MALIGNANT CODE VARIATION IDENTIFICATION USING DEEP LEARNING

A.Nagarjuna Reddy, Associate professor, Sridevi Women's Engineering College, Hyderabad,
Email id: anr304@gmail.com
Shivani Thakur, B.tech, Dept of Information Technology, Sridevi Women's Engineering College, Hyderabad,
Email id: shivani1420thakur@gmail.com
Neelam Chaitanyasri, B.Tech, Dept of Information Technology, Sridevi Women's Engineering College, Hyderabad,
Email id: chaitanyasrineelam13@gmail.com
Dheekshitha Thaduri, B.Tech, Dept of Information Technology, Sridevi Women's Engineering College, Hyderabad,
Email id: dheekshithaduri2@gmail.com
Afreen Tabassum, B.Tech, Dept of Information Technology, Sridevi Women's Engineering College, Hyderabad,
Email id: ruhitabassum303@gmail.com

ABSTRACT: With the development of the Internet, malicious code attacks have increased exponentially, with malicious code variants ranking as a key threat to Internet security. The ability to detect variants of malicious code is critical for protection against security breaches, data theft, and other dangers. Current methods for recognizing malicious code have demonstrated poor detection accuracy and low detection speeds. This paper proposed a novel method that used deep learning to improve the detection of malware variants. In prior research, deep learning demonstrated excellent performance in image recognition. To implement our proposed detection method, we converted the malicious code into grayscale images. Then, the images were identified and classified using a convolutional neural network (CNN) that could extract the features of the malware images automatically. To test our approach, we conducted a series of experiments on malware image data from Vision Research Lab. The experimental results demonstrated that our model achieved good accuracy and speed as compared with other malware detection models.

Keywords: Malware variants, grayscale image, deep learning, convolution neural network.

1. INTRODUCTION

With the rapid development of information technology, the exponential growth of malicious code has become one of the main threats to internet security. A recent report from Symantec showed that 401 million malicious codes were found in 2016, including 357 million new malicious code variants. The appearance of malware in mobile devices and the Internet of Things (IoT) also grew rapidly. Till date, 68 new malicious code families and more than 10 000 malicious codes have been reported. This growth has posed a challenge for malicious code detection in cloud computing [14], [15]. As a key part of security protection, uncovering malicious code variants is particularly challenging. Malware detection methods consist primarily of two types of approaches: static detection and dynamic detection. Static detection works by disassembling the malware code and analyzing its execution logic. Dynamic detection analyzes the behavior of malicious code by executing the code in a safe virtual environment or sandbox. Both static and dynamic detection are feature-based detection methods. First, the textual or behavioral features of the malicious code are extracted and, then, the malicious code is detected or classified by analyzing these extracted features.

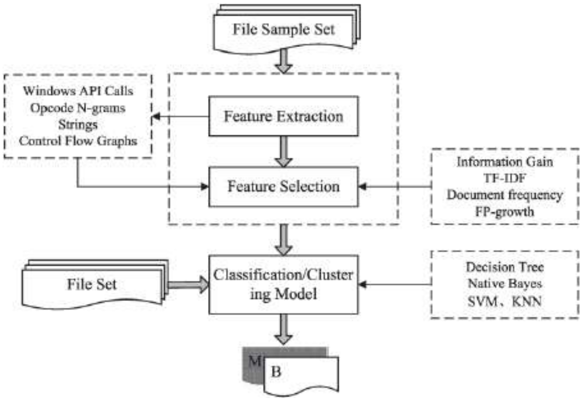


Fig.1 Malicious code detection based on data mining.

In recent years, several scholars have used data mining methods to analyze the features of malicious code. This approach has become the mainstay of malware detection because it is highly efficient and has a low rate of false positives compared with traditional heuristic-based detection methods. Fig. 1 illustrates the process of detecting malicious code using data mining. Unfortunately, methods based on feature analysis are often disrupted. The effectiveness of static feature analysis can be hampered by the obfuscation techniques that transform the malware binary into a self-compressed or uniquely structured binary. Dynamic feature analysis is often challenged by many kinds of countermeasures developed to produce unreliable results. Furthermore, some types of malicious code may be ignored by dynamic analysis because the execution environment does not comply with the rules.

