future internet

MDPI

Article

Multimodel Phishing URL Detection Using LSTM, Bidirectional LSTM, and GRU Models

Sanjiban Sekhar Roy [1], Ali Ismail Awad [2,*], Lamesgen Adugnaw Amare [1], Mabrie Tesfaye Erkihun [1] and Mohd Anas [1]

[1] School of Computer Science and Engineering, Vellore Institute of Technology, Vellore 632014, India
[2] College of Information Technology, United Arab Emirates University, Al Ain P.O. Box 17551, United Arab Emirates
* Correspondence: ali.awad@uaeu.ac.ae; Tel.: +971-3713-5531

Abstract: In today's world, phishing attacks are gradually increasing, resulting in individuals losing valuables, assets, personal information, etc., to unauthorized parties. In phishing, attackers craft malicious websites disguised as well-known, legitimate sites and send them to individuals to steal personal information and other related private details. Therefore, an efficient and accurate method is required to determine whether a website is malicious. Numerous methods have been proposed for detecting malicious uniform resource locators (URLs) using deep learning, machine learning, and other approaches. In this study, we have used malicious and benign URLs datasets and have proposed a detection mechanism for detecting malicious URLs using recurrent neural network models such as long short-term memory (LSTM), bidirectional long short-term memory (Bi-LSTM), and the gated recurrent unit (GRU). Experimental results have shown that the proposed mechanism achieved an accuracy of 97.0% for LSTM, 99.0% for Bi-LSTM, and 97.5% for GRU, respectively.

Keywords: phishing URL detection; long short-term memory (LSTM); bidirectional LSTM (Bi-LSTM); gated recurrent unit (GRU) RNN

Citation: Roy, S.S.; Awad, A.I.; Amare, L.A.; Erkihun, M.T.; Anas, M. Multimodel Phishing URL Detection Using LSTM, Bidirectional LSTM, and GRU Models. *Future Internet* **2022**, *14*, 340. https://doi.org/10.3390/fi14110340

Academic Editor: Claude Chaudet

Received: 12 September 2022
Accepted: 18 November 2022
Published: 21 November 2022

1. Introduction

Most of our daily activities are Internet-based, including communication, business, marketing, education, travel, and shopping. Therefore, with the massive growth of Internet usage, the likelihood of sharing personal information online has also grown rapidly, making sensitive information vulnerable to cybercrime. While the Internet has many benefits, it is also used by criminals to commit cybercrimes, including phishing. A phishing attack, an effective cybercrime, is a social engineering technique where a fraudulent message is sent through email, chat applications, or text to a victim on the premise of arriving from a safe source; its main aim is to trick the recipient to reveal sensitive information. According to a phishing and fraud report, phishing attacks soared by 220% during the COVID-19 pandemic [1]. In a phishing attack, criminals attempt to steal private information, such as login credentials and financial details, from individuals for fraudulent use [2].

The first known instance of a phishing attack occurred in the mid-1990s, when a group of hackers or phishers called the warez community stole login credentials and personal information from AOL users [3]. In early 2000, attackers turned their attention to financial systems and launched an attack on E-Gold in June 2001 [4]. By 2003, phishers registered several domain names that resembled the names of legitimate commercial sites such as eBay and PayPal and sent mass-mailings to customers asking them to visit the sites and provide their personal information and credit card details [5]. In 2020, Google registered 2.02 million malicious sites, which is a 19% increase over 2019. In 2021, CISCO's cybersecurity threat trend reported that 90% of data breaches occur due to phishing. Phishing attacks are a major and serious issue worldwide. The prevention of such attacks is becoming increasingly complicated [6]. Various strategies have been proposed to overcome phishing; among

them, two methods stand out: traditional and nontraditional. The first includes legal, education and awareness, blacklist/whitelist, visual similarity, and search engine; the second comprises techniques such as artificial intelligence (AI)-based, content-based, deep-learning- and machine-learning-based, heuristics-inspired, data mining, and fuzzy-rule-based techniques [7].

Blacklisting is the most commonly used method by modern browsers to detect phishing websites. However, this protection method fails to detect the zero-day phishing sites [8,9]. Machine-learning-based techniques have also been used to detect phishing uniform resource locators (URLs). To detect a phishing website, URLs first need to be analyzed for feature extraction, then a training set is built using the extracted features along with their labels, followed by supervised machine-learning techniques [10]. In this study, we focus on deep learning. Recurrent neural networks (RNNs) are among the most common deep-learning techniques for the classification and prediction of sequential data [11]. This paper presents a classification method for detecting malicious URLs using normal LSTM, bidirectional long short-term memory (Bi-LSTM), and gated recurrent units.

The research contributions in the study are highlighted in the following points:

- In all web-based malicious activities, users are required to click a URL; using this URL's information, we aim to develop a deep learning model that detects malicious URLs.
- The URL is padded as a step of sequences; therefore, instead of RNN, we have proposed LSTM-based architectures such as LSTM, Bi-LSTM, and gated recurrent units (GRU), as RNNs are subject to the vanishing gradient problem. However, with the ability to better understand the URL input, Bi-LSTM has an edge in terms of accuracy in detecting malicious URLs. The performance of the proposed models LSTM, Bi-LSTM, and GRU was analyzed using different metrics, such as precision, recall, F1 score, and accuracy.
- The architecture and working steps of each proposed algorithm are demonstrated in detail, and a detailed comparative performance with other existing models is conducted.
- The proposed model can be used for real-time website detection.

The structure of the paper is organized as follows: Section 2 presents a literature review and related works. Section 3 provides a conceptual understanding, theoretical foundations, and mathematical model of the proposed mechanisms. Section 4 introduces and discusses the experimental results and outcomes. Finally, concluding remarks and future work are presented in Section 5.

2. Related Work

In [12], the authors proposed a novel generalized phishing detection system based on an AV binary-modified equilibrium optimizer with k-nearest neighbors (KNN). The system uses a binary version of a modified equilibrium optimizer (AV-BMEO) for feature selection and a K-nearest neighbor machine learning algorithm for classification.

Balogun et al. in [13] introduced a functional tree meta-learning mechanism for phishing site detection. A functional tree-based (FT) model is highly effective for detecting phishing and legitimate websites with better accuracy, and it is recommended. In [14], researchers proposed a novel method to detect a phishing URL website using a self-attention convolutional neural network (CNN) algorithm. The authors used an imbalanced dataset and a generative adversarial network (GAN) deep learning model to produce data for an imbalanced dataset. Next, they combined the CNN deep learning model and multithread self-attention to build the classifier. Linear and nonlinear space-transformation-based methods have been used [15] for malicious URL detection via feature engineering. In this study, a two-stage distance metric technique was developed for linear transformation and the Nyström approach for kernel approximation was introduced for both linear and nonlinear transformations.

A machine-learning-based predictive model to classify websites as phishing and legitimate was introduced in 2020 [16]. The authors in this study proposed a machine-learning-based system to detect phishing websites. Support vector machines, convolutional

neural networks, and K-nearest neighbor machine learning algorithms were used to detect phishing websites. Haynes et al. proposed a lightweight URL-based phishing detection method using natural language processing transformers for mobile devices. They applied the artificial neural network (ANNs) model to URL-based and HTML-based website features to distinguish malicious from legitimate URLs, and the proposed method can only be used on mobile devices [17]. In 2018, the authors of [18] implemented a neural-network-based model to detect phishing websites. This model used neural-network-based classification with a simple and stable Monte Carlo algorithm.

