A Character Level Word Encoding Deep Learning Model for Combating Cyber Threats in Phishing URL Detection

Rajendran Bhojan

School of Mathematics and Computer Science,
The Papua New Guinea University of Technology, Private mail bag, Lae 411,
Morobe Province, Papua New Guinea

Corresponding author: rajendran.bhojan@gmail.com, rajendran.bhojan@pnguot.ac.pg

Abstract: A cyber threat is generally a malicious activity which tend to damage or steal data or in general something which disrupts the digital life. Security errors, DoS attacks, malware, and data theft are some of these dangers. A kind of cyber threat known as "phishing" occurs when attackers impersonate a legitimate URL or website to collect user information. Out of the total cyber-crimes reported in last quarter around the globe, 21% falls in the phishing category. Phishing is often performed as a social engineering method and the conventional detection techniques were largely relied on the manual reports. Recently, techniques for machine learning have been used to identify phishing. Owing to the recent advancement in the deep learning methods, many possibilities have also been discussed on using the same for achieving better performance. To identify malicious URLs, this research presents a lightweight deep-learning model that can function quite efficient enough even in energy-saving systems. The experimental results of the proposed model showed substantial improvements considering the parameters of comparison to other cutting-edge methods. There was an enhancement in the rate of accurately detecting true positives. Furthermore, the experiments confirmed that the proposed approach operates quite efficiently even when phishing detection systems are in energy-saving modes.

Keywords: Cyber Security, Cyber-crimes, Data Sanitization, Tokenization, Machine Learning, Deep Learning.

1. Introduction

Cyber-crimes and threats are nowadays becoming more frequent which brings out lot of hindrance and struggle for any business and end users to move along digitally. Recent findings from cybersecurity analysts [1] indicate a marked increase in the amount of data that has been compromised and stolen, even from commonplace sources like workplaces, IoT devices, and mobile devices. Additionally, recent studies highlight that many individuals are vulnerable to phishing sites, leading to substantial losses in data and finances. To combat these threats effectively, raising awareness among companies and individuals is essential, along with the implementation of robust computational methods for detecting phishing attacks.

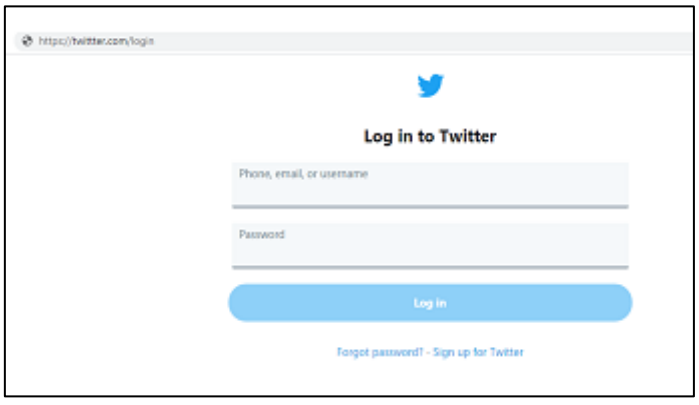

Fig. 1. Phishing Twitter site

Phishing URLs typically employ social engineering tactics to mimic genuine website URLs, intending to deceive users into giving attackers access to private information. Such a scenario, where a well-known social networking site is being phished as illustrated (see Fig. 1). In this example, The URL of the authentic website has an extra character appended, "t", while maintaining an identical appearance to the official site. If visited by an unsuspecting user, this fraudulent site prompts them to enter their credentials, which are then captured by the site's backend and sent to the attackers. In recent times, these types of websites have proliferated and are inflicting significant harm on the digital ecosystem. In the last quarter of 2019 alone, 138,564 phishing sites were discovered, according to the US Council's most recent report on cyber threats. The study emphasizes how rapidly these behaviors have increased, which are increasingly challenging to detect due to attackers' adeptness at employing multiple layers of redirection, thereby obfuscating URLs. The FBI's 2019 report [3] underscores the staggering financial impact, estimating nearly $44 billion in total losses attributed to phishing websites.

Identifying malicious websites is straightforward for cybersecurity experts but challenging for naive computer users who often overlook exact URLs (see Fig. 2). To develop techniques for detecting phishing websites that are based on machine learning, researchers often draw on the experience of security experts. One commonly employed method is URL blacklists, maintained by organizations that anti-cyber threats [4], can provide databases of verified phishing sites in real-time. Anti-cyber threat groups depend more and more on technical contributions from the research community as a result of the notable rapid increase in the number of phishing sites. Maintaining an updated URL blacklist requires active participation from both organizations and individuals. Although these manual techniques are efficient, they are also expensive and time-consuming [5]. Hand-crafted methods can prevent phishing incidents upon repeated encounters but do not provide warnings for initial interactions when users are about to enter their data.

