Windows Portable Executor Malware detection using Deep learning approaches

MSc Internship
Cyber Security

Yogesh Bharat Parmar
Student ID: x18176402

School of Computing
National College of Ireland

Supervisor: Mr. Vikas Sahni

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Windows Portable Executor Malware detection using Deep learning approaches

Yogesh Bharat Parmar
x18176402

Abstract

Malware's are the main barriers in the growth of digital acceptance. Every system or device in the world is connected with the internet and internet became the main source of spreading malware's. In the windows operating system various types of malware's are found, detection of such malware in timely manner is a challenging task. Almost every executive file in the windows operating system is in the PE (Portable Executor) format. PE file begins with the header that includes the various information about the file such as type of application, space requirement, libraries used and many more. Our main goal, in this work is to detect the windows malware without relying on any explicit signature based methods. In order to solve such issues, we are using different types of deep neural networks. CNN, RNN and CONV-LSTM models are used in order to detect the malicious behaviour from the Portable executable samples. The target value can be either malicious or legit. The classification results of every model will be analysed and compared using different classification metrics and achieved a highest classification accuracy of 94.18% using CONV-LSTM model.

1 Introduction

Malicious software most commonly referred to as Malware is a kind of specifically designed software that is used to penetrate and cause disturbance or damage to any system or network without the proper knowledge of its user. Malware is the common terms that are practically used to denote all kind of cyber threats and intrusion. They are branched into various types but comes under two simple categories as stand-alone malware and viral infectors. These kinds of malware cause destruction to the system or the network and also lead to loss of valuable data and confidential files. They can also be classified into different types based on their functioning. Most popular infectors are trojans, adware, worms, backdoors, spyware etc. Installing themselves in the host system or network they try to spy and steal the information. With the growing technology and the advent of many evolutions in the IT, the domain has led to an increase in cyber-threat activities. As the protection mechanism gets stronger the malware gets typical to be identified. Detecting these intrusions can be a typical task and they are done by detecting the signals and signatures obtained from such malware. Some of the identification of methodologies and software is used to make out its presence. But nowadays the detection process is becoming very difficult because of the protection mechanism used by these infectors. They have multiple polymorphic layers and use many methodologies that will enhance itself and avoid whatever detection mechanism used by the host system. Many analysis that is carried out in the detection of malicious software using emulsion techniques is found in [1]. Some of the other classic methods are also discussed in his paper. To make the analysis, there are currently two methods widely practised in the detection of the presence of malware. They are the static and dynamic methods of analysis. They are carried out online with the assistance of knowledge experts. The outcomes of the process are compiled together and are known as the "signature". This method takes account of all static signals to analyse the process after they are designed before the implementation. Some kind of malicious software even tackles this process through the use of obfuscation. However, as an enhanced mechanism, the model of detection based on dynamic signals has been implemented as formulated by [2]. All these methods use some behavioural techniques to detect the presence of malware if found any. Some of them use hypervisor based on the instrumentation.

Other statistical techniques are also found to be employed in the detection process. They are the methodologies that generate the response based on the patterns that correspond to the behaviour of the malware. Opposite to that another of finding out the presence of malicious software is done with the aid of expert analysis which on the whole searches for the deterministic signatures done by [3]. As following the current age, the most commonly used types of malware detection systems are the signature-based and non-signature-based detection methods. Some unique bits are analysed by the signature-based methods in the identification process of malware but on the other hand, the non-signature based detecting model uses the whole process of reading actions and pre-define database to find the presence of malicious software. The first time that is the signature-based model needs constant updates and is limited only with its database but it has the most top quality of accurate precision. The main drawback of this method is that the identification time is more or less length and is compared to the range of the attack duration. While the other model called non-signature-based detection has the backlog of producing false results. It is observed that false positives and negatives are displayed sometimes but it is a rare phenomenon. But this technique also helps in the reduced identification time, prevents the window from being misidentified. They can also identify many kinds of malware issues like Zero-day etc, while this process can go completely undetected when using the signature-based method. Some of the pure software implementations are also discussed. They often perform the functions in the trusted computing base. Many times, the software used for malware detection is more vulnerable to the same cyber threats that every other system possess. The protection strategy can be disabled by malicious software in major cases as depicted by [4].

Windows is mostly used operating system, has the highest number of consumer base. The windows is mainly popular for its simple user interface and its executable file capability, which can run or install any program in a simple manner. Some of the intruders, misuse this capability by providing the malware executable files in order to perform fraudulent activities. For privacy and data protection it is very important to detect and cease these kind of activities in a timely manner. Traditional malware detection methods were mainly using the signature based methods. The disadvantage with signature based method is the change is single bit or just few bytes in malicious code can make the malware undetectable. Therefore, our main goal is to detect the Windows malware without relying on explicit signature database. In this work we are proposing the deep neural network based solution in order to accurately classify the executable samples as malware or legit. Convolution neural network, Recurrent Neural Network and Convolution with LSTM recurrent neural network has been implemented in order to perform a comparative analysis between the different deep neural networks also to find out the best one.

2 Research Question

- How deep learning algorithms accurately classifies the Malicious Portable Executors ?
- Which model provides the best classification results for detection of malicious portable executors and what is the maximum accuracy can be achieved ?

