70 ransomware)**
Total Number of Classes	12 (7 Ransomwares, 5 good ware)
Batch size	86
Training and Testing split	70% / 30%
Number Channels or Z dimension	5 (of size 32, 64, 64, 32, and 32)
Kernel	3 X 3
Activation function	Relu
Max-pooling	2x2
Dropout	No
Epoch	100
Learning rate	0.0005
Optimizer	Adam
Moment decay (Beta)	2 (0.98 each)
Gradient Clipping	1
Total Parameters	417,410
Total Trainable Parameter	417,410

We initially used higher batch normalization and dropout layers, which caused the training phase to be distracted and underfitted and caused the model's loss function to be reduced drastically, but the training and validation accuracy remained constant at 0.5%. While this is a common issue with Siamese networks, we tried several learning rate parameters, and tuned other hype-parameter values such as additional layers, additional filters, and Z channels, as shown in the parameter table 5, making the model more complex. Specifically, to make the training more robust and to prevent overfitting, we added two more parameters alpha value 1 and alpha value 2, which are known as moment decay [38] of value 0.98, specifically for smoothing out the gradient and controlling the learning rate's variance, respectively.

Gradient clipping [39] was also used in the training process to ensure more stability and to be less prone to diverging. This ensures that the learning rate update remains small, preventing the model from overshooting.

3.6 Testing and Validation

In this section, we elucidate the steps taken to enhance the performance of the proposed Siamese model, specifically on the unobserved segment of the dataset, thereby emphasizing our efforts to address model generalization beyond the training data.

The learning rate is an important parameter in training neural networks that determines the optimization steps taken by the model. For some particular sets of experiments, we assigned smaller learning rates of 0.0001, which resulted in higher accuracy and F1-score in certain configurations. This indicated more precise convergence but was slow.

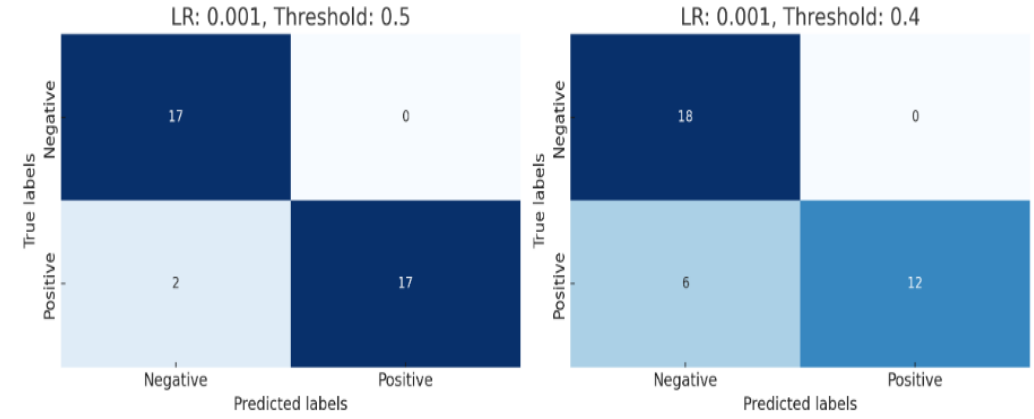
On the other hand, we tested higher learning rates of 0.0005, which achieved the highest accuracy and maintained a balanced precision and recall. The outcome provides reassurance that we have determined the optimal learning rate for the proposed model in terms of both convergence speed and performance quality.

The performance metrics are directly influenced by the threshold value. The threshold value serves as a boundary line for the pairs of inputs and, therefore, affects the confusion matrix, accuracy, precision, recall, F1-score, and AUC scores. We found that a threshold value of 0.5 resulted in higher recall scores, indicating that the model excelled in identifying true positives but at the expense of false positives, as shown in Table 5. Subsequently, by using a lower threshold value of 0.4, we observed a mix of outcomes, sometimes leading to increased precision as shown in Figure-10.

