Detecting Malicious Content from Extracted API Call Sequence by Applying Deep Learning and Machine Learning Algorithm

MSc Internship
MSc in Cyber

Aleena Gerard
Student ID: 18211593

School of Computing
National College of Ireland

Supervisor: Niall Heffernan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Aleena Gerard		
Student ID:	18211593		
Programme:	MSc in Cyber Security	**Year:**	2019-2020
Module:	Internship		
Supervisor:	Niall Heffernan		
Submission Due Date:	17/08/2020		
Project Title:	Detecting Malicious Content from Extracted API Call Sequence By Applying Deep Learning And Machine Learning Algorithm		
Word Count:	**Count** 6530	Page 20	

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

I agree to an electronic copy of my thesis being made publicly available on NORMA the National College of Ireland's Institutional Repository for consultation.

Signature: Aleena Gerard

Date: 17/08/2020

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

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

Detecting Malicious Content from Extracted API Call Sequence by Applying Deep Learning and Machine Learning Algorithm

Aleena Gerard
x18211593

Abstract

Nowadays, the growth of malware is increasing exponentially in variance and numbers parallelly with the expansion of digital world. In cybersecurity field malware has become the major issue and many attempts of work has been contributed to this field. Machine learning algorithms has been used for the detection of malware as it can identify obscure patterns in big datasets. Deep learning algorithms has potential varied layers which overcomes the limitations of machine learning algorithms as it provides high accuracy in classification in various domains. For the detection, API call sequences provides information about the behavioural attributes in a program. The goal of this research is to classify and detect malware from a high dimensional API call sequences dataset using deep learning algorithms like CNN, RNN and LSTM. The performance of these algorithms is being compared with traditional machine learning algorithms like LR, LDA, KNN, DT, NB with the same high dimensional dataset. This implementation and comparison study results in 98 percent of accuracy for both deep learning and machine learning algorithm.

1 Introduction

Malware detection is a crucial step towards cyber-attacks as it thrives through the vulnerabilities or any mistakes by the user in a system. Malware authors are trying find to various methods spread malwares and hide them from being detected. Their main goal is to make money and gain access to data with least per cent work. In 2020, in around 60% of the organizations the malware was spread through employees. Around 50% of the business PCs has been infected and re infected the same year. Also, ransomware attacks have caused business disruption to 51% of the organizations in 2020, and due to cyber security teams, that are understaffed[1]. In 2019 around 439000 new variants of malware have been detected.

The two main approaches in detecting malware is static malware analysis and dynamic malware analysis. In static analysis we examine the binary code, analyzing the paths of execution and finding the malicious code without performing the execution. Static analysis is vulnerable to code obfuscation because we even can perform the analysis by comparing extracted features with formerly identified malware signatures or features. The signature-based methods are reliably fast and no specified infrastructure for the analysis and collection of data. The limitations in the static analysis are resolved by dynamic malware analysis or also known as behavioral analysis. This method prevents obfuscation as it is based on behavioral data like system calls or API calls. In dynamic analysis there are two main approaches, API call analysis and Control flow analysis[1]. In both the approaches detection

[1] https://www.comparitech.com/antivirus/malware-statistics-facts/

is performed by analyzing the similarity among the familiar and unfamiliar ones. A sandbox environment is required to execute the program for collecting the data in dynamic analysis. After the data collection it is fed for behavioral based detection algorithm[2].

Deep learning algorithms has shown tremendous success in different domains like, speech recognition, image classification, natural language processing. These algorithms have also shown the capability of work in classification and malware detection by the usage of static and dynamic malware analysis[3]. Deep learning helps in the feature extraction from data at a low level with high abstraction. In this research we are using deep learning algorithms, CNN (Convolutional Neural Network), LSTM (Long sort term memory) and RNN (Recurrent Neural Network) on API call sequences. We have tried to classify the API call sequences among malware and benign. A study has been done by applying deep learning algorithms and machine learning algorithms for classifying API call sequences to compare the performance. The API call sequence dataset used for the research is from ieee-dataport.org. Various analysis method and models have been studied in the literature review portion, followed by research methodology, design specification, implementation of the project and evaluating the results obtained from the research. The last section highlights the discussion, conclusion and the scope of future work.

1.1 Research Objective:

.

"Detecting Malicious Content from Extracted API Call Sequence By Applying Deep Learning And Machine Learning Algorithm."

We have also performed a comparison analysis between the performance of deep learning and traditional machine leaning models implemented on high dimensional data.

2 Related Work

There have been many approaches made for malware detection in following years. Signature based detection were present in the earlier days by analyzing the binary code by not executing the code. In dynamic analysis, we run the malware samples and it is been observed that no data is being spread to other systems and removing the infection. Malware detection using machine learning algorithms and deep learning algorithms has shown better results and accuracy in detection and classification of malware.

2.1 Static Analysis

According to S Singla, E Gandontra, D Bansal, S Sofat, explains a simple technique has been presented based on static features which are extracted from windows PE files. The PE files are taken from the payload and header of the malware which includes Opcode frequency and function call frequency. The experiment results from the approach shows that the function call frequency is more robust than the Opcode frequence[4]. Extraction PE files is very important as HD Pham, TD Le, Tn Vu points out the scalability issues related to methods which uses more time to train or imbalanced datasets, and they proposed a method to PE analysis and gradient boosting tree algorithm by reducing the time for training[5]. ANASTASIA a system made by H Fereidooni, M cooni, D Yao, A Sperduti in android applications for malware detection by analyzing applications statically. They have tried to extract as many features possible and trained them with algorithms to identify the most performant one. To achieve maximum detection, they have used grid analysis with cross validation[6]. Apart from the specific features Li Chen obtains knowledge from objects and natural images to focus on the domain of static malware analysis which results in the acceleration in training time. They have demonstrated the results by three experiments explains the accuracy, true positive false positive and F1 score[7].

2.2 Dynamic Analysis

Y Ki, E Kim, HK Kim explains the malware detection using dynamic analysis from the extracted API call sequences. The malware authors are using the obfuscation techniques very much which in turns it very important to analyze the behavior of the malware. They have applied DNA sequence alignment algorithms on the API call sequences which have been extracted [1]. The malware could be harmful for the system, and in the traditional way methods has inaccurate detection and these methods takes a lot of time. So, M Asha and K Marimuthu has built a system where malware is detected in an Application Programmable interface which in turns classifies all the types of malwares as Trojans, worms, virus. For the pattern matching process, Rete algorithm has been used and MDBNS(Multi-Dimensional Naive Bayes Classification) has been used for the classification of malware that appeared in API call sequences[8]. GG Sundarkumar, I Nwogu V Ravi has proposed a new model were extraction of feature is done by text mining, and for feature selection LDA(Latent Dirchlet Allocation). In this method they have found SVM and DT to be not different significantly[9].

