INTERNSHIP REPORT ON MALWARE DETECTION USING DEEP LEARNING

Interned under Dr. Shyam Lal

Electronics and Communication Department

National Institute of Technology, Karnataka

Submitted By:

Shreeya Sudhesh Rao

BTech in Computer and Communication Engineering

Manipal Institute of Technology

Contents

* Introduction…………………………………………………………………………………………………………3
* Methodology…………….…………………………………………………………………………………………4-21
* Results and Discussion………………….…………………………………………………………………………………………22
* References…………………………………………………………………………………………………………….22-23
* My role in the internship and skills gained……………………………………………………………23

INTRODUCTION

The aim of this internship was to build algorithms based on Deep Learning techniques, that detects malware. Malware is an undesired and harmful software that can be present in computers and pretends to be legitimate when in reality it can cause a wide range of cyber security problems. These malwares have to be detected and eliminated as soon as possible from the system and hence Malware detection is crucial for maintaining cyber safety and security. It provides protection from hackers and other forms of cybercrimes. This is extremely needed in today’s times as the number of cyber security issues have been rapidly increasing by the day.

4 algorithms were to be implemented based on Deep Learning that detected and categorized malware. This was done with the help of Google Colab and the Python language. The libraries present in Python for deep learning tasks had to be familiarised and the used. Different tasks like data pre-processing, creation, training and evaluating of models was carried out. Prior to this, a general skill building of image processing and deep learning algorithms was performed. For training and testing of the model, the Malimg dataset was used.

METHODOLOGY

1. Importing the dataset and using pre-processing it

Malimg dataset was collected and used for this project. Malware exists in systems as files consisting of binary data. In the dataset, these binary data from different malware types was collected and converted into grayscale images. It was observed that malware images from the same families showed similar characteristics and textures. This knowledge was important and based on this , the dataset could be used in the models to classify malware into their different families. There were different malware images belonging to 25 different families.

The datasets were mounted on Google Drive and imported into the notebook. Following this, several data pre-processing steps were performed such as converting the data into array format, slicing of the arrays as necessary and reshaping of the arrays to make it suitable for training and testing use. On the labels dataset, one hot encoding was also performed to make it more apt for model usage

1. Construction of the models

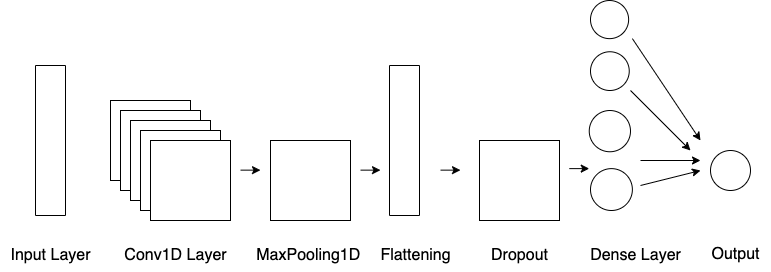
All the models constructed here are based on the deep learning architecture.

Deep Learning Models:

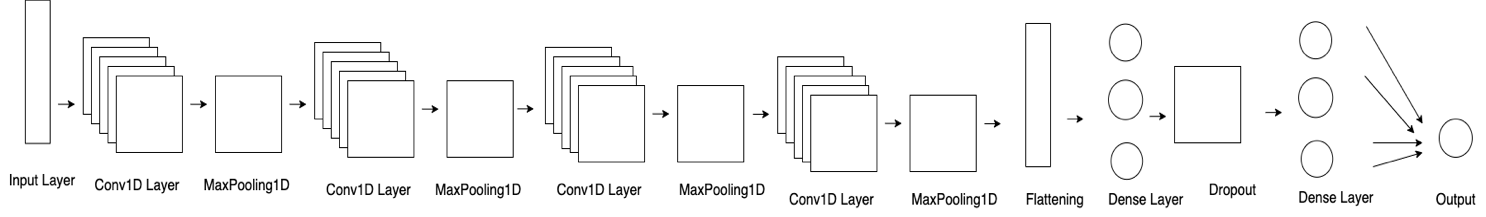
Deep Learning models are inspired from the workings of the human brain, and how the brain functions to classify, recognize and predict things. It efficiently makes meaning of large amounts of data by training itself. The deep learning models used here are based on Convolutional Neural Networks (CNN), and Recurrent Neural Networks (RNN). Convolutional Neural Networks are mainly used for spatial data and Recurrent Neural Networks are mainly used for temporal data. Models can use these as building blocks for malware detection algorithms.

