Ujah-Ogbuagu *et al.*
Journal of Electrical Systems and Inf Technol (2024) 11:7
https://doi.org/10.1186/s43067-023-00128-8

Journal of Electrical Systems and Information Technology

RESEARCH Open Access

A hybrid deep learning technique for spoofing website URL detection in real-time applications

Bridget C. Ujah-Ogbuagu[1*], Oluwatobi Noah Akande[2] and Emeka Ogbuju[3]

*Correspondence:
bcujah-ogbuagu@ndc.gov.ng

[1] Department of Science and Technology, Centre for Strategic, Research and Studies, National Defence College, Abuja, Nigeria
[2] Department of Computer Science, Baze University, Abuja, Nigeria
[3] Department of Computer Science, FU Lokoja, Lokoja, Koggi State, Nigeria

Abstract

Website Uniform Resource Locator (URL) spoofing remains one of the ways of perpetrating phishing attacks in the twenty-first century. Hackers continue to employ URL spoofing to deceive naïve and unsuspecting consumers into releasing important personal details in malicious websites. Blacklists and rule-based filters that were once effective at reducing the risks and sophistication of phishing are no longer effective as there are over 1.5 million new phishing websites created monthly. Therefore, research aimed at unveiling new techniques for detecting phishing websites has sparked a lot of interest in both academics and business with machine and deep learning techniques being at the forefront. Among the deep learning techniques that have been employed, Convolutional Neural Network (CNN) remains one of the most widely used with high performance in feature learning. However, CNN has a problem of memorizing contextual relationships in URL text, which makes it challenging to efficiently detect sophisticated malicious URLs in real-time applications. On the contrary, Long Short-Term Memory (LSTM) deep learning model has been successfully employed in complex real-time problems because of its ability to store inputs for a long period of time. This study experiments with the use of hybrid CNN and LSTM deep learning models for spoofing website URL detection in order to exploit the combined strengths of the two approaches for a more sophisticated spoofing URL detection. Two publicly available datasets (UCL spoofing Website and PhishTank Datasets) were used to evaluate the performance of the proposed hybrid model against other models in the literature. The hybrid CNN-LSTM model achieved accuracies of 98.9% and 96.8%, respectively, when evaluated using the UCL and PhishTank datasets. On the other hand, the standalone CNN and LSTM achieved accuracies of 90.4% and 94.6% on the UCL dataset, while their accuracies on the PhishTank dataset were 89.3% and 92.6%, respectively. The results show that the hybrid CNN-LSTM algorithm largely outperformed the standalone CNN and LSTM models, which demonstrates a much better performance. Therefore, the hybrid deep learning technique is recommended for detecting spoofing website URL thereby reducing losses attributed to such attacks.

Keywords: Deep learning, Spoofing URL detection, Textual feature extraction and hybrid CNN-LSTM

Introduction

In today's digital world, the internet has brought many benefits to mankind and has enriched lives and our daily activities. Most of our daily activities such as communications, businesses, marketing, education, traveling, and shopping are internet-based. As a result of the massive growth of the Internet due to its heavy usage, the need to share personal information online has rapidly grown. This has made many individuals and organizations vulnerable to cyber insecurity. Unfortunately, the internet is also being used by criminals to commit many cybercrimes including spoofing websites. Spoofing attack is an example of social engineering technique attack in which a fraudulent message is sent via email, social media chat applications, or mobile text to a victim to trick them to reveal sensitive information. A more adverse effect was felt given the twist the COVID-19 Pandemic brought to internet leverage, leading to incremental growth in internet usage. According to a cybercrime fraud report, spoofing attacks rose from 56 to 220% during and after the COVID-19 Pandemic [29]. In a spoofing attack, cybercriminals attempt to steal personal information, such as user login credentials and financial details, from individuals or an organization for fraudulent or malicious use [7]. The first known instance of a spoofing attack happened in the late 90 s when a group of hackers hacked and stole personal information and login credentials from AOL users [2]. In early 2000, cyber criminals turned their attention more to financial systems, following an attack launched on E-Gold in June 2001 [24]. By the year 2003, the criminals took to registering several domain names that appeared like the names of popular legitimate commercial sites, such as Paypal and eBay, followed by sending mas mass emails to the customers of the websites, asking them to visit the websites and provide their sensitive details such as login and credit card details [14]. In the year 2020, Google registered and blacklisted 2.02 million malicious websites, which increased to 19% from 2019. CISCO reported that 90% of data breaches trend in 2021 resulted from spoofing attacks [29]. Spoofing attacks are a major issue of serious concern all over the world. As the attacks are becoming increasingly complicated, prevention is also becoming more and more sophisticated. Various methods have been used to overcome spoofing attacks, such as legal, awareness, education, and blacklisting using visual similarity, and search engine barring. All these can be categorized into the traditional approach. Another method involves the non-traditional approach involving the use of Machine Learning (ML) algorithms, Intelligence (AI), content-based, heuristics, data mining, and even fuzzy-rule-based methods [9, 11].