3 Literature Review

It is evident that the cyber-attacks have evolved along with the technologies and it is very difficult to detect and take them down using the existing antivirus and cybersecurity software. The ineffectiveness is observed in the identification of well-developed threats and the usage of endpoint sensors in tackling the above is frequently seen in the lower level enterprises. In the research model proposed by [5] enhanced model as a solution to the above-mentioned issue. They have used normal auto windows set up to detect the presence of malware. The audit logs are used in this process. They are cost-effective ones and provide better accuracy. They emit signals to detect the presence of malicious systems and notify their presence if found any. They can be used along with the other networks and additional cybersecurity antivirus software. This model provides more accurate results of nearly 83 per cent and very low false positives nearly about 0.1 per cent which takes a toll on and include about 70 per cent of all the malware detection. The audits that are used in this process do not record the types of malware found but detects them correctly. Audits like Albeit can be easily fed into any type of network and be grouped into multiple databases. This technique can be processed with less initial cost and also extensive research analysis is required to take this implementation in the long run. More significantly, the presence of external layers such as the one used for detecting the audit logs and the route of the cyber attacks etc may require extra time to make the defence after the identification of the presence of malware. [6] Xu in his research paper proposes a new method for the detection of the presence of malicious software by the GPU technique which is used for the analysis of the patterns that are observed while finding the malware's presence.

This method recognises the memory that is stored in the system when already encountered with malware in the past. By using this method, the desired visibility and enhanced security are acquired. Gotten feedback is used to detect the presence of malware signals. Most of the malicious software are of two main categories. They are the kernel rootkits and the assaults that are happened on the memory of the system to corrupt. The accuracy rate is found to be somewhere around 99 per cent and the developed model detects all kind of cyber threats and malicious presence. The kind of false alerts like the displayed signals of the false positive and false negative are found to be at the rate of less than 5 per cent on a comparative total. This method can be devised on a system very easily and can be used on the model of hardware-oriented identification of malware. They can be used on the system or a network. It sources all the needed information from the system's memory or the available online sources. It takes a call on the holistic approach model that helps in the better detection and identification process. Firstly, a trial process is done by including the test run of kernel server and the functions and the risks associated with it. The performance level that is observed on the rootkit is comparatively higher than all the existing previous models. The false positives were also observed to be present in very less per cent. Some memory manipulative threats are easily tackled by this proposed model. One of the greatest pluses of this approach is that it makes use of the machine learning algorithms to detect the malware's presence using signatures and other traditional human prediction features. It is the main asset in the process of the critical analysis process. Another proposal of malware identification is done by [7]. They have listed out the responsible details that are based on the deep neural networks are used in building this whole method.

The goal of this proposed technique is to enhance the predictability and the rate of accurate identification. It tests all the generic components and layers before giving the notification of the presence of malware. It also assures lower false positives and negatives. The testing process is also done at a very quick rate stating the effectiveness of the particular model that is proposed. The percentage of malware detection is found to be 95 per cent on the rough calculation and the prediction of false positives and negatives is somewhere around the range of 0.1 per cent. the input consists of around more than 450,000 database that has the information of malware codes and types that are obtained from various companies and cyber treat security services. Also, while taking the count the method checks the process of generation of false positives and negatives at the endpoint. By doing such a method the live displaying of data from the input along with the process of analysis done for the newly obtained data. The user can also have access to the number of data that is being collected and stored in a day. The explanation is also given for the detection of false positives and negatives if alerted about any. The given data about the false intimation is then converted into a storage tank with the display of threats ranked in order. Nowadays the improvement and evolution in the technology have taken hold of nearly all the malware detection and prevention process using all the mechanisms of AI. Such a process is done using the protocols of cloud security in the network and many other mechanisms that are alike. While accessing the cloud data, the servers store all the dynamic memory files which traces and find out the same samples from the input database.

Once the system finds the presence of any cyber intrusion or the malware it uses the registered memory. Likewise, it also tracks information from thousands of clients and companies. In another model, the neural networks are employed in the identification mechanism to find out the portable execution files that are hidden and not visible. It analyses all the files and classifies them as malware or normal files based on the functioning of the library cells while reducing the unsearched spaces in the domain of machine learning for the identification of the presence of malware. Also, this model shows us the required precision and ongoing analysis that is comparatively better than other existing research models. The overall precision observed by the model is around 98 per cent on the whole while making the analysis and identification process in a dataset which contains more than 4000 sets of data which is included in the set file. The packers are not well defined in the made method model which in turn affects the working flow of the project and the usage of all other features. This proposed model is exhibited by [8]. To extract more files and proceed with the method of classification the outcomes obtained from the hybrid analysis can be of great help. However, to classify the unidentified data the neural network is employed. Another machine learning approach that deals with the same process of identifying the missed out portable execution files with accuracy and a higher rate of detecting the presence of malware. The static method is found to be used in the methodology proposed by [9]. It just groups all the features by the process of integration of datasets and units that deals with the major PE files present in the domain. Every attribute is measured in the process of analysis including the methods of accuracy, the F-measure and the rate of its precision. Their initial research displays all the grouped attributes that are established to find all the unidentified protocols of malicious softwares. The dataset's accuracy is found to be in the range of 92 per cent and the relativity is near to 84 per cent out of all the techniques. The method of random forest is also used in the detection process. Another non-signature method involved in the identification process of malware using the neural networks and other portable execution files in the taken window samples. The process involves the set of integrated attributes, some hybrid models and heuristics that are the main parts of the framework in non-signature-based models. Some of the structural and behavioural patterns are analysed in the process of engaging machine learning protocols in the analysis of the presence of malware if found any. Some of the simple systems are used in the creation of such a model as interpreted by [10]. Every cyber threat or malicious software has a special signature or a pattern when it enters the host system or network. Such a pattern if analysed properly can help in the quick detection of the presence or intrusion of malware very easily. Some of the main features that are used in this operational process are Opcode-n-gram, the strings that are in the PE files and some of the typical classifiers and byte-n-gram etc. These will help in the identification of the files and to determine whether they are malicious are not. This is done by a thorough reading of the files and headers of the particular data type. The framework tells the specification of execution data and normal data that is present in the windows sample taken for the analysis. These files that are present in the windows are called as portable execution files and common object files sometimes short-termed as PE and COFF. The PE is mostly vast and contains many headers and features that are base for the detection mechanism. Those can be complied and fed into the system memory just from the place where they are located. The segments that are present in the PE is also huge and has more attributed to data for example constants, variables etc. The PE also provides the user with the total description of all the API that are uploaded and for time being is stored in the library of the device. The below diagram represents the working and the description of the PE header working the samples of the windows.