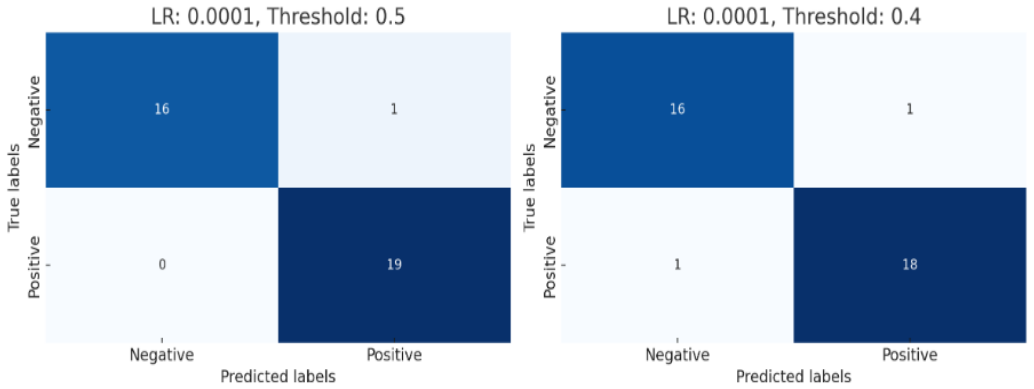
Table-5. Model performance on test and validation data.

Lr.rate	**Thr. value**	**Acc.**	**Prec.**	**recall**	**F1-score**	**AUC**
0.001	0.5	0.9411	1	0.8978	0.9485	0.95
0.001	0.4	0.8378	1	0.6756	0.8043	0.85
0.0001	0.5	0.9767	0.9571	1	0.9731	0.98
0.0001	0.4	0.9411	0.9521	0.9589	0.9549	0.95
0.0005	0.5	0.9767	1	0.9544	0.9789	0.98
0.0005	0.4	0.9767	0.9571	1	0.9731	1

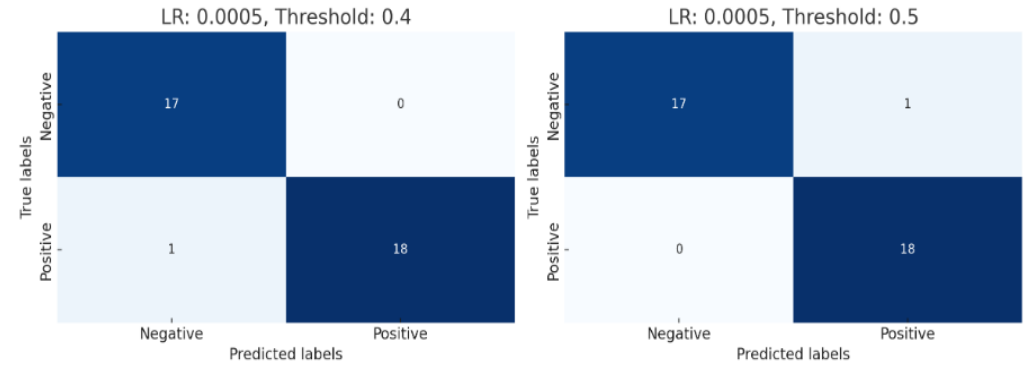
Lr* = learning rate, Thr.value = Threshold value, Prec= precision.



(a)



(b)



(c)

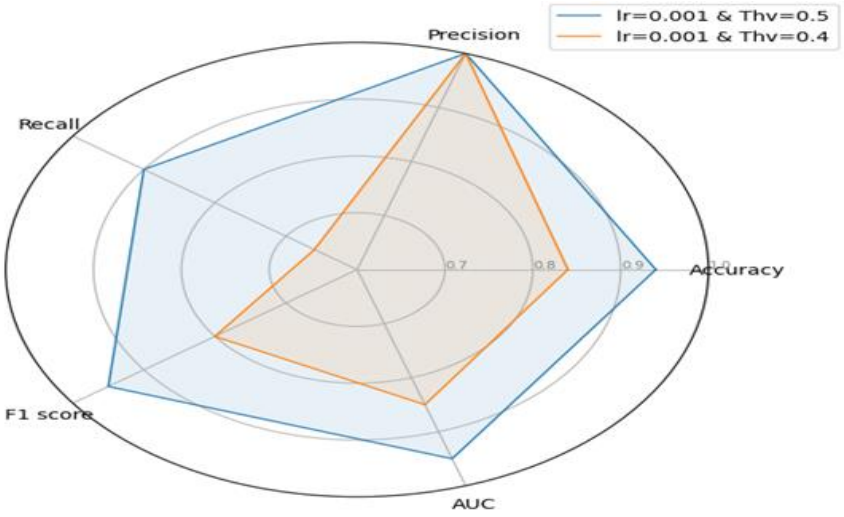
Figure-8. Confusion matrix showing the effect of learning rate and threshold values.

