MALWARE CLASSIFICATION INTO FAMILIES
BASED ON FILE CONTENTS AND CHARACTERISTICS

A Thesis Presented to the Department of Computer Science

African University of Science and Technology

In partial fullment of the requirements for the award of

Master Of Science Degree In Computer
Science

By

Jabir Shehu Minjibir

Supervised by
Prof. Lehel Csato

November 29, 2017

MALWARE CLASSIFICATION INTO FAMILIES
BASED ON FILE CONTENTS AND CHARACTERISTICS

By
JABIR SHEHU MINJIBIR

A THESIS APPROVED BY THE DEPARTMENT OF COMPUTER SCIENCE

RECOMENDED:...

Supervisor, Prof. Lehel Csato

....

Head, Department of Computer Science

APPROVED :..

Chief Acadamic Officer

November 29, 2017
Date

Abstract

The use of malicious software *(malware)* as an instrument for carrying out different criminal activities both organised and non-organised have become the major threat faced by today's world of connectivity. Frequency and complexity of such cyber-attacks makes it difficult for computer antivirus companies to efficiently handle the high value of the new malwares released using traditional approaches that depends mainly on signature. As a result, machine learning approaches are now the best home for this problem, and have demonstrate a great success. One of the challenges now is finding a method that is reasonably fast, and can practically adopted. In this work, we try some of the best machine learning models Convolutional Neural Network (CNN) in one of the new computer generation language namely Julia.

Acknowledgement

I owe my deepest gratitude to my father and mother for their constant counselling and guidance. To my family especially my brothers, I could not have made it without your immense and tireless support, I am really grateful. This thesis would not have been possible without my supervisor Prof. Lehel Csato who supported and encouraged throughout. I will also like to thank Mr. Charles, he has made available his support in a number of ways.

Contents

Table of Figures

Table of Tables

Chapter 1 Introduction

1.1 Overview

Following the exponential increase in the number of devices that are being connected to the Internet, criminal gangs from different parts of the world are taking advantage of this new development. They use the Internet as a medium for carrying out criminal activities, launching attacks against large corporations for monitory gain [**Error! Reference source not found.**] and against governments organisations and public infrastructures to make political statements. Due to this, the malware industry has recently gained momentum [**Error! Reference source not found.**], [**Error! Reference source not found.**], [**Error! Reference source not found.**] and is growing at an alarming rate.

Most of the organised cybercrimes depend solely on the use of different kinds of malware with varying complexity to infect the victim's machine, allowing them to gain full access and the ability to remotely control the compromised machine. These machines are connected to form a powerful network [**Error! Reference source not found.**], that can be called upon, and used for carrying out any number of attacks, in most cases for monitory gain.

Consequently, attackers (malware authors) are now using a network of infected devices, otherwise known as bot-nets [**Error! Reference source not found.**], as an opportunity to expand their scope and mode of attack to a higher level, with more sophisticated tools to launch attacks that are difficult to defend against. This problem leads to a regular appearance of new malware [**Error! Reference source not found.**], making it difficult for the computer security and antivirus companies to manually handle the large number of malware released effectively within a reasonable time limit [**Error! Reference source not found.**].

Nevertheless, malware authors are not always creating new malware from scratch. Rather, they use an automated technique [**Error! Reference source not found.**] to generate the malware from an existing malware that was known to have been successfully used in the past. The newly created malware contains some percentage of code so the predecessors share some characteristics [**Error! Reference source not found.**], making the new malware coding [**Error! Reference source not found.**] variants [**Error! Reference source not found.**] similar to old malware coding. The code snippet that is common to all the variants is called a signature [**Error! Reference source not found.**], [**Error! Reference source not found.**] which brings the malware together into a family.

The practical methods employed in the security industry combat this threat requires the accurate identification of the malware family [**Error! Reference source not found.**], which uses a manual process. This is not only time consuming, but it is also susceptible to detection evasion. With the current obfuscation technique used by the attackers [**Error! Reference source not found.**], these traditional methods are not keeping up with the current malware attacks [**Error! Reference source not found.**], [**Error! Reference source not found.**]; therefore, they are no longer efficient.

Comment [MT1]: Needs a date

Artificial Intelligence and machine learning on the other hand, are being considered as an alternative solution to the malware attacks. This research work adapts the use of machine learning technique to identify variants of malware and classify them into their respective families based on their file contents and characteristics.

1.2 Motivation

Apart from personal and private information that we often store in our personal computers and mobile devices, most of our critical infrastructures which includes banking systems, nuclear power plants, transportation and utilities are run and controlled by computers [**Error!**

Reference source not found.]. These infrastructures have become one of the primary targets for cyber criminals due to their high value to society.

Most of these criminals employ the use of malware to facilitate their criminal activity. For this reason, the task of securing this infrastructure ensures the well-being of society, and prevents the occurrence of any catastrophic event such as a nuclear attack by these perpetrators.