While approaching for the models for the detection of malware two kinds of techniques can be analysed. These major methods are employed in many of the approaches and it is proven to be well administered. They two major types are static analysis and dynamic analysis. The first method that is static analysis in which all the data samples and the files are analysed for the presence of malicious software. The second type, the dynamic analysis is the process where the examination is done on the samples while the execution of the file is in the process. The static analysis is found to be the most appropriate method that is used for the identification of malware that is mostly used in the commercial industry of cybersecurity services. It is usually done by applying the malware samples of datatype based on the bites level and the pattern of signatures present. The method of blacklisting is very efficient and well-deployed and is found to be accurate. The accuracy is well improved and the detection and display of false positives and negatives are found to be lower on a comparative scale. The detection rate is found to be around 90 per cent in the model proposed by [11]. Since every method of identification is done based on the

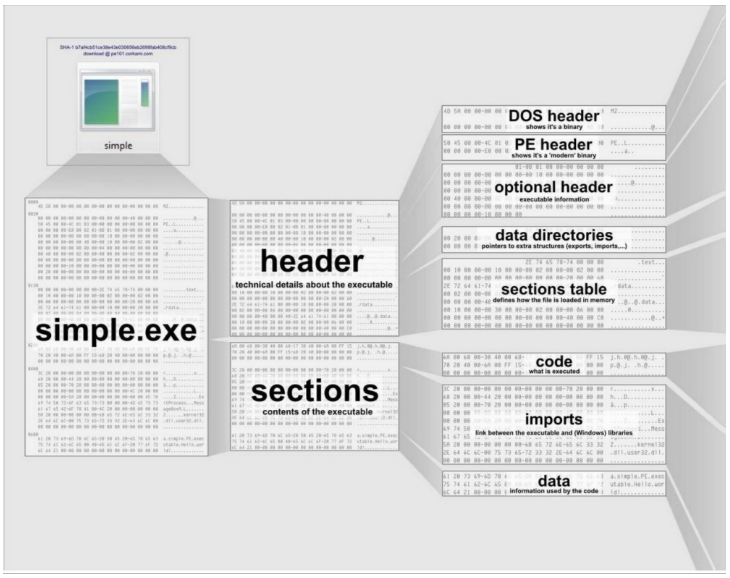
simple
simple.exe
header
technical details about the executable
sections
contents of the executable
DOS header
shows it's a binary
PE header
shows it's a 'modern' binary
optional header
executable information
data directories
pointers to extra structures (exports, imports,...)
sections table
defines how the file is loaded in memory
code
what is executed
imports
link between the executable and (Windows) libraries
data
information used by the code

Figure 1: PH header

signature samples no higher modifications or changes can be done because it will alter all the signature patterns that are taken for the analysis. It will alter all the binary and compiler codes that are used for the process. Many more advanced and complex methods can be used for the implementation technique. Some of the other changes in the analysis can be done at a very low budget. The outcome is found to be accurate but other lags are found between the analysed malware and detection process. The antivirus engines are processed to detect it within the swift amount of time. This model is found to be better doing than the existing models and the methodology was proposed by [12].

The static approach is complemented by the addition of a dynamic approach along with it. The behaviour pattern is completely analysed and then the detection method is properly done. The analysis is smoothly done without errors and the good step of the dynamic method is to recycle the present antivirus technique. This software is taken into account by the automatic process in the virtual zone. After the virtualisation then the samples are allowed to run on the endpoint and detect the anomaly and prevent it from interfering into the system. Making the automated variety run along with the other detection software can be very difficult on a larger scale. The process finds it very difficult to detect it while it is present and being in the sandbox. It blocks the malicious activities that are found in the host system and it complexes the functioning between the dataset and the automated dynamic behaviour. Another method of taint analysis is also a concept of the dynamic method that is proposed by the [13]. This model involves the implementation of a system on the automatic basis and its set is applied on its own. While designing an enhanced model for the detection of malware presence the dynamic behaviour must be completely used for better results. A complementary approach is also carried out in the identification of such anomalies and it penetrates all the layers to block and reverse the malware with the help of antivirus protocols. These functions use the endpoint and analysis the complete behaviour of the dataset and have no other extra option but to block and shut down the complete server to take action as a defence mechanism. Rather concentrating on the customised approach in developing a proposed model [14] in their paper puts forward the method of a model that solves the practical difficulties that stop the enhancement of designing an accurate and effective method for the detection of the presence of malicious software in the host system or network. For this method, no additional maintenance will be required but it functions effectively at low cost with moderate care and supervision. In this research work, the negative issues are tackled and lessened by using the enhanced superior activity of logs that are collected as the windows PE samples. The most audit logs are present in the SIEMs collection and are utilised well. Some of the other approaches try to limit and lessen the data samples to show accuracy as higher. But with the use of windows audit, this model takes the data into the administration level in the network without the want for the installation of any other external software. It also reduces the load in the network as well as the overload of the system.

4 Methodology

Our proposed methodology, uses the multiple deep learning models in order to detect the malicious portable executable. A series of multiple steps has been performed in order to implement the proposed framework. A flow diagram of proposed architecture is shown in Figure 2. The steps includes data collection, data pre-processing, feature extraction, feature selection and training of model. We will discuss about each steps in upcoming sub-sections.