Based on these building blocks, the different models developed here are 1D Convolution with one layer, 1D Convolution with multiple layers, Resnet 34 Model, CNN with one layer + LSTM ( Long Short Term Memory) , CNN with multiple layers + LSTM. These models were trained to achieve maximum accuracy and minimize the loss. All the models make use of keras and tensorflow libraries to get built.

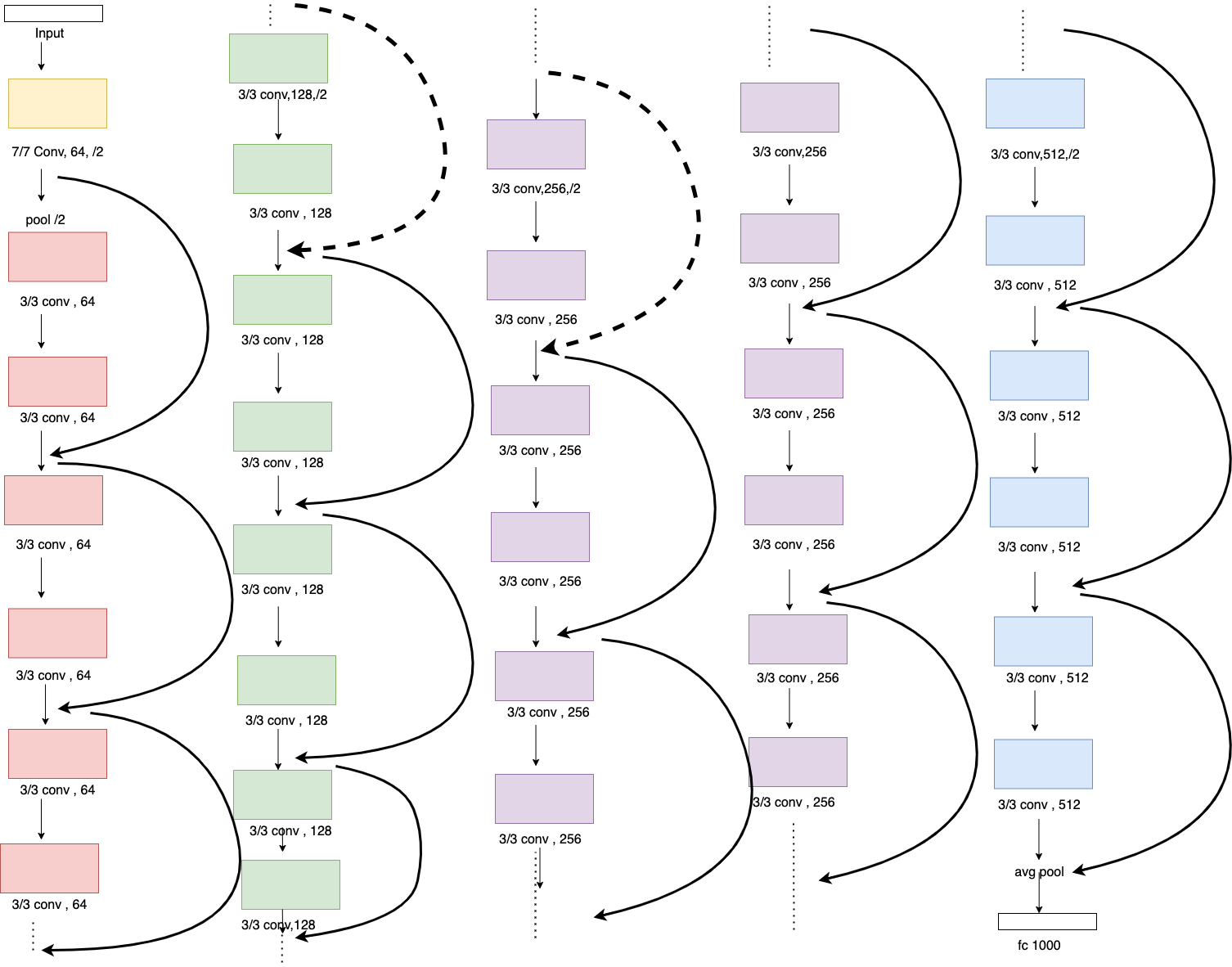
1D Convolution with one layer: This model is created with one Convolution Layer, the activation function used is ReLu and the same amount of passing is used. The output is then flattened and passed into a Dense Layer followed by dropout which reduces overfitting. The activation function used in this model is used SoftMax.