Stack Autoencoders (SAEs) model can be used for intelligent malware detection has been proposed by W Hardy, L Chen, S Hou, Y Ye. They have applied on the PAI call sequences which has been extracted from the PE files and this detection is having 2 phases, supervised backpropagation and unsupervised pretraining[10]. To detect the dynamic signature that a malware tries to hide, C Fan, H Hsiao tried hooking techniques by making a taring program. Later to identify the malware they compared the benign and malicious programs using data mining techniques.[11]

For the malware detection in the android system, S Hou, A saas has proposed a dynamic analysis in which they developed a malware detection system name DriodDelver which uses API call sequences that taken from the smali code by applying DBN(Deep Belief Network) by managing the inherent relationship between the API calls. Also, on android malware detection we can proceed analysis by obtaining data flow related API features and enhance K- nearest neighbor classification model. The API list which are data flow related are optimized by KNN, LR and BN machine learning models. S wu, P wang had used to enhance the sensitivity in data transmission analysis [12].

2.3 Machine Learning Algorithms

A two-stage model which uses LSTM model to learn features and then it is used in second classifier as an input. R Agrawal, J Stockes, has extended this model by including important API call parameter by using LSTM model. They have explored many parameters and to learn along with them they have derived the representations. They are also using event sequences along with the parameter and arrange by giving importance to their presence. By using LSTM model by X ma, S guo explained a model by analyzing the local maliciousness of the malware. Based on API fragments they had implemented a anti interference detection framework by training and validating the samples which are interference handled. By doing this they have proved that this method can prevent interference[2].

To increase the accuracy and efficiency in malware detection among large volumes and different types of malware C Chen, S Wang, Gu Lai uses CNN (Convolutional Neural Network). Also shows the efficiency of the method in recognizing malicious and benign codes and malware which is hiding inside the benign codes[13]. A Oliveria, R Sassi has proposed a method using DGCNN (Deep Graph Convolutional Neural Networks) in which from API call sequences they extracted behavioral graphs and used them as inputs for DGCNN. Then it is given to a fully connected layer and then it is pass on to sigmoid layer binary classification [3]. Whereas, M yeo, Y koo, T hwang used CNN with MLP, SVM and RF with five-fold cross validation applied on 35 features that are taken from packet flow data rather than from protocols or port numbers[14].

Recurrent neural network is a type of deep learning approach which are excellent in sequencing data like speech and text. The traditional approaches failed to exploit the multi label dependency of an image. Researchers J wang, T Yang uses RNN to resolve this multi-label image problem. It can give attention to various regions of images when different kinds of labels are predicted. RNN can be used to resolve different classification problems[15].According to R Wirth considering data mining a difficult process which needs proper methodology and tools where it can manage and understand the communication in the complex process. The researchers have proposed CRISP-DM to be effective for documenting, planning and interaction and better for repeatable process and huge projects were large number of people are involved. C Yin, Y Zhu has proposed that using a deep learning approach like RNN can be used to analyze the performance of model among the multiclass classification and binary classification. And the results show the high accuracy in modelling and the performance showed the superiority in binary classification. [16]

3 Research Methodology

In this research we propose a model to detect the malware from API call sequences and applying deep learning algorithms for the classification and detection of malware. API call sequences will provide details about the files which will help us to detect behavior of a malware. The main aim of the project is to build a new malware detection system using deep learning algorithm and enhance the accuracy to that. We are using CRISP-DM (Cross Industry Standard Process of Data Mining) approach is applied.

3.1 Crisp DM:

The Crisp-DM gives us a perspective of the planning a project in data mining. It stands for cross industry process of data mining. There are six phases in a data mining life cycle according to CRISP DM. In business understanding, we analyze the goals and requirements in a business perspective. Data understanding is performed by having a better understanding of data by identifying the quality problem, find out the insights into the data or explore any subsets in the data. The phase in which we cover all the necessary steps to build a final dataset is called data preparation. In modelling we decide the modelling techniques to be used and applied. In evaluation phase we decide the model which is accurate and can discover the unseen data and can find out any of the business issues. In the deployment phase, we will deploy the model in an operating system to analyze any unseen data. We analyze the unseen data we have achieved and discover a new mechanism to represent as a new solution for the problems in a business perspective[17].

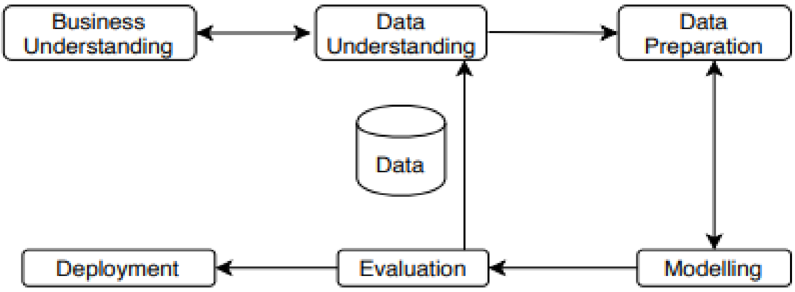


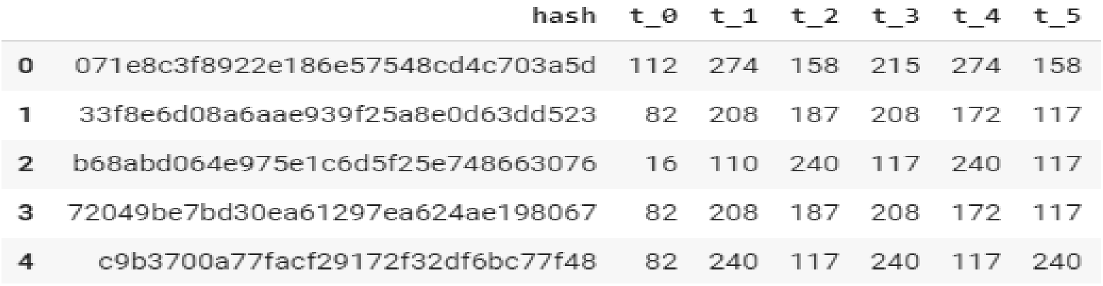
Figure 1 : Crisp DM

3.1.1 Business understanding

The aim of this step is to analyze business perspective behind the purpose of the research project and the factors which are influenced from the output of the project. The purpose of the research project is to detect malware from a high dimensional malicious dataset by using deep learning algorithms. This will help the users to detect malwares by reducing the high risk of cyber-attacks due to malwares. The method shows better accuracy than other methods used previously by detecting even the hidden malwares. Many traditional machine learning algorithms has been used before for the detection of malware, here we have compared results with traditional machine learning algorithms and deep learning has shown better accuracy. This research study has optimal addition to the detection of malware will be useful for preventing attacks.