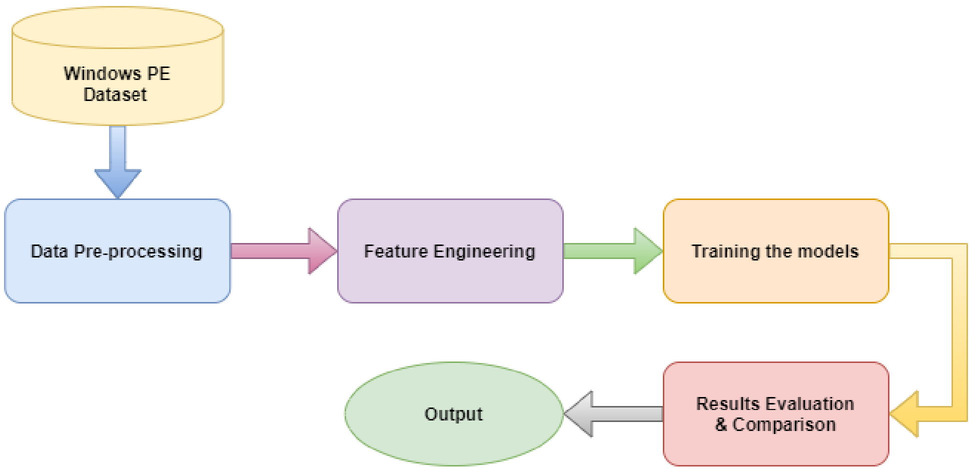

Figure 2: Proposed Framework for detection of Malicious Windows PE

4.1 Data Collection

The windows portable executor data file has been collected from kaggle. The dataset contains more than 200,000 Windows PE Samples with 486 features without including the target variable. The Windows PE samples can be either malicious or benign. Some of the important features of the dataset are has_configuration, has_debug, has_exceptions, has_exports, has_imports, has_nx, has_re-locations, has_signature, has_resource and many more. The target features can be classified in binary format as malicious or not-malicious. The dataset contains the discrete and continuous features. Overall the size of dataset is more than 340MB.

4.2 Data Pre-processing

Data Pre-processing is the first and most crucial step of data analytics. A noisy data can reduce the accuracy of the model to a great extent. The dataset also contains many null values, which has been dropped from the dataset. Also label encoder technique has been used in order to convert the discrete values into the numbers. The correlation value of features in the dataset has been calculated, the highly correlated features should be remove from the dataset.

4.3 Feature Engineering

There are mainly two kinds of feature engineering techniques we have used, it includes feature extraction and feature selection methods. The dataset already contains 486 features, extracting more features is not a feasible option as the number of features are already more. Highly correlated feature in the dataset affects the accuracy of model. Also as the number of features are very high and all the features in the dataset may not be useful therefore in order to select the important features we using principle component analysis (PCA). It simplifies the complexity for high dimensional data while retaining the trends and common patterns. We have applied principle component analysis in order to reduce the dimensionality of data. After implementing PCA, we achieved the optimal number of features as 85 (including the target feature) shown using graph in Figure 3.

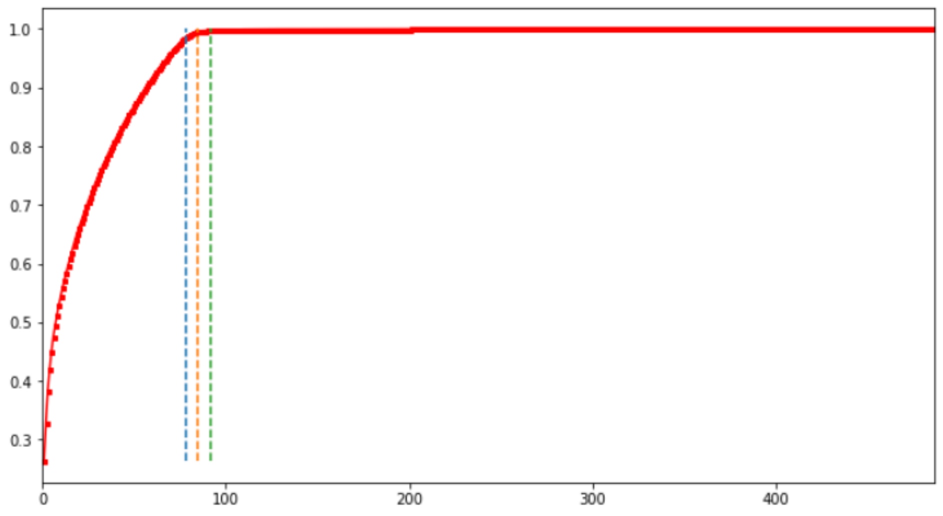

Figure 3: Dimensionality reduction using PCA

After reducing the dimensionality of dataset, there are no highly correlated features available in dataset, that can be shown with the help of histogram in Figure 4.

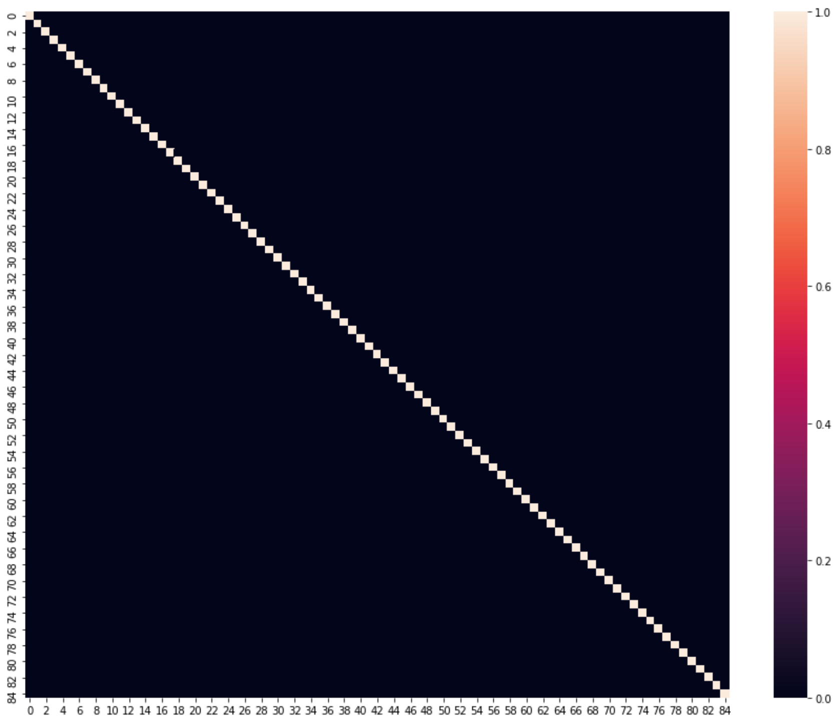
Figure 4: Histogram for representing the correlation between Features

