ISSN: 2995-3286

Research Article

International Journal of Media and Networks

Phishing URL Detection Using CNN-LSTM and Random Forest Classifier

Hemant Gurung, Roshan Nepal and Sopnil Nepal*

National College of Engineering Kathmandu, Nepal

***Corresponding Author**
Sopnil Nepal, National College of Engineering Kathmandu, Nepal.

Submitted: 2023, Oct 27; **Accepted:** 2023, Nov 06; **Published:** 2024, May 27

Citation: Gurung, H., Nepal, R., Nepal, S. (2023). Phishing URL Detection Using CNN-LSTM and Random Forest Classifier. *Int J Med Net, 2*(5), 01-06.

Abstract

This paper presents the classification of phishing URL's apart from legitimate URL's with the use of machine learning and deep learning techniques. Phishing is defined as an act to steal the private information by pretending to be a legitimate entity which they are not. Machine learning model, Random Forest classifier is trained on the extracted features based on Address Bar, Domain and HTML and JavaScript of the URL. On the other hand, CNN-LSTM hybrid model was trained to learn the character sequence features of the given URL and make the classification. The dataset used was public data from Kaggle which was downloaded from their website. The dataset contained 11,430 URLs: 5,715 legitimate URLs and 5,715 phishing URL. Hereafter, we classified the URL of the current address bar as legitimate or phishing with the use of previously trained model. Thus, proposed paper focuses on the study and development of models for detection of phishing sites so that properties of various URLs can be learnt by feature extraction and can be classified as accurately as possible.

Keywords: Phishing Website Detection, Convolutional Neural Network, Long Short-Term Memory Network, Random Forest, Machine Learning

1. Introduction

In recent years, as internet technology begins to evolve every day, it has brought great convenience to human society. There is no denying the fact that internet has become a primary source for information and data sharing. The internet era is booming and its usage only goes uphill from here on now. Many people share their information like their email, location, credit card information, bank details, etc. for various purpose whether it may be online shopping, charity, online banking or different purpose. They share this information with the legitimate companies. With a lot of users sharing such information, internet is expected to be infested with people who intend to steal this sensitive information. According to the FBI, phishing was the most common type of cybercrime in 2020 and phishing incidents nearly doubled in frequency, from 114,702 incidents in 2019, to 241,324 incidents in 2020 [1]. With these statistics as evidence, it is pretty obvious that phishing has been causing a lot of problems for innocent users of internet. In current scenario, to mitigate the effect of phishing there are roughly about three techniques widely used. The first way is through the user awareness. The second and the most common way is by blacklisting the phishing websites. However, the disadvantage of this approach is that, to blacklist a website it should be proven as a phishing website. The third way that has proven to be the most effective is to use machine learning and deep learning techniques that learns about the characteristic features of previous malicious links and can make accurate distinctions in the future based on previous predictions made [2]. Current mainstream machine learning methods of phishing website detection extract statistical features from the URL or extract relevant features of the webpage, such as the layout, Domain information or HTML& JavaScript and then classify these features but machine learning algorithms do not analyze the sequence or the positions of words in a URL and also 63% of phishing websites have a lifespan of only 2 hours after which they change either expire or change their domain name [3]. In order to use the machine learning techniques that focuses on the statistical features of URL and also to exploit the orientation and sequence learning capability of deep learning, we propose a CNN-LSTM model along with Random Forest, they belong to the field of deep learning whereas Random Forest classifier belongs to the field of machine learning.

2. Related Works

Qiao Zhang et al. proposed a phishing website detection technology based on CNN-BiLSTM algorithm. Their model attempted to solve the problems of existing phishing web page detection methods with manual feature extraction. Their method first performed word segmentation processing on URL based on sensitive word segmentation, then converted it into a feature vector matrix that automatically extracts its local features through CNN and acquired its bidirectional long-distance dependent features through BiLSTM. Their model classified the phishing and legitimate URLs with accuracy of 98.84%[3].

T. Sujithra et al implemented various machine learning algorithms to reduce the false positives in detecting new phishing sites. They attempted to identify the best machine learning algorithm to detect phishing sites with high accuracy than the existing techniques. After implementing various classifying algorithms, they found that XGBOOST classifier outperformed the rest.