2. LITERATURE REVIEW

Evaluating Machine Learning Classifiers to detect Android Malware

Malware Detection using conventional methods is incompetent to detect new and generic malware. For the investigation of a variety of malware, there were no ready-made machine learning datasets available for malware detection. So we generated our dataset by downloading a variety of malware files from the world's famous malware projects. By performing unstructured data collection from the downloaded APK files and feature mining process the final dataset was generated with 16300 records and a total of 215 features. There was a need to evaluate the performance of the generated dataset with supervised machine learning classifiers. So in this paper, we propose a malware detection approach using different supervised machine learning classifiers. Here supervised algorithms, Feature Reduction Techniques, and Ensembling techniques are used to evaluate the performance of the generated dataset. Machine Learning classifiers are evaluated on the evaluation parameters like AUC, FPR, TPR, Cohen Kappa Score, Precision, and Accuracy. We also represented the results of classifiers using Bar plots of Accuracy and plotting the ROC curve. From the results of machine learning classifiers, the performance of the CatBoost Classifier is highest with Accuracy 93.15% having a value of ROC curve as 0.91 and Cohen Kappa Score as 81.56%.

A Static Malware Detection System Using Data Mining Methods

A serious threat today is malicious executables. It is designed to damage computer system and some of them spread over network without the knowledge of the owner using the system. Two approaches have been derived for it i.e. Signature Based Detection and Heuristic Based Detection. These approaches performed well against known malicious programs but cannot catch the new malicious programs. Different researchers have proposed methods using data mining and machine learning for detecting new malicious programs. The method based on data mining and machine learning has shown good results compared to other approaches. This work presents a static malware detection system using data mining techniques such as Information Gain, Principal component analysis, and three classifiers: SVM, J48, and Na\"ive Bayes. For overcoming the lack of usual anti-virus products, we use methods of static analysis to extract valuable features of Windows PE file. We extract raw features of Windows executables which are PE header information, DLLs, and API functions inside each DLL of Windows PE file. Thereafter, Information Gain, calling frequencies of the raw features are calculated to select valuable subset features, and then Principal Component Analysis is used for dimensionality reduction of the selected features. By adopting the concepts of machine learning and data-mining, we construct a static malware detection system which has a detection rate of 99.6%.

Mobile Malware Detection using Anomaly Based Machine Learning Classifier Techniques

Mobile phones are a significant component of people's life and are progressively engaged in these technologies. Increasing customer numbers encourages the hackers to make malware. In addition, the security of sensitive data is regarded lightly on mobile devices. Based on current approaches, recent malware changes fast and thus become more difficult to detect. In this paper an alternative solution to detect malware using anomaly-based classifier is proposed. Among the variety of machine learning classifiers to classify the latest Android malwares, a novel mixed kernel function incorporated with improved support vector machine is proposed. In processing the categories selected are general information, data content, time and connection information among various network functions. The experimentation is performed on MalGenome dataset. Upon implementation of proposed mixed kernel SVM method, the obtained results of performance achieved 96.89% of accuracy, which is more effective compared with existing models.

Credroid: Android Malware Detection By Network Traffic Analysis

Android, one of the most popular open source mobile operating system, is facing a lot of security issues. Being used by users with varying degrees of awareness complicates the problem further. Most of the security problems are due to maliciousness of android applications. The malwares get installed in mobile phones through various popular applications particularly gaming applications or some utility applications from various third party app-stores which are untrustworthy. A common feature of the malware is to access the sensitive information from the mobile device and transfer it to remote servers. For our work, we have confined ourselves to defining maliciousness as leakage of privacy information by Android application. In this paper we have proposed a method named as CREDROID which identifies malicious applications on the basis of their Domain Name Server(DNS) queries as well as the data it transmits to remote server by performing the in-depth analysis of network traffic logs in offline mode. Instead of performing signature based detection which is unable to detect polymorphic malwares, we propose a pattern based detection. Pattern in our work refers to the leakage of sensitive information being sent to the remote server. CREDROID is a semi-automated approach which works on various factors like the remote server where the application is connecting, data being sent and the protocol being used for communication for identifying the trustworthiness (credibility) of the application. In our work, we have observed that 63% of the applications from a standard dataset of malwares are generating network traffic which has been the focus of our work.