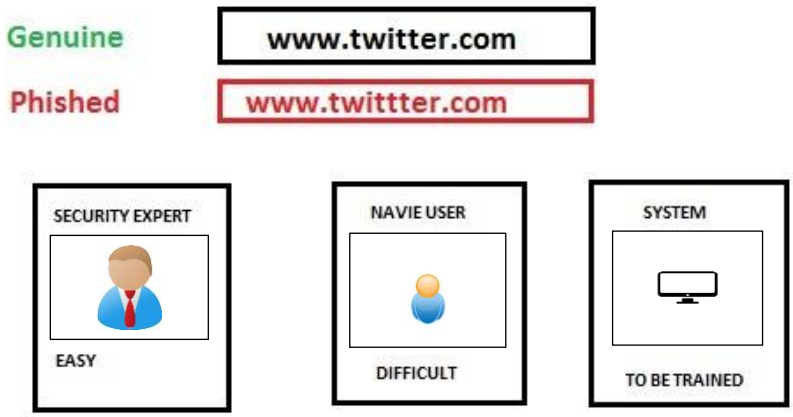

Fig. 2. Difficulty level of detecting phishing

Providing a basic model for phishing detection is the aim of this effort. It presents a strategy based on deep learning to improve the efficacy and precision of phishing using URLs. This study is new in that it investigates if the model can be implemented on devices with limited resources. The suggested technique builds a training set and evaluates its effectiveness using a large dataset of legitimate and phishing URLs. In summary, this paper presents its main contributions as follows:

- Presenting a character-level multi-space technique-based deep learning model for phishing URL detection, exploring improvements over conventional models like CNN.
- Conducting experimental testing and assessments to show the model's efficacy and functionality.

The following is the format of the paper's succeeding sections: While Section 3 offers an overview of previous research addressing related topics; Section 2 expands on the problem statement in depth. Section 4 outlines the research background, followed by Section 5 which details the proposed model, its application,

assessment, and outcome discussion. The paper's conclusion, Section 6, discusses potential directions for further study.

2. Problem Definition

Phishing site detection has been accomplished via the use of conventional machine learning approaches [6-7]. The URLs of websites are first examined to identify characteristics suggestive of malicious intent (see Fig. 3). Subsequently, using traditional machine learning techniques like SVM and k-NN, these labeled features are used by machine learning algorithms to create a training set and create detection models. However, these methods often struggle to detect newly injected phishing sites, relying heavily on existing databases or repositories. Recently, there has been a significant shift towards adopting deep learning methods for this purpose, driven by their successful applications. Unlike traditional approaches that involve complex feature selection processes, deep learning allows ML experts to utilize data directly without explicit feature engineering by cybersecurity experts (see Fig. 3).

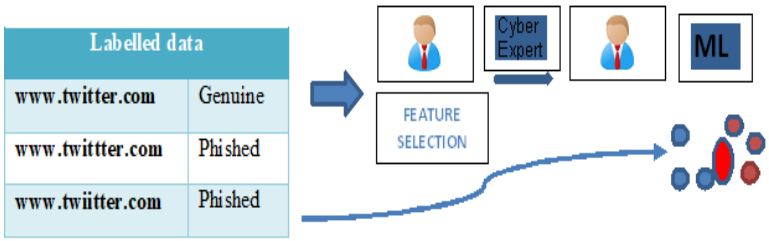

Fig. 3. Problem with Conventional Method

3. Related Works

Recently, phishers target users' financial information, personal information, and other sensitive data, and website phishing has become a serious security concern [10]. The literature has proposed several list-based classical phishing detection techniques, but they have not been able to keep up with the internet's phishing sites growing exponentially. An overview of numerous significant techniques for identifying phishing websites that have been suggested in the literature is given in the following section.

P. Yang et al (2019) the MFP method, introduced in [11], utilizes multi-dimensional feature selection through deep learning. Initially, it extracts character sequences from the current URL and swiftly classifies them using a deep learning approach. Subsequently, it combines features related to statistics and web code into a multi-level feature set. This method significantly reduces the detection time for phishing URLs. Evaluations conducted on a dataset containing thousands of phishing URLs obtained a minimum of 0.61% for false positives and 92.15% for accuracy.

Rao S et al (2019) proposed a categorization technique to overcome the disadvantages of conventional methods by using meta-heuristic characteristics taken from source codes, URLs, and outside services [12]. To assess the model, eight distinct machine-learning techniques were used; random forest performed the best. Multiple evaluations were conducted with various classifiers to determine the optimal approach. The PCA-based random forest (PCA: RF) outperformed other configurations, achieving a high accuracy of 89.61%.

Surya Srikar et al (2020) introduced an anti-phishing mechanism for URLs. Initially, they extracted lexical and host properties of websites. The following phase included training machine learning (ML) and deep learning (DL) models using a combination of host data, URL features, and natural language processing (NLP) properties [13]. The proposed approach achieved a detection rate of 82.5% for identifying phishing URLs.

Yerima et al (2020) introduced a deep learning approach to enhance accuracy in detecting phishing URLs, utilizing traditional CNNs to achieve high classification accuracy [14]. They used 5984 phishing sites and 6598 real websites to test their model. Results from the experiments showed how well the CNN model performed at identifying new phishing URLs. Moreover, the technique produced a 90.36% accuracy rate and an F1 measure of 0.91, outperforming conventional ML classifiers tested on the same dataset.

Somesha et al (2020) introduced a model for phishing URL detection employing deep neural networks. Their approach combined Long Short-Term Memory (LSTM) and CNN features, reducing them to just 10 features [15]. The overall accuracy for the approach was 80.12%. By relying on a single third-party service, their approach enhanced robustness against failures and increased efficiency in the detection process.