In [19], a model for detecting spam emails was proposed, which is a hybrid system of neural networks. The proposed approach used a technique that automatically adds the number of emails to corpus datasets. Babagoli et al. [20] developed a model based on a heuristic-based regression approach combined with a decision tree algorithm and a wrapper feature selection approach to detect phishing websites. The authors of [21] proposed a model that uses the extracted heuristic features from the website itself. The authors used eight different machine learning (ML) algorithms for the evaluation, and the principal component analysis random forest (PCA-RF) algorithm yielded the highest accuracy and image analysis. Yasin and Abuhasan [22] developed an intelligent classification model for detecting phishing emails. Publishers mainly used two models: knowledge discovery, which extracts the features from the given string, and data mining, which selects the best classification model. A Java program was used to extract the features from the email header and body, after which a data mining algorithm was applied to the extracted features to determine the algorithm with the best results.

In this study, we checked the potential capabilities of LSTM, Bi-LSTM, and GRU for detecting malicious URLs. Compared to RNN, LSTM is a better choice, and RNNs are difficult to train for the input dataset of URLs because they have long-term temporal dependencies. The reason for this is that the gradient decay of the loss function results in nonpolynomial time. By contrast, LSTM uses special units with additional memory cells and can retain information for a long time. Therefore, regardless of which URLs LSTM passes using hidden layers, it preserves such information of URLs; therefore, it is unidirectional, whereas Bi-LSTM tends to run the URLs training data in two ways: first, from past to future, and second, from future to past. LSTM has one hidden layer state to train URLs, whereas Bi-LSTM has two hidden states for training. Finally, GRU is almost similar to LSTM with a less complex structure compared to LSTM [23–25].

3. Proposed Models

The overall architecture of the proposed approach is shown in Figure 1. The proposed system has as illustrated in Figure 1 has four main structures: input URL, data preprocessing, training, and classification. The input data is legitimate and malicious URLs dataset collected from a Kaggle source. The next step is data preprocessing, in which we developed a character-embedding mechanism that encodes all the available characters in the input URL into a numerical form. Further data preprocessing steps and mechanisms are briefly discussed in Section 4.2. In the next step, the training set is fed into the proposed deep learning model to train it to perform the desired task. Once our model is trained and evaluated, the last step is prediction. In this step, real-time URL data are passed to the model, which predicts the maliciousness or legitimacy of the given URL.

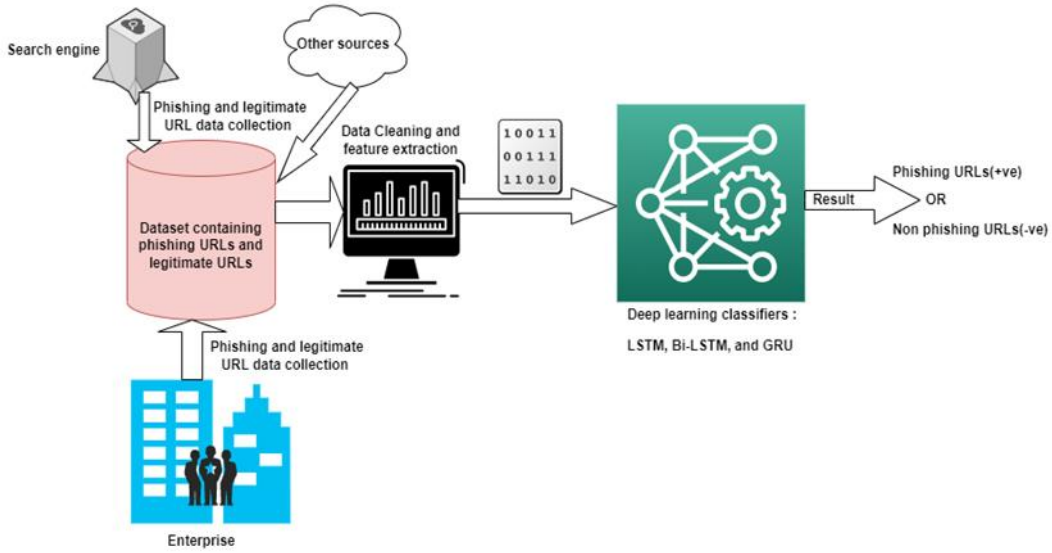


Figure 1. Conceptual architecture of the three proposed models for detecting phishing URLs.

3.1. Proposed Model I: Long Short-Term Memory (LSTM)

Recurrent neural networks are a form of neural network that is good for processing sequence data prediction. As RNNs process more steps, they are more susceptible vanishing gradients than other neural networks [26]. LSTMs and GRU-based RNNs are methods for overcoming the challenges of simple RNNs [27]. LSTM, proposed in 1997 by Hochreiter and Schmidhuber, is an evolution of RNNs capable of learning long-term dependencies and remembering input information for a long period through gates [28]. It is composed of a cell state, an input gate, a forget gate, and an output gate [29]. The detailed architecture of LSTM is shown in Figure 2 [30].

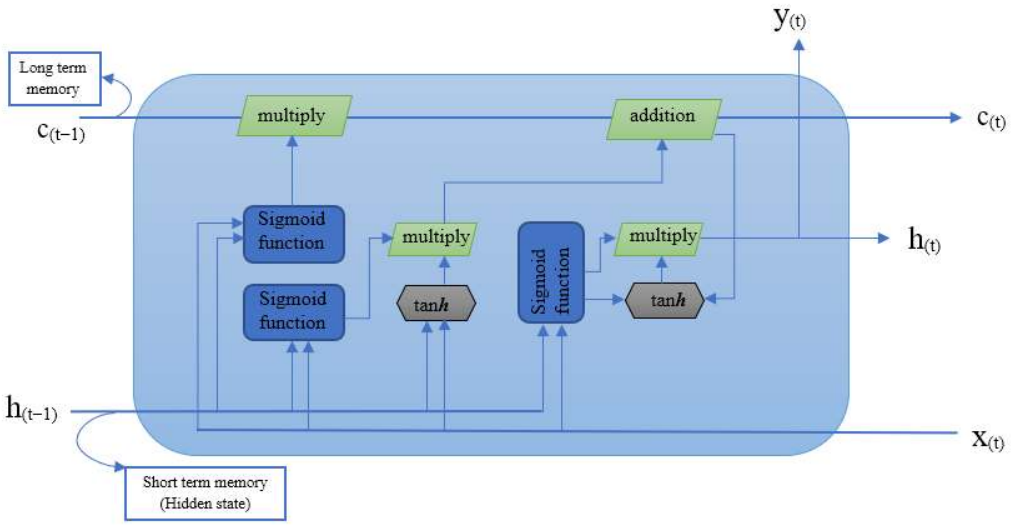


Figure 2. Detailed architecture of the long short-term memory (LSTM) model.

In Figure 2, C_t = cell state, C_{t-1} = old cell state, f_t = forget gate, i_t = input gate, O_t = output gate, h_t = hidden state, tanh = hyperbolic tangent activation function, $\widetilde{C}_t$ = cell update, and σ = sigmoid activation function.

