Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 3, Issue 4, 2025

INTELLIGENT MALWARE DETECTION USING NEURAL ARCHITECTURES: A COMPARATIVE STUDY OF CNN, LSTM, FNN AND BI LSTM

Nabia Shaheen[*1], **Sagar Lohana**[2], **Muhammad Ramzan**[3]

[*1]*Computer Science, Muhammad Ali Jinnah University, Karachi, Pakistan .*
[2]*Computer Science, Shaheed Zulfikar Ali Bhutto Institute of Science and Technology University Gharo Campus,Pakistan.*
[3]*Computer Science, Institute of Business Management (Iobm), Korangi Creek, Karachi, Sindh*

[*1]snabiasep@gmail.com, [2]sagar.lohana@ghr.szabist.edu.pk, [3]muhammad.ramzan@iobm.edu.pk

DOI: https://doi.org/10.5281/zenodo.15244956

Keywords
INDEX TERMS LSTM), Long Short-Term Memory ;(CNN) Convolutional Neural Networks; Feed forward Neural Networks (FNN) ; Bidirectional LSTM (Bi-LSTM); machine learning ,deep learning.

Article History
Received on 11 March 2025
Accepted on 11 April 2025
Published on 19 April 2025

Corresponding Author: *
Nabia Shaheen[*1]

Abstract

Identification and classification of malware is still a major problems in the area of cyber security because of consistent appearance of new variants. The paper discusses our approach to develop an accurate method for virus detection using the complementary skills of artificial intelligence. Malware detection has become a difficult task with the appearance of clever dangerous programs, which exist undetectable by signature-based antivirus process, such as mutating malware. The paper is dedicated to the use of deep learning models, especially Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN), Feed forward Neural Networks (FNN), and Bidirectional LSTM (Bi-LSTM), for malware detection through the analysis of Windows executable API call sequences. Pre-processing the data involved two steps: tokenizing the API calls and disassociating the malware exemplars from the encoded binary values. Data sequencing pattern identification was ensured by adopting suitable loss functions, optimizers, and validation methods when training models. The CNN model was the most accurate among them, almost 92%, precisely because of its Conv1D and MaxPooling layers that catch spatial patterns very well. The LSTM and Bi-LSTM modes seem to perform even better, with a 90% accuracy rate. The FNN model, which is of course the simplest architecture, shows the lowest accuracy above 80% as well. The data shows that CNN is the best model for this malware detection task since it has high accuracy and it trains quickly. The LSTM and Bi-LSTM models prove to have a performance that is the strongest but they are the ones that will take more time to train. The study implies that deep learning cognitive networks can easily discern and classify malicious API calling patterns when marinated well with this complex system. The suggestion is to continue the studies on ensemble models, data augmentation and hyper parameter tuning for the future to improve accuracy and generalization. Through such research, it will become feasible to design and implement a strong and reliable cyber security architecture stand up the most powerful, cutting edge malware threats of today.

INTRODUCTION

Malware is the shortened term of malicious software and it is a number of unwanted programs that are created with a purpose of getting into computer systems and networks and can also harm or interrupt them. It comprised viruses, worms, Trojans, ransomware, and spyware which are all different means and aims at attacking the computers and gaining unauthorized access to the system. With the expansion of dependency on digital solutions across all industries, malware attacks are becoming more comprehensive and more often, which may lead to critical threats such as data corruption or unavailability, data leaks, and disruptions of regular business processes. Acquiring knowledge about the workings and nature of these bad-wares is pertinent to the establishment of viable strategies, complementing modern AI technologies to increase the efficiency of detection and prevention. The topic of this paper is detection of malware and artificial intelligence system with the goal of suggesting new measure that could be taken in order to minimize the risk of cyber threats.

Traditionally, malware has been identified using conventional signature techniques. This makes it difficult to distinguish between different kinds of malware. The malicious application's presence of several polymorphic layers is the cause. These layers make it challenging to identify malware, and they also enable side mechanisms that upgrade to a new version faster. Because of this, antivirus software had a very tough time detecting such malware. As a result, traditional techniques for identifying malware are inadequate for accurately detecting malware.

Finding out whether a file contains malware or not is very crucial. A lot of issues are brought about by the rise in malware, and businesses are losing vital data and dealing with other issues. Secondly, malware can swiftly inflict significant damage to a system by slowing it down and encrypting a significant amount of data on a personal computer. This study contains a thorough explanation of the flexible framework for 'machine learning' and 'deep learning' algorithms, Malware detection is achievable using these techniques. The basis is that these algorithms make it simple to differentiate between files that are malware-free and those that are not.

The risk of cyber-attack is rising in tandem with the demand for online services. As a result, several Malware files in the system are causing problems for individuals and corporations. Because of these malware files, it is incredibly difficult to protect or retain sensitive data from personal computer systems. There are several approaches available to achieve the objective of locating malware on the computer system. This indicates that some traditional approaches to malware detection are inadequate for correctly identifying malware. This means that some malware, which can take many different forms, could sneak into the system and go undetected. It means that traditional malware detectors were unable to find any malware on the machine. Such a malware detector that can immediately identify any malware that is on the system is required in light of these problems.

The desecration for this research lies in the fact that better solutions for detecting malware are required to improve and leverage deep learning. The need to efficiently represent data at multiple levels of abstraction presents a significant challenge in this regard. Deep learning models promise an efficient solution to this challenge. The prior research has pursued this direction in many ways; however, the vast majority compares basic classification methods with straightforward AI solutions or the relatively simple deep learning algorithms only while there is a large unexplored space of more complex models.

Thus, the issue addressing which this research is carried out consists in designing and modeling effective deep learning architectures for the enhanced malware identification. Thus, using these advanced models to learn, the focus is on attaining higher accuracy, identifying malware samples that current models cannot easily detect, and, hence, creating a more effective barrier against the threat actors and their evolving tactics. Such advanced deep learning techniques, which are still in their developmental stages, have not yet been fully investigated or implemented, and this research aims to fill this gap and take its part in the constant readiness to counteract malicious activities in digital domains.

Malware is posing a significant threat to computing gadgets and digital systems in the rapidly evolving

digital age [1]. Malicious software, or malware, strategically designed by attackers with negative intent, has the capability to infiltrate networks, compromise vital infrastructure, and pilfer sensitive data [2]. Malware refers to malicious software designed to evade detection by malicious attackers [3]. As the concept of information society advances with technologies like the IOT (internet of things) has raised security concerns, posing a significant obstacle to industrial development, prompting cybercriminals to target specific devices and networks .These attackers exploit system vulnerabilities using malware, a term encompassing software designed to harm operating systems (OS) [4] .
The surge in malware attacks is particularly pronounced with the transformation of daily interactions driven by the development of mobile technology. However, its extensive usage in online pursuits such as education, social media, banking, shopping, and surfing is significant makes them susceptible to virus invasion and attacks. we provide the research on Internet of Things (IoT) attacks, vulnerabilities, and mitigations reported the use of artificial intelligence techniques to detect cybersecurity assaults on the Internet of Things [4].
In order to examine the state of malware detection techniques at the moment, this paper reviews the literature.

