الجامعة الوطنية
National University
Science & Technology للعلوم والتكنولوجيا

DEPARTMENT OF ELECTRICAL & COMMUNICATION ENGINEERING

THESIS TITLE

MALWARE ATTACKS CLASSIFICATION USING DEEP LEARNING

A MASTER THESIS

Submitted by

Samiya Al-Shukaili

Student No: NU190678

Supervised By

Mr.Saleh Al Arimi and Dr.Vijayalakshmi.

A Thesis Submitted in partial fulfilment of

The requirement for the Degree of

Master of Science

In

Information Technology

Academic Year: 2020-21

DECLARATION

Declaration by the PG Student

I declare that this ***Master Thesis*** titled **MALWARE ATTACKS CLASSIFICATION USING DEEP LEARNING** Is my own work and has not been submitted in any form for another degree or diploma at any university or other institutions of tertiary education. Information derived from the published work of others has been acknowledged in the text and a list of references is given. I am fully aware of the College's policy on plagiarism and cheating, and that the penalty for submission of plagiarized report could result in a 'fail' in Dissertation. I have submitted a copy of this full report in electronic form to my supervisor.

Signature of the Student: Samiya Date: 22/08/2021

Name of the Student: Samiya Mohammed Suliman Al -Shukaili

Student Number: NU190678

Certificate by the Supervisor

The ***Master Thesis*** titled ____

____

is the bonafide work of Mr./ Ms. ____, and bearing student number _______________ carried out under my supervision. I certify that the work presented in the project report / dissertation is carried out by him / her, and that he / she has achieved the set objectives of the project / dissertation. Information derived from the published work of others has been acknowledged in the text and a list of references is given at the end of the report. I have personally checked this final report for originality / plagiarism through the *SafeAssign* tool on the Blackboard and, to the best of my knowledge and belief, satisfied that the report is free from plagiarism.

Signature of the Supervisor: Date:

Name of the Supervisor:

Countersigned by HoD:

الــجامـعـة الوطـنيـة
National University
Science & Technology للعلــوم والتكنـــولوجيا

COLLEGE OF ENGINEERING

Agreement between Student and Staff on IPR of Final Year Project

I am fully aware that, CE recognizes that student, associated staff and CE jointly own the IPR in any work that we produce as part of our course of study/employment at CE, since College resources and academic expertise from College are utilized during this process. In this regard that

“I Mr./Ms. **Samiya Mohammed Al- Shukaili** declare that research / Technical project carried out under the supervision of the faculty member **Mr.Saleh Al Arimi and Dr.Vijayalakshmi** in the Department of Electrical and Communication Engineering during my normal course study in the Semester - B of the Academic year 2020-21at CE will abide by the College IPR policy.

With regard to the revenue generated by that IP due to its commercialization; I do understand that I will be treated and rewarded as per the CE-Research Policy. I agree to the terms and conditions that will be negotiated depending on the conditions applicable for the commercialization of a particular project and agreed portion of the total revenue (lump sum payment, royalty or any other form) accruing from the commercial exploitation of IP jointly owned by me, associated staff and CE shall be credited to CE.

Student Name:	Samiya Al-Shukaili	Project Supervisor Name:	…………………… ………………
Signature & Date:	Samiya 22/08/2021	Signature & Date:	…………………… ………………

ABSTRACT

No one can ignore the value of the cyber security in any organization that protecting their valuable assets from cyber criminals. Cyber-attacks are increasing in every minute in every single day. It has gain a sight impact on people lives and organization loss. To avoid these, organization are spending huge efforts and cost for every signal attack. In deeded, cyber security tools such as firewall, intrusion detection system, intrusion prevention system..etc. These cyber security tools are vulnerable and easy to get hacked with cyber-attacks. Artificial intelligence is now shown to be expanding in multiple areas of technology on a daily basis. It has evolved into a highly powerful tool in the field of cyber-security. Artificial intelligence can be used to detect and classify cyber-attacks. In fact, there are already some studies on machine learning and deep learning that classify the attacks. This thesis, will present an LSTM model to classify Malware-attacks in order give more attention to IT administrators to reduce the impact of cyber-attacks inside the organization. In addition, the Malware API Calls dataset will be collected form GitHub site for cyber security records for previous years. The implementation of the project will be done in python using Anaconda software with Jupyter Notebook. In addition, the developed model will be trained and tested using Tensor flow and keras libraries. Then, the developed model will be evaluated by confusion matrix and statistics parameters such as Accuracy, Precision, Sensitivity as well as Specificity and F score1. The obtained results of proposed LSTM model has achieved overall accuracy 92.59% satisfactory score. The proposed will be demonstrated that efficiently classify online Malware dataset with a clear comparison result with previous researchers work in order to study the difference in accuracy level. Finally, some recommendation will be given in order to improve quality of the proposed LSTM model and how efficiency will reduce negative effects of the organization cyber-attacks.

Keywords: Cyber-Attacks, Malware Attacks, Needs Malware Attack Classification, Needs of Deep Learning Techniques, LSTM Algorithm, API call sequence.

AKNOWLEDGMNETS

First and Foremost, I would like to thank my first supervisor **"Mr.Saleh Salim Khamis Al Araimi"** for his advice and cooperation throughout the process during this project. And for his kindness as well as his great ideas. He volunteered his personal time to discuss all these concepts. This project would not have been completed without his great assistance and expertise.

Second, I'll never forget to appreciate "**Dr. Vijayalakshmi** " my second supervisor, for giving me the strength and motivation to complete this project. Her cheerfulness and kindness were much appreciated.

Beside my Advisors, I would like to express my deepest appreciation to "**National University of Science & Technology"** for supporting me and giving me this opportunity to complete my master. And helping me to complete this thesis and other research papers.

I would like to thank our Head of department "**Mr. Ali Abdullah Hamed Al Mahruqi** ". His words of support and collaboration have been really helpful.

I am grateful to my closer friend **"Ms. Nasreya Nasser Salim Al Hinai**" for her encouragement and invaluable assistance.

Last but not least, I extend my appreciation, pleasure, and respect to my husband for his continuous encouragement and advice throughout the project.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER1 : INTRODUCTION:

1.1 Cyber-Attacks:

No one can deny the number of the cyber-attacks is getting increased in every minutes through the virtual world which causing huge effort and big loss on individual as well as organization side. According to (Oman Observer, 2021) the ministry of technology and communication in Sultanate of Oman in 2019, it has reported to more than 14 billions of hacking attacks. And more than 300 security incidents that were discovered and prevented. In deeded, the need for classifying these cyber-attacks has become very powerful tool in order to detect cyber-attack in a quick response and reduce the negative impact on the organization. In fact, there has been some traditional techniques which classify cyber-attacks but there has been some disadvantage and difficult with massive dataset files. There has been some security tools like IDS/IPS, Firewall and many other which are rule based system to detect cyber-attack that is already defined in the database. In fact, there are some previous studies used to classify the cyber-attack using different the machine learning algorithms.

1.2 PROBLEM DEFINITION:

The main problem is as the number of attacks are getting increased, most the organization are spending their fund and efforts for cyber security tools to protect their system and database. Unfortunately, those cyber security tools are vulnerable to get hacked. Also, the CIA Triad in cyber security which means confidentiality, integrity and Availability. Confidentiality means that the information is getting only accessed by authenticated users. And second, Integrity is to insure that the attacker is not able to hijack the information and he don't have the rights to modify the information. Then, Availability, the information should be available all time. So, the organization should keep tracks who is accessing and modifying that information. More ever, cyber-attacks are having a negative impact on the organization because they block the day-to-day operations which costing huge money and efforts. This organization should have a priority to negative impact of the cyber-attacks based on attacks classification. The best solution by applying the deep learning Techniques, the attacks could be identified and separated into multi classes to be easy identified and prevented. Based on the existing DL model, we will build a new model to provide a comprehensive efficient classification model that can classify network traffic records of massive traffic dataset into different classes.

1.3 Malware attacks and The Needs of classifying malware attacks:

Nowadays malware has become as the destruction weapon in cyber security space. Malware attacks are used in many defense area like Distributed denial of service, spam, virus and more other. In fact, it is used for propagation or to communicate to the command control of the server to receive the commands. Basically most of cyber-attacks are using the malware attacks to hack the system in order to make the disruption or to steal the data. So, properly the malware is the most sophisticated software used by the attackers to lunch their attacks. For that reason, it is very essential thing to detect these malware. In addition, the existing systems are not capable enough to detect all malware and it take much time for detecting them. So, there is a massive needs to build an effective system that are capable for detecting the new malware.

1.4 Needs of Deep Learning:

No one can deny that the needs for deep learning in cyber-security is getting increase for many reasons. First, number of attacks are getting increased in every minutes. So, the need for strong measures which it can be perform using deep learning. Second, nowadays this cybercrime is on sale. Professional cyber criminals are playing very smart. They are selling these customized hacking as a services. Third, now every device is generating logs. So, there is more data in network logs and which is very difficult to analyze massive data manually. Fourth thing is intrusion detection system and intrusion prevention system are mostly pattern based devices which means that basically detecting and preventing the attack that it is already saved in the database and whenever new attack will come will compare it with what is stored in the database but if it is zero day attack is coming, it will be failed to detect it because that pattern is not saved in the database. Fifth thing is more IOT devices are getting increases every day and everything is smart device. And those smart devices are creating something new which called big data. So, it is necessary to protect this data in cyber-security by using Deep Learning Techniques.

1.5 LSTM algorithm:

Long term short memory is a sort of recurrent neural network which has the feedback connections was defined according to (Supriya Shende, 2020).To illustrate more, long short term memory model has the solution in order to avoid any long term dependencies because LSTM has feedback connections and design. In addition, they defined the LSTM unit contain 4 main parts which are cell, Input gate, output gate and forget gate. The LSTM cell is used to remember or forget the information.

1.6 Application Programming Interface call sequence:

The best choice for feature extraction in malware classification is an application programming interface. In addition, Malware will also spread to a large variety of various lengths, times, and types of malware. As a Consequence, long short Term Memory will be best approach in order to classify Malware attacks with high accuracy. LSTM model is efficient at processing time series data like API call sequences.

1.7 Project Aim:

The main aim to develop a Long term short memory model to classify malware attacks.

1.8 Project Objectives:

- To review pervious works in Malware classification using deep learning Techniques.
- To collect malware dataset records for previous years.
- To learn the implementation of deep learning techniques in python.
- To develop an LSTM model and train the developed model to classify malware families.
- To test and validate the model to classify malware into different classes.
- To demonstrate the results of neural network model that efficiently classify online Malware dataset.

1.9 PROJECT SCOPE AND SIGNIFICANCE:

Classification of cyber-attacks using deep learning will help to act as proactive measures for cyber-attacks. This project can be used in any organization that are dealing with cyber-attacks to protect their valuable assets. If any organization wants to analyze and monitor their network in order to create the dataset. Then, the attacks classification can be applied based on their negative impact.

1.10 PROJECT CHALLENGES:

There are some challenge while applying neural deep learning in cyber-security attacks. First of all, lack of labelled samples of the cyber-attacks. Because we having more zero day attacks. So, we don’t having the knowledge of this attacks. Second challenges is very large dataset size with massive data. Third thing is the Data preprocessing which have many thing to apply on the dataset to convert into a readable format in which we can apply the analysis.

1.11 INNOVATION:

The following is a summary of the paper's major innovation:

- Presenting a detailed and effective classification model for classifying cyber-attack records in a cyber-security attacks dataset using deep learning algorithms. Specifically, using long short term memory.
- Presenting detailed preprocessing operations for the collected dataset records in order to convert into readable format.
- Providing detailed Experimental results as well as full development and validation processes.
- Providing a detailed performance analysis, including a confusion matrix and class statistics to learn more about the system's efficiency such as Accuracy, Precision, Sensitivity, Specificity and F score1.
- Comparing the obtained result of the proposed LSTM model with previous researchers work.

1.12 Summary:

Big companies are spending a lot of money in order to detect from cyber-attacks in every year. In order to give an extreme surveillance, this project is a model based deep learning will be developed for classification of Malware. This classification of malware will be achieved by a long Short Term Memory model for better performance. Then, the developed model will be evaluated by some parameters like as precision, accuracy, sensitivity as well as specificity and F score1. The implementation of the project will be done in python language using Anaconda Navigator software with Jupyter Notebook interface.

1.12.1 Thesis Outline:

- Chapter 1: Describes the problems of cyber- security attacks and its solutions. And it gives a details look into research idea about malware classification based on deep learning techniques.
- Chapter 2: Describes the Literature review with overview of the cyber-attack detection concept, different machine learning and deep learning classification methods. And it give a closer view of long short term memory model of the previous researchers work. At the end, it present a comparison study between our proposed LSTM model and previous researchers work.
- Chapter 3: Describes the methodology of proposed LSTM model with details of dataset collection and the Requirements Tools. As well as, it describes the methodological framework of proposed LSTM model for malware classification.
- Chapter 4: Presents a slight view of the importance of python libraries which used for creating the proposed LSTM model.
- Chapter 5: Presents a detailed analysis of training and testing the proposed LSTM model. In addition, it shows the evaluation process of the proposed LSTM model. And at the last, it gives a comparison study based on the obtained result of the proposed LSTM model with other similar researcher work.
- Chapter 6: concludes the summary of the proposed LSTM model with some suggestion recommendations in order to improve the quality of the proposed LSTM model.

