URL_trigger: Real time solution for Detection Malicious URL using Deep Learning

Omar LAMRABTI, Abdellatif MEZRIOUI, and Abdelhamid BELMEKKI

STRS LAB, RAISS TEAM, National Institute of Posts and Telecommunications, RABAT, Morocco
o.lamrabti@inpt.ac.ma,mezrioui@inpt.ac.ma, belmekki@inpt.ac.ma

Abstract. Evolution of technology has many drawbacks as well as benefits, requiring a great investment to ensure security where the human component is the core of this equation. To that end, malicious URLs are among tactics that target users unawareness, unable to distinguish between legitimate and malicious URLs, inducing them to interact with malicious links that hosts varieties of malicious contents such as malware, phishing, or drive-by downloads to carry out cyberattacks. To contribute to this defiance, research focused on Machine Learning (ML) systems. Unfortunately, to withstand new attacks via ML, features collection must be a continuous task that consumes time and energy, which is considered a major drawback. However, Deep Learning (DL) mitigates this defiance by learning from unstructured data without supervision. Nevertheless, adoption of DL doesn't cover all well-known DL models in one experience. In this paper, we introduce "URL_trigger", a solution based on DL and ML in order to detect malicious URLs. To reach this goal, the solution includes an intelligent system that collects and detects malicious URLs from various sources (e.g. twitter, zone-h, pastebin, virustotal) that will be saved continuously in our dataset. URL_trigger provides real-time evaluation of malicious URLs via various techniques, including: blacklisting, lexical features, host-based features, content-based features, machine learning, and deep learning. We use CNN, RNN, RCNN, and DNN as models for learning URL_trigger. Achieving an accuracy of 99% confirms the effectiveness of our system in terms of detecting malicious links compared to other solutions that use DL.

Keywords: Deep Learning, Machine Learning, Malicious URL, URL_trigger.

1 INTRODUCTION

In the last decade, malicious URL detection has become a very important cybersecurity task. With the exponential increase in cyberattacks, it has become imperative to take measures to prevent these attacks. To address this growing concern, researchers have used machine learning (ML) and deep learning (DL) techniques to develop more effective malware detection systems.

The use of deep learning models such as LSTM and CNN has been shown to be effective in detecting malicious URLs. In this paper, we discuss how deep learning models have been used to detect malicious URLs.

The main contributions of this work are:

- Proposing a hybrid solution based on deep learning models for detecting malicious URLs and phishing websites based on website URL only.
- Combining the advantages of LSTM and CNN in processing legitimate and benign URLs.
- Using a large dataset from Twitter, Zone-h, Pastebin, VirusTotal, DomCop and PhishTank, which contains almost 7 million phishing and legitimate URLs.
- Testing many optimizers such as (Adam, Nadam, SGD, AdamW)
- Achieving the best results in terms of accuracy, which reached 99.1% for CNN and 98.82% for LSTM, and precision 99.50% for CNN and 98.95% for LSTM.

This article is organized as follows: Section II provides background information on various deep learning models used in phishing and malicious URL detection, as well as an overview of related work in this area at ML and DL. Section III describes our basic idea regarding the architecture of URL_Trigger and explains the approach in detail. In Section IV, we discuss the details of the experiment with URL_Trigger by giving the results and an analysis of the experiment with the results compared to other research. Finally, in section VI we give the conclusions and future work.

2 BACKGROUND AND RELATED WORKS:

A URL (Uniform Resource Locator) is a string of characters that provides access to a resource on the Internet. Cybercriminals use malicious URLs to spread malware, viruses, or phishing attacks. These attacks can result in significant financial losses or data breaches for the user.

Recent academic work has investigated various methods for detecting malicious URLs using machine learning and deep learning algorithms. One of these methods is using convolutional neural networks (CNNs) to classify URLs as malicious or benign based on the patterns present in their structure. In [1], a study that used a multi-stage method to classify phishing websites based on CNNs with character embedding and RF, researchers achieved 99.35% detection accuracy, which outperformed traditional machine learning algorithms. Another promising approach is the use of recurrent neural networks (RNNs) with attention mechanisms. In [2], the authors of a study evaluate several deep learning architectures, specifically recurrent neural networks (RNNs), identity recurrent neural networks (I-RNNs), long short term memory (LSTM) architectures, convolutional neural networks (CNNs), and convolutional long short term memory neural networks (CNN-LSTM), to avoid manual feature engineering by modeling the real known benign and malicious URLs in character level language.

In addition to CNNs and RNNs, researchers have also explored the use of other deep learning techniques, such as long-term memory networks (LSTMs). In [3], authors present Monarch, which crawls web services to detect whether a URL is malicious or benign. The implementation of Monarch includes a URL aggregator that crawls URLs from different data stream sources (e.g., Twitter, Web). Then, a feature extractor processes the collected features from the crawled URL and transforms them. Finally, a logistic regression classifier was trained to efficiently detect malicious URLs.

Overall, this recent scientific work demonstrates the potential of machine learning and deep learning techniques for detecting malicious URLs. These methods have been shown to be effective in distinguishing between malicious and benign URLs by analyzing the structural patterns of URLs. As cyber attackers develop increasingly sophisticated techniques to spread malware, it is important to continue to research and improve the accuracy of these detection systems.