4.4 Model Training

After performing data pre-processing and feature engineering steps the filtered data will be provided to deep learning models to perform predictive analysis. In our experiment we are using 3 different models for training and performing a comparative analysis. The models are Convolution Neural Network (CNN), Recurrent Neural Network (RNN) and Convolution-LSTM which is basically a combination of CNN and Long short term memory (recurrent neural network). The 70% of the data is used for training whereas, remaining 30% of data will be used for testing purposes. The functionality of each of the model will be explained in the Design section.

4.5 Evaluation of results

After training the model, the obtained results will be evaluated using various performance metrics, which includes Accuracy, loss, precision, recall and F1-score. These metrics will be calculated for every individual model in order to find out the best one. The definition of each of the metrics is explained in below subsections.

4.5.1 Precision

Precision is the primary metrics, used for the binary classification. To calculate the precision, Correctly predicted positive values are divided by total predicted positive values. Low false positive rate indicates a higher precision score. It can be defined as :

$$Precision = TruePositive/(TruePositive + FalsePositive)$$

4.5.2 Recall

Recall score is mainly used to calculate the sensitivity of model. It represents the false negative values generated by the model. The recall score can be defined as :

$$Recall = TruePositive/(TruePositive + FalseNegative)$$

4.5.3 F1-score

F1-score combinely calculates the false positive and false negative observations. It is mainly obtained from precision and recall values. The F1-score can be defined as :

$$F1score = 2 * ((precision * recall)/(precision + recall))$$

4.5.4 Accuracy

The accuracy is the ratio of correctly predicted value divided by total observations. It can be defined as :

$$Accuracy = (TP + TN)/(TP + FP + FN + TN)$$

4.5.5 Loss

To calculate the error of the model during the optimization process a loss function is calculated. Sometimes, the loss is inversely proportional to accuracy of the model.

5 Design Specification

The architecture and design of each model used in our experiment is different from one another. We will discuss about the architecture and design of each model in following subsections.

5.1 Convolution Neural Networks (CNN)

Convolution Neural networks are the type of deep neural networks, which consist of input, output and hidden layers. The complexity of the model, mainly depends on the number of inputs and number of hidden layers. The more number of hidden layers, more will be the complexity of model. CNN model is mainly used for object detection, image processing and image classification purposes. Most commonly, ReLu is used as the activation function in CNN. The architecture of CNN is shown with the help of diagram in Figure 5.

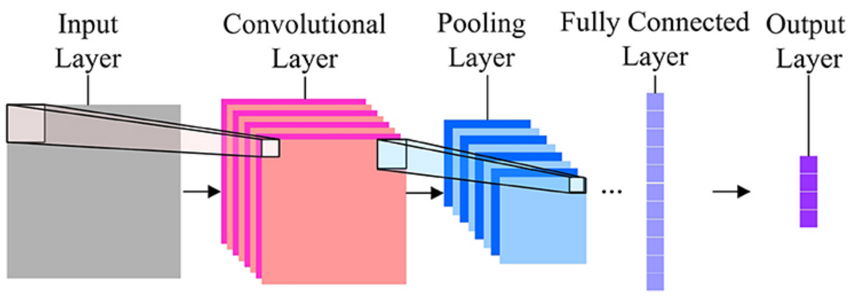

Figure 5: CNN Architecture

5.2 Recurrent Neural Network (RNN)

The inspiration for recurrent neural network has been taken from multilayer-perceptron model. In recurrent neural network output from previous steps are fed into the current step, that is the reason it is called recurrent model. Recurrent neural networks are very popular for processing time-series data. RNN transforms the independent activation to dependent function by assigning the same weights and biases to all layers. The output of previous layer becomes the input of another layer in RNN model. The architecture of recurrent neural network is shown in Figure 7

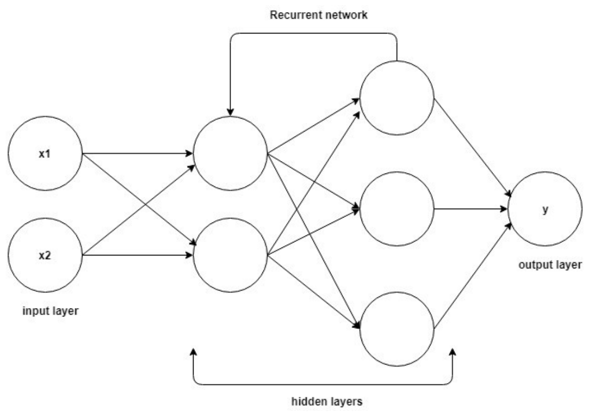

Figure 6: RNN Architecture

5.3 Convolution LSTM Model

Convolution LSTM is mainly designed for sequence prediction problems. Where the CNN layer is mainly used for feature extraction over input data and later,it is combined with LSTMs to support sequence prediction. From the previous studies, we have found that the combination of CNN model with LSTM provides the better classification performance as compared to other algorithms. Therefore, we has selected this combination for our prediction.

