Classification and Analysis of Android Malware Images Using Feature Fusion Technique

Jakkadanam Sumithra

Department of Computer Science, Sri Venkateswara University, Tirupati

***Abstract*—** *Security breaches due to attacks by malicious software (malware) continue to escalate posing a major security concern in this digital age. With many computer users, corporations, and governments affected due to an exponential growth in malware attacks, malware detection continues to be a hot research topic. Current malware detection solutions that adopt the static and dynamic analysis of malware signatures and behaviour patterns are time consuming and have proven to be ineffective in identifying unknown malwares in real-time. Recent malwares use polymorphic, metamorphic, and other evasive techniques to change the malware behaviours quickly and to generate a large number of new malwares. Such new malwares are predominantly variants of existing malwares, and machine learning algorithms (MLAs) are being employed recently to conduct an effective malware analysis. However, such approaches are time consuming as they require extensive feature engineering, feature learning, and feature representation. By using the advanced MLAs such as deep learning, the feature engineering phase can be completely avoided. Recently reported research studies in this direction show the performance of their algorithms with a biased training data, which limits their practical use in real-time situations. There is a compelling need to mitigate bias and evaluate these methods independently in order to arrive at a new enhanced method for effective zero-day malware detection. To fill the gap in the literature, this paper, first, evaluates the classical MLAs and machin learning architectures for malware detection, classification, and categorization using different public and private datasets. Second, we remove all the dataset bias removed in the experimental analysis by having different splits of the public and private datasets to train and test the model in a disjoint way using different timescales. Third, our major contribution is in proposing a novel image processing technique with optimal parameters for MLAs and deep learning architectures to arrive at an effective zero-day malware detection model. A comprehensive comparative study of our model demonstrates that our proposed deep learning architectures outperform classical MLAs.*

I. INTRODUCTION

The super packed functionalities and artificial intelligence (AI)-powered applications have made the Android operating system a big player in the market. Android smartphones have become an integral part of life and users are reliant on their smart devices for making calls, sending text messages, navigation, games, and financial transactions to name a few. This evolution of the smartphone community has opened new horizons for malware developers. As malware variants are growing at a tremendous rate every year, there is an urgent need to combat against stealth malware techniques. This paper proposes a visualization and machine learning-based framework for classifying Android malware. Users make use of these applications to their maximum advantage and tend to communicate, entertain business, relax, and educate themselves. The rapid adoption of such applications has resulted in the generation and sharing of sensitive information. The popularity of the Android operating system has also attracted cybercriminals to develop malicious applications to exploit Android users for monetary benefits. To perform classification, the aforementioned algorithms must be used in linear combination with machine learning classifiers such as Support Vector Machine (SVM), K-Nearest Neighbor (KNN), and Random Forest (RF).

1.1 Artificial intelligence:

Artificial intelligence (AI) is the ability of a computer program or a machine to think and learn. It is also a field of study which tries to make computers "smart". As machines become increasingly capable, mental facilities once thought to require intelligence are removed from the definition. AI is an area of computer sciences that emphasizes the creation of intelligent machines that work and reacts like humans. Some of the activities computers with artificial intelligence are designed for include: Face recognition, Learning, Planning, Decision making etc.,

Artificial intelligence is the use of computer science programming to imitate human thought and action by analysing data and surroundings, solving or anticipating problems and learning or self-teaching to adapt to a variety of tasks.

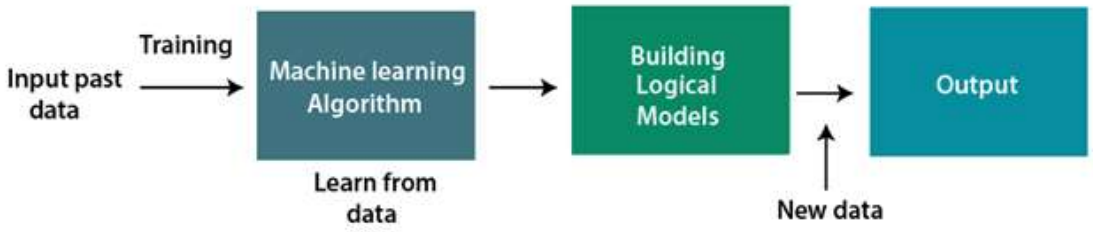
1.2 Machine Learning

Machine learning is a growing technology which enables computers to learn automatically from past data. Machine learning uses various algorithms for building mathematical models and making predictions using historical data or information. Currently, it is being used for various tasks such as image recognition, speech recognition, email filtering, Facebook auto-tagging, recommender system, and many more.