According to their research, XGBOOST algorithm had accuracy of 94.7% [4].

Peng Yang et al. proposed a multidimensional feature phishing detection approach based on a fast detection method by using deep learning. In the first step, character sequence features of the given URL were extracted and used for quick classification by deep learning, and this step did not require third-party assistance or any prior knowledge about phishing. In the second step, they combined URL statistical features, webpage code features, webpage text features and the quick classification result of deep learning into multidimensional features. The approach could reduce the detection time for setting a threshold. With this approach they were able to achieve accuracy of 98.99% [5].

Rakotoasimbahoaka et al. proposed a reliable, generic and flexible system. They proposed a hybrid approach based on Machine Learning and Deep Learning methods (CNN-LSTM-RF). They performed manual feature extraction for RF (Random Forest) algorithm and automatic feature extraction with CNN-LSTM model. CNN_LSTM_RF 10 produced an interesting result, with convergence after three epochs and an accuracy rate of malicious URL detection of 96%. They also tried experimenting with RF_CNN_LSTM hybrid but with this model the performance was poor and the malicious URL detection accuracy was just 50% [6].

Vysakh S Mohan et al. proposed S.P.O.O.F Net: Syntactic Patterns for identification of Ominous Online Factors. It was a combination of Convolutional Neural Network and Long Short-Term Memory Neural Network. The proposed architecture was found to outperform existing threat detection strategies like blacklisting, sink holing and machine learning based classifiers for malicious URL detection. S.P.O.O.F Net overcomes drawbacks of methodology of traditional methods, like the requirement of a domain level expert for constant maintenance of the database the classifier is trained on, because the threats are ever changing. With this model they were able to achieve accuracy of 95.2 % [7].

Hitesha Gupta et al attempted to perform early phishing detection using XGBOOST classifier. Thy tried to solve the problem of stealing confidential information from the victims using legitimate websites or email. They used three datasets have used for this simulation. The accuracy achieved by this method was highest in comparison to the other machine learning classifying algorithms i.e., 98.45%. Using XGBoost classification, the total F1-measure obtained by the FRS function choice was 98.45% [8].

3. Methodologies

For our proposed methodology we have basically used two algorithmic models namely random forest classifier and CNN LSTM hybrid model (LSTM is used in order to make future classification based on previous classifications made) and we have also used 1D convolution neural network for CNN is used to learn about the sequence of characters present in URL.

3.1 Random Forest Algorithm

Random Forest is a machine learning algorithm. For the implementation of this algorithm, several features are extracted from the collected dataset. The URL embedding matrix used in deep learning cannot fully represent the phishing website information. The different features are:

• **IP Address in URL:** It is unusual to see IP address in the address bar of the browser while surfing the internet. Example: “http://120.30.3.3/abc.php”. If such URLs appear then it might be a phishing one.

• **"@" Symbol in URL:** If “@” symbol is present in the URL then everything to the left of “@” is ignored and only the right part of URL is taken into consideration by the browser thus providing an easy gateway to phishing sites.

• **Length of URL:** Phishers sometimes exploit the features of browsers by creating a very long URL in order to hide the true identity of the URL. It is unusual to see long URLs while browsing internet. Most legit URLs are at max 200 characters long.

• **Redirection "//" in URL:** If “//” is present in URL then it means that there is redirection to another URL. Phishers can put their phishing links after ‘//’ so that users can be redirected to their site.

• **"http/https" in Domain name:** HTTP does not have a security mechanism for data encryption and has no SSL certificate. Having no HTTPS makes it more likely to be a phishing URL.

• **Using URL Shortening Services “Tiny URL”:** URL shortening is a method to reduce the length of URL that creates another URL of smaller length. The smaller length URL will redirect to the original website. But with shortened URL, the originality of the URL is now masked and new users will have no idea what is the website that the link will lead to.

• **Prefix or Suffix "-" in Domain:** The “-” symbol is mostly used to mimic the legitimate website. Example: “https://www.pay-pal.com”. To the naïve users it may seem like the legit one. It is unlikely to see URLs with dash symbol frequently.

