Empirical Study on Malicious URL Detection Using Machine Learning

Ripon Patgiri(✉), Hemanth Katari(✉), Ronit Kumar(✉), and Dheeraj Sharma(✉)

National Institute of Technology Silchar, Silchar 788010, Assam, India
ripon@cse.nits.ac.in, hemanth.katari@gmail.com, ronit.kumar1194@gmail.com, dheerajsharma.nits@gmail.com

Abstract. In this paper, the malicious URLs detection is treated as a binary classification problem and performance of several well-known classifiers are tested with test data. The algorithms Random Forests and support Vector Machine (SVM) are studied in particular which attain a high accuracy. These algorithms are used for training the dataset for classification of good and bad URLs. The dataset of URLs is divided into training and test data in 60:40, 70:30 and 80:20 ratios. Accuracy of Random Forests and SVMs is calculated for several iterations for each split ratio. According to the results, the split ratio 80:20 is observed as more accurate split and average accuracy of Random Forests is more than SVMs. SVM is observed to be more fluctuating than Random Forests in accuracy.

Keywords: Malicious URL detection · Network security
Machine Learning · Random Forest · Suport vector machine · SVM

1 Introduction

In today's world, there is a rapid advancement in technology. With the advancement of technology, there is a similar development in the Internet. Internet involvement in social and business fields is increasing in large scale. The increasing use of the internet for such purposes increases the scope for cyber criminal activities. As the connectivity and the number of users grow, there is a proportional increase in attackers. The Government, industry and individuals are the victims. It is a difficult task to predict the future threats and their nature, and practically unsolvable. Malware or malicious websites become one of the major threat for cyber security. Whereas malicious URLs, in particular, becomes a serious threat of cyber security. Malicious URL is a common and serious threat to cyber-security. Malicious URLs host content abnormalities, such as spamming, phishing attacks, exploiting users, etc. They allow unsuspected users as victims of attacks by drivers. They incur huge monetary loss of billions of dollars every year worldwide. It is very important to firstly detect and act on such attacks

G. Fahrnberger et al. (Eds.): ICDCIT 2019, LNCS 11319, pp. 380–388, 2019.
https://doi.org/10.1007/978-3-030-05366-6_31

frequently for security [5]. Generally, such detection's are done through the use of big blacklists [5]. In practice, it is not possible to have exhaustive blacklists [5]. Today's naive implementation of detection techniques is insufficient to address billions of URLs encountered in everyday life. Machine Learning techniques is used to address the problem as a binary classification problem in large scale [7]. There are various classifiers in Machine Learning which give high accuracy in classification of good and bad URLs [1,2,8]. Moreover, Huang et al. detects Malicious URL using a greedy selection algorithm [3]. Similarly, Liu et al. also provides experimental study on URL detection using Machine Learning algorithms [4]. Vu et al. performs cost-sensitive malicious URL detection using a Decision Tree algorithm [9]. In this empirical study, we perform- (a) collecting a dataset which consists of huge number of URL's which consists of both malicious and non malicious URLs, (b) divide the collected dataset into two subsets in the ratio of 80:20 for training purposes and testing purposes, (c) extract features from the training data categorized into lexical features, network based features and host based features, (d) training the system using the training data and Machine Learning algorithms like Random Forest algorithm and Support Vector Machines (SVM), and (e) testing the system by providing test data and calculating the accuracy using each of the algorithms. Our aim is to provide a comprehensive investigation on detection of Malicious URLs by using Machine Learning algorithms like SVM and Random Forest classification algorithm.

2 Proposed System

Our proposed system uses Machine Learning algorithms and analyzes using the various features obtained from the URL for classification purpose. Figure 1 represents the complete flow diagram of our proposed system which consists of the following phases- collection of data, features extraction, training model, testing the model, query phase and the final output phase. It also represents the source of data collection, extracted features and the models used for classification purposes and the output consisting of accuracy and classification result.

