DGA MALWARE DETECTION USING MACHINE LEARNING

[1]J Sreedevi, [2]Ponnam Pooja

Department of Computer Science and Engineering Mahatma Gandhi Institute of Technology, Hyderabad, India. [1]jsreedevi_cse@mgit.ac.in, [2]ponnampooja1@gmail.com

Abstract— **In today's world malware detection is one of the major problems in cyber security due to its effects to computer security. Threat actors are utilizing a domain generation algorithm (DGA), which can allow malware to communicate with C2 (Command and Control) by generating a variety of network locations. Prediction of domain names that are generated using the Domain Generation Algorithm (DGAs) is a challenging cyber security task in real time. Though there are some traditional malware control methods, such as blacklisting are insufficient to handle this DGA threats. In this study we specifically developed a machine learning based framework which involves classification and prediction of malware behavior. The proposed multi-layer deep learning model is used to tackle the problem of long-term memory, which gave more accuracy compared to other models developed in this work. Thus, the three algorithms are used for training and predictive malware analysis on multiple parameters with feature extraction, including error factor, accuracy rate, and overall performance. Result of the model from the evaluation measures provides a high accuracy rate and a lesser mean absolute error value. In the proposed work, various types of features are extracted from the database and it plays a major role in detecting the malware domains with high accuracy. The proposed work is to find the best binary classifier for DGA based malware detection.**

Keywords: Malware, Cyber Security, Domain Generation Algorithm, Machine Learning, feature extraction.

I.INTRODUCTION

Malware is malicious software that is specifically designed to disrupt, damage, or gain unauthorized access to computer system. This Malware can be classified into different types like virus, trojan horse, adware, spyware, etc. The scope of the malware harm could be anywhere from removing files, destroying software to reformatting the hard disk. Computer malware is a program that, when executed, infects a computer, poses a threat to the integrity of the system. As internet is growing widely, various domain names are generated regularly by botnets and malware using various DGAs. Because of large number of malware domains names that are generated by the DGAs, it is difficult to identify and blacklist them.

The Domain Generation Algorithm is in use because malware that depends on a fixed domain or IP address is quickly blocked which then hinders operations. So, rather than bringing out a new version of malware or setting everything up again at a new server, malware agents switches to a new domain at regular intervals using domain generation algorithm. In this approach, we explore machine learning algorithms including feature extractions, classification, clustering, and prediction techniques to understand those DGA domains. Some of the essential challenges of this research are

- To achieve good results on a malware data set in a real-world Environment.
- Propose a multi-model ensemble for problem to get very high accuracy for classification.
- To efficiently classify domains, using three types of features: statistical features and Structural features and linguistic features.
- To Control threat attacks.
- To accurately identify and cluster domains that originate from known DGA-based techniques.

Using real-time active malicious domains derived from DGAs on the public internet, measures the accuracy of the proposed approach, specifically, threat intelligence feeds collected from Bambenek Consulting over a period of one year were obtained through daily manual querying demonstrated trends of ongoing threats. Furthermore, the accuracy of these machine learning models to be improved by using feature selection algorithms to select the most essential features and reducing the size of the dataset which leads to lesser computations. As the size of domain name dataset increases a multi-layer model is used which tackles the problem of short-term memory.

OBJECTIVES

i) To carry out more in-depth analysis to understand the malware for better detection and predictability.

ii) To extract the features from various domain names from real world environment to reduce the size of the dataset for easy computation.
iii) To achieve comparatively good accuracy
iv) To analyze the different models