• **Favicon:** Favicons may be described as the logo that appears on the tab of the web pages. They are used to provide the visual identity of websites. Phishers can use the favicons of legitimate websites in order to mimic them that is generated from another website.

• **Request URL:** Many phishing websites have request URLs to load the components of webpage like image, icons, etc. from another site. Clicking on these foreign components may lead to redirecting to another site.

• **URL of anchor:** The anchor tag in phishing URLs are used to redirect to external websites. Legit sites have anchor tags that mostly redirect to the same domain name.

• **Links in Script:** The script tag of html of website contains the JavaScript for that site. It is expected that the JavaScript is loaded from the same domain name for that website and no external and suspicious scripts are loaded.

It will result in total of 14 manually extracted features. To extract these features, different python libraries like re, urllib, I p address, BeautifulSoup, who is and requests were used. The number of estimators used was set to 100. For each feature, if the feature satisfies the condition specified to be declared as a phishing URL, the value 1 is assigned else the value 0 is assigned to it. The features with labels of 0 and 1 are then shaped into array and are passed to the model for classification.

3.1.1 Random Forest Pseudocode

1. Randomly select "k" features from total "m" features.
2. n Where k << m Among the "k" features, calculate the node "d" using the best split point.
3. Split the node into daughter nodes using the best split on the basis of highest information gain where highest information gain is calculated using :

$IG(S, A) = Entropy(S) - \sum((|S_v| / |S|) * Entropy(S_v))$
$Entropy(S) = - \sum p_i * \log_2(p_i)$; i = 1 to

4. Repeat 1 to 3 steps until "l" number of nodes has been reached.
5. Build forest by repeating steps 1 to 4 for "n" number times to create "n" number of trees. [9]

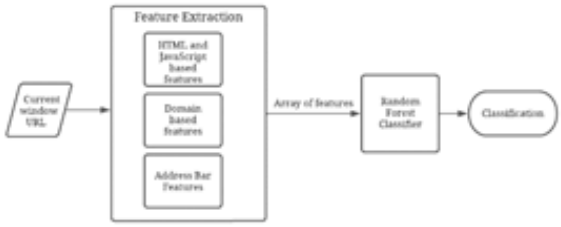

Figure 1: Random Forest Working Mechanism

3.2 The CNN-LSTM Algorithm

In first step using CNN-LSTM model, firstly, the URL is to be segmented on character level. Then the normalization is to be performed on the segmented URL. For normalization we would fix the maximum length of URL as maximum length of URL present in our dataset. Let it be L. If the length of a sample URL is less than L then padding would be added in order to extend it up to length L. Then character map dictionary will be implemented in order to convert the characters into one hot code sequence.