Data: Collection of reliable and informative dataset is a very important aspect in dealing with learning based problems for classification or regression. The data consisting of both malicious and non-malicious URLs with labels need to be collected for training and testing purpose from a reliable source in order to get better accuracy and classification result.

Feature Selection and Extraction: The selection of features is an important and difficult phase where the dataset in hand is very big. This makes detection of patterns and finding a correlation among features too heavy for computation. In Machine Learning, a feature is an individual measurable property or characteristic or an attribute of a phenomenon being observed. Choosing informative, differentiating and independent features is a vital step for efficient algorithms

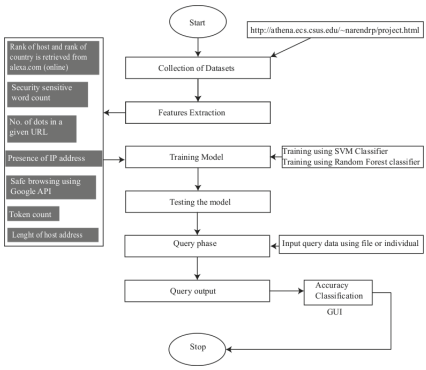


Fig. 1. Flow diagram representing the proposed solution

in pattern recognition, classification and regression purposes. Variable selection and attribute selection are referred for feature extraction. It includes the process of selecting selective features which are important for model training for classification. The problem requires to classify among the URLs as Benign or Malicious. So, in order to address the problem, we would require to design a model and train it using the features extracted from the training data. The next phase after the collection of data is extracting useful and informative features which are sufficient for the description of URLs and which can be mathematically interpreted for training using Machine Learning models. Simply, using an URL will not directly allow good classification method. So, it is important to select suitable features based on some rules or hypothesis to obtain a good feature from the set of URL. Thus, the quality of the extracted features from the URLs is prominent for the quality of the resulting malicious URL classification model. The features for classifying an URL can be of many types which can be classified into lexical features, host based features and web content features.

Lexical Features: lexical features are the features obtained based on the characteristics of the URL name or URL String. The most commonly used lexical features include statistical properties of the URL string, namely, (a) the length of the URL, (b) length of host name, (c) number of characters consisting of the host name, (d) length of path, (e) domain token count, (f) path token count, (g) average domain token length, (h) average path token length, (i) count of Security sensitive words, (j) number of dots, (k) presence of IP address, and (l) presence of .exe in URL.

Host Based Features: Host-based features are obtained from the host-name properties of the URL. The features include - safe browsing, rank of host and Country, and site popularity. Safe browsing is a Google service that lets client applications check URLs against Google's constantly updated lists of unsafe web resources. Rank of host and Country is a location based feature corresponding to

an IP address corresponding to the URL. The Location information comprises the physical Geographic Location - e.g. country/city to which the IP address belongs. "Site popularity" is one of the prominent feature for URL classification. Site Popularity measured as the increase in traffic of web using competitive analytical methods. It is generally estimated by counting the number of incoming links from other web pages to these web pages. Link popularity refers to the number of backlinks (incoming links) that points to a given URL. It can be considered as a reputation measure of an URL. Malicious sites tend to have a lower value of link popularity, whereas many non-malicious URL, especially the popular ones, tends to have a higher value of link popularity. Both the link popularity of a URL and the link popularity of the URL's domain are used in our method. Link popularity (LPOP) can be easily obtained from a search engine. In our proposed solution, we have used traffic information obtained from Alexa.com. Alexa's traffic estimates and ranks are based on the browsing behavior of the people in our global data panel, which is a sample of all internet users.