The traditional approach is often limited due to the complexity and sophistication of the attackers. One of the most commonly used techniques by modern browsers to detect and blacklist spoofing websites. This approach is limited because it does not detect zero-day attacks, which refer to cyber-attacks that target software vulnerabilities that are unknown to the software vendor or developer [8]. Zero-day attacks pose significant challenges for cybersecurity because the attackers exploit vulnerabilities that are unknown to defenders and are thus difficult to mitigate [25]. The ML-based techniques have also been useful in detecting fake URLs. The ML-based approach is specifically a supervised ML approach in which URLs are first acquired and analyzed, following feature extraction, and finally the building of a training set with labels, using the features extracted [23]. The major challenge with the ML approach is that the accuracy is

affected if the features of the URLs are not carefully extracted and analyzed [31]. The deep learning approach is the newest ML approach, which has steps further above the normal ML approach, has equally been recently employed in spoofing URL detection [26]. The results of such studies show a considerable improvement over normal ML method [26]. Examples of the deep learning methods that have recently been employed in spoofing URLs detection includes the Convolutional Neural Networks (CNNs) and the Recurrent Neural Networks (RNNs) [3, 26]. The two state-of-the-art approaches undoubtably resulted in better results in spoofing URL detection; however, they are also limited in various ways [26]. The CNN, for instance, is good in extracting the intrinsic features in texts [12], but has a problem of memorizing contextual relationships in text, which makes it challenging to efficiently detect sophisticated malicious URLs in real-time applications [17]. On the other hand, the RNN such as Long Short-Term Memory (LSTM) performs better in textual memorization because of its ability to store inputs for a long period of time [16]. However, the LSTM still has a small challenge with extracting textual features in text such as the URL [26]. Hence, leveraging the collective strengths of the two deep learning algorithms could produce a sophisticated spoofing URL detection method best suited for real-life applications in the era of digital explosion.

This study proposes a hybrid deep model composed of CNN and LSTM for malicious URLs detection in real-time applications. The model leverages on the great ability of the CNN in extracting features in text and the LSTM being a sequential model remembers sequential input such as URLs over a long period. The novel hybrid deep learning spoofing URLs detection model eliminated the identified limitations of various spoofing URL websites detection approaches in the literature after training and testing the hybrid model on publicly available datasets. The proposed hybrid model was evaluated on Accuracy, Precision, Recall, F1-score, and Loss, as against the state-of-the-art approaches in the literature.

The rest of the paper is organized in the following structure: Sect. 2 presents the review of related works. A conceptual understanding and the methodology of the proposed model are provided in Sect. 3. Section 4 discusses the implementation decision, experiments, results, and comparative evaluation with different algorithms. Finally, Sect. 5 provides the conclusion and suggestions direction for future work.

Related works

Mao et al. [19] introduced a mechanism for detecting phishing pages based on the visual resemblance of web pages. Features were extracted from the Cascading Style Sheet (CSS) of web pages. The most relevant features were carefully chosen for grading the websites similarities. The proposed technique was validated using 9307 authenticated phishing websites. The dataset included spoofing URLs that attempt to deceive users of Paypal, eBay, Apple, and other well-known websites. Their method surpassed the performance of six other related techniques in literature.

In 2020, Li et al. [18] proposed linear and nonlinear space-transformation-based methods for malicious URLs detection using feature engineering. The study developed a two-stage distance metric technique for linear transformation, while the second stage was used for kernel approximation for linear and nonlinear transformations. In 2021, a functional tree meta-learning mechanism for spoofing site detection was proposed

by Balogun et al. [6]. In the study, the authors recommended a functional tree-based for detecting spoofing and legitimate websites given the achieved accuracy. Xiao et al. [30] proposed a method to detect a phishing URL website using a self-attention CNN algorithm. A generative adversarial network (GAN) deep learning model was used to produce an imbalanced dataset for the model, in which the CNN model was combined with multithread self-attention to build a classification mode [22]. A predictive ML-based model was introduced to classify websites as spoofing or genuine, by Abedin et al. [1]. The study proposed an ML-based system to detect fake URLs using Support Vector Machines (SVM), CNN, and K-Nearest Neighbor (KNN) ML algorithms. Feng et al. [12] implemented an IA-based classification model with a Monte Carlo algorithm to detect spoofing websites. A model for detecting spam emails was introduced by Smadi et al. [26]; the authors proposed a technique that adds spoofing emails to corpus datasets. Haynes et al. [15] proposed a lightweight URL-based spoofing detection method using Natural Language Processing (NLP) transformers for mobile devices. The study applied Artificial Neural Network (ANNs) model to URLs and HTML-based website features to distinguish fake from legitimate URLs. Gandotra and Gupta [13] presented a method for enhancing the detection of spoofed websites using machine learning. In the study, the authors utilized a wide range of web-page, URL, and HTML elements to identify counterfeit websites. The features were initially employed independently to categorize webpages, and subsequently, all the traits were combined for the purpose of classification. The findings indicate that the attributes categorized based on URL are the most efficient in classifying the webpages. Hence, the proposed approach brought a substantial enhancement in classification accuracy. The random forest classifier emerges as the most effective, with an accuracy of 99.5%, and a False Positive Rate (FPR) is 0.006 and the false negative rate (FNR) is 0.005.