In the LSTM model, the cell state C_t is the main chain for the forward data flow. Two steps must be updated: one is from the forget gate f_t, which decides which information to keep or forget from C_t. Data from h_{t-1} and information from x_t are moved through the σ values that yield a 0 or 1. Here, 1 represents keep, and 0 represents forget. The final

result from the forget gate is multiplied by the old cell state C_{t-1} Equations (1) and (2). The second is from the input gate i_t and the candidate memory cell $\widetilde{C}_t$, which decides whether to add new information. In an input gate, a sigmoid layer determines whether a piece of new information should be added. In the candidate memory cell, the input from h_{t-1} and the current input pass through a hyperbolic tangent function. The result from the input gate was multiplied by the values of the candidate memory cell. Finally, the two values from the first and second steps were added to update the cell state [31].

$$f_t = \sigma(_{W_t}\,[h_{t-1,X_t}\,] + b_f\,) \tag{1}$$

$$M_t = f_t * C_{t-1} \tag{2}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{3}$$

$$\widetilde{C}_t = \tan \mathrm{h}\,(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{4}$$

$$C_t = M_t + i_t * \widetilde{C}_t \tag{5}$$

$$o_t = \sigma(W_o[h_{t-1}\,,\, x_t] + b_o) \tag{6}$$

Finally, the output gate determines the output value. First, the previous output and current input pass through a sigmoid function and are then multiplied by the newly updated cell after passing through a tanh function. See Algorithm 1.

$$h_t = o_t * \tan \mathrm{h}\,(C_t) \tag{7}$$

Algorithm 1: Regular LSTM for phishing URL detection

Input	:	URL $\{x_1, x_2, x_3 \ldots .x_L\}$, L = length of URL
Output	:	Phishing or legitimate (Y)
Step 1	:	**Begin**
Step 2	:	For $t = 1$ to L do
Step 3	:	calculate the value of f_t using Equation (1)
Step 4	:	if $f_t = 0$, then forget the information
Step 5	:	else, the forget value is *1* keep the information
Step 6	:	End if
Step 7	:	Calculate an input gate value i_t Equation (3)
Step 8	:	Calculate the *candidate key* $(\widetilde{C}_t)$ of another weight matrix in candidate memory cell Equation (4)
Step 9	:	Calculate the cell state C_t Equation (5)
Step 10	:	Compute the value of the output cell state O_t and multiply i_t by tanh of C and store it in h_t Equations (6) and (7)
Step 11	:	return (h_t, C_t)
Step 12	:	End for
Step 13	:	Y = SoftMax $(h_1, h_2 \ldots h_L)$
Step 14	:	**End**

3.2. Proposed Model II: Bidirectional Long Short-Term Memory (Bi-LSTM)

In a Bi-LSTM network, the information flows in two directions through the backward and forward layers [30], whereas in regular long-short-term memory, there is only one possible way of information flow, either using a backward or forward layer. In the Bi-LSTM model, the output layer can obtain information from the past and future states simultaneously. The general architecture of the Bi-LSTM is shown in Figure 3 [32,33]. See Algorithm 2.

$$A_t^f = \tan \mathrm{h}\left(W_{xA}^f x_t + W_{AA}^f A_{t-1}^f + b_A^f\right) \tag{8}$$

$$A_t^b = \tan \mathrm{h}\left(W_{xA}^b x_t + W_{AA}^b A_{t+1}^b + b_A^b\right) \tag{9}$$

$$y_t = W^f_{Ay} A^f_t + W^b_{Ay} A^b_t + b_y \quad (10)$$

where A^f_t = forward-layer output sequence, A^b_t = backward-layer output sequence, y_t = output vector, σ = activation function used to merge A^f_t and A^b_t, $W_/$ = weight matrix, and $b_/$ = bias.

Algorithm 2: Bi-LSTM for phishing URL detection

Input : URL $\{x_1, x_2, x_3 \ldots .x_L\}$, L = length of URL
Output : Phishing or legitimate (Y)
Step 1 : **Begin**
Step 2 : For $t = 1$ to L do
Compute forward layer output sequence A^f_t Equation (8)
Step 3 : End for
Step 4 : For $t = L$ to 1 do
Compute backward layer output sequence A^b_t Equation (9)
Step 5 : End for
Step 6 : Obtain Y by merging A^f_t and W^b_t using sigmoid activation function
Step 7 : **End**

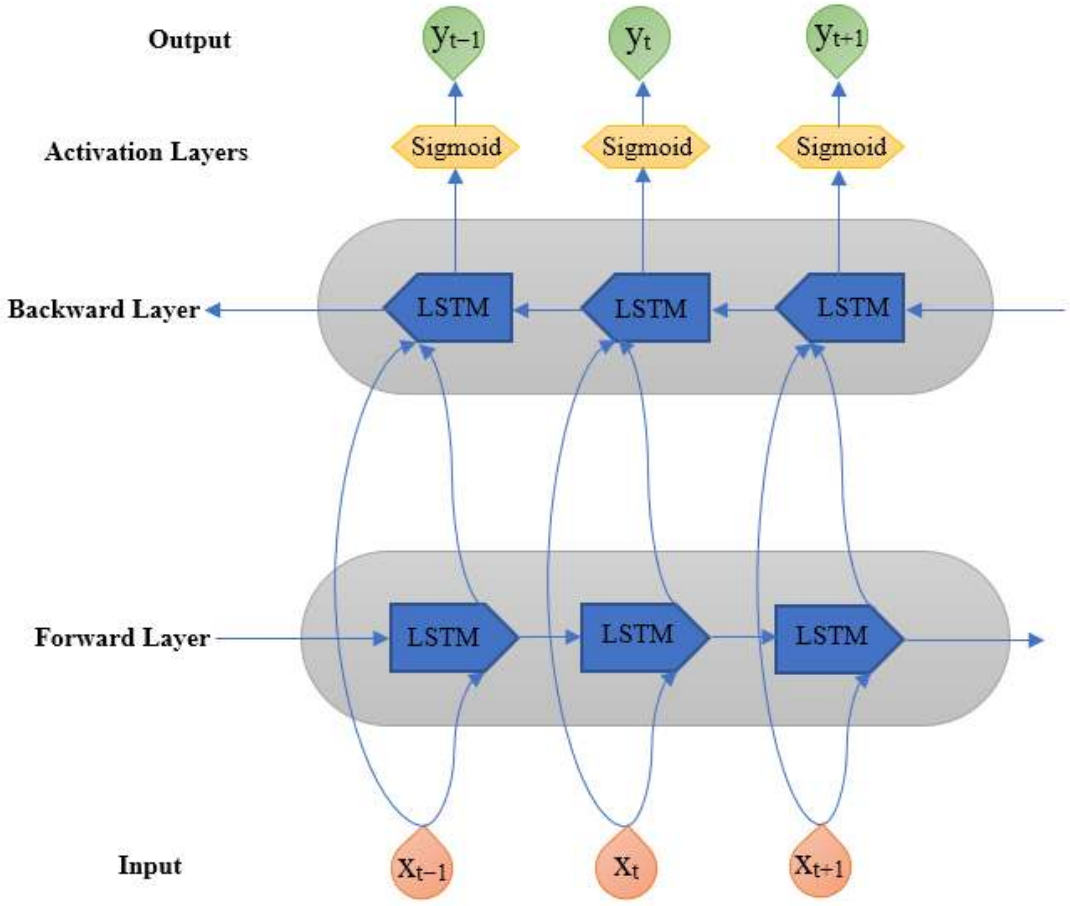


Figure 3. Detailed structure of the bidirectional long short-term memory (Bi-LSTM) model.

