Neural Computing and Applications
https://doi.org/10.1007/s00521-021-06401-z

S.I. : MACHINE LEARNING APPLICATIONS FOR SECURITY

A hybrid DNN–LSTM model for detecting phishing URLs

Alper Ozcan[1,2] · Cagatay Catal[3] · Emrah Donmez[4] · Behcet Senturk[5]

Received: 3 January 2021 / Accepted: 29 July 2021

Abstract
Phishing is an attack targeting to imitate the official websites of corporations such as banks, e-commerce, financial institutions, and governmental institutions. Phishing websites aim to access and retrieve users' important information such as personal identification, social security number, password, e-mail, credit card, and other account information. Several anti-phishing techniques have been developed to cope with the increasing number of phishing attacks so far. Machine learning and particularly, deep learning algorithms are nowadays the most crucial techniques used to detect and prevent phishing attacks because of their strong learning abilities on massive datasets and their state-of-the-art results in many classification problems. Previously, two types of feature extraction techniques [i.e., character embedding-based and manual natural language processing (NLP) feature extraction] were used in isolation. However, researchers did not consolidate these features and therefore, the performance was not remarkable. Unlike previous works, our study presented an approach that utilizes both feature extraction techniques. We discussed how to combine these feature extraction techniques to fully utilize from the available data. This paper proposes hybrid deep learning models based on long short-term memory and deep neural network algorithms for detecting phishing uniform resource locator and evaluates the performance of the models on phishing datasets. The proposed hybrid deep learning models utilize both character embedding and NLP features, thereby simultaneously exploiting deep connections between characters and revealing NLP-based high-level connections. Experimental results showed that the proposed models achieve superior performance than the other phishing detection models in terms of accuracy metric.

Keywords Phishing · Machine learning · Deep learning · Phishing detection

✉ Alper Ozcan
alper.ozcan@nisantasi.edu.tr; alperozcan@akdeniz.edu.tr

Cagatay Catal
ccatal@qu.edu.qa

Emrah Donmez
emrah.donmez@ozal.edu.tr

Behcet Senturk
behcet.senturk@ogrenci.ankara.edu.tr

1 Department of Computer Engineering, Nisantasi University, Istanbul, Turkey

2 Department of Computer Engineering, Akdeniz University, Antalya, Turkey

3 Department of Computer Science and Engineering, Qatar University, Doha, Qatar

4 Department of Management Information Systems, Turgut Ozal University, Malatya, Turkey

5 Department of Computer Science and Engineering, Ankara University, Ankara, Turkey

1 Introduction

Phishing is a cyber-attack based on social engineering, which aims to steal confidential data such as credit card numbers, login credentials, and passwords. Most of the time, the attacker registers a fake domain address, designs a website that mimics the actual website of the organization carefully, and sends a mass email to thousands of people to instruct recipients to click on a link in the fake website. These types of emails apply different kinds of threats, scare users to take some actions that the attackers want, and redirect users to a webpage designed to impersonate the login page of a real website.

Nowadays, the term phishing is widely used in traditional media, social media, and scientific literature. Since different researchers present their definitions of phishing, there exist a large number of definitions in the literature [17]. One of these definitions used by The Anti-Phishing Working Group (APWG) is as follows [6]: "Phishing is a crime employing both social engineering and technical subterfuge to steal consumers' identity data and financial account credentials." Because of diverse definitions, Lastdrager [40] performed a study on the definition of phishing and proposed the following consensual definition to allow future research to be aligned: "Phishing is a scalable act of deception whereby impersonation is used to obtain information from a target" [40].

A phishing attack is generally characterized with the following three aspects [54]:

- A legitimate entity is spoofed.
- A website is used for the spoofing process.
- Confidential information is requested and retrieved.

