Intelligent Detection Technique for Malicious Websites Based on Deep Neural Network Classifier

Article · December 2022

DOI: 10.5815/ijeme.2022.06.05

4 authors:

Mustapha Adamu Mohammed
Kwame Nkrumah University Of Science and Technology
7 PUBLICATIONS **9** CITATIONS
SEE PROFILE

Seth Alornyo
University of Electronic Science and Technology of China
12 PUBLICATIONS **30** CITATIONS
SEE PROFILE

Michael Asante
Kwame Nkrumah University Of Science and Technology
50 PUBLICATIONS **118** CITATIONS
SEE PROFILE

Bernard Obo Essah
University of Ghana
6 PUBLICATIONS **4** CITATIONS
SEE PROFILE

I. J. Education and Management Engineering, **2022, 6, 45-54**
Published Online on December 8, 2022 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijeme.2022.06.05

Intelligent Detection Technique for Malicious Websites Based on Deep Neural Network Classifier

Mustapha A. Mohammed*
Kwame Nkrumah University of Science and Technology, Kumasi, 00233, Ghana & Koforidua Technical University, Koforidua, 00233, Ghana
E-mail: adamu.mohammed@ktu.edu.gh
ORCID iD: https://orcid.org/0000-0003-4190-3970
*Corresponding Author

Seth Alornyo
Koforidua Technical University, Koforidua, 00233, Ghana
E-mail: sabigseth@ktu.edu.gh
ORCID iD: https://orcid.org/0000-0003-4548-9690

Michael Asante
Kwame Nkrumah University of Science and Technology, Kumasi, 00233, Ghana
E-mail: masante.csi@knust.edu.gh

Bernard O. Essah
Department of Mathematics, St. Gregory Catholic SHS, Gomoa East, 00233, Ghana.
E-mail: boessah@st.ug.edu.gh
ORCID iD: https://orcid.org/0000-0001-6765-4111

Received: 23 August, 2022; Revised: 22 October, 2022; Accepted: 10 November, 2022; Published: 08 December, 2022

Abstract: A major risk associated with internet usage is the access of websites that contain malicious content, since they serve as entry points for cyber attackers or as avenues for the download of files that could harm users. Recent reports on cyber-attacks have been registered via websites, drawing the attention of security researchers to develop robust methods that will proactively detect malicious websites and make the internet safer. This study proposes a deep learning method using radial basis function neural network (RBFN), to classify abnormal URLs which are the main sources of malicious websites. We train our neural network to learn benign web characteristics and patterns based on application layer and network features and apply binary cross entropy function to classify websites. We used publicly available datasets to evaluate our model. We then trained and assessed the results of our model against conventional machine learning classifiers. The experimental results show a very successful classification method, that achieved an accuracy of 89.72% on our datasets.

Index Terms: Deep learning, radial basis function neural network, malicious websites, malicious URLs.

1. Introduction

Malicious websites host unsolicited content such as spams, phishing and drive by downloads which often entice naive users to fall victims to the activities of cybercriminals. The attackers use sophisticated methods to trick individuals into disclosing sensitive data or install malicious software in their systems, that eventually lead to financial loss to these individuals and organizations. Compromised uniform resource locators (URLs) have been the main mode of operation used by the cyber attackers to craft those rogue webpages in order to launch their illicit activities.

Malicious websites have multiplied in number during the past few years. As of the beginning of 2018, one out of every thirteen URLs was malicious, resulting in a staggering 7.8% of all URLs as being malicious [1]. These numbers have gone up 2.8% as of 2020, indicating a rising trend in attack vectors via malicious websites.

A cyber attacker can use malicious website to impersonate a benign site in order to acquire sensitive data, including passwords, account details, or credit card information. Malicious URLs have thus become a major problem, because it is difficult to examine each one individually and add each URL to a black list. The difficulty of analyzing and indexing URLs to segregate malicious websites from legitimate ones is a major source of concern to researchers. URLs have been traditionally detected by blacklist techniques. However, these approaches have become ineffective at detecting new generations of malevolent websites [1].

Creators of new generation rogue websites use obfuscation techniques, such as misspelt names to modify the URLs for them to seem genuine in other to evade blacklist methods employed by many antivirus software. To alleviate this problem, Do Xuan et al [2], Gabriel et al [3], Catak et al [4] and a host of many other researchers have in the past few years employed machine learning methods to detect malignant URLs.