Classification: Random Forest is a supervised classification algorithm. Random Forest can be used for regression and classification tasks. Random Forest classifier creates a set of decision trees from randomly selected features. Then, it calculates votes from different the decision tree for each predicted target and the highest voted class is considered the final prediction. The random-forest algorithm brings extra randomness into the model while growing the trees. Instead of searching best feature while splitting node, it searches for the best features among random subset of features. Similarly, Support Vector Machine (SVM) is a supervised classification algorithm. SVM is a discriminating classifier that separates defined by separating hyper-plane. More clearly, SVM takes training data and separates data into categories divided by a clear gap called the hyper-plane. SVM tries to find out the best or optimal hyper-plane, which has the largest distance from the nearest point, in high dimensions, which clearly separates training set into categories. Support vectors are the data points nearest to the hyper-plane. The goal is to choose a hyper-plane with the greatest possible margin between the hyper-plane and any point within the training set, giving a greater chance of new data being classified correctly.

Method Comparison: In order to evaluate the models, cross validation score is used to measure the accuracy. Accuracy is the overall success rate of the method in terms of predictions. The models Random Forest and SVM are compared using minimum, maximum and average accuracy's of test data.

3 Experimental Results and Discussions

The labeled data collected consists of malicious and benign URL undergoes the feature extraction process, and then, the data are divided into two, particularly, the training and the testing data [6]. The training data are passed through

various methods for feature extraction and labelling the training data, and then, it is passed to various models such as Random Forest and SVM. And, the trained model is tested using the training dataset, i.e., one having features without labels and we calculate the accuracy of the model. The above process is repeated for 3 data splits, i.e., 80:20, 70:30, and 60:40, the antecedent being the training dataset and consequent being tested dataset. The Fig. 2 represents the experimental flow diagram. The dataset contain both Benign and Malicious URLs. A Uniform Resource Locator (URL) consists of protocol, domain name and path. From the URL, various lexical features, host-based features and site popularity features are extracted. A dataset containing Benign and Malicious URLs is collected from the following Source [6].

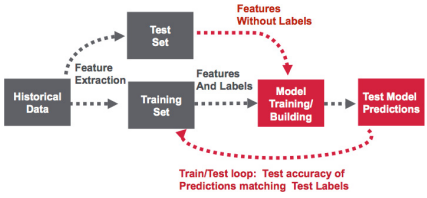


Fig. 2. Experimental flow diagram

3.1 Feature Selection and Extraction

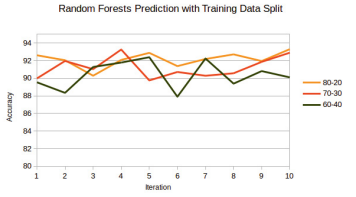
Lexical Features: In order to extract these lexical features from URLs, the URL is first broken into a set of words or Tokens according to the delimiters (‘.’, ‘/’, ‘?’, ‘=’, ‘-’) in the URL. It is also very important to have the distinction between the tokens belonging to the host name, the path, the top-level domain and the primary domain name. **Tokenise(URL)** is the method used in implementation to generate tokens from an input URL string. This method takes the URL String as input and returns average token length, token count and largest token’s length. Length of the URL is obtained by accessing the URL which is passed as an argument to **Feature_Extraction(URL)** method in the form of string and calculating the length of the string directly. After obtaining the tokens from the URL, we can similarly obtain other features by counting the number of tokens in the domain and path, and calculating the string length in other cases. The bag of words model is employed in order to count the number of security sensitive words. In this process, a security sensitive bag of words is created and stored in the form of an array and the collected tokens are checked if any of them matches with any of words present in the array, and if present, we can increase the counter for security sensitive words and have the final count. The security sensitive keywords consists of words, namely, ‘confirm’, ‘account’, ‘banking’, ‘secure’, ‘ebayisapi’, ‘webscr’, ‘login’, ‘signin’ and so on. This method takes in an array of words or tokens as input and returns an integer representing the count of security sensitive words. Number of dots is obtained by using a

string matching algorithm. Ip Address presence can be determined by making use of regular expression. It is also checked if the URL has '.exe' extension using string matching algorithm.