Machine Learning is said as a subset of artificial intelligence that is mainly concerned with the development of algorithms which allow a computer to learn from the data and past experiences on their own. The term machine learning was first introduced by Arthur Samuel in 1959. We can define it in a summarized way as: "Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed".

A Machine Learning system learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it. The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately.

Suppose we have a complex problem, where we need to perform some predictions, so instead of writing a code for it, we just need to feed the data to generic algorithms, and with the help of these algorithms, machine builds the logic as per the data and predict the output. Machine learning has changed our way of thinking about the problem. The below block diagram explains the working of Machine Learning algorithm:



1.2.1 Features of Machine Learning:

- Machine learning uses data to detect various patterns in a given dataset.
- It can learn from past data and improve automatically.
- It is a data-driven technology.
- Machine learning is much similar to data mining as it also deals with the huge amount of the data.

1.2.2 Classification of Machine Learning

At a broad level, machine learning can be classified into three types:

1. Supervised learning
2. Unsupervised learning
3. Reinforcement learning

1) Supervised Learning

Supervised learning is a type of machine learning method in which we provide sample labelled data to the machine learning system in order to train it, and on that basis, it predicts the output.

The system creates a model using labelled data to understand the datasets and learn about each data, once the training and processing are done then we test the model by providing a sample data to check whether it is predicting the exact output or not.

The goal of supervised learning is to map input data with the output data. The supervised learning is based on supervision, and it is the same as when a student learns things in the supervision of the teacher. The example of supervised learning is **spam filtering.**

Supervised learning can be grouped further in two categories of algorithms:

- **Classification**
- **Regression**

2) Unsupervised Learning

Unsupervised learning is a learning method in which a machine learns without any supervision. The training is provided to the machine with the set of data that has not been labelled, classified, or categorized, and the algorithm needs to act on that data without any supervision. The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns.

In unsupervised learning, we don't have a predetermined result. The machine tries to find useful insights from the huge amount of data.

It can be further classifieds into two categories of algorithms:

- **Clustering**
- **Association**

1.3 Random Forest Algorithm

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

Below are some points that explain why we should use the Random Forest algorithm:

- It takes less training time as compared to other algorithms.
- It predicts output with high accuracy, even for the large dataset it runs efficiently.
- It can also maintain accuracy when a large proportion of data is missing.

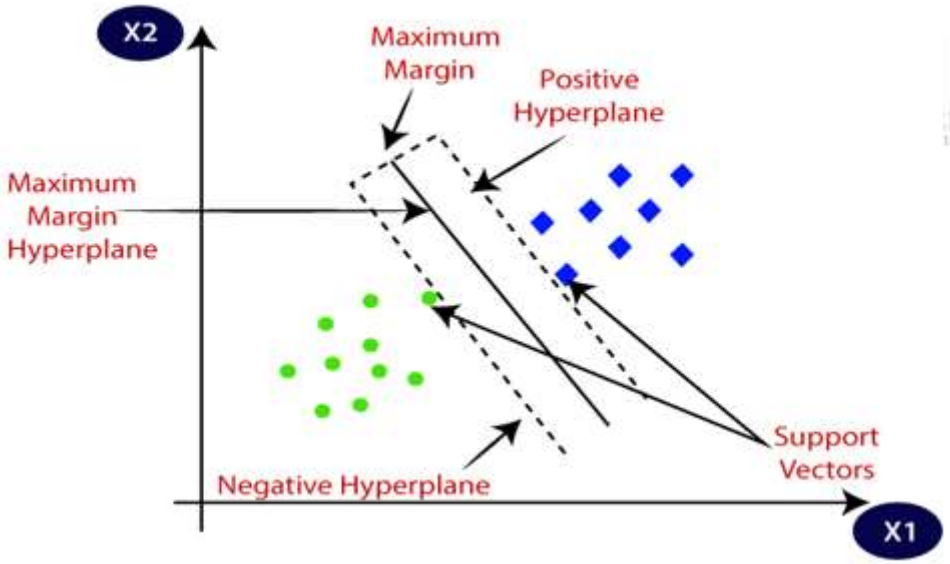
Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