Using a collection of URLs as training data, machine learning techniques create model frameworks that can be applied to the data and learn a predictive function to categorize a given URL as malicious or legitimate. In contrast to blacklisting techniques, machine learning approaches can generalize to new URLs. However, some properties and nature of URLs, as well as the huge amounts of data required often results in difficulty in training the URL classifier in respect of learning appropriate function and scalability [1]. Consequently, the training complexity of the model becomes high, even though in theory, large amount of training data should make the machine learning model better.

As a result, a group of scalable learning methods called online learning [5], has been extensively used for this classification task. The sparseness of the URLs' bag-of-words feature representation poses yet another difficulty to this approach. Every word that could possibly appear in any URL is turned into a feature. Therefore, it is possible that tens of thousands of sparse features will be produced.

In order to increase learning efficiency and effectiveness, a learning approach should take advantage of this feature sparsity. There are many more challenges associated with the task of detecting malicious websites based on URLs, and this continue to add to the growing call on the research community to address them.

Researchers in cybersecurity have focused attention towards the use of deep neural networks to develop more robust detection schemes to proactively safeguard our cyber space.

This study introduces a deep learning method for analyzing websites with malicious and benign characteristics.

we extracted and analyzed several URL features to formulate a supervised learning problem. Our network aims to learn important information from the input dataset and reconstructs the given URL sample using the hidden layers. The network only uses legitimate instances from the training data to extract the benign URL patterns and characteristics.

We have utilized binary cross entropy as loss function to distinguish between malicious and normal ULR instances. The choice of this loss function is based on the fact that benign samples will produce a low reconstruction error while a high error will be generated when using the malignant samples. Our model achieves accuracy of 89.72% on our dataset.

We perform several experiments on our data based on existing machine learning methods and compare the results with that of our deep learning model.

The following is a summary of this paper's contributions:

- We propose a deep learning method based on the radial basis function neural network to build a detection model for malicious websites. We have used only legitimate instances to train our model.
- We have extracted and analyzed several URL features to formulate a supervised learning problem and build a model to classify malignant and legitimate websites, based on network properties and application layer features.
- We compare our neural network-based model to state-of-the-art machine learning model including, Logistic Regressions, SVM Classifier, CAT boost Classifier and Random Forest Classifier. The result shows a superior performance in favour of our approach.

The rest of the paper is structured as follows. The literature on the identification of malicious websites is reviewed in ection 2. We present our methodology in section 3. Section 4 demonstrates the experimental analysis. Finally, the conclusion of the work is discussed in section 5.

2. Related Works

Researchers have proposed different approaches to analyze malicious URLs, and by extension malicious websites. Anomalous website detection methods can be categorized as malicious website recognition, malicious website source identification and malicious website mitigation. Studies on malignant websites recognition have previously been based on URL characteristics [2] using signature-based techniques [6,7,8], machine learning approaches [9,10,1] and more recently, deep learning methods [11,12,13,14]. The signature-based approaches maintain a database of known malignant URLs. If an unknown URL is accessed, there is a query triggered on the dataset and all compromised URLs are blacklisted. The shortfall of this methods is that new malignant URLs are hardly detected if they are not part of the list. Do et al [2], investigated a couple of machine learning based algorithms for building malicious URL detection models. The authors pointed out that characteristics of URLs is grouped into static and dynamic behaviours. A machine learning algorithm is selected and applied to build a detection model based on the nature of the URL characteristics.

Consequently, Ma et al [15], Eshete et al [16] and Purkait et al [17], extracted static features such as html content, lexical properties and host information of URLs and applied online learning algorithms [5,18] with support vector machines to static features. Also, the authors in [19, 20] extracted dynamic features of URLs such as system calls to detect compromised webpages. Machine learning techniques still suffer some limitations such that they are increasingly becoming ineffective at detecting malicious websites. Attackers have resorted to implementing sophisticated evasive techniques like using java scripts to plant malicious codes into webpages. These codes get executed at the client side, allowing the attackers to perform all manner of attacks.

Online learning algorithms have been much successful in this field. The recent surge in volumes of training data for suspicious websites detection, which comes in millions of features have prompted the research community to employ deep learning methods, which automatically learn the URL features as well as perform feature selection. Classification models are then built using these features.

The authors in [11,12,14] have successfully applied deep learning methods to detect malicious webpages in recent years. Even though deep learning methods have produced promising outcomes in the task, it requires complex computation for models to be trained. This challenge becomes more pronounced when new URL information are made accessible and the model needs to be retrained.

Our method employs a deep learning approach based on a radial basis function neural network for the recognition and detection of malicious websites. We suggest a framework in which each URL sample is assigned to either the legitimate URL or malicious categories modeled as a binary classification problem.