3.1.2 Data Acquisition

In this research we are using dataset, malware analysis datasets: API all sequences from ieee-dataport.org. The malware dataset is having 42,797 malware API call sequences and 1,079 good ware API call sequences. These are extracted from calls elements form the cuckoo sandbox and every API call sequence constitute first 100 non repeated API calls consecutively. We Have downloaded the dataset using Kaggle API. The dataset is having 100 columns.[2]



	hash	t_0	t_1	t_2	t_3	t_4	t_5
0	071e8c3f8922e186e57548cd4c703a5d	112	274	158	215	274	158
1	33f8e6d08a6aae939f25a8e0d63dd523	82	208	187	208	172	117
2	b68abd064e975e1c6d5f25e748663076	16	110	240	117	240	117
3	72049be7bd30ea61297ea624ae198067	82	208	187	208	172	117
4	c9b3700a77facf29172f32df6bc77f48	82	240	117	240	117	240

Figure 2 Five rows of data

[2] https://ieee-dataport.org/open-access/malware-analysis-datasets-api-call-sequences

3.1.3 Data preparation

Nowadays a large amount of data is created every day in different forms. The raw data we obtain are in unstructured which contain many missing values, outliners etc. So it is important that the structured data should be fed to machine learning to understand. To achieve this, we perform numerous data mining tasks into the data before fed it into machine learning algorithms.

Quantity	Predictors	Categorical Variable	Predicted Quantity
43876	t0, t1….t100	Integer 0 : Goodware Integer 1: Malware	0s - 1070 1s - 42797

Figure 3 Data Contents

Data Exploration (EDA): The understanding of the data is done in this step. To identify the chances of any variance or bias we study the structure of data. Through data visualization and statistics, we analyze the distribution of data, outliners, data summary.

MissingValues: One of the major concerns in the quality of data is missing values. Depending on the nature of the characteristics the methods used can be replaced using mean, mode, median. We must remove the data rows and columns if the risk of data loss is high like the missing values of students, grade or ethnicity for an example.

Outliners: Using the boxplots we are we can identify the anomaly of the dataset. The chance of happening is less as due to the large data capture. The large amount data transaction can also result in fraud involvement. So, the outliners result in biased or skewed output and makes it important to abolish outliners from the data. Some applications appeal analysis of data individually.

Data Conversion: The requirements data type is different for many machine learning algorithms, but the dataset contains different kinds of attributes. So, there is a problem in handling these categorical variables, which are of two types, ordinal and nominal. Dummy variables are present in nominal variables. In ordinal variables a new column will be generated corresponding to the numbers in orders.

Data Scaling: To avoid comparison and weightage, it is suggested to have all variables in a same level. For example, if we consider income and age, the later can be dominated. Through standardization and normalization, we can achieve all variables in the same scale.

3.1.4 Feature Selection

Feature selection is the process of minimizing the input variables when we are building a predictive model by identifying the most relevant variable in determining the target variable. It prevents the selection of irrelevant features from including in the model which will lead to predictions which are inaccurate. Methods like statistical-based feature selection is used for the evaluation of relation among target variable and input variable with the usage of statistics. And, the selection of input variable which has the better association with the target variable. So, we conduct many statistical tests for selecting the features which in training of the model to accomplish better predictions. The advantages of feature selection include, it addresses the issues like overfitting and improve the accuracy by predicting the better results. It also enhances the training time and minimize the complexity by making it easier to interpret[18]

The widely used feature selection methods are Filter Method, Wrapper Method, Embedded Method, Univariate Selection, Principle Component Analysis (PCA). Some of the methods are mentioned below.
Filter Method: Generally, filter methods are used as a preprocessing step and the feature selection is not depended on ant machine learning algorithms. The selection of features is based on scores it obtained from the statistical test which were related to the variable outcome. ANOVA, Chi-Square, Pearson's Correlation, LDA statistical methods are used for the selection of relevant features in relation with target variable[3].

Wrapper Method: We can implement wrapper method using 3 mechanisms like Forward Selection, Backward Elimination, Recursive feature selection. In forward selection, we perform iteration by starting with no feature and keep adding after each iteration. In backward feature selection, we will remove the less significant feature from each iteration for increasing the efficiency of the model. In recursive feature selection, by using an optimization algorithm we identify the best feature for training the model.

Principle Component Analysis (PCA): The process of obtaining the important variable from huge set of variables among a dataset. It minimizes the dimensionality by the implementation of linear algebra of the data.

3.1.5 K-fold Cross Validation

We generally evaluate accuracy of the model by obtaining information from different test set but this method is less reliable as the accuracy among the different test set can be different. CV (K-fold Cross Validation) is the process of division of data into different folds and these folds will be having the training set for training the model and a test set for the evaluation of the model. The results of the method are not biased when compared to other methods. Upon diving the dataset into folds which will be in equally size and will treat the first fold as validation set. We first shuffle the dataset randomly followed by splitting the dataset. In CV dataset is divided into K number of folds, in which every fold is also used as a testing set such as k=10 will be 10-fold cross validation. We will hold up the test dataset from the train dataset and fit a model on the training dataset and conduct the evaluation on the test set.

3.1.6 Modelling

A machine learning approach which are mainly depended on neural network architecture with various layers of processing units. And it applies transformations like non-linear and linear to the input data. The neural network architecture can be applied to wide range of data lie text, numbers, image, audio, and the combination of these data. In the last decade deep learning has acquired an immense recognition in research areas and industry. The results of these methods have shown good and promising results compared to other machine learning algorithms. In this research we have used deep learning algorithms like CNN (Convolutional Neural Network) algorithm, LSTM (Long-Term Short-Term Memory)

[3] https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/

and RNN (Recurrent Neural Network). Also, we have used traditional machine learning algorithms to compare the performance of the deep learning algorithms. [19]

3.1.7 Evaluation

This phase in the Crisp-Dm is focused on the result of the project. The accuracy of the deep learning algorithms we have used is very high. We have then calculated the performance of the deep learning model using traditional machine learning models. The accuracy has been calculated using AUC and ROC.

4 Design Specification

4.1 CNN

CNN is a deep neural network which are applied to analyze the visual imaginary to perform image classifications, image recognition, object detections. CNN will test and train the data which are going to pass through convolutional layers which have filters, pooling layers and fully connected layers and then apply a SoftMax function. It is a fully connected network where neurons in one layer will be connected to every other neuron in different layers. CNN can be used for differentiating the input given as image and also designate importance to objects in the image or to many other aspects as in biases and weights. CNN has the capability to understand the characteristics and filters.[20]