The rise of machine learning for detection and classification of malware: Research developments, trends and challenges

The struggle between security analysts and malware developers is a never-ending battle with the complexity of malware changing as quickly as innovation grows. Current state-of-the-art research focus on the development and application of machine learning techniques for malware detection due to its ability to keep pace with malware evolution. This survey aims at providing a systematic and detailed overview of machine learning techniques for malware detection and in particular, deep learning techniques. The main contributions of the paper are: (1) it provides a complete description of the methods and features in a traditional machine learning workflow for malware detection and classification, (2) it explores the challenges and limitations of traditional machine learning and (3) it analyzes recent trends and developments in the field with special emphasis on deep learning approaches. Furthermore, (4) it presents the research issues and unsolved challenges of the state-of-the-art techniques and (5) it discusses the new directions of research. The survey helps researchers to have an understanding of the malware detection field and of the new developments and directions of research explored by the scientific community to tackle the problem.

3. IMPLEMENTATION

Existing system

Network security is the key challenge for many enterprise level applications which are mainly used to identify the security threats associated across different types of attacks by hackers as well as intruders who are always involved in spoofing the data during the data transmission.

The following report has been prepared in light of the analysis of the various problems associated with network and cyber security based datasets and the applications which are using this datasets across various enterprise level architectures like network and cyber security, This methodology helps in Building Development, k-means analysis and Machine Learning Implementations. It has been aptly demonstrated that the effective integration of Data Analytics methodologies has led to economized efforts in the practices of the various organizations facilitating their operations by drawing crucial insights from the relevant data in light of the concealed patterns, preferences of Consumers, trends prevalent in the market and the respective unknown correlation of the different elements under various circumstances of the environment.

Limitations:

❖ Machine learning gave a summary of various machine learning techniques that were previously proposed for malware detection. Unlike Machine Learning, Deep learning skips the manual steps of extracting features.

Proposed Method

This paper proposed a completely unique method that used deep learning to enhance the detection of malware variants. To implement our proposed detection method, we converted the malicious code into gray scale images. Then the photographs were identified and classified employing a convolutional neural network (CNN) that might extract the features of the malware images automatically. To check our approach, we conducted a series of experiments on malware image data from Vision lab .

Advantages:

The experimental results demonstrated that our model achieved good accuracy and speed as compared with other malware detection models.

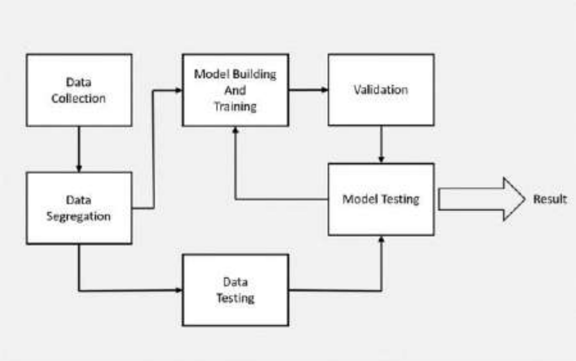


Fig.2: System architecture

Malware detection methods rely mainly on analysis of the features of malicious codes (e.g., static features and dynamic features). More powerful detection methods based on various machine learning techniques also use these features to uncover malicious codes or their variations. However, these approaches become less effective when detecting malicious code variants or unknown malware. The malware visualization method can handle code obfuscation problems, but it suffers from the high time cost needed for complex image texture feature extraction (e.g., GIST and GLCM). Moreover, these feature extraction methods also demonstrate low efficiency when exposed to large datasets. The challenge for building malware detection models is to find a means for extracting features effectively and automatically. Moreover, the data imbalance problem imposes another challenge. Of the large quantity of malware generated each year, a substantial portion includes variants that belong to existing malicious code families or groups. Usually, the number of malicious code variations differs greatly among various code families. The challenge is to build a universal detection model that can deal with the huge volume of variations, so that it can work well across malware families.

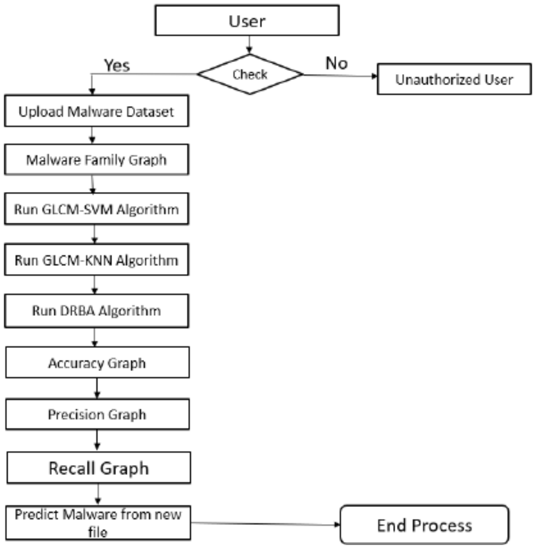


Fig.3: Data Flow diagram

4. ALGORITHMS

Convolutional Neural Network:

Convolutional Neural Networks (CNNs) are a deep learning approach to tackle the image classification problem, or what we call computer vision problems, because classic computer programs face many challenges and difficulties to spot objects for several reasons, including lighting, viewpoint, deformation, and segmentation .This technique is inspired by how the attention works, especially the visual area function algorithm in animals. CNN are arranged in three dimensional structures with width, height, and depth as characteristics. Within the case of images, the peak is the image height, the width is the image width, and therefore the depth is RGB channels.

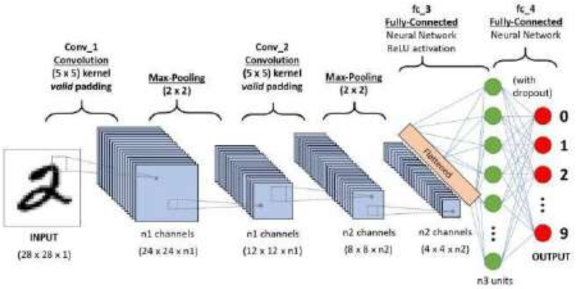


Fig.4: CNN model

Dynamic Range based Algorithm:

Deep learning has the ability to learn the essential characteristics of data sets from a sample set. As a powerful tool of artificial intelligence, deep learning has been applied widely in many fields, such as recognition of handwritten numerals, speech recognition, and image recognition. Because of it powerful ability to learn features, many scholars have applied deep learning to malware detection. Using deep learning techniques, Yuan et al. designed and implemented an online malware detection prototype system, named Droid-Sec. Their model achieved high accuracy by learning the features extracted from both static analysis and dynamic analysis of Android apps. David et al, presented a similar but more compelling method that did not need the type of malware behavior. Their work was based on a deep belief network (DBN) for automatic malware signature generation and classification. Compared with conventional signature methods for malware detection, their approach demonstrated increased accuracy for detecting new malware variants. Unfortunately, these methods remained based on the analysis of features extracted by static analysis and dynamic analysis. Therefore, to a greater or lesser extent, they continued to be subject to the limitations of feature extraction. To address this problem, we employed a CNN network to learn the malware image features and classify them automatically.

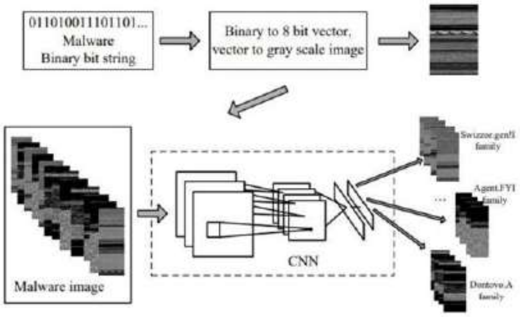


Fig 5: Dynamic-Range-based algorithm

Binary Malware to Gray Image in general, there are several ways to transform binary code into images. In this paper, we used the visualization of executable malware binary files.

Gray Level Co-occurrence Matrix:

A GLCM uses the texture classification concept. The texture classification concept is classified using the homogeneity value. The homogeneity value is calculated for every pixel to present inside the image. After calculating the homogeneity values, a matrix of values is created. If there is a change in the homogeneity value of the particular pixel, then the GLCM value is calculated. In the brain, the X-ray tumor part is different from the rest of the gray mass. The gray mass has a different texture in comparison to the tumor texture. At that point, GLCM is the best approach. If there is a sharp change in the matrix value, there is the highest chance of getting the tumor. GLCM is the best approach for classifying the pixel by pixel values. Gray level co-occurrence value is mostly used in X-ray analysis. X-ray is a black and white film, and it contains different shades of the gray.

