University of North Carolina at Charlotte
ITCS-6156 Mid term Report
Fall 2019

SEMANTIC MALWARE DETECTION USING DEEP LEARNING

THE MEAN SQUARES

Collaborators:

Arjun Manevannan
Gopal Sharma
Muthu Priya Shanmugakani Velsamy
Smrithi Suriyanarayanan

Instructor:

Prof. Minwoo Lee Ph.D

Key Terms: *Static Analysis, Dynamic Analysis, Code Obfuscation, malware, bytecode, RNN, CNN, DanQ, LSTM, Micro Model*

I. Introduction

A malware is any software that is intentionally designed to cause damage to a computer, server, client, or computer network. The industry has come a long way from the days of signature based approach for identifying malwares. Detecting malwares using machine learning has been employed widely. We have made use of deep learning framework, specifically a combination of CNNs and RNN to tackle the issue.

Data Description

Here we have a set of known malware files representing a mix of 9 different families. Each malware file has an **ID**, a **20 character hash value** uniquely identifying the file, and a **Class label**, an integer representing one of 9 family names to which the malware may belong to **Ramnit, Lollipop, Kelihos_ver3, Vundo, Simda, Tracur, Kelihos_ver1, Obfuscator.ACY, Gatak.**

II. Literature Survey

[1] In the first stage the executables file is classified into malware using Static and Dynamic analysis and in second stage the malware executables file is categorized into corresponding malware family. In order to evaluate the performance of various classical machine learning classifiers, they have used the following methodologies, Adam optimizer and binary cross entropy are used as loss function. Batch normalization is used to prevent overfitting. Sigmoid is used in output layer which gives 0 or 1 as output where 0 is benign and 1 is malware. **[2]** Transform a malware/benign file into a grayscale image (Once the dataset is ready, we should convert each file into a 256x256 grayscale image). CNN approach is used since it uses input layer (convolutional layer), one or more hidden layers and output layer where the early layers are used to detect the low level features like edges, length, height, width, etc.of the images and finally all the layers are combined at the final stage to produce a valuable result. CNN will automatically extract features & classify data into various classes (Feature learning). **[3]** Static analysis is perceived as vulnerable to obfuscation and evasion attack. Here comes the emergence of dynamic analysis. One problem with dynamic analysis is that the time it takes to perform the analysis is high. Also if the malware data increases then simultaneously, dynamic analysis time also increases. Hence to improve the performance of the malware detection, behavior analysis is used. It was concluded that incremental technique in behavior-based analysis gives better performance in time and memory requirement than regular clustering. **[4]** Gray images are generated from the malware and benign files. For feature extraction and malware image classification DL approach (Convolutional Neural networks) is used. Using this approach they have got an accuracy of 96%. **[5]** Malware detection is done through static malware analysis through opcode sequence pattern of malware. They have used disassembly tool IDA Pro to obtain opcode sequence of malware. In the last 2 stages LSTM approach is used. This proposed method have got the accuracy of 0.99 average AUC. **[7]** They are using dynamic analysis approach, through behavioural analysis of the malware. Inorder to detect the malware in its early stage through dynamic analysis, they are taking snapshots of the malware behavior in the first 20 seconds of the malware execution.

Since the data (snapshots) they are collecting are based on time series sequence, they are using RNN methodology.
Based on our research, we will be using static analysis on the files as Dynamic analysis requires controlled environment to execute the files. This also requires deep domain knowledge to analyze the API calls made by the malicious file to the underlying operating system. Additionally, certain malwares are capable of evading the model during runtime.

III. Method

Accomplished Milestones

- **Feature Extraction:**

  The data obtained from Microsoft contains .bytes representation of the infected files. This representation is then converted into features through unigram bag of words to obtain the feature set from the given data. Since bytecode contains hex code from 00 to FF (256 values) we end with 256 dimensional data. We also add the size of the files as a feature as it was found useful for classification (observed through the EDA phase). This data is then normalized to bring the values between 0 and 1. Further malware samples belonging to classes other than the above mentioned ones have been used. These files are converted to their respective object code through a custom python script.

- **Classification Procedure:**

  Since we have 9 classes of malwares present, we are faced with a multi class classification problem. Once the files are vectorized and train-test split, we model classifiers through KNN, Logistic Regression, Random Forest, Xgboost. The models are created as micro models**[8],** which helps dispose and retrain the model for a specific classifier alone as opposed to retraining the whole model, which we see fit in the ever changing malware domain.

- **Performance Measure:**

The performance measure we have used to evaluate the models is multi class log loss to evaluate the models. This metric quantifies the accuracy of a classifier by penalising false classifications. Minimising the log loss is equivalent to maximising the accuracy of the classifier.

Detailed Approach: We planned to use a hybrid approach called DanQ which combine both Convolutional Neural Networks (CNN) and Recursive Neural Networks (RNN). The DanQ architecture is described below.