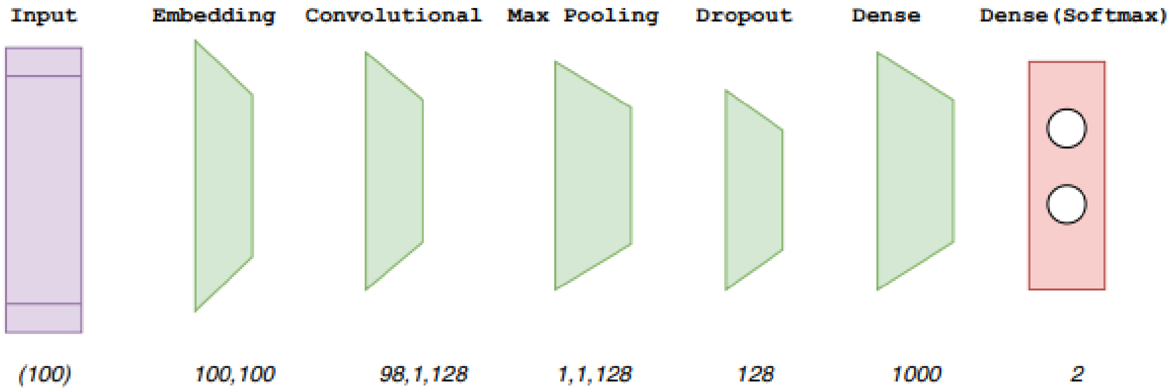


Figure 4 CNN Architecture

The CNN has three layers, convolutional layer, Pooling layer and Fully Connected. The architecture of the CNN is very important as the performance is depending on the architecture like the number of building block layers, are used and composed, and relation of building blocks specified. An order 3 tensor is taken as an input by CNN like if an image a taken as an input with W columns, H rows and color channels (R,G,B). CNN can handle inputs like higher order tensor. A series of processing is applied sequentially onto he input. Each processing step is called a layer like pooling layer, convolutional layer, loss layer, normalization layer, fully connected layer etc.

Embedding: The embedding layer is a flexible layer that can be initialized with different weights and embedding will be learned for every word that is present in the dataset. And it is the hidden layer that is present at first in a network.

Convolutional 2D network: In 2D convolutional layer the input of the process will be a three dimensional, which abstract features from the image we give us input by performing scanning with the help of filters. The 2D defines the movement of the filter in two dimensions.

Max Pooling: The process of merging is known as pooling. The selection process of maximum elements from area of map contains features which are covered by the filters. The result of the max poling layer will a feature map which contains the maximum features from the preceding map.

Flatten: In flatten layer, the input's spatial dimensions will be collapsed and change it into a channel dimension. Only sequence input will be supported by this layer. The output from the convolution layers will be flattened to create a long feature vector and this will be associated to the fully connected layer.

Dropout: Dropout is a method to tackle the problem of overfitting in case of using large neural network. This will drop units randomly during the training process from the network which avois the co adaptation of units. This will help regularization methods and performance will be improved.

Dense(Fully connected layer): The deep connected layer in neural network is the dense layer , where every neurons are connected to each other neurons in the network. It uses a linear operation where all nodes inputs and connected to output nodes by a weight.

SoftMax activation function: This function changes the k values vector which add up to 1. The values could be higher than one, positive, negative as input. The SoftMax function will change any of these input values between 1 and 0 and they are represented as probabilities.

We have applied traditional machine learning algorithms like LR, LDA, KNN, DT, NB on the dataset and calculated the accuracy of the model. We have performed this to calculate the performance of the deep learning algorithms applied to the data.

4.2 LSTM

LSTM stands for long short-term memory, and it introduced the memory into the neural networks. To avoid the long-term dependency problem, LSTM are designed especially. To know if there is a time lag and to find out the time duration, LSTM is applicable for process, classify and predict them. As LSTM can REMEMBER the values over the time intervals. It is recurrent neural type which can perform learning order dependencies in problems like sequence prediction. It eliminates the gradient problem. The structure of LSTM is like a chain, where the modules which are repeated has different structure.

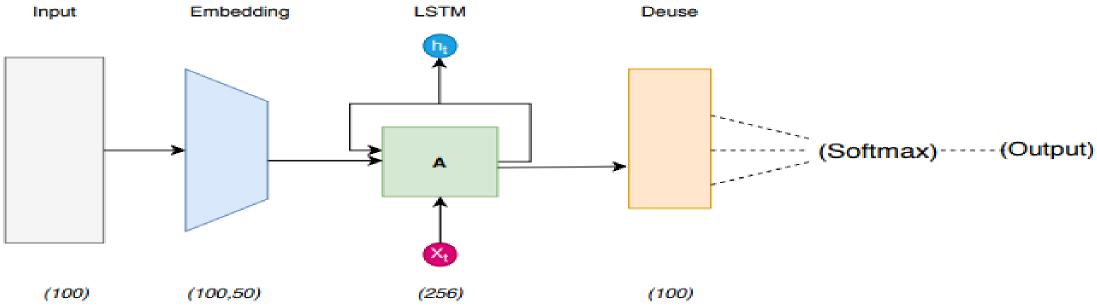


Figure 5 LSTM Architecture

The initial step in the LSTM process is to choose the information giving to the cell state. Forget gate layer, a sigmoid layer will be deciding on this. The second step in the process is to agree on what new information is going to cell state. For that another sigmoid layer named input gate layer will choose the values for updation. And a vector will be created by tanh layer which contains the candidate values and we will join them and construct the update. And in the last step we will choose the output where it is a filtered version which will be based on the cell state[21].

4.3 RNN

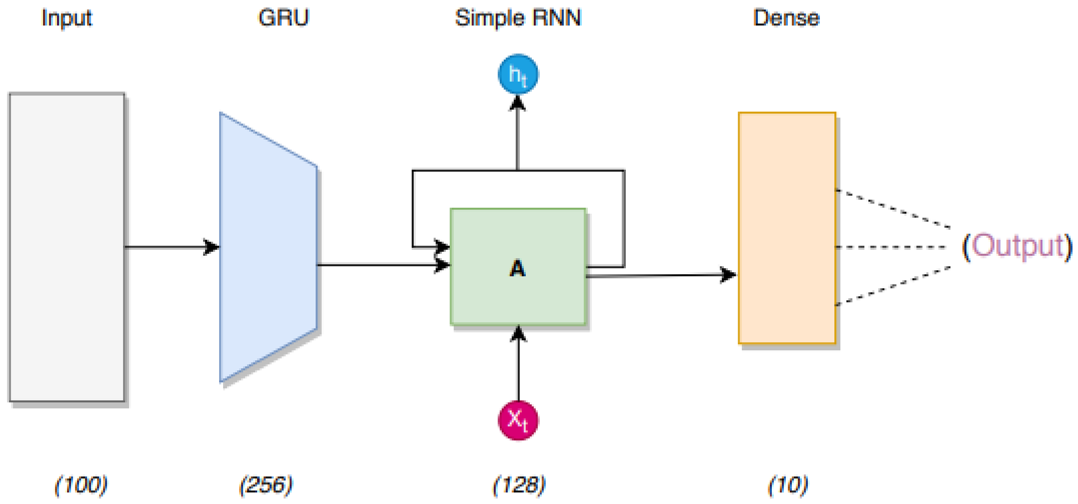


Figure 6 RNN Architecture

In RNN (Recurrent Neural Network) the output earlier step is given as an input to the present step. A memory is present in RNN which remembers every information which has been fed to it. Using the Hidden state, an important feature which remembers sequence data and there are various hidden layers are present. When input is received the first hidden layer will be activated and these activations will be later passed on t the next layer and the activations from that layer will be transferred to another layer to deliver output. Categorization of every hidden layer is done with biases and weights.
In this network there is a linear graph among the nodes artificial neural class of network which has a temporarily sequence. The dynamic behavior of the temporal exhibit is allowed by the artificial neural network class. RNN's internal memory is used to process the sequence inputs. By the introduction of the feedback loop RNN can perceive and memorizing which are related to the preceding step[22].