3.3. Proposed Model III: Gated Recurrent Unit (GRU)-Based RNN

The third model, the GRU-based RNN, is similar to regular LSTM. As in the normal gate, three gates were used; the GRU has two gates, namely the "reset gate" and "update gate" [34,35]. Here, r_t forgets the LSTM cell gate. It is a combination of the previous hidden

state and current input and determines how much of the past information is neglected. A general representation of the GRU is shown in Figure 4. See Algorithm 3.

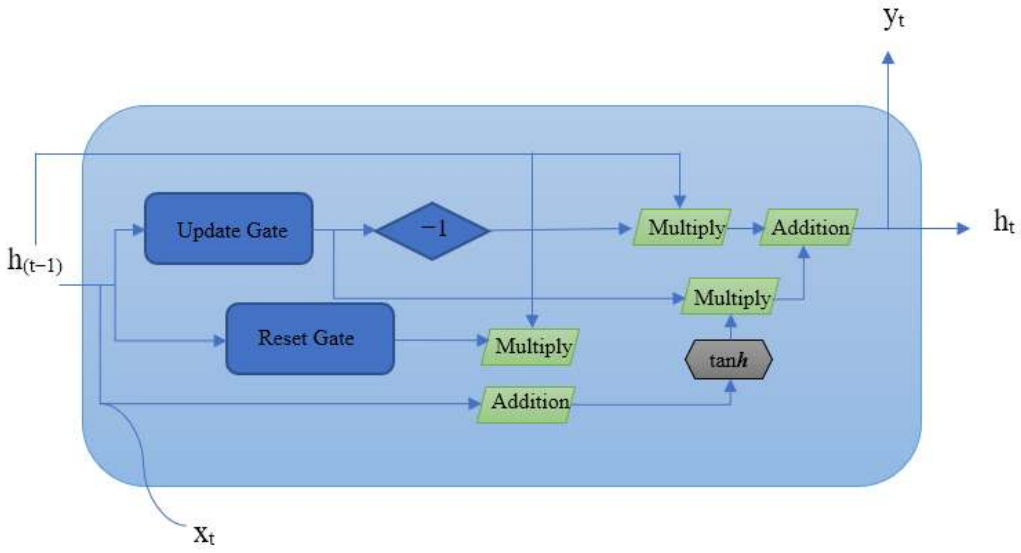


Figure 4. Operational diagram representing the functions of the GRU-based RNN model.

The GRU model is implemented using the following equations [30,36,37]:

$$r_t = \sigma\left(W_r \cdot \left[h_{t-1}, x_t\right]\right) \tag{11}$$

$$z_t = \sigma\left(W_z \cdot \left[h_{t-1}, x_t\right]\right) \tag{12}$$

$$\widetilde{h}_t = \tanh\left(W \cdot \left[r_t * h_{t-1}, x_t\right]\right) \tag{13}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \widetilde{h}_t \tag{14}$$

where r_t = reset gate, z_t = update gate, $\widetilde{h}_t$ = intermediate memory, and h_t = output.

Algorithm 3: GRU-based RNN for phishing URL detection

Input : URL $\{x_1, x_2, x_3 \ldots .x_L\}$, L = length of URL
Output : Phishing or legitimate (Y) (Y = 0: Legitimate, Y = 1: Malicious)
Step 1 : **Begin**
Step 2 : For each URL $x_1..x_L$ do
Step 3 : h_{t-1} and x_t merge and pass through a sigmoid function and the result stored in r_t, Equation (11).
Step 4 : Compute z_t from h_{t-1} and x_t (different biases and weights Equation (12)
Step 5 : Compute $\widetilde{h}_t$ by combining the new input x_t with the reset h_{t-1} and pass their output through a tanh function Equation (13)
Step 6 : Subtract z_t from a vector containing all 1s and multiply it with the previous hidden state.
Step 7 : The output from Step 4 z_t multiplied with the output from Step 5 $\widetilde{h}_t$.
Step 8 : Combine the output from Step 6 with the output from Step 7 and store in h_t Equation (14)
Step 9 : End for
Step 10 : Obtain Y from h_t
Step 11 : **End**

4. Experimental Work and Results

4.1. Dataset

This section provides information regarding the dataset used to evaluate the phishing detection models proposed in this study. We used a dataset from Kaggle contain-

ing 450,176 URLs, along with 345,738 legitimate and 104,438 phishing URLs [https://www.kaggle.com/datasets/siddharthkumar25/malicious-and-benign-urls] (Accessed on 11 September 2022). The numbers of the phishing and legitimate sites used for implementation are shown in Figure 5.

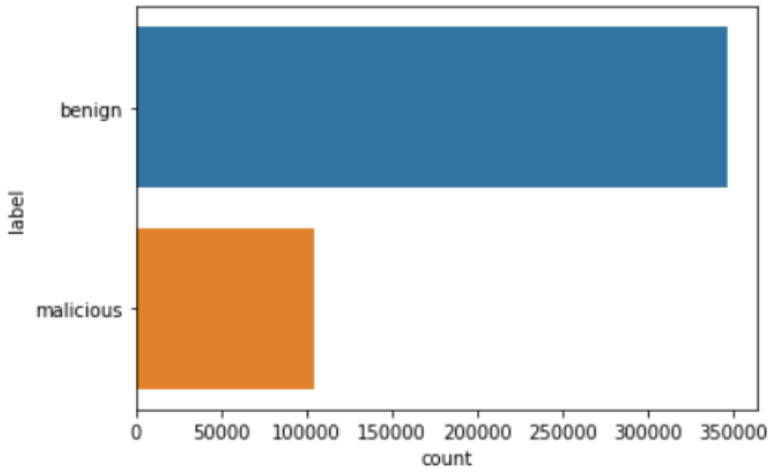


Figure 5. Length of malicious and legitimate URLs.

4.2. Data Preprocessing

In the data processing part, the first step that must be performed is transforming every character in the URL into a numerical form. In this stage, the characters from the URL are transformed into numbers using a character-level tokenization technique. Before passing these tokens to our deep learning model, we must ensure that the variable-length sequence is the same. For a fixed number of characters in a URL length (L), if the given input URL is greater than L, the excessive characters will be removed, and if the given input URL length is less than the fixed length L, an appropriate number of zeros will be added to the matrix before or after the characters in each row. This sequence of numbers is then turned into an embedding using embedding mechanisms, and finally, the translated URL is transferred into the proposed system layers.

4.3. Result and Discussion

This section presents the experimental results for each proposed algorithm using different performance metrics.

4.3.1. Performance Metrics

To evaluate the proposed phishing URL detection method using different deep learning techniques, we used a set of different evaluation metrics. Some of the measurements used to analyze our work performance are the confusion matrix (Table 1), which is one of the metrics mainly used to analyze and evaluate the performance of the URL detection mechanism.

Table 1. Confusion matrix for malicious and legitimate classes.

		Predicted Class	
		0 (Legitimate)	**1 (Malicious)**
Actual class	0 (legitimate)	True negative (TN)	False positive (FP)
	1 (malicious)	False negative (FN)	True positive (TP)

Where, TP: The number of malicious URLs classified as malicious by the model. TN: The number of legitimate URLs classified as legitimate by the model. FP: The number of legitimate URLs classified as malicious by the model. FN: The number of malicious URLs classified as legitimate by the model.

4.3.2. Evaluation of the Proposed Models

Seventy percent of the dataset was used for training, and thirty percent for the test set. Different evaluation methods are used in the experiments. Figure 6a–c shows the confusion matrices for the proposed LSTM, Bi-LSTM, and GRU models, respectively.