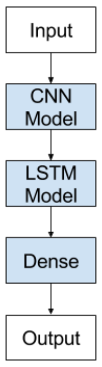

Figure 7: Convolution LSTM model Architecture

6 Implementation

The experiment was performed in a single machine, where the programming language is used as python3. Along with the python3, we have used jupyter notebook for live code and visualization. Predicting Windows Portable executor is a binary classification problem. Therefore, all the deep learning models, which were best suited for classification task have been utilized in our experiment. The dataset has been divided into training and test set with the ration of 70:30. All the deep learning model has been trained for 25 epochs. The ReLu is used as the activation function for all the models. The following specifications are used to implement the proposed framework :

- Main memory (RAM) : 6 GB
- GPU : 1xTesla K80 @ 2.20 GHz
- Number of Cores : 2
- Hard disk : 40GB
- Operating system : Windows 10
- Programming Language : Python
- User Interface : Jupyter Notebook
- Libraries Used : Pandas, numpy, matplotlib, sklearn, keras and seaborn

7 Evaluation

The Evaluation has been performed over 200,000 Windows portable executor samples. Three different deep learning algorithms Convolution Neural Network (CNN), Recurrent Neural network (RNN) and Convolution with LSTM recurrent model have been implemented. The precision, recall, accuracy, f1-score and loss has been calculated for each model. In every experiment we will discuss evaluate a metric performance for all the 3 models, which will be running over 25 epochs. The cross-validation techniques has been used in our proposed approach. Therefore, the results for every metrics will be evaluated for training set and validation set as well. Metrics on training set will represent the progress of model in terms of training. Whereas, from validation set the quality of model is measured in order to make the new predictions.

7.1 Experiment 1 / Precision, Recall and F1-score Comparison

In the first experiment we have run the model over 25 epochs and calculated the precision, recall and f1-score for every model to have comparative analysis. The Precision scores for all the 3 algorithms is shown in Figure 8.

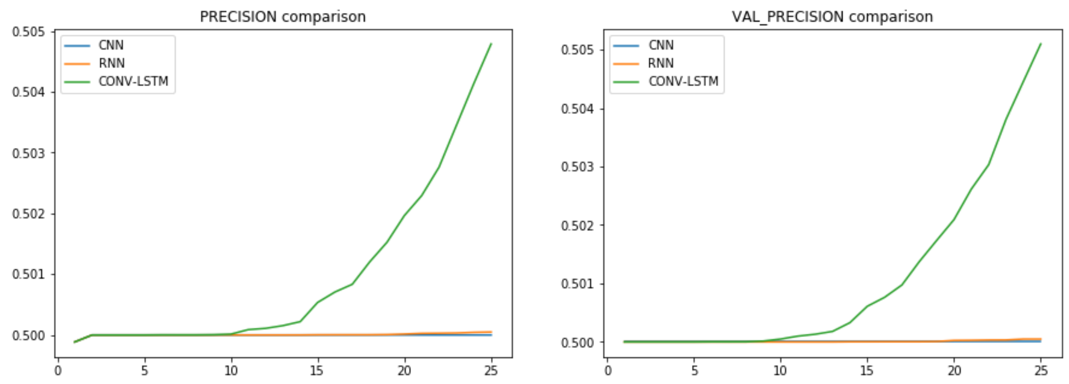

Figure 8: Training and Validation Precision Comparison

The maximum precision score obtained is 0.505 using Convolution-LSTM model for both the training and validation set. From the graph it has been also observed that, on increase in the number of epochs the precision score for CONV-LSTM model is also increasing. Whereas for CNN and RNN, the precision score is constant. The lowest precision score by all model is 0.50. Lower precision rate clearly indicates high false positive rates in prediction. Whereas, in case of recall score, 0.99 highest recall score has been noted for both training and validation set as shown in Figure 9. The highest recall value by all the models represent the low false negative observations.

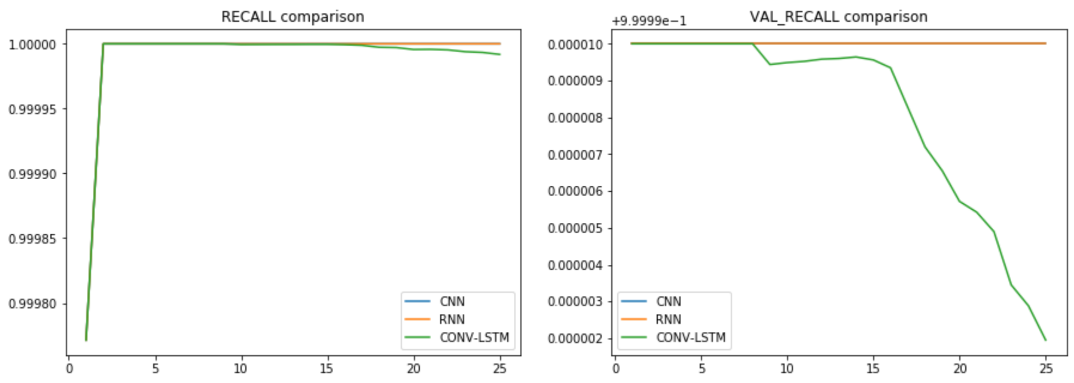

Figure 9: Training and Validation Recall Comparison

The F1-Score Comparison for all the models is shown in Figure 10. On increasing the number of epoch, we have observed a significant rise in the f1-score for all the algorithms.

The f1-score achieved by CNN model is 0.9005 and the F1-score for RNN model is 0.8932 for validation set. The highest f1-score of 0.9210 has been achieved by Convolution-LSTM model.