The above confusion matrices highlighted different learning rates and threshold values that impact the model's ability to successfully classify the ransomware behavior and goodware behavior. It also illustrates the trade-offs between true positives, true negatives, false positives, and false negatives detection.

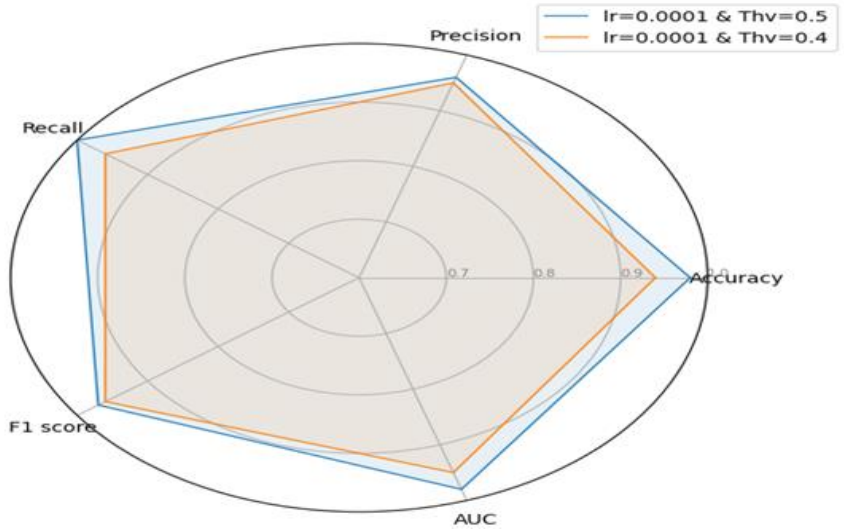
4. RESULTS AND DISCUSSIONS

In this section, we conduct a detailed analysis of our model's performance across different learning rates and threshold values, crucial for optimizing its classification accuracy and efficiency. We assess their impact on key performance metrics, including accuracy, precision, recall, F1-score, and Area under the Curve (AUC).

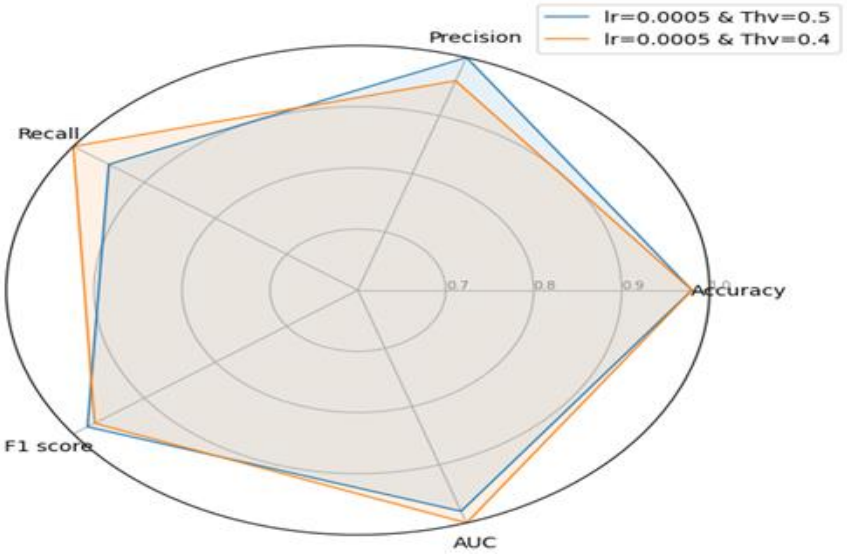
Using a learning rate of 0.001 and a threshold value of 0.5, the model achieved an accuracy rate of 94.11%, a precision rate of 100%, a recall rate of 89.78%, an F1-score of 94.85%, and AUC of 95%, as depicted in figure 9(a). This setup shows a high level of precision, indicating that the model is highly reliable when predicting a positive class. However, the recall rate suggests that the model is somewhat conservative since it missed a small portion of actual positive cases.