The following are the contributions made by the paper:
• Describes Neural networks for detecting malware as well as new technology trends that contribute to its formation.
• Provides an overview of recent research on malware detection.
• Describes key techniques and strategies for detecting malware.
• Talks about the difficulties of today and suggests fresh presumptions for methods of detecting malware.
• Offers a methodical summary of malware detection techniques and strategies for additional research.
The remainder of the document is structured as follows: The problem definition is presented in Section II by reviwing different papers. Section III explains malware detection methodologies and algorithms, and Section IV discusses malware detection approaches and presents an evaluation of malware detection techniques. Section V provides a conclusion and future direction.

I. LITERATURE REVIEW
Malware detection is a concept which encompasses the protective measures and tools necessary to identify, prevent and minimize the damaging effects of malware threats. Some of the well-known techniques.

1) Static File Analysis
It is simply the method used for evaluating the code of a file to find hints of malicious intention without actually executing it. To determine whether the file is malicious or not, first see if the file name has suspect values, such as an IP address, or file hash [5, 6].

2) Dynamic Malware Analysis
According to this, it is supposed to emulate a malicious code in an isolated area which is known as a sandbox. The security analysts might have to review the malware operating on this isolated system and analyze it in detail without worrying that e-mail might infect their computer or spread through the network of the company [6, 7].

3) Signature-Based Detection
Signature-based monitoring looks for the infectious behavior using typical malware signatures that are stored in the relational database [8].

4) Traditional Methods vs. AI-Based Approaches
The application of traditional malware detection techniques of anti-virus issuers, including static and dynamic analysis, is meant for the purpose of detecting whether the application holds the potential for malicious behavior. These approaches embody unique indicators representing such patterns and signatures to isolate known threats [9].
Machine learning incorporated into AI systems enables the systems to accurately expose various threats from evolving malware in the malware detection process. Humanly, the learning and deep learning techniques, especially, are coming more and more in use by advanced cybercrimes detection

because of their features which are capable to analyze gigantic data and detect complex patterns which most of the traditional techniques cannot work well with them [10] .Deep learning based detection techniques such as CNN, RNN, LSTM, and a encoder require sophisticated mechanisms to improve their accuracy; long short-term memory (LSTM) is specially known for its memory cell capability [11]. Electronic malware identification and machine learning techniques are put together in sophisticated malware detection systems to be more preventive and discover threats that are not well known. With this proactive approach, advanced malware variants that might avoid detection by conventional signature-based techniques can be found. AI technology combined with conventional detection techniques improve cyber security systems' overall efficacy against contemporary malware threats. Regarding Windows platform security, a lot of effort has gone into shielding computers from online threats. This section describes the many approaches and deep learning algorithms employed by several research teams to identify Windows malware [12] shows how malware detection techniques are being applied to deep learning and machine learning. The article includes a complete list of all models used by other researchers in the field of malware detection. The following models are mentioned in the paper, however the list includes decision trees, k-nearest neighbor, multi-level perceptron's, and support vector machines. The authors used RF Deep Neural Network Decision Tree and Auto-Encoder SVM for virus identification[13].

A deep learning and feature selection technique to malware detection is provided which improves the accuracy of identifying malicious applications on computer networks. Two malware-free and infected datasets are employed in the system, along with two intelligent computer programs that were trained on the data. Several machine learning techniques, such as logistic regression, AdaBoost, gradient boosting, decision trees, K-NN, decision trees, and random forests have been used in prior studies [14].

The authors have specifically proposed and evaluated two deep learning models: the LSTM model and the Dense Layers model. They have also used feature selection and a number of splitting scenarios in their assessment. In the first dataset, the Dense Model obtained 99.99% accuracy without feature selection, whereas in the second dataset, the Ensemble Learning technique achieved 94.15% accuracy. In the first dataset, the Dense Model had the highest accuracy. A machine learning-based technique to improve current malware detection classifiers by integrating features from Portable Executable file header data is presented by [15].

The study looks into how machine learning methods might be used to identify dangerous executable files on networks. It looks at 28 features from the functions of four different kinds of PE files, packaging, imported DLLs, and metadata. Machine learning models such as SVMs, DT, RF, naive Bayes, decision trees, and SVM in all scenarios are covered in the study. Algorithms other than the random forest model performed worse [16].

Future study on assessing the proposed system using metrics like the ROC curve and F1-measure on big data sets and various malware types is suggested in the publication [17].

The MCSC technique was presented by and uses CNN and SimHash for malware classification. They utilized hash values to detect comparable viruses using locality-sensitive hashing (LSH) after breaking down malware code and turning it into grayscale images. They were surprisingly 98% accurate in their identification. MalDeep is a deep learning-based malware detection tool for binary files [18].

The suggested approach yielded results with 95% accuracy by using API frequency vectors and PCA-initialized op-code bi-gram matrices [19].

Detecting malware more precisely and efficiently is heavily dependent on the extraction and selection of particular features. Multiple creative methods have been designed to find the best traits and to go through the process of extraction effectively. These observations come from pertinent research articles. Three essential steps comprise the fundamental framework for machine learning-based malware detection: feature extraction, feature selection, and classification are the common ML tasks [20]. A sandbox offers an environment that highlights the abilities of malware. Extracting the n-gram domain from malware's behaviors in sandbox is the main research area. This involves the following five stages of extraction that encompass attributes well-suited to describing malware samples' behavior in a manner that is as reliable as possible in classifying them [21].

A feature-based machine learning technique, the Feature Extraction and Selection Tool (Fest) presented in this paper, is used mainly for Android malware detection tasks. In Fest, malware detection models benefit from improved strategies like blending feature extraction and selection steps to obtain higher accuracy of models [22]. The discovery of this investigation has a fruitful result in the fact that the advanced feature extraction and selection play a key role to increase the detection power of the malware detection system. The system can dodge the cyber threats with ever-increasing complexity later, thus fortifying cyber security [22].