- **Step-1:** Select random K data points from the training set.
- **Step-2:** Build the decision trees associated with the selected data points (Subsets).
- **Step-3:** Choose the number N for decision trees that you want to build.
- **Step-4:** Repeat Step 1 & 2.
- **Step-5:** For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

1.4 Support Vector Machine (Svm)

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.



SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:

Hyperplane: There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.

The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.

We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

Support Vectors:

The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

1.5 K-Nearest Neighbour (KNN)

K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique. K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories. K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm. K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems. K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data. It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset. KNN algorithm at the training phase just stores the dataset and when it gets new data, and then it classifies that data into a category that is much similar to the new data.

The K-NN working can be explained on the basis of the below algorithm:

Step-1: Select the number K of the neighbors

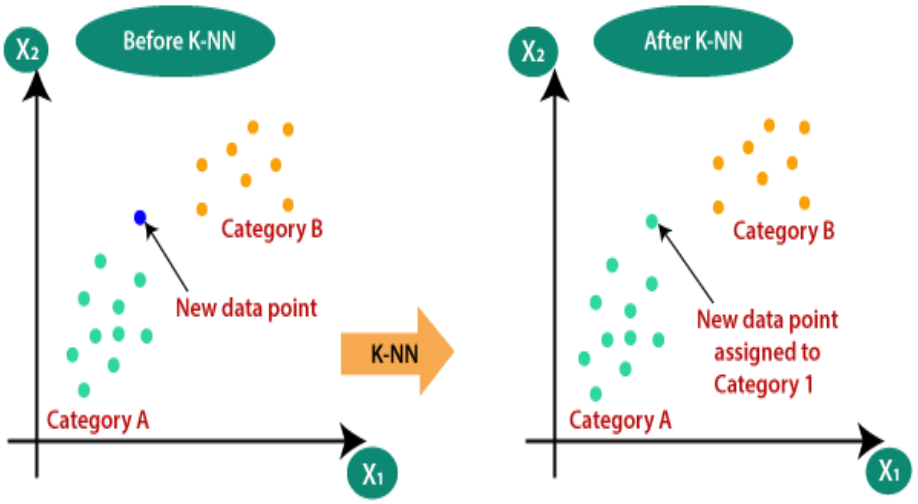
Step-2: Calculate the Euclidean distance of K number of neighbors

Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.

Step-4: Among these k neighbors, count the number of the data points in each category.

Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.

Step-6: Our model is ready.



II. LITERATURE REVIEW

[1] TITLE: Measuring the cost of cybercrime

AUTHOR: R. Anderson et al. - 2013

DESCRIPTION: The main categories of cybercrime we set out what is and is not known of the direct costs, indirect costs and defence costs – both to the UK and to the world as a whole. We distinguish carefully between traditional crimes that are now "cyber" because they are conducted online (such as tax and welfare fraud); transitional crimes whose modus operandi has changed substantially as a result of the move online (such as credit card fraud); new crimes that owe their existence to the Internet; and what we might call platform crimes such as the provision of botnets which facilitate other crimes rather than being used to extract money from victims directly. As far as direct costs are concerned, we find that traditional offences such as tax and welfare fraud cost the typical citizen in the low hundreds of pounds/euros/dollars a year; transitional frauds cost a few pounds/euros/dollars; while the new computer crimes cost in the tens of pence/cents. However, the indirect costs and defence costs are much higher for transitional and new crimes. For the former they may be roughly comparable to what the criminals earn, while for the latter they may be an order of magnitude more.