(a)



(b)



(c)

Figure-9. A radar chart showing the effect of varying learning rates and threshold values on model performance.

After testing the model with a learning rate of 0.001 and a threshold value of 0.4, we observed an accuracy of 83.78%. The precision remained at 100%, while the recall decreased to 67.56%. The F1-score also decreased to 80.43%, and the AUC decreased to 85%. The decrease in accuracy and recall suggests that the model became less effective at identifying true positives, despite maintaining a high precision.

However, when we decreased the learning rate to 0.0001 with a threshold of 0.5, the model's performance improved significantly, as shown in Figure-9(b). It achieved an accuracy of 97.67%, precision of 95.71%, a perfect recall of 100%, F1-score of 97.31%, and AUC of 98%. This configuration indicates a highly effective balance between precision and recall, showing that the model is excellent at identifying true positives without significantly increasing false positives. At a threshold of 0.4 and a learning rate of 0.0001, the model showed consistently high performance with an accuracy of 94.11%, a precision rate of 95.21%, a recall rate of 95.89%, an F1-score of 95.49%, and a AUC of 95%. These results indicate that the model is both precise and sensitive, and is capable of accurately detecting positive cases.

When the learning rate was slightly increased to 0.0005 and the threshold was set to 0.5 as shown in Figure-9(c), the model achieved an identical accuracy of 97.67% as compared to the lower learning rate at the same threshold. However, the model achieved a perfect precision rate of 100%, a recall rate of 95.44%, an F1-score of 97.89%, and an AUC of 98%. These results demonstrate exceptional performance, especially in terms of precision and F1-score, indicating that the model is highly reliable at predicting positive classes and maintaining high accuracy overall.

After adjusting the threshold to 0.4 and setting the learning rate at 0.0005, the model delivered exceptional performance across all the metrics. It achieved an accuracy of 97.67%, precision of 95.71%, recall of 100%, F1-score of 97.31%, and an AUC of 100%. This configuration represents the best balance achieved by the model. This indicates an excellent ability to identify true positives without increasing false positives, and the model achieved perfect performance in distinguishing classes as measured by the AUC. Adjusting the learning rate and threshold values has a profound impact on the model's performance across various metrics. Lower learning rates (0.0001 and 0.0005) with appropriately chosen threshold values tend to yield the best overall performance in terms of accuracy, precision, recall, F1-score, and AUC, demonstrating the importance of fine-tuning these parameters for optimal model performance.