In fact, a parallel research project is in various stages of investigating how to secure devices on the Android platform. In this paper, looks at using machine learning to detect malicious executable files on networks (PE files). The paper puts forth a model of malware detection for Android apps on the basis of 21 machine learning algorithms, which combines features extracted from NDTF, Y-MLP DNN and FF clustering. For real-world apps, the model showed a high identification rate of 98.8%. In light of the significant challenges in the field, the paper makes recommendations for future research in malware detection for Android apps, including the use of ensemble learning techniques for increased accuracy and robustness, deep learning algorithms for extracting complex features, real-time malware detection systems, and zero day malware detection models [23]. Permission-based method makes inventive use of manifest files and Android package permissions to generate feature vectors for neural network training. Even with an accuracy rate of 88%, this method offers novel insights.

In order to obtain a strong F1 score, MalDozer disassembles classes. dex files and trains neural networks with an emphasis on API method call sequences. Similarly, to broaden the scope of Android malware detection, provide permission-based methods employing graph clustering and multiple characteristics with an F1 score of about 80%, investigate unsupervised malware prediction based on resource utilization. With[online Android malware detection system, which extracts three important features and uses deep belief networks for detection, the environment becomes more diverse [24].

Weighted softmax loss for deep convolutional networks on malware photo classification further expands the repertory by addressing imbalances in malware families. Suggest a CNN-based method combine deep auto-encoder [25].

Lastly, [10] introduced CANDYMAN, integrating dynamic analysis with Markov chains for effective malware detection. This is because most daily exchanges of various information through the internet depend largely on digital devices and systems thus making the entire industry more vulnerable to extreme repercussions in the future. Digital technology continues to expand at a rate that comes with accompanying malware risk., there are approximately three and a half billion internet users. Economic rewards induce cyber criminals to deploy various attacks using different malware For instance, threats (like Trojan horses, worms, and viruses) contribute to the growing menace. Targeting the digital devices or system's confidentiality, integrity, and accessibility. The cost of worldwide maliciousness will be about half trillion indicating why it should be guarded against Malwares are the first threat and distributed AV vendors' detection system as the first line of defense against these threats [10].

[Malware detection, which impacts on legal, reputation and financial consequences to companies is one of the most crucial elements to deal with security issues. As such, deep learning is seen as one of the reliable technologies available today and it gives automatic extraction of advanced features that can support the detection processes [26].

In contrast, Convolutional Neural Networks (CNN) is adept at extracting local hierarchical characteristics derived from data samples, making them commonly used for image classification. Although typically associated with image analysis, we posit that CNN is a superior choice for malware detection when compared to RNN. Unlike RNN, CNN can recognize features regardless of their location, allowing for the identification of translation or distortion caused by adding redundant API calls or machine instructions. For example, in the image space a hand can still be recognized however it is posed. In contrast, text unit meanings change

depending on this surrounding context. Malware classification utilizing CNN has not received much attention in the literature, and this topic hasn't even been thoroughly discussed. In the MDS based on hybrid analysis proposed by. Features were extracted using hybrid analysis and feature vectors classified with CNN. But using CNN only with API calls or machine instructions perhaps has not excited much interest--maybe because it is difficult for malware to be presented as an image, and hard also that files (or images) can be of incredibly varying sizes [27].
Therefore, this research focuses on addressing this gap by employing such complex deep learning models such as Feed forward Neural Networks (FNN), Convolutional Neural Networks (CNN), Long Short-Term Memory networks (LSTM), and Bidirectional LSTM (Bi-LSTM). These models have the ability to perform complex task with improved performance. Therefore, building and modeling efficient deep learning models for improved malware identification is the main gap which needed to be solved.This research makes several significant contributions to the field of malware detection through the application and refinement of advanced deep learning techniques .Among them, the most deliverable aspects include a vast number of researches and real-world realizations of the state-of-the-art deep learning approaches like Feed forward Neural Networks (FNN), Convolutional Neural Networks (CNN), Long Short-Term Memory networks (LSTM), and Bidirectional LSTM (Bi-LSTM) for malware detection. Through these models, this research builds upon the non-linear pathways of standard machine learning exercises and elementary structures of deep learning to pursue increased efficacy and resilience in detecting and categorizing malware samples.

II. PROPOSED METHODLOGY

A thorough methodology was used in this study to analyze a malware dataset that originally included a variety of malware file types. Dataset was pre-processed with all possible care! Malware byte codes are converted to numbers columns by using API calls. Following transformation, this dataset was arranged into two separate folders: 'Types,' which was a grouping of the malware type and 'Calls,' which was the unique ID linked to the API calls. Initially, some machine learning and deep learning models to analyze the project were selected which were feed forward neural network (FNN), convolutional neural network (CNN), long short-term memory (LSTM), and bidirectional LSTM. These models were trained in detail for readiness to be crosschecked using the calculated data and to evaluate the value of their predictive power. The accuracy of each model was calculated as a metric for performance quantification. Obtaining a score for each model's accuracy was the next step in this process. A comparative analysis was conducted and the comparison table between the models was created to highlight the key strengths and weaknesses. Machine learning algorithms gain insights via this modelling comparison which help to identify the effective ways of malware classification on the basis of model used and the dataset and scoring parameter that are taken into considerations. In this well-designed plan, the valuable insights were obtained, which made it possible to get deep understanding of the performance characteristics of different machine and deep learning models, while doing the malware analysis.

1. LSTM (Long Short-Term Memory)

Recurrent neural network (RNN) that simulates sequential input and was created to solve the vanishing gradient issue that traditional RNNs typically run into. It uses sophisticated internal structure that is capable of regulating the ordered input/output in order to achieve the goal of memory [28] . An outline of LSTM architecture is provided here:

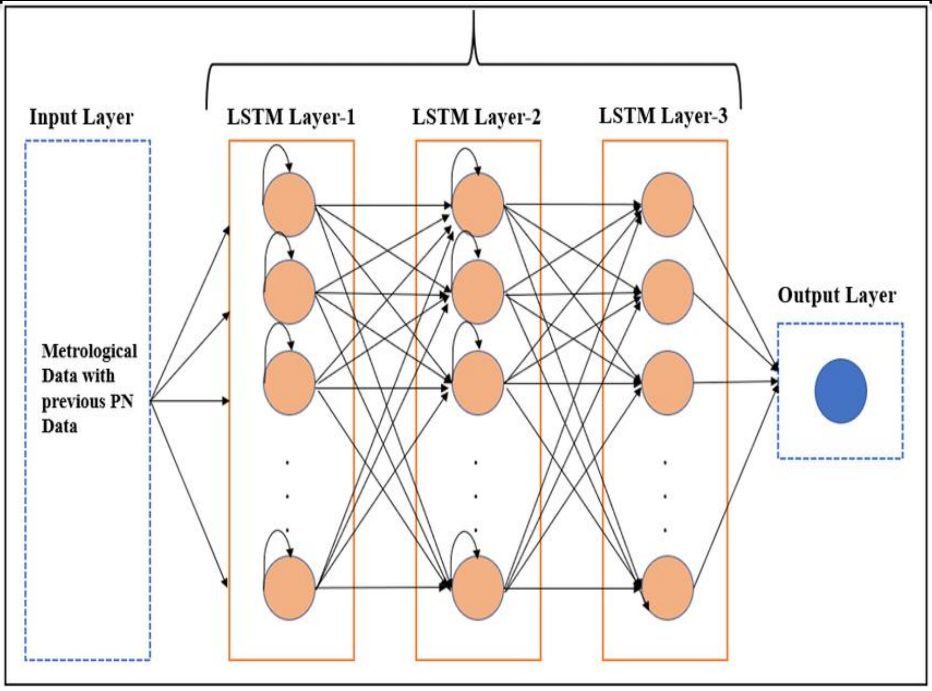


Fig 1 Sequential Long Short-Term Memory (LSTM) architecture

In traditional RNNs, a hidden state is kept by the network. Every step updates this state to reflect the data given in the sequence. Frequently, a simple activation function like tanh or ReLU is applied to obtain the hidden state at each time step based on the input and the previous hidden state. Consequently, these associations are unable to create the full context of the idea with longer sequences since such networks suffer the vanishing gradient problem, when the gradients of back propagation diminish in value with an increase in number of operations[29].

2. CNN Architecture

Convolutional neural networks (CNNs) are one type of Deep Learning neural network design that is widely used in computer vision. "Computer vision" is the area of artificial intelligence that allows computers to interpret and process images and other visual data. Convolutional neural networks (CNNs), an enlarged version of artificial neural networks, are mostly utilized for feature extraction from matrix datasets that resemble grids. For example, visual datasets like photos or videos that contain a lot of data patterns. It consists of fully integrated input, pooling, and convolutional layers [30].

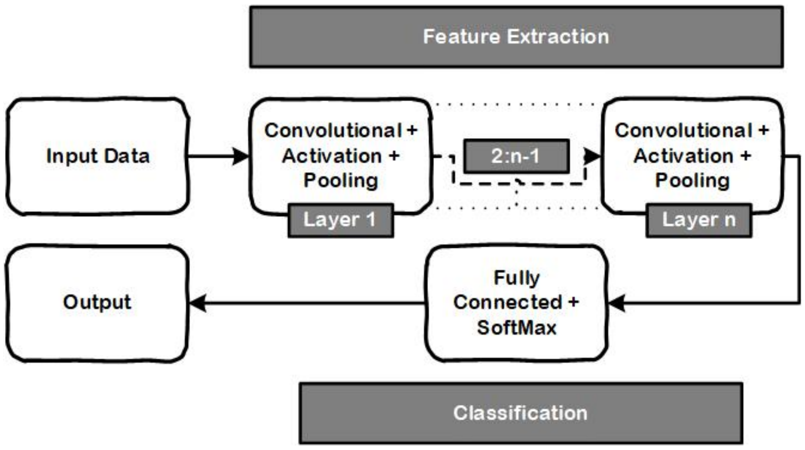

Feature Extraction
Input Data
Convolutional + Activation + Pooling
Layer 1
2:n-1
Convolutional + Activation + Pooling
Layer n
Output
Fully Connected + SoftMax
Classification

Figure 2: Block diagram of CNN architecture

3. FNN Architecture

A single artificial neural network, feed forward neural network (FNN), sometimes called as feed forward network, is a simple type of neural network where connections between the nodes do not form cycles. The primary elements of its design are an input layer, one or more hidden layers, and an output layer. Each layer consists of multiple nodes or neurons, and weighted connections connect every node of each layer to all other nodes in other layers. The next step will be transferring the received data into the network which multiplies it by the connection weights and then feeds it to the non-linear activation functions located in the hidden layers. This system through the use of differing layers of incoming data, can draw out the linkages which are hard to find and other complex patterns. If the last layer is chosen, the result is output in the form of projected numbers or probabilities. With a specific loss function in mind, and the need for accuracy in a training process, fine-tuning of the FNN through connection weights changing back propagation and the optimization methods such as stochastic gradient descent is what is needed. This simplicity of function may help in universal application of such network to a variety of tasks, ranging from simple classification to problems like regression and pattern recognition [31].

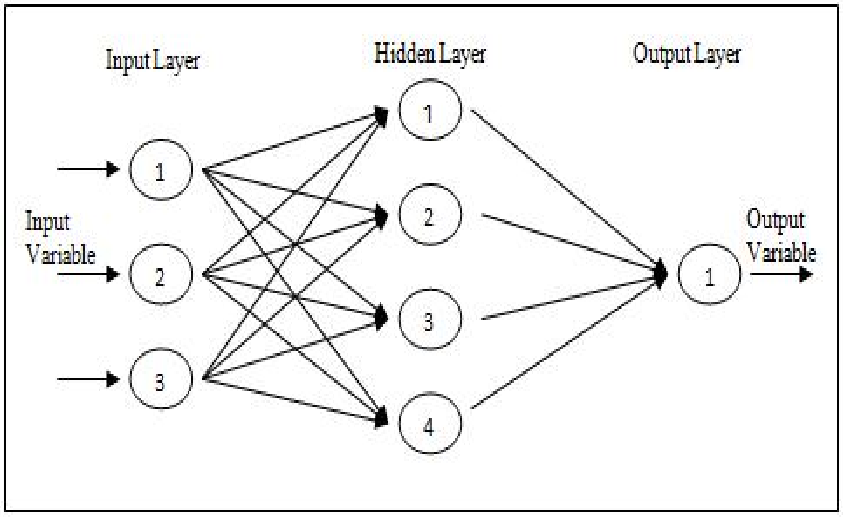


Figure 3 Architecture of FNN

4. Bi-LSTM Architecture

Chunks of information are frequently fed directly into the system from the input port of Bi-LSTM neural block diagram through a single layer. The input could be a sequence of characters, words, or any other time-based data, mostly expressed as feature vectors. While the input layer positions the LSTM block as the forward and the backward LSTM layer as the next, the structure is typical of recurrent neural networks. The forward LSTM has memory units and gates that maintain contextual information across time since the unit's process and keep track of information throughout the whole sequence . The backward LSTM works oppositely to the forward one and at the same time processes the data from beginning to the end ,BI-LSTM networks utilize this one-way system to gather contextual information spanning on either side of the sequence . This forms a foundation for their knowledge and with time, they come to comprehend the facts in depth. The results from these two layers are usually either combined (concatenated) or integrated in some other way for tasks like sequence prediction or classification to create a comprehensive representation that may be passed through other neural network layers [31].