1.3 Problem Statement

More than 300 billion dollars is annually spent on damage costs as a result from the use of malware for organised cybercrimes e.g. DDoS attacks and non-organised cybercrimes e.g. stealing credit cards numbers. Forms of malware include the use of techniques like bot-nets [**Error! Reference source not found.**] and root-kits, which has led to the advancement of the malware industry and an increase in the number of malware authors. Consequently, there is an exponential growth of malware [**Error! Reference source not found.**], [**Error! Reference source not found.**], [**Error! Reference source not found.**], [**Error! Reference source not found.**],both in terms of complexity and number.

1.4 Objectives

The main objectives of this thesis include:

- Study the characteristics of malware based on its file content and characteristics.
- Train a classifier that can discriminate between different families of a malware given set of features of a malware.
- Improve the performance of existing methods of malware classification.
- Try new algorithms and probe for new ideas that can be useful in the classification process, or that should be avoided.

1.5 Report Structure

This report is structured into chapters from one to five as follows:
Chapter One introduces the research with a brief overview, definition of the problem to be solved and the main objectives of the research work.

In Chapter Two literature associated to this research work will be reviewed to explain the concepts used throughout the paper, find different methods that can be used to achieve the research objectives and provide supporting evidence for the research findings.

Chapter Three looks at the methodology used in the research. It explains the different algorithms, techniques and the possible modifications and adjustments that could be made. Implementation of the research will be explained in detailed. It also contains the methods used to prepare the data sets, different tools, how the environment was set up for the research, and how the research was conducted.

Result presentation and discussion will be presented in Chapter Four. The research observations, based on findings and the data set used for the research work will be explained. Chapter Five explains both the scientific and the technical challenges faced during the experimentation. Suggestions and future work will also be discussed in this chapter, and the conclusion will be discussed.

Chapter 2 Literature Review

2.1 Introduction

This chapter looks at various machine learning techniques that are used by other researchers, in order to see the structure that is carried out using different methods, in an attempt to classify malware into their respective families. The results obtained from those experiments as well the issues faced in both scientific and technical aspects will be used as a springboard for the research. Lastly, techniques were chosen and discussed that were used as a starting point for the research.

2.2 Malware

Malicious software commonly known as malware [**Error! Reference source not found.**] is any computer program or application that is purposely design to cause harm [**Error! Reference source not found.**] and execute against the interest of the host machine. Over recent years, malware have evolved through several forms. Modern malware now tends to be the most dangerous malware [**Error! Reference source not found.**3]. The major reason for this is that today's malware is mainly designed for monetary gain, and so the authors use advanced techniques for the design; which makes it harder to detect.

2.3 Major Categories of Malware

There are many forms of malware, but the following are the major categories of malware.

2.3.1 Virus

Viruses make the most noticeable damage to computers, which makes it the most popular among all the other types of malware. Usually, a virus stores itself as a single file and can transfer from one system to another via physical devices, mostly USB drives. Its operation ranges from popping multiple windows, disabling system ports, deleting important system files and crashing the whole system in some cases.

2.3.2 Worm

This type of malware is mainly known for two of its common behaviours. The first behaviour is that it can replicate itself multiple times and save the copies in many locations within the infected computer. Secondly, it can spread from one system to another within a short period of time, propagating without human intervention through network connection by either sending a copy of its self via email attachments, or even post itself as a link to websites, when an infected computer visits social websites. Its ability to do this makes it the fastest malware to spread compared to the other types.

2.3.3 Trojan Horse

The Trojan Horse is considered to be the most dangerous of all malware types due to its ability to mask itself as a legitimate program that appears friendly or even sometimes it claims to be protecting the user against threats, for example pose itself as an anti-malware. Purposely, Trojan Horses are designed to ensure that the host machine is always up and running so that the user will not suspect any unwanted activity, while they watch and harvest or collect vital and private information about the user which can include login details, credit card numbers and sometimes even steal the control of the system. They generally serve as a tunnel to transport information to and from the infected computer.

2.3.4 Hybrid

This type of malware has recently emerged. It comprises the functionality of two or more of the above types of malware. It is engineered to be robust and persistent and to get around most counter majors taken by anti-malware systems by either encrypting or modifying their code, changing their form dynamically and performing other tricks to pass any security check. Its operation is unpredictable, providing its author with a greater advantage to perform many types of operation.

The most common types of malware, are mentioned, however, currently it is difficult to draw a clear margin between these categories, especially with the emergence of the hybrid malware. Authors of the hybrid malware are using multiple techniques to create more of these multipurpose malwares.

2.4 Malware Classification

It was estimated that there is several billion dollars annually paid out in damage, caused by cyber threats, which are mostly carried out by malware. Antivirus and cyber security companies are constantly looking for sustainable ways of combating malware that is scalable and practically applicable. This research explores some of the prominent techniques that are currently used in malware classification.

2.4.1 Dynamic Analysis

Dynamic analysis has high results in detecting a class of a malware due to it is ability to observe and record the execution trace. Usually a controlled (sandbox) environment [**Error! Reference source not found.**], [**Error! Reference source not found.**] is set up then the malware allowed to run [Annachhatre et al., 2013]. During its execution, all its operations, resources request, network access attempt and memory usage are recorded. The result obtained is then analysed by an expert to find its signature and deduce its family.