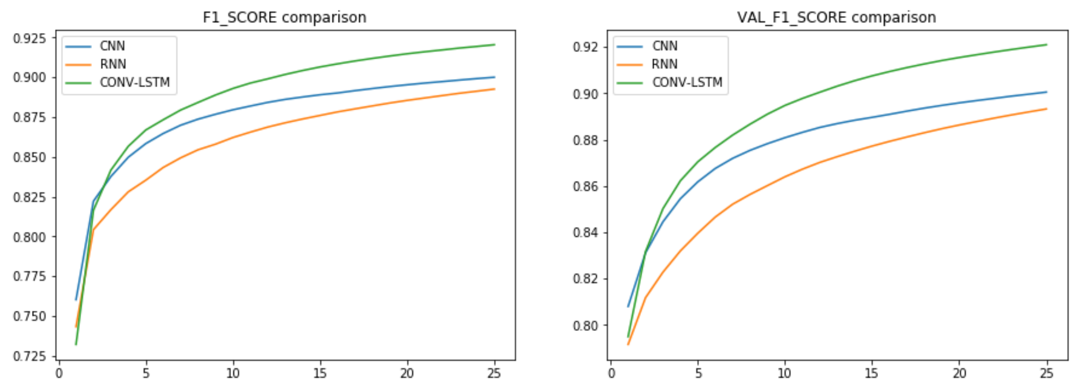

Figure 10: Training and validation F1-score Comparison

Overall, we can conclude that highest precision, recall and f1-score has been achieved by Convolution-LSTM model. It means with help of convolution LSTM we can achieve, lowest false positive and false negative values.

7.2 Experiment 2 / Loss Comparison

Loss is mainly anti-proportional to accuracy of the model. If there is reduction in the value of loss for every epoch, there is a high chance of increase in accuracy over every epoch. In the Figure 11, we have found a decrease in loss value for every epoch. Minimum loss, higher the model performance. For validation set, the lowest loss achieved is 0.1596 using Convolution-LSTM model. In the seconds place, RNN has achieved the better results with loss value of 0.1868 for validation set.

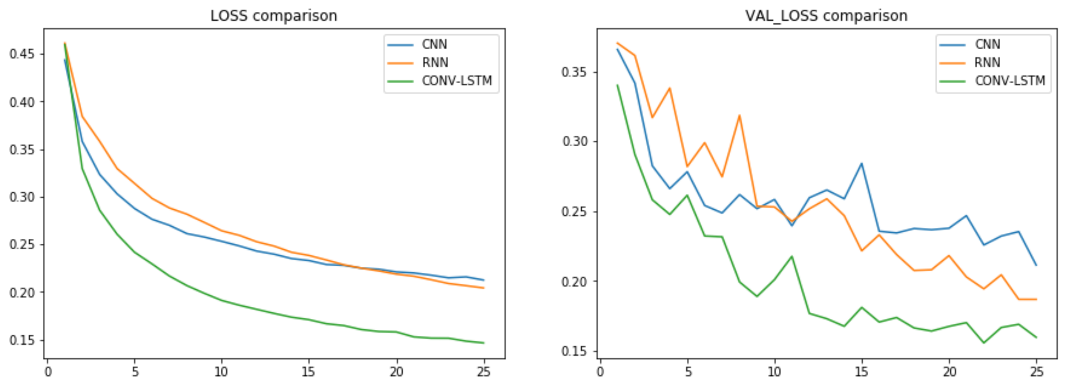

Figure 11: Training and validation Loss Comparison

7.3 Experiment 3 / Accuracy Comparison

Accuracy is one of the important measure for predictive analysis. We have calculated the accuracy for all the 3 algorithms over every epoch. We have found a increase in accuracy score, over increase in the number of epochs for both the training and validation set. A validation accuracy of 92.91% has been achieved by CNN model. Whereas, RNN provided the accuracy of 92.97% and again a highest accuracy of 94.18% has been achieved by convolution LSTM model for validation set. An accuracy comparison graph for all the deep learning models have been shown in Figure 12

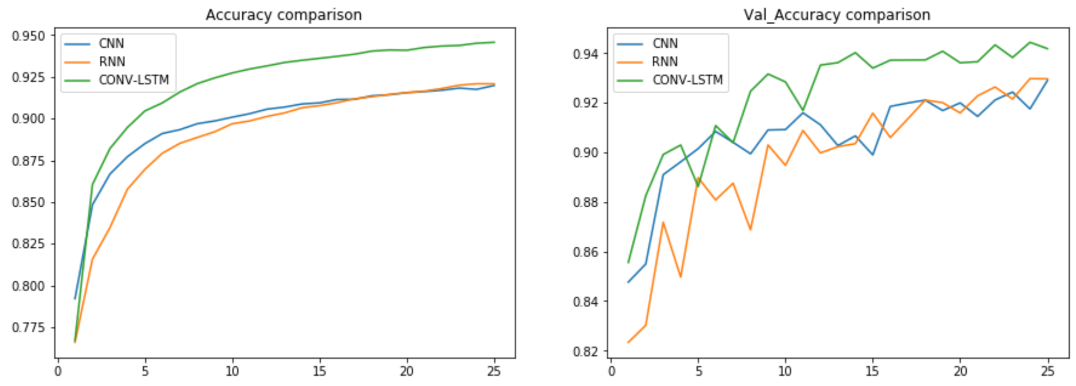

Figure 12: Training and validation Accuracy Comparison

7.4 Discussion

On Comparing all the metrics over 3 different deep learning models, we can conclude that combination of Convolution network with LSTM recurrent neural network provides the high performing results. Whereas, CNN and RNN model achieved almost same accuracy for training the model over 25 epochs which is definitely less than the convolution-LSTM model. The precision value of all three models are found to be very low, it means our prediction will contains more false positive values. The recall score of every model is very high, so can conclude that our prediction will contains less false negative values. We have plotted a confusion matrix for highest performing model, which is Convolution-LSTM model. The confusion matrix of CONV-LSTM model is shown in Figure 13.