3. Materials and Methods

We first introduce the basic theory of the radial basis function network. Then we outline the architecture of our proposed neural network, along with the characteristics of model framework in this section.

3.1 Basic Theory of Radial Basis Function Neural Network.

Radial basis function network (RBFN) is a special type of deep neural network that employs radial basis functions as an activation function [24]. The output unit of this neural network is a linear combination of inputs and neuron parameters. RBF function is a real number function ψ, whose value depends on the distance between the inputs and some fixed point, such that $\psi(p) = \psi(\parallel p \parallel)$. Any function that is presented in the form, $\psi(p) = \psi(\parallel p \parallel)$ is recognized as a radial function. When they are expanded into the hidden space, the functions are then referred as radial basis functions.

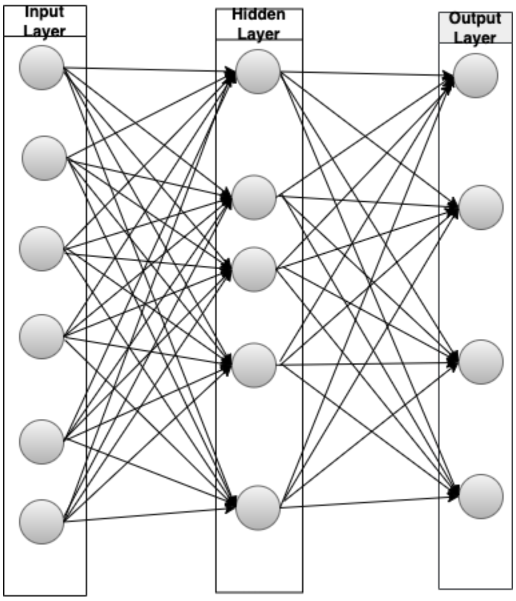

Fig. 1. Basic network architecture of RBFN

An RBFN is a three-layer network with input units, hidden units, and output layers at its core. It looks much similar to multiple layer deep feed forward neural network. The only difference is in the hidden layer, where instead of using some other activation functions usually employed in feed forward neural networks, we use the radial basis functions. RBFN neural network has a wide area for research, including function approximation, time series prediction, system control and classification problems.

3.2 Our Model

This section outlines the deep learning-based technique proposed for identifying malicious websites. We define websites detection as a classification problem by assigning a label indicating a normal or malicious website based on reconstruction features on our datasets. Deep learning approaches have been preferred over conventional machine learning methods in such tasks. They can represent the input feature exactly, by automatically extracting and distinguishing the features using many processing layers. The proposed model uses a radial basis function, which removes the important characteristics to estimate a decent approximation of the input feature space. The deep radial basis neural network, adopted for these studies comprises a single node that calculates a linear combination of inputs given a vector of real values as inputs, then outputs a 1 if the result is greater than some threshold and -1 if otherwise, defining an algorithm for supervised learning of binary classifier. We present the training methodology in figure 2.

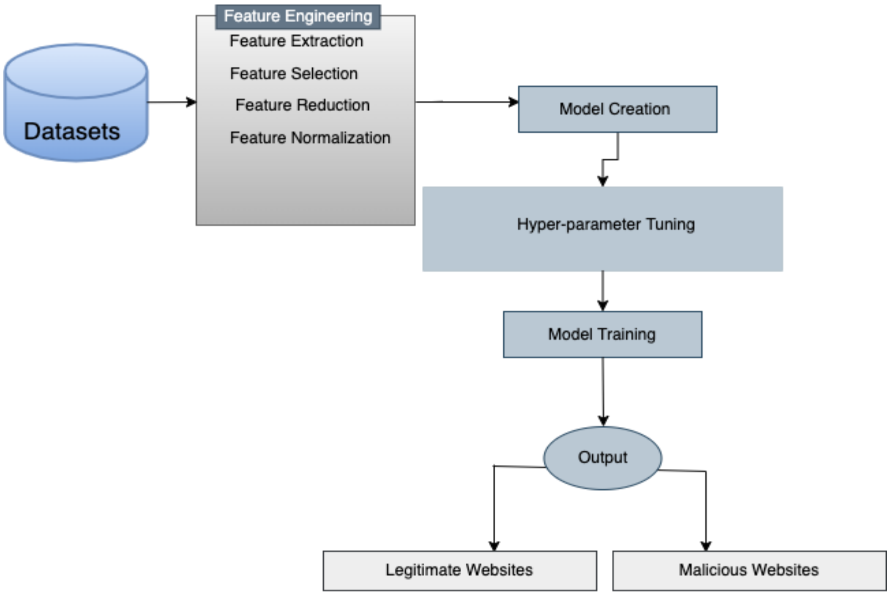

Fig. 2. The methodology of our Approach

Given the datasets $x_1, x_2,...,x_n$ as inputs, with the outputs $\mathrm{o}\left(x_1, x_2,...,x_n\right)$, given by the perceptron is computed by Eq. 1.