CHAPTER2 : LITERATURE REVIEW:

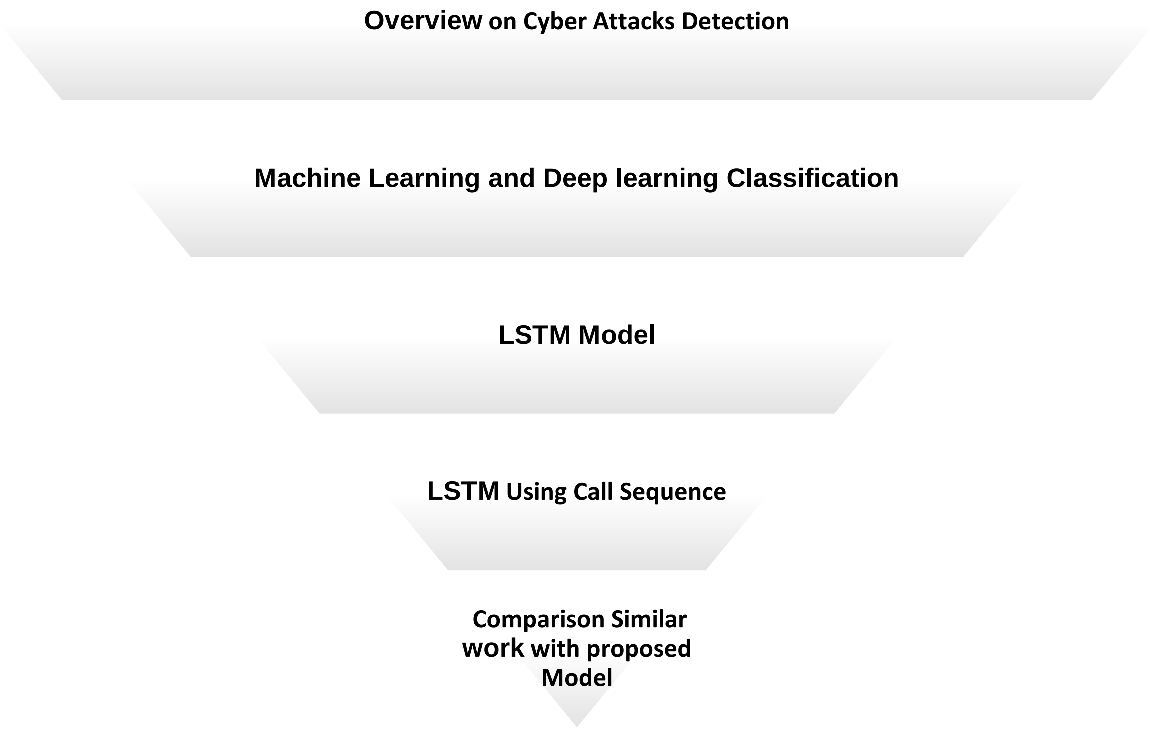

Overview on Cyber Attacks Detection
Machine Learning and Deep learning Classification
LSTM Model
LSTM Using Call Sequence
Comparison Similar work with proposed Model

Figure 2.1: Literature Review structure

2.1 Introduction: Overview on Cyber Attacks Detection:

No one can ignore that the cyber-attacks are becoming a serious issue for each organization. In fact, there are danger challenges in cyber-attacks detection. Almost all organizations spending their funds to use some detection solution in order secure their infrastructures against these cyber-attacks. But still these detection needs to improve with advanced attacks. With the development of cyber-attacks on the cyber space, the number of the malware attacks are getting increased in every minute especially with unknown attack. As A consequence, there is a dangerous problem which needs a larger attention to look for these new malware and analyze them. For this Issue, there has been some existing approaches on malware classification using signature based and Behavior based in order to classify malware attacks. In fact, there are different methods that have been used like image classification, binary, opcode, sequence call, and others.

(Lee and Choi, 2017) developed malware classification based on anti-virus software in order to analyze the malware detection model. In addition, their methodology was achieved by taking the keyword for each malware from the signature based anti-virus as character for the malware group definition. As a result of their proposed model, the classification of malware was achieved by 81% of whole malware. However, their developed model need some improvement in order to check the meaning of the keywords before classifying them. Another thing the number of similar groups needs to be reduced.

(Komatwar and Kokare, 2021) conducted a study of previous research studies, including different types of classification, malware types, and malware images. According to their survey on different Machine Learning algorithms, the malware detection was divided into type’s behavior based for Application programming interface call feature and Signature Based for binary. According to their study, the survey was focused on the malware image classification features encryption and decryption and this it could be capable only for malware Images classification.

(Lallie *et al.*, 2021) proposed a time line anylsis for cyber crimes during Covid 19 outbreak. In addition, They also look at the gaps between the initial epidemic in China and Covid 19 and how they relate to cyber attacks. They analyezed the announcement and social media stories that attackers used to hack huge number of victims as all the people arouund the world moved to work from home. More ever, Their method was achieved with UK case study of cyber crimes. According to their research, there has been a rise in the number of cyber attacks and criminal activities that attempt to target users by sending files to open or download, which then function as malware. The finding was shown that there was not law enforcement and strong polices to defence these cyber attacks. As consequence, they recommended to give more awareness of cyber crimes to the governments and public people to know how to deal with cyber attacks which their dealing with announcement and social media.

(Alhayani *et al.*, 2021) discussed the most effective techniques to defend against cyber-attacks. More specifically, they presented all the technical steps that help the organizations to detect cyber-attacks and response them such as recovery, encryption, security measures and Wi-Fi security. In other side, they presented all technical steps from users side which controlling and monitoring the user and giving them the awareness. According to these research, the finding was showed that the organization need for active process which act as the response incident which managed during the cyber-attacks. Another thing, there is a massive needs for continuous updating cyber security strategies and plans in order to defense cyber-attacks.

2.2 Existing work in general: Machine Learning and deep learning Classification:

In Artificial Intelligence, there have been numerous classification techniques proposed. Machine learning, on the other hand, is the process through which machines learn to behave like humans without the need for human interference. In addition, there three types of machine learning which are supervised learning for the training the label data. Another type is unsupervised learning which used for unlabeled data. As well as the reinforcement learning act can train the data from the behavior of human beings. More ever, deep learning which is a sort of machine learning but it used for huge data and provide high performance. More specifically, classification is the process of classifying data into different classes. There are different types of classification using machine learning and deep learning techniques such as binary classification, multi class classification, multi label classification and imbalanced classification. There have been a number of previous studies that have used machine learning and deep learning to classify attacks.

2.2.1 Machine learning:

(Churcher *et al.*, 2021) highlighted multiple machine learning algorithms in order to compare which algorithm is the best for attacks classification using machine learning. Their methodology was tested on binary classification and multi class attacks classification datasets. In addition, several metrics were used to evaluate their performance, including accuracy, precision, recall, F1 score, and log loss. More specifically, the obtained result was found that the best algorithm for binary classification was Random Forest Algorithm with 99% accuracy for DoS and HTTP attacks. In contrast, the k-nearest neighbor algorithm and artificial Neural Network algorithm were the best Algorithm for Multi class classification attacks which is 4% higher than Random Forest Algorithm. According to their result, the proposed model was more focused on multi class classification with weighted dataset.

(Thakkar and Lohiya, 2021) stated that the needs of the attacks classification in network security as become very vital things. In addition, they mentioned that this classification will enhance the security measurement and security mechanisms be protecting the valuable information in the network. They demonstrated how to use machine learning classifiers to select features techniques. Their methodology involved applying the FS algorithm to the NSL-KDD dataset, which focused on Ratio Features such as filter, wrapper, and embedding. With measured metrics including accuracy, recall, F-score, and precision, the end result was highly successful and useful to the system. Each attack type was examined separately, and the accuracy was determined to be 99.6% for DoS, 98.9% for Probe, 98.2% for R2L, and 99.1% for U2R. According to their developed model, the collected dataset was having some limitations.

2.2.2 Deep learning:

(Tama and Rhee, 2017) introduced a deep learning model to classify attacks in wired and wireless networks using three datasets UNSW-NB15, CIDDS-001 and GPRS. In addition, their methodology was tested with multiple datasets, UNSW-NB15 has 42 characteristics, with 31.94 % normal and 68.06 % malicious classes. In addition, Grid Search was used to test the results, which included accuracy, precision, recall, and false alarm rate. The outcome was 100 % accurate, but they also recommended that attack detection might be more effective if it was tested using a variety of validation methods and datasets. According to result, the finding of the developed deep neural network was not able to find the difference between the deep neural network and other algorithms.

(Alom *et al.*, 2018)presented a brief survey on deep learning approaches which covered a variety of algorithms. More ever, they evaluated different Deep learning algorithms on different application domains. They stated that the Artificial Intelligence has been discovered since 1950 by McCarthy while Machine learning has been discovered1980. In fact, Deep learning is the subset of machine learning based on artificial neural network and it was developed from 2006 onwards. In addition, Different methods were used on this overview which included SDK framework, standard development kits and bench marks datasets. Furthermore, their survey was conducted with deep learning techniques like supervised learning, unsupervised learning and reinforcement learning. The outcome of this survey was concluded that Deep Learning is the Learning techniques by estimating model parameters in order for the trained model to complete a task and perform the result. According to their study, their resource of the survey was collected from multiple algorithms from supervised, unsupervised and reinforcement learning.

(Das and Morris, 2018) provided a survey on traffic classification, intrusion detection, and e-mail filtering using machine learning algorithms. In addition, they also used a variety of datasets to determine which the best cyber security dataset was for the researchers. They used the MODBUS and ICS datasets, as well as four machine learning algorithms: J48, one R, Naive, and Random Forest. According to their findings, the J48 algorithm had the greatest accuracy among the algorithms tested, while Random Forest coming in second. Their survey was focused only on the machine learning algorithms.

(Bagui *et al.*, 2019) created a hybrid feature selection to categorize uncommon cyber-attacks using two classifiers, machine learning nave Bayes and decision tree J48. In addition, they used the UNSW-NB15 dataset, which was generated using IXIA Perfect Storm. By using their approach, they were able to classify cyber-attacks using a mix of k-means clustering and feature selection. In comparison to decision trees, Nave Bayes was able to improve classification accuracy. While decision trees were able to achieve a higher classification rate for all attack families. According to their developed model their approach was only focused on machine learning and one dataset.

(Maleh, 2019)has introduced the problem of malware classification. In addition, he proposed a scalable method for classifying malware into family groups using a convolutional neural network and a recurrent neural network with long short term memory. Additionally, the method was developed by conducting a case study for the BIG Cup 2015 using the Microsoft malware dataset. On the validation data, the obtained result showed that the achieved accuracy was 98.73% and the average loss was 0.0698.

(Ferrag *et al.*, 2020) presented a survey on deep learning approaches for cyber security threats detection. In addition, they used a seven-category classification system to classify 35 cyber datasets. Furthermore, they also looked at seven deep learning models and compared their performance in binary and multiclass classification using the CSECIC-ID2018 dataset and the Bot IOT Dataset. More ever, the achieved result was calculated using a variety of metrics, including accuracy, false alarm rate, detection rate, and indicators. The obtained results was found that the Deep Neural Network model was having high true Negative rate. And as the same time Recurrent Neural Network model was shown a high detection rate for 7 different attacks. The last but not least according to the findings, the Convolutional Neural network model was found to have the greatest detection rate for 4 different attacks. Their study was fully mixed the supervised learning and unsupervised learning algorithms.

(Nadu, 2021) has investigated several types of detection for cyber-attacks using deep learning techniques. As a result, he defined network protection as the methods for preventing unwanted access to a network. As a consequence, he performed a study using two deep learning algorithms to find the top Deep learning approaches that have been utilized in the last three years. In addition, the method of proposed model of deep learning was by developed framework using KDD-dataset. Then the dataset was splitting into train and test data. Then choose the best model either recurrent neural network or convolutional neural network. Finally, the obtained result according sigmoid activation function and 1 dimension Convolutional Neural network was achieved the accuracy with 0.732%. On other hands, the accuracy for recurrent neural Network was 0.4057%. According to his research, each technique has certain limitations in terms of detecting specific patterns and terms using an intrusion detection system.

2.3 LSTM Model:

Long short term memory is a type of recurrent neural network that performs better than standard recurrent neural networks in terms of memory. In fact, due of its higher accuracy, LSTM is ideal for classification, particularly for call sequences. In addition to that, LSTM is used a supervised learning methods to predict the data. There have been a number of previous studies that have used LSTM to classify attacks.

(Vinayakumar *et al.*, 2018) proposed long short term memory model for android malware detection with static and dynamic analysis. Furthermore, the technique for the created model was done by employing the APK Tool for benign and malicious applications to collect dataset from MalGenome. In addition, there were 279 malicious apps implemented in the Linux operating system using Keras and Tensor flow. Additionally, all of the approaches were developed using an LSTM framework that consisted of 32 memory blocks, one cell, 1000 epochs, and a learning rate of 0.1. As a consequence, the LSTM keeps track of long-term dependencies. The obtained result was achieved better performance than the traditional recurrent neural network. More specifically according to the result, the finding showed that the Android malware detection for static analysis was achieved with 0.975 and dynamic analysis was achieved with 0.939. According to their proposed model, the LSTM result was showed that there was much super performance than the traditional machine learning algorithms. However, still there are some limitations on the findings, if it is better to implement on real malicious applications.

(Yang *et al.*, 2020) proposed LSTM model in order to classify the malicious codes using Java Scripts. In addition, they also presented a study on how to identify different types of malware using Java script. More specifically, the provided model's method was achieved by implementing 174 byte code feature utilizing V8 engine, an open source java script written in C++ and launched in Google Chrome. More important, the method was accomplished by providing word vector to extract the word feature, which was provided as an open source by Google in 2013. Then, the LSTM model was proposed with 39 epochs, 32 batch size and 128 for embedding dims. As Consequence, the developed system was tested and compared with four different algorithms which include Random Forest, Support Vector Machine, Naive Bayes and LSTM. The observed result show that the overall amount of byte codes was 100,000. However, several byte code features were duplicated, leading to poor result. In addition, the performance accuracy of LSTM model was 99.57% and F-score was 98.37%.The Last thing but not least, according to their developed model, the LSTM model was having better result than support vector machine and random forest.

(Muhuri *et al.*, 2020) developed an LSTM model for detecting and classifying cyber-attacks. In addition, the approach was also accomplished by categorizing the NSL-KDD Dataset and combining various selection features for the LSTM model. Furthermore, their proposed LSTM model was categorized into binary and multiclass classification with five attack types using the NSL-KDD Dataset, including Normal, DoS, Probing, U2R, and R2L. Moreover, the classification performance was measured using metrics such as accuracy, recall, precision, F-score, and confusion matrix. The obtained result was compared with support vector machine and Random Forest as well. The finding was shown that the accuracy for multi class classification was higher than the support vector machine and random forest. However, the accuracy for binary classification was similar to random forest and higher than the support vector machine. Last but not least, the achieved result was shown that the accuracy for binary classification with122 attacks features was 82.68% while the accuracy for multi class classification for 99 attacks feature was 93.88%. According to their result, they used LSTM with more than one method to classify the cyber-attacks.

(Dang, Di Troia and Stamp, 2021) presented four different LSTM models to identify 20 malware families. They used malware old dataset 2015 and new malware 2021. Furthermore, they achieved their methodology by applying opcodes and natural language processing features such as word embedding, Bidirectional Long Short Term Memory, and Convolutional Neural Network. The findings showed that long short term memory is extremely powerful, according to the proposed model, but one of the LSTM layers had a bad outcome. Last thing but not least, according to the acquired results was showed excellent performance because of different LSTM models such as word embedding, bidirectional long short term memory models, and convolutional neural network.

(Thapa and Duraipandian, 2021) developed a long short term memory model for detecting malicious traffic. In addition, the proposed model was developed using a methodology that included many experimental setups. The proposed model was additionally validated using a real-time testing environment that included a wireless network, a laptop, an attacker device, and an external traffic monitoring device. According to experimental result, the finding was showed that LSTM model was very efficient and the performance was increased 5% than other traditional previous models. Last but least, the accuracy for developed LSTM model was achieved with 99.5% in detecting the malicious traffic. According to their developed model with massive dataset and the number of epoch was reaching 100, the accuracy was very effective.