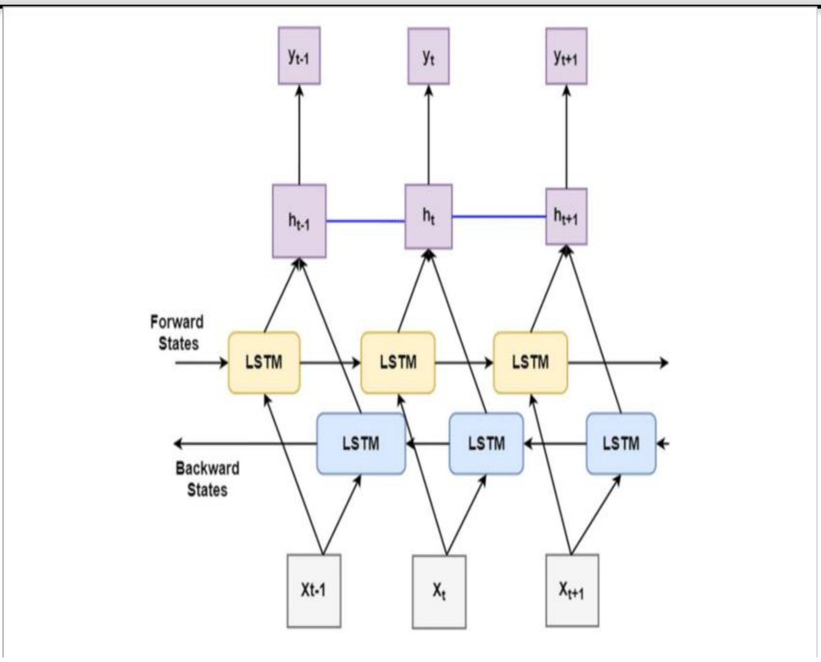


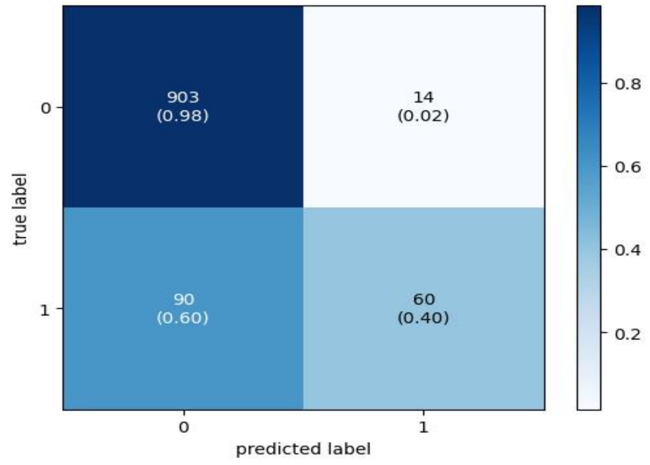
Figure 4 Bidirectional LSTM for Sequential Data Processing

III. RESULT AND DISCUSSION

The paper is dedicated to the use of deep learning models, especially Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN), Feedforward Neural Networks (FNN), and Bidirectional LSTM (Bi-LSTM), for malware detection through the analysis of Windows executable API call sequences.

1) LSTM Based Classification Model

The LSTM model has a sequential design that allows it to recognize temporal patterns in sequential data, such API requests. Included are LSTM layers, dropout for regularization, and dense output layers. The model performed better overall, with very few outliers, both in terms of accuracy and loss. Initially, the training loss was 0.1147 and the accuracy was 87.04%; however, the accuracy and validation losses were 89.78% and 0.0869, respectively. The overall pattern of improvement in both the loss and accuracy on the training data showed that the model was learning and becoming more accurate as the training continued.



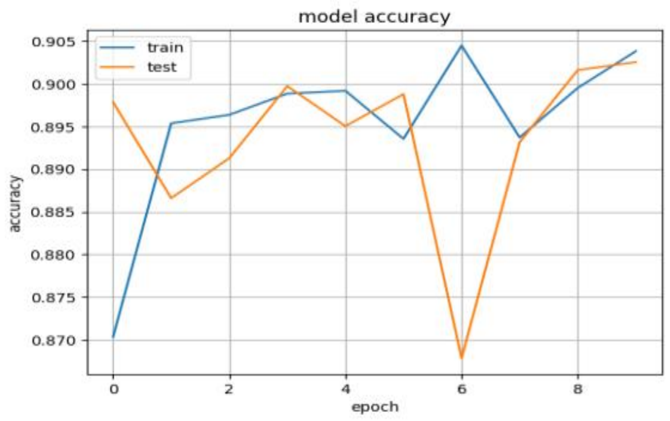
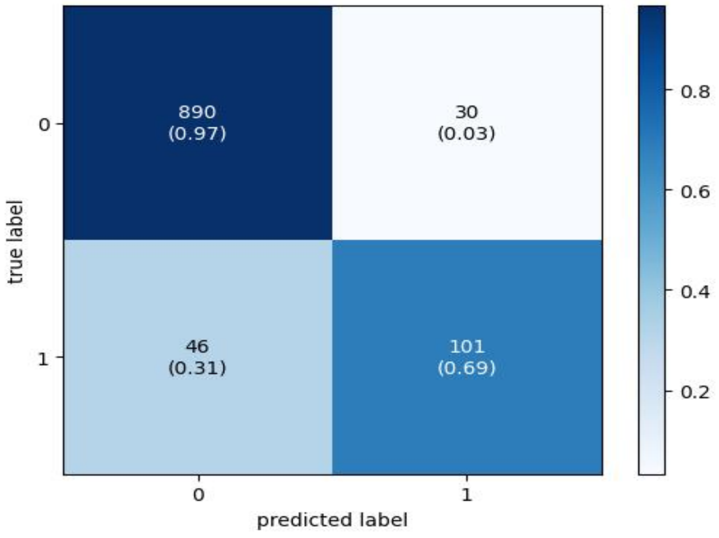


Fig 5 LSTM Model evaluation and accuracy

2) CNN Based Classification Model

convolutional neural networks to find spatial patterns in data. It has MaxPooling, Conv1D, and dense layers for categorization. The Convolutional Neural Network (CNN) continuously showed a rise in accuracy and a fall in loss throughout ten epochs. Starting with an initial training accuracy of 85.26% and a validation accuracy of 86.22%, the network's performance steadily improved. The accuracy above 90% by Epoch 6 for both training and evaluation showed effective learning and generalization. Throughout the trend, high accuracy and low loss were maintained, although there were noticeable fluctuations in validation loss that might have been brought on by noise or overfitting. When everything was taken into account, the CNN demonstrated strong resilience and learning ability during the training stage.