5 Implementation

The dataset named API Call Sequence is sourced from the Malware analysis dataset of IEEE DataPort. There are 100 attributes that labels the call to be Malware or Goodware resulting in a dichotomous output. Deep learning algorithms and traditional machine learning classification methodologies are implemented on the dataset to perform a comparative analysis with respect to model efficiency in addressing the high dimensional dataset. Deep learning algorithms have proved to outperform traditional machine learning algorithms in the regard. However, the appropriate selection of the machine learning model aided with data pre-processing results in enhanced model efficiency which is competent with the deep learning techniques.

The variables in the dataset is closely scrutinised through exploratory data analysis to detect the anomaly in the attributes. To assess the data quality missing values are checked for all the variables. Correlation matrix is computed taking the independent attributes to eliminate the highly correlated variables from the model which impacts the variance of the weights. For distributional understanding of the variables, statistical computations like skewness and kurtosis in performed.

Pre-processing of the data involved feature selection and splitting the dataset into training and validation subset of data. Widely used Filter method is implemented for feature selection. Here, the relevant attributes are selected for the model based on the scores obtained from the statistical hypothesis tests with respect to their association with the predicted variable. This is to mention that filter method is adopted for feature selection because correlation between the explanatory variable is taken care in the EDA phase of the analysis. Additionally, to prevent overestimation or underestimation of the models due to induced bias or variance K-fold validation is used to split the dataset in training and test datasets. This iterative method involves every data point to be a part of both training and test dataset.

The implemented deep learning techniques CNN, LSTM and RNN are evaluated through loss function and model accuracy. The neural networks are trained implementing the optimization process often referred as the gradient descent which demands for the loss function. The primary purpose of the loss function is to compute the model error. However, the maximum likelihood method facilitates in the selection of the right loss function depending on the assigned tasks like prediction or classification for the neural network. Here, in the research work binary classification is performed, that is depending upon the multiple attributes a call is classified as Malware or Goodware. In classification problems the output nodes necessarily result in probabilistic outcome of a class. Binary classification essentially has a single node in the output layer requiring an activation function to represent the output as a probabilistic value. Adam optimiser is used employed because of its enhanced efficiency, reduced implementation complexity and memory usage. Moreover, it can successfully address problem involving high dimension as illustrated by the used dataset in the research. The Adam optimization algorithm[4] can be explained as first order stochastic gradient descent depending on estimated moments of lower order. The loss function used to calculate the model error is the spare categorical cross-entropy which best fits for target variables that are integers.

The appropriate machine learning techniques like LDA, KNN, DecisionTree, SVM, LR and Naïve Bayes for binary classification task is implemented. For the model evaluation AUC and ROC dependent on the specificity and sensitivity is computed.

[4] https://arxiv.org/abs/1412.6980

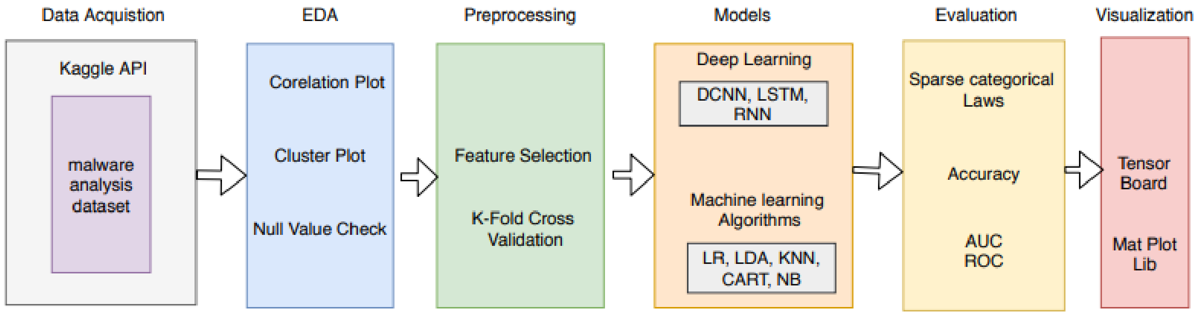
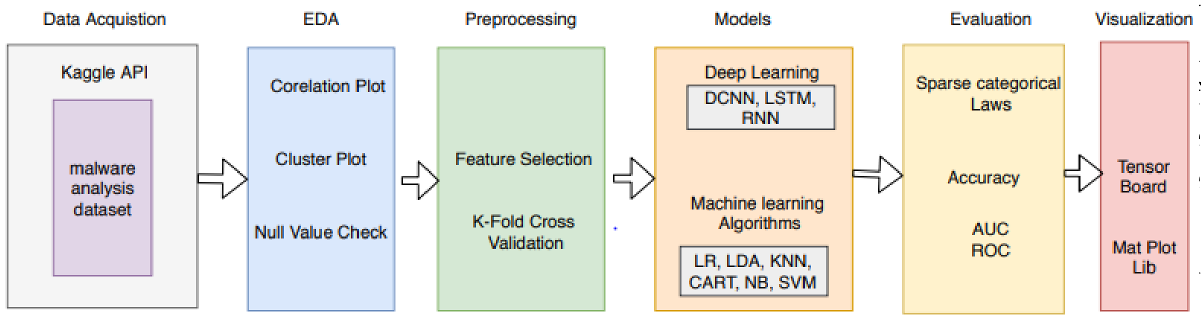


Figure 7 Flow Diagram



6 Evaluation

The experiments are broadly segmented into Deep learning algorithms and Machine learning algorithms. Deep learning algorithms like CNN, LSTM and RNN are implemented followed by implementation of the traditional machine learning techniques like LR, LDA, KNN, DT and NB. A comparative analysis is performed between the deep learning and machine learning methods through assessing the respective model accuracy paired with the loss value. Here, deep learning algorithm is expected to have enhanced accuracy considering the high data dimensionality.

6.1 Experiment 1

The section consists of implementation of three deep learning algorithms to perform the classification task (malware and goodware). The dataset is split into training and validation set in the ratio of 80% and 20% respectively. To access the classification efficiency of the model accuracy is calculated followed by graphical presentation with y-axis as the accuracy values and x-axis as the number of epochs trained. To draw a parity in performance of the algorithms when implemented on the training and validation dataset two graphs are merged. However, clear distinction is made through representing the validation and training accuracy curve with yellow and red lines, respectively. Both the set of data is run for equal number of epochs. Additionally, to validate the quality of the model train the loss function is plotted for both the dataset with the loss values on y-axis and number of epochs on x-axis.