2.4 Focused on: LSTM Using Call Sequence

Call sequence classification is the most popular technique with Long Short Term memory. As consequence, all best achieved performance was using the dynamic call sequence in order to classify malware attacks. There has been several studies on LSTM recurrent neural network and convolutional neural network which focusing on the call sequences.

(Maulana and Kusuma, 2020) developed two deep learning algorithms which are long short term memory and Nested LSTM model in order to classify malware attacks based on their call sequence. More ever, their methodology of proposed model was achieved by applying word2vec features with malware dataset from two resources Github and Virus share. Furthermore, they used dynamic Malware analysis for malware execution to analyze 13356 samples using cuckoo sandbox. According to the outcomes, the finding was showed that the LSTM was having better performance more than the nested LSTM. More specially, the accuracy for LSTM was 98.6% while the accuracy of Nested LSTM model was 93.11%.The achieved result was compared with support vector machine model which is one of machine learning models as the bench mark. The last but not least, comparison result was showed that the deep learning algorithms was having high level of accuracy for malware detection rather than the machine learning algorithms.

Long short term memory sequential model was created by (Catak *et al.*, 2020) in order to identify malware attacks based on Application Programming Interface Sequence Call using Windows Operating System. More specific, they used call sequence model for malware classification to demonstrate how malware could act correctly. Their methodology of the proposed LSTM model was achieved using Call Sequence Technique by developing a new dataset for malicious file which it is available in Github source. In addition, they used Cuckoo sandbox free open source in order to run and analyze the malicious software. The obtained result was compared with other machine learning algorithms. The finding was showed that LSTM model was achieving a satisfactory result with accuracy 95% and F-1 score with 0.83.

(Girinoto *et al.*, 2020) presented a deep learning framework in order to compare three LSTM Architectures for Malware to classify these malware attacks from Windows Application programming interface for call sequence. More specifically, they developed three different models which are Sequential model, Bidirectional LSTM model and a concatenated model that combined the first and second model. Furthermore, their approach was created using the LSTM architecture and several circumstances like as call back, batch normalization, dropout, and attention mechanism. According to the proposed model, the resulting performance was improved in adding dropout and attention mechanism scenarios as it decrease the loss value and increase the attention on strength information using LSTM and Bidirectional LSTM Models.

(Schofield et al., 2021) proposed a convolutional Neural Network to classify malware attacks based on Application programming interface call Sequence using windows system. The technique behind the presented model was built using a database of 5385 API call streams categorized as eight different types of malware. More specifically, the feature extraction approach for CNN was achieved using a one-dimensional CNN using two methods. First, Categorical Vector for binary classification and the second method was using Term frequency – inverse frequency vector for multi class malware classification. In addition, the their dataset was having 7,107 samples of malicious files of different types of malware such as Trojan, Back door, Adware, Dropper, Virus, Worms, Downloader and Spyware. According to their obtained result, Categorical Vector had a little greater accuracy than Term frequency – inverse frequency vector. Furthermore, the outcome was compared to machine techniques including support vector machine, logistic regression, K-nearest neighbor, and Random forest. The researchers found that CNN outperformed than other standard model, with an accuracy of 91.0 %.

2.5 Comparison Similar work with proposed LSTM Model:

Nowadays number of attacks are getting increased every minute in each organization. Therefore, the needs for detecting the attacks becomes very essential part in order to prevent these malicious attacks. Deep learning techniques are effective technology that are widely used in cyber security field in increase the security measurement level with incredible technology. Detection/classification of malicious attacks will help to reduce time in order to identify the type of malicious attacks.

(Hwang *et al.*, 2019) developed an LSTM based on deep learning techniques in order to classify malicious traffic at the packet stage. They proposed an LSTM framework that takes word embedding to classify malicious traffic. Specifically, they developed a dynamic word embedding technique and they used LSTM to learn the correlation structure between fields in the packet header and classify whether an incoming packet is normal or malicious traffic. More ever, they proposed model using multiple datasets with PCAP format and the size of 3.71 GB. Actually, they used ISCX2012, USTC-TFC2016 and IoT dataset from Robert Gordon University. Also, they used IoT dataset collected on their network Mirai bonet. In fact, they used one of the most well-known datasets is USTC-TFC2016 with ten types of malicious traffic from online website. As well as they used ten types of normal traffic which were collected using network traffic simulating IXIA BSP. They summarized their dataset in forms of malicious and normal traffic in terms of Server Massage Block, Peer to Peer and Instant Message. In addition, they developed four different types of attack traffic TCP SYN (41 GB), TCP ACK (2.4 GB), HTTP POST (103 GB), UDP (127.06 GB) from total of 667 GB attack data. They mentioned that their goal is to classify incoming packets into normal or malicious classes without using Word embedding to extract semantics and syntax features from this sentence. They considered the meaning rather than the whole sentence. In addition, they formulated a series of basic LSTM cells, each of which contains an input gate, an output gate, and a forget gate. They used activation function is the sigmoid function. In addition, they developed a deep learning model using Tensor Flow and Keras in python libraries which run on Ubuntu system 64 bit server with an Intel processor with 2.2 GHz, 32 GB Ram and an NVIDIA RTX Tesla K80. According to their obtained result was found on the developed model train 60% and 40% dataset. They found, if the proposed system detects a malicious packet, it will send alarm to the network administrator and direct them to offline traffic classification systems. In addition, they tested the developed model in terms of Accuracy, precision, recall, f1-score, FAR, loss. They evaluated the performance of proposed model using some parameters such as true positive, false positive, true negative and false negative. Specifically, they achieved nearly 100%

accuracy in detecting the malicious traffic. From this paper has been discover that there are some limitation with evaluated performance parameters. In addition, the developed model can be more with measurement of sensitivity and selectivity.

(Supriya Shende, 2020) has addressed Long term memory model based on deep learning techniques for classification attacks in network security. They proposed Long Short-Term Memory model for detecting cyber-attacks. As a consequence, the administrator may react and respond to the alert in order to avoid such actions. More specifically their methodology was trained and tested the detection system using the NSL-KDD dataset which is a modified version of the KDD99 dataset. In addition, they divided the NSL-KDD dataset into train 60% and 40% dataset using Mat lab 2019 b. Their developed model for two types of attacks classification. One for binary classification and second for multi class classification to DOS, U2R, Normal, Probe and R2L. More ever, their performance was evaluated the LSTM model to test the time for binary classification is 8.28 sec and 3.25 sec for the second method multiclass classification. In addition, the optioned result was found the accuracy was 99.2% for binary classification and 96.9% for multi classification. The developed system was evaluated using confusion metrics and the result was found that the Sensitivity was 99.26 % as well as the Specificity was 99.26 % and the False Positive Rate was 0.97. According to the findings in this paper, the classification attacks may cover a range of validation methods and datasets.

(Andrade *et al.*, 2019) an LSTM model has been developed to classify five different kinds of malware. In addition, they had included a significant number of datasets for malware classification. More specifically, their approach was created using an LSTM model that used Malware and Clean ware in a Multi-class dataset containing five types of malware: Backdoor, Root Kit, Trojan, Virus, and Worm. In particular, they proposed LSTM model using 19740 multi class Dataset. More ever, they also implemented the code in a keras environment with an Application Programming Interface on a Windows system, as well as an open source environment called Tensor flow. As a consequence of their developed LSTM model, an estimated time arrival of 1141 seconds per epoch was attained with a 67.60 % accuracy. In addition according to the findings, Rootkit Class has the greatest true positive rate of 92.19 % and the best false negative rate of 7.81 %. The Trojan class, on the other hand, had the worst true positive rate of 51.06 % and the worst false negative rate of 48.94 %. According to their developed model, they could achieve the efficient result using LSTM but still the dataset was very small.

Table 2.1: Comparison similar work with proposed Model

Source	Aim	Methods	Tools	Findings
(Hwang *et al.*, 2019)	To develop an LSTM based on deep learning techniques in order to classify malicious traffic at the packet stage	They proposed an LSTM framework that takes word embedding to classify malicious traffic. Multiple datasets are used with PCAP format and the size of 3.71 GB. Actually, they used ISCX2012, USTC-TFC2016 and IoT dataset from Robert Gordon University. Also, they used IoT dataset collected on their network Mirai bonet.	Tensor Flow and Keras in python libraries which run on Ubuntu system 64 bit server with an Intel processor with 2.2 GHz, 32 GB Ram and an NVIDIA RTX Tesla K80	They mainly focused on packet header to extract packet semantic meaning for malware classification. Thus, they achieved nearly 100% accuracy in detecting the malicious traffic. Because they used multiple datasets and they used huge number of epochs nearly 200 for training dataset.
(Supriya Shende, 2020)	To develop Long Short-Term Memory model for detecting cyber-attacks.	They used NSL-KDD dataset which is a modified version of the KDD99 dataset. They used two types of attacks classification, One for binary classification and second for multi class classification.	They used Mat lab 2019 b	The Accuracy was 99.2% for binary classification and 96.9% for multi classification. Because, they used two types of classification and different classes in the dataset.
(Andrade *et al.*, 2019)	To develop LSTM model to classify five different	They used Multi-class 19740 datasets called malware and cleanware in multiclass	They implemented the code in a keras environment with an Application	The Accuracy was 67.60%. According to their developed model, they could achieve

	kinds of malware.	scenarios containing five types of malware: Backdoor, Root Kit, Trojan, Virus, and Worm.	Programming Interface on a Windows system, as well as Tensor flow.	the efficient result using LSTM but still the dataset was very small.
Proposed Model	To develop a Long term short memory model to classify malware Attacks.	The dataset was collected from GitHub site which called Malware API Call dataset (Application Programming Interface) with more than 7 thousand malwares. This dataset will be divided to multiple malware types in form of TXT files with classes of malware such as Virus, Trojan, Adware, Backdoor, Worms, Spyware, Dropper and downloader.	Tensor flow Library in Python which is the basic library for creating deep learning models. And Keras Library in Python which it is a high-performance architecture based on Tensor Flow 2.0 that can be scalable to massive GPU clusters or a complete TPU.	The achieved Accuracy was 92.59%.

With comparison with previous work, we selected these number of research papers which are closer to the proposed model. The achieved accuracy was higher, because they used more than one method of malware classification and multiple datasets. In other hands, one work was less accuracy because of the limitation on the size of dataset.

2.6 Summary:

In this chapter a number of previous work has been discussed in order to classify cyber- attacks using machine learning and deep learning. It observed that the cyber security is very essential topic in order to overcome the challenges of cyber-attacks.

CHAPTER3 : METHODOLOGICAL FRAMEWORK:

This chapter will describe the proposed methodology for developing LSTM model. In addition, it will present the collection Malware dataset. It will show a detailed description of the structure of LSTM model. More ever, it will present all Requirements tools which needed for developing LSTM. Lastly, it will explain the proposed LSTM model Framework for malware classification.

3.1 Proposed Methodology for LSTM model

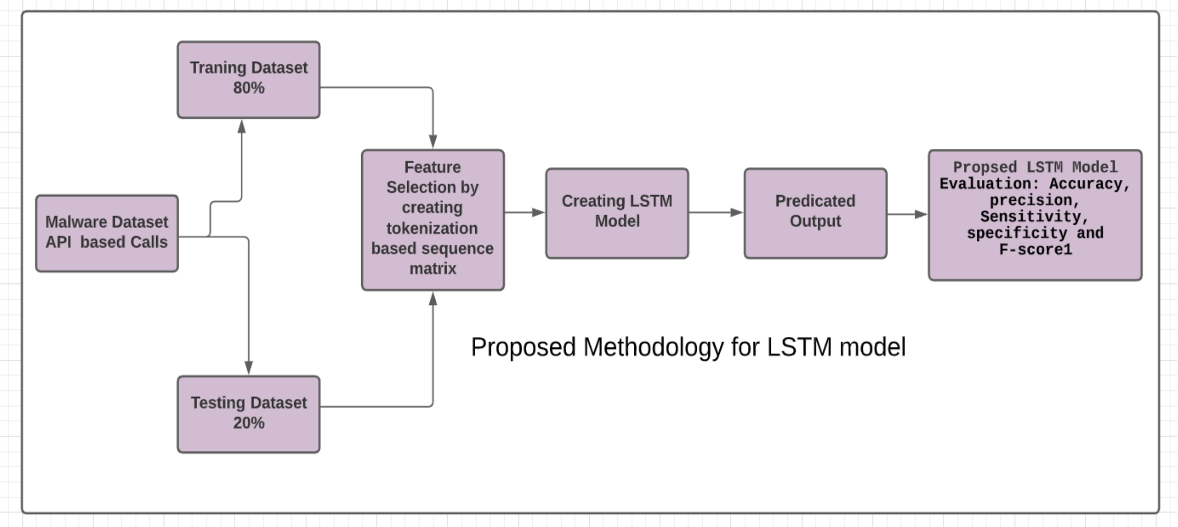

Traning Dataset 80%
Malware Dataset API based Calls
Feature Selection by creating tokenization based sequence matrix
Creating LSTM Model
Predicated Output
Propsed LSTM Model Evaluation: Accuracy, precision, Sensitivity, specificity and F-score1
Proposed Methodology for LSTM model
Testing Dataset 20%

Figure 3.1: Proposed Methodology for LSTM model

As shown from figure 3.1, the Diagram Described the proposed methodology specifically, for purpose of this research, the data was collected from online site from GitHub. The collected data were having a massive Malware attack records during network activities for last previous years. The dataset stored as TXT file with more 7 thousands malware records. Then, the dataset analyzed with 80% for training and 20% testing to identify from some parameters based on sequence matrix to classify Malware attacks. The values of the identified parameters were structured in a matrix array and it used for as input for neural network model. Long short term memory model will be used as algorithm. Then the developed model will be evaluated based on the performance analysis, including a confusion matrix and class statistics to learn more about the system's efficiency such as confusion matrix, accuracy, Precision, Recall or Sensitivity, Specificity and F score1.