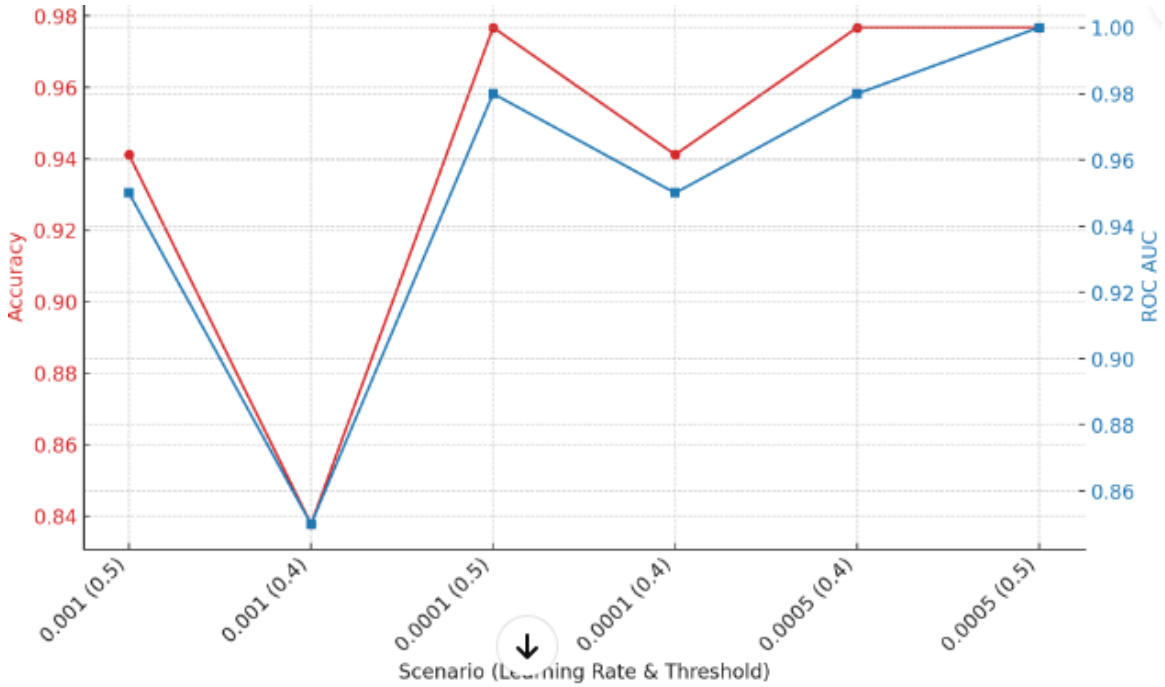


Figure-10. The influence of learning rate and threshold value on accuracy and ROC AUC using time-series graph.

Figure-10. Illustrates the impact of learning rate and threshold values on the model's accuracy and AUC. These measures provide a comprehensive understanding of a model's performance, encompassing its capacity to accurately identify targets, minimize false positives, effectively balance precision and recall, and overall predictive excellence.

The analysis provides valuable insights into the influence of different parameters on model outcomes. When the learning rate is set to 0.001 and the threshold is set to 0.5, the model performs exceptionally well with an accuracy rate of 94.11% and a perfect precision score. This configuration results in an F1-score of 0.9485 and an AUC of 0.95, indicating a good balance between

sensitivity and specificity. It is interesting to note that decreasing the threshold to 0.4 at the same learning rate leads to a decrease in accuracy and recall, highlighting the importance of threshold adjustments on model performance. Figure-9 also shows the impact of learning rate and threshold values on the model's accuracy and AUC.

After reducing the learning rate to 0.0001, we observed an improved model accuracy of 97.67% at both threshold levels (0.5 and 0.4), with slight variations in precision and recall. The AUC at a threshold of 0.5 reached an impressive 0.98, which was consistent with the model's performance at a learning rate of 0.0005 and a threshold of 0.5.

This consistency suggests that lower learning rates, when finely tuned with appropriate threshold values, can significantly improve the model's reliability and interpretability.

Notably, when the learning rate was set to 0.0005 and the threshold was set to 0.4, the model achieved an unparalleled blend of metrics: an accuracy of 97.67%, a precision of 95.71%, a perfect recall, an F1-score of 0.9731, and a AUC of 1. This configuration stands out as it demonstrates the potential for achieving optimal performance through meticulous parameter optimization.
The data in the confusion matrix is crucial to understand the nuances of model performance. It provides detailed information about true positive, false positive, true negative and false negative rates in different scenarios. This detailed analysis not only confirms the model's ability to distinguish between classes but also emphasizes the importance of learning rate and threshold settings in maximizing model precision and recall. By incorporating these insights, it becomes clear that strategic parameter tuning is essential for improving model performance. This may guide future research toward more effective and reliable predictive modeling.

The proposed model was also evaluated using the current studies and state-of-the-art models that use small samples of malware and ransomware datasets, as mentioned in Table-7. The parameters considered include implementing similar techniques of image recognition, accuracy, precision, F1 score, and AUC as in [4].