[5] https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

6.1.1 Classifier CNN

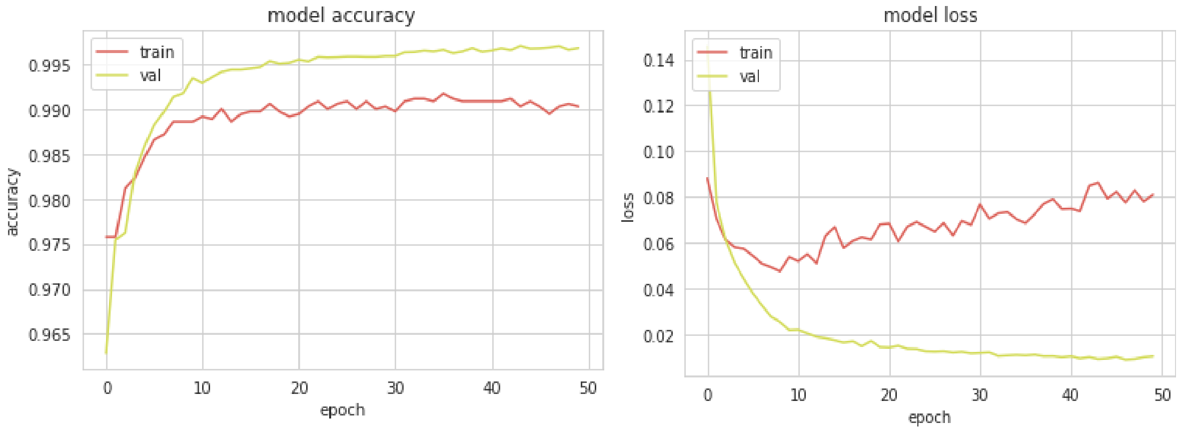


Figure 8 Accuracy Curve(Left) and Loss Function(Right)

Accuracy: Initially, model accuracy is high for both the training and validation dataset where the approximate value is 97 percent. The training and validation curve reaches the peak of almost 99 percent accuracy in the 5th epoch and then a stationary oscillatory curve is maintained, till the 50th epoch. There is a difference of 0.005 oscillatory distance between the two curves. The consistent accuracy value over the consecutive epochs signifies the high classification efficiency of the model.

Loss: Loss is calculated for train and validation. The loss value signifies how well a model performs in this two sets after every iteration of optimization. The loss value in the initial stages is very high but after the 5th epoch the loss value takes a steep fall. After the 10th epoch the loss value of the validation set further decreases at gentle rate and stabilises at a value less than 0.02. The training loss fluctuates between 0.06 and 0.08 after the 10th epoch and becomes stable after 45th epoch. The loss value is significantly low which implies very good optimisation of the model weights after every epoch.

6.1.2 Classifier LSTM

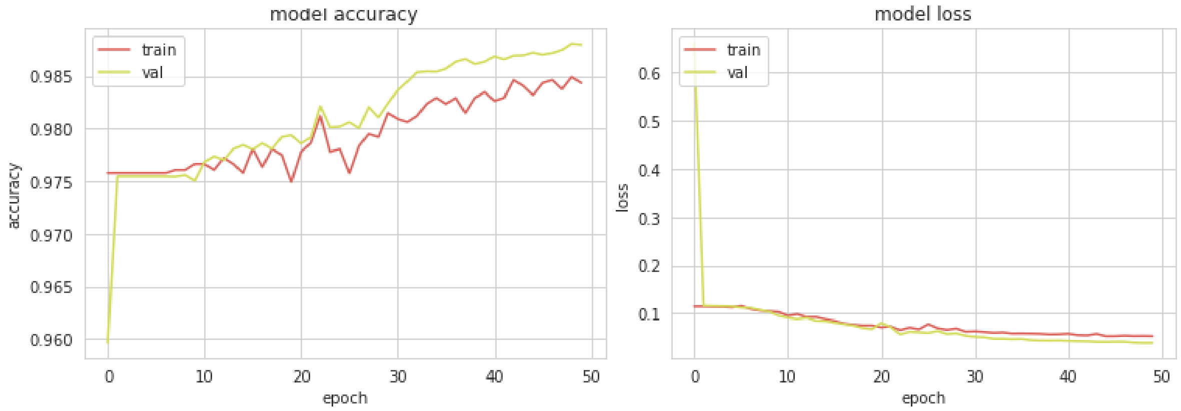


Figure 9 Accuracy Curve(Left) and Loss Function(Right)

Accuracy: For initial few training phase the accuracy curve is stationarity. However, the model yielded very high accuracy of approximately 97% at the first iteration. Further, training of the model resulted in gradually improved accuracy. The graph suggests that approximately 99% of the calls are correctly classified as malware or goodware by the model. The validation curve is in line with the training curve showing that the data point selected in the training dataset is a good representation of dataset.

Loss: In the first 5 epoch the graph illustrates the loss value for training and validation has almost reached the state of stability with minimal gap between the two curves. This is a representation of a good fit. However, further training of a good fit model often leads to model overfit. Here, the model is trained for significant number of epochs resulting in decreasing loss value signifying improvement in the classification model.

6.1.3 Classifier RNN

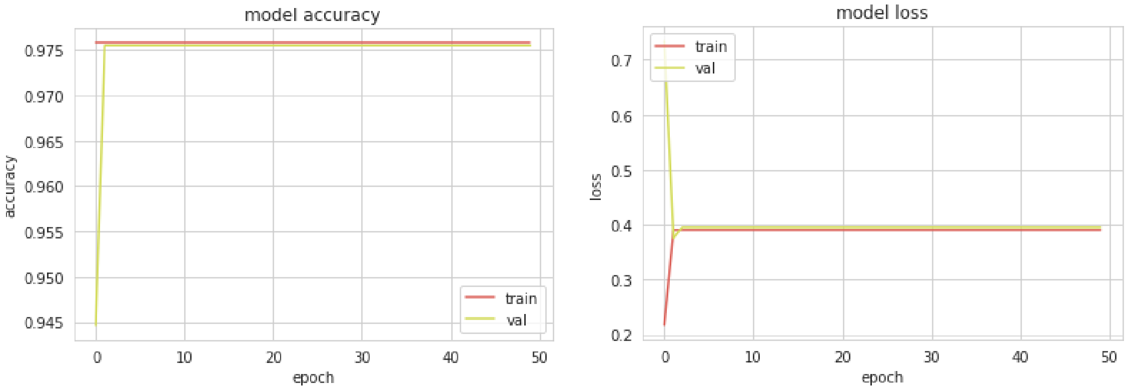


Figure 10 Accuracy Curve(Left) and Loss Function(Right)

Accuracy: The graph illustrates that the training curve and the validation curve has attained very high accuracy at the 1st epoch of approximately 97% which showed no modulation throughout the training phase. This explains that the optimum set of weights are obtained through high learning rate. Thus, no improvement in the classification task will be observed despite training the model for multiple epochs.