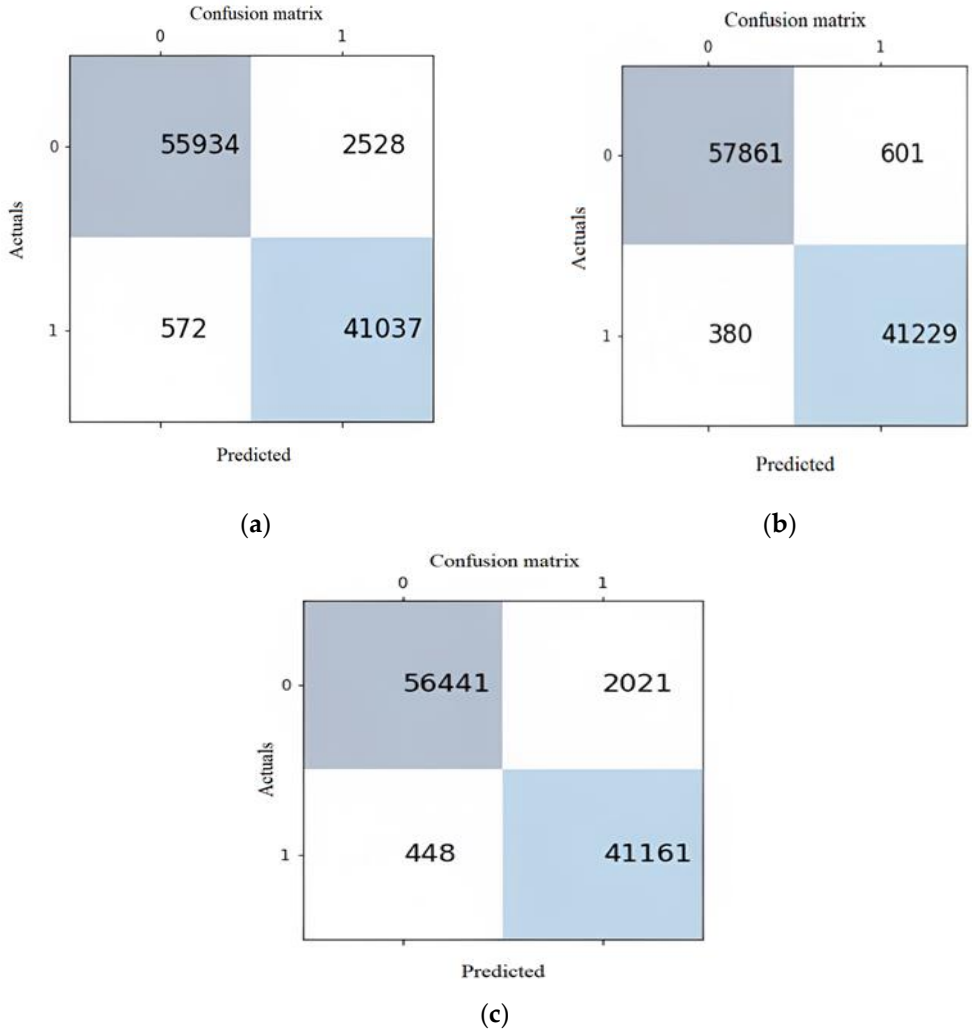
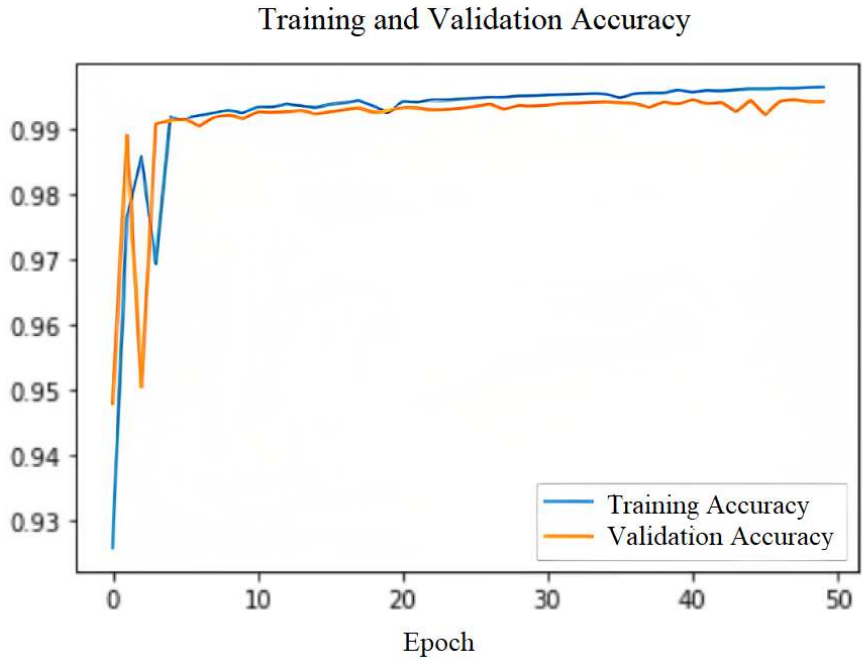


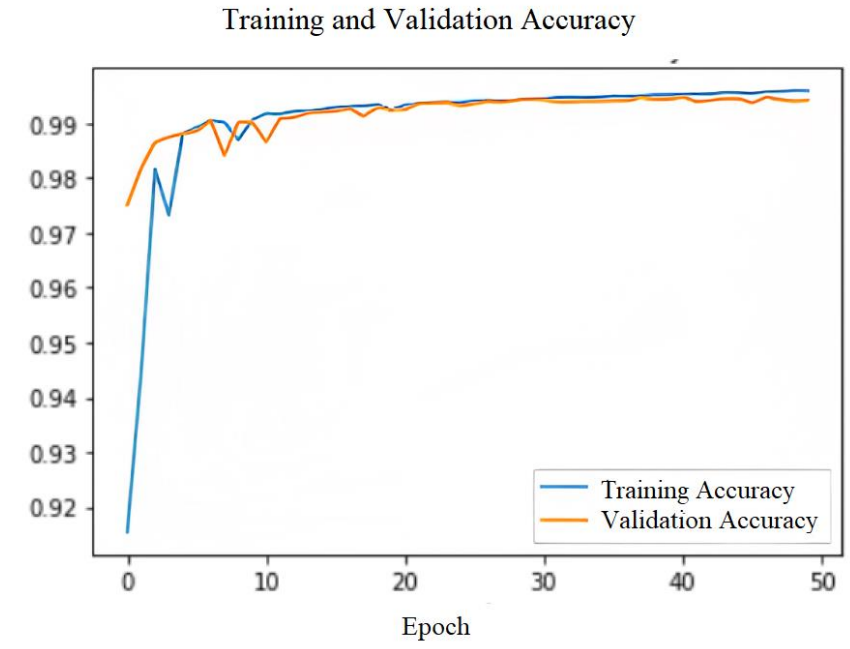
Figure 6. Confusion matrices for LSTM (**a**), Bi-LSTM (**b**), and GRN-based RNN algorithm (**c**).

Figure 7a–c shows the training and validation accuracy of the proposed neural networks. The orange curve represents the validation accuracy, and the blue curve represents the training accuracy of the models. The training and validation loss of each proposed LSTM, Bi-LSTM, and GRU model are shown in Figure 7a–c, respectively.

The blue curves in Figure 7a–c indicate the training accuracy and the orange curves describe the validation accuracy.