These attacks mostly result in [3] the loss of confidential customer information, financial loss, the loss of Intellectual Property (IP), and weakening the trust [63] and national security [54]. The Anti-Phishing Working Group (APWG), which is an international coalition of 2200 institutions, publishes phishing activity trend reports each year. In 2020, software-as-a-service (SaaS)/webmail users were the biggest targets of phishing (i.e., 34% of all attacks) [6]. The number of phishing attacks particularly increased after the Coronavirus disease 2019 (COVID-19) pandemic (i.e., mid-March 2020), and attacks used the COVID-19 themes [6] against healthcare facilities and workers. About 70% of the healthcare attacks addressed facilities having less than 500 employees because these small facilities have probably weaker security systems due to smaller security budgets [6]. The largest scam request was $976,522; however, the problem was discovered before the money was transferred [6]. It was also reported that 75% of phishing websites use SSL (Secure Sockets Layer) protection and as such, a website using the SSL protocol is not always a legitimate website. As seen in these examples, a phishing attack is one of the most critical attacks for organizations and internet users.

There exist seven types of communication media: email, website, online social network, messenger, blogs & forums, mobile apps, voice over IP (VOIP). These communication platforms can be targeted by phishing attacks. These attacks mostly target the devices like computers, VOIP devices, and smart devices [10]. Phishing attack approaches can be categorized into the following two main categories [10]:

- Attack Launching: Some of the attack launching techniques are as follows: URL spoofing, email spoofing, social network collaboration, the man in the middle attack, spear phishing, website phishing, spoofed mobile browser, reverse social engineering, intelligent voice reaction, and abusing settings of social networks.
- Data Collection: There exist two basic data collection approaches. The first one is the automated data collection using key loggers, recorded messages, and fake website pages. The other category is the manual data collection using social engineering.

Counter measurements can be categorized into the following four main categories [10]:

- Machine Learning (ML): Traditional machine learning algorithms (a.k.a., shallow learning) such as Support Vector Machines (SVM), Decision Trees (DT), and Random Forests (RF) can be applied to develop phishing detection models.
- Deep Learning (DL): Deep learning is a sub-branch of ML and focuses on developing deep networks using different layer types such as convolution layer, pooling layer, dropout layer, and fully connected layer (a.k.a., dense layer). Algorithms such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Autoencoders, Restricted Boltzmann Machine (RBM), Deep Belief Network, and Deep Neural Networks can be used to develop deep learning-based phishing detection models.
- Scenario-based Techniques: Different scenarios are used to detect the attacks.
- Hybrid Techniques: A combination of different approaches is used to create a better model in terms of accuracy and precision.

From the machine learning perspective, the phishing detection problem can be considered as a binary classification task (i.e., legitimate or fake). Deep learning algorithms have been applied successfully for several classification problems [7, 9] including text classification [14]. Recent studies demonstrated that RNN algorithms can provide higher performance in phishing URL detection [45].

RNN's ability that differs from other neural networks is that they can learn the connections between sequential structures. To achieve this task, RNNs process one item at a time [59]. Since each character in the URL has a semantic relationship with the character on its right and left sides, RNNs can learn the connections between sequential characters. RNN-based methods receive URL characters directly as input and they do not need manual feature extraction to classify URLs. Each input character is translated by a 128-dimension embedding. The translated URL is padded as a 150-step sequence, as expressed in [8] to make it usable for feeding models. In this study, a specific RNN cell called LSTM is used because the vanishing gradient problem of the simple RNN that limits the training of deep RNNs is avoided in this algorithm.

The main objective of this research is to develop a better phishing detection model based on deep learning algorithms in terms of accuracy metric. Another important consideration of this research is to avoid the undesired complexities. One difficulty in the current phishing prediction models is the complexity of the extracted feature sets. There is no consensus among researchers on the selection of the features and therefore, each model proposes and investigates several combinations of different feature types, resulting in complicated models. Some models have too much focus on character embedding-based features and do not consider the high-level hand-crafted connections from NLP features. The other category of approaches focuses on only high-level connections and dismisses the character embedding-based features. In the proposed models, the power of these two strategies are combined in a way that the prediction model can detect phishing attacks effectively.