Feng et al (2018) introduced an innovative algorithm that uses a neural network-based classification technique to identify phishing websites. The model used a low-risk minimization strategy to improve both accuracy and generalizability. The Monte Carlo approach was also used to simplify and stabilize the training process [16]. The suggested model's usefulness was shown by experimental research, which showed high True Positive and False Negative rates. There was a noticeably high True Positive Rate (TPR) and False Negative Rate (FNR). Additionally, studies showed that the accuracy level was around 92%.

W. Ali et al (2019) enhance phishing prediction ability, a hybrid model integrating evolutionary feature selection techniques and Deep Neural Networks (DNNs) was developed [17]. In this model, to determine which traits are the most important and to assign them the proper weights, a heuristic technique called GA is used. These selected website features are then weighted by GA for training to enhance accurate phishing site prediction. The experimental findings show that in contrast to other methods, the recommended technique performs higher in terms of specificity, sensitivity, and Mean Squared Error (MSE).

4. Background and Motivation

Detecting phishing websites computationally is typically approached as a supervised machine learning problem. Models are trained using elements from websites, including URLs, methods, and HTML text, to perform harmful detection tests. For phishing detection, the classifier is an essential component of many machine-learning techniques [18]. A typical supervised learning procedure for identifying dangerous websites (see Fig. 4), since highly informational features may greatly improve speed, feature selection is an important first step in these approaches. Optimal feature selection requires expert knowledge, and selecting common features can be challenging due to variations across different application domains. Incorrectly selecting features can lead to substantial loss of significant information. Therefore, deep learning methods are suitable because they eliminate the need for feature selection, thereby enabling faster system performance without degradation from inappropriate feature choices. Deep learning models can utilize unprocessed data directly for training, simplifying the process, and effectively identifying patterns for accurate decision-making. Inspired by these benefits, this work presents a deep learning model based on deep learning principles that develop a phishing detection system using raw Uniform Resource Locators (URLs).

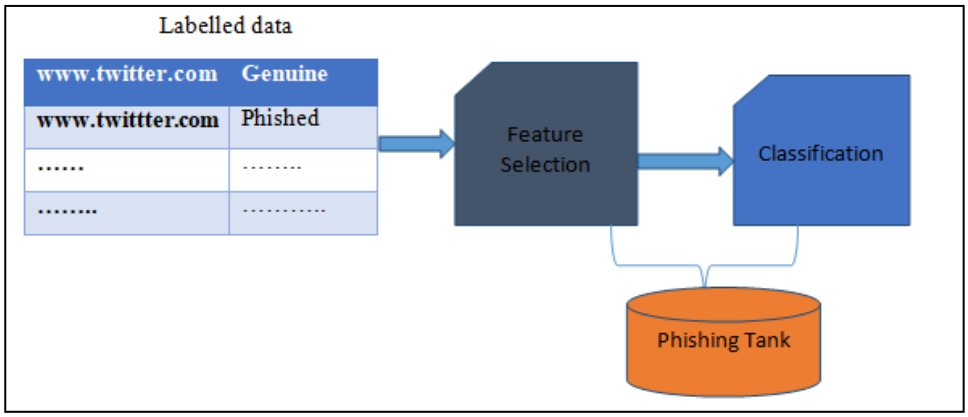

Fig. 4. Classical Phishing detection

5. Proposed Methodology

Deep Learning Framework: Phishing URL Detection (DLF:PUD)

The suggested methodology DLF:PUD (see Fig. 5).