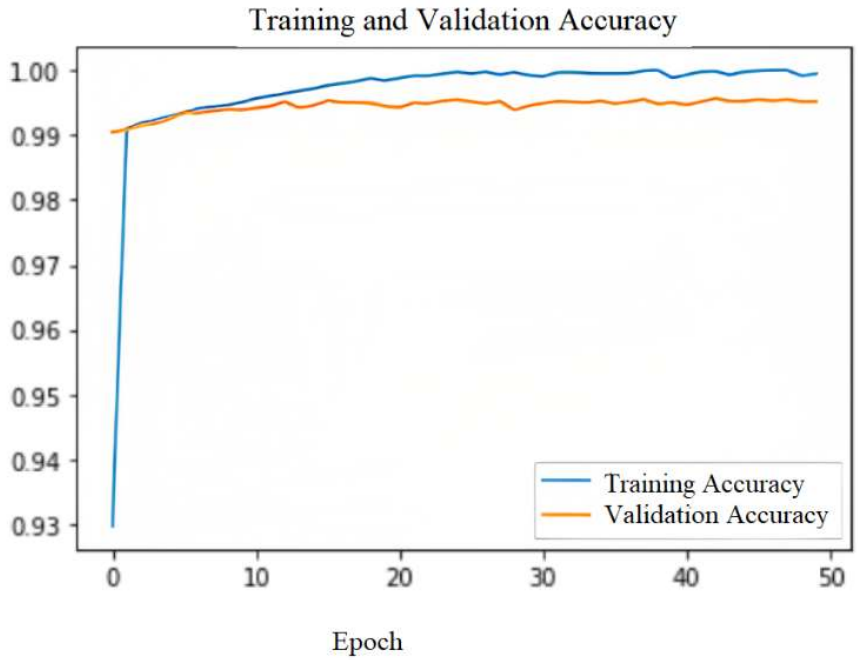


(**a**)



(**b**)

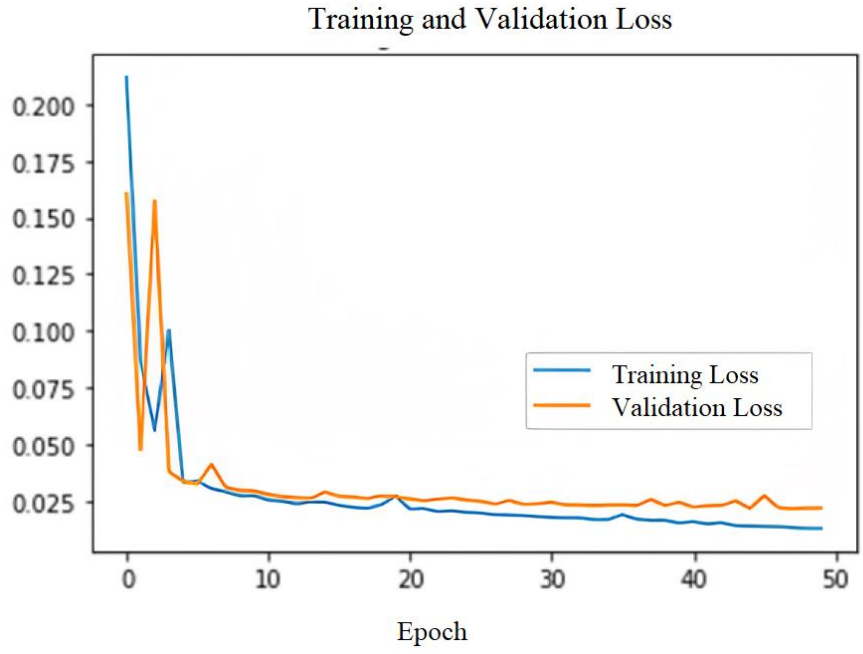
Figure 7. *Cont.*



(**c**)

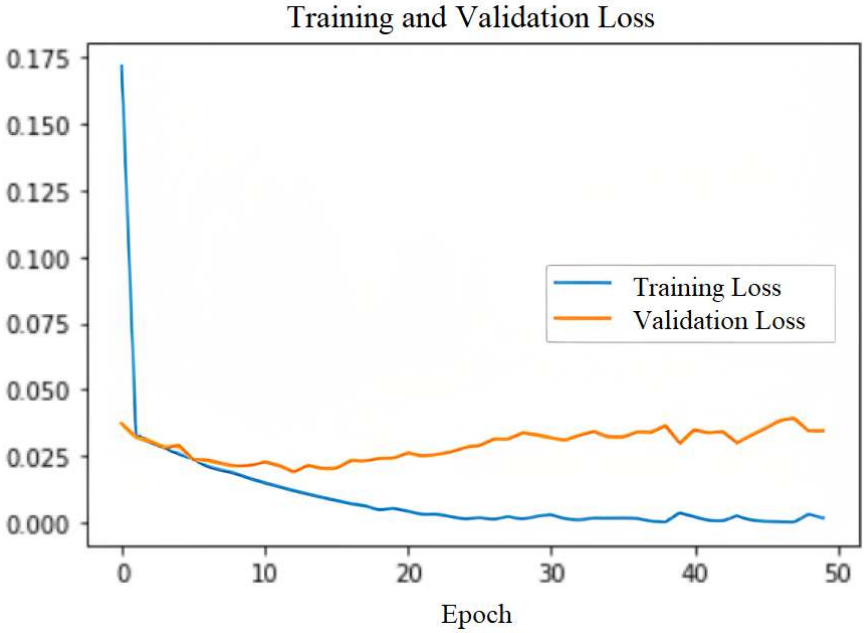
Figure 7. (**a**) Training and validation accuracy for LSTM model. (**b**). Training and validation accuracy for Bi-LSTM model. (**c**). Training and validation accuracy for GRN-based RNN algorithm.

In Figure 8a–c, the blue curve indicates the training loss for all the methods and the orange curve describes the validation loss. In Figure 8, the *x*-axis indicates the number of epochs, and the *y*-axis represents the loss. Table 2 shows the training accuracy achieved using the LSTM, Bi-LSTM, and GRU networks.

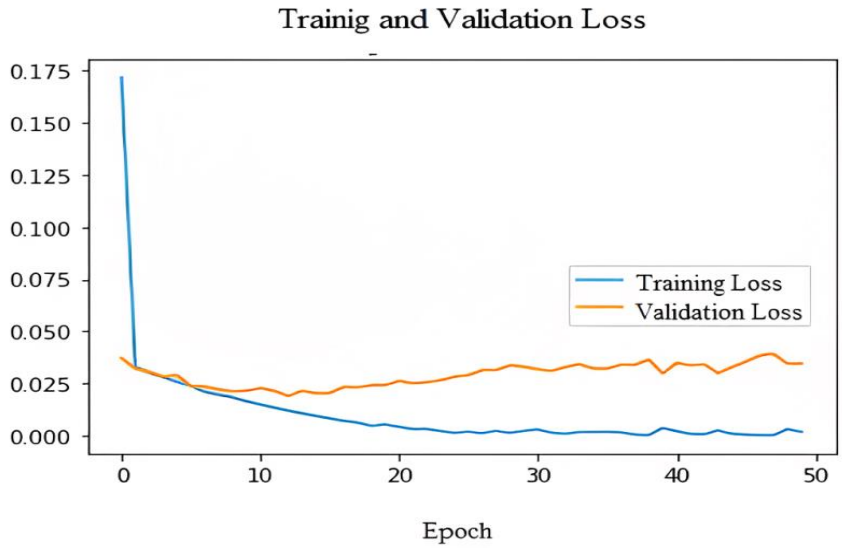


(**a**)

Figure 8. *Cont.*



(**b**)



(**c**)

Figure 8. (**a**) Training and validation loss for LSTM model. (**b**) Training and validation loss for Bi-LSTM model. (**c**) Training and validation loss for GRN-based RNN model.