[2] TITLE: Large-scale identification of malicious singleton files,

AUTHOR: B. Li, K. Roundy, C. Gates, and Y. Vorobeychik -2017

DESCRIPTION: We study a dataset of billions of program binary files that appeared on 100 million computers over the course of 12 months, discovering that 94% of these files were present on a single machine. Though malware polymorphism is one cause for the large number of singleton files, additional factors also contribute to polymorphism, given that the ratio of benign to malicious singleton files is 80:1. The huge number of benign singletons makes it challenging to reliably identify the minority of malicious singletons. We present a large-scale study of the properties, characteristics, and distribution of benign and malicious singleton files. We leverage the insights from this study to build a classifier based purely on static features to identify 92% of the remaining malicious singletons at a 1.4% percent false positive rate, despite heavy use of obfuscation and packing techniques by most malicious singleton files that we make no attempt to de-obfuscate. Finally, we demonstrate robustness of our classifier to important classes of automated evasion attacks.

[3] TITLE: Towards understanding malware behaviour by the extraction of API calls

AUTHOR: M. Alazab, S. Venkataraman, and P. Watters- 2010

DESCRIPTION: With evasion techniques such as polymorphism and metamorphism malware is able to fool current detection techniques. Thus, security researchers and the anti-virus industry are facing a herculean task in extracting payloads hidden within packed executables. It is a common practice to use manual unpacking or static unpacking using some software tools and analyse the application programming interface (API) calls for malware detection. However, extracting these features from the

unpacked executables for reverse obfuscation is labour intensive and requires deep knowledge of low-level programming that includes kernel and assembly language. This paper presents an automated method of extracting API call features and analysing them in order to understand their use for malicious purpose. While some research has been conducted in arriving at file birthmarks using API call features and the like, there is a scarcity of work that relates to features in malcodes. To address this gap, we attempt to automatically analyse and classify the behavior of API function calls based on the malicious intent hidden within any packed program.

[4] TITLE: Zero-day malware detection based on supervised learning algorithms of API call signatures

AUTHOR: M. Alazab, S. Venkatraman, P. Watters, and M. Alazab - 2011

DESCRIPTION: Zero-day or unknown malware are created using code obfuscation techniques that can modify the parent code to produce offspring copies which have the same functionality but with different signatures. Current techniques reported in literature lack the capability of detecting zero-day malware with the required accuracy and efficiency. In this paper, we have proposed and evaluated a novel method of employing several data mining techniques to detect and classify zero-day malware with high levels of accuracy and efficiency based on the frequency of Windows API calls. This paper describes the methodology employed for the collection of large data sets to train the classifiers, and analyses the performance results of the various data mining algorithms adopted for the study using a fully automated tool developed in this research to conduct the various experimental investigations and evaluation. Through the performance results of these algorithms from our experimental analysis, we are able to evaluate and discuss the advantages of one data mining algorithm over the other for accurately detecting zero-day malware successfully

[5] TITLE: 'Cybercrime: The case of obfuscated malware

AUTHOR: M. Alazab, S. Venkatraman, P. Watters, M. Alazab, and A. Alazab, - 2012

DESCRIPTION: Cybercrime has rapidly developed in recent years and malware is one of the major security threats in computer which have been in existence from the very early days. There is a lack of understanding of such malware threats and what mechanisms can be used in implementing security prevention as well as to detect the threat. The main contribution of this paper is a step towards addressing this by investigating the different techniques adopted by obfuscated malware as they are growingly widespread and increasingly sophisticated with zero-day exploits. In particular, by adopting certain effective detection methods our investigations show how cybercriminals make use of file system vulnerabilities to inject hidden malware into the system.