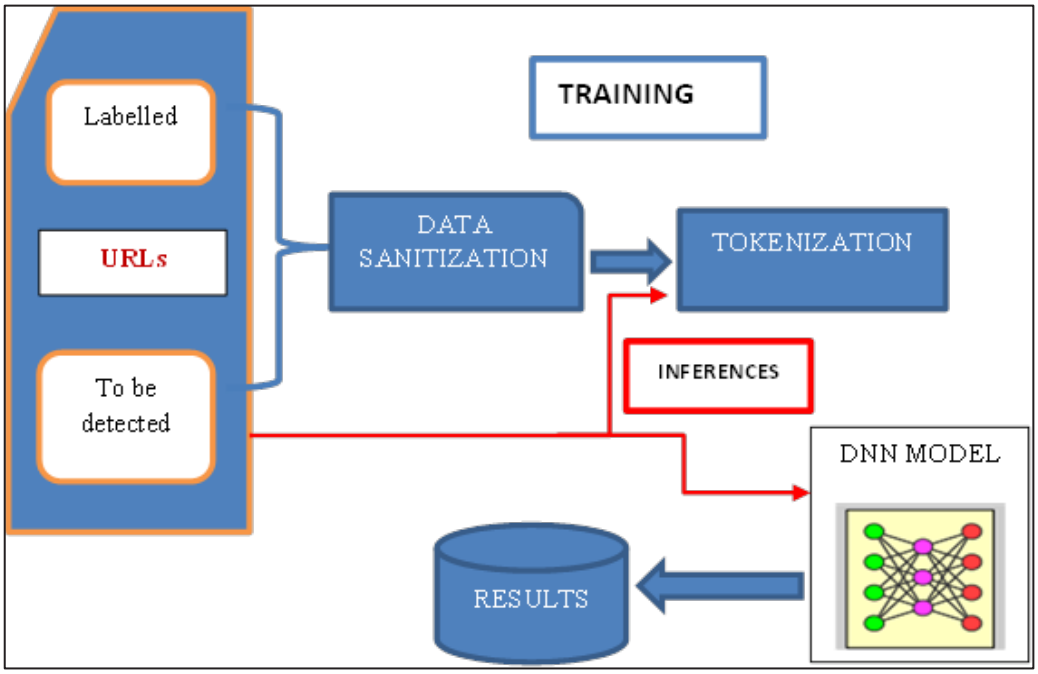

Fig. 5. The proposed Methodology

5.1 Data Sanitization

Sanitizing data is the first step in the suggested model. To avoid their impact on the detection performance of phishing URLs across various datasets, this procedure eliminates popular URL prefixes like "http://" and "https://". Failure to remove these prefixes could lead to inconsistency in URL representations, significantly impacting the model's accuracy. For instance, if a phishing repository's URLs all have the same prefix, the model might incorrectly classify all URLs with that prefix as phishing sites. To vectorize the characters in the URL, a tokenizer is employed. The proposed method utilizes character-level tokenization to ensure that the model does not rely on the semantic meaning of words within URLs, which is crucial since many phishing site names are constructed at the character level. Malicious URLs often contain slight variations from the original site that can easily go unnoticed by users. For example, as per the case under discussion, the original website "twitter.com" becomes "ttwitter.com" when the character "t" is added. There are several layers in the proposed Deep Neural Network (DNN), including embedding, convolutional, dropout, and sigmoid layers (see Fig. 6). Table 1 provides a comprehensive overview of the layer setup of the DNN model, including information on the kernel's size, the filters that were used, and the word embedding layer's output.

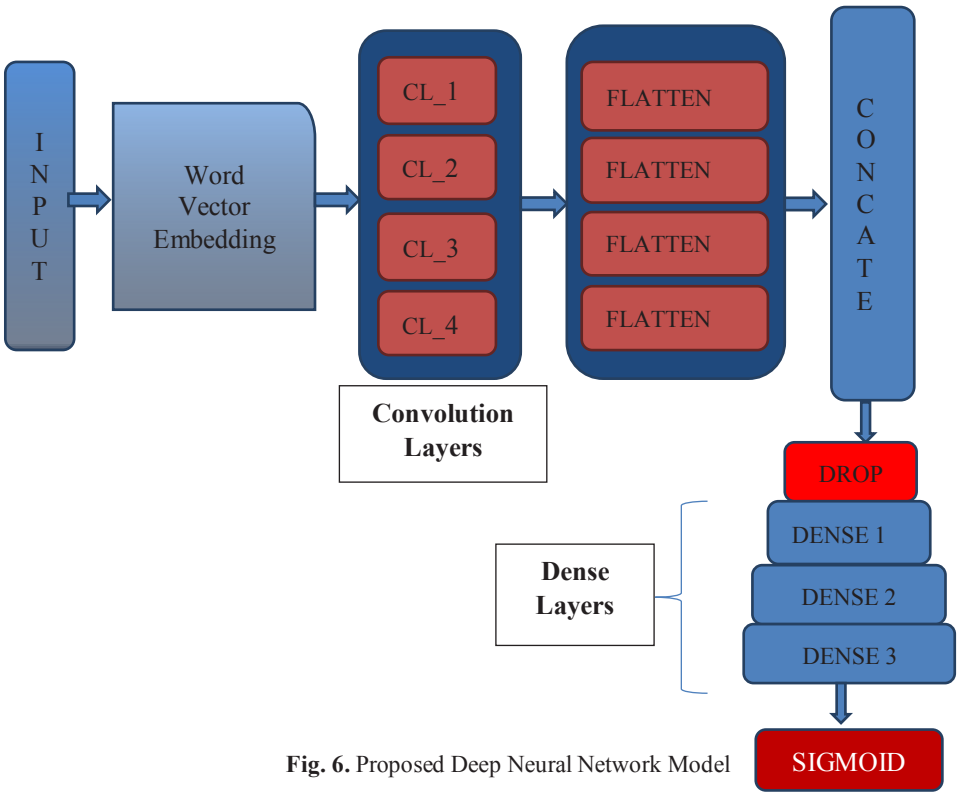

Fig. 6. Proposed Deep Neural Network Model

Table 1. Configuration of proposed DNN Model

Dimensions of Output		
Embedding	32	
	Filter Size	**Kernel Size**
CL_1	128	4
CL_2	128	6
CL_3	128	10
CL_4	128	20
	Dropout Rate	
Drop out	0.5	
	Number of Units	
Dense_1	64	
Dense_2	64	
Dense_3	64	

Each layer of the suggested model is described below**.**

Embedded Layer: This layer serves as the initial stage in Deep Neural Network models for addressing NLP tasks. Alongside tokenization, a vector output is produced by the embedding layer. The differences between word embedding and one-hot encoding (see Fig. 7, 8). The coefficients of each vector in the word embedding layer depict relationships among different characters, enhancing NLP performance by visualizing these relationships.

Convolution Layer: In the suggested paradigm, after the embedding layer, four layers of this type are employed. Each layer applies a kernel and filter to extract significant features and filter out less relevant information. The operation involves element-wise multiplication, with summaries occurring between the filters and the relevant data portions. Instead of sequential processing, parallel layers are utilized. Every layer is set up to extract characteristics by concentrating on a single window containing successive characters. Concatenation is performed after each convolutional layer's outputs have been flattened.