Table 2. Comparison table for the accuracy of the proposed LSTM, Bi-LSTM, and GRU models.

Model	Accuracy (%)
LSTM	96.9
Bi-LSTM	99.0
GRU	97.5

4.3.3. Comparative Analysis

In the literature, several methods have been proposed for URL detection. A feed-forward neural network was employed to classify URL as legitimate or malicious. One study [38] proposed a model to detect URL using a feed-forward neural network. The malicious URL dataset used contained 48,006 legitimate website URLs. The trained model

exhibited an accuracy of 97%. They performed feature extraction, which reduced 16 features to 2 features. Once the model was trained, to make it easy to test the new link, they deployed the web app using a Python framework named Flask. The user can place the URL, and the model can classify whether the URL is malicious or legitimate.

Figure 9 shows a comparison of the accuracy of the different algorithms, and Table 3 shows extended information about precision, recall, and F1 score. In this figure, we have compared the performance of other existing algorithms such as logistic regression (LR) [32], XGBoost (XGB) [34], multinomial naive Bayes (MNB) [35,36], and k-nearest neighbor (KNN) [37,38]. It can be observed from Table 3, that the accuracies obtained using LSTM, Bi-LSTM, GRU, LR, XGBoost, MNB, and KNN were 97%, 99%, 98%, 96%, 85%, 95.7%, and 92.4%, respectively.

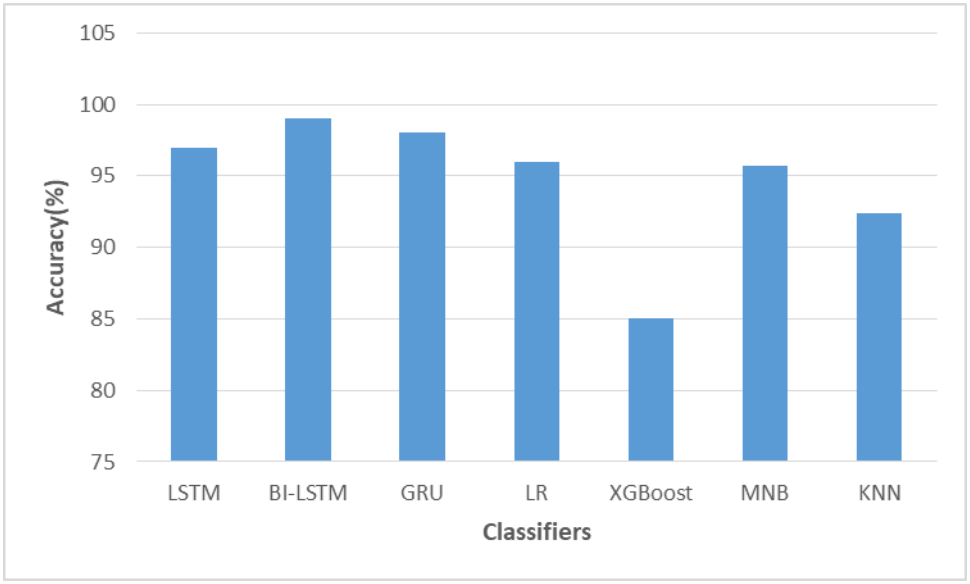


Figure 9. Accuracy comparison of the proposed models and other models.

Table 3. Performance comparison of the proposed model and other machine algorithms used for classifying legitimate and malicious URLs.

Algorithm	Class	Precision	Recall	F1 Score	Accuracy
LSTM	Legitimate	0.99	0.96	0.97	0.97
	Malicious	0.94	0.99	0.96	
Bi-LSTM	Legitimate	0.99	0.99	0.99	0.99
	Malicious	0.99	0.99	0.99	
GRU	Legitimate	0.99	0.97	0.98	0.98
	Malicious	0.95	0.99	0.97	
LR	Legitimate	0.96	0.91	0.93	0.96
	Malicious	0.96	0.99	0.98	
XGBOOST	Legitimate	0.96	0.48	0.64	0.85
	Malicious	0.83	0.99	0.90	
MNB	Legitimate	0.94	0.91	0.93	0.957
	Malicious	0.97	0.98	0.97	
KNN	Legitimate	0.81	0.96	0.88	0.92
	Malicious	0.98	0.91	0.94	

5. Conclusions

Individuals, government organizations, and industries are always subject to phishing attacks. Attackers create a phishing website that imitates a legitimate site to steal personal information. This paper proposed deep learning techniques such as long short-term memory (LSTM), bidirectional long short-term memory (Bi-LSTM), and gated recurrent unit (GRU). The proposed model was tested on URLs public datasets. We used various performance metrics to evaluate the proposed approaches. The experimental results showed that Bi-LSTM produced the best result in all evaluation measures among the three proposed models. The proposed Bi-LSTM model achieved an accuracy of 99.0%. In the future, we would like to use other deep learning algorithms to detect phishing websites using massive, imbalanced datasets.

Author Contributions: The work presented here was performed in collaboration with all the authors. Conceptualization, S.S.R. and A.I.A.; Formal Analysis, L.A.A., M.T.E. and M.A; Writing—review and editing, S.S.R. and A.I.A.; Writing—original draft, S.S.R., A.I.A., L.A.A., M.T.E. and M.A; Supervision, S.S.R.; Visualization, L.A.A., M.T.E. and M.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare that there is no conflict of interest.