$$o\left(x_1, x_2,...,x_n\right) = \begin{cases} 1 \text{ if } w_0 + w_1 x_2 + w_2 x_2 +,...,+ w_n x_n > 0 \\ -1 \qquad\qquad \text{Otherwise} \end{cases} \tag{1}$$

where w_i is the weight that determines the contribution of input x_1 to the perceptron output. $w_1, w_2,..., w_n$, were the weights applied on the inputs, and w_0 is the bias as shown in figure 3. First, we calculate the summation term $\sum_{i=0}^{n} w_i x_i$, for all *i* in the range *0* to *n*. Once the summation is computed, we apply the activation function. O is the actual output which is compared against the targeted outputs. If both target outputs and the actual output are same, then $w_1, w_2,..., w_n$ are the final weights, otherwise, we need to go back and modify these weights, the same process is repeated. Learning the weights $\left(w_1, w_2,..., w_n\right)$ however remains a challenge. Start with a random weight and iteratively apply the perceptron to each training sample, modifying the perceptron weights if a sample is misclassified as one method of learning an acceptable weight vector. Once the perceptron successfully classifies all of the training data, this process is repeated iteratively as many times as feasible through the training samples. The weights are modified at each epoch according to the training rule of the perceptron which revises the weight w_i associated with input x_i, according Eq. 2.

$$w_i \leftarrow w_i + \Delta w_i \tag{2}$$

$$\Delta w_i \leftarrow \eta\left(t - 0\right) x_i \tag{3}$$

where η is the learning rate, which is assigned to a small value like 0.1. t is the target output, o is the actual output, and x_i is the input associated with w_i. Eq. 3 will modify the weights in each epoch. The procedure is repeated until all the samples are correctly classified.

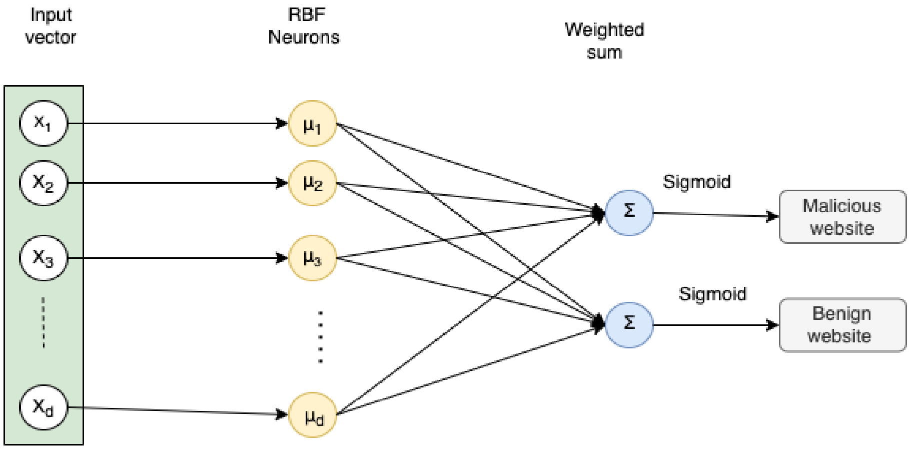

Fig. 3. Network Architecture of our proposed RBFN Neural Network.

Figure 3 presents the network architecture of our RBFN neural network, which is similar to multi-layer feedforward neural networks [22]. The architecture consists of an input vector, a layer of RBNF neurons at the hidden space and an output layer. The input vector is an n-dimensional vector denoting the website features collected and which we are going to classify. The entire input vector is shown to each of the RBFN neurons in the second layer. Each of the neurons maintains 3 internal prototype vector μ. Each RBNF neuron generates a value of 0 or 1 after comparing the input vector to its prototype. If the input vector matches prototype, the output of that neuron will be 1 (malicious website), otherwise it will be 0 (benign website). The outputs of the network consist of two nodes to perform a binary classification of websites. Here, the output nodes are showing a weighted sum Σ, and based on this weighted sum, each output node is going to calculate the category it belongs to (each output node computes some score). By placing the input in the category with the highest score, a classification decision is made. Due to the fact that each output node computes the score for a separate category, each output node has a unique set of weights. RBNF neurons that fall under its category will receive positive weight from the node, while those that don't will receive negative weight.