Concatenating layer: To make further processing easier, specifically, this layer is designed to combine properties from earlier levels. Unlike conventional concatenation methods, convolutional layers' (CLs') outputs combine with the embedding layer's outputs in this instance. The original context of the data is preserved using this method, which is essential for identifying phishing URLs.

Drop layer: During training, overfitting is avoided with the help of this regularization procedure. During training, neurons are chosen at random and turned off. These deactivated neurons are temporarily excluded during forward passes, and backward passes do not update their respective weights.

Dense layers: The suggested model is improved by adding more capabilities to extract the most important information since this layer is completely concatenated. To analyze patterns obtained from the convolutional layers, these attributes are essential.

Sigmoid layers: These utilize the sigmoid function to detect phishing URLs. The sigmoid function outputs values between 0 and 1, which are used to compute prediction probabilities in the last layers of the suggested DNN model.

t	1	0	0	0	0	0	0	0
w	0	1	0	0	0	0	0	0
i	0	0	1	0	0	0	0	0
t	0	0	0	1	0	0	0	0
t	0	0	0	0	1	0	0	0
e	0	0	0	0	0	1	0	0
r	0	0	0	0	0	0	1	0
.	.	.	.	.	.	.	-	-
.	.	.	.	.	.	.	-	-
c	0	0	0	0	0	0	0	1
o	0	0	0	0	0	0	0	0
m	0	0	0	0	0	0	0	0

Fig. 7. Hot Encoding

t	0.13	0.85	0.76	0.12	0.14	0.26	0.75	0.62
w	0.15	0.23	0.42	0.32	0.14	0.34	0.91	0.72
i	0.37	0.65	0.72	0.32	0.61	0.42	0.32	0.41
t	0.45	0.32	0.15	0.38	0.74	0.38	0.86	0.84
t	0.94	0.32	0.18	0.37	0.67	0.52	0.65	0.74
e	0.56	0.61	0.84	0.38	0.81	0.91	0.52	0.38
r	0.85	0.32	0.17	0.45	0.37	0.64	0.49	0.38
.	-	-	-	-	-	-	-	-
.	-	-	-	-	-	-	-	-
c	0.52	0.34	0.51	0.53	0.47	0.68	0.92	0.19
o	0.37	0.65	0.72	0.32	0.61	0.42	0.32	0.41
m	0.21	0.29	0.68	0.75	0.37	0.19	0.83	0.91

Fig. 8. Word Embedding

5.2 Evaluation

In the next section, the suggested DNN model is evaluated using settings given in Table 1. The experiments were conducted on a system equipped with a GPU, minimal RAM and ROM, and a standard graphics card. The experiment included 10,234 URLs in all, 3,004 phishing URLs, and 7,234 genuine URLs. Once redundant URLs were eliminated before usage, the official repository is where the benign URLs were obtained. For training, 10% of all URLs were set aside. The accuracy measure used was the number of True Positive (TP) events. Section 6 contains specifics on the performance measures.

The proposed model achieved a True Positive (TP) accuracy of 98.13%. Other DNN-based methods for detecting phishing URLs have utilized similar structures, even when the dense and convolutional (CL) layer designs vary. As a result, the effects of these suggested layers are examined. The impact of the suggested thick layers is seen in Table 2 which is graphically represented (see Fig. 9). When the number of thick layers was first investigated, the results showed that the TP accuracy increased steadily with each new layer. When the suggested model reached the sigmoid layer, its accuracy reached its maximum of 98.13%.

Table 2 – Effect of 3- Tier Dense layer

	Accuracy
Proposed	98.13 %
1 Dense Layer	82.65 %
2 Dense Layer	84.26 %
3 Dense Layer	86. 58 %

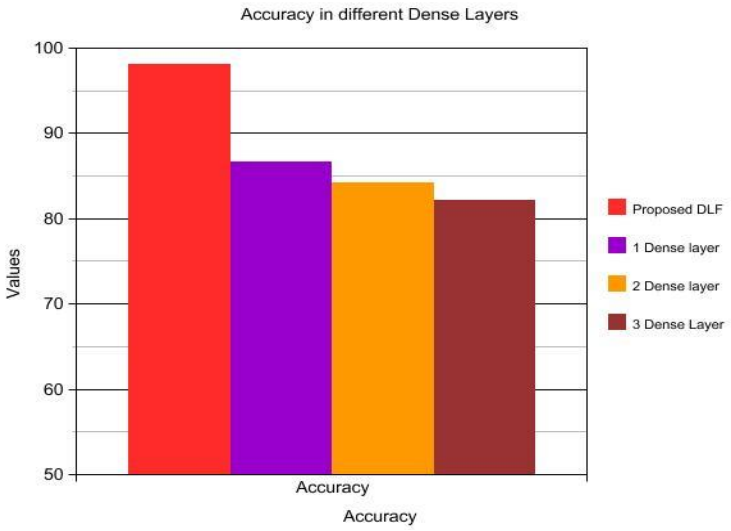

Fig. 9. Comparison of several dense layers' accuracy

The convolutional layers' effects are shown in Table 3 and represented graphically (see Fig. 10). A comparable trend indicates that the number of CL layers grows with increasing performance. In the proposed DNN model, using CL1, CL2, and CL3 resulted in incremental TP rates of 80.15%, 82.65%, and 84.21%, respectively.