II. LITERATURE SURVEY

To differentiate DGA domain names from normal domain names, researchers have discovered that DGA-generated domain names contain significant features. Therefore, many studies aim to target blocking those DGA domain names as a defense approach. Existing traditional malware control methods, such as blacklisting, static string-matching approaches, hashing schemes are insufficient to handle DGA threats. Several researchers have been studying to bring a zero-day malware, some of this research papers are discussed below. Hynek, Karel, et al [1] focused on removing the false positive malicious domains i.e., by making use of publicly available blacklisted ip address and the servers communicating with them and whenever a malicious domain is found based on statistical comparison this study[1] confirms it as malicious network and gives an alert to the system. Li,Yi, et al [2]proposed machine learning framework which takes input of DNS queries and initially by using packet filter technique the identified as malicious domains are directly given as output otherwise a two-level model and a prediction model using SVM, Random forest and some other boosting algorithms are applied to identify malware behavior. Sethi, Kamalakanta, et al, [3] proposed a work where malware and clean files dataset is taken and a cuckoo sandbox is used to analyze the behavior of this files and in this model [3] some feature extraction steps are involved after that the machine learning algorithms such as SVM, logistic regression, random forest are applied to detect the malware files. Akarsh, S., et al, [4] introduced Long Short-term Memory architecture[4] which can handle domain names of large size which overcomes the problem of vanishing gradient descent by recurrent model, the input dataset is sent to lstm architecture[4] and this model is trained and tested as binary as well as multiclass classifier to identify the DGA domains. Yan, Dingkui, et al, [5] proposed a novel system, called Pontus, which includes training phase and classification phase to detect malware algorithm generated domains, this phases include common steps first the domains are filtered out by using dynamic filter from DNS traffic. Then feature extraction is performed which involves linguistic features to classify the domains. Yu, Bin, et al, [6] used the approach of convolutional neural network (CNN) and recurrent neural network (RNN) based architectures [6], it is faster to train and to score, and less prone to overfitting. Schüppen, Samuel, et al, [7] designed FANCI, a novel system for detecting infections with DGA based malware by monitoring non-existent domain (NXD) responses in DNS traffic but it gives high false positives. Ghafir, Ibrahim, and Vaclav Prenosil, [10] detection method is based on a blacklist of malicious IPs. The list of blacklisted IPs is automatically updated each day by checking communication between the blacklisted IPs and sends an alert to the system, this detection is in the real time.

III.SYSTEM DESIGN

A. ARCHITECTURE

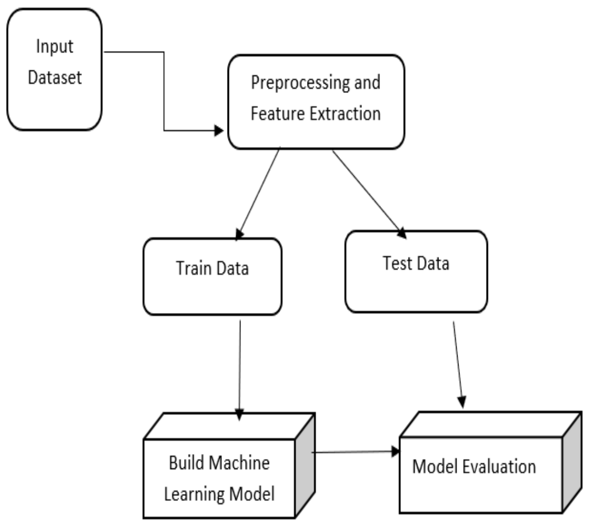

Input Dataset
Preprocessing and Feature Extraction
Train Data
Test Data
Build Machine Learning Model
Model Evaluation

Figure 1. System Architecture

This work involves the implementation of two machine learning models and one deep learning model to classify and predict the DGA domains.

The implementation of this models has a common sequence of events which are necessary.

- The dataset is pre-processed to remove unwanted and unnecessary data to make the model readable by the machine.
- Then, load the preprocessed DGA and Benign domain dataset to set the labels and combine them. Then the dataset is divided into train data and test data.
- Further, the feature extractor technique is used to extract features from the domain names filtered in the first component.
- Then, machine learning algorithms are trained using dataset, which is used to predict the accuracy by using test data.
- Once the model is built, then evaluate the by testing.

B. MODULES

- Data Preprocessing
- Feature Extraction
- Model Building

Data Preprocessing

The data is pre-processed to remove unwanted and unnecessary data. In this stage we are handling noise data by ignoring the data if the data is either duplicate or empty. Pre-processing refers to the transformations applied to our data before feeding it to the algorithm.

Data preprocessing is a technique that is used to convert the raw data into a machine understandable data set. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis.

Feature Extraction

The process of extracting data from the files is called feature extraction. The goal of feature extraction is to obtain a set of informative and non-redundant data. It is essential to understand that features should represent the important and relevant information about our dataset which requires a lot of testing and research. Moreover, it is very domain-specific, so general methods apply here poorly. The feature extractor is used to extract features from the domain names filtered in the first component. Each domain name is considered as a string. To efficiently classify domains, we use two types of features: linguistic features and DNS features.