Ref.	Model	Accuracy	Precision	F1-score	AUC
[4]	Few-shot learning method based on the Siamese Neural Network	89 %	85%	86%	98%
[27]	Pre-trained deep CNN model	75%	77%	75%	94%
[26]	Pre-trained Inception-V1 on ImageNet	75%	79 %	76%	96%
[27]	Xception	73 %	78 %	77%	96%
Proposed model	Few-shot learning with Siamese network using architectural principles of AlexNet and VGG	97.6%	95.7%	97.3%	100%

Accuracy measures the proportion of true results. At 98%, the accuracy of the proposed model surpasses the pre-trained deep CNN model and the pre-trained Inception-V1 on ImageNet (both at 75%), Xception (73%), and the few-shot learning method based on the Siamese Neural Network (89%). This significant improvement suggests that the proposed model is exceptionally adept at classifying images of ransomware correctly, largely because of the enhanced learning capabilities introduced by integrating features of AlexNet and VGG into a Few-shot learning framework.

Precision assesses the model's ability to return only relevant instances. The proposed model achieves a precision of 95%, outperforming the pre-trained deep CNN model (77%), the pre-trained Inception-V1 on ImageNet (79%), Xception (78%), and the Few-shot learning method based on the Siamese Neural Network (85%). This shows that the proposed model is more reliable in identifying true positives while minimizing the risk of false alarms.

The F1-score is a metric that considers both precision and recall, by taking the harmonic mean of the two. Compared to other models being evaluated, with F1-scores ranging from 75% to 86%, the proposed model outperforms them all. This means that the proposed model performs optimally, striking the right balance between precision and recall, making it highly effective in scenarios where both are critical.

The proposed model achieves a perfect AUC ROC of 100%, showing its superior ability to discriminate between classes across all thresholds. This contrasts with the other models, where AUC ROC scores range from 94% to 98%, signifying that the proposed model not only excels in identifying true positives and true negatives but also maintains this excellence across all levels of decision thresholds.

The proposed model's architectural integration of AlexNet and VGG principles within a Few-shot learning Siamese network framework is central to its superior performance. This integration leverages the deep, hierarchical feature extraction capabilities of AlexNet and VGG, combined with the efficient learning of novel classes from limited examples that Few-shot learning frameworks offer.

5. CONCLUSION AND FEATURE WORK

Finally, we conclusively establish that the proposed model surpasses existing models notably in accuracy, precision, F1-score, and AUC. Integrating AlexNet and VGG's architectural principles into a Few-shot learning framework, along with the comparative learning strengths of the Siamese network, has enabled us to achieve exceptional performance. This highlights the importance of architectural innovation and the potential of combining existing methodologies to achieve breakthroughs in deep learning applications.

REFERENCES

[1] Statista Research Department. 2022. Percentage of organizations victimized by ransomware attacks worldwide from 2018 to 2022. https://www.statista.com/statistics/204457/businesses-ransomware-attack-rate/. Accessed: Oct. 11, 2022. [Online]. Available: https://cyber-edge.com/wp-content/uploads/2022/04/CyberEdge-2022-CDR-Report.pdf

[2] M. Huisman, J. N. van Rijn, and A. Plaat. A survey of deep meta‑learning.

[3] F. G. Mohammadi, M. H. Amini and H. R. Arabnia. 2020. An Introduction to Advanced Machine Learning: Meta-Learning Algorithms, Applications, and Promises. in Optimization, Learning, and Control for Interdependent Complex Networks, M. H. Amini, Ed., in Advances in Intelligent Systems and Computing. Cham: Springer International Publishing, pp. 129-144. doi: 10.1007/978-3-030-34094-0_6.

[4] J. Zhu, J. Jang-Jaccard, A. Singh, I. Welch, H. AI-Sahaf, and S. Camtepe. 2022. A few-shot meta-learning based siamese neural network using entropy features for ransomware classification. Computers & Security, 117: 102691, doi: 10.1016/j.cose.2022.102691.