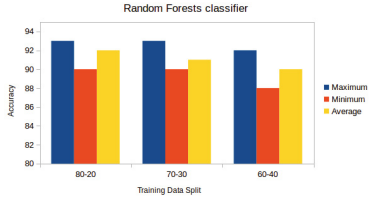
Host Based Feature: Safe browsing information can be obtained by passing the URL as a request to safe browsing API of Google. The response is obtained in the form of a code which represent unique information about the URL passed as request. The response to the URL passed as a request to an online website Alexa.com is obtained which provides us with the rank of country which hosts the particular ip address. It is a fact that mostly malicious sites are registered in less reputable hosting centers or regions.

3.2 Training the Models for Classification

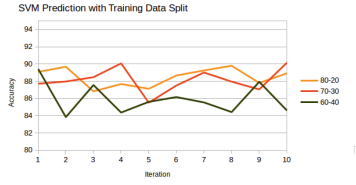
The data, which is in CSV format, are read using pandas package of python. The non-string columns from training data are extracted. The data are sent to Random Forests and SVM. Python consists a sklearn package which is used for Machine Learning applications. The Sklearn.ensemble package contains Random Forest Classifier module. Number of Trees is passed as an argument to Random



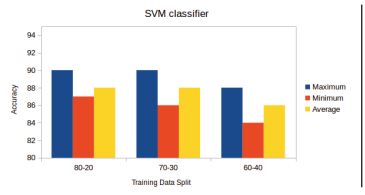
(a) Accuracy obtained in each iteration for different division ratio of data using Random Forest classifier



(b) Minimum,Maximum and Average accuracy obtained using Random Forest classifier at different split ratio of dataset



(c) Accuracy obtained in each iteration for different division ratio of data using SVM classifier



(d) Minimum, Maximum and Average accuracy obtained using SVM classifier at different split ratio of dataset

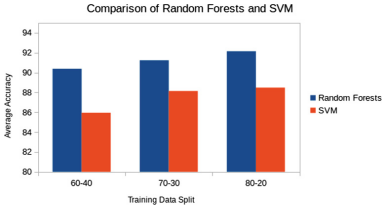
Fig. 3. Accuracy evaluation using difference Machine Learning algorithm.

Forests function. The model is trained using fit function which takes training data with output after feature extraction as arguments. The accuracy of trained model is tested using test data using **cross_val_score** function cross validation module. The test data and predicted values with accuracy is printed as output.

The accuracy of Random Forests is calculated for training data split ratios of 60:40, 70:30 and 80:20 for 10 iterations. For each iteration, the corresponding accuracy of each split ratio is depicted in Fig. 3a. It is clearly depicted that the Random Forests model varies from 88 to 92% in accuracy. The variance is smaller compared to SVMs. A histogram is also depicted in Fig. 3b for comparison of different ratios 60:40, 70:30 and 80:20 for minimum, maximum and average accuracy's. The average accuracy is calculated as the average of accuracy of 10 iterations. The minimum and maximum values are the smallest and biggest value in 10 iterations respectively.

3.3 Training with SVMs

The sklearn package contains SVM module. The function Support Vector Classification (SVC) is invoked for SVM object. The model is trained using fit function which takes training data, and outputs after feature extraction. The accuracy



(a) Comparison of Average accuracy obtained using both the models at different split ratio

Iteration	Random Forest classifier	SVM classifier
1	0.89544	0.89415
2	0.88343	0.83842
3	0.91282	0.87556
4	0.91783	0.84368
5	0.92389	0.85610
6	0.92242	0.85542
7	0.89392	0.84423
8	0.92242	0.85542
9	0.90815	0.87933
10	0.90101	0.84610

(b) Accuracy for each iteration using both the classifiers at 60:40 split ratio

Iteration	Random Forest classifier	SVM classifier
1	0.92610	0.89080
2	0.91964	0.87967
3	0.91033	0.88464
4	0.93262	0.90067
5	0.89757	0.85514
6	0.90714	0.87504
7	0.90285	0.89010
8	0.90567	0.87949
9	0.91862	0.87051
10	0.92902	0.90148