Features of the DGA dataset are:

- NoS (Number of Subdomains): represents the number of subdomains (Ignore valid public suffixes).
- SLM (Subdomain Length Mean): represents the mean of subdomain length (Ignore valid public suffixes).
- HwP (Has www Prefix):
- HVTLD (Has a Valid Top-Level Domain):
- CSCS (Contains Single-Character Subdomain)
- CTS (Contains Top Level Domain as Subdomain): (Ignore valid public suffixes)
- UR (Underscore Ratio): Represents the ratio of underscore (Ignore valid public suffixes)
- CIPA (Contains IP Address): (Ignore valid public suffixes)
- Contains-digit (Contains digit): (Ignore valid public suffixes)
- Vowel-ratio (The ratio of vowel): (Ignore valid public suffixes)
- Digit-ratio (The ratio of digit): (Ignore valid public suffixes)
- RRC (The ratio of repeated characters in a subdomain): *(Ignore valid public suffixes)*
- RCC (The ratio of consecutive consonants): (Ignore valid public suffixes)
- RCD (The ratio of consecutive digits): (Ignore valid public suffixes)
- Entropy (The entropy of subdomain): (Ignore valid public suffixes)

Table 1. Structural features.

Features	Ex: prata.pt	Ex: tbaxcrnxirtmuusq.eu
DNL (Domain Name Length)	8	19
NoS (Number of Subdomains)	1	1
SLM (Subdomain Length Mean)	5.0	16.0
HwP (Has www Prefix)	0	0
HVTLD (Has a Valid Top Level Domain)	1	1
CSCS (Contains Single-Character Subdomain)	0	0
CTS (Contains Top Level Domain as Subdomain)	0	0
UR (Underscore Ratio)	0.0	0.0
CIPA (Contains IP Address)	0	0

From table 1, nine structural features are generated. For the example, prata.pt, DNL (The length of the domain name) is 8. It only has 1 subdomain, so its NoS value is 1. The length of the subdomain (SLM) is the length of 'prata', which equals to 5.0. It does not have www Prefix, so its Hwp value is 0. '.pt' is a valid top-level domain, so its HVTLD domain is 1. It does not contain single-character subdomain, so the CSCS value is 0. So does the CTS. The ratio of underscore (UR) for example is 0 also. And it does not have an IP address

Table 2. Linguistic features.

contains_digit (Contains digit)	0	0
Vowel_ratio (The ratio of vowel)	0.4	0.25
Digit_ratio (The ratio of vowel)	0.33	0.0

Based on linguistic analysis, three linguistic features are generated from the domain. Whether a domain contains a digit (contains-digit), the ratio of the vowel in a domain and the ratio of the digit. The value of these linguistic features can be known from Table 2.

Table 3. Statistical features.

RRC (The ratio of repeated characters in a subdomain)	0.25	0.33
RCC (The ratio of consecutive consonants)	0.4	0.625
RCD (The ratio of consecutive digits)	0	0
Entropy (The entropy of subdomain)	1.92	3.5

There are also 4 statistical features that will be generated. From Table 3, RRC represents the ratio of repeated characters in a subdomain. RCC represents the ratio of consecutive consonants, RCD represents the ratio of consecutive digits and Entropy means the entropy of sub-domain.

C.MODEL BUILDING

Machine Learning algorithms used in this work used Random Forest, logistic Regression and LSTM, these algorithms are trained on train dataset to detect the DGA domains. Here before building the model the data is splitted into 80% train data and 20% test data. The models are built on train data with relevant features. Once the models are built the accuracy of these models are evaluated based on test dataset.

CLASSIFICATION ALGORITHMS

A. Random Forest (RF)

Random forest is a bunch of decision trees. It is an ensemble model. A random forest model will take all predicting results from its inner decision trees as a vote.

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.

B. Logistic Regression

Logistic Regression is a technique to analyse a dataset which has a dependent variable and one or more independent variables to predict the outcome in a binary variable, meaning it will have only two outcomes. The dependent variable is categorical in nature. Dependent variable is also referred as target variable and the independent variables are called the predictors. Logistic regression is a special case of linear regression where we only predict the outcome in a categorical variable. It predicts the probability of the event using the log function. A Sigmoid function/curve is used to predict the categorical value. The threshold value decides the outcome.