This paper focuses on using DL algorithms for detecting phishing URLs. Particularly, a novel hybrid deep learning model, which combines the power of a Deep Neural Network (DNN) model using the lexical and statistical analysis of URLs and a long short-term memory network (LSTM)-based model using character embedding-based features, was proposed and validated using phishing detection datasets.

In previous studies, researchers applied DNN and LSTM algorithms separately to solve this challenging problem, however, each of these algorithms has its own pros and cons. In this study, we demonstrated how to integrate these two algorithms to fully utilize the power of these algorithms, therefore, a novel hybrid neural network was proposed and validated in this study. This proposed network is able to combine character embedding-based features and the manual NLP features.

Using character embedding with CNN-based models had the following limitations: (1) CNN-based models had high memory needs (2) CNN-based models could not find long-distance dependent features. Novel hybrid architecture that uses RNN-based models instead of CNN-based models can cope with this challenge [74].

The main contributions of this paper are threefold shown as follows:

- Novel hybrid phishing detection models that combine the power of hand-crafted features and character embedding-based features were proposed and evaluated in this study. This allows the proposed models to extract salient deep features by distilling and combining information from the given features. To the best of our knowledge, deep learning-based phishing models in literature preferred one of these two feature engineering strategies instead of combining them.
- Thanks to the novel hybrid architecture that integrates two kinds of feature sets, the proposed models demonstrated better performance than the other models proposed in the literature in terms of accuracy metric.
- In the proposed architecture, the general model initially consists of two different parts that are DNN and LSTM-based. Later, these two parts are connected at a certain point. Finally, the model turns into a standard DNN model, ending with a final node used to classify the URLs. As such, the error value to be used during the back-propagation is the same for both the DNN and the LSTM-based parts. This ensures that the two parts are optimized in harmony. With the help of this architecture, an optimized model is built compared to the models that use character embedding-based attributes and hand-crafted attributes separately.

The following sections are organized as follows: Section 2 presents the related work. Section 3 explains the methodology, Sect. 4 discusses the experimental results, and Sect. 5 presents the discussion. Section 6 concludes the paper.

2 Related work

Many research papers have been published so far on the phishing detection problem [37]. In the following subsections, survey articles, general phishing detection models, and deep learning-based studies are presented. In Sect. 2.1, survey articles published in this research field are discussed briefly. In Sect. 2.2, some of the important phishing detection studies are explained. In Sect. 2.3, selected deep learning-based studies are discussed.

2.1 Selected survey studies on phishing detection

The survey article of [37] categorized phishing detection studies into the following four main categories: Blacklist-based detection, rule-based heuristics, visual similarity, and machine learning-based detection models. They reported that machine learning-based models provide promising results, which achieve high accuracy while detecting zero-hour attacks.

In their survey article, [65] presented the following six phishing website detection categories: Search engine-based, heuristics and machine learning-based, phishing blacklist and white list-based, visual similarity-based, domain name system (DNS)-based, and proactive phishing URL-based approaches. They stated that the search engine-based approaches are the easiest solutions, however, there are several challenges of using this type of detection model.

Aleroud and Zhou [3] categorized phishing countermeasures (i.e., preventing/detecting attacks) into the following five categories: Machine learning, text mining, human users, profile matching (e.g., blacklists, visual matching), and others (e.g., search engines, ontology, client-server authentication). They reported that machine learning, text mining, and human users-based approaches are the most widely used techniques and semantics-based techniques in the other category are overlooked. Also, they concluded that these studies focused on e-mails and websites, however, little attention has been paid to social networks, blogs, forums, voice, and instant messaging (IM).

Dou et al. [23] presented the following categories for phishing detection studies: Visual similarity, page content-based, URL-based, blacklist-based, hybrid, and others. They also provided the most used feature categories as follows: Page content (e.g., page rank, non-matching links, page style), blacklist (e.g., blacklist and whitelist), visual similarity (e.g., color feature, block-level similarity, coordinate features, text pieces, and style), URL (e.g., IP address properties, length of the URL, use of Hypertext Transfer Protocol Secure (HTTPS) protocol, WHOIS properties), and others (e.g., web traffic and routing information).