This approach used to work when there were few samples observed daily; the occurrence was not as frequent as today. Despite its success, the biggest downfall with this approach is that it takes time and effort, and is inefficient considering the number of new malwares released daily.

2.4.2 Static Analysis

In contrast to dynamic analysis, static analysis focuses on the binary file structure, function length, function call, operation code [Annachhatre et al., 2013] and so on. By examining how the internal code is structured, static analysis can find a malware and its family.

Binary image texture analysis is also considered a static analysis technique as it involves the use of tools to convert the malware binary into an image. Image processing techniques are then used against the images obtained to find patterns and compute distance between malware images to find out if there are any similarities between the image at hand and any known family. This method is considered the best among the methods as it takes a reasonable time and the results are considered good.

2.4.3 Machine Learning

This technique is the state-of-the-art method used in almost all areas of analysis. It is a modern analysis technique that involves the use of both statistical and computer science to train a computer system how to automatically identify, classify or even how to carry out some antivirus tasks.

Machine learning has gained traction in both industry and academia following its promising results at various antivirus tasks. Several algorithms are available for machine learning, some of which are more suited for specific domains of problems and some can be applied to more than one problem domain with right tuning. Basically, there are two main types [**Error! Reference source not found.**] of machine learning namely: supervised learning and unsupervised learning. When solving machine learning problems, one of these techniques has to be employed, but sometimes the mixture of the two techniques can be applied, known as semi-supervised learning.

2.4.4 Supervised Learning

In supervised learning, the algorithm is fed with labelled data samples or an example is used as a guide [**Error! Reference source not found.**], the complete example data samples are called training sets. The process of training the classifier with the data samples [**Error! Reference source not found.**] is called fitting, and the larger the examples given to the algorithm the more the accuracy is improved. The classifier will adjust its internal parameters based on the individual labels assigned to each data sample in the training set. Supervised learning problems can be divided into two groups: either regression or classification.

Regression

A problem is said to be a regression if the value of the output from the classifier is a continuous value [**Error! Reference source not found.**] [**Error! Reference source not found.**], refer to Figure 2.1.

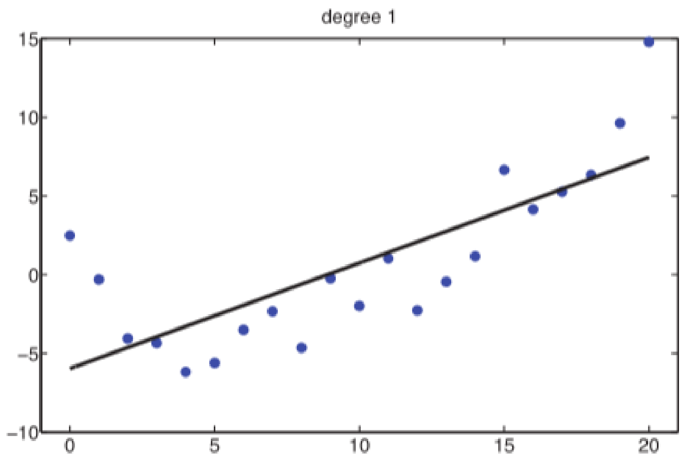

Figure 1 Regression

Classification

In classification, the value output of produce by the classifier is discrete [**Error! Reference source not found.**] [**Error! Reference source not found.**]. Classification can be a multi-class problem when there are many possible outputs [**Error! Reference source not found.**] [**Error! Reference source not found.**], refer to Figure 2.2.

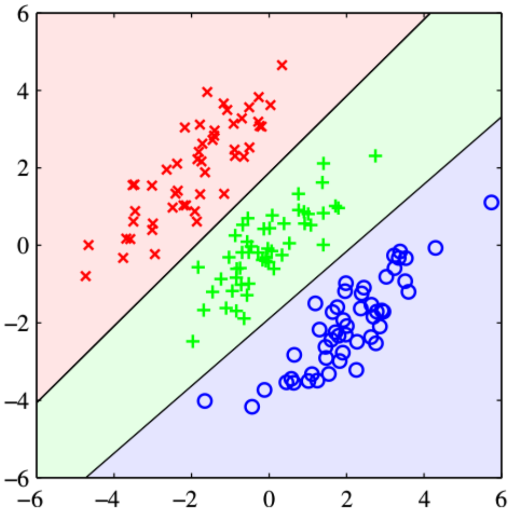

Figure 2 Classification

2.4.5 Unsupervised Learning

In contrast to supervised learning, unsupervised learning provides the raw data to the algorithm and it is the algorithm that detects the features [**Error! Reference source not found.**], the possible groups or classes that can be deduced from the data set. It then puts each sample from the training set into its respective class. Unsupervised learning is sometimes known as automatic feature detection.

2.4.6 Deep Learning

Deep learning is a special type of neural network that has many hidden layers, which has proven to be effective especially in image recognition problems [**Error! Reference source not found.**]. It can be used to train both supervised learning and unsupervised learning depending on the initial problem.