Table 3. Effect of Convolutional Layers

	Accuracy
Proposed	98.13%
3 Convolutional Layer	84.21%
2 Convolutional Layer	82.65%
1 Convolutional Layer	80.15%

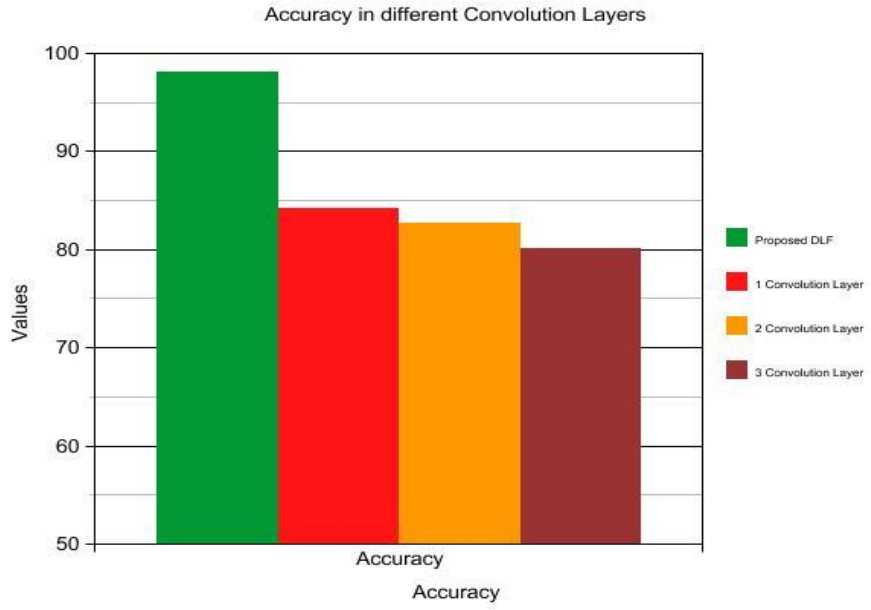

Fig. 10. Evaluation of Various Convolutional Layers' Accuracy

The proposed model also aimed to concatenate outputs from word embedding to retain unprocessed information, resulting in improved performance. Table 4 shows embedding layer outputs with and without concatenation. It shows a significant decrease in TP values when concatenation is not utilized.

Table 4. Effect of Concatenation

True Positive	
Proposed	98.13%
Without Concatenation	83.41%

5.3 Implementation with Use-Case

Performance evaluation of the proposed model, it is integrated into a low-resource device. Computational complexities are computed for assessment at every stage of the model. The execution timings for each stage, including DNN inference, tokenization, and sanitization, are shown in Table 5. Ten trials could be conducted, and the average execution times were determined. A significant amount of the overall execution time was accounted for by the inference layer, which took 105 milliseconds for each URL request. Sanitization and tokenization, on the other hand, required less than a nanosecond for each step. The overall assessment time per URL was 125 milliseconds on average, which is a noteworthy accomplishment in contexts with limited resources.

Table 5. Execution Time

Execution Time	
Data Sanitization	0.01265
Tokenization	0.195
Deep Neural Network model	124

Subsequent research included applying word- and character-level embedding to the suggested model. Throughout the embedding, the parameters stayed the same, for example, the number of convolutional and dense layers. Convolutional layer outputs were concatenated and then sent via sigmoid and dropout layers for further processing. The open-source Pi tool was used for evaluation, although it ran into an out-of-memory fault. This suggests that in contexts with limited resources, execution is not possible. This emphasizes how effective the suggested model is. In contrast, the suggested model and the model that included

embedding at the word and character levels were implemented independently. The suggested DNN model was executed in 67 milliseconds, according to the results, while the model with both embedding took 97 milliseconds. Therefore, the proposed model demonstrated 30% greater efficiency.

5.4 Comparative Analysis

The proposed model is evaluated comparatively across various parameters as outlined below:

Performance Metrics

The suggested model's overall performance is evaluated and errors are identified using statistical parameters. The following are some of the many metrics that are used in assessment. The primary assessment criteria are shown in Table 6.1.

Table 6.1: Fundamental Evaluation Metric

	Includes the condition or target object	Contains the desired item or circumstance
Tests Negative or Accepted Null Condition	True Negative	False Negative
Tests Positive or Rejected Null Condition	False Positive	True Positive

i. True Positive

A true positive is a result that is positive and rejects the null hypothesis, with the final condition being recognized as matching and accurate. A is a symbol of true positive. It has the following definition:

$$TP = n_{11} = \text{number of such individuals} \quad (1)$$

ii. True Negative

When it is said that the ultimate condition is correct and non-matching, it is considered a true negative, which is a finding that supports the null hypothesis. The symbol B stands for true negative. It has the following definition:

$$TN = n_{00} = \text{number of such individuals} \quad (2)$$

iii. False Positive

An inaccurate rejection of the null hypothesis is known as a false positive, this occurs when a positive result is reported yet the final condition is shown as matching. The letter C represents a false positive. The following is its definition:

$$FP = n_{01} = \text{number of such individuals} \quad (3)$$

iv. False Negative

The term "false negative" refers to a negative result that erroneously accepts the null hypothesis when it is said that the final condition is not matching and is incorrect. A false negative is represented by the symbol D. The following is its definition:

$$FN = n_{10} = \text{number of such individuals} \quad (4)$$

v. Recall

It assesses the model's overall performance, especially when false-negative situations are present, using a probability-based accuracy test. This is determined as

$$Recall := \frac{True\ Positive}{True\ Positive + False\ Negative} \quad (5)$$

vi. Precision

The accuracy of the model is tested using probability to show how well it performs overall when there are positive results. It is computed as

$$Precision := \frac{True\ Positive}{True\ Positive + False\ Positive} \quad (6)$$

vii. F1-Measure

The combined effect of recall and accuracy in a situation is assessed using this cumulative metric. A case's accuracy and recall ranges will determine whether the F-Measure is expressed between 0 and 1, or between 0 and 100. It is measured by

$$F1 - Measure := 2 * \frac{(R*p)}{R+P} \quad (7)$$

where p or P refers to precision and R for recall.

Table 6.2. Comparing Precision, Recall and F1-Measure.

METHODS	PRECISION	RECALL	F1-MEASURE
DLF:PUD	98.18	98.15	98.14
MFP	92.54	93.21	92.54
PCA:RF	90.21	89.26	89.25
DL:CNN	81.25	80.24	80.65
NB-LSTM	91.25	92.54	92.5
GA	80.21	81.25	80.75

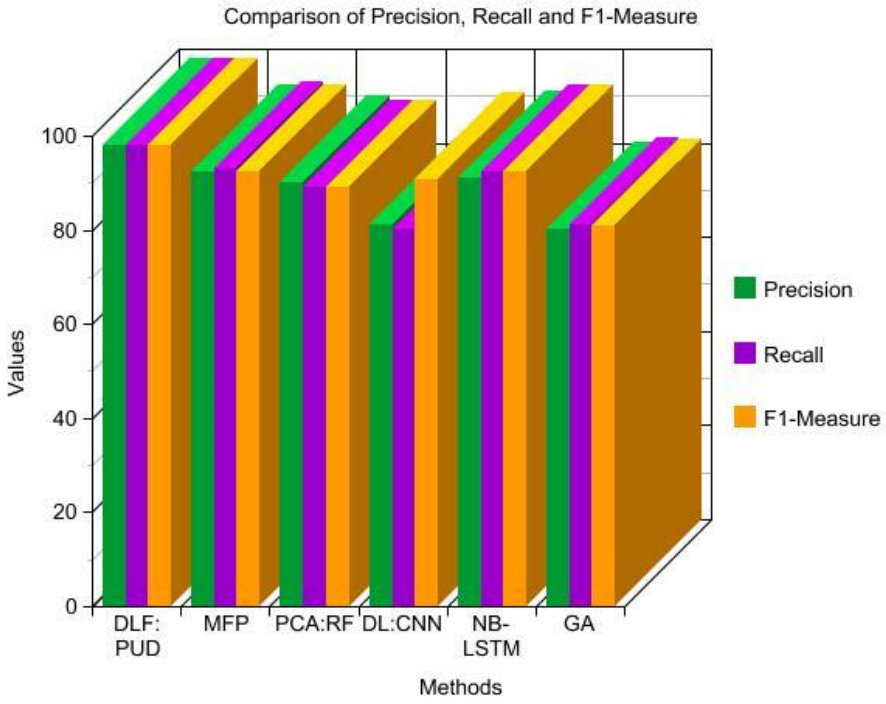

Fig. 11. Precision, F1, and Recall Comparison

Table 6.2 presents comparative data for performance metrics including Precision, Recall, and F1-Measure, as depicted graphically (see Fig. 11). The findings show that the suggested approach outperforms current approaches.

i. Accuracy

Dividing the total number of true occurrences by the total number of cases to get the overall accuracy value.

$$Accuracy := \frac{True\ Positive + True\ Negative}{True\ Positive + True\ Negative + False\ Positive + False\ Negative} \quad (8)$$

Table 6.3 presents the accuracy comparison statistics, which is graphically represented (see Fig. 12). It is observed that compared to the other approaches, the suggested method TFC: LSTM is more accurate.