C. LSTM (Long Short-Term Memory Neural Network)

Before analyzing LSTM, understand the problem of short-term memory, Recurrent Neural Networks suffer from short-term memory. If a sequence is long enough, it is hard in carrying information from earlier time steps to later ones. So, if you are trying to process a paragraph of text to do predictions, RNN's may leave out important information from the beginning. During back propagation, recurrent neural networks suffer from the vanishing gradient problem. Gradients are values used to update a neural networks weight. The vanishing gradient problem is when the gradient shrinks as it back propagates through time. If a gradient value becomes extremely small, it doesn't contribute too much learning. So, in recurrent neural networks, layers that get a small gradient update stops learning. Those are usually the earlier layers. If these layers don't learn, RNN's can forget what it seen in longer sequences, thus having a short-term memory.

For example, LSTM is an application to tasks such as unsegmented, connected handwriting recognition, or speech recognition. A general LSTM unit is composed of a cell, an input gate, an output gate, and a forget gate. The cell remembers values over arbitrary time intervals, and three gates regulate the flow of information into and out of the cell. LSTM is well-suited to classify, process, and predict the time series given of unknown duration.

The input gate is responsible for the addition of information to the cell state. This job of selecting useful information from the current cell state and showing it out as an output is done via the output gate. A forget gate is responsible for removing information from the cell state, the information that is no longer required for the LSTM to understand things or the information that is of less importance is removed via multiplication of a filter. This is required for optimizing the performance of the LSTM network. The detailed architecture of LSTM cell and it's performance is depicted in figure: 2.

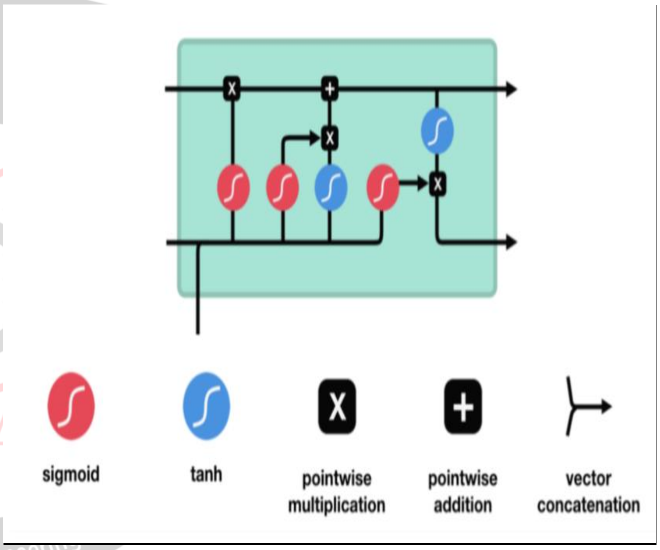


Figure. 2. LSTM cell and it's operation

The first step in LSTM is to decide what information should be thrown away from the cell state. This decision is made by a sigmoid layer called the "forget gate layer". The next step is to decide what new information is to store in the cell state. This has two parts. First, a sigmoid layer called the input gate layer decides which values to update. Next, a tanh layer creates a vector of new candidate values, that could be added to the state. In the figure 2, each line carries an entire vector, from the output of one node to the inputs of others. The pointwise operations represents vector addition. Lines merging denote concatenation, while a line forking denotes its content being copied and the copies going to different locations.

IV.TRAINING AND VALIDATION

In this study, after preprocessing of the data which includes removing valid public suffixes from the domain, a python module was developed for feature extraction, feature selection and generation of training and testing dataset. This module extracts the relevant features from the domain names which are useful for detection and classification. After that unnecessary column are dropped from the dataset generated after feature extraction.

A. TRAINING

In our proposed work, Scikit-Learn library was used for the detection and classification of malware. A Binary Classifier model developed by Scikit learn is used for detecting a given sample as malware or benign using its feature vector of data set. The two datasets generated using feature extraction and feature selection are called Marco dataset and Micro dataset. Macro dataset is used for detection purpose and Micro dataset is for classification purpose. These datasets are divided into training and testing sub datasets in 80%:20% ratio for training and testing of model respectively. The training dataset is used to produce Model using machine learning algorithms (Logistic Regression, Random Forest, LSTM) using the Scikit learn library.