References

1. Warburton, D. Phishing Attacks Soar 220% During COVID-19 Peak as Cybercriminal Opportunism Intensifies. Available online: https://www.f5.com/company/news/features/phishing-attacks-soar-220--during-covid-19-peak-as-cybercriminal (accessed on 27 January 2022).
2. Bitaab, M.; Cho, H.; Oest, A.; Zhang, P.; Sun, Z.; Pourmohamad, R.; Kim, D.; Bao, T.; Wang, R.; Shoshitaishvili, Y.; et al. Scam Pandemic: How Attackers Exploit Public Fear through Phishing. In Proceedings of the APWG Symposium on Electronic Crime Research (eCrime), Boston, MA, USA, 16–19 November 2020; pp. 1–10.
3. Agrawal, P.; Mangal, D. A Novel Approach for Phishing URLs Detection. *Int. J. Sci. Res. (IJSR)* **2015**, *5*, 1117–1122. [CrossRef]
4. Rekouche, K. Early phishing. *arXiv* **2011**, arXiv:1106.4692.
5. Gupta, B.B.; Arachchilage, N.A.G.; Psannis, K.E. Defending against phishing attacks: Taxonomy of methods, current issues and future directions. *Telecommun. Syst.* **2017**, *67*, 247–267. [CrossRef]
6. Chung, J.; Koay, J.-Z.; Leau, Y.-B. A Review on Social Media Phishing: Factors and Countermeasures BT—Advances in Cyber Security. In Proceedings of the International Conference on Advances in Cyber Security, Penang, Malaysia, 8–9 December 2020; pp. 657–673.
7. Dinler, Ö.B.; Şahin, C.B. Prediction of phishing web sites with deep learning using WEKA environment. *Avrupa Bilim Teknol. Dergisi.* **2021**, *24*, 35–41. [CrossRef]
8. Carroll, F.; Adejobi, J.A.; Montasari, R. 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.* **2022**, *3*, 170. [CrossRef]
9. Sheng, S.; Wardman, B.; Warner, G.; Cranor, L.F.; Hong, J.; Zhang, C. An empirical analysis of phishing blacklists. In Proceedings of the 6th Conference on Email and Anti-Spam, Mountain View, CA, USA, 16–17 July 2009.
10. Rao, R.S.; Vaishnavi, T.; Pais, A.R. CatchPhish: Detection of phishing websites by inspecting URLs. *J. Ambient. Intell. Humaniz. Comput.* **2019**, *11*, 813–825. [CrossRef]
11. Connor, J.T.; Martin, R.D.; Atlas, L.E. Recurrent neural networks and robust time series prediction. *IEEE Trans. Neural Netw.* **1994**, *5*, 240–254. [CrossRef] [PubMed]
12. Minocha, S.; Singh, B. A novel phishing detection system using binary modified equilibrium optimizer for feature selection. *Comput. Electr. Eng.* **2022**, *98*, 107689. [CrossRef]
13. Balogun, A.O.; Adewole, K.S.; Raheem, M.O.; Akande, O.N.; Usman-Hamza, F.E.; Mabayoje, M.A.; Akintola, A.G.; Asaju-Gbolagade, A.W.; Jimoh, M.K.; Jimoh, R.G.; et al. Improving the phishing website detection using empirical analysis of Function Tree and its variants. *Heliyon* **2021**, *7*, e07437. [CrossRef]
14. Xiao, X.; Xiao, W.; Zhang, D.; Zhang, B.; Hu, G.; Li, Q.; Xia, S. Phishing websites detection via CNN and multi-head self-attention on imbalanced datasets. *Comput. Secur.* **2021**, *108*, 102372. [CrossRef]

15. Li, T.; Kou, G.; Peng, Y. Improving malicious URLs detection via feature engineering: Linear and nonlinear space transformation methods. *Inf. Syst.* **2020**, *91*, 101494. [CrossRef]
16. Abedin, N.F.; Bawm, R.; Sarwar, T.; Saifuddin, M.; Rahman, M.A.; Hossain, S. 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 2020; pp. 1125–1130. [CrossRef]
17. Haynes, K.; Shirazi, H.; Ray, I. Lightweight URL-based phishing detection using natural language processing transformers for mobile devices. *Procedia Comput. Sci.* **2021**, *191*, 127–134. [CrossRef]
18. Feng, F.; Zhou, Q.; Shen, Z.; Yang, X.; Han, L.; Wang, J. The application of a novel neural network in the detection of phishing websites. *J. Ambient Intell. Humaniz. Comput.* **2018**, 1–15. [CrossRef]
19. Smadi, S.; Aslam, N.; Zhang, L. Detection of online phishing email using dynamic evolving neural network based on reinforcement learning. *Decis. Support Syst.* **2018**, *107*, 88–102. [CrossRef]
20. Babagoli, M.; Aghababa, M.P.; Solouk, V. Heuristic nonlinear regression strategy for detecting phishing websites. *Soft Comput.* **2018**, *23*, 4315–4327. [CrossRef]
21. Rao, R.S.; Pais, A.R. Detection of phishing websites using an efficient feature-based machine learning framework. *Neural Comput. Appl.* **2018**, *31*, 3851–3873. [CrossRef]
22. Yasin, A.; Abuhasan, A. An Intelligent Classification Model for Phishing Email Detection. *Int. J. Netw. Secur. Its Appl.* **2016**, *8*, 55–72. [CrossRef]
23. Pascanu, R.; Mikolov, T.; Bengio, Y. On the difficulty of training Recurrent Neural Networks. In Proceedings of the 30th International Conference on International Conference on Machine Learning ICML, Atlanta, GA, USA, 16–21 June 2013.
24. Ilya, S.; Oriol, V.; Quoc, V.L. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems-Volume 2 (NIPS'14)*; MIT Press: Cambridge, MA, USA, 2014; pp. 3104–3112.
25. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
26. Le, X.H.; Ho, H.V.; Lee, G.; Jung, S. Application of Long Short-Term Memory (LSTM) Neural Network for Flood Forecasting. *Water* **2019**, *11*, 1387. [CrossRef]
27. Rahman, L.; Mohammed, N.; Al Azad, A.K. A new LSTM model by introducing biological cell state. In Proceedings of the 2016 3rd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT), Dhaka, Bangladesh, 22–24 September 2016; pp. 1–6. [CrossRef]
28. Graves, A.; Schmidhuber, J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Netw.* **2005**, *18*, 602–610. [CrossRef]
29. Graves, A.; Jaitly, N.; Mohamed, A. Hybrid Speech Recognition with Deep Bidirectional LSTM. In Proceedings of the 2013 IEEE Workshop on Automatic Speech Recognition and Understanding, Olomouc, Czech Republic, 8–12 December 2013; pp. 273–278.
30. Han, P.; Wang, W.; Shi, Q.; Yang, J. Real-time Short-Term Trajectory Prediction Based on GRU Neural Network. In Proceedings of the IEEE/AIAA 38th Digital Avionics Systems Conference (DASC), San Diego, CA, USA, 8–12 September 2019. [CrossRef]
31. Irie, K.; Tüske, Z.; Alkhouli, T.; Schlüter, R.; Ney, H. LSTM, GRU, highway and a bit of attention: An empirical overview for language modeling in speech recognition. In Proceedings of the 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, 8–12 September 2016; pp. 3519–3523. [CrossRef]
32. Wright, R.E. Logistic regression. In *Reading and Understanding Multivariate Statistics*; Grimm, L.G., Yarnold, P.R., Eds.; American Psychological Association: Washington DC, USA, 1995; pp. 217–244.
33. Rahman, M.M.; Watanobe, Y.; Nakamura, K. A bidirectional LSTM language model for code evaluation and repair. *Symmetry.* **2021**, *13*, 247. [CrossRef]
34. Chen, T.; He, T.; Benesty, M.; Khotilovich, V.; Tang, Y.; Cho, H.; Chen, K. Xgboost: Extreme Gradient Boosting. Available online: https://cran.microsoft.com/snapshot/2017-12-11/web/packages/xgboost/vignettes/xgboost.pdf (accessed on 11 September 2022).
35. Kibriya, A.M.; Frank, E.; Pfahringer, B.; Holmes, G. Multinomial naive bayes for text categorization revisited. In Proceedings of the Australasian Joint Conference on Artificial Intelligence, Cairns, Australia, 4–6 December 2004; pp. 488–499.
36. Kumar, S.; Sharma, A.; Reddy, B.K.; Sachan, S.; Jain, V.; Singh, J. An intelligent model based on integrated inverse document frequency and multinomial Naive Bayes for current affairs news categorization. *Int. J. Syst. Assur. Eng. Manag.* **2022**, *13*, 1341–1355. [CrossRef]
37. Peterson, L.E. K-Nearest Neighbor. Available online: http://scholarpedia.org/article/K-nearest_neighbor (accessed on 27 January 2022).
38. Xu, H.; Zhang, L.; Li, P.; Zhu, F. Outlier detection algorithm based on k-nearest neighbors-local outlier factor. *J. Algorithms Comput. Technol.* **2022**, *16*, 17483026221078111. [CrossRef]