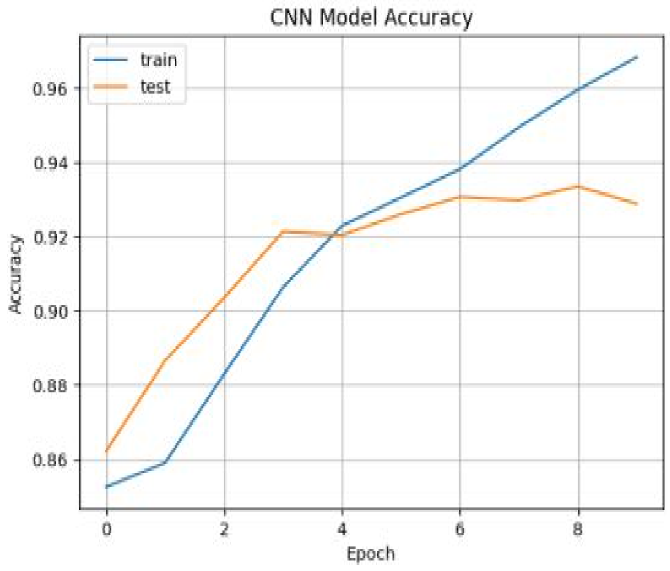
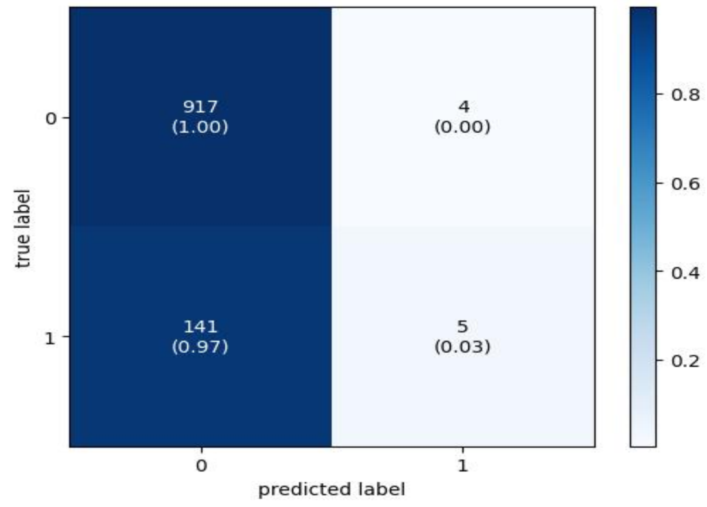


Fig 6 CNN Model evaluation and accuracy

3) FNN Based Classification

Over the course of ten training epochs, (FNN) shown a promising decrease in loss and improvement in accuracy. Initially, the accuracy was a respectable 72.58%, but the model had a significant training loss of 11.4220. With a validation accuracy of 86.13%, the validation loss was likewise high at 3.0223. But accuracy increased and loss fell progressively as training went on. By the last epoch, the validation accuracy was 86.41%, the validation loss was 0.4442, and the model's loss had decreased to 0.5588. The final test accuracy of

86.41% validates the model's ability to generalize well to new data, as evidenced by its constant improvement over the epochs.



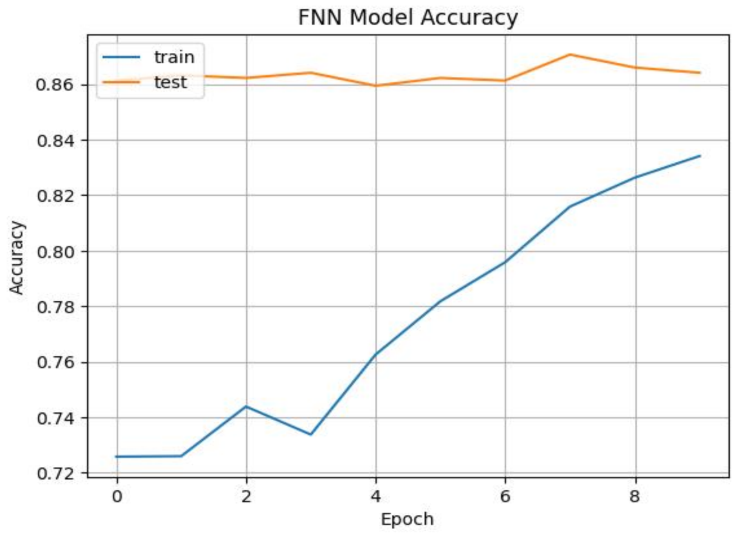
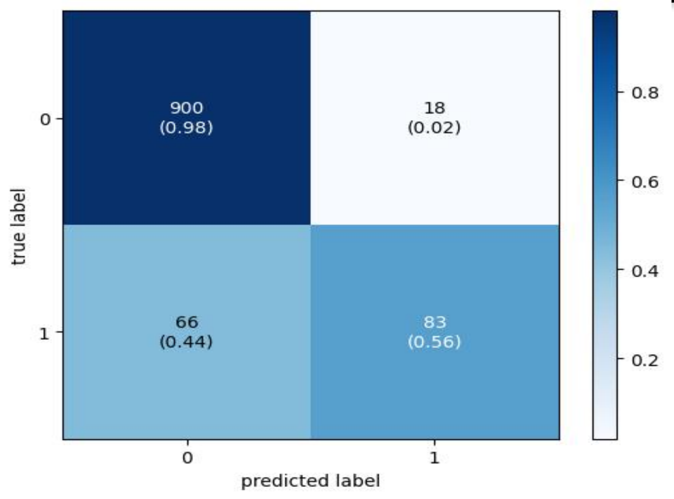


Fig 7 FNN Model evaluation and accuracy

4) BI-LSTM

Over the course of ten training epochs, the Bi-directional Long Short-Term Memory (Bi-LSTM) model consistently improved. The model began with an accuracy of 85.78% and a training loss of 0.4060. Over time, accuracy grew and the model's loss continuously dropped. The training loss decreased to 0.2414 with an accuracy of 91.64% at the fifth epoch, while the validation loss decreased to 0.2788 with an accuracy of 91.47%. The validation accuracy increased to 92.13%, while the training loss further lowered to 0.2056 with an accuracy of 93.15% by the last epoch. The model demonstrated its resilience and reliability as evidenced by its final test accuracy of 92.13%, which shows that it learned well and generalized well.