B. TESTING

In this section, the performance of proposed malware detection framework is analyzed with extensive experimental results using 1 million malicious samples and 1 million benign samples for this experiment. Scikit-Learn library produced results with detailed accuracy of each class and confusion matrix for binary classifier. For malware analysis, the dataset is divided into training and testing set. The training set contains 80% of malware samples, and the testing set contains 20% of malware samples. The results are obtained for detection and classification using the two models developed from each of the machine learning algorithms: (Logistic Regression, Random Forest) and one deep learning algorithm:(Long Short-Term Memory).

The results of the experimental evaluation are explained in the following subsections. In Result Analysis four different performance metrics are used i.e., Accuracy, Precision, Recall and F-measure to evaluate the performance of proposed malware analysis framework. This performance metrics can be expressed in terms of TP (True Positive), FP (False Positive), TN (True Negative) and FN (False Negative).

- True Positive (TP): It represents the amount of domain names that are correctly predicted as malicious.
- True Negative (TN): It represents the amount of domain names that are correctly predicted as benign.
- False Positive (FP): It represents the amount of domain names that are benign but incorrectly predicted as malicious.
- False Negative (FN): It represents the amount of domain names that are malicious but incorrectly predicted as benign.

Accuracy:

It represents the amount of correct predictions (TP and TN) over all kind of predictions.

Accuracy= (TP+TN)/TP +FP+TN+FN

Precision:

The precision is the ratio TP /(TP+FP) where TP is the number of true positives and FP is the number of false positives. The precision is the ability of the classifier not to label as positive a sample that is negative.

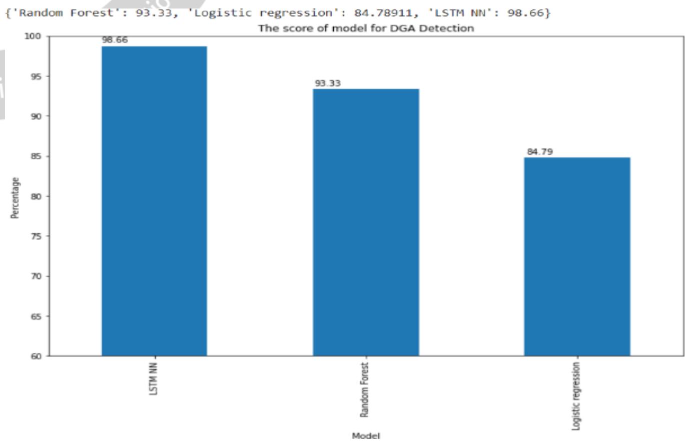
Recall: The recall is the ratio **TP/(TP+FN)** where **TP** is the number of true positives and **FN** is the number of false negatives. The recall is the ability of the classifier to find all the positive samples.

F1-Score: In statistical analysis of binary classification, the F-score or F-measure is a measure of a test's accuracy. It is calculated from the precision and recall of the test. The F-score is the harmonic mean of the precision and recall. F1-Score= 2* (recall*precision)/ (recall +precision)

V. RESULTS ANALYSIS

Malware detection results with various classifiers used in this study. It can be observed from Figure 4.4 that LSTM gives a high detection rate with an accuracy of 98.66% with 16 selected features. This accuracy is highest among those of other classifiers. Similarly, we achieved high value for Precision, Recall and F-Measure using Decision tree compared to Logistic Regression. These metrics are highest in comparison to logistic regression classifier used in this work. The detailed comparison of this Model is shown in the Graph 1.

Graph 1. Graphical comparison of various models

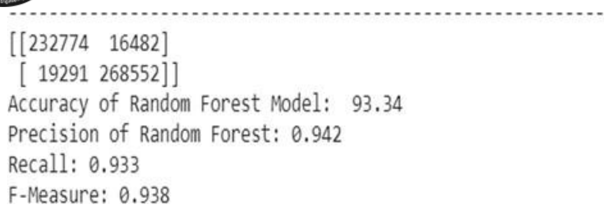


```
{'Random Forest': 93.33, 'Logistic regression': 84.78911, 'LSTM NN': 98.66}
```

Graph 1. shows the accuracy of different Algorithms in which LSTM shows highest accuracy.