Convolutional Neural Network (CNN)

CNN is one of the most effective algorithms when it is provided with a sufficient training set. In most cases, it is used in image classification or recognition problems [**Error! Reference source not found.**]. However, it can be adopted to many problem domains with the correct turning. It leverages multiple neurons at different layers that learn only a fractional percentage of the object in question at each level, while passing what they have learned to the layer above them for further turning, refer to Figure 2.3. It demonstrates great performance and is practical in production, especially in image recognition, [**Error! Reference source not found.**].

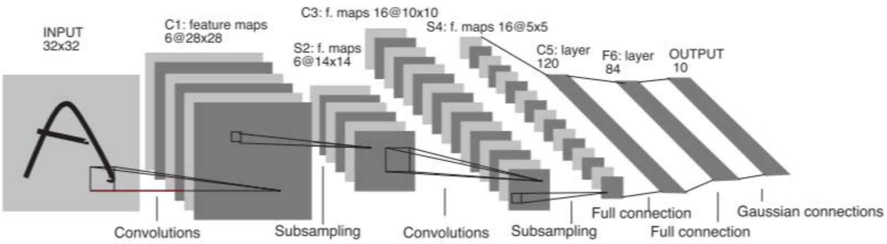

Figure 3 Convolutional Neural Network (CNN)

2.5 Related Works

Below are some the reviewed works, including a discussion on both their advantages and limitations.

In 2011 [**Error! Reference source not found.**], proposed a method of malware classification using a visualisation technique. The method works by converting the malware binary file into a grey-scale image, then applying image processing techniques. They observed images that belong to the same family of malware have a close texture pattern and layout. This follows the proposition of a method that does not require the execution traces or the disassembly file of the malware file. Classification accuracy of the approach is around 98% and has the ability to reveal the class of malware, even if an obfuscation technique is applied to the malware in order to disguise it. Originally, the method adopted from [**Error! Reference source not found.**] scheme recognition using spatial envelop.

Comment [MT2]: Something is missing

The major issue observed with this approach is that, due to its reliance on global-based image features, the malware authors with knowledge of how this classification technique works can devise a means of getting around, by either adding a lot of code that is not intended to do anything related to the main purpose of the malware itself or by spreading the main code to a location in the source that is initially not known.

The study by [**Error! Reference source not found.**13] employed the use of an approach by training multiple large neural networks with more than 2.6 million training sets. They obtained an impressive result that has two class error rates of 0.49 and 0.42% for single and ensemble neural networks, respectively. The research also used random projections to shrink down the size of the number of features (dimensional).

Unfortunately, this approach has two issues. Firstly, it shows no considerable improvement in performance by increasing the number of hidden layers. It was even observed to have slightly dropped in performance when the two and three hidden layer models were used.

Secondly, pre-training shows no positive effect in the performance of their model, which is common practice to use in pre-train data to achieve a higher gain in performance in deep neural networks. This makes the model harder to train.

An approach was proposed by [], which is based on examining the function length within a given sample of malware. The method also took into account the pattern and the shape of the functions, which they found to be of higher importance in setting a barrier between different

Comment [MT3]: Incomplete

classes. Using statistical methods, researchers were able to determine the specific features of any malware based on the length, pattern and shape of the function within that particular sample.

The greatest advantage of this method is that it is scalable and automatic, therefore it can be applied to a large samples of data set with reasonable resources. In essence, scalability is one of the major concerns of malware industry following the time taken to respond to any new malware threat.

Arguably, this approach has some draw-backs. One of which is that it can only be applied to one category of malware namely Trojan Horse. Although the Trojan Horse is the most dangerous malware, with the modern techniques used by the authors in creating hybrid and multi-function malware, this approach cannot be used independently as a solution. There is a need to augment it with other methods to make it more effective.

The first approach that incorporates both static and dynamic analysis was proposed by [**Error! Reference source not found.**]. This approach combines features extracted using dynamic analysis and combines it with features obtained through static analysis to train and classify malware.

One major consideration about the practical applicability of this approach, it shows a decrease in accuracy on the most recent data sets used in the experiment. This indicates that the sustainability of this approach as a standard solution might not be effective as it requires constant periodic training of the classifier. It might suffer from problems faced by both dynamic and static analysis.

Another approach using CNN was proposed by Gibert and Javie [**Error! Reference source not found.**]. In their approach, they use a combination of the two different techniques used by Nataraj S. Karthikeyan [**Error! Reference source not found.**] as discussed above and Yoon Kim [**Error! Reference source not found.**]. Their result is not as good as the initial techniques they have adopted but, it is very close. Their method took less time than the original work while at the same time was demonstrated to be faster than the two techniques combined.

Comment [MT4]: Check

In their feature work, they have stated that despite that their method has slightly less accuracy then the adopted ones, certain modifications can be made to achieve higher results. One of their suggestions is that the result of the fully connected layers of the two independent CNNs from the approaches used should be merged together as one, and serve it to another separate CNN as an input. This has a potential of eliminating the separate problems encountered in the independent CNNs, and this will be adopted in this research.