In 2022, a heuristic-based regression method combined with a decision tree algorithm and a wrapper feature selection model approach was introduced by Babagoli et al. [5], to detect spoofing websites ULRs. In the study, the authors used eight ML algorithms for the evaluation and reported that the principal component analysis random forest algorithm achieved the highest accuracy on image analysis. Yasin and Abuhasan [31] developed a classification model for detecting phishing emails. In the study, a Java program was used to extract the features from emails header and body, after which a data mining algorithm was applied to extract features to train the model to determine which of the ML algorithms achieved the best results. Das et al. [10] presented a hybrid feature-based anti-spoofing technique that focuses on extracting features solely from the URL and hyperlink information on the client-side. The extracted features were merged to form a hybrid feature set, which was used to train the classification models employing Random Forest, Decision Tree, Support Vector Machine, Logistic Regression, and XG Boost machine learning classifiers. The XG Boost classifier yielded the most favorable outcome with superior metrics. Almutair and Alshoshan [4] developed a tool which facilitates verification of the received URLs by utilizing a pre-established white-list database to ascertain their legitimacy. The proposed tool assesses any given URL using the database. The tool does not categorize any URL that is not present in the database as phishing, rather such URL is classified as unknown. It was reported that the assessment showed that the tool reached a 96.8% accuracy in identifying both genuine positive and

true negative cases. However, the tool requires enhancement in order to increase its reliability when it comes to handling unfamiliar URLs.

A deep learning approach for detecting phishing sites was proposed by Aldakheel et al. [3]. The approach utilizes a Convolution Neural Network (CNN)-based model for precise distinction between legitimate and phishing websites. The performance of the proposed model was evaluated based on Uniform Resource Locators (URL) features. The proposed model specifically comprises seven layers, beginning with the input layer down to the seventh layer, which incorporates a layer with pooling convolution, and output layers. The CNN-based model distinguishes phishing websites from legitimate websites with an accuracy of 95.77%.

The gaps identified in the reviewed literature indicated that the spoofing URL methods proposed by the various studies are not suitable for real-time applications because of their low sophistication. Spoofing URL detection in real-time applications requires a highly sophisticated technique to counter the highly sophisticated contemporary spoofing attacks. In this study, a hybrid CNN-LSTM deep learning model is proposed for detecting malicious URLs. The hybrid CNN-LSTM approach leverages on the strength of LSTM in holding unto its sequential input such as URLs for a long period and the high efficiency of the CNN in feature extraction of the URLs features. The proposed model utilized both character embedding of the textual URL features to explores the intricate relationships between URL characters at a higher level to develop a sophisticated hybrid deep learning model for efficient detection of malicious websites in real-time applications.

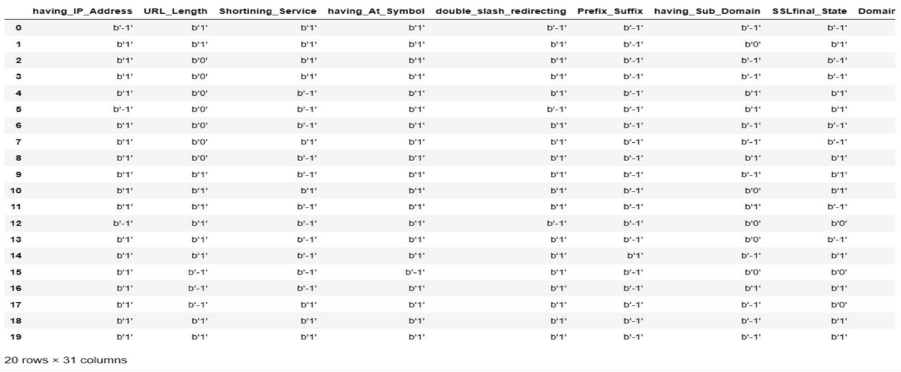
The proposed hybrid CNN-LSTM model

The methodology employed in this study involves four major steps as shown in Fig. 5. The steps include data collection, data preprocessing, model development and model evaluation.

Data collection

The study used two datasets to validate the model proposed. The first dataset was acquired from UCL Spoofing Websites Dataset repository [20, 27].The repository contains a collection of datasets extracted from a set of legitimate and phishing websites. The UCL dataset provides a labeled target variable that indicates if a URL is legitimate or not, which makes it suitable for training a supervised learning model. The UCL spoofing website dataset consists of 11,055 instances of genuine and malicious URLs, with 31 features relating to websites' domain and URL characteristics. Examples of the attributes include: "having IP Address", "URL length", "shortening service", "having "@" symbol", "prefix/suffix separation", "SSL final state and so on. The sample of the UCL dataset is shown in Fig. 1.