3 OVERVIEW OF URL_Trigger:

3.1 DATA PREPROCESSING

As shown in Figure 1, URL_Trigger is a combination of several processes. It uses a 'URL collector' that collects URLs from various data streams such as Twitter, VirusTotal using their API stream, and other resources such as zone-h and Pastebin. The collector stores the collected data in our dataset in real time. We also use the PhishTank dataset, which contains phishing websites, and DomCop, which contains legitimate websites, as resources for training our solution. After processing, we obtained a dataset containing 3,461,988 phishing URLs and 3,520,300 legitimate URLs. Due to the lack of resources and because it is better to use a balanced dataset, we used the above values in Table 1 for the rest of this work. The dataset was randomly split into three subsets: 0.6 training, 0.2 validation, and 0.2 testing data to avoid overfitting.

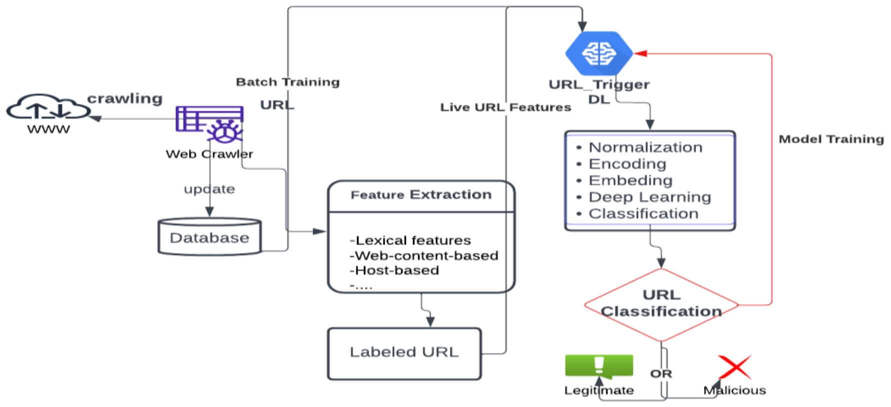

Fig. 1. URL_Trigger architecture

Table 1. Dataset structure

DATA	TRAINING	TESTING
Benign	84900	40000

Malicious	79000	39000
TOTAL	163 900	79000

3.2 URL_Trigger approach

From this experience, the URL_Trigger system uses a hybrid architecture that balances the benefits of manually created features and character embedding based features. The crawled URLs are preprocessed by eliminating capital letters. Since URLs can have different lengths, we set a global length of L=150. If the length of the URL is greater than 150, our solution captures only the first 150 characters; however, if the number of characters is less than 150, we add zeros up to 150.

The first method, using manually created features, first extracts lexical features, i.e., various URL components such as the address, which consists of the hostname, path, and so on. Web content-based features by searching for (ie: the number of suspicious JavaScript objects, number of hyperlinks in the web page). Then, we search for the host-based features to find out the IP addresses of each of the examined URLs, and also look for information about the autonomous system number (ASN), which helps us to find out if the website has already been classified into the category of bad ASNs. The collected information forms a large feature vector, which we hash into a 1024-dimensional vector using the feature hashing trick. The hashed features are fed to the LSTM model.

In the second method, which uses character embedding based features for phishing URL detection with the CNN model, each URL is embedded into a 32-dimensional vector using the character embedding method [4]. We set the vector size to 94 since there are 26 lowercase letters, 26 uppercase letters, 10 numbers, and 32 special characters. We obtained a matrix of size 163900 × 150 for training and 79000 × 150 for validation. Each URL character is embedded in a 64-dimensional vector. These matrices are passed to an embedding layer using the batch size of 64 as 64x150. This embedding layer constructs a matrix of size 94 × 256. This process is a dictionary search in which each character, regardless of the characters that precede or follow it, is mapped to the appropriate vector that represents the vocabulary matrix. We apply one-dimensional convolution, since URL is one-dimensional, to obtain a dense representation of URL characters. The URL_Trigger neural network architecture contains five deep layers, including three convolutional layers and two fully connected layers. Each convolutional layer has a filter of length 32, i.e., the filters are each applied to 32 characters at a time. We pass the convolutional output to a pooling layer, which acts as condensing layer and returns the maximum feature value. Next, we use a stride of length 2, which converts the obtained feature map into two equal parts. Then, the results of the convolutional layer and the pooling layer were passed as a one-dimensional vector to the two fully connected layers. Finally, we added the last layer, which has one output. We use a 'sigmoid' activation that takes the output of the previous layers and the output of this layer and converts it to a value between 1 or 0 (1 malicious, 0 benign).

3.3 URL_Trigger Hyper parameters and EVALUATION INDICATORS:

The CNN and RNN networks require tuning of parameters to achieve the best results in distinguishing malignant from benign URLs. In this experience, we focused on hidden layers, batch size, hidden layer units, dropout, stride, optimizer, and learning rate. We tested with dropout = 0.1, learning rate = 0.02, and batch size = 64. For the CNN, we tested with 16, 32, 64, and 128 filters, and the 64 filters gave the best result. To evaluate the performance of our proposed system, we use the metrics of accuracy, F1-score, and precision.