Chapter 3 Methodology

3.1 Introduction

In this chapter, the data set used will be explained with the methods used to extract the features to suit the intended procedure in this research; in order to facilitate a better understanding of the procedures followed in carrying out the classification experiment. The architecture of the model together with the chosen algorithm will be discussed with some of its advantages, as well as the justification behind choosing the algorithm.

Finally, the chapter will conclude by going through the detailed procedure used in training the classifier and the evaluation method used for verifying the accuracy of the model produced.

3.2 Data Set

Kaggle website hosts machine learning competitions [**Error! Reference source not found.**] where any algorithm can compete to achieve a good performance. The data set used was posted by Microsoft as a machine learning challenge (Microsoft Malware Classification Challenge (BIG 2015) on the site [**Error! Reference source not found.**], containing about 500 gigabytes of data which was compressed to about 40 gigabytes of 7z format. Each file in the data set, both the training and test sets, was named with a 20-character unique hash value.

There are nine different classes of malware, and each malware sample belongs to one of these classes called families, with each class having a name. Each family is also numbered one through nine (1 to 9) as shown in the list, and can be referred to by both its name and number interchangeably. Below is the list of names and numbers of the families and classes.

Malware Families:

1. Ramnit
2. Lollipop
3. Kelihos_ver3
4. Vundo
5. Simda
6. Tracur

7. Kelihos_ver1
8. Obfuscator.ACY
9. Gatak

The entire data set is divided into two separate sets, the first set is a training set containing 10686 samples of both disassembly files (.asm) and their corresponding binary file (bytes) of the same malware. There is a comma separated values file (CSV) accompanying the training set that the list of all the names files in the set and indicates the class sample.

The other set is the test set which is 17.77 gigabytes of data, like the training set, each malware sample in the set is in pairs of the binary file (bytes) and the corresponding disassembled file (.asm), bearing the same name with the exception of the extensions. However, unlike the training set, in this set none of its sample is labelled with its class or family.

3.3 Feature Extraction

Although there is no universally accepted definition of feature extraction; put simply it is a method of filtering out common characteristics, patterns or behaviour from a sample after careful observation. The extracted features can then be used as a criterion for discriminating between samples of different class. This process is usually about making an intelligent guess of the characteristics that one deems to be important in setting a barrier between samples of different families.

In the research the chosen malware image visualisation technique was used for the classification process as discussed in [**Error! Reference source not found.**]. The approach uses the individual pixels of the image as the features for training the model. However, since the malware samples are not in image form, it has to be converted first. All the samples were read as a stream of binary, and then each one of them was written back as an image with the same name except that the file type and extension changed to either.jpg or.png.

3.4 Model Architecture

The architecture of the model is depicted in the Figure 3.1 and shows the individual phases of the model, their functions and how they are logically related.

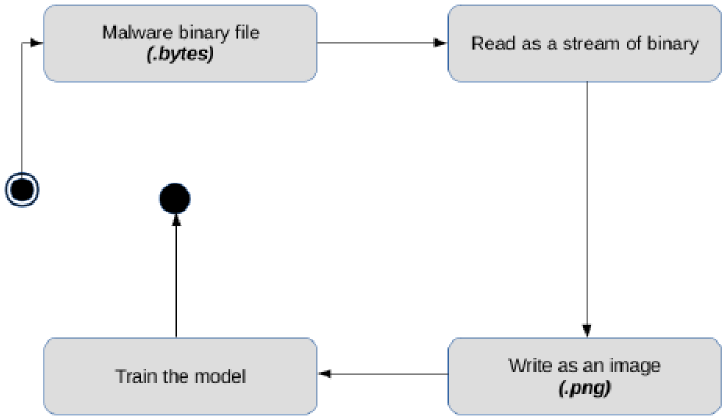

Figure 4 Model architecture

The first phase is where the malware binary files within the data set are read as binary [**Error! Reference source not found.**] stream and then converted to an image format. However, the files are of varying sizes, and consequently, the images generated from this file are also of varying sizes. In terms of dimension, regardless of their respective classes, so there is no uniformity. Some of the images have more than 2800 x 2800 pixels in the data set, which will be computationally expensive and will take time to train the model, there is also a high probability of inconsistency. Therefore, the images with large dimensions were scaled down to smaller dimensions while at the same time normalising the smaller images to the same size that is both practically and computationally acceptable. All the images have to be uniform for the model to be effective. All the images used in the research were re-sized to 32 x 32 pixels and a representation of the original image was obtained. Essentially, images are a multidimensional array of numbers [**Error! Reference source not found.**].

In the second phase, the individual pixels of the images were used as features, treating each image as one-dimensional array by joining all the rows of its element into a single row using reshape operation turning it to 1 x (N x N). This will not temper with the values of the individual pixels and can be restored to its original shape by applying the reshape operation, taking N to be the square root of the length of the array. The label of each training sample is added as the first element of the linear array obtained, and then all the images were packed together as N x L giant 2-dimensional array, where L is the total number of the training samples, which then used as the feature matrix for training the classifier.