The PhishTank Dataset contains 30,647 spoofing websites and 27, 998 authentic website URLs totaling 58,645 instances [21]. The dataset consists of a group of (6) features based on URL properties, domain properties, URL dictionary, URL file name, URL Parameters, resolving URL and external services. There are a total of 112 features in the dataset Fig. 2 is the screenshot of the first 10 rows of PhishTank dataset.



	having_IP_Address	URL_Length	Shortining_Service	having_At_Symbol	double_slash_redirecting	Prefix_Suffix	having_Sub_Domain	SSLfinal_State	Domain
0	b'-1'	b'1'	b'1'	b'1'	b'-1'	b'-1'	b'-1'	b'-1'	
1	b'1'	b'1'	b'1'	b'1'	b'1'	b'-1'	b'0'	b'1'	
2	b'1'	b'0'	b'1'	b'1'	b'1'	b'-1'	b'-1'	b'-1'	
3	b'1'	b'0'	b'1'	b'1'	b'1'	b'-1'	b'-1'	b'-1'	
4	b'1'	b'0'	b'-1'	b'1'	b'1'	b'-1'	b'1'	b'1'	
5	b'-1'	b'0'	b'-1'	b'1'	b'-1'	b'-1'	b'1'	b'1'	
6	b'1'	b'0'	b'-1'	b'1'	b'1'	b'-1'	b'-1'	b'-1'	
7	b'1'	b'0'	b'1'	b'1'	b'1'	b'-1'	b'-1'	b'-1'	
8	b'1'	b'0'	b'-1'	b'1'	b'1'	b'-1'	b'1'	b'1'	
9	b'1'	b'1'	b'-1'	b'1'	b'1'	b'-1'	b'-1'	b'1'	
10	b'1'	b'1'	b'1'	b'1'	b'1'	b'-1'	b'0'	b'1'	
11	b'1'	b'1'	b'-1'	b'1'	b'1'	b'-1'	b'1'	b'-1'	
12	b'-1'	b'1'	b'-1'	b'1'	b'-1'	b'-1'	b'0'	b'0'	
13	b'1'	b'1'	b'-1'	b'1'	b'1'	b'-1'	b'0'	b'-1'	
14	b'1'	b'1'	b'-1'	b'1'	b'1'	b'1'	b'-1'	b'1'	
15	b'1'	b'-1'	b'-1'	b'-1'	b'1'	b'-1'	b'0'	b'0'	
16	b'1'	b'-1'	b'-1'	b'1'	b'1'	b'-1'	b'1'	b'1'	
17	b'1'	b'-1'	b'1'	b'1'	b'1'	b'-1'	b'-1'	b'0'	
18	b'1'	b'1'	b'1'	b'1'	b'1'	b'-1'	b'-1'	b'1'	
19	b'1'	b'1'	b'1'	b'1'	b'1'	b'-1'	b'-1'	b'1'	

20 rows × 31 columns

Fig. 1 A screenshot of the UCL dataset. The second dataset is the PhishTank Dataset which was produced by a collaborative project that collects and shares phishing URLs reported by the community [28]

```
dataset = pd.read_csv('PhishTank.csv') # loads the dataset from file
dataset.head(10) # shows first five rows of dataframe
```

	qty_dot_url	qty_hyphen_url	qty_underline_url	qty_slash_url	qty_questionmark_url	qty_equal_url	qty_at_url	qty_and_url	qty_exclamation_url	qty_space_url
0	3	0	0	1	0	0	0	0	0	0
1	5	0	1	3	0	3	0	2	0	0
2	2	0	0	1	0	0	0	0	0	0
3	4	0	2	5	0	0	0	0	0	0
4	2	0	0	0	0	0	0	0	0	0
5	1	0	0	2	0	0	0	0	0	0
6	2	0	0	0	0	0	0	0	0	0
7	2	0	0	3	0	0	0	0	0	0
8	2	0	0	0	0	0	0	0	0	0
9	1	0	0	2	0	0	0	0	0	0

10 rows × 112 columns

Fig. 2 A screenshot of the PhishTank dataset

Data preprocessing

The preprocessing process involved dataset normalization, tokenization and word embedding. The first preprocessing attempt was done to ensure that there are no missing values in the dataset. Afterward, the numerical features were normalized. Tokenization process employed involves URLs character transformation into a numerical form. Here, character-level text tokenization is done in which the URLs texts are vectorized by the Tokenizer function of the Keras library. To ensure that the variable length sequence is the same, sixty (60) was used as the fixed length of URLs considering the average length of URLs (25 to 50) characters. This implies that any input URL greater than 60 will be truncated, and any input URL length less than the fixed 60 lengths will be padded with the appropriate number of zeros. The converted sequence of numbers was then turned into an embedding which is finally is transferred to the CNN layer to maximize local features extraction before passing to the LSTM layer for learning temporal or long-term dependencies and capturing contextual information. The dataset was split in the ratio of 8:2 for training and testing, respectively. This implies that 80% was for training and 20% for testing or evaluation on real-world data.