Loss: Loss value over multiple epochs explains the optimization quality of the model. Here, the stationary curve throughout the training period signifies that the weights cannot be further optimized with the utilized parameters of the neural network.

6.1.4 Comparison

To draw a comparison between the implemented deep learning with respect to their classification efficiency AUC-ROC curve has been plotted with sensitivity as y-axis and the specificity conjugate as the x-axis. Here, the AUC for all the implemented deep learning models has value greater than 0.5 illustrating high efficiency in performing the classification task. The relative high AUC value of approximately 0.87 for the DCNN signifies that there is 87 percent chance of accurately classifying the calls as malware or goodware. This can be inferred that DCNN has improved classification efficiency outperforming the other implemented deep learning algorithms

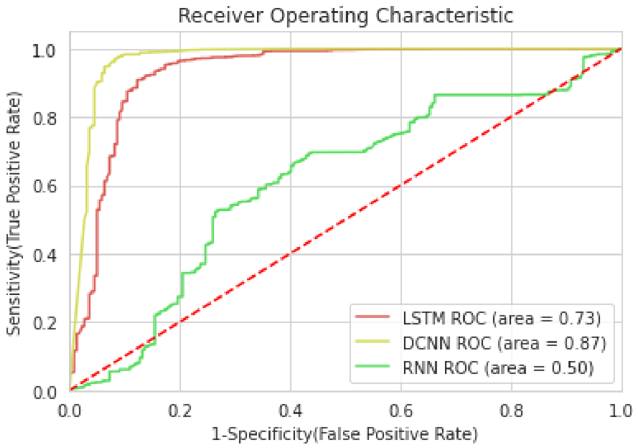


Figure 11 AUC – ROC Curve for Deep learning Models

6.2 Experiment 2

The experiment section involves implementation of the traditional machine learning algorithms like LR, LDA, KNN, Decision tree and Naïve Bayes to perform the classification task. Prior to model fit the dataset is divided into training and validation set using K-fold cross validation. Here, K-fold cross validation is used to train the model to draw a parity between the deep learning and machine learning algorithm since it is an iterative method to assess the model efficiency. The table demonstrates that most of the machine learning algorithms have maintained high accuracy. The Naïve Bayes have comparatively lower accuracy. Accuracy value yielded in each iteration for respective models are not diverse as the standard deviation of the accuracy for the corresponding models are significantly low. The adjacent boxplot illustrates that the performance of all the models are consistent. Outliers are detected in case of LR, LDA and DT, and is not present in case of LDA and NB. However, there are no significant high number of outliers. So, it can be fairly concluded that the k-fold validation brought the consistent accuracy at through the iterations.

Classifier	Accuracy (Mean)	Accuracy (SD)	AUC
LR	0.9829	0.0021	0.7100
LDA	0.9761	0.0041	0.7600
KNN	0.9856	0.0021	0.7100
DT	0.9835	0.0026	0.8300
NB	0.8953	0.0064	0.6900

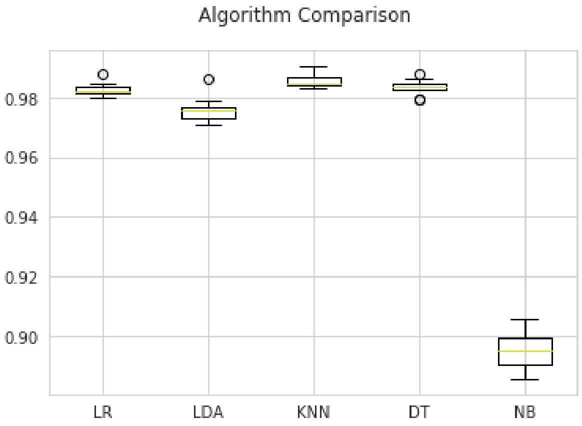


Figure 12 Descriptive Statistics Computed for the accuracy values(Left) Boxplot for accuracy values(Right)

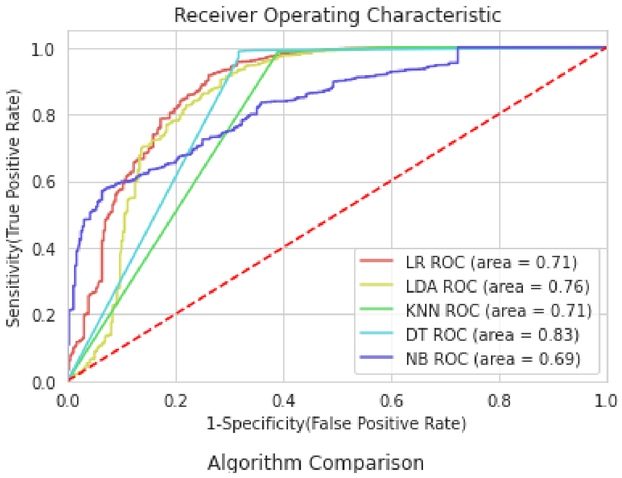


Figure 13 AUC – ROC Curve for Machine learning Models

To perform a comparative analysis between the implemented machine learning models AUC-ROC curve is plotted. There is a parity between the machine learning models in performing classification task with respect to the model accuracy as the AUC values for the models are significantly greater than 0.50 and are closely knit. However, the decision tree algorithm has relatively improved classification performance compared to the other implemented models.

6.3 Discussion

The accuracy reached for the deep learning and machine learning algorithm is relatively high at a very less epoch which might be the result of overfitting of the model. This shows that the deep learning algorithms and traditional machine learning algorithms have similar classification efficiency for high dimensional dataset. However, for the prevention of the overfitting or underfitting of the model we have considered various measures. Over fitting can be fairly avoided if the entire dataset can be used for the model training. dividing them into training and testing dataset, respectively. The dataset that has been used for the project consists of 100 attributes, which is a very high dimensional data. Thus, data pre-processing is performed prior to machine learning model fit. K-fold cross validation is used for diving the dataset in training and testing dataset, where batches are formed and each datapoint is utilized. Parameter selection has been done using filter. Also, apart from using the entire dataset, the other important solution that has been adapted in our project is the usage of optimizers. Adam Optimizer has been used for this purpose. Adam optimizer is a stochastic gradient descent method that is based on the adaptive learning rate optimization, where the advantages of learning of each parameter is done and is highly preferred as it signifies optimum learning at a faster rate.

7 Conclusion

The main of the project is detect malicious content from extracted API call sequences and applying deep learning and machine learning algorithm. We have combined and checked the performance among the three deep learning algorithms like CNN, RNN, LSTM. All of the deep learning algorithm has shown high accuracy, among them CNN has shown the highest accuracy of 87 percent. The dataset used for performing this is done on a high dimensional data with 100 attributes in it. For each deep learning algorithm, 50 epochs have been done. The implementation of traditional machine learning algorithms like LR, LDA, KNN, DT, NB to validate the performance against the deep learning algorithms. For traditional machine learning algorithms also, high dimensional data has been used, even though the machine learning algorithms are not well known for good performance for high dimensional data. All of the mentioned machine learning algorithms has acquired high accuracy. Among all of them NB nearly got less accuracy.