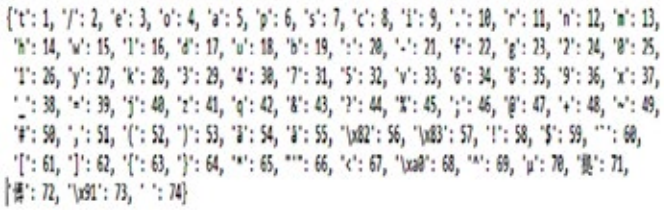

Figure 2: Character Map Dictionary

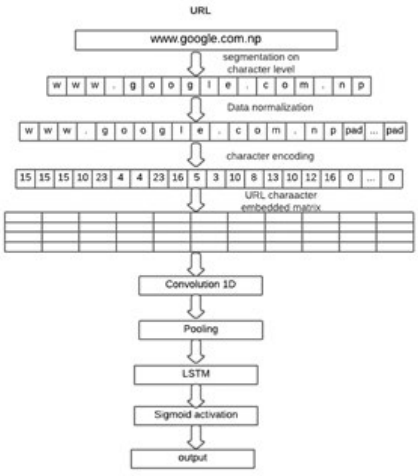

Figure 3: CNN LSTM Working Mechanism

In first step using CNN-LSTM model, firstly, the URL is to be segmented on character level. Then the normalization is to be performed on the segmented URL. For normalization we would fix the maximum length of URL as maximum length of URL present in our dataset. Let it be L. If the length of a sample URL is less than L then padding would be added in order to extend it up to length L. Then character map dictionary will be implemented in order to convert the characters into one hot code sequence. The one hot code sequence consists many zeros and are inefficient for computation and storage. Embedding layers convert this one hot sequence into fixed length vector representation with reduced dimension that makes them computation efficient. Then convolution will be performed on embedding matrix. URL is a 1D character sequence so Convolution1D is suitable. With convolution the deep correlation features among the characters in a URL will be extracted. CNN will learn about the positions of different characters in a URL and how deeply are the characters related to one another. After convolution, pooling is performed. The result of pooling is the input to LSTM.

The output of pooling contains sequence of embedded vector representation. This sequence can be treated as a time series data as for different timestamp different value from pool is obtained. LSTMs are great for time series input. LSTM learns the sequential information among the features obtained from CNN. LSTM implement memory cells that can remember both the long term and short-term sequence. LSTM captures the context of URL sequence and dependency.
The output of LSTM is then subjected to sigmoid activation that performs a binary classification on the output of LSTM.

4. Experiments and Results

The research is based on two sets of algorithm:
a. Random forest classifier
b. CNN LSTM algorithm

The efficiency of the proposed algorithms are tested using the performance parameters namely: Accuracy, Sensitivity and precision.
These parameters are calculated as:

i. Accuracy: It is the percentage of all normal and anomaly instances that are correctly classified.

$$\text{Accuracy (A)} = (TP+TN) / (TP+TN+FP+FN).$$

ii. Sensitivity: Sensitivity is a measure of the proportion of actual positive cases that got predicted as positive (or true positive). Sensitivity is also termed as Recall.

$$\text{Sensitivity} = (TP) / (TP+FN)$$

iii. Precision: Precision refers to the quality of positive prediction made by the model. Precision can be derived by the number of true positives divided by the total number of positive predictions (i.e. the number of true positives plus the number of false positives).

Precision= (TP) / (TP+FP)

Where
True Positive (TP) = URL that are actually detected as legitimate.
True Negative (TN) = URLs that are actually detected as phishing.
False Positive (FP) = URLs that are predictively detected as legitimate.
False Negative (FN) = URLs that are predictively detected as phishing.

4.1 Dataset for Testing

From the total dataset, the data was split as 80% for training and 20% for testing purposes. After the model was trained, the model was again tested for the entire URL of the dataset.

4.2 Results

The random forest model used 100 estimators and was trained on the training data. When implemented on the testing data it had an accuracy of 70.034%. The confusion matrix for random forest model evaluated against test data (n=20% of 11430) is demonstrated in Table 1:

n=2286	predicted: legitimate	predicted: phishing
Actual: legitimate	899	258
Actual: phishing	427	702

Table 1: Confusion Matrix for Random Forest

The same test data was also used for evaluation of CNN-LSTM model. For the testing data, the model provided an accuracy of 94.7%.The confusion matrix for CNN_LSTM model evaluated against test data is shown in Table 2:

n=2286	predicted: legitimate	predicted: phishing
Actual: legitimate	1101	56
Actual: phishing	65	1064

Table 2: Confusion matrix for CNN LSTM

4.3 Data Sample Result

4.3.1 Random Forest

The result of Performance parameter calculation for Random Forest and Dataset are classified in following table and graph:

URL count	Actual Legitimate	Predicted Legitimate	Actual Phishing	Predicted Phishing	Accuracy	Sensitivity	Precision
1-2000	997	772	1003	693	73.25%	77.4%	71.3%
2001-4000	999	777	1001	645	71.1%	77.77%	68.5%
4001-6000	1011	800	989	553	67.13%	79.12%	64%
6000-8000	980	762	1020	650	69.94%	77.75%	67.3%
8000-10000	1003	769	997	623	69.6%	76.66%	67.27%
10000-11430	725	582	705	433	70.5%	80.27%	68.14%
Average					70.25%	78.16%	67.75%