1D Convolution with multiple layers: This follows the same architecture as the previous one with the difference being that there are more than one Convolution layers present. The number of nodes in each Convolution layer are 64, 200, 180, 220, 200 respectively. After each Convolution layers, there is a max pooling operation performed with pool size 2. There is then a flattening operation performed, followed by two dense layers which nodes 150 and 25. The activation functions used in all the layers except the last is ReLu and SoftMax is used in the last dense layer. The number of epochs used here are 100.



Resnet 34 Model: Resnet34 is an image classification model, structured as a 34 layer convolutional neural network. RestNet is different from traditional neural networks in the sense that it takes residuals from each layer and uses them in the subsequent connected layers (similar to residual neural networks used for text prediction). Implementing this model requires defining of the functions called identity block and convolutional block and then the Resnet model function. This model is then used with 800 epochs.



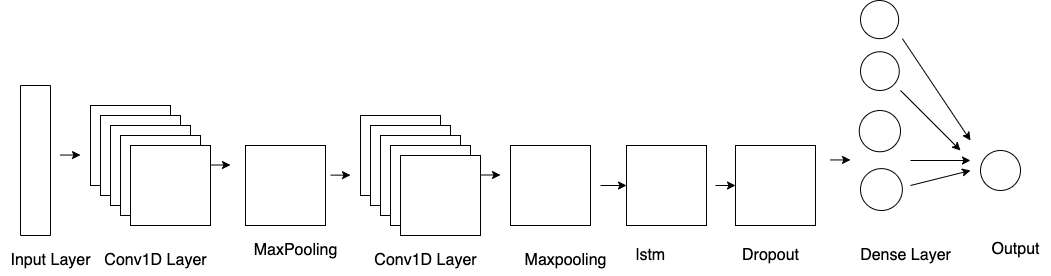
LSTM Model: Long Short Term Memory networks – usually just called “LSTMs” – are a special kind of RNN, capable of learning long-term dependencies. They work tremendously well on a large variety of problems, and are now widely used. LSTMs are explicitly designed to avoid the long-term dependency problem present in classical RNNs. We first create a model with one CNN layer and LSTM layer and another model with multiple CNNs layer and LSTM layer. This is done by adding LSTM layer to the above two CNN models created. 1000 epochs are used to train the model.

CNN 1 LAYER + LSTM MODEL

Box and whisker chart

Description automatically generated

CNN MULTIPLE LAYERS + LSTM



All the above models are compiled with loss = categorical\_crossentropy , Adam optimizer and metric used in accuracy.