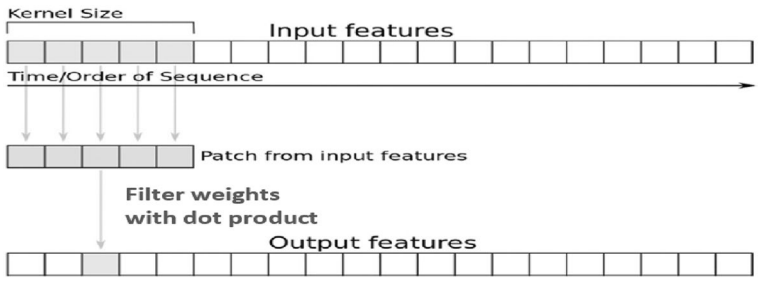


Fig. 3 A 1-D convolution operation of the input arrays [12]

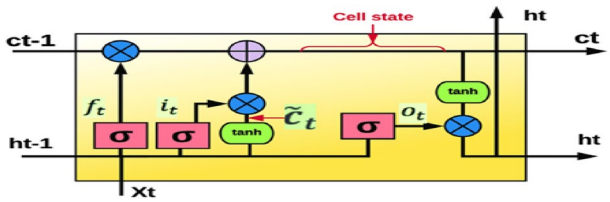


Fig. 4 Architecture of LSTM [9]

The hybrid CNN-LSTM model development

The hybrid model proposed involved CNN and LSTM deep learning techniques.

Convolutional neural network (CNN)

A Convolutional Neural Network (CNN) involves the multiplication of matrices that provide outputs to incorporate for the further training process [17]. This method is known as convolution and that is why this type of neural network is called a convolutional neural network [9]. In the case of URL text processing, the texts in a ULR are represented as word vectors used for training the CNN. A CNN can be one or multi-dimensional. In this study, a one-dimensional CNN (Conv1D) as shown in Fig. 3 was used for text feature extraction [12]. The CNN Conv1D operation entails input, convolution, pulling, and flattening. The convolution operation involves applying filters to the input sequence. Each filter has a fixed width of 64 and can be seen as a sliding window that moves across the input sequence.

After the convolution operation, the pooling operation is applied to reduce the dimensionality and extract important features. Max pooling is a used technique in Conv1D.

After the pooling operation, the resulting feature map can be flattened into a one-dimensional vector to be passed into subsequent layers. This flattening operation simply reshapes the pooled feature map.

Long short-term memory (LSTM)

RNNs are a type of neural network that is good for sequential data prediction. However, as RNNs process many steps, they are susceptible to vanishing gradients [22]. LSTMs are a way of overcoming the challenges of standard RNNs. LSTM was proposed in 1997 by Hochreiter and Schmidhuber to overcome the shortcoming of RNN [16]. LSTM is

capable of learning long-term dependencies and remembering input information for a long period through gates. As shown in Fig. 4, it is composed of a cell state, input, and output gates. The structural architecture of the LSTM is shown in Fig. 4. In the LSTM, word embedding makes it possible to represent text as vectors, which are referred to as word vectors with each word having a unique word vector. This is because when dealing with text classification and neural networks, the input texts must take a vector (matrix numeric) format so that they can be fed to the network. These word vectors are referred to as word embeddings.

where C_t = Cell state, C_{t-1} = previous cell state, f_t = forget gate, i_t = input gate, O_t = Output gate. h_t = hidden state, h_{t-1} = Previous hidden state, x_t = Input at timestep, tanh = hyperbolic tangent activation function, $C\tilde{}_t$ = Cell update, σ = sigmoid activation function, y_t = output.

The hybrid CNN-LSTM model for spoofing URLs detection

In the CNN network, a filter of fixed size window iterates through the training data, which at each step, multiplies the input with the filter weights and gives an output that is stored in an output array. This output array is a feature map output filter of the data. In this way, a feature was detected from the input training data as shown in Fig. 3. The size of the filter is specified as kernel size and the number of filters specifies the number of feature maps to be used. This is how the CNN was utilized to learn local features in the URL text that is directly derived from the training data. First, the Conv1D processes the input vectors and extracts the local features at the text level. The output of the CNN layer is the feature maps that become the input for the LSTM layer. The LSTM layer uses the local text features extracted by the CNN to learn the long-term dependencies of the features of URLs that classify them as authentic or malicious, as depicted in the model architecture in Fig. 5.

The proposed hybrid model was implemented in Python and its performance were evaluated on two real-world URLs datasets. The model was implemented on Jupyter Notebook with GPU. The frameworks used for the implementation of the hybrid models are Keras Python 10 package and Tensorflow. The packages used for reading the dataset and mathematical processing are Pandas, Numpy, and Scikit-learn. Data preprocessing

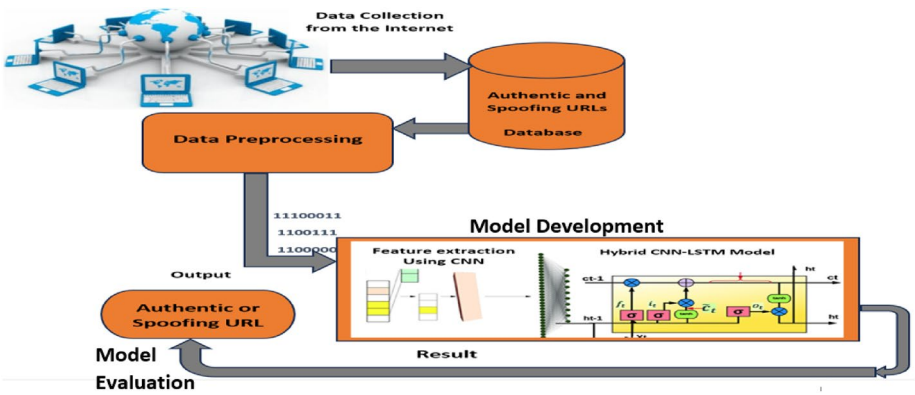


Fig. 5 Architectural framework of the proposed hybrid CNN-LSTM model

was done with the NLTK package. Finally, the Matplotlib package was used for results evaluation in terms of plotting graphs.