Table 3: Performance Parameter of Random Forest

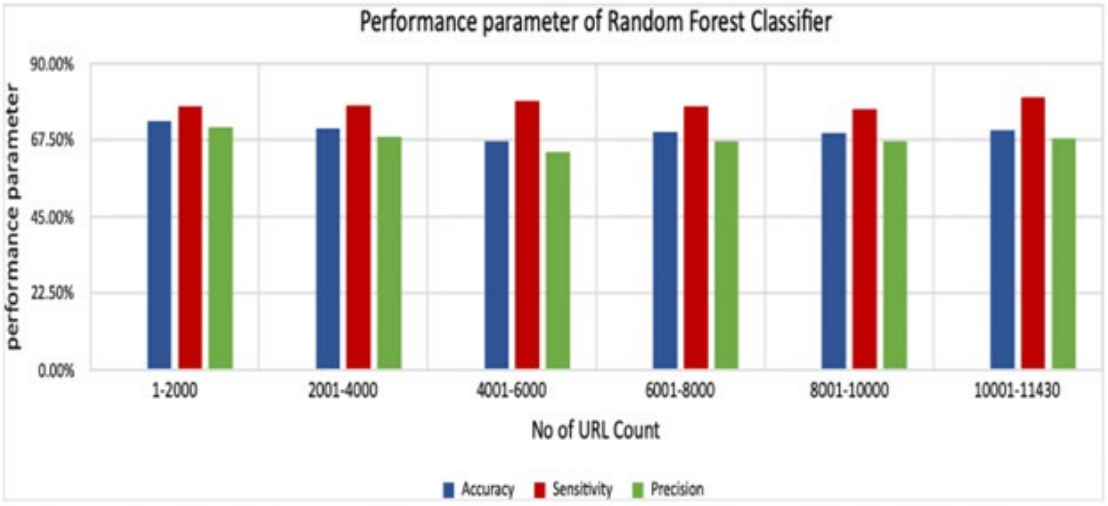

Figure 4: Performance Parameter of Random Forest Graph

The Table 3 and Figure 4 above shows the calculated value of performance parameters i.e., Actual legitimate, Predicted Legitimate, Actual Phishing, Predicted Phishing, Accuracy, Sensitivity and Precision. Here, the average accuracy of algorithm is 70.25% with highest accuracy value as 71.1% and lowest of 67.13%. Similarly average sensitivity obtained here is 78.16% with highest sensitivity value of 80.27% and lowest is 76.66%. Finally, Average precision is obtained for this dataset is 67.75% with highest value of 71.3% and lowest of 64%.

4.3.2 Deep Learning Model

The result of Performance parameter calculation for Deep Learning Model and Dataset are classified in following table and graph:

URL count	Actual Legitimate	Predicted Legitimate	Actual Phishing	Predicted Phishing	Accuracy	Sensitivity	Precision
1-2000	997	927	1003	956	94.15%	95.55%	95.49%
2000-4000	999	945	1001	956	95.05%	95.05%	95.04%
4000-6000	1011	937	989	958	94.75%	94.81%	94.07%
6000-8000	980	916	1020	992	95.4%	95.48%	95.3%
8000-10000	1003	910	997	956	93.3%	93.41%	93.03%
10000-11430	725	679	705	671	94.4	93.26%	93.13%
Average					94.3%	94.59%	94.51%

Table 4: Performance parameter calculation for Deep Learning Model

The Table 4 and Figure 5 above shows the calculated value of performance parameters i.e. Actual legitimate, Predicted Legitimate, Actual Phishing, Predicted Phishing, Accuracy, Sensitivity and Precision. Here, we can see that the average accuracy of algorithm is 94.3% with highest accuracy value as 95.5% and lowest of 93.3%. Similarly, average sensitivity obtained here is 95.59% with highest sensitivity value of 95.55% and lowest is 93.26%. Finally, Average precision is obtained for this dataset is 94.51% with highest value of 95.49% and lowest of 93.03%.