[5] Pattern Recognition and Machine Learning. Accessed: Feb. 14, 2024. [Online]. Available: https://link.springer.com/book/9780387310732

[6] Introduction to Semi-Supervised Learning | SpringerLink. Accessed: Feb. 14, 2024. [Online]. Available: https://link.springer.com/book/10.1007/978-3-031-01548-9

[7] G. Pu, L. Wang, J. Shen and F. Dong. 2021. A hybrid unsupervised clustering-based anomaly detection method. Tsinghua Science and Technology, 26(2): 146-153, doi: 10.26599/TST.2019.9010051.

[8] S. Razaulla *et al*. 2023. The Age of Ransomware: A Survey on the Evolution, Taxonomy, and Research Directions. IEEE Access, 11: 40698-40723, doi: 10.1109/ACCESS.2023.3268535.

[9] P. Parkar and A. Bilimoria. 2021. A Survey on Cyber Security IDS using ML Methods. in 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India: IEEE, pp. 352-360. doi: 10.1109/ICICCS51141.2021.9432210.

[10] S. Sharma, C. R. Krishna, and R. Kumar. 2021. RansomDroid: Forensic analysis and detection of Android Ransomware using unsupervised machine learning technique. Forensic Science International: Digital Investigation, 37: 301168, doi: 10.1016/j.fsidi.2021.301168.

[11] B. M. Khammas. 2020. Ransomware Detection using Random Forest Technique. ICT Express, 6(4): 325-331, doi: 10.1016/j.icte.2020.11.001.

[12] J. Hwang, J. Kim, S. Lee, and K. Kim. 2020. Two-Stage Ransomware Detection Using Dynamic Analysis and Machine Learning Techniques. Wireless Pers Commun, 112(4): 2597-2609, doi: 10.1007/s11277-020-07166-9.

[13] Y. Yilmaz, O. Cetin, B. Arief and J. Hernandez-Castro. 2021. Investigating the impact of ransomware splash screens. Journal of Information Security and Applications, 61: 102934, doi: 10.1016/j.jisa.2021.102934.

[14] F. Faghihi and M. Zulkernine. 2021. RansomCare: Data-centric detection and mitigation against smartphone crypto-ransomware. Computer Networks, 191: 108011, doi: 10.1016/j.comnet.2021.108011.

[15] S. N.k. 2021. Ant Colony Optimization based light weight Binary Search for efficient signature matching to filter Ransomware. Applied Soft Computing, 111: 107635, doi: 10.1016/j.asoc.2021.107635.

[16] G. Ramesh and A. Menen. 2020. Automated dynamic approach for detecting ransomware using finite-state machine. Decision Support Systems, 138: 113400, doi: 10.1016/j.dss.2020.113400.

[17] T. Wisanwanichthan and M. Thammawichai. 2021. A Double-Layered Hybrid Approach for Network Intrusion Detection System Using Combined Naive Bayes and SVM. IEEE Access, 9: 138432-138450, 2021, doi: 10.1109/ACCESS.2021.3118573.

[18] F. Mercaldo. 2021. A framework for supporting ransomware detection and prevention based on hybrid analysis. J Comput Virol Hack Tech, 17(3): 221-227, doi: 10.1007/s11416-021-00388-w.

[19] [19] S. Jung and Y. Won. 2018. Ransomware detection method based on context-aware entropy analysis. Soft Comput, 22(20): 6731-6740, doi: 10.1007/s00500-018-3257-z.

[20] A. Yang *et al*. 2022. Application of meta-learning in cyberspace security: a survey. Digital Communications and Networks, doi: 10.1016/j.dcan.2022.03.007.

[21] S. Mascarenhas and M. Agarwal. 2021. A comparison between VGG16, VGG19 and ResNet50 architecture frameworks for Image Classification. in 2021 International Conference on Disruptive Technologies for Multi-Disciplinary Research and Applications (CENTCON), pp. 96-99. doi: 10.1109/CENTCON52345.2021.9687944.

[22] A. Krizhevsky, I. Sutskever and G. E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM, 60(6): 84-90, doi: 10.1145/3065386.