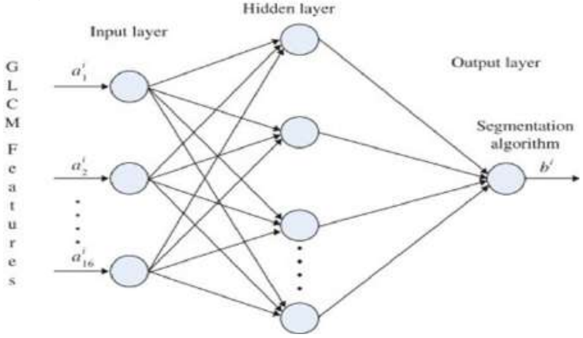


Fig 6: Gray-Level Co-occurrence matrix

K-Nearest Neighbor Algorithm:

K-Nearest Neighbors is one of the most basic yet essential classification algorithms in Machine Learning. It belongs to the supervised learning domain and finds intense application in pattern recognition, data mining and intrusion detection. It is widely disposable in real-life scenarios since it is non-parametric, meaning, it does not make any underlying assumptions about the distribution of data (as opposed to other algorithms such as GMM, which assume a Gaussian distribution of the given data). We are given some prior data (also called training data), which classifies coordinates into groups identified by an attribute.

Support Vector Machine Algorithm:

“Support Vector Machine” (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is the number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well (look at the below snapshot). The SVM algorithm is implemented in practice using a kernel. The learning of the hyperplane in linear SVM is done by transforming the problem using some linear algebra, which is out of the scope of this introduction to SVM.

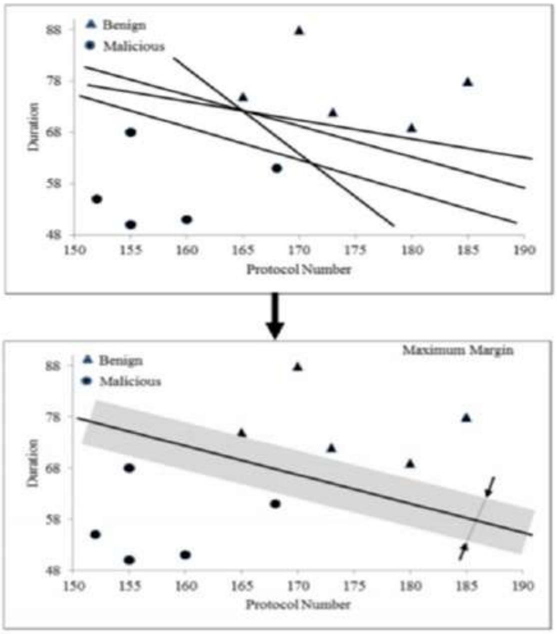


Fig 7: Support Vector Algorithm

5. EXPERIMENTAL RESULTS

In this paper the author is using CNN (Convolution Neural Network) to predict malicious code. From the internet various software's can be downloaded and this software may contain malicious code and upon execution of such software can cause file corruption or data loss. All existing technique may use static or dynamic technique to identify such malicious code but its detection rate is to less and to overcome from this problem author using CNN algorithm to train CNN model and this trained model can be applied on new malicious code or test data to detect malware family. Extraction of image texture features are called GLCM and the author is using malware dataset to implement the above algorithm and this dataset contains binary data and this binary data can be converted to gray color images as both binary and image data contains values between 0-255. Generated images can be used to train DRBA CNN models.

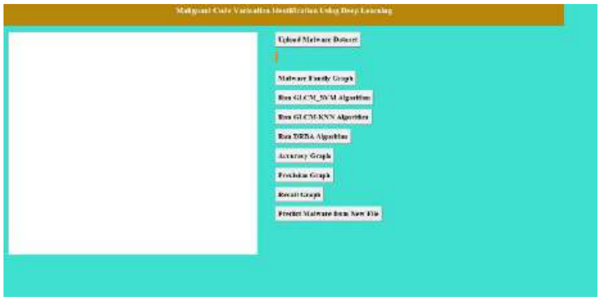


Fig.8: Home screen

In above screen click on 'Upload Malware Dataset' button to upload malware dataset

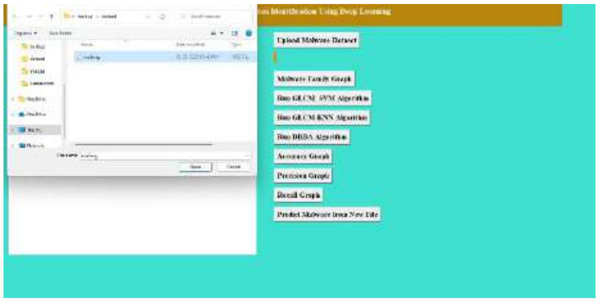


Fig.9: Upload malware dataset

In above screen selecting and uploading ‘malimg.npz’ dataset file and after uploading dataset will get below screen