Goel and Jain [29] focused on mobile phishing attacks and defense approaches and categorized attacks into the following categories: phishing through social engineering (e.g., Short Message Service (SMS), VoIP, website, e-mail), phishing through a mobile application (e.g., similarity attack, notification attack, floating attack, forwarding attack), phishing through malware (e.g., ransomware, botnet, key loggers), phishing through the online social network (e.g., malicious link, fake profile), phishing through content injection (e.g., cross-site scripting attack), phishing through the wireless medium (e.g., Bluetooth, Wi-Fi), and technical subterfuge (e.g., SSL based attack, session hijacking, DNS poisoning, through compromised webserver). Furthermore, they discussed the following anti-phishing solutions: Machine learning, optical character recognition, URL-based, mobile Quick Response Code (QR-code), text mining, and blacklist-based approaches.

Benavides et al. [11] investigated deep learning-based phishing detection studies and published a systematic literature review (SLR) paper. They reported that Deep Neural Networks (DNN) and Convolutional Neural Networks (CNN) are the most widely used approaches.

2.2 General phishing detection studies

There are several phishing detection models developed so far. In this subsection, some of these prominent studies are explained.

Prakash et al. [52] stated that attacks usually apply simple changes to URLs such as changing the top-level domain name. Their system called PhishNet reveals this observation by using two components. In the first component, five heuristics enumerate simple combinations of known phishing sites to discover phishing URLs. The second component consists of a matching algorithm that divides a URL into multiple components and matches individual entries in the blacklist. [75] presented a new framework for content-based phishing detection using a Bayesian approach. Their model takes into account textual and visual content to measure the similarity between the protected web page and suspicious web pages. In this context, they introduced a text classifier, an image classifier, and an algorithm that combines the results obtained from the classifiers. The prominent feature of their system is the use of the Bayesian model to predict the matching threshold.

Lee and Kim [44] proposed a suspicious URL detection system called WARNINGBIRD for Twitter. The system analyzes the correlations of URL redirect chains extracted from several tweets. The proposed methods to discover associated URL redirect chains using frequently shared URLs and identify suspicions. According to their results, the classifier detected suspicious URLs accurately and efficiently.

da Silva et al. [18] implemented a phishing prediction model based on a number of features. The purpose of the proposed model is to evaluate the static features. Static aspects refer to items such as keywords and patterns through the phishing URL. In the study, in addition to the quantitative data, a qualitative analysis of parameters that do not determine aspects such as relationships and similarities between attributes was performed.

Li et al. [46] proposed a combination of linear/nonlinear domain conversion methods to represent the core problem more clearly and to improve the performance of classifiers in identifying malicious URLs. For linear transformation, they implemented the singular value decomposition algorithm to obtain a perpendicular space and linear programming to solve an optimal distance measure as a two-step distance measure learning approach. In nonlinear transformation, they proposed the Nyström method for the kernel approach.

Zhu et al. [76] proposed a neural network model based on decision trees and optimum feature selection. They developed an incremental selection approach to remove duplicate points from public datasets using the traditional K-medoids clustering algorithm. An optimal feature selection algorithm was designed to eliminate the negative and unhelpful features.

Tan et al. [64] proposed anti-phishing approaches based on graph theory. The first stage of the proposed technique involves removing hyperlinks from the web page under review and bringing in relevant local web pages. [71] presented a new approach to phishing detection based on an inverted matrix online sequential over-learning machine that takes into account three types of features to characterize a website. They used the Sherman Morrison Woodbury equation to reduce matrix inversion. They introduced the online queue extreme learning machine to update the training model.

2.3 Deep learning-based studies

Recently, several deep learning-based phishing detection models have been developed. Some of these prominent studies are explained in this subsection.

Yi et al. [72] mainly focused on implementing a deep learning framework to detect phishing websites. The study first designed two types of features for web phishing: Original features and interactivity features. These features are used in a detection model based on Deep Belief Networks (DBN). DBN-based detection model provided promising results during the tests using real IP streams.