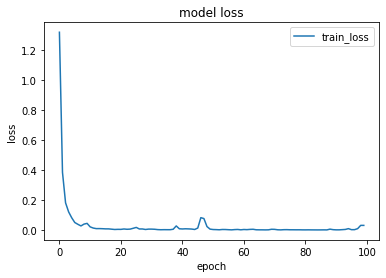
1. Performance evaluation

After training the model , the loss and accuracy are plotted against the number of epochs and visualized. The models are then tested and evaluated. Confusion matrix, classification report and number of mismatches are also computed. Precision, recall and f1 score are given importance to compare the different models.

ROC curves also plotted.

Conv 1D with one layer

Training loss

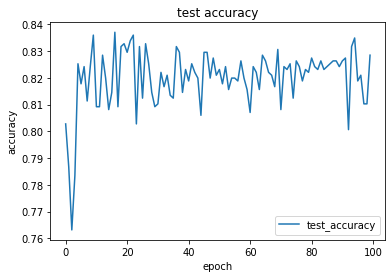


Training accuracy

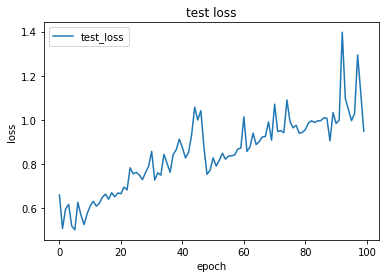
Shape

Description automatically generated

Test accuracy



Test loss



Shape

Description automatically generated

Conv1D with multiple layers accuracy and loss plots

Training accuracy

A picture containing graphical user interface

Description automatically generated

Training loss

Graphical user interface

Description automatically generated

Test accuracy

Text, whiteboard

Description automatically generated with medium confidence

Test loss

A picture containing text

Description automatically generated

Shape

Description automatically generated

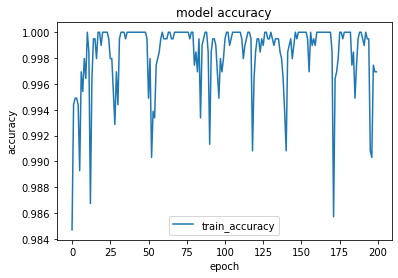
RESNET 34 MODEL

Training loss

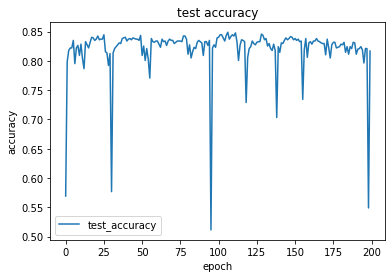
Graphical user interface

Description automatically generated with medium confidence

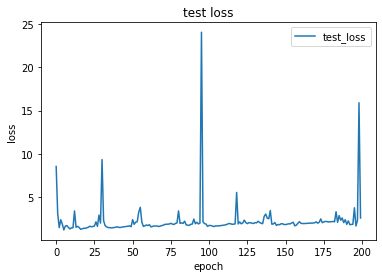
Training accuracy

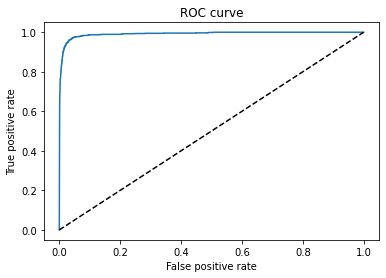


Test accuracy



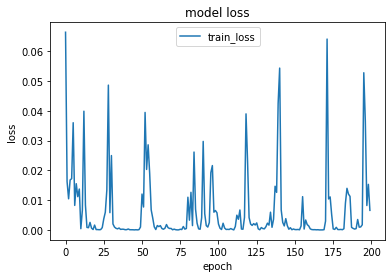
Test loss





LSTM + CNN 1 LAYER

Training loss



Training accuracy

Chart, histogram

Description automatically generated

Test accuracy

Chart

Description automatically generated

Test loss

A picture containing histogram

Description automatically generated

Shape

Description automatically generated

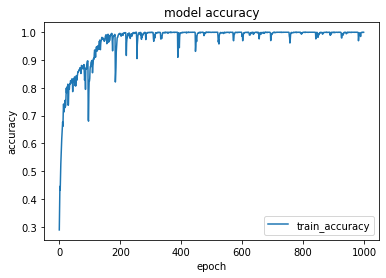
LSTM + multiple CNN layers

Training loss

Chart, histogram

Description automatically generated

Training accuracy



Test loss

Chart, histogram

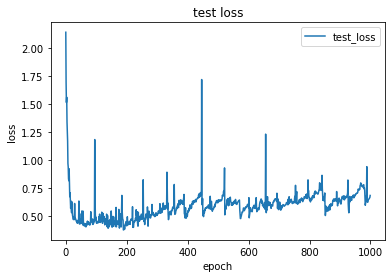
Description automatically generated

Test accuracy

Graphical user interface

Description automatically generated

Test loss



Shape

Description automatically generated

Results and Discussion:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model | Epochs | Accuracy | Precision | Recall | F1 score |
| Cnn 1 layer | 100 | 83% | 81% | 78% | 79% |
| Cnn multiple layers | 100 | 90.7% | 91% | 84% | 85% |
| RESNET 34 | 800 | 83% | 79% | 80% | 85% |