III. PROBLEM STATEMENT

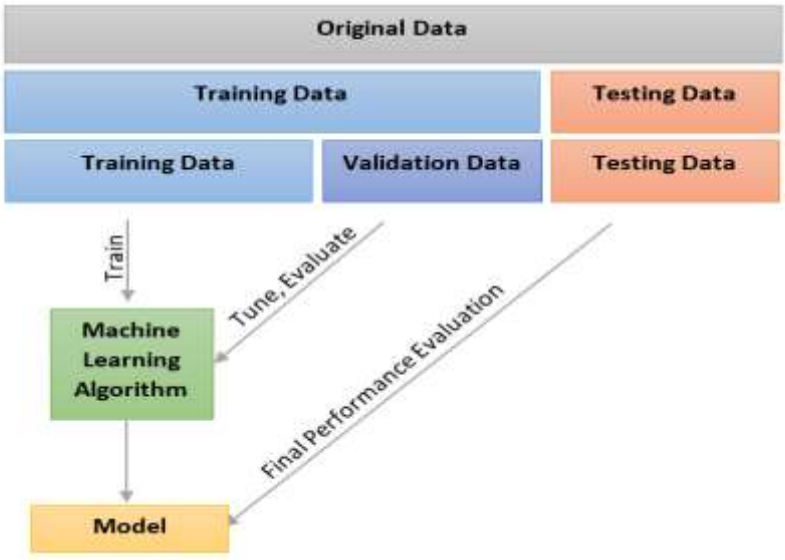
Existing System

This manuscript proposes Summing of neurAl architecture and Visualizati On Technology for Android Malware identification (SARVOTAM). The system converts the malware non-intuitive features into fingerprint images to extract the quality information. A fine-tuned Convolutional Neural Network (CNN) is used to automatically extract rich features from visualized malware. The experiments were done using the DREBIN dataset. A total of fifteen different combinations of the Android malware image sections were used to identify and classify Android malware. The softmax layer of CNN was substituted with machine learning algorithms like K-Nearest Neighbor (KNN), Support Vector Machine (SVM), and Random Forest (RF) to analyze the grayscale malware images. The classification results showed that our method is able to achieve an accuracy of 92.59% using Android certificates and manifest malware images.

Disadvantages

- Final results get impacted more with initial seeds
- Final results are impacted more by order of data

IV. ARCHITECTURE



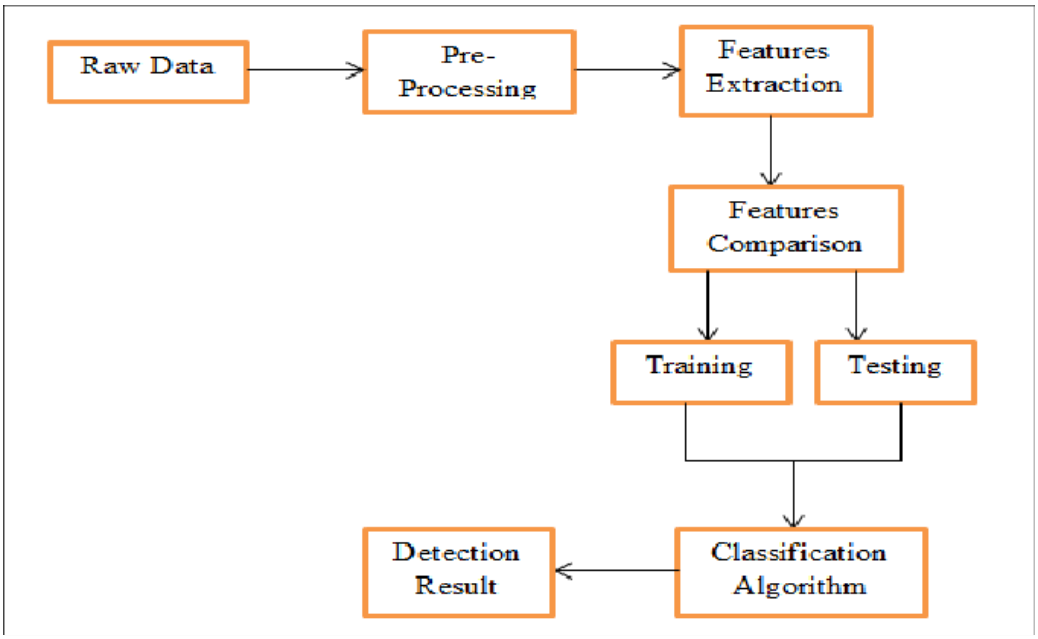
PROPOSED SYSTEM

- The classification performance was evaluated against every malware file section.
- The model is most suited for the identification and classification of Android malware using Machine learning algorithm (KNN, SVM, and Random Forest).
- The final output is detection and produce in '0's and '1's.

ADVANTAGES

- It was found that the classification performance of all the classifiers eventually increased when feature was deployed.
- The primary focus of this study was on the feature extraction technique to identify the descriptors.
- Accuracy is high.

V. SYSTEM ARCHITECTURE



SYSTEM MODULES

MODULE 1: DATA COLLECTION

Data collection is the process of gathering and measuring information from countless different sources.