Lastly, the feature matrix obtained from the previous phase was used as an input to the algorithm used in training the model. CNN was chosen as the algorithm for training the model primarily for two reasons. As discussed in the previous chapter, CNN has a good performance especially in image classification problems such as the converted samples into images done in this research. Secondly, an advantage of CNN is that it does not require manual feature extraction when dealing with images, providing it with labelled examples will be enough to get started, as it learns the features when it iterates over the images.

3.5 Implementation

CNN was implemented as has been discussed. Below the number of layers and the parameters of the configuration, used in the research, are described.

3.6 Training

Due to the constraints by computational resources, only a subset of the data set could be used. From the training set the first 30 samples were picked from each class, making a total sum of 270 samples. The table below provides a clear representation of the selection.

Comment [MT5]: Please include

3.7 Testing

The testing of the model was done through cross-validation, with a total of 45 samples, 5 from each class. The reason for using cross-validation is that it tests the model as the training is still on going, and provides feedback on the progression of the training. Refer to the tabular form, the number of samples used for the testing.

FAMILY	TRAINING	TESTING
Ramnit	30	5
Lollipop	30	5
Kelihos_ver3	30	5
Vundo	30	5
Simda	30	5

Tracur	30	5
Kelihos_ver1	30	5
Obfuscator.ACY	30	5
Gatak	30	5
Total	**270**	**45**

Table 1 Number of malwares use for training and testing

3.7.1 Convolution Layers

ConvLayer1, ConvLayer2, ConvLayer3

kernel = (5,5)

padding = (2,2)

stride = (1,1)

ConvLayer1, ConvLayer2

filter = 32

ConvLayer2, ConvLayer3

filter_init = 0.01

ConvLayer1

filter_init = 0.0001

ConvLayer3

filter = 64

3.7.2 Pooling Layers

PoolLayer1, PoolLayer2, PoolLayer3

kernel = (3, 3)

padding = (2, 2)

stride = (2, 2)

3.7.3 Normal Layers

NormLayer1, NormLayer2

kernel = 3

scale = $5e{-}5$

power = 0.75

3.7.4 Other Configurations

Total number of iterations 40000.

Batch size of 200

3.8 Summary

In this chapter, the method, procedure and the implementation details of the model were discussed. Although, the number of sample files used seems to be little, compared to the total number of samples in the data set, the model produced was found to be largely accurate in classifying the malwares. The method can also be applied to samples larger than the data set itself without requiring much modifications.

It is also worth mentioning that, considering the size of the data set, training the model takes time. It also requires relatively high processing power and consumes system memory as well as extra storage space, in addition to the space required to store the data set itself. Due to this reason, only a subset of the data set, specifically 30 samples, were used from each class for the training, making a sum of 270 samples, and 45 samples; 5 from each class were used for testing the model. Even with that number of images, it took roughly 22 hours to train and evaluate the model.

Chapter 4 Result

4.1 Introduction

This chapter presents the results obtained from the outcome of the research, and the observations made on these outcomes. Mainly, the focus will be on objective value, accuracy, number of samples, batch size and the number of interactions performed on each batch. Then these values will be analysed to see how they relate to each other.

4.1.1 Objective Value

Looking at the objective value, it can be seen that starting at a random value of 2.5 it keeps decreasing as the training progresses. This is due to the adjustment in the weights and bias by the loss function during back propagation. This means that the model is learning with each new fix iteration over the sample batches.

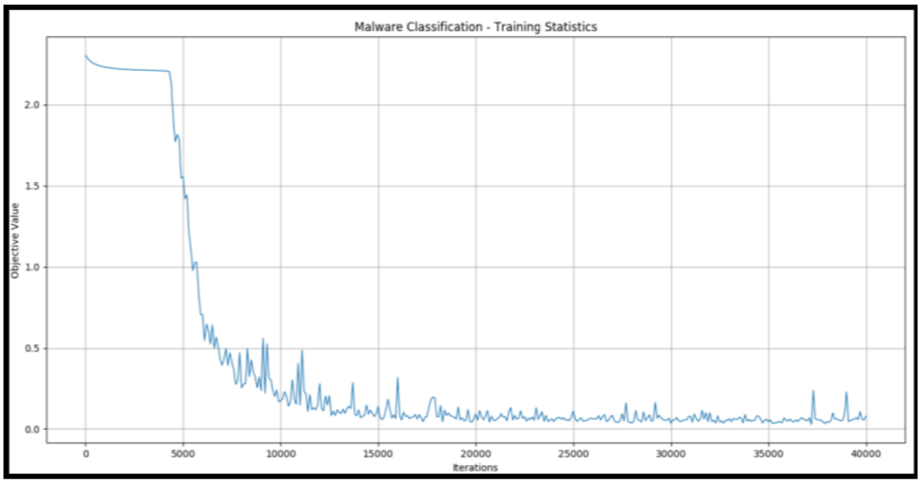

Figure 5 Objective value graph

4.1.2 Accuracy

The graph below is the graph of accuracy against the number of iterations. It provides a visual evidence of the different changes of accuracy at certain constant interval of the iterations.