The hybrid CNN-LSTM model is made up of five (5) layers, namely the input embedding, CNN, the max pooling, LSTM, and the dense (output) layers. The first layer of the hybrid model is the Keras embedding layer, which feeds the training data utilized in a word embedding matrix with the input size of 60 (the max length of the URLs); the input is converted into a dense vector of size 64. In the second layer, the Conv1D feature extraction uses 64 filters of size 5, with the default is Rectified Linear Unit (ReLU) activation function. Following this layer is the feature extraction, which is the pooling of the feature vectors generated by CNN (1 layer), pooled by feeding them into a MaxPooling1D layer with a window size of 2. This was done to reduce the feature vector size and the number of parameters so that the computations will not affect the efficiency of the network. In the fourth layer, the pooled feature maps are now fed into the LSTM layer (1 layer), which outputs the long-term dependent feature maps while retaining their memory. The LSTM output dimension is also set to 64, and the linear activation function of Keras is used in this layer as the default activation. In the fifth layer, a dense layer is used to classify the trained feature vectors by shrinking the output space dimension to 2, which now corresponds to the predicted label of authentic or spoofing URL. The Sigmoid activation function was applied in this layer. The Adaptive Moment Estimation (ADAM) optimizer was used to train the model, and the binary cross-entropy loss function was used for calculating the accuracy of the results. The training was conducted with a batch size of 64 trained for 10 epochs.

Model evaluation

The model was validated and evaluated on the test dataset using five performance evaluation metrics namely: Accuracy, Precision, Recall, and F-score and Loss. The accuracy of malicious URL detection in Eq. (1) pertains to the capacity of a machine learning model to accurately categorize URLs as either malicious or benign. Precision, measured in Eq. (2), is the quotient of the number of genuine positives and the total count of URLs that the machine learning model has identified as positive. Precision, in essence, quantifies the ratio of genuinely malicious URLs to all URLs that the model has identified as malicious. Recall, also known as sensitivity, as shown in Eq. (4), is a statistical measure that quantifies the proportion of correctly identified malicious URLs out of the total number of actual malicious URLs in a given dataset. The F1-score measured in Eq. (4) is a statistic employed in machine learning to assess the efficacy of a binary classification model. The harmonic mean is calculated using precision and recall, both of which are additional metrics employed in categorization. Loss, [measured in Eq. (5)] in the context of machine learning, quantifies the degree of performance of a model on a specific dataset. The discrepancy between the observed output and the anticipated output for each instance is computed, and subsequently aggregated or averaged over all instances. The objective of a machine learning algorithm is to minimize the loss by modifying the parameters of the model throughout the training phase. Loss is synonymous with the consequence incurred for an inaccurate prediction. The loss is the binary cross-entropy loss suitable for binary classification. The Loss defined in Eq. (5) is the binary entropy

Table 1 Performance evaluation with UCL spoofing websites dataset

Algorithm	Accuracy (%)	Precision	Recall	F1-score	Loss
LR [25]	83.1	0.83	0.83	0.84	0.5650
DT [10]	83.8	0.85	0.83	0.83	0.5534
SVM [1]	82.7	0.81	0.83	0.82	0.5774
CNN [3]	90.4	0.90	0.92	0.93	0.0950
LSTM [26]	94.6	0.95	0.94	0.95	0.0695
Proposed hybrid model CNN-LSTM	98.9	0.97	0.98	0.99	0.0213

Table 2 Performance evaluation using the PhishTank dataset

Algorithm	Accuracy(%)	Precision	Recall	F1-score	Loss
LR [25]	80.1	0.79	0.80	0.81	0.5850
DT [10]	82.5	0.82	0.81	0.83	0.5434
SVM [1]	80.7	0.78	0.83	0.80	0.5874
CNN [3]	89.3	0.88	0.90	0.88	0.1550
LSTM [26]	92.6	0.93	0.92	0.93	0.0995
Proposed hybrid model CNN-LSTM	96.8	0.96	0.97	0.97	0.0413

loss applied to a binary classification problem. The metrics are mathematically defined as follows:

$$\text{Accuracy} = \frac{\text{TN} + \text{TP}}{\text{TN} + \text{TP} + \text{FN} + \text{FP}} \tag{1}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

$$F1 - \text{score} = \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

$$\text{Loss} : L = \frac{1}{N}\sum_{i=1}^{N}\left[yi\log\left(\widehat{y}\right) + \left(1 - yi\right)\log\left(1 - \widehat{yi}\right)\right]) \tag{5}$$

The evaluation metrics are defined in Eqs. (1, 2, 3, 4, and 5). Where TN = True Negative, TP = True Positive, FN = False Negative, FP = False Positive. Where: N = Total number of samples, y_i = the true label (0 or 1), log = Natural logarithm.