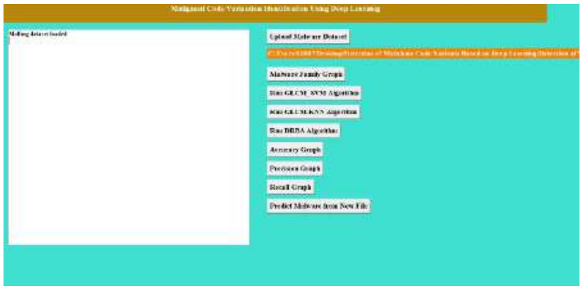


Fig.10: Malware dataset loaded

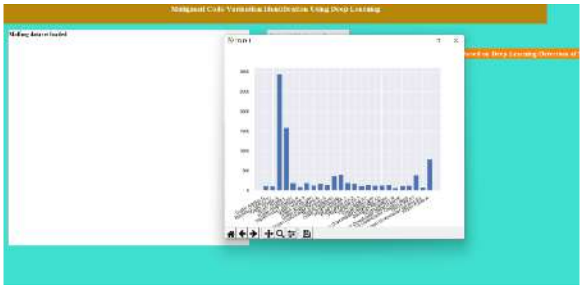
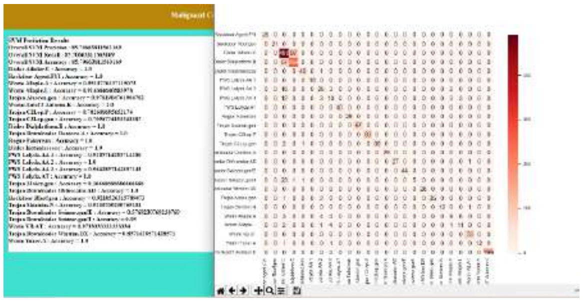


Fig.11: Malware family Graph

In the above screen click on Malware Family graph it opens a graph which represents no . of malware with respect to families of malware. click on ‘Run GLCM_SVM Algorithm’ and calculate its precision, recall and accuracy



```
SVM Prediction Results
Overall SVM Precision : [illegible]
Overall SVM Recall : [illegible]
Overall SVM Accuracy : [illegible]
Dialer.Adialer.C : Accuracy : 1.0
Backdoor.Agent.FYI : Accuracy = 1.0
Worm.Allaple.A : Accuracy = [illegible]
Worm.Allaple.L : Accuracy = [illegible]
Trojan.Alueron.gen : Accuracy = [illegible]
Worm.AutoIT.Autorun.K : Accuracy : 1.0
Trojan.C2Lop.P : Accuracy = [illegible]
Trojan.C2Lop.gen : Accuracy = [illegible]
Dialer.Dialplatform.B : Accuracy = 1.0
Trojan.Downloader.Dontovo.A : Accuracy = 1.0
Rogue.Fakerean : Accuracy = 1.0
Dialer.Instantaccess : Accuracy = 1.0
PWS.Lolyda.AA.1 : Accuracy : [illegible]
PWS.Lolyda.AA.2 : Accuracy = 1.0
PWS.Lolyda.AA.3 : Accuracy = [illegible]
PWS.Lolyda.AT : Accuracy = 1.0
Trojan.Malex.gen : Accuracy = [illegible]
Trojan.Downloader.Obfuscator.AD : Accuracy = 1.0
Backdoor.Rbot!gen : Accuracy = [illegible]
Trojan.Skintrim.N : Accuracy : [illegible]
Trojan.Downloader.Swizzor.gen!E : Accuracy = [illegible]
Trojan.Downloader.Swizzor.gen!I : Accuracy = [illegible]
Worm.VB.AT : Accuracy = [illegible]
Trojan.Downloader.Wintrim.BX : Accuracy = [illegible]
Worm.Yuner.A : Accuracy = 1.0
```

Fig.12: GLCM_SVM algorithm

In the above screen the right side graph represents the confusion matrix of SVM and in left side displays the SVM overall precision, recall, and accuracy value of each malware family class.