Mathematical explanation. RBNF network used in this analysis is similar to multiple layer feedforward neural networks. The distinction is in the hidden layer, where we use the radial basis function rather than the activation functions often used in feed forward neural networks [23]. We focus on one of the hidden neurons to provide a clearer explanation (i. e. kth node for example) in the RBNF neuron layer, shown in figure 4.

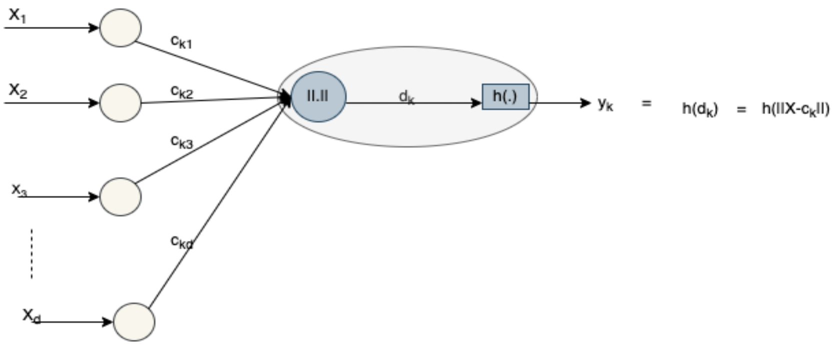

Fig. 4. kth hidden node as an example.

In a hidden node, we have two elements $\|.\|$ and the Euclidean distance d$_k$ defined by equation 4, that we are going to calculate. The connection weight c is used between the input layer and the hidden layer (the center). The use of the center in RBFNF network here, is different from the use of weight W in traditional multiple layer feedforward neural network.

$$d_k = \|x - c_k\| \tag{4}$$

where, $k = 1, 2, 3, ...$

$$c_k = \left[c_{k1}, c_{k2}, ..., c_{kd} \right] \tag{5}$$

$$x = \left[x_1, x_2, ..., x_d \right] \tag{6}$$

c_k can be determined by a training algorithm and

$$d_k = \left\| x - c_k \right\| = \sqrt{\left(x_1 - c_{k2} \right)^2 + \left(x_2 - c_{k2} \right)^2 + ... + \left(x_d - c_{kd} \right)^2} \tag{7}$$

Once we calculate the d_k and feed it into radial basis function $h(.)$, we can find y_k i. e. the output. We can represent as Eq. 8.

$$y_k = h\left(d_k \right) = \left(\left\| x - c_k \right\| \right) \tag{8}$$

The commonly used basis function is the Gaussian function defined as:

$$h\left(d_k \right) = e^{-\frac{d_k^2}{2\sigma^2}} \tag{9}$$

with parameters $\sigma > 0$. Therefore, the distance d$_k$ is calculated according to the input vector x (input features) as well as the connection center c. The sigmoid $\sigma > 0$ is the width of the Gaussian function which is defined by the training algorithm.

4. Experiments Analysis

We first provide the datasets used for our investigations in this section. We next go into the details regarding our experimental setup and the metrics used for the evaluation of performance of the resulting model. Finally, we present the experimental findings from our suggested model and contrast them with earlier studies.

4.1 Our Datasets

We used publicly available datasets obtained at Kaggle.com. We dynamically analysed web applications honeypot interactions. Various python scripts were crafted to generate the data of each URL, which were then systematically analyzed. Starting with 530181 URLs, each was verified and later reduced to 63191 samples with the help of request library in python.

4.2 Datasets pre-processing

Neural networks generally perform better when the data is scaled to the bounds of the activation function. Since we have a binary classification problem at hand, we used sigmoid activation function within the RBFN network. Therefore, we normalized the input data to the range of [0, 1]. We have used the Min-Max Normalization for each input feature, which rescales the range of features using Eq. 10.

$$X_i = \frac{q_i - \min(q)}{\max(q) - \min(q)} \tag{10}$$

Where, X_i is a d-dimensional feature vector taken from the training samples, and q_i is the i^{th} normalized data point.

We created a column in the malicious data so that the column values of the malicious data will be 1 and column in the legitimate website data has values of 0. This gives us a supervised learning problem such that if the value is 0, it is legitimate data and if the value is 1, it is a malicious data. We then combined those two datasets along the $y-\text{axis}$. We take out the y-values as our ground truth values and the rest of the data used for training the model. We finally convert the data into a 1-dimensional array from n-dimensions. We now have a dataset comprising 1781 rows and 21 columns.

Table 1 presents a brief description of selected features of the dataset.