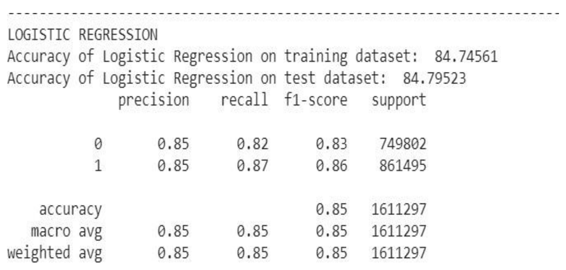
Figure 4. Random Forest accuracy

Figure 4 gives the information of random forest algorithm which gave accuracy of 93.34%, precision of 0.942, recall of 0.933 and F-Measure of 0.938.



```
------------------------------------------------------------
[[232774  16482]
 [ 19291 268552]]
Accuracy of Random Forest Model:  93.34
Precision of Random Forest: 0.942
Recall: 0.933
F-Measure: 0.938
```

Figure 5. Logistic regression accuracy



```
------------------------------------------------------------------
LOGISTIC REGRESSION
Accuracy of Logistic Regression on training dataset:  84.74561
Accuracy of Logistic Regression on test dataset:  84.79523
              precision    recall  f1-score   support

           0       0.85      0.82      0.83    749802
           1       0.85      0.87      0.86    861495

    accuracy                           0.85   1611297
   macro avg       0.85      0.85      0.85   1611297
weighted avg       0.85      0.85      0.85   1611297
```

Figure 5. gives the information of logistic regression algorithm which gives accuracy of 84.79%, precision of 0.85, recall of 0.87 and F-Measure of 0.85.

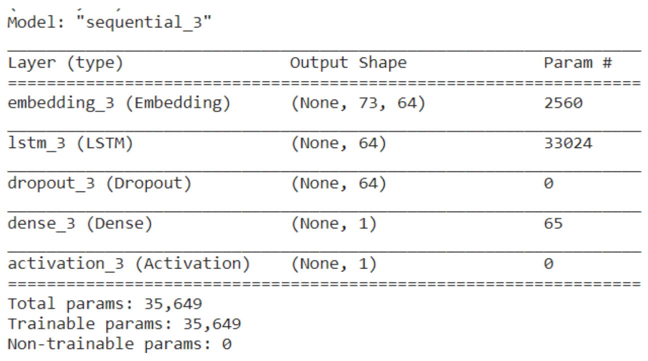
LONG SHORT-TERM MEMORY

A. A. Embedding

Keras has an embedding layer which maps character level domain names to the dense vector representations. Keras embedding layer addition maps the inputs together closely. The embedding is also learned separately during the training.

B. Configuration of Hyper parameters for Binary Class LSTM network consisted of 128 memory units. A dropout of 0.5 is used to avoid the overfitting of the model and removing some of the weights randomly results in regularization and better generalization of the data. The sigmoid nonlinear activation function was used for obtaining the binary class probabilities. Batch size of 128 was used and the model was trained for 2 epochs. All the hyper parameters were selected after repeated trails with various configurations and optimal values were obtained and detailed model summary is specified in figure 6.

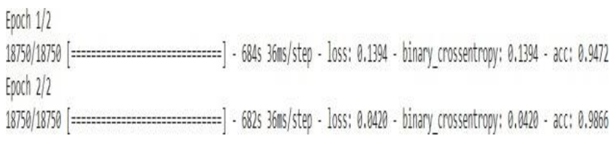
Figure 6. Binary class LSTM model summary



```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_3 (Embedding)      (None, 73, 64)            2560
_________________________________________________________________
lstm_3 (LSTM)                (None, 64)                33024
_________________________________________________________________
dropout_3 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 65
_________________________________________________________________
activation_3 (Activation)    (None, 1)                 0
=================================================================
Total params: 35,649
Trainable params: 35,649
Non-trainable params: 0
```

In LSTM model, it's cell uses both sigmoid and tanh activation functions internally and dropout on the input means that for a given probability, the data on the input connection to each LSTM block will be excluded from node activation and weight updates

Figure 7. LSTM model epochs results



```
Epoch 1/2
18750/18750 [==============================] - 684s 36ms/step - loss: 0.1394 - binary_crossentropy: 0.1394 - acc: 0.9472
Epoch 2/2
18750/18750 [==============================] - 682s 36ms/step - loss: 0.0420 - binary_crossentropy: 0.0420 - acc: 0.9866
```

Figure 7, gives the information of LSTM algorithm which gives accuracy of 98.66%, loss -0.042, binary cross entropy -0.042.