3.2 Dataset:

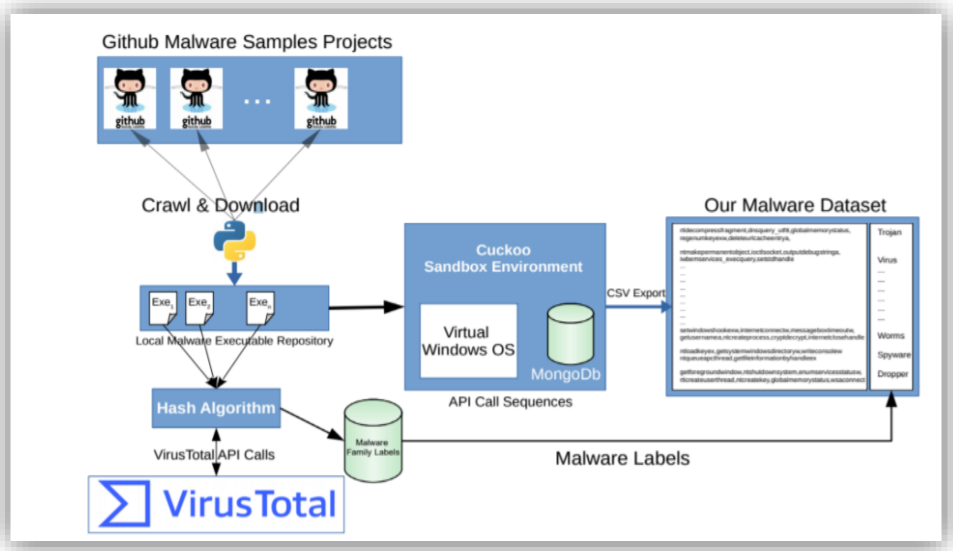
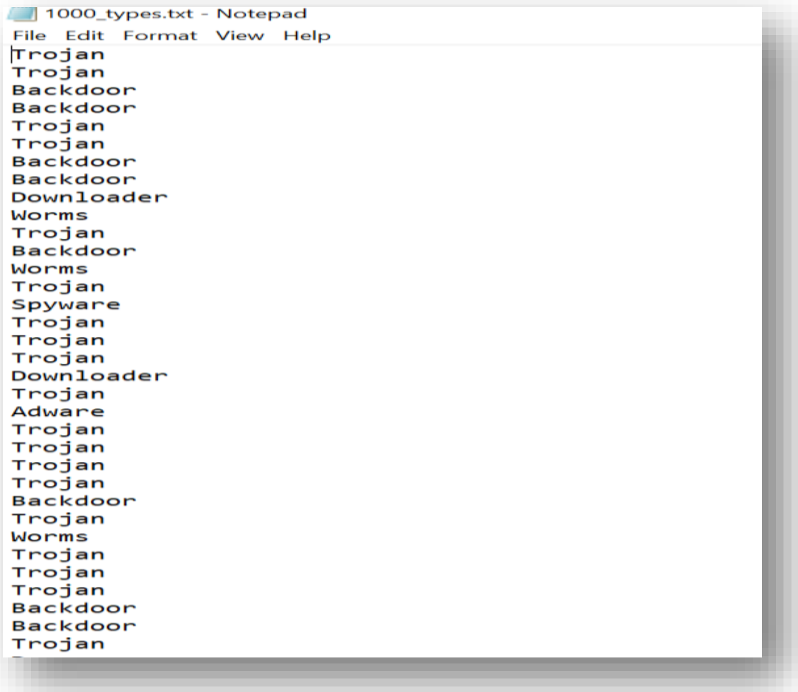


Figure 3.2: Generation of Malware dataset API based Calls by the cyber security researcher called Dr.Ferhat Ozgur Catak.(Catak and Yazı, 2019).

There are more online different datasets for cyber-attacks. The chosen dataset was collected about malware dataset from GitHub site which called Malware API Calls dataset (Application Programming Interface) made by one of cyber security researcher called Dr.Ferhat Ozgur Catak . (Catak *et al.*, 2020) developed more than 7 thousand malwares from different malware types which are malicious software and codes that's cause the damage for the system. In addition, as seen from the figure 3.2, they also used a variety of technologies to create this dataset. First of all, the Cuckoo sandbox software was installed on Linux system in order to run the malicious codes and analyze it's behavior work as in a real environment. On the other hand, windows operating system PE was installed virtually on the server side and the firewall was turned off and no updates for operating system to prevent any malicious codes. The obtained result of the Malware API calls, Network traffic and memory dump were saved in the Mango Database with format JSON. The collected result were analyzed the behavior of malware API based sequence calls using cuckoo sandbox. Second important thing, online services was used to analyze the malware code of files and get the report analysis using Virus Total service. This Virus Total required hash value for each malware and stored the result in the database. They developed a dataset for the API. They got the virus's MD5 hash values from Github. The VirusTotal API was then used to retrieve the hash values. They gathered the malicious malware families from multiple antivirus vendors' VirusTotal reports. As their obtained result was matching between windows API and different malware types, they analyzed different anti-virus software to detect multiple malicious codes. Thus, if the most of applications agreed with same result that a specific code is malicious. Some different classes of malware such as Virus, Trojan, Adware, Backdoor, Worms, Spyware, Dropper and downloader.



```
Trojan
Trojan
Backdoor
Backdoor
Trojan
Trojan
Backdoor
Backdoor
Downloader
Worms
Trojan
Backdoor
Worms
Trojan
Spyware
Trojan
Trojan
Trojan
Downloader
Trojan
Adware
Trojan
Trojan
Trojan
Trojan
Backdoor
Trojan
Worms
Trojan
Trojan
Trojan
Backdoor
Backdoor
Trojan
```

Figure 3.3: Different Malware classes (Virus, Trojan, Adware, Backdoor, Worms, Spyware, Dropper and downloader), (Catak *et al.*, 2020)

3.3 LSTM Structure:

One method of dealing with malware protection challenges is to identify malicious software. In fact, any anti-virus the used in any operating system Application programing interface to decide whether the program is malicious or not. The best deep learning technique is long short term memory which mostly used with sequential data. Developing LSTM model which is a subset of Recurrent Neural Network and it is used to learn sequence data for Malware API Call dataset. This model will consists of creating sequential model, three LSTM Layers, Hidden and Output layer.

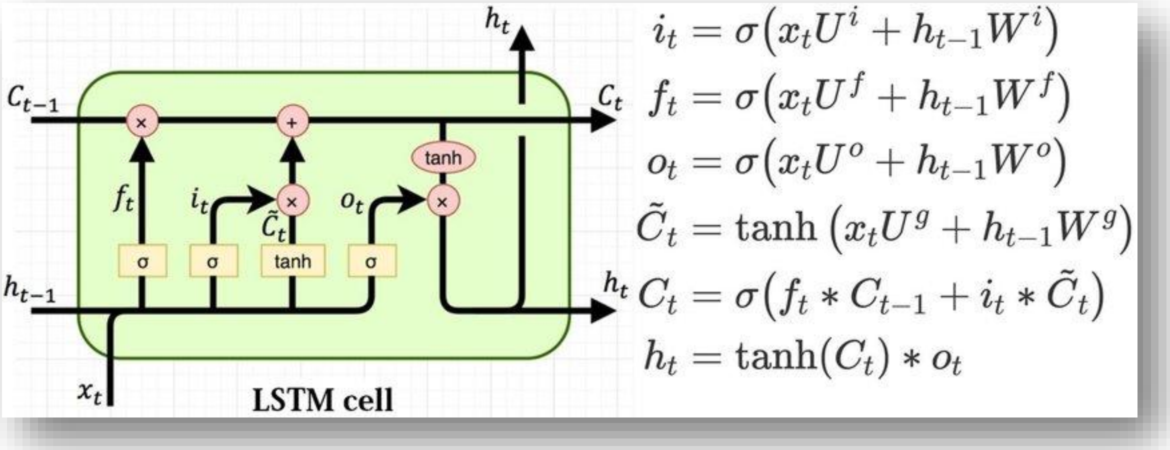

h_t
C_{t-1}
C_t
f_t
i_t
$\tilde{C}_t$
o_t
tanh
σ
σ
tanh
σ
h_{t-1}
h_t
x_t
LSTM cell
$i_t = \sigma\left(x_t U^i + h_{t-1} W^i\right)$
$f_t = \sigma\left(x_t U^f + h_{t-1} W^f\right)$
$o_t = \sigma\left(x_t U^o + h_{t-1} W^o\right)$
$\tilde{C}_t = \tanh\left(x_t U^g + h_{t-1} W^g\right)$
$C_t = \sigma\left(f_t * C_{t-1} + i_t * \tilde{C}_t\right)$
$h_t = \tanh(C_t) * o_t$

Figure 3.4: LSTM Cell Structure (Varsamopoulos, Bertels and Almudever, 2018).

(Varsamopoulos, Bertels and Almudever, 2018) illustrated the design of the LSTM cell as well as the calculation equations that explain the LSTM gates. Each LSTM cell contains three gates: input, forget, and output, as shown from figure 3.4. The W indicates the link between hidden layers. While U represents the weight matrix as a hidden layer input. Internal memory is represented by the letter C. In addition, Xt displays the input or point in the proper sequence. While ht represents the output to hidden layers that are linked. tanh is dense layer and the a sigmoid function is an activation function.

3.4 Tools:

There are some requirements and experiments setup for the collected data set. The Anaconda software was used for the implementation that supports jupyter notebook and allows to run Python language version 3.6. There are primary Python library which used mostly in deep learning. Specifically, tensor flow Library in Python which is the basic library for creating deep learning models. And Keras Library in Python which it is a high-performance architecture based on Tensor Flow 2.0 that can be scalable to massive Graphics Processing Unit clusters or a complete Tensor Processing Unit.

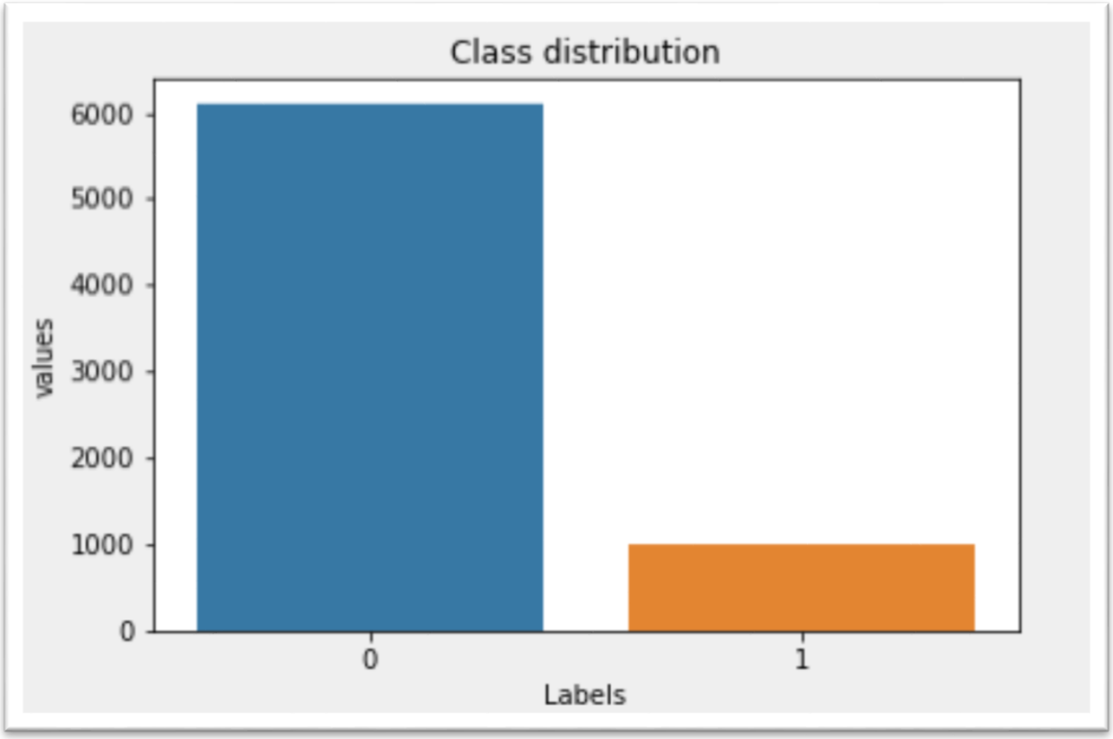


Figure 3.5: Total number of positive and negative Malwares API calls dataset which represent zero and one

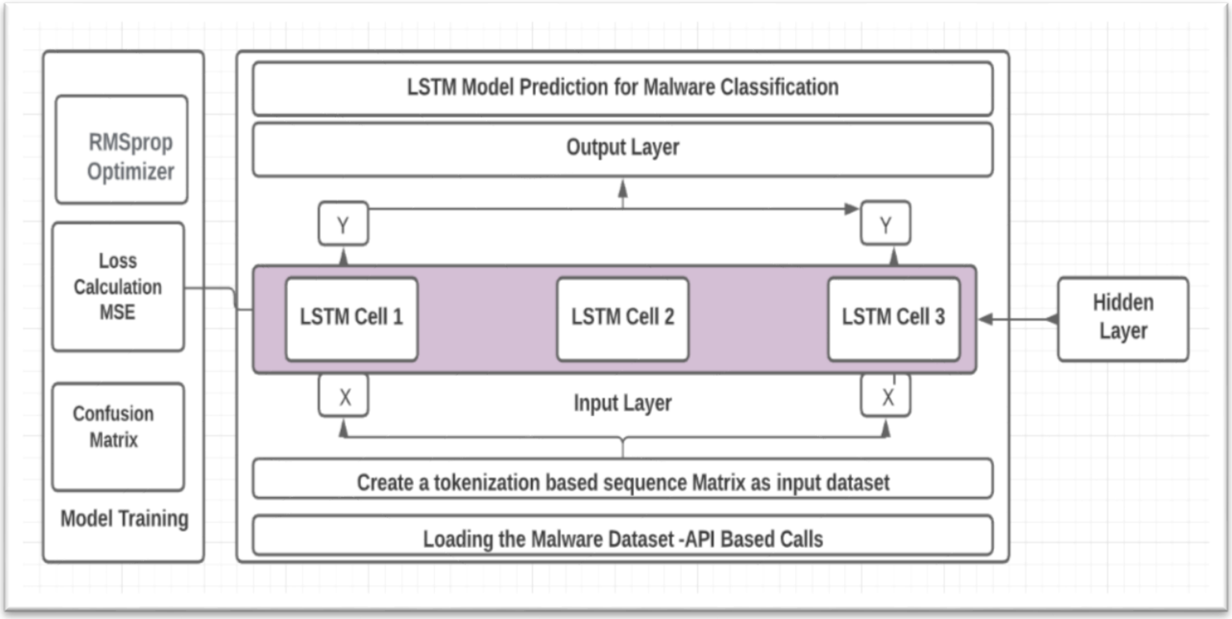

LSTM Model Prediction for Malware Classification
Output Layer
Y
Y
RMSprop Optimizer
Loss Calculation MSE
Confusion Matrix
Model Training
LSTM Cell 1
LSTM Cell 2
LSTM Cell 3
Hidden Layer
X
Input Layer
X
Create a tokenization based sequence Matrix as input dataset
Loading the Malware Dataset -API Based Calls

Figure 3.6: Proposed LSTM Framework for malware Classification.