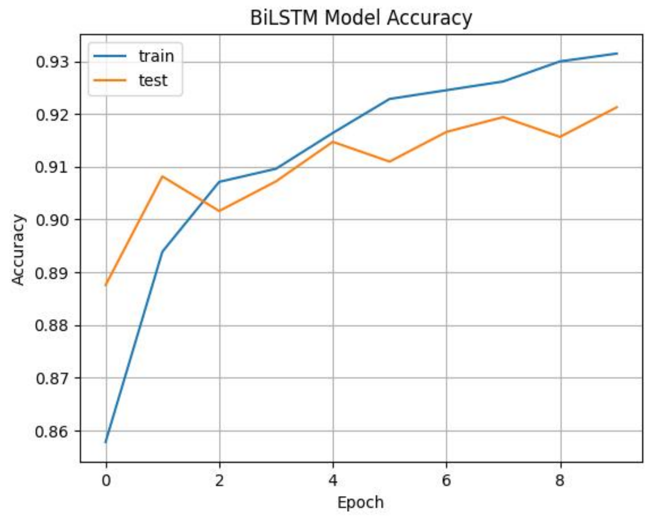


Fig 8 BI-LSTM Model evaluation and accuracy

5) **COMPARISON BETWEEN MODELS**

The models themselves, the data, and the training procedure may all have an impact on the variations in accuracy between the LSTM, CNN, BI-lSTM, and FNN models. The following are a few potential explanations for the variations in accuracy

Model Architecture: Text and other sequence data are the specialty of LSTM (Long Short-Term Memory) models. They have the ability to identify temporal dependencies in the information. Geospatial patterns in data can be effectively captured by CNN models. Even while FNN (Feedforward Neural Network) models can learn intricate patterns, LSTM or CNN models might perform better with sequence data. Complexity and Capacity: Compared to FNN models, LSTM and CNN models usually possess a greater ability to acquire intricate patterns. Because LSTM and CNN designs are designed with additional layers and parameters, they may capture more complex patterns in the data. Effective representation

of the data may be difficult for FNN models, particul arly if this length is important for prediction. Lower performance could result from FNN models' inability to adequately represent the sequential information included in the text input. Training Data: A model's performance can be greatly impacted by the quantity and caliber of training data. The FNN model could not generalize effectively to new cases if it is not trained on a sufficient amount of representative and diverse data. Hyperparameter tuning: Factors like learning rate, dropout rate, batch size, and so on have a significant impact on how well deep learning models perform. Performance differences may result from hyperparameters that are adjusted differently for each model. When a model learns to memorize the training data rather than drawing generalizations from it, this is known as overfitting. Because LSTM and CNN models are good at capturing patterns, they are less likely to overfit. FNN models, however, may be more prone to overfitting, particularly if the model capacity is large in comparison to the volume of training data. Random Initialization: Neural network models typically have random initial weights. Variations in initializations can result in distinct convergence behaviours during training, which can impact the accuracy at the end. You may need to evaluate your LSTM, CNN, and FNN models' performance on test or validation data, examine the training curves, and experiment with other architectures.and hyper parameters in order to pinpoint the precise causes of the discrepancies in accuracy between them.

The CNN model was the most accurate among them, almost 92%, precisely because of its Conv1D and MaxPooling layers that catch spatial patterns very well. The LSTM and BiLSTM modes seem to perform even better, with a 90% accuracy rate.

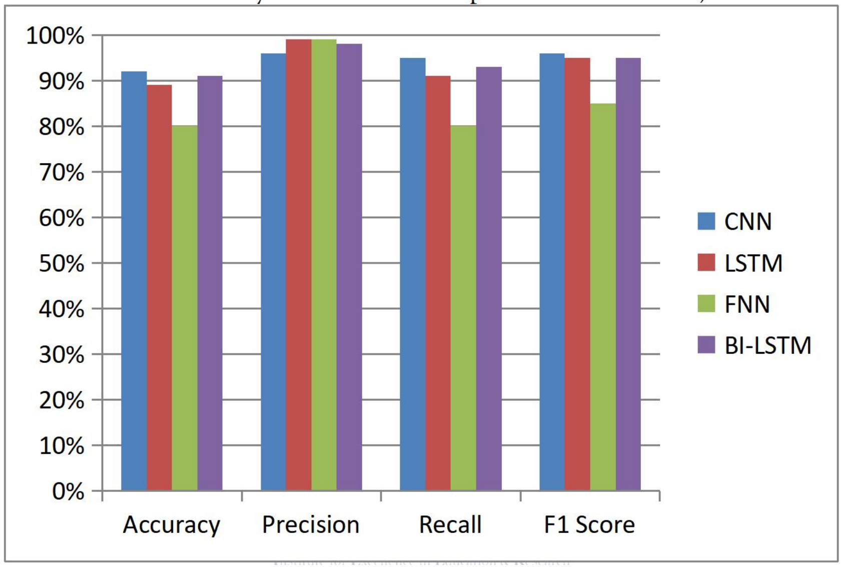


Figure 9

Results

Hence, this research's objective of developing a procedure to compare various deep learning models is helpful in outlining their relative strengths and ready for application in solving real-world problems.

IV. CONCLUSION AND FUTURE WORK

Ultimately, the CNN model outperformed the other models under evaluation, exhibiting excellent accuracy and quick training times. At almost 92%, the CNN model had the highest test accuracy. The Conv1D and MaxPooling layers in this model's architecture helped it perform better by enabling it to capture spatial patterns. Furthermore, CNN models are computationally efficient because they usually require less training time than LSTM models. At about 80%, the FNN model's accuracy was the lowest. The intricate sequential patterns in the data were difficult for the FNN model to grasp because it was a simpler feed forward neural network. This decreased performance suggests that more sophisticated designs, such as CNN or LSTM, are more appropriate for this kind of work. However, aspects like intractability, training time, and computational resources should also be taken into account when selecting a model for deployment.

To further enhance model generalization, it is advised that future study investigate hyper parameter tuning in greater detail, experiment with ensemble models, and look at other data augmentation methods. The study also emphasizes how crucial it is to use cutting-edge architectures for malware detection jobs involving complicated sequential data.

REFERENCES

[1] A. Nasif, Z. A. Othman, and N. S. Sani, "The Deep Learning Solutions on Lossless Compression Methods for Alleviating Data Load on IoT Nodes in Smart Cities," vol. 21, no. 12, pp. 4223, 2021.

[2] M. M. U. Rathore, A. Paul, A. Ahmad, B.-W. Chen, B. Huang, and W. Ji, "Real-time big data analytical architecture for remote sensing application," vol. 8, no. 10, pp. 4610-4621, 2015.