Wei et al. [67] proposed a lightweight deep learning algorithm to detect malicious URLs and enable a real-time and energy-saving phishing detection system. They showed that the proposed method can run in real-time on an energy-saving embedded single board computer.

Er and Ravi [56] discussed a new framework approach using the, a content-based approach to detecting phishing web sites (CANTINA) approach (DMLCA) and the software defined network (SDN)-based prevention system against phishing attacks. They reported that the SDN-based approach improves network security effectively.

Rao et al. [55] proposed a mobile application called PhishDump to categorize legitimate and phishing websites on mobile devices. PhishDump works with multiple models using the long short-term memory (LSTM) and support vector machine (SVM) classifier. Because PhishDump focuses on extracting attributes from URLs, it has several advantages over other studies, such as fast computation and language independence.

Subasi and Kremic [62] provided an intelligent phishing website detection framework. They used different machine learning models to classify websites as legitimate or phishing. They proposed learners (Adaptive Boosting (AdaBoost) and Multiboost) that can improve anti-phishing and work against attacks.

De La Torre Parra et al. [20] proposed a cloud-based distributed deep learning framework for phishing and Botnet attack detection. The model consists of two basic security mechanisms working collaboratively: (1) Distributed Convolutional Neural Network (DCNN) model to detect phishing and application layer Distributed Denial of Service (DDoS) attacks and (2) A cloud-based temporary long short-term memory (LSTM) network model hosted on the backend to detect botnet attacks and feed CNN to detect phishing attacks.

Wei et al. [68] applied CNN algorithms. Unlike previous studies where traffic statistics or web content were analyzed, only URL text was analyzed. As such, the method worked faster compared to the other models. [4] proposed a deep learning-based model that uses character-level CNN and website URLs for phishing detection.

Adebowale et al. [1] focused on the design of a deep learning-based phishing detection solution that leverages the universal resource finder and website content such as images, text, and frames. To create a mixed classification model, CNN and LSTM algorithms were used. [61] proposed a deep learning model to determine the legitimacy of a website. URL heuristics and third-party service-based features were used to train deep learning models. They minimized the number of features to achieve high accuracy and reduced the dependency on third-party services.

In Table 1, five prominent detection studies from the following four dimensions presented: Dataset, method, performance evaluation parameters, and accuracy metric. These studies have been selected with respect to the recently developing deep learning methodologies.

3 Methodology

This section explains the proposed hybrid deep learning model used in this study.

3.1 Our hybrid deep learning model

In this study, both traditional machine learning methods (i.e., k-Nearest Neighbors (kNN) and tree-based methods) and deep learning algorithms (i.e., RNN and CNN-based methods) [25, 58] have been applied. During the experiments, hand-crafted NLP features were used for traditional machine learning methods and the DNN network.

Considering algorithms using only NLP features, there are several studies using machine learning and deep learning algorithms. An example is [23] that extracts NLP-based features from the URL and implements machine learning models such as the Regression and Support Vector Machine (SVM) algorithms.

In hybrid models combining two different feature sets, a CNN-based model can be used instead of the RNN-based model used for character embedding features. However, a CNN-based model has high memory requirements and could not expose long-distance dependent features [74].

LSTM and DNN algorithms can solve more complex problems compared to the shallow learning algorithms (i.e.,

Table 1 Summary of the related work

	Subasi and Kremic (2020)	Parra et al. (2020)	Aljofey et al. (2020)	Wei et al. (2019)	Our Approach (2021)
Dataset	UCI Machine Learning Repository: Phishing Websites Dataset	Detection of IoT botnet attacks N_IoT	Alexa, openphish, spamhaus.org, techhelplist.com etc.	Alexa, hphosts, Joewein, malwaredomains, and phishtank	Ebbu2017, PhishTank, Marchal2014
Method	AdaBoost + SVM and Multiboosting	RNN-LSTM, DNN	Character level CNN	DNN	DNN+BiLSTM
Evaluation metrics	Accuracy, F1-Score and ROC Curves	Precision, Recall, F1-Score, Accuracy	Precision, Recall, F1-Score, Accuracy	Accuracy, Execution Time	Accuracy, F1-Score, AUC
Accuracy	97.61%	94.30%	95.02%	86.63%	99.21%