Results and discussion

The results obtained with UCL and PhishTank datasets are shown in Tables 1 and 2, respectively. The result obtained from the proposed model was compared with the state-of-the-art Logistics Regression (LR), Decision Tree (DT), Support Vector Machine

(SVM), CNN, and LSTM models. As shown in Table 1, the proposed hybrid model achieved the highest Accuracy, Precision, Recall, F1-score, and the lowest Loss of 98.9%, 0.97, 0.98, 0.99, 0.0213, respectively.

In the second experiment in Table 2, in which the model was trained on the PhishTank Dataset, the result also shows that the proposed CNN-LSTM model achieved the highest performance, with the Accuracy, Precision, Recall, F1-score and the lowest Loss of 96.8%, 0.96, 0.97, 0.97, 0.0413. This shows that hybrid CNN-LSTM achieved all-round best performance as against the existing state-of-the-art machine learning techniques.

Moreover, the accuracy results obtained with UCL spoofing website dataset showed a better overall performance than that obtained with the PhishTank website as shown in Fig. 6. This could be attributed to the differences in the attributes present in UCL and PhishTank datasets.

Conclusion

The devastating impact of website URL spoofing targeted at individuals, organizations and government call for an efficient measure to detect and prevent the attackers from perpetuating their craft. The paper proposed an efficient hybrid CNN-LSTM model to detect and counter spoofing website URLs spoofing attacks via textual feature extracting using CNN and learning of the extracted textual features using LSTM. The model was trained on two online publicly available datasets and evaluated on five metrics. The overall results obtained showed that the hybrid CNN-LSTM model achieved the highest performances of 98.9% Accuracy on the first dataset (UCL), while the second dataset (Phishing) achieved 96.8% Accuracy. The standalone CNN and LSTM models on the hand achieved the Accuracies of 90.4% and 94.6%, respectively, on the UCL dataset, while the Phishing dataset resulted in 89.3% and 92.6% Accuracies, respectively. The results obtained has shown that the proposed technique is more efficient in detecting phishing URLs than the individual deep models and other ML models in the literature; therefore, the hybrid technique is strongly recommended for use in spoofing website detection. The limitation of the study stems from the fact that other hybrid ML and deep models were not considered in the study. Also, other variants of phishing URL datasets

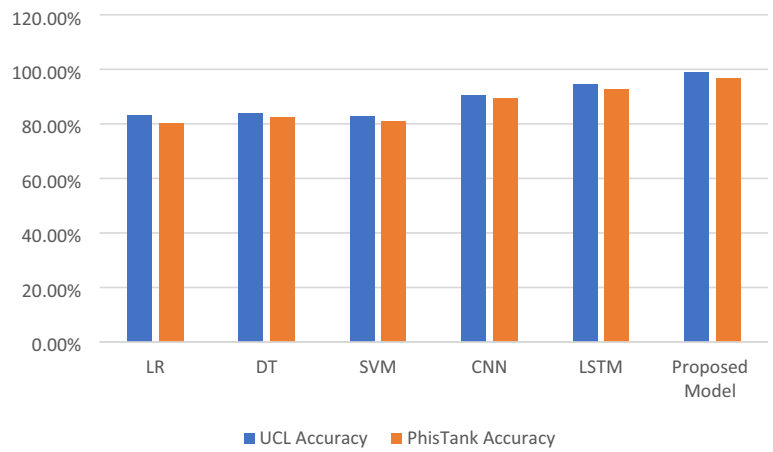


Fig. 6 Accuracy of the proposed model with UCL spoofing website and PhishTank datasets

(e.g., imbalance) were not considered. In the nearest future, the authors intend to train the model on more datasets of varying natures, develop other hybrid deep learning technique and further compare performance with the proposed method.

Acknowledgements
Not applicable.

Author contributions
BC conceptualized the research, designed the study, analyzed the results, as well as contributed largely in writing the manuscript. ON reviewed literature, collected the data for the study and presented a graphical view of the results. EO reviewed the article draft and pointed out some important points leading to an improvement in the article quality.

Funding
Not applicable.

Availability of data and materials
The datasets used and/or analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Competing interests
The authors declare that they have no competing interests.