- CNN - It is used as the first layer of feature extraction from the images. Feature extraction includes grasping the patterns of the malware and this will be further utilised by the RNN layer for better prediction of the malware sequence patterns.
- Pooling Layer - This layer is used for dimensionality reduction in the malware images.
- RNN - Long Short Term Memory (LSTM) - This layer makes use of the previous layers outputs as its input and will predict the next occurence of the malware sequence. This will picturize the long term relationship between the malware signature patterns.

Novelty:

DanQ approach has been used for 'Predicting the DNA sequences'. From our research, we see similarities in predicting the DNA sequences with malware file type prediction. this approach has not been implemented for detecting/predicting malware. So our approach could be a new step towards malware detection when compared to all the existing technologies and solutions. As an overall architecture, DanQ provides us various layers of defense in predicting the malwares.

IV . Difficulties/Problems Faced

- **Data Gathering -** Since malware files are highly capable of causing havoc if ended in the wrong hands; thus getting hold of such data files, especially in huge numbers was a daunting task. This task was severely underestimated during planning phase and overshoot our initial deadlines by a large margin.

- **Computing Constraints -** Training multi class classification models takes a long time (over 3 days for .asm files) which makes it almost impossible to train with the resources at hand. We were also faced with storage constraints as the dataset easily exceeds half a terabyte when uncompressed.

V. Timeline

Milestones/Tasks	Estimated time	Work Assignment	Reason for Updation
Malware data collection	October 4 – October 8	Arjun, Gopal	Data gathering period is shortened. Additional data gathering is in progress.
Exploratory data analysis (EDA)	October 8 – October 14	Team work	On schedule
ML implementation	October 11 - October 18	Arjun, Smrithi	Model training has been on hold due to computing constraints. A request to education cluster has been placed.
DL exploration with malware	October 15 - October 25	Muthu Priya, Smrithi	On schedule.
DL implementation	October 20 - November 13	Arjun, Gopal	Worked in parallel with DL exploration. Work in progress
Experiments	November 7 - November 22	Team work	Expected to be completed by that time
Final Report	November 25 - December 5	Team work	Expected to be completed by that time
Project Poster	December 2 - December 6	Team work	Expected to be completed by that time

VI. Feedback Response

Annotation #	Question	Answer
2	Are you interested in a specific type of these or classifying all of them?	The types listed are part of our exploration list. We are specifically interested in classifying malware.
3	Is input to ML is the entire file?	Yes, we are providing the n-gram bag of words representation of the file's bytecode as input.
4	Citation for the mentioned statement "The time taken to reverse engineer gives way for the malicious code to encroach into the system"	Cited as [6] under references. "Zero-day Malware Detection based on Supervised Learning Algorithms of API call Signatures"
5	Citation for "Zero-day malware detection based on supervised learning algorithms of API call signatures"	Cited as [6] under references. "Zero-day Malware Detection based on Supervised Learning Algorithms of API call Signatures"
6	What are the different ML classifiers used in "Zero-day malware detection based on supervised learning algorithms of API call signatures"	Naive Bayes algorithm, k−Nearest Neighbor algorithm, Sequential Minimal Optimization algorithm, Backpropagation Neural Networks algorithm, and J48 decision tree
7	Citation for "Microsoft's Malware Classification Dataset"	https://www.kaggle.com/c/malware-classification/data
8	Any specific DL model you plan to use?	We are planning to use DanQ, which is a combination of CNNs and RNNs.
9	Are you going to adopt existing features or devise new ones?	We will be devising new features.
10, 11, 12	Timeline Updation	Made changes to our timeline according to the professor's suggestion
13	As commented above, you can focus on new feature models to make notable	Yes, we will be focussing on new feature models.

	difference.	
14	Do not implement your own. just use others to save time	We will be using the existing implementations.

VIII. References

1. R.Vinayakumar et al., “Robust Intelligent Malware Detection Using Deep Learning” in IEEE, 2019.
2. Ria Kulshrestha “Malware Detection Using Deep Learning (Malware Detection Using Convolutional Neural Networks)”, Retrieved from https://towardsdatascience.com/malware-detection-using-deep-learning-6c95dd235432
3. K. Rieck, P. Trinius, C. Willems, and T. Holz, “Automatic analysis of malware behavior using machine learning,” Journal of Computer Security, 2011.
4. Sunoh Choi Information Security Division ETRI Daejeon, South Korea ; Sungwook Jang ; Youngsoo Kim ; Jonghyun Kim, “Malware Detection using Malware images and Deep Learning,” in IEEE, 2017.
5. Renjie Lu, “Malware Detection with LSTM using Opcode Language”, arXiv:1906.04593v1 [cs.CR] 10 Jun 2019.
6. Mamoun Alazab, Sitalakshmi Venkataraman, Paul Watters, Moutaz Alazab, “Zero-day Malware Detection based on Supervised Learning Algorithms of API call Signatures” in Proceedings of the 9th Australian Data Mining Conference.
7. Matilda Rhode ,Pete Burnap and Kevin Jones, “Early-stage malware prediction using recurrent neural networks”, ELSEVIER, 2018.