traditional machine Learning algorithms). Furthermore, LSTM network can store past information for a long time, however, Recurrent Neural Networks (RNN) are unable to do this task for long periods. LSTMs have internal state, they are aware of the temporal structure in the inputs, and they can model parallel input series separately [15]. There are several limitations of Multi-Layer Perceptron (MLP) algorithms (i.e., stateless, unaware of temporal structure, messy scaling, fixed size inputs, fixed size outputs) [15]. As such, we aimed to combine the power of LSTM and DNN algorithms in a single model and presented how to perform this integration effectively.

Bi-LSTM includes two LSTMs. One of them receives the input in forward direction, and the other one processes in backward direction. For the current prediction, bi-LSTM can use both left and right context; therefore, it provides better performance mostly in NLP tasks that include a lot of information in the right context. However, compared to the traditional LSTM algorithms, they are more complex and as such, they need more computational power. There are also other LSTM architectures proposed in literature [15] such as vanilla LSTM (i.e., simple LSTM), stacked LSTM, CNN-LSTM, encoder–decoder LSTM, and generative LSTM. Generative LSTM is mostly used to generate new sequences and therefore, it is not feasible to use it in this context. Stacked LSTM uses several LSTM layers stacked on top of another one. In CNN-LSTM model, CNN learns the features and LSTM is used for prediction. In encoder–decoder LSTM, one LSTM network encodes input sequences and the other one decodes the encoding. The advantage of adopting bi-LSTM in our experiments is that it can use both right and left context effectively and therefore, we achieved superior performance than the other models used during our experiments.

The novel hybrid architecture proposed in this study consists of the combination of two different neural networks. Alternatively, these two networks can be separated from each other and training can be performed separately. In this way, two different models are obtained. Finally, the weighted average of the results of these two models can be calculated for classification. However, such a procedure will increase training and prediction times. In addition, since the training of the two networks will be separate, some deep features may not be exposed. This leads to a decrease in accuracy.

For methods containing the RNN and CNN, character embedding-based features were used as input. There are some significant features in hand-crafted NLP features that are not possible to extract with character embedding technique. On the other hand, it has been observed that RNN-based methods established with the help of character embedding-based features can detect deep and hidden connections between characters. Therefore, new hybrid approaches have been proposed by combining these two different approaches and evaluated on public datasets. First, the DNN and LSTM algorithms were integrated for a novel phishing detection model. While NLP features are included in the system used by DNN, character embedding is given to the LSTM algorithm as input. As a second hybrid model, bidirectional LSTM (BiLSTM) was used instead of LSTM and it was observed that this model provided better performance than the previous model. These structures are shown in Fig. 1.

3.2 Time complexity

In order to calculate the time complexity of the proposed models, the time complexity of the DNN and LSTM-based sections of the model should be calculated separately. For the DNN section, the time complexity is equal to the sum of the number of parameters for each layer because the time is dominated by the matrix multiplications for multi-layer perceptron (MLP) layers. As such, the time complexity for the DNN section is $O(4p_1)$, where 4 is the number of layers and p_1 is the average number of parameters per layer, which depends on the input and output