3.5 Proposed LSTM Framework for malware Classification:

In this research, we propose LSTM model framework for malware classification in order to work as proactive model for organizations to reduce the number of malware attacks as shown from figure 3.6. For Application Programing Interface calls for previous year was collected from GitHub for network activities, a large number of malicious software was built for the dataset. Malware classification are generated using the proposed LSTM model which classify the malware into different classes. First of all, loading different types of python's libraries into Juypter environment using Anaconda Navigator software. Second, loading the dataset and reading the Data in to python environment. The dataset divided into 80% for training and 20% for testing. Then, to extract the feature of sequence, the source codes are tokenized. Third, The Tensor Flow Framework and Keras are used to construct the LSTM model. The LSTM algorithm is implemented using the tenser flow's application programming interface. In fact, for high computation in a complicated environment, tensor flow is the ideal solution. Because, it can be configured with different layer for complex classification and training the dataset. This technique uses the tensor flow and keras architecture to find malware codes that are similar. The malware codes are then used to figure out which malware family they belong to. In addition, the LSTM model use the weighting values as the input layer. The configuration between input layer and output layer called the dense layer or fully connected layer. In the proposed LSTM model, there are one embedding layer, three LSTM layers and three dense layers with (256, 128, 1) neurons, respectively. The data is received by the first layer, from the previous layer or the dense layer which includes input variables. More ever, in order to overcome the over fitting problems, LSTM algorithm is using the dropout layer. The neural networks are calculated using activation function and linear function for malware classification. Soft sign activation function which scale between 1 and -1 while the linear multiplied by 1. At the training LSTM model, loss function mean square error is used for calculating the difference between the predicated parameters and observed parameters. In addition, the neural network get trained using the RMSprop optimizer is best option for LSTM model. The last but not least, the LSTM model evaluated using confusion matrix which is a summary of the malware classification results which calculate correct and incorrect predication. All parameters are calculated to evaluate the performance of the proposed model like confusion matrix, accuracy, Precision, Sensitivity, Specificity and F score1.

3.6 Summary:

This chapter described the methodology of the proposed LSTM model. It showed the collection of Malware dataset API based calls from Github online site. And it showed the generation process of the Dataset which was created by (Catak and Yazı, 2019). In addition, it presented all the software tools need for creating LSTM model. Lastly, it descrebed the development of the theoritacl framework of the propsed LSTM model.

CHAPTER4 IMPLEMENTATION:

This chapter will shows the important of Python Libraries and python codes that used to create the proposed LSTM model. More specifically, the Proposed LSTM model was created using Anaconda Navigator software with juypter Notebook using Keras and Tensor flow libraries. There are mainly 5 stages of the development the LSTM model using different Python Libraries. First of all, Importing Python Libraries into the Juypter Notebook Interface using pip install followed by the python library name such as pip install tensor flow. Sometimes, already the python library is defined and installed and some time there have been several errors because of the proxy or internet issues which needs to be debugged and fixed. Second stage is loading the malware dataset using Panda library. Third stage is splitting the Dataset into the Training with 80% and testing 20% using sklearn library. Fourth stage is developing LSTM Model and compiling it with keras and tensor flow Python Libraries using sequential model, Dense, LSTM layer and dropout Layer. Fourth stage, fitting the proposed LSTM to the Training and testing Dataset using keras library. Lastly, evaluating stage of the proposed LSTM model using Sklearn and mlxtend libraries with summarization of history, epoch and number of patch size.

Table 4.1: Python Libraries.

Python Libraries	**The Purpose of Used**	**The Python Code**
Pandas	To read the data from TXT file.	import pandas as pd
Matplotlib	To plot the figures and lines	import matplotlib.pyplot as plt
Seaborn	To represent graphics based on matplotlib	import seaborn as sns
Sklearn	To encode the label to multiple classes.	From sklearn.preprocessing import LabelEncoder.
	To split the dataset in to Training and Test.	from sklearn.model_selection import train_test_split
	To calculate the accuracy and provide the summary of the implementation of the malware classification.	from sklearn.metrics import confusion_matrix

Keras	To convert text to sequence	From keras.preprocessing.text import Tokenizer.
	Dropout to set the input value as zero and to reduce the neural network over fitting	from keras.layers import LSTM, Dense, Dropout, Embedding
	To make sure each sequence have equal size length.	from keras.preprocessing import sequence
	To allows the call back function.	from keras.utils import np_utils
	To create LSTM with multiple input and output layers.	from keras.models import Sequential
	To remove the entire feature	from keras.layers import SpatialDropout1D
mlxtend	To plot the summary of performance with confusion matrix.	from mlxtend.plotting import plot_confusion_matrix

Table 4.2: Process of creating the LSTM Model:

Requirements	**The Purpose**
Sequential	To create neural network
LSTM	To add LSTM Layer
Dropout	To add dropout layer in order to avoid over fitting issues.
Dense or fully connected	To configure densely layer which connect between the input layer and output layer. All neurons receive the information from the previous layer.

4.1 Summary:

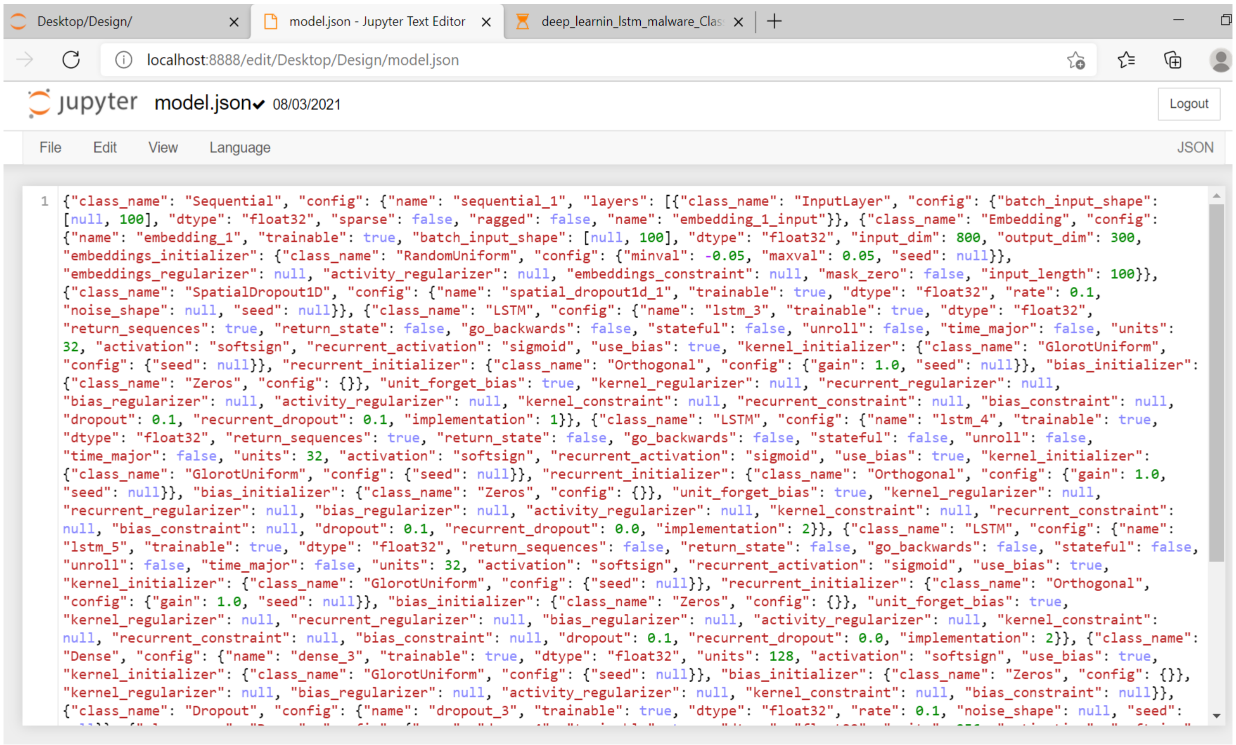
In this chapter, it described a slight view of the importance of python libraries which used for creating the proposed LSTM model. In addition, summary development of the proposed model was given in order to classify malware attacks using short term memory algorithm. Also, it described the requirement layers for creating of the LSTM model.

CHAPTER5 : ANALYSIS AND FINDINGS

In this chapter, the proposed LSTM model first will be trained 80% and tested 20% from Malware dataset in order to analyze the obtained result. Second, the Max validation accuracy and min Validation Accuracy will be observed and analyzed. Third, the proposed LSTM model will be evaluated automatically with some confusion matrix and parameters. On other hands, these matrix will be calculated manually using math formula in order to compare the collected results. Lastly, discussion of the obtained result of the proposed LSTM model with other researchers work.

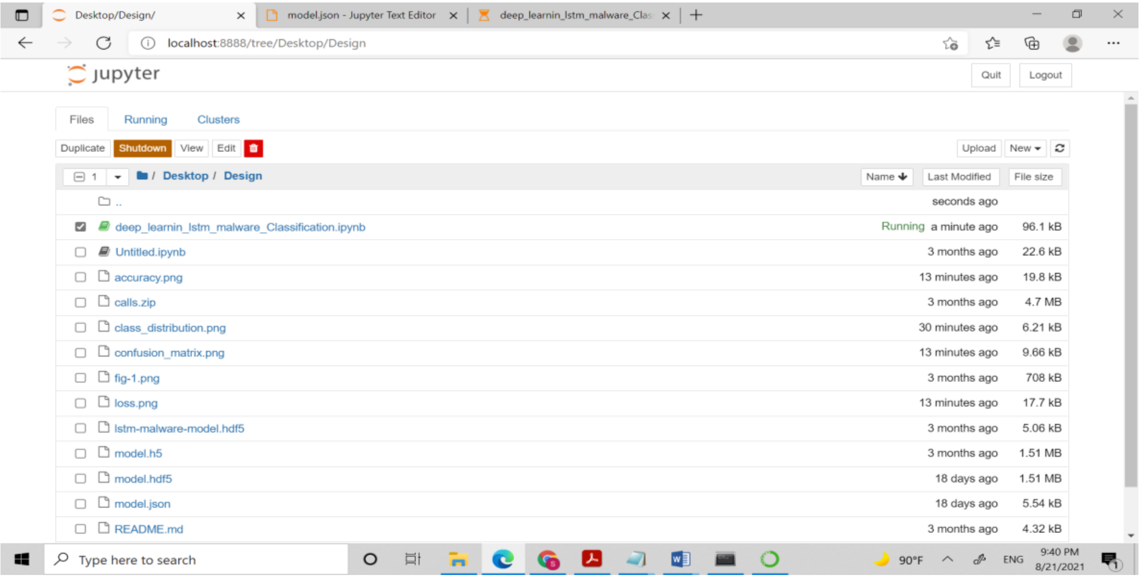
5.1 Training and Testing the Proposed Model:

The proposed LSTM model created using python language with tenser flow and keras libraries. There are some steps of creating the LSTM model. First of all, sequential neural network is created with soft sign activation function. Second, adding the Layers of LSTM, dense, dropout layers and hidden node till all three LSTM layers are created. Third, complied all sequential objects of the proposed LSTM model. Fourth, the fit the created LSTM model to training the neural network. Fifth thing, after training the proposed model, call back function is used to save all model configuration as “model .JSON” and sometimes convert to HDF5 file format if the data was very huge. Lastly, get a brief summary for the proposed LSTM model using model. Summary () method as shown in the figure. The output predication of LSTM model with all trainable parameters are calculated such as total time, steps, accuracy, validation accuracy, loss, validation loss. This proposed LSTM model can be trained and tested in order to evaluate the performance of the model based on it’s accuracy. The proposed LSTM model developed using python function and different parameters such as sequential, embedding, dropout….etc. This model has be trained with RMSprop optimizer to train the neural network in order to increase the learning rate. More importantly, the proposed LSTM should be trained with mean squared error loss to determine how well it fits data points by calculating the difference between predicted values and actual values. The developed LSTM model has to be trained with 1000 batch size and 15 epochs to call the selection model in order to obtained the reliable information within specific time.


Desktop/Design/
model.json - Jupyter Text Editor
deep_learnin_lstm_malware_Clas
localhost:8888/edit/Desktop/Design/model.json
jupyter
model.json
08/03/2021
Logout
File
Edit
View
Language
JSON