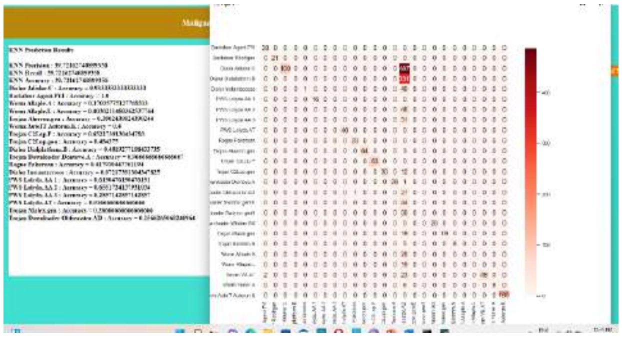


Fig.13: GLCM-KNN Algorithm

In above screen displaying KNN confusion matrix with overall accuracy and separate malware family accuracy

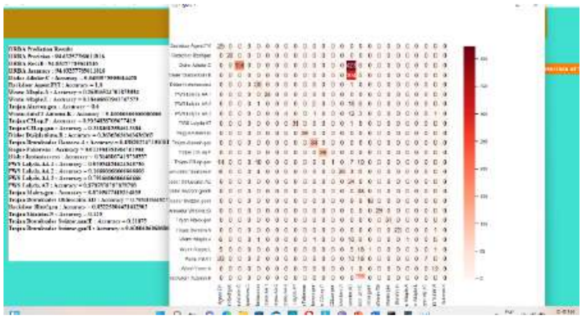


Fig.14: DBRA algorithm

In the above screen the DRBA algorithm has 99.83% accuracy which is higher than SVM and KNN and on the right side we can see a confusion matrix for each malware family. Now click on ‘Accuracy Graph’ button to get accuracy graph.

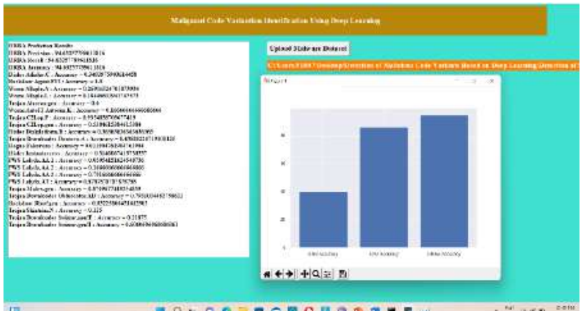


Fig.15: Accuracy graph

In above screen x-axis represents algorithm name and y-axis represents accuracy score and in all algorithm DRBA got high accuracy and now click on ‘Run Precision Graph’ button to get below graph

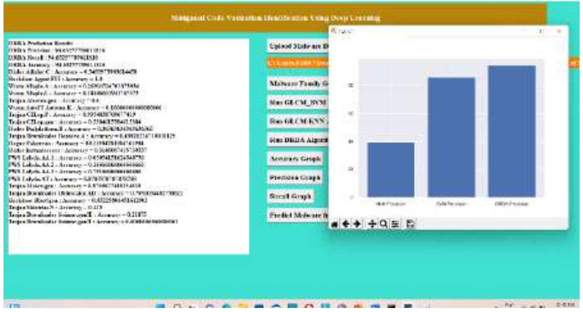


Fig.16: Precision graph

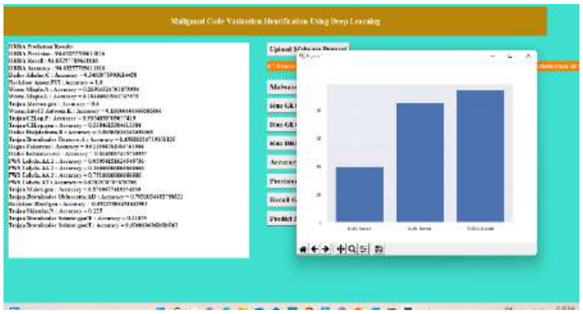


Fig.17: Recall graph

Now click on ‘Predict Malware from New File’ button to upload new malware file and then application will detect type of malware available in that file by applying DRBA CNN trained model.

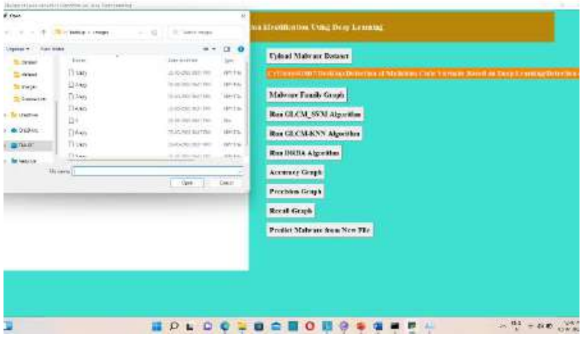


Fig.18: Predicting malware from New file.