Table 1. Data Description

Feature/Variable	Description of values associated with variables
URL	unknown URLs identified and analyzed
URL_LENGTH	character count in the URL
NUMBERSPECIALCHARACTERS	amount of special characters found in the URL example: # , ˙ , ´ , ¯ , {},
CHARSET	significance of categorical value in character encoding
SERVER	values indicating operating system of server that produced packet response
CONTENT_LENGTH	HTTP header’s content size
WHOIS_COUNTRY	values here indicate server response countries
WHOIS_STATEPRO	values indicate the states reported from the server response
WHOIS_REGDATE	date server was registration (format: DD/MM/YYY HH:MM)
WHOISUPDATEDDATE	current updated date from analyzed server
TCPCONVERSATIONEXCHANGE	amount of TCP packets communicated between server and honeypot client
DISTREMOTETCP_PORT	number of ports detected that are different from TCP
REMOTE_IPS	Overall number of IPs connected to honeypot
APP_BYTES	number of transferred bytes
SOURCEAPPPACKETS	source of packets sent from honeypot to server
REMOTEAPPPACKETS	number of packets received from server
APP_PACKETS	overall number of IP packets generated during communication between honeypot and server
DNSQUERYTIMES	DNS packets generated during communication between honeypot server
TYPE	values indicating type of web page analyzed (1= malicious websites, and 0 = legitimate websites

Obviously, the data had lots of issues that needed to be cleaned up where we deemed appropriate. For example, the URL column was dropped since it’s a unique identifier. Also, columns like DNSQUERYTIMES, and server had null values and so dummy values were used to replace those nulls. Yet again, the data had lots of categorical features like "TCP_CONVERSATION_EXCHANGE","URL_LENGTH","APP_BYTES","SOURCE_APP_PACKETS","REMOTE_APP_PACKETS","SOURCE_APP_BYTES","REMOTE_APP_BYTES”, so these were ignored in our model. Highly correlated features such as in the case of longer URLs, which had the tendency to contain many special characters were also dropped. With the dataset cleaned up, it was scaled and split into train-test-split, with test size of 30% and the remaining 70% used for training the network.

4.3. Metrics used for performance evaluation

By using the standard SciKit Learn class from python language, we evaluated the performance of the machine learning models and our proposed model based on Accuracy (Acc), Precision (Prec), Recall (Rec) and F-Score(F1). We tracked the classification report and the confusion matrix. These evaluation methods were defined by the following:

$$A_{cc} = \frac{TP + TN}{TP + TN + FP + FN} \tag{11}$$

$$P_{rec} = \frac{TP}{TP + FP} \tag{12}$$

$$P_{rec} = \frac{TP}{TP + FN} \tag{13}$$

$$F_1 = \frac{\text{Pres} * \text{Rec}}{\text{Prec} + \text{Rec}} * 2 \tag{14}$$

Where:
TP = True Positive representing the number of instances correctly predicted as malignant websites.
TN = True Negative, representing the number of instances correctly predicted as legitimate websites.
FP = False Positive False, represents the number of instances incorrectly predicted as malignant websites.
FN = False Negative, represents the number of instances incorrectly predicted as legitimate websites.

4.4 Experimental set up

All experiments in this study were conducted on a physical machine, having the following specifications: Intel Core i5,1.4 Quad-Core processor with 8 GB 2133 MHz LPDDR3 memory, on MacBook Pro (2020 edition), running macOS Monterey version 12.4 (64 bit) operating system. The TensorFlow deep learning framework was used to build the RBFN neural network-based model, in which all the RBNF-based algorithms were implemented in python language. Specifically, the TensorFlow version 1.2.3, and python version 3.7.7 were used.

4.5 Experimental Results

We compared the detection performance of the RBFN-based model with some of the most well-established machine learning methods, including Logistic Regressions, SVM Classifier, CAT boost Classifier and Random Forest Classifier, to evaluate our proposed model on the datasets. We analyzed precision, recall, F-score, and accuracy values of all the approaches. For fairness's sake, we trained the machine learning models under the same condition with the same pre-processed dataset and environment described earlier. The comparison results are presented in Table 2.