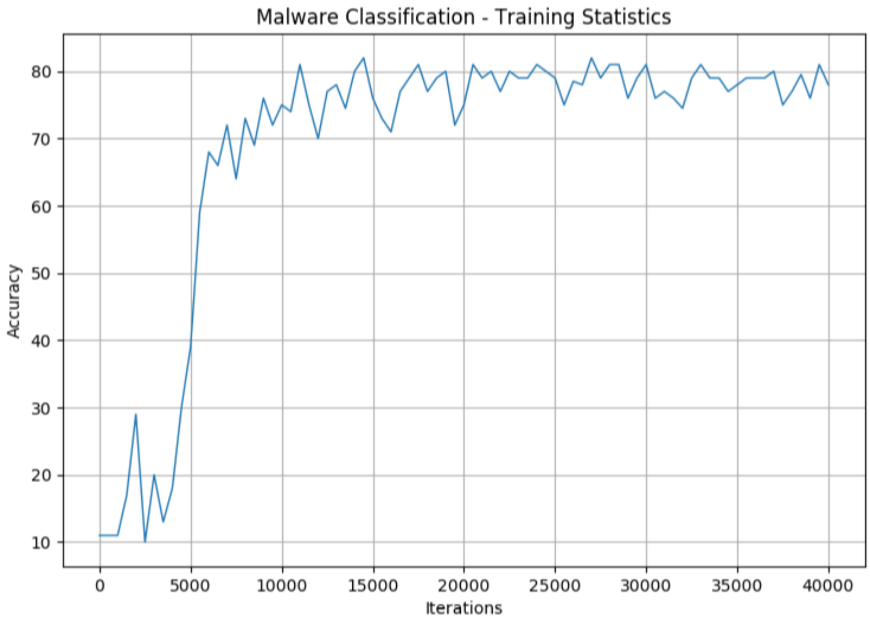

Figure 6 Accuracy graph

4.1.3 Summary

From the graphs above it should be observed that the prediction accuracy of the model is increasing with the increase in the number of iterations, but stops increasing and starts fluctuating within a given range. Although, the number of iterations keeps increasing, the accuracy stops increasing, this is an indication that a threshold is reached where there will be no further gain in accuracy, no matter the number of iterations. It should also be noted that as the objective value decreases, the accuracy increases. The accuracy varies directly with number of iterations. This shows a strong inverse relationship between the two values.

Chapter 5 Conclusion

5.1 Introduction

A convolutional neural network (CNN) classifier was trained in the research to classify malware samples using images obtained from the malware binary into its respective family with an accuracy of 78%. An understanding of Julia, which is a new generation computer language for scientific and machine learning computation, was achieved.

5.2 Challenges

The language used (Julia) is still under active development, and as such, there were issues with the new features of the language. Some of the script that was written in one version often broke after an upgrade or an update because of the deprecation of packages by the developers.

5.3 Recommendations

Due to the large size of samples, the research used a fraction of the whole data set available to train the model. This is not enough to rely on the model for practical use. In order to achieve that, the whole data set should be utilised for training the model. Although this will require high computing power, storage space and consume time, the language use (Julia) has a very good parallelism and distribution capability and it can be used on HPCs to speed up the training.

It will be useful if multiple techniques could be employed alongside binary image classification, which will hopefully improve the accuracy and performance of the model.

Bibliography

Comment [MT6]: Check all references

Ahmadi, M., Ulyanov, D., Semenov, S., Trofimov, M., and Giacinto, G. (2015). Novel Feature Extraction, Selection and Fusion for Effective Malware Family Classification.

Annachhatre, C., Austin, T. H., and Stamp, M. (2015). Hidden Markov Models for Malware Classification. *Journal of Computer Virology and Hacking Techniques*, 11(2):59–73.