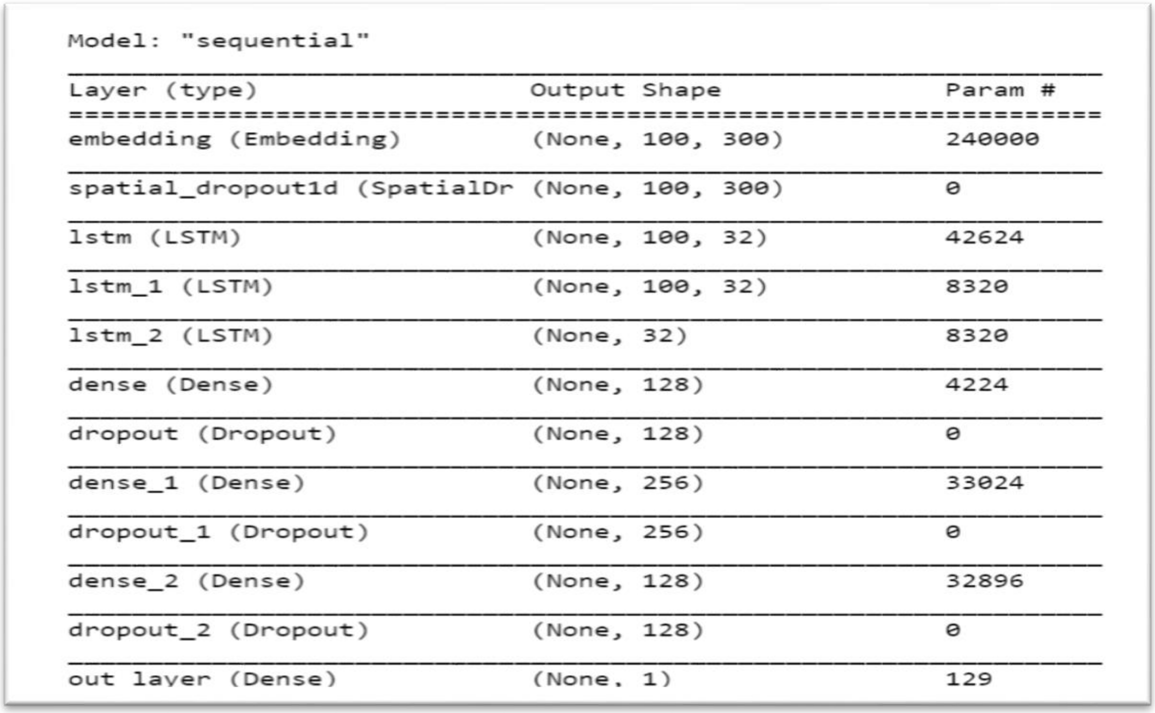
```
{"class_name": "Sequential", "config": {"name": "sequential_1", "layers": [{"class_name": "InputLayer", "config": {"batch_input_shape":
[null, 100], "dtype": "float32", "sparse": false, "ragged": false, "name": "embedding_1_input"}}, {"class_name": "Embedding", "config":
{"name": "embedding_1", "trainable": true, "batch_input_shape": [null, 100], "dtype": "float32", "input_dim": 800, "output_dim": 300,
"embeddings_initializer": {"class_name": "RandomUniform", "config": {"minval": -0.05, "maxval": 0.05, "seed": null}},
"embeddings_regularizer": null, "activity_regularizer": null, "embeddings_constraint": null, "mask_zero": false, "input_length": 100}},
{"class_name": "SpatialDropout1D", "config": {"name": "spatial_dropout1d_1", "trainable": true, "dtype": "float32", "rate": 0.1,
"noise_shape": null, "seed": null}}, {"class_name": "LSTM", "config": {"name": "lstm_3", "trainable": true, "dtype": "float32",
"return_sequences": true, "return_state": false, "go_backwards": false, "stateful": false, "unroll": false, "time_major": false, "units":
32, "activation": "softsign", "recurrent_activation": "sigmoid", "use_bias": true, "kernel_initializer": {"class_name": "GlorotUniform",
"config": {"seed": null}}, "recurrent_initializer": {"class_name": "Orthogonal", "config": {"gain": 1.0, "seed": null}}, "bias_initializer":
{"class_name": "Zeros", "config": {}}, "unit_forget_bias": true, "kernel_regularizer": null, "recurrent_regularizer": null,
"bias_regularizer": null, "activity_regularizer": null, "kernel_constraint": null, "recurrent_constraint": null, "bias_constraint": null,
"dropout": 0.1, "recurrent_dropout": 0.1, "implementation": 1}}, {"class_name": "LSTM", "config": {"name": "lstm_4", "trainable": true,
"dtype": "float32", "return_sequences": true, "return_state": false, "go_backwards": false, "stateful": false, "unroll": false,
"time_major": false, "units": 32, "activation": "softsign", "recurrent_activation": "sigmoid", "use_bias": true, "kernel_initializer":
{"class_name": "GlorotUniform", "config": {"seed": null}}, "recurrent_initializer": {"class_name": "Orthogonal", "config": {"gain": 1.0,
"seed": null}}, "bias_initializer": {"class_name": "Zeros", "config": {}}, "unit_forget_bias": true, "kernel_regularizer": null,
"recurrent_regularizer": null, "bias_regularizer": null, "activity_regularizer": null, "kernel_constraint": null, "recurrent_constraint":
null, "bias_constraint": null, "dropout": 0.1, "recurrent_dropout": 0.0, "implementation": 2}}, {"class_name": "LSTM", "config": {"name":
"lstm_5", "trainable": true, "dtype": "float32", "return_sequences": false, "return_state": false, "go_backwards": false, "stateful": false,
"unroll": false, "time_major": false, "units": 32, "activation": "softsign", "recurrent_activation": "sigmoid", "use_bias": true,
"kernel_initializer": {"class_name": "GlorotUniform", "config": {"seed": null}}, "recurrent_initializer": {"class_name": "Orthogonal",
"config": {"gain": 1.0, "seed": null}}, "bias_initializer": {"class_name": "Zeros", "config": {}}, "unit_forget_bias": true,
"kernel_regularizer": null, "recurrent_regularizer": null, "bias_regularizer": null, "activity_regularizer": null, "kernel_constraint":
null, "recurrent_constraint": null, "bias_constraint": null, "dropout": 0.1, "recurrent_dropout": 0.0, "implementation": 2}}, {"class_name":
"Dense", "config": {"name": "dense_3", "trainable": true, "dtype": "float32", "units": 128, "activation": "softsign", "use_bias": true,
"kernel_initializer": {"class_name": "GlorotUniform", "config": {"seed": null}}, "bias_initializer": {"class_name": "Zeros", "config": {}},
"kernel_regularizer": null, "bias_regularizer": null, "activity_regularizer": null, "kernel_constraint": null, "bias_constraint": null}},
{"class_name": "Dropout", "config": {"name": "dropout_3", "trainable": true, "dtype": "float32", "rate": 0.1, "noise_shape": null, "seed":
```

Figure 5.1: LSTM model. json file which stand for JavaScript Object Notation that can be easily readable by humans. It is metadata in tensor flow and it is converted to HDF5 file which is stand for Hierarchical Data Format version 5 which work as file directory for massive data.



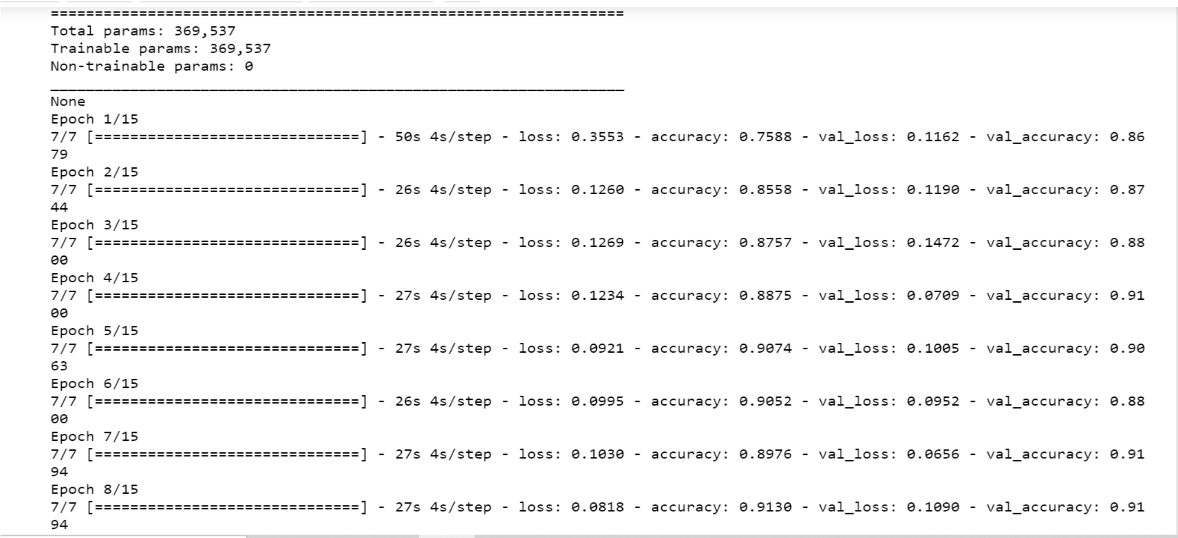
	Name		Last Modified	File size
	/ Desktop / Design			
	..		seconds ago	
☑	deep_learnin_lstm_malware_Classification.ipynb	Running	a minute ago	96.1 kB
☐	Untitled.ipynb		3 months ago	22.6 kB
☐	accuracy.png		13 minutes ago	19.8 kB
☐	calls.zip		3 months ago	4.7 MB
☐	class_distribution.png		30 minutes ago	6.21 kB
☐	confusion_matrix.png		13 minutes ago	9.66 kB
☐	fig-1.png		3 months ago	708 kB
☐	loss.png		13 minutes ago	17.7 kB
☐	lstm-malware-model.hdf5		3 months ago	5.06 kB
☐	model.h5		3 months ago	1.51 MB
☐	model.hdf5		18 days ago	1.51 MB
☐	model.json		18 days ago	5.54 kB
☐	README.md		3 months ago	4.32 kB

Figure 5.2: Python files which saved all the proposed LSTM model



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 100, 300)          240000
_________________________________________________________________
spatial_dropout1d (SpatialDr (None, 100, 300)          0
_________________________________________________________________
lstm (LSTM)                  (None, 100, 32)           42624
_________________________________________________________________
lstm_1 (LSTM)                (None, 100, 32)           8320
_________________________________________________________________
lstm_2 (LSTM)                (None, 32)                8320
_________________________________________________________________
dense (Dense)                (None, 128)               4224
_________________________________________________________________
dropout (Dropout)            (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 256)               33024
_________________________________________________________________
dropout_1 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_2 (Dense)              (None, 128)               32896
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
out_layer (Dense)            (None, 1)                 129
```

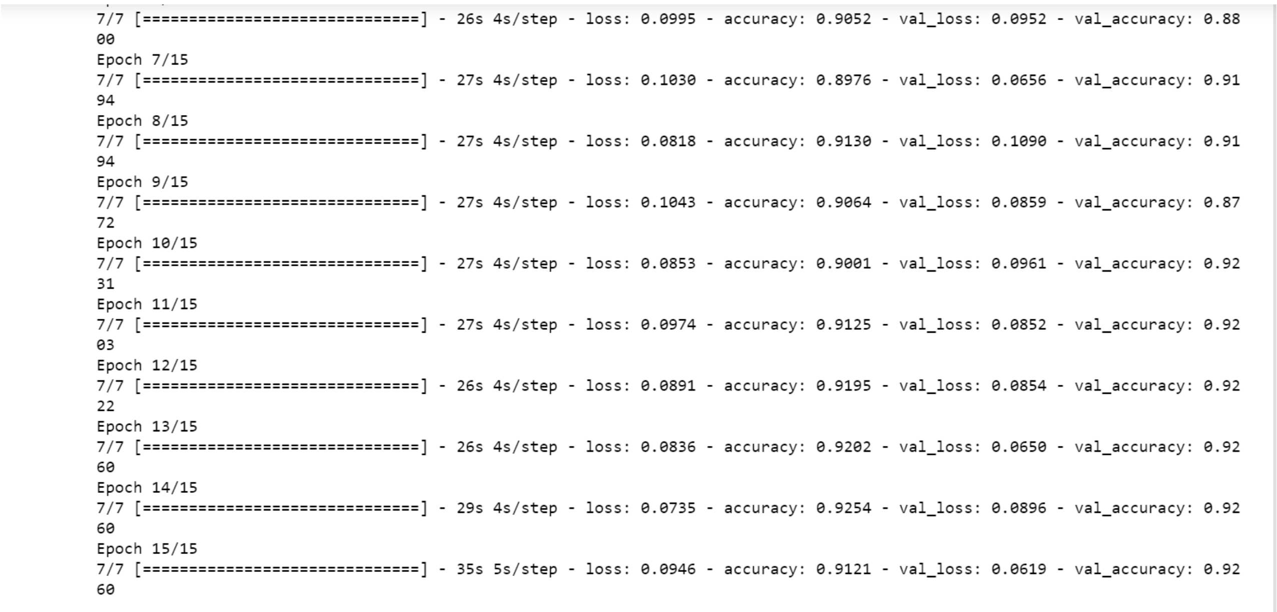
Figure 5.3: Proposed Sequential LSTM Model summary. It shows multiple layers (embedding, LSTM, Dropout and dense) with each parameters.



```
=================================================================
Total params: 369,537
Trainable params: 369,537
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/15
7/7 [==============================] - 50s 4s/step - loss: 0.3553 - accuracy: 0.7588 - val_loss: 0.1162 - val_accuracy: 0.86
79
Epoch 2/15
7/7 [==============================] - 26s 4s/step - loss: 0.1260 - accuracy: 0.8558 - val_loss: 0.1190 - val_accuracy: 0.87
44
Epoch 3/15
7/7 [==============================] - 26s 4s/step - loss: 0.1269 - accuracy: 0.8757 - val_loss: 0.1472 - val_accuracy: 0.88
00
Epoch 4/15
7/7 [==============================] - 27s 4s/step - loss: 0.1234 - accuracy: 0.8875 - val_loss: 0.0709 - val_accuracy: 0.91
00
Epoch 5/15
7/7 [==============================] - 27s 4s/step - loss: 0.0921 - accuracy: 0.9074 - val_loss: 0.1005 - val_accuracy: 0.90
63
Epoch 6/15
7/7 [==============================] - 26s 4s/step - loss: 0.0995 - accuracy: 0.9052 - val_loss: 0.0952 - val_accuracy: 0.88
00
Epoch 7/15
7/7 [==============================] - 27s 4s/step - loss: 0.1030 - accuracy: 0.8976 - val_loss: 0.0656 - val_accuracy: 0.91
94
Epoch 8/15
7/7 [==============================] - 27s 4s/step - loss: 0.0818 - accuracy: 0.9130 - val_loss: 0.1090 - val_accuracy: 0.91
94
```

Figure 5.4: Evaluation the training LSTM model with 15 Epochs, patch size 1000 with 369,537 trainable parameters.

As figure 5.4 shows calculation of some parameters such as accuracy, validation accuracy, loss, validation loss. Initially, it observed that the first epoch with 50 second and 4 steps. The loss percentage was 35% and accuracy start with 75%. While the validation loss initiate with 11% and validation accuracy start with 86%.