Appendix:

To visualize high dimensional data KNN clustering was performed on the dataset which resulted in six distinct clusters.

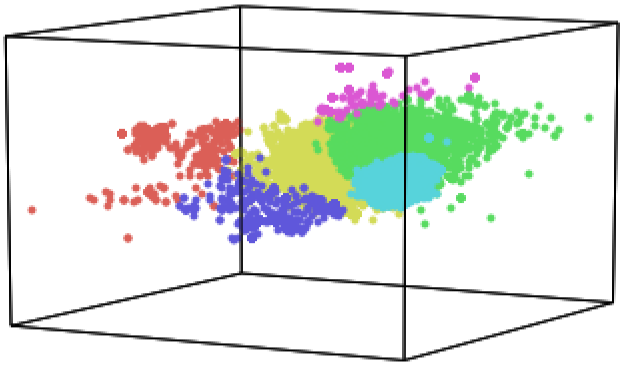


Figure 14 KNN Clusters

References

[1] Y. Ki, E. Kim, and H. K. Kim, “A novel approach to detect malware based on API call sequence analysis,” *Int. J. Distrib. Sens. Networks*, vol. 2015, 2015, doi: 10.1155/2015/659101.

[2] X. Ma, S. Guo, W. Bai, J. Chen, S. Xia, and Z. Pan, “An API Semantics-Aware Malware Detection Method Based on Deep Learning,” *Secur. Commun. Networks*, vol. 2019, 2019, doi: 10.1155/2019/1315047.

[3] A. Oliveira, U. N. De Julho, and U. N. De Julho, “Behavioral Malware Detection Using Deep Graph Convolutional Neural Networks,” pp. 1–17, 2019, doi: 10.36227/techrxiv.10043099.v.

[4] S. Singla, “A Novel Approach to Malware Detection using Static Classification,” *Int. J. Comput. Sci. Inf. Secur.*, vol. 13, no. 3, pp. 1–5, 2015.

[5] H. Pham, T. D. Le, and T. N. Vu, *Static PE Malware Detection Using Gradient*, no. January. Springer International Publishing, 2018.

[6] H. Fereidooni, M. Conti, D. Yao, and A. Sperduti, “ANASTASIA: ANdroid mAlware detection using STatic analySIs of applications,” *2016 8th IFIP Int. Conf. New Technol. Mobil. Secur. NTMS 2016*, 2016, doi: 10.1109/NTMS.2016.7792435.

[7] L. Chen, “Deep Transfer Learning for Static Malware Classification,” pp. 1–9, 2018, [Online]. Available: http://arxiv.org/abs/1812.07606.

[8] M. A. Jerlin and K. Marimuthu, “A New Malware Detection System Using Machine Learning Techniques for API Call Sequences,” *J. Appl. Secur. Res.*, vol. 13, no. 1, pp. 45–62, 2018, doi: 10.1080/19361610.2018.1387734.

[9] G. G. Sundarkumar, V. Ravi, I. Nwogu, and V. Govindaraju, “Malware detection via API calls, topic models and machine learning,” *IEEE Int. Conf. Autom. Sci. Eng.*, vol. 2015-Octob, pp. 1212–1217, 2015, doi: 10.1109/CoASE.2015.7294263.

[10] W. Hardy, L. Chen, S. Hou, Y. Ye, and X. Li, “DL4MD: A Deep Learning Framework for Intelligent Malware Detection,” *Proc. Int. Conf. Data Min.*, pp. 61–67, 2016, [Online]. Available: https://search.proquest.com/openview/a090ba95404b143e4bbfbb4e0b6bebab/1?pq-origsite=gscholar&cbl=1976357.

[11] C. I. Fan, H. W. Hsiao, C. H. Chou, and Y. F. Tseng, “Malware detection systems based on API log data mining,” *Proc. - Int. Comput. Softw. Appl. Conf.*, vol. 3, pp. 255–260, 2015, doi: 10.1109/COMPSAC.2015.241.

[12] S. Wu, P. Wang, X. Li, and Y. Zhang, “Effective detection of android malware based on the usage of data flow APIs and machine learning,” *Inf. Softw. Technol.*, vol. 75, pp. 17–25, 2016, doi: 10.1016/j.infsof.2016.03.004.

[13] C. M. Chen, S. H. Wang, D. W. Wen, G. H. Lai, and M. K. Sun, “Applying Convolutional Neural Network for Malware Detection,” *2019 IEEE 10th Int. Conf. Aware. Sci. Technol. iCAST 2019 - Proc.*, pp. 1–5, 2019, doi: 10.1109/ICAwST.2019.8923568.

[14] M. Yeo *et al.*, “Flow-based malware detection using convolutional neural network,” *Int. Conf. Inf. Netw.*, vol. 2018-Janua, pp. 910–913, 2018, doi: 10.1109/ICOIN.2018.8343255.

[15] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu, “CNN-RNN: A Unified Framework for Multi-label Image Classification,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-December, pp. 2285–2294, 2016, doi: 10.1109/CVPR.2016.251.

[16] R. Wirth, “CRISP-DM : Towards a Standard Process Model for Data Mining,” *Proc. Fourth Int. Conf. Pract. Appl. Knowl. Discov. Data Min.*, no. 24959, pp. 29–39, 2000, doi: 10.1.1.198.5133.

[17] V. Plotnikova, M. Dumas, F. P. Milani, and R. Kitt, “Towards a data mining methodology for the banking domain,” *CEUR Workshop Proc.*, vol. 2114, pp. 46–54, 2018.

[18] A. Purpura, C. Masiero, G. Silvello, and G. A. Susto, “Feature selection for emotion classification,” *CEUR Workshop Proc.*, vol. 2441, pp. 47–48, 2019.

[19] A. Hernández-Blanco, B. Herrera-Flores, D. Tomás, and B. Navarro-Colorado, “A Systematic Review of Deep Learning Approaches to Educational Data Mining,” *Complexity*, vol. 2019, 2019, doi: 10.1155/2019/1306039.

[20] S. Hershey *et al.*, “CNN architectures for large-scale audio classification,” *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, pp. 131–135, 2017, doi: 10.1109/ICASSP.2017.7952132.

[21] C. Zhou, C. Sun, Z. Liu, and F. C. M. Lau, “A C-LSTM Neural Network for Text Classification,” 2015, [Online]. Available: http://arxiv.org/abs/1511.08630.

[22] Y. Guo, Y. Liu, E. M. Bakker, Y. Guo, and M. S. Lew, “CNN-RNN: a large-scale hierarchical image classification framework,” *Multimed. Tools Appl.*, vol. 77, no. 8, pp. 10251–10271, 2018, doi: 10.1007/s11042-017-5443-x.