In order to use the data we collect to develop practical artificial intelligence (AI) and machine learning solutions, it must be collected and stored in a way that makes sense for the business problem at hand.

MODULE 2: PRE-PROCESSING

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

MODULE 3: MODEL IMPLEMENTATION

Random Forest

KNN

SVM

RANDOM FOREST

Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables as in the case of regression and categorical variables as in the case of classification. It performs better results for classification problems.

Working of Random Forest Algorithm

Before understanding the working of the random forest we must look into the ensemble technique. Ensemble simply means combining multiple models. Thus a collection of models is used to make predictions rather than an individual model.

Ensemble uses two types of methods:

1) Bagging– It creates a different training subset from sample training data with replacement & the final output is based on majority voting. For example, Random Forest.

2. **Boosting**– It combines weak learners into strong learners by creating sequential models such that the final model has the highest accuracy.

Steps involved in random forest algorithm:

Step 1: In Random forest n number of random records are taken from the data set having k number of records.

Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression respectively.

1) *Important Features of Random Forest*

1. Diversity- Not all attributes/variables/features are considered while making an individual tree, each tree is different.

2. Immune to the curse of dimensionality- Since each tree does not consider all the features, the feature space is reduced.

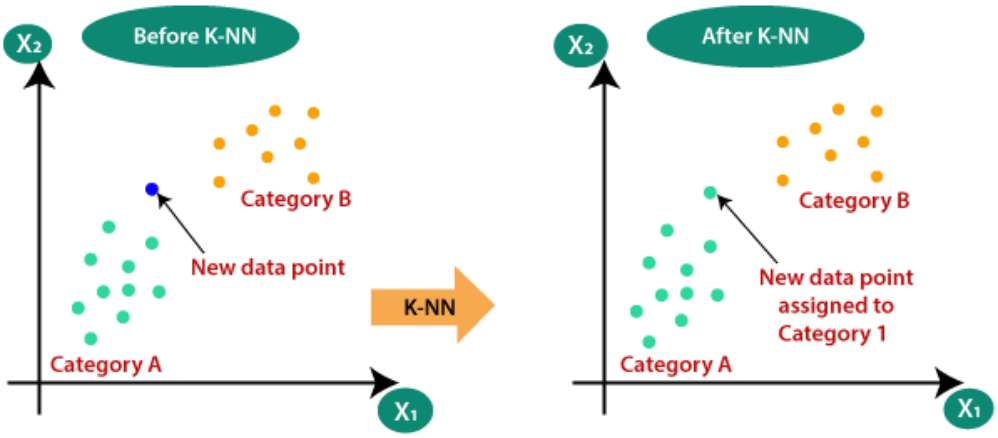
3. Parallelization-Each tree is created independently out of different data and attributes. This means that we can make full use of the CPU to build random forests.

4. Train-Test split- In a random forest we don't have to segregate the data for train and test as there will always be 30% of the data which is not seen by the decision tree.

5. Stability- Stability arises because the result is based on majority voting/ averaging.

KNN

- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.
- It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.



B.

The K-NN working can be explained on the basis of the below algorithm:

- **Step-1:** Select the number K of the neighbors
- **Step-2:** Calculate the Euclidean distance of K number of neighbors
- **Step-3:** Take the K nearest neighbors as per the calculated Euclidean distance.
- **Step-4:** Among these k neighbors, count the number of the data points in each category.
- **Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.
- **Step-6:** Our model is ready.