Received: 13 September 2023 Accepted: 6 December 2023
Published online: 24 January 2024

References

1. Abedin NF, Bawm R, Sarwar T, Saifuddin M, Rahman MA, Hossain S (2021) Phishing Attack detection using machine learning classification techniques. In: Proceedings of the 3rd international conference on intelligent sustainable systems (ICISS), Thoothukudi, India, 3–5 December. 90(17):1125–1130
2. Agrawal P, Mangal D (2015) A novel approach for phishing URLs detection. Int J Sci Res 5(30):1117–1122
3. Aldakheel EA, Zakariah M, Gashgari GA, Almarshad FA, Alzahrani AIA (2023) A Deep learning-based innovative technique for phishing detection in modern security with uniform resource locators. Sensors 23(9):4403. https://doi.org/10.3390/s23094403
4. Almutairi A, Alshoshan AI (2022) Developing a webpage phishing attack detection tool. In: Arai K (ed) Intelligent computing. Lecture notes in networks and systems. Springer, Cham
5. Babagoli M, Aghababa MP, Solouk V (2022) Heuristic nonlinear regression strategy for detecting phishing websites. Soft Comput 23:4315–4327
6. Balogun AO, Adewole KS, Raheem MO, Akande ON, Usman-Hamza FE, Mabayoje MA, Akintola AG, Asaju-Gbolagade AW, Jimoh RG (2021) Improving the phishing website detection using empirical analysis of FunctionTree and its variants. Heliyon. 7:e07437
7. Bitaab M, Cho H, Oest A, Zhang P, Sun Z, Pourmohamad R, Kim D, Bao T, Wang R, Scam SY et al (2020) Pandemic: how attackers exploit public fear through phishing. Proc APWG Symp Electr Crime Res. 8(118):1–10
8. Carroll F, Adejobi JA, Montasari R (2022) How good are we at detecting a phishing attack? Investigating the evolving phishing attack email and why it continues to successfully deceive society. SN Comput Sci 3:170
9. Chung J, Koay J-Z, Leau Y-B (2020) A review on social media phishing: factors and countermeasures BT—advances in cyber security. Proc Int Conf Adv Cyber Secur 18(31):657–673
10. Das Guptta S, Shahriar KT, Alqahtani H (2022) Modeling hybrid feature-based phishing websites detection using machine learning techniques. Ann Data Sci. https://doi.org/10.1007/s40745-022-00379-8
11. Dinler ÖB, Sahin CB (2021) Prediction of phishing websites with deep learning using WEKA environment. Avrupa Bilim Teknol Dergisi 7(24):35–41
12. Feng F, Zhou Q, Shen Z, Yang X, Han L, Wang J (2021) The application of a novel neural network in the detection of phishing websites. J Ambient Intell Humaniz Comput 43(1):1–15
13. Gandotra E, Gupta D (2021) Improving spoofed website detection using machine learning. Cybern Syst 52(2):169–190. https://doi.org/10.1080/01969722.2020.1826659
14. Gupta BB, Arachchilage NAG, Psannis KE (2017) Defending against phishing attacks: taxonomy of methods, current issues and future directions. Telecommun Syst 67:247–267
15. Haynes K, Shirazi H, Ray I (2021) Lightweight URL-based phishing detection using natural language processing transformers for mobile devices. Proc Comput Sci 191(8):127–134
16. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
17. LeCun Y, Bottou L, Bengio Y, Haffner P (2018) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324
18. Li T, Kou G, Peng Y (2020) Improving malicious URLs detection via feature engineering: Linear and nonlinear space transformation methods. Inf Syst 91(4):101494
19. Mao J, Tian W, Li P, Wei T, Liang Z (2017) Phishing website detection based on effective CSS features of web pages. Wirel Algor Syst Appl

20. Mohammad R, McCluskey L (2015). Phishing websites. UCI machine learning repository. https://doi.org/10.24432/C51W2X
21. PhishTank (2023). PhishTank URL dataset. https://www.phishtank.com/
22. Rao RS, Pais AR (2018) Detection of phishing websites using an efficient feature-based machine learning framework. Neural Comput Appl 31(42):3851–3873
23. Rao RS, Vaishnavi T, Pais AR, Ambient J (2022) Detection of phishing websites by inspecting URLs. Intell Humaniz Comput 11(8):813–825
24. Rekouche, K. (2011). Early phishing. arXiv:1106.4692
25. Sheng S, Wardman B, Warner G, Cranor LF, Hong J, Zhang C (2009) An empirical analysis of phishing blacklists. In: Proceedings of the 6th conference on email and anti-spam, Mountain View, CA, USA. 7(10):81–90
26. Smadi S, Aslam N, Zhang L (2021) Detection of online phishing email using dynamic evolving neural network based on reinforcement learning. Decis Supp Syst 107(9):88–102
27. UCI Machine Learning Repository (2015). Spoofing URL dataset. https://archive.ics.uci.edu/dataset/327/phishing+websites
28. Vrbančič G, Fister IJ, Podgorelec V (2023) Datasets for phishing websites detection. Data Brief. https://doi.org/10.1016/j.dib.2020.106438
29. Warburton D (2022). Phishing attacks soar 220% during COVID-19 peak as cybercriminal opportunism intensifies. Accessed on 27 June 2023 from https://www.f5.com/company/news/features/phishing-attacks-soar-220--during-covid-19-peak-as-cybercriminal
30. Xiao X, Xiao W, Zhang D, Zhang B, Hu G, Li Q, Xia S (2021) Phishing websites detection via CNN and multi-head self-attention on imbalanced datasets. Comput Secur 108(9):102372
31. Yasin A, Abuhasan A (2019) An intelligent classification model for phishing email detection. Int J Netw Secur Appl 8(7):55–72

Publisher's Note