[23] S. Liu and W. Deng. 2015. Very deep convolutional neural network based image classification using small training sample size. in 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR), pp. 730-734. doi: 10.1109/ACPR.2015.7486599.

[24] K. He, X. Zhang, S. Ren and J. Sun. 2016. Deep Residual Learning for Image Recognition. in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770-778. doi: 10.1109/CVPR.2016.90.

[25] C. Szegedy *et al*. 2015. Going Deeper with Convolutions. Presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1-9. Accessed: Feb. 14, 2024. [Online]. Available: https://www.cv-foundation.org/openaccess/content_cvpr_2015/html/Szegedy_Going_Deeper_With_2015_CVPR_paper.html

[26] Z. Cui, L. Du, P. Wang, X. Cai and W. Zhang. 2019. Malicious code detection based on CNNs and multi-objective algorithm. Journal of Parallel and Distributed Computing, 129: 50-58, doi: 10.1016/j.jpdc.2019.03.010.

[27] S. Yue and T. Wang. 2022. Imbalanced Malware Images Classification: a CNN based Approach. arXiv, doi: 10.48550/arXiv.1708.08042.

[28] L. Chen. 2018. Deep Transfer Learning for Static Malware Classification. arXiv, doi: 10.48550/arXiv.1812.07606.

[29] W. W. Lo, X. Yang and Y. Wang. 2019. An Xception Convolutional Neural Network for Malware Classification with Transfer Learning. in 2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS), pp. 1-5. doi: 10.1109/NTMS.2019.8763852.

[30] Z. Cui, F. Xue, X. Cai, Y. Cao, G. Wang and J. Chen. 2018. Detection of Malicious Code Variants Based on Deep Learning. IEEE Transactions on Industrial Informatics, 14(7): 3187-3196, doi: 10.1109/TII.2018.2822680.

[31] S. Sharma and S. Singh. 2021. Texture-Based Automated Classification of Ransomware. J. Inst. Eng. India Ser. B, 102(1): 131-142, doi: 10.1007/s40031-020-00499-w.

[32] A. Arabo, R. Dijoux, T. Poulain and G. Chevalier. 2020. Detecting Ransomware Using Process Behavior Analysis. Procedia Computer Science, 168: 289-296, doi: 10.1016/j.procs.2020.02.249.

[33] K. C. Roy and Q. Chen. 2021. DeepRan: Attention-based BiLSTM and CRF for Ransomware Early Detection and Classification. Inf Syst Front, 23(2): 299-315, doi: 10.1007/s10796-020-10017-4.

[34] U. Zahoora, M. Rajarajan, Z. Pan, and A. Khan. 2022. Zero-day Ransomware Attack Detection using Deep Contractive Autoencoder and Voting based Ensemble Classifier. Appl. Intell. doi: 10.1007/s10489-022-03244-6.

[35] Y. S. Joshi, H. Mahajan, S. N. Joshi, K. P. Gupta and A. A. Agarkar. 2021. Signature-less ransomware detection and mitigation. J Comput Virol Hack Tech, 17(4): 299-306, doi: 10.1007/s11416-021-00384-0.

[36] T.-M. Liu, D.-Y. Kao and Y.-Y. Chen. 2020. LooCipher Ransomware Detection Using Lightweight Packet Characteristics. Procedia Computer Science, 176: 1677-1683, doi: 10.1016/j.procs.2020.09.192.

[37] M. Hirano, R. Hodota and R. Kobayashi. 2022. RanSAP: An open dataset of ransomware storage access patterns for training machine learning models. Forensic Science International: Digital Investigation, 40: 301314, doi: 10.1016/j.fsidi.2021.301314.

[38] Demon: Improved Neural Network Training with Momentum Decay | IEEE Conference Publication | IEEE Xplore. Accessed: Feb. 21, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/9746839

[39] A. Bajaj. 2024. Understanding Gradient Clipping (and How It Can Fix Exploding Gradients Problem). neptune.ai. Accessed: Feb. 21, 2024. [Online]. Available: https://neptune.ai/blog/understanding-gradient-clipping-and-how-it-can-fix-exploding-gradients-problem