5. Conclusion and Future Works

From the above analysis we can infer that for the above dataset, random forest Algorithm followed by CNN-LSTM Random Forest model provided accuracy of around 70% and CNN-LSTM model provided around 94%. 14 different features were extracted manually for random forest model. The lower accuracy on random forest model maybe due to the fact that many phishing websites have been shut down so the feature extraction by making a request to such website and web scraping for such websites was not very optimal. In this work, we have performed the work by taking the total 11430 sample URLs where we further divided those sample in six batches with sample size 2000 URLs per sample. We have used 14 different parameters (Ip address in URL, '@' symbol in URL, length of URL, '//' redirection in URL, "http/https" in URL, Tiny URL service, Prefix or Suffix containing "-", External favicon source, External Request URL, Link in Script and link tags, Server Handler Form, Submitting to email, I frame redirection). From result of the sample test above we achieved the average accuracy of random forest 70.25% and for CNN LSTM of about 94.3%.

In future, this work can be extended and enhanced as follows:
1. Distribution of data can be changed i.e. both large size and small data size samples can be taken instead of equal size for testing the result.
2. The trained models can be implemented either in form of browser extension or web application for the real time detection of phishing URLs.

Acknowledgements

We would like to thank the Department of Computer and Electronics Engineering, National College of Engineering for providing us the working opportunity and motivation to enhance our knowledge and provide us a new experience of teamwork.

Compliance with Ethical Standards

• The authors have no conflict of interest.
• The authors did not receive support from any organization for the submitted work.
• No funding was received to assist with the preparation of this manuscript.
• No funding was received for conducting this study.
• No funds, grants, or other support was received.
• The study was conducted without involvement of experimentation on humans or animals.

Competing Interests

• The authors have no relevant financial or non-financial interests to disclose.
• The authors have no competing interests to declare that are relevant to the content of this article.
• All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.
• The authors have no financial or proprietary interests in any material discussed in this article.

Data Availability Statement

The initial relevant data corresponding to this paper was accessed from https://www.kaggle.com/datasets/shashwatwork/web-page-phishing-detection-dataset. The data was modified around our research and feature extraction was performed resulting new dataset which can be found uploaded to the following links:
Dataset1(URL with labels):
https://figshare.com/articles/dataset/URL_csv/21070183
Dataset2(URL with extracted features)
https://figshare.com/articles/dataset/phishingfeatures_csv/21070198
All data generated or analyzed during this study are included in the published article.

References

1. V. Patel, "eBook on How to Protect Yourself and Your Company from Phishing for Free," 9 June 2021. Available: https://druvstar.com/how-to-protect-yourself-and-your-company-from-phishing-for-free/.
2. Sravanth, M., T. Reddy, K. Nagendra, and G. Aashirvad. "Phishing Website Detection Based On Multidimensional Features Driven By Deep Learning." *International Journal of Techno-Engineering* (2021): 105-112.
3. Zhang, Q., Bu, Y., Chen, B., Zhang, S., & Lu, X. (2021). Research on phishing webpage detection technology based on CNN-BiLSTM algorithm. In *Journal of Physics: Conference Series* (Vol. 1738, No. 1, p. 012131). IOP Publishing.
4. Sujithra, T., Dwivedi, N., & Utakarsha, A. (2020). Detection of phishing websites using deep learning and machine learning. *Journal of Critical Reviews, 7*(8).
5. Yang, P., Zhao, G., & Zeng, P. (2019). Phishing website detection based on multidimensional features driven by deep learning. *IEEE access, 7,* 15196-15209.
6. Rakotoasimbahoaka, A., Randria, I., & Razafindrakoto, N. R. (2019). Malicious URL detection by combining machine learning and deep learning models. *Artificial Intelligence for Internet of Things, 1*.
7. Mohan, V. S., Vinayakumar, R., Soman, K. P., & Poornachandran, P. (2018, May). Spoof net: syntactic patterns for identification of ominous online factors. In *2018 IEEE Security and Privacy Workshops (SPW)* (pp. 258-263). IEEE.
8. H. Gupta and R. Shrivastava, "Early Phishing Attack Detection using XGBoost Classifier," JASC: Journal of Applied Science and Computations, pp. 138-145, 2019.
9. Polamuri, S. (2017). How the random forest algorithm works in machine learning. *Retrieved December, 21.*