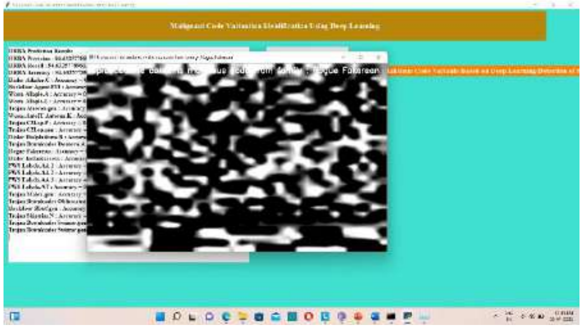


Fig 19: Malware Prediction

In above screen predicted malware name displaying on image and text area.

6. CONCLUSION

This paper proposed a novel method to improve the detection of malware variants through the application of deep learning. First, this method transformed the malicious code into grayscale images. Next, the images were identified and classified by a CNN that could extract the features of the malware images automatically. Because of the effectiveness and efficiency of the CNN for identifying malware images, the detection speed of our model was significantly faster than speeds achieved by other approaches. Our experimental results on 9342 grayscale images of 25 malware families showed that the proposed approach achieved 94.5% accuracy with good detection speed. In this study, the CNN framework required all input images to have a fixed size, which limited our model. In future work, we would like to use the SPP-net model to allow images of any size to be used as input. The SPP-net can extract features at variable scales, thanks to a spatial pyramid pooling layer. We can introduce that layer into our model between the last subsampling layer and the fully connected layer to improve our models flexibility. In addition, the transformation of malicious code into color images would be a good topic for future research.

REFERENCES

[1] J. Bouvrie, Notes on Convolutional Neural Networks, 2006.
[2] M. Christodorescu, S. Jha, S. A. Seshia, D. Song, and R. E. Bryant, “Semantics-aware malware detection,” in Proc. 2005 IEEE Symp. Security Privacy, 2005, pp. 32–46.
[3] Z. Cui, B. Sun, G. Wang, Y. Xue, and J. Chen, “A novel oriented cuckoo search algorithm to improve dv-hop performance for cyber–physical systems,” J. Parallel Distrib. Comput., vol. 103, pp. 42–52, 2017.
[4] O. E. David and N. S. Netanyahu, “Deepsign: Deep learning for automatic malware signature generation and classification,” in Proc. 2015 Int. Joint Conf. Neural Netw., 2015, pp. 1–8.
[5] D. D´ıaz-Pernil, A. Berciano, F. Pena-Cantillana, and M. A. Guti ˜ errez- ´ Naranjo, “Bio-inspired parallel computing of representative geometrical objects of holes of binary 2d-images,” Int. J. Bio-Inspired Comput., vol. 9, no. 2, pp. 77–92, 2017.
[6] H. Gao, Y. Du, and M. Diao, “Quantum-inspired glowworm swarm optimisation and its application,” Int. J. Comput. Sci. Math., vol. 8, no. 1, pp. 91–100, 2017.
[7] J. R. Goodall, H. Radwan, and L. Halseth, “Visual analysis of code security,” in Proc. 7th Int. Symp. Vis. Cyber Security, 2010, pp. 46–51.
[8] K. Han, J. H. Lim, and E. G. Im, “Malware analysis method using visualization of binary files,” in Proc. 2013 Res. Adapt. Convergent Syst., 2013, pp. 317–321.
[9] T. Isohara, K. Takemori, and A. Kubota, “Kernel-based behavior analysis for android malware detection,” in Proc. 2011 7th Int. Conf. Comput. Intell. Security, 2011, pp. 1011–1015.
[10] Y. Jia et al., “Caffe: Convolutional architecture for fast feature embedding,” in Proc. 22nd ACM Int. Conf. Multimedia, 2014, pp. 675–678.
[11] A. Khoshkbarforoushha, A. Khosravian, and R. Ranjan, “Elasticity management of streaming data analytics flows on clouds,” J. Comput. Syst. Sci., vol. 89, pp. 24–40, 2017.
[12] A. Khoshkbarforoushha, R. Ranjan, R. Gaire, E. Abbasnejad, L. Wang, and A. Y. Zomaya, “Distribution based workload modelling of continuous queries in clouds,” IEEE Trans. Emerging Topics Comput., vol. 5, no. 1, pp. 120–133, 2017.