4 EXPERIENCE DISCUSSION

Combined with the experience with handcrafted features, the Keras library was chosen to implement the proposed RNN model. After various tests, we selected the best hyperparameter values. Adam was chosen as the optimization function because it provides better performance in terms of accuracy than the other tested optimizers (i.e.: nadam, sgd, AdamW, Adadelta). ReLU was used as the activation function for the hidden layers, and the sigmoid function was used for the output layer. We set the dropout rate to 0.3 for the global dropout layers. We use 500 epochs with 64 batch sizes for training. The LSTM model provides 98.82% accuracy.
On the other hand, the graph of training accuracy for the CNN model shows that the loss has decreased fairly steadily over time.

However, we can clearly see in the graph that the accuracy has been steadily increasing over time, we had a small dip, but again the accuracy has seen a peak at epoch 50 and then fluctuations followed, we have a comeback, and it has resolved back to 99.1% accuracy, which is very relevant as a result and explains that our model works very well.

Table 2. Results of URL_Trigger

Model	Accuracy	F1-score	PRECISION
LSTM	98,82 %	98,62%	98,95%
CNN	99,1%	98,95%	99,50%

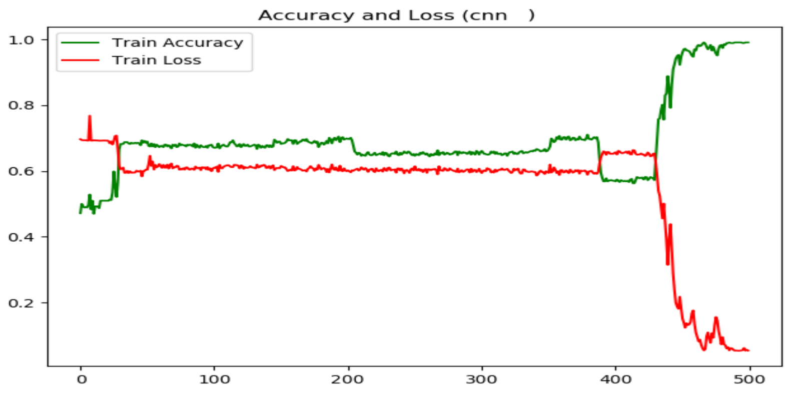

Fig. 2. Taining and Validation Accuracy for CNN

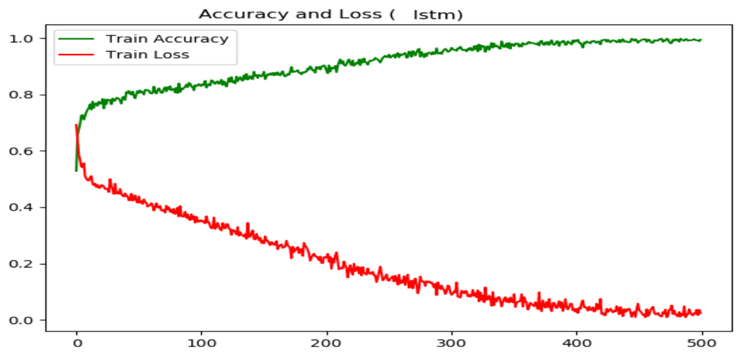

Fig. 3. Taining and Validation Accuracy for LSTM

5 Conclusion and Future Work

In this paper, we presented our new model for phishing detection called URL_Trigger. Our model uses two methods to classify legitimate or malicious URLs. In the first method, we use a hand-craft features method to analyzes a URL and classify that web page as phishing or legitimate using a deep neural LSTM network. The second method, the URL_Trigger model, is trainned based on the CNN model, which automatically decides whether a URL is phishing or legitimate by learning patterns from trained data. Both the first and second methods achieved the best results in terms of accuracy, which are 98.82% and 99.1%, respectively.

Future work will focus on detecting suspicious activity in a system using machine learning and deep learning (i.e.: malware, ransomware...).

References

1. Yang R, Zheng K, Wu B, Wu C, Wang X. Phishing Website Detection Based on Deep Convolutional Neural Network and Random Forest Ensemble Learning. Sensors (Basel). 2021 Dec 10;21(24):8281. doi: 10.3390/s21248281. PMID: 34960375; PMCID: PMC8709380.
2. Ravi, Vinayakumar & Kp, Soman & Poornachandran, Prabaharan. (2018). Evaluating deep learning approaches to characterize and classify malicious URL's. Journal of Intelligent & Fuzzy Systems. 34. 1333-1343. 10.3233/JIFS-169429.
3. Kurt Thomas, Chris Grier, Justin Ma, Vern Paxson, and Dawn Song. 2011. Design and evaluation of a real-time URL spam filtering service. In Security and Privacy (SP), 2011 IEEE Symposium on. IEEE.
4. Zhang, X.; Zhao, J.; LeCun, Y. Character-level convolutional networks for text classification. Adv. Neural Inf. Process. Syst. 2015,28, 649–657.