Table 6.3. Accuracy Comparison

METHODS	Accuracy
DLF:PUD	98.13
MFP	92.15
PCA:RF	89.61
DL:CNN	80.12
NB-LSTM	92.50
GA	80.79

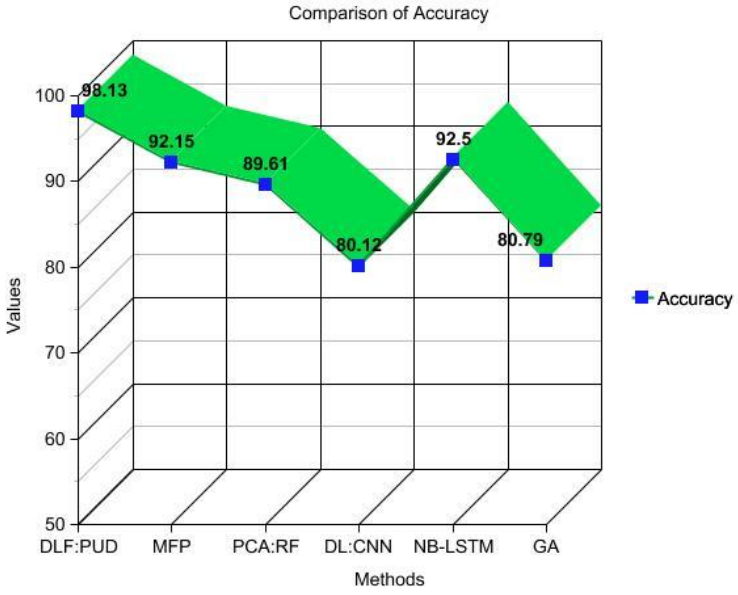

Fig. 12. Accuracy Curve

6. Conclusion

Phishing websites pose a significant cybersecurity threat that demands urgent attention from researchers. Despite numerous conventional methods proposed to address this issue, they often suffer from performance limitations, particularly in execution time due to the additional work involved in choosing features. To successfully identify phishing URLs even in resource-constrained contexts, this study develops a Deep Neural Network (DNN) model. The suggested method replaces the conventional word-level embedding with character-level embedding. Experimental evaluation demonstrated its efficiency, achieving a record-high accuracy of 98.13%. Continued assessment of the character-level embedding revealed a 30% reduction in execution time compared to traditional methods employing multiple word embedding's. Even in settings with limited resources, the outcomes were encouraging. The suggested model's future developments will concentrate on using deep learning to identify phishing websites based on online content.

References

1. Humayun, M., Niazi, M., Jhanjhi, N. et al. Cyber Security Threats and Vulnerabilities: A Systematic Mapping Study. Arab J Sci Eng 45, 3171–3189 (2020)
2. L. Liu, O. De Vel, Q. Han, J. Zhang and Y. Xiang, "Detecting and Preventing Cyber Insider Threats: A Survey," in IEEE Communications Surveys & Tutorials, vol. 20, no. 2, pp. 1397-1417, Secondquarter 2018,
3. https://www.fbi.gov/investigate/cyber
4. Jain A.K., Gupta B.B. (2018) PHISH-SAFE: URL Features-Based Phishing Detection System Using Machine Learning. In: Bokhari M., Agrawal N., Saini D. (eds) Cyber Security. Advances in Intelligent Systems and Computing, vol 729. Springer, Singapore
5. El Aassal, S. Baki, A. Das and R. M. Verma, "An In-Depth Benchmarking and Evaluation of Phishing Detection Research for Security Needs," in IEEE Access, vol. 8, pp. 22170-22192, 2020
6. Khonji, Mahmoud & Iraqi, Youssef & Jones, Andy. (2013). Phishing Detection: A Literature Survey. IEEE Communications Surveys &amp Tutorials. PP. 1-31. 10.1109/SURV.2013.032213.00009.
7. A.A. Orunsolu, A.S. Sodiya, A.T. Akinwale, A predictive model for phishing detection, Journal of King Saud University - Computer and Information Sciences, 2019
8. P. Yang, G. Zhao and P. Zeng, "Phishing Website Detection Based on Multidimensional Features Driven by Deep Learning," in IEEE Access, vol. 7, pp. 15196-15209, 2019
9. Phishing Attack Trends Re-Port-1Q, May 2018, [online] Available: https://apwg.org/resources/apwg-reports/.
10. Y. Ahmad, M. Selvakumar, A. Mohammed and A.-S. Samer, "TrustQR: A new technique for the detection of phishing attacks on QR code", Adv. Sci. Lett., vol. 22, pp. 2905-2909, Oct. 2016
11. P. Yang, G. Zhao and P. Zeng, "Phishing Website Detection Based on Multidimensional Features Driven by Deep Learning," in IEEE Access, vol. 7, pp. 15196-15209, 2019
12. Rao, R.S., Pais, A.R. Detection of phishing websites using an efficient feature-based machine learning framework. Neural Comput & Applic 31, 3851–3873 (2019)
13. Surya Srikar Sirigineedi, Jayesh Soni, and Himanshu Upadhyay. 2020. Learning-based models to detect runtime phishing activities using URLs. In Proceedings of the 2020 the 4th International Conference on Compute and Data Analysis (ICCDA 2020). Association for Computing Machinery, New York, NY, USA, 102–106
14. S. Y. Yerima and M. K. Alzaylaee, "High Accuracy Phishing Detection Based on Convolutional Neural Networks," 2020 3rd International Conference on Computer Applications & Information Security (ICCAIS), Riyadh, Saudi Arabia, 2020
15. Somesha, M., Pais, A.R., Rao, R.S. et al. Efficient deep learning techniques for the detection of phishing websites. Sadhana 45, 165 (2020)
16. Feng, F., Zhou, Q., Shen, Z. et al. The application of a novel neural network in the detection of phishing websites. J Ambient Intell Human Comput (2018)
17. W. Ali and A. A. Ahmed, "Hybrid intelligent phishing website prediction using deep neural networks with genetic algorithm-based feature selection and weighting," in IET Information Security, vol. 13, no. 6, pp. 659-669, 11 2019
18. Rao, R.S., Vaishnavi, T. & Pais, A.R. CatchPhish: detection of phishing websites by inspecting URLs. J Ambient Intell Human Comput 11, 813–825 (2020)