SVM

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

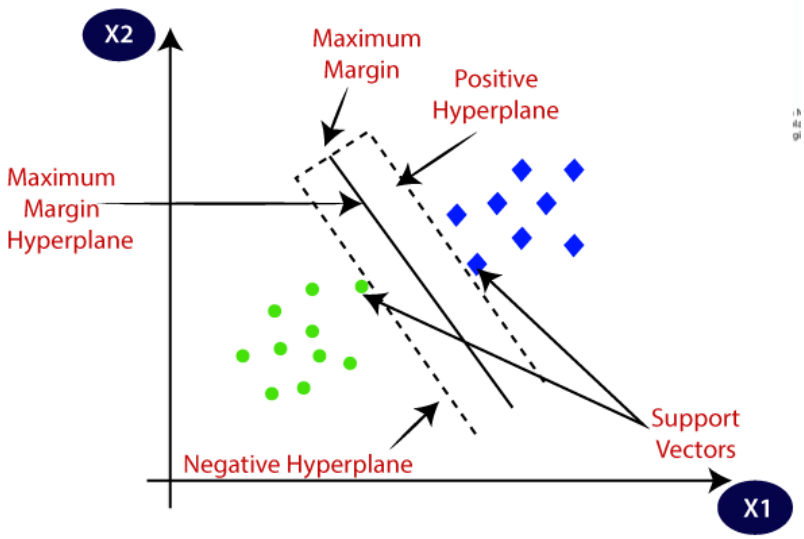
The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:

Types of SVM

SVM can be of two types:

- **Linear SVM:** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
- **Non-linear SVM:** Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.



MODULE 4: **CLASSIFICATION**

Train and test should be set according to the user convenience and play with the basic hyper parameters like epoch, learning rate, etc. to improve the accuracy.

I have converted the image to grayscale so that we will only have to deal with 2-d matrix otherwise 3-d matrix is tough to directly apply ML algorithms to, especially not recommended for beginners.

MODULE 5: DETECTION MALWARE

The classification performance was evaluated against every malware file section. The results obtained using the Feature Extraction strategy are compared with each classifier and final detection result gives whether malware present or not. Decision is regarded as a superior algorithm in terms of efficient memory utilization, high accuracy and low running times. It is simply highly robust and scalable technique to handle all sorts of noise from huge data sets and convert the data into a ready acceptable form for precision results. Hence the input is in csv format the output is displayed in 0's and 1's. Where '0' is no malware '1' is malware detected. Finally confusion matrix is plotted.

VI. CONCLUSION

The practice of using Machine Learning methods as compared to conventional methods for malware detection is growing rapidly. The supervised machine learning methods need a dataset as an input that will train the model and after gaining experience from it the model will perform on the test data. For that, we created our dataset providing a variety of malware files downloaded from the world's various famous malware projects. Our dataset generation process includes Android File

Collection, Decompilation, and Feature Mining phases. Here we have proposed a malware detection process using machine learning classifiers. We have evaluated the performance of the generated dataset by using supervised classifiers.

REFERENCES

[1] R. Anderson et al., ''Measuring the cost of cybercrime,'' in The Economics of Information Security and Privacy. Berlin, Germany: Springer, 2013, pp. 265–300.

[2] B. Li, K. Roundy, C. Gates, and Y. Vorobeychik, ''Large-scale identification of malicious singleton files,'' in Proc. 7th ACM Conf. Data Appl. Secur. Privacy. New York, NY, USA: ACM, Mar. 2017, pp. 227–238.

[3] M. Alazab, S. Venkataraman, and P. Watters, ''Towards understanding malware behaviour by the extraction of API calls,'' in Proc. 2nd Cybercrime Trustworthy Comput. Workshop, Jul. 2010, pp. 52–59.

[4] M. Tang, M. Alazab, and Y. Luo, ''Big data for cybersecurity: Vulnerability disclosure trends and dependencies,'' IEEE Trans. Big Data, to be published.

[5] M. Alazab, S. Venkatraman, P. Watters, and M. Alazab, ''Zero-day malware detection based on supervised learning algorithms of API call signatures,'' in Proc. 9th Australas. Data Mining Conf., vol. 121. Ballarat, Australia: Australian Computer Society, Dec. 2011, pp. 171–182.

[6] M. Alazab, S. Venkatraman, P. Watters, M. Alazab, and A. Alazab, ''Cybercrime: The case of obfuscated malware,'' in Global Security, Safety and Sustainability & e-Democracy (Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering), vol. 99, C. K. Georgiadis, H. Jahankhani, E. Pimenidis, R. Bashroush, and A. Al-Nemrat, Eds. Berlin, Germany: Springer, 2012.

[7] M. Alazab, ''Profiling and classifying the behavior of malicious codes,'' J. Syst. Softw., vol. 100, pp. 91–102, Feb. 2015.