Table 2. Further improvement using different activation functions

Machine Learning Classifier	Accuracy	Precision	Recall	F1Score
Logistic Regressions	0.8625	0.3353	0.580	0.5238
SVM Classifier	0.8832	0.6129	0.2754	0.3800
CAT boost Classifier	0.9016	0.6364	0.8227	0.7126
Random Forest Classifier	0.8958	0.6438	0.7872	0.6954

Table 3. Results obtained by RBNF deep neural network model

Accuracy	Precision	Recall	F1Score
0.8972	**0.7129**	**0.77764**	**0.6845**

Confusion Matrix $[450 \quad 12][50 \quad 19]$

The results obtained by the simple deep learning model shown in table 3 indicates a slight improvement, which means it performs well at classifying the malignant websites as can be seen also in the confusion matrix and the F1 score. Based on the results obtained by the trained RBNF neural network, we tried to further improve on this by adjusting several of its parameters and using different optimization algorithms, including limited memory BFGS optimization algorithm (L-BFGS), Stochastic gradient descent (SGD), and Adam.

Table 4. Model improvement using different optimizers (solvers).

	Activation function (Solver)	Accuracy	F1Score
0	L - BFGS	87.947269	52.380952
1	SGD	88.710565	47.948718
2	Adam	88.353917	39.420040

From Table 4, it is clear that the L-BFGS optimizer is the best solver, producing the highest accuracy and F1 scores. Therefore, this optimer may be adopted and used for further model improvements. In Table 5, we used the L-BFGS optimizer with several other activation functions, including Identity activation, logistic, tanh, ReLU. The results further shows that the logistic activation function produced 90.2% accuracy and 57.4% F1 score

Table 5. Further improvement using different activation functions

	Activation function	Accuracy	F1Score
0	Identity activation	86.440678	16.279070
1	Logistic activation	90.207156	57.377049
2	Tanh activation	89.077213	55.384615
3	Relu activation	88.700565	52.380952

The goal for our proposed model is to find a more efficient model that can classify malignant websites. By optimizing and adjusting several parameters, our proposed RBNF based model has proven to be very effective. Additionally, as shown in Table 2, we contrast the performance of the suggested strategy with baseline machine learning models. So, the base machine learning model was CAT boost classifier that produced 90.16% accuracy. However, the model produces poor precision and recall scores. It can be seen from the confusion matrix, the model is good at predicting legitimate websites, but poor at predicting malignant websites, this makes the machine learning models ineffective if we were to implement the models in real-world applications. It can however be seen from Table 3 that deep learning model can considerably improve on predicting malicious websites.

5. Conclusion and Future Works

Traditional malicious website detection methods fail to classify sophisticated URLs with unexpected patterns, hence detection of the resulting roque websites has become one of the most challenging tasks in the internet usage. This study proposes a new deep learning-based detection method, focusing on malicious URLs. Our proposed deep neural network framework is based on radial basis function that can effectively model both legitimate and malicious websites. It detects the abnormality from the dataset using binary cross entropy as a function to map inputs to the output.

Our experimental analysis shows that the proposed model can detect abnormality in URL data. It is our intension to extend this binary classification task into a multiclass classification problem, where specific malicious website types would be determined.

References

[1] D. Sahoo, C. Liu, and S. C. Hoi, “Malicious url detection using machine learning: A survey,” arXiv preprint arXiv:1701.07179, 2017.

[2] C. Do Xuan, H. D. Nguyen, and V. N. Tisenko, “Malicious url detection based on machine learning,” International Journal of Advanced Computer Science and Applications, vol. 11, no. 1, 2020.

[3] A. D. Gabriel, D. T. Gavrilut, B. I. Alexandru, and P. A. Stefan, “Detecting malicious urls: A semi-supervised machine learning sys Use the "Insert Citation" button to add citations to this document. tem approach,” in 2016 18th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), pp. 233–239, IEEE, 2016.

[4] F. O. Catak, K. Sahinbas, and V. Dörtkarde¸s, “Malicious url detection using machine learning,” in Artificial intelligence paradigms for smart cyber-physical systems, pp. 160–180, IGI Global, 2021.

[5] S. C. Hoi, D. Sahoo, J. Lu, and P. Zhao, “Online learning: A comprehensive survey,” Neurocomputing, vol. 459, pp. 249–289, 2021.

[6] C. Seifert, I. Welch, and P. Komisarczuk, “Identification of malicious web pages with static heuristics,” in 2008 Australasian Telecommunication Networks and Applications Conference, pp. 91–96, IEEE, 2008.

[7] S. Sheng, B. Wardman, G. Warner, L. Cranor, J. Hong, and C. Zhang, “An empirical analysis of phishing blacklists,” 2009.

[8] S. Sinha, M. Bailey, and F. Jahanian, “Shades of grey: On the effectiveness of reputation-based “blacklists”,” in 2008 3rd International Conference on Malicious and Unwanted Software (MALWARE), pp. 57–64, IEEE, 2008.