Annachhatre, C., Austin, T. H., Stamp, M., Vigna, G., Jonsson, E., Kruegel, C., Rieck, K., Holz, T., Willems, C., Islam, R., Tian, R., Batten, L. M., Versteeg, S. C., Gibert, D., Bejar, J., Bailey, M., Oberheide, J., Andersen, J., Mao, Z. M., Jahanian, F., Nazario, J., Nataraj, L., Yegneswaran, V., Porras, P., Zhang, J.,, J. B.,, Y. M. Y. Y.,, S. M.,, Y. M. Y. Y., Tian, R., Batten, L. M., Versteeg, S. C., Dahl, G. E., Stokes, J. W., Deng, L., Yu, D., Nataraj, L., Karthikeyan, S., Jacob, G., Manjunath, B. S., Gandotra, E., Bansal, D., Sofat, S., Ahmadi, M., Giacinto, G., Zhong, Y., Yamaki, H., Takakura, H., Hutchison, D., Mitchell, J. C., Rieck, K., Trinius, P., Willems, C., Holz, T., Kinable, J., Kostakis, O., Versteeg, S. C., Hutchison, D., Mitchell, J. C., Islam, R., Tian, R., Batten, L. M., Versteeg, S. C., Annachhatre, C., Rieck, K., Holz, T., Willems, C., Patrick, D., Bailey, M., Oberheide, J., Andersen, J., Mao, Z. M., Jahanian, F., Nazario, J., Nataraj, L., Yegneswaran, V., Porras, P., Gibert, D., Ulyanov, D., Semenov, S., Trofimov, M., Giacinto, G., Zhong, Y., Yamaki, H., Takakura, H., Nataraj, L., Karthikeyan, S., Jacob, G., Manjunath, B. S., Gandotra, E., Bansal, D., Sofat, S., Kinable, J., Kostakis, O., Tian, R., Islam, R., Batten, L. M., Versteeg, S. C., Hutchison, D., Mitchell, J. C., Rieck, K., Trinius, P., Willems, C., Holz_aff2n3, T., Tian, R., Batten, L. M., Versteeg, S. C., Dahl, G. E., Stokes, J. W., Deng, L., Yu, D., Bai, Jinrong, Yanrong Yang, Shiguang Mu, Ma., Y., Kim, Y., Tabish, S. M., Shafiq, M. Z., and Farooq, M. (2013). Linux Malware Detection using Byte N-grams.

Bai, Jinrong, Y. Y. S. M. and Yu, M. (2013). Linux Malware Detection using Byte N-grams. *International Journal of Advancements in Computing Technology*, 5(7):288–295.

Bailey, M., Oberheide, J., Andersen, J., Mao, Z. M., Jahanian, F., and Nazario, J. (2007). Automated Classification and Analysis of Internet Malware. *Proceedings of the 10th International Conference on Recent Advances in Intrusion Detection*, pages 178–197.

Bishop, C. M. (2013). *Pattern Recognition and Machine Learning*, volume 53.

Comment [MT7]: Inomplete

Dahl, G. E., Stokes, J. W., Deng, L., and Yu, D. (2013). Large-scale Malware Classification Using Random Projections and Neural Networks. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3422–3426.

Gandotra, E., Bansal, D., and Sofat, S. (2014). Malware Analysis and Classification: A Survey. *Journal of Information Security*, 05(02):56–64.

Gibert, D. and Bejar, J. (2016). Convolutional Neural Networks for Malware Classification.

(October).

Hutchison, D. and Mitchell, J. C. (2013). *Detection of Intrusions and Malware, and Vulnerability Assessment*, volume 7591.

Islam, R., Tian, R., Batten, L. M., and Versteeg, S. (2013). Classification of Malware Based on Integrated Static and Dynamic Features. *Journal of Network and Computer Applications*, 36(2):646–656.

Kim, Y. (2011). Convolutional Neural Networks for Sentence Classification.

Microsoft (2015). Kaggle-Microsoft Malware Classification Challenge.

Murphy, K. P. (2012). A Probabilistic Perspective.

Nataraj, L., Karthikeyan, S., Jacob, G., and Manjunath, B. S. (2011a). Malware Images: Visualization and Automatic Classification. *Proceedings of the 8th International Symposium on Visualization for Cyber Security*, page 4.

Nataraj, L., Yegneswaran, V., Porras, P., and Zhang, J. (2011b). A Comparative Assessment of Malware Classification Using Binary Texture Analysis and Dynamic Analysis. *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence*, pages 21–30.

Oliva, A. and Torralba, A. (2001). Modeling the Shape Of The Scene: A Holistic Representation Of The Spatial Envelope. *International Journal of Computer Vision*, 42(3):145–175.

Rieck, K., Holz, T., and Willems, C. (2008). Learning and Classification of. *Dimva*, 5137:108–125.

Rieck, K., Trinius, P., Willems, C., and Holz_aff2n3, T. (2011). Automatic Analysis of Malware Behavior Using Machine Learning. *Journal of Computer Security*, 19(4):639–668.

Tabish, S. M., Shafiq, M. Z., and Farooq, M. Malware Detection using Statistical Analysis of Byte-Level File Content Categories and Subject Descriptors.

Tian, R., Batten, L. M., and Versteeg, S. C. (2008). Function Length As A Tool For Malware Classification. *3rd International Conference on Malicious and Unwanted Software, Malware2008*, (March):69–76.

Tian, R., Islam, R., Batten, L., and Versteeg, S. (2010). Differentiating Malware From Cleanware Using Behavioural Analysis. *Proceedings of the 5th IEEE International Conference on Malicious and Unwanted Software, Malware 2010*, pages 23–30.

Comment [MT8]: Incomplete

Comment [MT9]: Incomplete

Comment [MT10]: Incomplete

Comment [MT11]: Date

Vigna, G., Jonsson, E., and Kruegel, C. (2003). *Recent Advances in Intrusion Detection*, Volume 1907.

Zhong, Y., Yamaki, H., and Takakura, H. (2012). A Malware Classification Method Based on Similarity of Function Structure. *Applications and the Internet (SAINT), 2012 IEEE/IPSJ 12th International Symposium on*, pages 256–261.