(c) Accuracy for each iteration using both the classifiers at 70:30 split ratio

Iteration	Random Forest classifier	SVM classifier
1	0.89985	0.87714
2	0.92048	0.89677
3	0.90291	0.86825
4	0.92086	0.87673
5	0.92885	0.87125
6	0.91379	0.88661
7	0.92188	0.89247
8	0.92714	0.89793
9	0.91953	0.87801
10	0.93302	0.88919

(d) Accuracy for each iteration using both the classifiers at 80:20 split ratio

Fig. 4. Accuracy evaluation.

of trained model is tested using test data using **cross_val_score** function cross validation module. The test data and predicted values with accuracy is printed as output. The accuracy of SVM is calculated for training data split ratios of 60:40, 70:30 and 80:20 for 10 iterations. For each iteration the corresponding accuracy of each split ratio is depicted in Fig. 3c. It is clearly plotted that the Random Forests model varies from 82 to 90% in accuracy. The variance is more compared to SVM. The histogram is also plotted in Fig. 3d for comparison of different ratios 60:40, 70:30 and 80:20 for minimum, maximum and average accuracy.

3.4 Comparison of Models

The training models, Random Forests and SVM, are compared with their average accuracy of 10 iterations in plotted histogram in Fig. 4a. From the three split ratios 60:40, 70:30 and 80:20, the average accuracy of SVM is less compared to Random Forest Classifier. The accuracy of Random Forests and SVM are listed in the given in Figs. 4b, c, and d. The three figures consist of accuracy of models with training splits 80:20, 70:30 and 60:40 ratios for 10 iterations. The accuracy values are listed on a scale of 0 to 1. From the comparison, Random Forest gives more accuracy than SVM. SVM accuracy fluctuates more than Random Forests with iterations.

4 Conclusion

Malicious Web sites are the basis of most of the criminal activities over the internet. The dangers that arise due to the malicious sites are enormous and the end-users must be prohibited from visiting such sites. The users should prohibit themselves from clicking on such Uniform Resource Locator (URL). The detection of malicious URLs is a binary classification problem and several Machine Learning Algorithms, namely Random Forests, SVMs and Naive Bayes are implemented on training dataset. Also, it has been seen that the Random Forest classifier performs better for the particular problem than the SVM classifier.

References

1. Choi, H., Zhu, B.B., Lee, H.: Detecting malicious web links and identifying their attack types. WebApps **11**, 11 (2011)
2. Gabriel, A.D., Gavrilut, D.T., Alexandru, B.I., Stefan, P.A.: Detecting malicious URLs: a semi-supervised machine learning system approach. In: 2016 18th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), pp. 233–239. IEEE (2016)
3. Huang, D., Xu, K., Pei, J.: Malicious URL detection by dynamically mining patterns without pre-defined elements. World Wide Web **17**(6), 1375–1394 (2014)

4. Liu, C., Wang, L., Lang, B., Zhou, Y.: Finding effective classifier for malicious URL detection. In: Proceedings of the 2018 2nd International Conference on Management Engineering, Software Engineering and Service Sciences, ICMSS 2018, pp. 240–244. ACM, New York (2018)
5. Ma, J., Saul, L.K., Savage, S., Voelker, G.M.: Learning to detect malicious URLs. ACM Trans. Intell. Syst. Technol. (TIST) **2**(3), 30 (2011)
6. Narendra, P.: Malicious URL detection. http://athena.ecs.csus.edu/narendrp/project.html
7. Vanhoenshoven, F., Nápoles, G., Falcon, R., Vanhoof, K., Köppen, M.: Detecting malicious URLs using machine learning techniques. In: 2016 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1–8. IEEE (2016)
8. Verma, R., Das, A.: What's in a URL: fast feature extraction and malicious URL detection. In: Proceedings of the 3rd ACM on International Workshop on Security and Privacy Analytics, pp. 55–63. ACM (2017)
9. Vu, L., Nguyen, P., Turaga, D.: Firstfilter: a cost-sensitive approach to malicious URL detection in large-scale enterprise networks. IBM J. Res. Dev. **60**(4), 4:1–4:10 (2016)