| LSTM + 1 CNN layer | 1000 | 92.5% | 86% | 84% | 85% |
| LSTM + multiple CNN layers | 1000 | 92.7% | 86% | 84% | 85% |

Malware detection is a crucial task that can prevent a lot of cyber-attacks. With careful detection, we can prepare the system to eliminate the malware.

The outcome of this paper is to implement algorithm that detects malware and categorises it into a class.

It takes in the malware image details as input and trains model based on that to correctly classify the malware input into a family. Data set pre-processing is carried out and then model is created to classify malware. Different models such as CNN , CNN + LSTM , RESNTET 34 model have been developed and each of them detects and classifies the malware with a good amount of accuracy.

References:

* <https://www.google.com/url?sa=t&source=web&cd=&ved=2ahUKEwjD8ZHEi_f7AhWDynMBHaOLDQQQFnoECA8QAQ&url=https%3A%2F%2Fmachinelearningmastery.com%2Fcnn-long-short-term-memory-networks&usg=AOvVaw1Owhmin7ZlYoHh5DC1kSDL>
* <https://www.tensorflow.org/tutorials/images/cnn>
* <https://www.analyticsvidhya.com/blog/2021/09/building-resnet-34-model-using-pytorch-a-guide-for-beginners/>
* <https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjlvYL1i_f7AhXI7HMBHYL6AdsQFnoECA8QAQ&url=https%3A%2F%2Fneurohive.io%2Fen%2Fpopular-networks%2Fresnet%2F&usg=AOvVaw1l8f-I3HwSzKpwq7BihE8B>
* <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>

MY ROLE IN THE INTERNSHIP AND SKILLS GAINED

I received training directly under Dr. Shyam Lal Sir and started the internship by gaining knowledge on image processing techniques. Following that, I learnt the basics of OpenCV and improved my python coding skills. I am now comfortable in using the various python libraries like NumPy, pandas, matplotlib, keras, tensorflow and so on.

I was exposed to different types of algorithms that can be used for image classification like CNN, LSTM ,RESNET 34 etc. . The techniques used to implement these models definitely improved my knowledge in Deep Learning. The topic of malware detection captured my interest from the very beginning and I was keen on working on a crucial issue like that. I also learnt how to tackle various challenges that I came across when training the models to achieve maximum accuracy.

The various learning resources and working environment provided was very favourable for my development and project progress.

Overall, this was a great learning experience to broaden my knowledge and to improve my skills in the field of Deep Learning.