[8] S. Huda, J. Abawajy, M. Alazab, M. Abdollalihian, R. Islam, and J. Yearwood, ''Hybrids of support vector machine wrapper and filter based framework for malware detection,'' Future Gener. Comput. Syst., vol. 55, pp. 376–390, Feb. 2016.

[9] E. Raff, J. Sylvester, and C. Nicholas, ''Learning the PE header, malware detection with minimal domain knowledge,'' in Proc. 10th ACM Workshop Artif. Intell. Secur. New York, NY, USA: ACM, Nov. 2017, pp. 121–132.

[10] C. Rossow, et al., ''Prudent practices for designing malware experiments: Status quo and outlook,'' in Proc. IEEE Symp. Secur. Privacy (SP), Mar. 2012, pp. 65–79.

[11] E. Raff, J. Barker, J. Sylvester, R. Brandon, B. Catanzaro, and C. Nicholas. (2017). ''Malware detection by eating a whole exe.'' [Online]. Available: https://arxiv.org/abs/1710.09435

[12] M. Krcál, O. Švec, M. Bálek, and O. Jašek. (2018). Deep Convolutional Malware Classifiers Can Learn from Raw Executables and Labels Only. [Online]. Available: https://openreview.net/forum?id=HkHrmM1PM

[13] M. Rhode, P. Burnap, and K. Jones, ''Early-stage malware prediction using recurrent neural networks,'' Comput. Secur., vol. 77, pp. 578–594, Aug. 2018.

[14] H. S. Anderson, A. Kharkar, B. Filar, and P. Roth, Evading Machine Learning malware Detection. New York, NY, USA: Black Hat, 2017.

[15] R. Verma, ''Security analytics: Adapting data science for security challenges,'' in Proc. 4th ACM Int. Workshop Secur. Privacy Anal. New York, NY, USA: ACM, Mar. 2018, pp. 40–41.

[16] Y. LeCun, Y. Bengio, and G. Hinton, ''Deep learning,'' Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[17] A. F. Agarap and F. J. H. Pepito. (2017). ''Towards building an intelligent anti-malware system: A deep learning approach using support vector machine (SVM) for malware classification.'' [Online]. Available: https://arxiv.org/abs/1801.00318

[18] E. Rezende, G. Ruppert, T. Carvalho, A. Theophilo, F. Ramos, and P. de Geus, ''Malicious software classification using VGG16 deep neural network's bottleneck features,'' in Information Technology-New Generations. Cham, Switzerland: Springer, 2018, pp. 51–59.

[19] J. Saxe and K. Berlin, ''Deep neural network based malware detection using two dimensional binary program features,'' in Proc. 10th Int. Conf. Malicious Unwanted Softw. (Malware), Oct. 2015, pp. 11–20.

[20] S. Tobiyama, Y. Yamaguchi, H. Shimada, T. Ikuse, and T. Yagi, ''Malware detection with deep neural network using process behavior,'' in Proc. IEEE 40th Annu. Comput. Softw. Appl. Conf. (COMPSAC), vol. 2, Jun. 2016, pp. 577–582.

[21] W. Huang, J. W. Stokes, ''Mtnet: A multi-task neural network for dynamic malware classification,'' in Proc. Int. Conf. Detection Intrusions Malware, Vulnerability Assessment, Cham, Switzerland: Springer, Jul. 2016, pp. 399–418.

[22] R. Pascanu, J. W. Stokes, H. Sanossian, M. Marinescu, and A. Thomas, ''Malware classification with recurrent networks,'' in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), Apr. 2015, pp. 1916–1920.

[23] T. Shibahara, T. Yagi, M. Akiyama, D. Chiba, and T. Yada, ''Efficient dynamic malware analysis based on network behavior using deep learning,'' in Proc. IEEE Global Commun. Conf. (GLOBECOM), Dec. 2016, pp. 1–7.

[24] B. Kolosnjaji, A. Zarras, G. Webster, and C. Eckert, ''Deep learning for classification of malware system call sequences,'' in Proc. Australas. Joint Conf. Artif. Intell. Cham, Switzerland: Springer, Dec. 2016, pp. 137–149.

[25] E. Raff et al., ''An investigation of byte n-gram features for malware classification,'' J. Comput. Virology Hacking Techn., vol. 14, no. 1, pp. 1–20, 2018.