[3] A. Abdallah, M. K. Ishak, N. S. Sani, I. Khan, F. R. Albogamy, H. Amano, and S. M. J. C. M. C. Mostafa, "An Optimal Framework for SDN Based on Deep Neural Network," vol. 73, no. 1, pp. 1125-1140, 2022.

[4] E. S. Alomari, R. R. Nuiaa, Z. A. A. Alyasseri, H. J. Mohammed, N. S. Sani, M. I. Esa, and B. A. Musawi, "Malware detection using deep learning and correlation-based feature selection," vol. 15, no. 1, pp. 123, 2023.

[5] G. Laurenza, L. Aniello, R. Lazzeretti, and R. Baldoni, "Malware triage based on static features and public apt reports." pp. 288-305.

[6] C. Yin, H. Zhang, M. Cheng, X. Xiao, X. Chen, X. Ren, and P. Bogdan, "Discovering Malicious Signatures in Software from Structural Interactions." pp. 4845-4849.

[7] A. G. Kakisim, M. Nar, N. Carkaci, and I. Sogukpinar, "Analysis and evaluation of dynamic feature-based malware detection methods." pp. 247-258.

[8] S. J. a. p. a. Talukder, "Tools and techniques for malware detection and analysis," 2020.

[9] Y. Zhang, and Y. Gu, "A Survey of Traditional and Machine Learning-based Malware Detection Techniques." pp. 1373-1378.

[10] M. Sewak, S. K. Sahay, and H. Rathore, "Comparison of deep learning and the classical machine learning algorithm for the malware detection." pp. 293-296.

[11] U.-e.-H. Tayyab, F. B. Khan, M. H. Durad, A. Khan, Y. S. J. J. o. C. Lee, and Privacy, "A survey of the recent trends in deep learning based malware detection," vol. 2, no. 4, pp. 800-829, 2022.

[12] D. Gibert, "Machine Learning for Windows Malware Detection and Classification: Methods, Challenges, and Ongoing Research," Malware: Handbook of Prevention and Detection, pp. 143-173: Springer, 2024.

[13] R. Damaševičius, A. Venčkauskas, J. Toldinas, and Š. J. E. Grigaliūnas, "Ensemble-based classification using neural networks and machine learning models for windows pe malware detection," vol. 10, no. 4, pp. 485, 2021.

[14] J. Pavithra, and J. F. Josephin, "Analyzing various machine learning algorithms for the classification of malwares." p. 012099.

[15] R. Vyas, X. Luo, N. McFarland, and C. Justice, "Investigation of malicious portable executable file detection on the network using supervised learning techniques." pp. 941-946.

[16] T. Shi, R. A. McCann, Y. Huang, W. Wang, and J. J. S. Kong, "Malware detection for internet of things using one-class classification," vol. 24, no. 13, pp. 4122, 2024.

[17] J. Zhang, Z. Qin, H. Yin, L. Ou, K. J. C. Zhang, and Security, "A feature-hybrid malware variants detection using CNN based opcode embedding and BPNN based API embedding," vol. 84, pp. 376-392, 2019.

[18] Y.-M. Kwon, J.-J. An, M.-J. Lim, S. Cho, and W.-M. J. S. Gal, "Malware classification using simhash encoding and PCA (MCSP)," vol. 12, no. 5, pp. 830, 2020.

[19] H.-Y. Kwon, T. Kim, and M.-K. J. E. Lee, "Advanced intrusion detection combining signature-based and behavior-based detection methods," vol. 11, no. 6, pp. 867, 2022.

[20] E. Debas, N. Alhumam, and K. Riad, "Unveiling the dynamic landscape of malware sandboxing: A comprehensive review," 2023.

[21] Y. Wu, M. Li, Q. Zeng, T. Yang, J. Wang, Z. Fang, L. J. C. Cheng, and Security, "DroidRL: Feature selection for android

malware detection with reinforcement learning," vol. 128, pp. 103126, 2023.
[22] M. Chaudhary, A. J. N. C. Masood, and Applications, "RealMalSol: real-time optimized model for Android malware detection using efficient neural networks and model quantization," vol. 35, no. 15, pp. 11373-11388, 2023.
[23] E. B. Karbab, M. Debbabi, A. Derhab, and D. J. a. p. a. Mouheb, "Android malware detection using deep learning on api method sequences," 2017.
[24] S. Yue, and T. J. a. p. a. Wang, "Imbalanced malware images classification: a CNN based approach," 2017.
[25] A. Martín, V. Rodríguez-Fernández, and D. J. E. A. o. A. I. Camacho, "CANDYMAN: Classifying Android malware families by modelling dynamic traces with Markov chains," vol. 74, pp. 121-133, 2018.
[26] A. Bensaoud, N. Abudawaood, and J. J. I. J. o. N. S. Kalita, "Classifying malware images with convolutional neural network models," vol. 22, no. 6, pp. 1022-1031, 2020.
[27] U. Kaur, H. Zhou, X. Shen, B.-C. Min, and R. M. Voyles, "RoboMal: Malware Detection for Robot Network Systems." pp. 65-72.
[28] A. A. Darem, F. A. Ghaleb, A. A. Al-Hashmi, J. H. Abawajy, S. M. Alanazi, and A. Y. Al-Rezami, "An adaptive behavioral-based incremental batch learning malware variants detection model using concept drift detection and sequential deep learning," vol. 9, pp. 97180-97196, 2021.
[29] A. Voulodimos, N. Doulamis, A. Doulamis, E. J. C. i. Protopapadakis, and neuroscience, "Deep learning for computer vision: A brief review," vol. 2018, no. 1, pp. 7068349, 2018.
[30] M. H. Sazlı, "A brief review of feed-forward neural networks," vol. 50, no. 01, 2006.
[31] C. Avci, B. Tekinerdogan, C. J. C. Catal, C. Practice, and Experience, "Analyzing the performance of long short-term memory architectures for malware detection models," vol. 35, no. 6, pp. 1-1, 2023.
[Figure 1] Surakhi, O., Zaidan, M. A., Fung, P. L., Hossein Motlagh, N., Serhan, S., AlKhanafseh, M., ... & Hussein, T. (2021). Time-lag selection for time-series forecasting using neural network and heuristic algorithm. Electronics, 10(20), 2518.
[figure 2] https://shareedilara.blogspot.com/2023/10/22-cnn-architecture-diagram.html .
[figure 3] Napagoda, N. A. D. N., & Tilakaratne, C. D. (2013). Artificial neural network approach for modeling of soil temperature: a case study for Bathalagoda area. Sri Lankan Journal of Applied Statistics, 13.
[figure 4] https://www.geeksforgeeks.org/bidirectional-lstm-in-nlp/.