```
7/7 [==============================] - 26s 4s/step - loss: 0.0995 - accuracy: 0.9052 - val_loss: 0.0952 - val_accuracy: 0.88
00
Epoch 7/15
7/7 [==============================] - 27s 4s/step - loss: 0.1030 - accuracy: 0.8976 - val_loss: 0.0656 - val_accuracy: 0.91
94
Epoch 8/15
7/7 [==============================] - 27s 4s/step - loss: 0.0818 - accuracy: 0.9130 - val_loss: 0.1090 - val_accuracy: 0.91
94
Epoch 9/15
7/7 [==============================] - 27s 4s/step - loss: 0.1043 - accuracy: 0.9064 - val_loss: 0.0859 - val_accuracy: 0.87
72
Epoch 10/15
7/7 [==============================] - 27s 4s/step - loss: 0.0853 - accuracy: 0.9001 - val_loss: 0.0961 - val_accuracy: 0.92
31
Epoch 11/15
7/7 [==============================] - 27s 4s/step - loss: 0.0974 - accuracy: 0.9125 - val_loss: 0.0852 - val_accuracy: 0.92
03
Epoch 12/15
7/7 [==============================] - 26s 4s/step - loss: 0.0891 - accuracy: 0.9195 - val_loss: 0.0854 - val_accuracy: 0.92
22
Epoch 13/15
7/7 [==============================] - 26s 4s/step - loss: 0.0836 - accuracy: 0.9202 - val_loss: 0.0650 - val_accuracy: 0.92
60
Epoch 14/15
7/7 [==============================] - 29s 4s/step - loss: 0.0735 - accuracy: 0.9254 - val_loss: 0.0896 - val_accuracy: 0.92
60
Epoch 15/15
7/7 [==============================] - 35s 5s/step - loss: 0.0946 - accuracy: 0.9121 - val_loss: 0.0619 - val_accuracy: 0.92
60
```

Figure 5.5: Evaluating the training LSTM model with 15 Epochs

From Figure 5.5, it has been noticeable that when the number of epoch increased, as consequence the accuracy is increasing till 92.5%. However, when the number of epochs is increasing, the loss decrease until it reach 7%.

Table 5.1: Max Validation-Accuracy and Min Validation Loss

Epoch	Max Val-Accuracy	Min Val-Loss
1	86%	11%
2	87%	12%
3	88%	14%
4	91%	7%
5	90%	10%
6	88%	9%
7	91%	6%
8	91%	10%
9	87%	8%
10	92%	9%
11	92%	8%
12	92%	8%
13	92%	6%
14	92%	8%
15	92%	6%

The obtained result of Max Validation-Accuracy and Min Validation Loss of the training proposed model. As seen from Table 5.1 during the training, good results are achieved with the validation-accuracy which increase the range from 86% to 92%. In comparison with min validation loss, good result are achieved during the training model with range 11% to 6%. It has been observe that when the number of epoch increased, as consequence validation accuracy increase until 92%. However, when the number of epochs is increasing, the validation loss reducing very slowly until 6%.

5.2 Evaluation the performance of proposed LSTM model:

The malware dataset API Calls collected from Github with 7107 thousands samples malware to analyze and develop LSTM model in order to classify malware attacks. The evaluation of the proposed LSTM used to evaluate the performance of the classification across different malware 8 classes such as Virus, Trojan, Backdoor, Worms, Adware, Dropper, Spyware and downloader. More ever, a comparison of the LSTM performance was conducted with other similar previous models of other researchers. In fact, the obtained result of proposed LSTM model already presented satisfactory result on prediction abilities of neural network. (Zeng, 2020) has stated that the Confusion Matrix is a useful technique for measuring the efficiency of scoring algorithm. For this reason, first the datasets for this proposed model divided in to two part 80% training and 20% testing using tensor flow platform. Second, the dataset processed into 278 unique tokens including max words with 800 and max length 100. Third, sequential model was created with three LSTM layers. As well as three hidden layers were created in order to improve the accuracy of the proposed model. More ever, the dropout layer is configured to solve over fitting issues. Fourth, there are some evaluation parameters used to evaluate the training model such as accuracy, validation accuracy, loss, validation loss. The total parameters and number of trainable parameters are calculated 369,537 parameters as shown in the figure 4. Due to different classes of malware in the dataset, the proposed LSTM model was evaluated its performance by measuring some parameters over all malware classes. More specific, different matrices were calculated such as Confusion Matrix, accuracy, precision, Sensitivity, specificity and F-score1.

- Confusion Matrix: As previously stated, the confusion matrix was chosen based on classification abilities as well as a well-known matrix for evaluating classification accuracy. The Confusion Matrix summarizes the classification results such as total number of correct and incorrect prediction.

Table 5.2: Confusion matrix

	Predicted 0	Predicted 1
Actual 0	True Negative	False Positive
Actual 1	False Negative	True Positive

- Accuracy: Accuracy is one of the most useful matrix which evaluate the performance of classification of proposed model.
- Accuracy = $\frac{Total\ of\ correct\ predications}{Total\ Number\ of\ predications}$
- Precision: it is the degree point of the scoring result which shows how it is closely to the measurement. In fact, it calculate out of all positives, how many were correctly identified as positive.
- Sensitivity or Recall: it calculate total number of actual positive able to be as correct prediction.
- Specificity: it calculate total number of true negative divided by True negative and false positive.
- F-score1: it is calculated by taking the mean value of Precision and Recall.

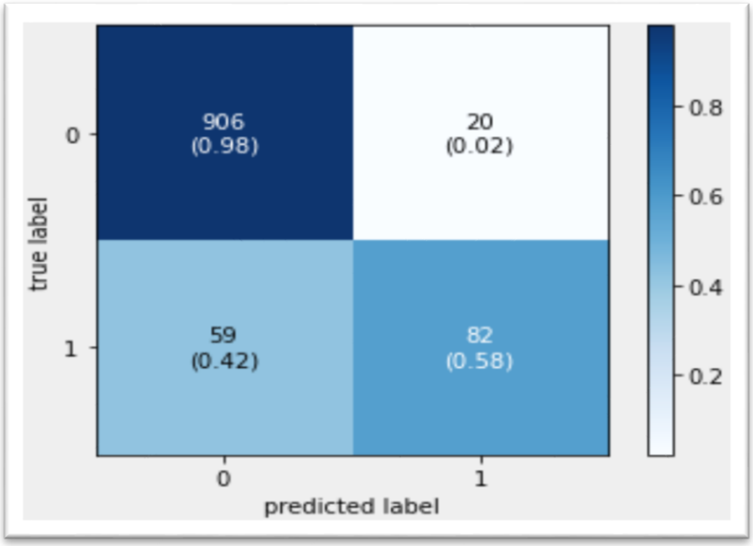
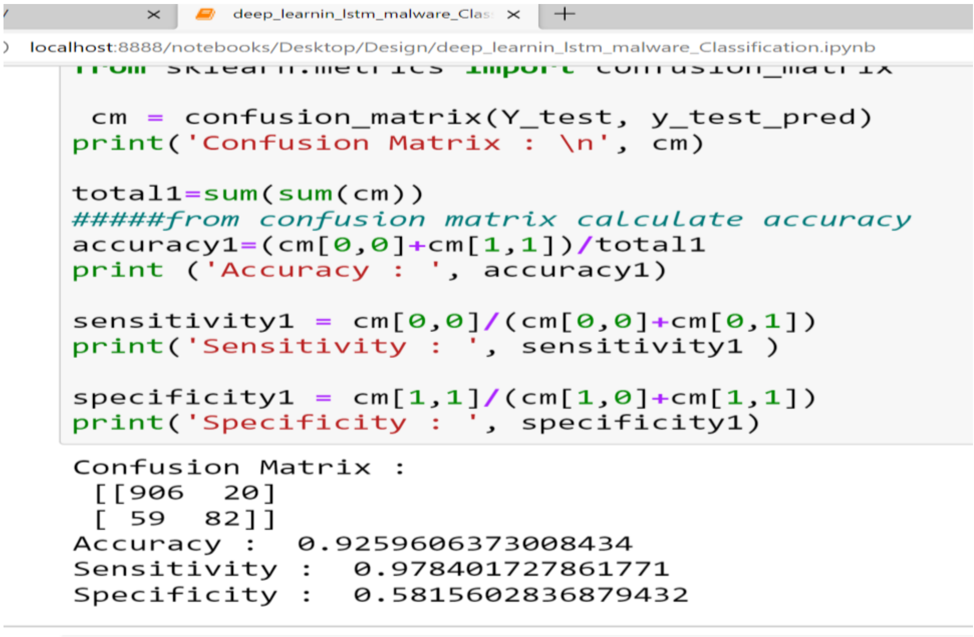


Figure 5.6: The obtained result of Confusion Matrix of Proposed LSTM model.


deep_learnin_lstm_malware_Clas
localhost:8888/notebooks/Desktop/Design/deep_learnin_lstm_malware_Classification.ipynb

```
from sklearn.metrics import confusion_matrix

 cm = confusion_matrix(Y_test, y_test_pred)
print('Confusion Matrix : \n', cm)

total1=sum(sum(cm))
#####from confusion matrix calculate accuracy
accuracy1=(cm[0,0]+cm[1,1])/total1
print ('Accuracy : ', accuracy1)

sensitivity1 = cm[0,0]/(cm[0,0]+cm[0,1])
print('Sensitivity : ', sensitivity1 )

specificity1 = cm[1,1]/(cm[1,0]+cm[1,1])
print('Specificity : ', specificity1)
```