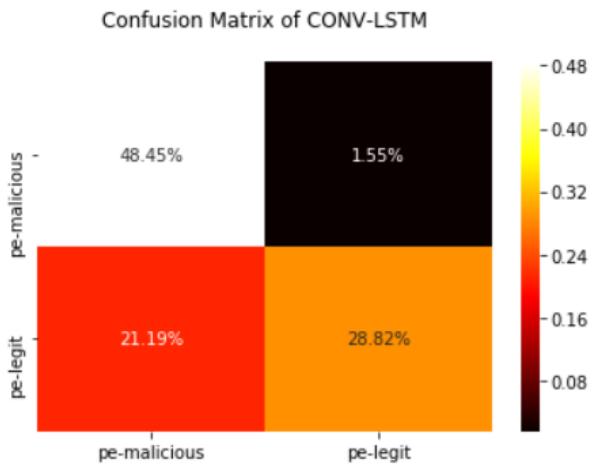

Figure 13: Confusion matrix of CONV-LSTM model

8 Conclusion and Future Work

We have performed various experiment and analysis in order to know the best model for malware detection in windows portable executor. We have trained the 200,000 Windows PE samples over the different deep learning models and found that the combination of Convolution network with LSTM recurrent neural network performs the best, in terms of every metric used such accuracy, loss, precision, recall and F1-score. A highest accuracy of 94.18% has been achieved, which is very much efficient to be used for detection of malware in windows operating system. The sensitivity score for every model is very high, therefore prediction score contains the false negative values only 1.55% for CONV-LSTM model. In future work, more combination of different deep neural network can be explored in order to achieve the more better performance. In production system, the data size might be more in size, which will require high computation power. To solve such issues, scalable solution such as hadoop and spark framework can be utilized.

References

[1] K. Rieck, T. Holz, C. Willems, P. Düssel, and P. Laskov, "Learning and Classification of Malware Behavior," in *Proceedings of the 5th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, ser. DIMVA '08. Paris, France: Springer-Verlag, Jul. 2008, pp. 108–125. [Online]. Available: https://doi.org/10.1007/978-3-540-70542-0_6

[2] G. Jacob, H. Debar, and E. Filiol, "Behavioral detection of malware: from a survey towards an established taxonomy," *Journal in Computer Virology*, vol. 4, no. 3, pp. 251–266, Aug. 2008. [Online]. Available: https://doi.org/10.1007/s11416-008-0086-0

[3] C. Wong, S. Bielski, A. Studer, and C. Wang, "Empirical Analysis of Rate Limiting Mechanisms," in *Recent Advances in Intrusion Detection*, ser. Lecture Notes in Computer Science, A. Valdes and D. Zamboni, Eds. Berlin, Heidelberg: Springer, 2006, pp. 22–42.

[4] "Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model - ScienceDirect." [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1877750316305099

[5] K. Berlin, D. Slater, and J. Saxe, "Malicious Behavior Detection using Windows Audit Logs," *arXiv:1506.04200 [cs]*, Aug. 2015, arXiv: 1506.04200. [Online]. Available: http://arxiv.org/abs/1506.04200

[6] Z. Xu, S. Ray, P. Subramanyan, and S. Malik, "Malware detection using machine learning based analysis of virtual memory access patterns," in *Proceedings of the Conference on Design, Automation & Test in Europe*, ser. DATE '17. Lausanne, Switzerland: European Design and Automation Association, Mar. 2017, pp. 169–174.

[7] J. Saxe and K. Berlin, "Deep neural network based malware detection using two dimensional binary program features," Oct. 2015, pp. 11–20.

[8] A. R. Mohammed, G. Sai Viswanath, K. Sai babu, and T. Anuradha, "Malware Detection in Executable Files Using Machine Learning," in *Advances in Decision Sciences, Image Processing, Security and Computer Vision*, ser. Learning and Analytics

in Intelligent Systems, S. C. Satapathy, K. S. Raju, K. Shyamala, D. R. Krishna, and M. N. Favorskaya, Eds. Cham: Springer International Publishing, 2020, pp. 277–284.

[9] R. Vyas, X. Luo, N. McFarland, and C. Justice, "Investigation of malicious portable executable file detection on the network using supervised learning techniques," May 2017, pp. 941–946.

[10] "Random forest classifiers: A survey and future research directions | Request PDF." [Online]. Available: https://www.researchgate.net/publication/284881039_Random_forest_classifiers_A_survey_and_future_research_directions

[11] A. Mohaisen and O. Alrawi, "AV-Meter: An Evaluation of Antivirus Scans and Labels," in *Detection of Intrusions and Malware, and Vulnerability Assessment*, ser. Lecture Notes in Computer Science, S. Dietrich, Ed. Cham: Springer International Publishing, 2014, pp. 112–131.

[12] G. V. V. i. o. o. t. founders, C. o. L. a. w. a. a. P. i. t. D. o. C. S. a. t. U. o. C. i. S. B. H. c. r. i. i. m. analysis, W. Security, V. Assessment, m. p. s. H. a. e. a. b. o. Security, M. Agents, a. o. o. I. C. H. h. b. t. P. C. o. t. I. S. o. R. A. i. I. Detection, o. t. I. S. o. Network, D. S. Security, o. t. I. S. o. Security, P. i. . H. i. k. f. organizing, r. a. i.-u. C. T. F. h. contest, c. iCTF, t. e. y. i. d. o. i. a. t. w. G. V. r. h. M. S. w. honors, P. D. f. P. d. Milano, Italy, I. 1994, 1998, r. H. i. a. m. o. IEEE, and ACM., "Antivirus Isn't Dead, It Just Can't Keep Up," May 2014, library Catalog: www.lastline.com. [Online]. Available: https://www.lastline.com/labsblog/antivirus-isnt-dead-it-just-cant-keep-up/

[13] A. Slowinska and H. Bos, "Pointless tainting? Evaluating the practicality of pointer tainting," Jan. 2009, pp. 61–74.

[14] H. Yin, D. Song, M. Egele, C. Kruegel, and E. Kirda, "Panorama: Capturing system-wide information flow for malware detection and analysis," in *CCS'07 - Proceedings of the 14th ACM Conference on Computer and Communications Security*, Dec. 2007, pp. 116–127. [Online]. Available: https://experts.syr.edu/en/publications/panorama-capturing-system-wide-information-flow-for-malware-detec