VI.CONCLUSION AND FUTURE SCOPE

Detecting DGAs is a grand challenge in security areas. DGAs are usually used by an attacker to communicate with a variety of servers. However, Blacklisting is good for handling static methods. DGAs are dynamic, so simply using the blacklisting is not enough for detecting a DGA. In this study, two machine learning models and a deep learning model was proposed to handle DGA threats. The proposed machine learning framework consists of a feature extractor and a machine learning model for classification and prediction of domains using real-time threat intelligence database fed over a one-year period where all domains live threats on the Internet. As the size of the data collected becomes larger and larger, a deep learning model is built to perform the classification, which has a better performance than the machine learning algorithms. Based on this study with the real-world feed, the proposed framework can effectively extract domain name features as well as classify and detect domain names to which it belongs.

This study proposes Machine learning models and LSTM model for the real time prediction of the malicious domain names with a reasonable accuracy for binary class. In LSTM model the domain names are converted into vectors by creating the dictionary of domain characters with Keras embedding. For future work, it would be interesting to analyze by training with even complicated models and by adding more number of layers for better accuracy. Testing other preprocessing techniques, embedding ,fine-tuning the hyper parameters and increasing the number of epochs may further improve the accuracy of the model. It uses a supervised learning approach and for preparing the training data set, data must be labelled manually.

REFERENCES

[1] Hynek, Karel, et al. "Evaluating Bad Hosts Using Adaptive Blacklist Filter." 2020 9th Mediterranean Conference on Embedded Computing (MECO). IEEE, 2020.

[2] Li, Yi, et al. "A machine learning framework for domain generation algorithm-based malware detection." IEEE Access 7 (2019): 32765-32782.

[3] Sethi, Kamalakanta, et al. "A novel machine learning based malware detection and classification framework." 2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security). IEEE, 2019.

[4] Akarsh, S., et al. "Deep learning framework for domain generation algorithms prediction using long short-term memory." 2019 5th International Conference on Advanced

Computing & Communication Systems (ICACCS). IEEE, 2019.

[5] Yan, Dingkui, et al. "Pontus: A Linguistics-Based DGA Detection System." 2019 IEEE Global Communications Conference (GLOBECOM). IEEE, 2019.

[6] Yu, Bin, et al. "Character level-based detection of DGA domain names." 2018 International Joint Conference on Neural Networks (IJCNN). IEEE, 2018.

[7] Schüppen, Samuel, et al. "{FANCI}: Feature-based automated nxdomain classification and intelligence." 27th {USENIX} Security Symposium ({USENIX} Security 18). 2018.

[8] Chin, Tommy, Kaiqi Xiong, and Mohamed Rahouti. "SDN-based kernel modular countermeasure for intrusion detection." International Conference on Security and Privacy in Communication Systems. Springer, Cham, 2017.

[9] Ghosh, Uttam, et al. "An SDN based framework for guaranteeing security and performance in information-centric cloud networks." 2017 IEEE 10th International Conference on Cloud Computing (CLOUD). IEEE, 2017.

[10] Ghafir, Ibrahim, and Vaclav Prenosil. "Blacklist-based malicious ip traffic detection." 2015 Global Conference on Communication Technologies (GCCT). IEEE, 2015.

[11] Khancome, Chouvalit, Veera Boonjing, and Pisit Chanvarasuth. "A two-hashing table multiple string pattern matching algorithm." 2013 10th International Conference on Information Technology: New Generations. IEEE, 2013.

[12] Langner, Ralph. "Stuxnet: Dissecting a cyberwarfare weapon." IEEE Security & Privacy 9.3 (2011): 49-51.

[13] Yamada, Ryo, and Shigeki Goto. "Using abnormal TTL values to detect malicious IP packets." Proceedings of the Asia-Pacific Advanced Network 34 (2013): 27-34.

[14] Felegyhazi, Mark, Christian Kreibich, and Vern Paxson. "On the Potential of Proactive Domain Blacklisting." LEET 10 (2010): 6-6.

[15] Ma, Justin, et al. "Beyond blacklists: learning to detect malicious web sites from suspicious URLs." Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. 2009.

[16] Rieck, Konrad, et al. "Learning and classification of malware behavior." International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment. Springer, Berlin, Heidelberg, 2008.