```
Confusion Matrix :
 [[906  20]
 [ 59  82]]
Accuracy :  0.9259606373008434
Sensitivity :  0.978401727861771
Specificity :  0.5815602836879432
```

Figure 5.7: Evaluation the proposed LSTM model with Python code for calculation Confusion matrix, Accuracy, Sensitivity and Specificity. As it has been noticeable the achieved accuracy almost 93%.

Table 5.3: The obtained result of the proposed LSTM model

Accuracy:	0.9259606373008434
Confusion Matrix :	[[906 20] [59 82]]
F-score Measure:	0.760
Precision:	0.750
Specificity:	0.978401727861771
Sensitivity:	0.5815602836879432

Table 5.4: Metrics Formula Calculated manually.

Accuracy:	$Accuarcy = \frac{TP+TN}{TP+FP+TN+FN} = \frac{82+906}{906+59+20+82} = 92.596\%$
Precision	$\text{Precision} = \frac{TP}{TP+FP} = \frac{82}{82+20} = 80\%$
Sensitivity	$\text{Sensitivity or Recall} = \frac{TP}{TP+FN} = \frac{82}{82+59} = 54\%.$
Specificity	$\text{Specificity} = \frac{TN}{TN+FP} = \frac{906}{906+20} = 98\%$
F-score1	$\text{F} - \text{score1} = \frac{2}{\frac{1}{Recall}+\frac{1}{Precision}} = \frac{2}{\frac{1}{45}+\frac{1}{80}} = 58\%$

As shown from table 5.4, The Accuracy which calculated manually it is truly 93%. It will show a little changes in Training LSTM model 92.59% because of late in the debugging and learning stage.

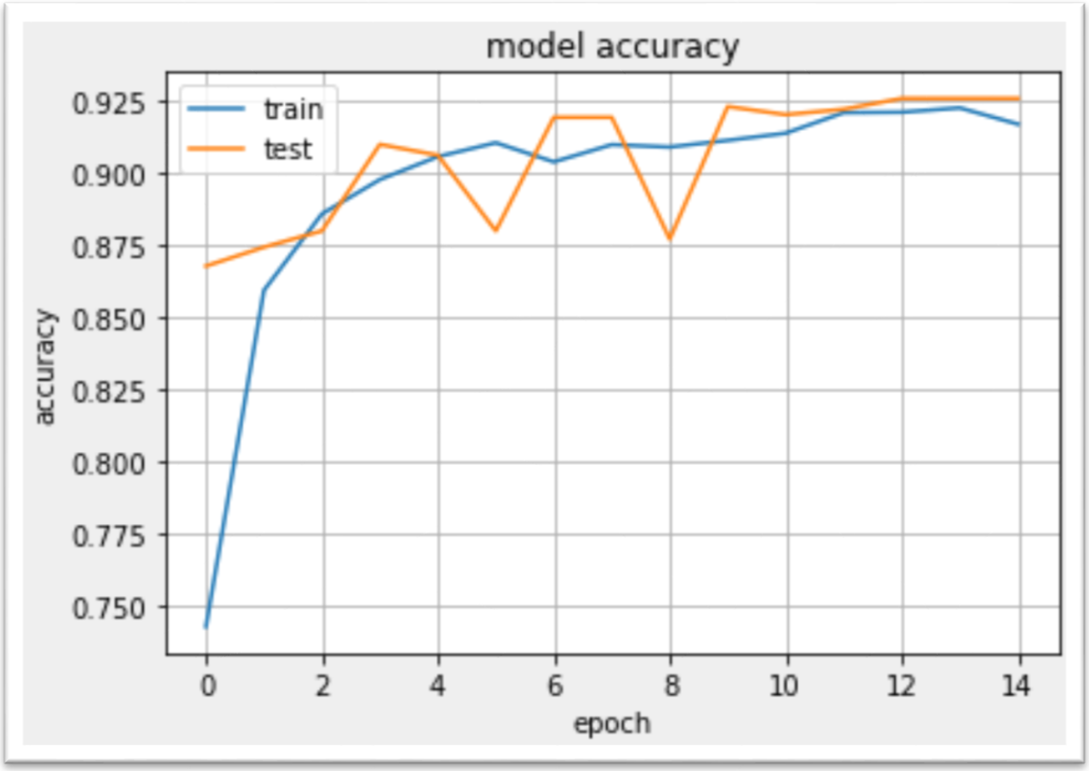


Figure 5.8: Model Accuracy curve shows the training and testing samples.

The proposed LSTM model has been trained and tested with 15 Epochs. As seen from figure 5.8, that the training proposed LSTM model started from 73% and testing model started 86%. While it has been observed that both training and testing samples it reach up to 93% percent of accuracy.

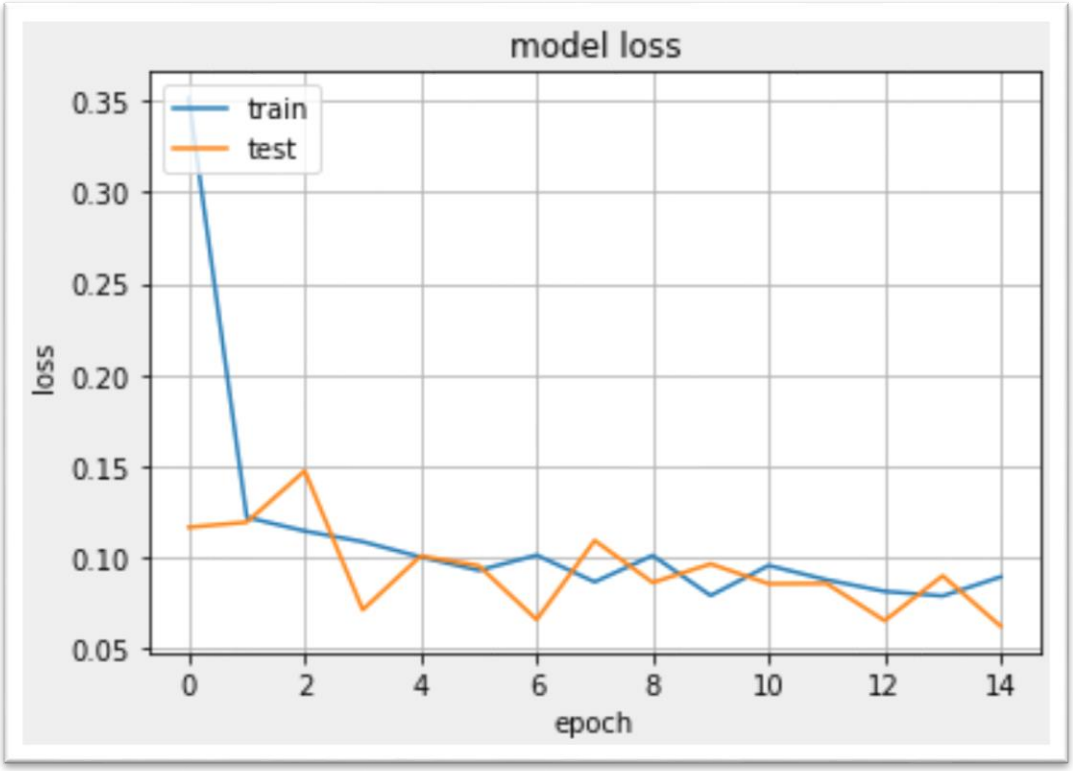
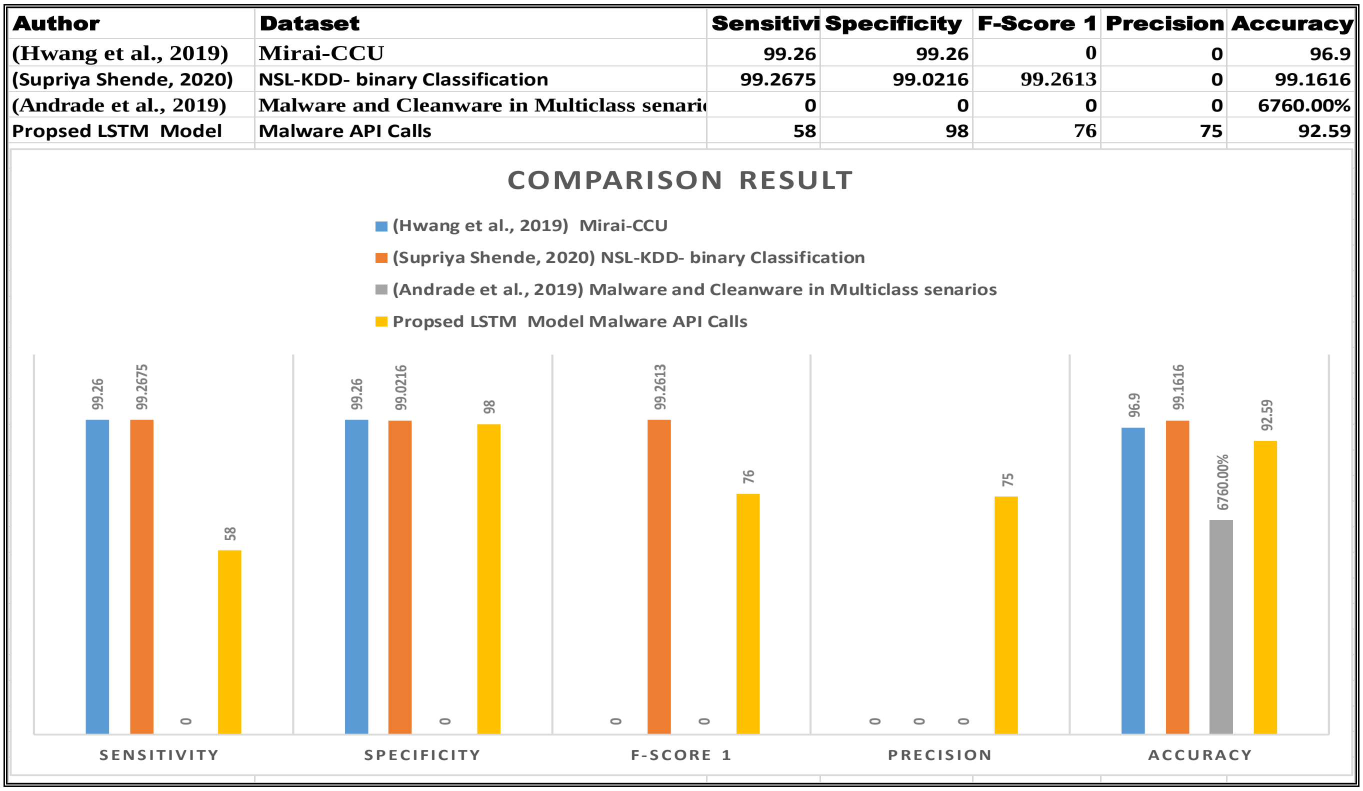


Figure 5.9: Model Loss curve shows the loss of training and testing samples.

The developed model has been trained and tested with 15 epochs. However, it has been noticeable from figure 5.9, that during training the proposed model the percentage of loss was starting to decrease from 35% from the first epoch until it reach 15 epochs with 7% loss percentage. Similarly, during testing the proposed LSTM model, the loss percentage started from 12% and decreased until it reached 7%.

5.3 Discussion of the obtained Result:

This section takes a look at the obtained result and gives a clear comparison with other pervious researchers work on classification of Malware using long short term memory model. The evaluation of the proposed LSTM model achieved overall 92.59% on malware API calls which successfully classify malware attacks. The close range of validation- accuracy from 86% to 92% of LSTM training model. Thus, this accuracy indicates that generally good performance, comparable with other researchers work as shown from figure below 5.10. Some researchers get more than our proposed model overall 96% and 99% accuracy in their LSTM models. (Hwang et al., 2019) and (Supriya Shende, 2020) according to their developed LSTM model, they used different datasets with multiple methods of classification of malware. As consequence, the accuracy score was very higher. On the other hand, (Andrade et al., 2019) proposed LSTM model for malware classification with overall accuracy 67%. According to their LSTM model, they have limitation in their dataset and less number of confusion matrices evaluated model. Thus, their obtained result for accuracy was less.



Author	Dataset	Sensitivi	Specificity	F-Score 1	Precision	Accuracy
(Hwang et al., 2019)	Mirai-CCU	99.26	99.26	0	0	96.9
(Supriya Shende, 2020)	NSL-KDD- binary Classification	99.2675	99.0216	99.2613	0	99.1616
(Andrade et al., 2019)	Malware and Cleanware in Multiclass senari	0	0	0	0	6760.00%
Propsed LSTM Model	Malware API Calls	58	98	76	75	92.59

Figure 5.10: Comparison Result Graph of the Proposed LSTM model with other researchers work.

In Summary, to sum up these points of the Proposed LSTM model:

- In this thesis, the Malware API calls Dataset of the proposed LSTM model was taken from the online site GitHub. Thus, to improve the accuracy score, we have to take real dataset in order to classify malware attacks more correctly.

- Most of pervious researchers work used different datasets and covered more malware family types using different method types. As consequence, they achieved higher accuracy which range from 96% to 99%. Thus, comparable with our proposed model, only one dataset was used with call sequence method with Malware API calls using tenser flow and keras libraries and 8 different classes in the dataset (Virus, Trojan, Adware, Backdoor, Worms, Spyware, Dropper and downloader).

As consequence, considering these two factors, a comparison result of Malware classification with other pervious researchers work. The obtained result of proposed LSTM model was achieved overall 93% as good result.

5.4 Summary:

This chapter described the development, Training and Evaluating the proposed LSTM model which classify the malware cyber-attacks. It presented a clear comparison of the obtained result of proposed LSTM model with other previous work.

CHAPTER6 : CONCLUSIONS AND RECOMMENDATIONS:

To summarize this project, there is serious need to classify cyber-attacks for the purpose of assuring the efficiency of cyber security policy within organization and it is highly needed in each organization reflecting the strong mechanism in order to protect valuable assets from any cyber-attacks. This thesis presented classification of malware-attacks using long term short memory Network model. And the malware API calls dataset collected online from Github for malware. In addition, several previous work was reviewed in Malware classification using machine learning and deep learning Techniques. The proposed LSTM model developed with Long Short Term memory algorithm using Keras and Tensor flow environment. The proposed LSTM model was trained 80% and tested 20% with Malware dataset API based calls. More ever, the proposed model has been evaluated and measured the performance using Confusion matrix and statics class parameters such as Accuracy, Precision, Sensitivity, Specificity and F score1. The obtained result was achieved good accuracy sore with overall 92.59% comparing with other previous researchers work. Finally, this proposed model trained to be able to act as proactive model and it will present a solution of cyber-attacks in real world and life scenarios.

6.1 Recommendation:

In order to improve the quality of the proposed model of malware classification there are some recommending points:

- Increasing number of malware types in the dataset, will help to have different malware family types which improve the performance of the proposed LSTM model.
- Increasing the number of epochs in evaluation the proposed LSTM model. As a consequence, faster response times and higher accuracy score could be predicted.
- Providing multiple dataset types with different years. In order to compare more results score with different dataset.
- Configuring LSTM model with different classification methods and adding more layers in order to increase the complexity and improve the model performance.

References:

1. Oman Observer. 2021. Oman thwarts 14 billion cyberattacks in 2019. [online] Available at: a &lt;https://www.omanobserver.om/oman-thwarts-14-billion-cyberattacks-in-2019/&gt; [Accessed 25 February 2021].
2. Alhayani, B. et al. (2021) 'Best ways computation intelligent of face cyber attacks', Materials Today: Proceedings, (March). doi: 10.1016/j.matpr.2021.02.557.
3. Alom, M. Z. et al. (2018) 'The history began from AlexNet: A comprehensive survey on deep learning approaches', arXiv.
4. Andrade, E. D. O. et al. (2019) 'A model based on LSTM neural networks to identify five different types of malware', Procedia Computer Science, 159(December), pp. 182–191. doi: 10.1016/j.procs.2019.09.173.
5. Bagui, Sikha et al. (2019) 'Using machine learning techniques to identify rare cyber-attacks on the UNSW-NB15 dataset', Security and Privacy, 2(6), pp. 1–13. doi: 10.1002/spy2.91.
6. Catak, F. O. et al. (2020) 'Deep learning based Sequential model for malware analysis using Windows exe API Calls', PeerJ Computer Science, 6, pp. 1–23. doi: 10.7717/PEERJ-CS.285.
7. Churcher, A. et al. (2021) 'An Experimental Analysis of Attack Classification Using Machine Learning in IoT Networks', pp. 1–32.
8. Dang, D., Di Troia, F. and Stamp, M. (2021) 'Malware classification using long short-term memory models', ICISSP 2021 - Proceedings of the 7th International Conference on Information Systems Security and Privacy, pp. 743–752. doi: 10.5220/0010378007430752.
9. Das, R. and Morris, T. H. (2018) 'Machine learning and cyber security', 2017 International Conference on Computer, Electrical and Communication Engineering, ICCECE 2017, (February), pp. 1–7. doi: 10.1109/ICCECE.2017.8526232.
10. Ferrag, M. A. et al. (2020) 'Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study', Journal of Information Security and Applications, 50, p. 102419. doi: 10.1016/j.jisa.2019.102419.
11. Girinoto et al. (2020) 'Comparison of LSTM Architecture for Malware Classification', Proceedings - 2nd International Conference on Informatics, Multimedia, Cyber, and Information System, ICIMCIS 2020, pp. 93–97. doi: 10.1109/ICIMCIS51567.2020.9354301.
12. Hwang, R. H. et al. (2019) 'An LSTM-based deep learning approach for classifying malicious traffic at the packet level', Applied Sciences (Switzerland), 9(16). doi: 10.3390/app9163414.
13. Komatwar, R. and Kokare, M. (2021) 'A Survey on Malware Detection and Classification', Journal of Applied Security Research, 16(3), pp. 390–420. doi: 10.1080/19361610.2020.1796162.

14. Lallie, H. S. et al. (2021) ‘Cyber security in the age of COVID-19: A timeline and analysis of cyber-crime and cyber-attacks during the pandemic’, Computers and Security, 105, pp. 1–20. doi: 10.1016/j.cose.2021.102248.
15. Lee, Y. and Choi, S. (2017) ‘A Lightweight Malware Classification Method Based on Detection Results of Anti-virus Software’, pp. 5–9. doi: 10.1109/AsiaJCIS.2017.20.
16. Maleh, Y. (2019) ‘Malware Classification and Analysis Using Convolutional and Recurrent Neural Network’, (April), pp. 233–255. doi: 10.4018/978-1-5225-7862-8.ch014.
17. Maulana, R. J. and Kusuma, G. P. (2020) ‘Malware classification based on system call sequences using deep learning’, Advances in Science, Technology and Engineering Systems, 5(4), pp. 207–216. doi: 10.25046/aj050426.
18. Muhuri, P. S. et al. (2020) ‘Using a long short-term memory recurrent neural network (LSTM-RNN) to classify network attacks’, Information (Switzerland), 11(5), pp. 1–21. doi: 10.3390/INFO11050243.
19. Nadu, T. (2021) ‘Cyber Attack Type Detection Using Deep Learning Algorithm’, 12(7), pp. 854–857.
20. Schofield, M. et al. (2021) ‘Convolutional Neural Network for Malware Classification Based on API Call Sequence’, pp. 85–98. doi: 10.5121/csit.2021.110106.
21. Supriya Shende (2020) ‘Long Short-Term Memory (LSTM) Deep Learning Method for Intrusion Detection in Network Security’, International Journal of Engineering Research and, V9(06), pp. 1615–1620. doi: 10.17577/ijertv9is061016.
22. Tama, B. A. and Rhee, K.-H. (2017) ‘Attack classification analysis of IoT network via deep learning approach’, Res. Briefs Inf. Commun. Technol. Evol., 3(15), pp. 1–9. doi: 10.22667/ReBiCTE.2017.11.15.015.
23. Thakkar, A. and Lohiya, R. (2021) ‘Attack classification using feature selection techniques: a comparative study’, Journal of Ambient Intelligence and Humanized Computing, 12(1), pp. 1249–1266. doi: 10.1007/s12652-020-02167-9.
24. Thapa, K. N. K. and Duraipandian, N. (2021) ‘Malicious Traffic classification Using Long Short-Term Memory (LSTM) Model’, Wireless Personal Communications. doi: 10.1007/s11277-021-08359-6.
25. Varsamopoulos, S., Bertels, K. and Almudever, C. G. (2018) ‘Designing neural network based decoders for surface codes Accelerated BWA-MEM View project hartes View project Designing neural network based decoders for surface codes’, (November), pp. 1–12. Available at: https://www.researchgate.net/publication/329362532.
26. Vinayakumar, R. et al. (2018) ‘Detecting Android malware using Long Short-term Memory (LSTM)’, Journal of Intelligent and Fuzzy Systems, 34(3), pp. 1277–1288. doi: 10.3233/JIFS-169424.
27. Yang, C. T. et al. (2020) ‘NetFlow Monitoring and Cyberattack Detection Using Deep Learning with Ceph’, IEEE Access, 8, pp. 7842–7850. doi: 10.1109/ACCESS.2019.2963716.
28. Catak, F. O. et al. (2020) ‘Deep learning based Sequential model for malware analysis using Windows exe API Calls’, PeerJ Computer Science, 6, pp. 1–23. doi: 10.7717/PEERJ-CS.285.
29. Catak, F. O. and Yazı, A. F. (2019) ‘A Benchmark API Call Dataset for Windows PE Malware Classification’, pp. 0–7. Available at: http://arxiv.org/abs/1905.01999.

30. Varsamopoulos, S., Bertels, K. and Almudever, C. G. (2018) ‘Designing neural network based decoders for surface codes Accelerated BWA-MEM View project hartes View project Designing neural network based decoders for surface codes’, (November), pp. 1–12. Available at: https://www.researchgate.net/publication/329362532.
31. Zeng, G. (2020) ‘On the confusion matrix in credit scoring and its analytical properties’, Communications in Statistics - Theory and Methods, 49(9), pp. 2080–2093. doi: 10.1080/03610926.2019.1568485.
32. Learning, E. and Bhandari, A., 2021. Confusion Matrix for Machine Learning. [online] Analytics Vidhya. Available at: <https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/> [Accessed 8 August 2021].