[9] D. Wang, S. B. Navathe, L. Liu, D. Irani, A. Tamersoy, and C. Pu, “Click traffic analysis of short url spam on twitter,” in 9th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing, pp. 250–259, IEEE, 2013.

[10] L. Xu, Z. Zhan, S. Xu, and K. Ye, “Cross-layer detection of malicious websites,” in Proceedings of the third ACM conference on Data and application security and privacy, pp. 141–152, 2013.

[11] E. Benavides, W. Fuertes, S. Sanchez, and M. Sanchez, “Classification of phishing attack solutions by employing deep learning techniques: A systematic literature review,” Developments and advances in defense and security, pp. 51–64, 2020.

[12] H. Le, Q. Pham, D. Sahoo, and S. C. Hoi, “Urlnet: Learning a url representation with deep learning for malicious url detection,” arXiv preprint arXiv:1802.03162, 2018.

[13] W. Yang, W. Zuo, and B. Cui, “Detecting malicious urls via a keyword-based convolutional gated-recurrent-unit neural network,” IEEE Access, vol. 7, pp. 29891–29900, 2019.

[14] J. Saxe and K. Berlin, “expose: A character-level convolutional neural network with embeddings for detecting malicious urls, file paths and registry keys,” arXiv preprint arXiv:1702.08568, 2017.

[15] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, “Identifying suspicious urls: an application of large-scale online learning,” in Proceedings of the 26th annual international conference on machine learning, pp. 681–688, 2009.

[16] B. Eshete, A. Villafiorita, and K. Weldemariam, “Binspect: Holistic analysis and detection of malicious web pages,” in International conference on security and privacy in communication systems, pp. 149–166, Springer, 2012.

[17] S. Purkait, “Phishing counter measures and their effectiveness–literature review,” Information Management & Computer Security, 2012.

[18] S. C. Hoi, J. Wang, and P. Zhao, “Libol: A library for online learning algorithms,” Journal of Machine Learning Research, vol. 15, no. 1, p. 495, 2014.

[19] Y. Tao, Suspicious URL and device detection by log mining. PhD thesis, Applied Sciences: School of Computing Science, 2014.

[20] G. Canfora, E. Medvet, F. Mercaldo, and C. A. Visaggio, “Detection of malicious web pages using system calls sequences,” in International Conference on Availability, Reliability, and Security, pp. 226–238, Springer, 2014.

[21] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, et al., “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,” IEEE Signal processing magazine, vol. 29, no. 6, pp. 82–97, 2012.

[22] D. Svozil, V. Kvasnicka, and J. Pospichal, “Introduction to multi-layer feed-forward neural networks,” Chemometrics and intelligent laboratory systems, vol. 39, no. 1, pp. 43–62, 1997.

[23] M. T. Musavi, W. Ahmed, K. H. Chan, K. B. Faris, and D. M. Hummels, “On the training of radial basis function classifiers,” Neural networks, vol. 5, no. 4, pp. 595–603, 199

[24] Y. Wang, Y. Li, Y. Song, and X. Rong, “The influence of the activation function in a convolution neural network model of facial expression recognition,” *Appl. Sci.*, vol. 10, no. 5, p. 1897, 2020.

Authors' Profiles

Mustapha Adamu Mohammed is currently pursuing his PhD in Computer Science. He received his M.Phil. degree in Computer Science from the Kwame Nkrumah University of Science and Technology. His research interest is the area of big data and statistics, machine learning and deep learning.

Seth Alornyo obtained his Ph.D in Software Engineering at University of Electronic Science and Technology of China (UESTC) and an M.Phil. Degree in Computer Science from the Kwame Nkrumah University of Science and Technology. His research interests include Cryptography and Network Security. He is a member of IEEE and a reviewer to selected international journals.

Michael Asante is an Associate Professor of Computer Science at the Department of Computer Science, Kwame Nkrumah University of Science and Technology. His research areas include Computer Security, Cyber Security, and Networking.

Bernard Obo Essah holds an MPhil., BSc. & HND all in Statistics and PGD (Mathematics Education). His research interests include machine learning and algorithms, Image pattern recognition, and deep learning, big data and mathematical statistics, multidimensional data visualization and analysis. Software interest includes Stata, R and Python.

How to cite this paper: Mustapha A. Mohammed, Seth Alornyo, Michael Asante, Bernard O. Essah, "Intelligent Detection Technique for Malicious Websites Based on Deep Neural Network Classifier", International Journal of Education and Management Engineering (IJEME), Vol.12, No.6, pp. 45-54, 2022. DOI:10.5815/ijeme.2022.06.05