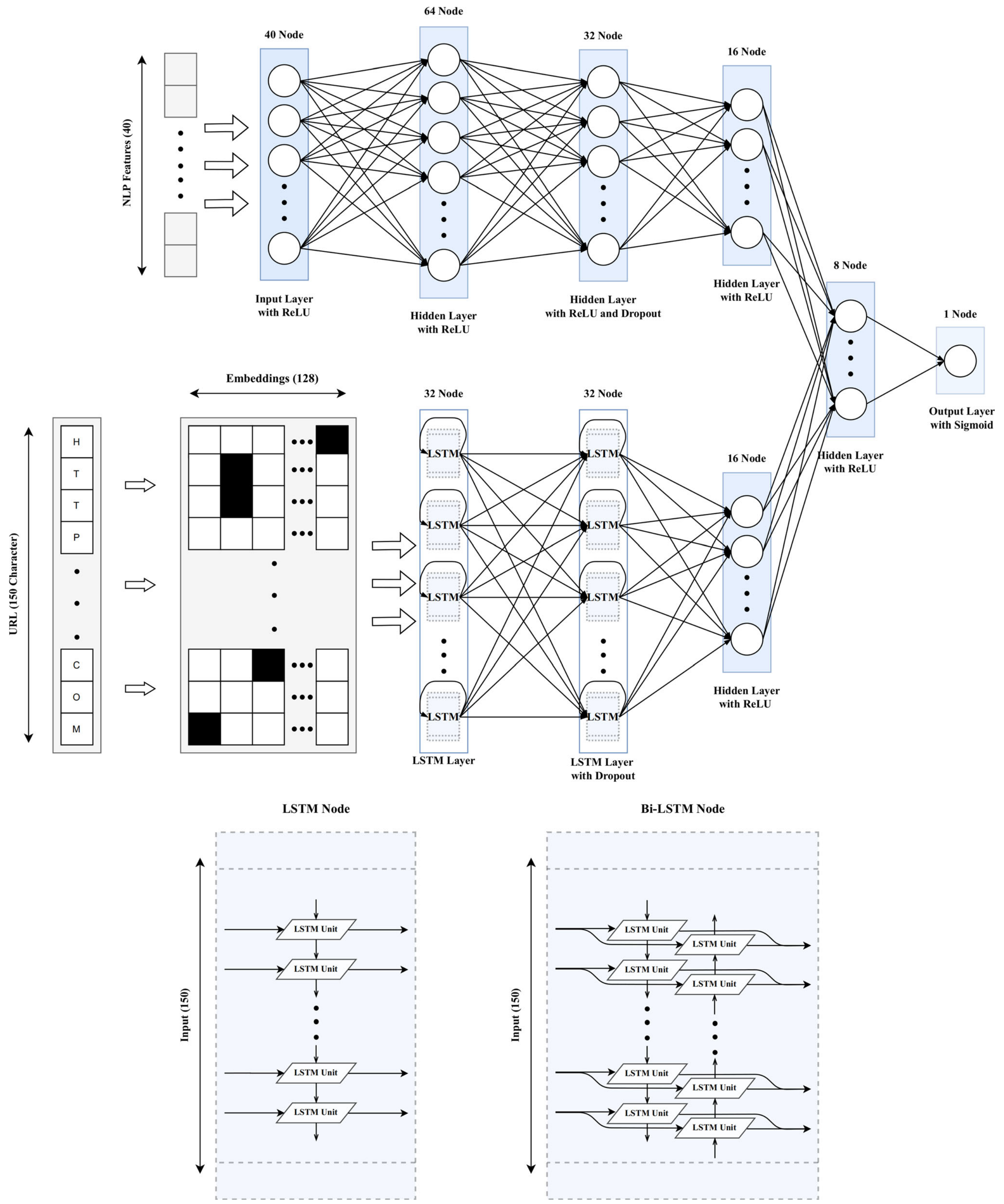

Fig. 1 Architecture of the Proposed Models

values of each layer. The time complexity for each layer in LSTM is $O(1)$ per weight because LSTM is local in space and time [32]. Therefore, the time complexity for the LSTM section is $O(w + p_2)$, where w is the total number of all weights in LSTM layers and p_2 is the number of parameters in the last layer of the LSTM section. For the model in which BiLSTM is used instead of LSTM, the time complexity is $O(2w + p_2)$ instead of $O(w + p_2)$ because calculations are made in two different directions in BiLSTM. Due to the structure of the hybrid model, when two separate sections are combined, two more MLP layers are used to get the final output value. The time complexity of this end part is $O(2p_3)$, where p_3 is the average number of parameters in these layers. Finally, the time complexity of the LSTM-based proposed hybrid model is $O(w + 4p_1 + p_2 + 2p_3)$, which is the sum of the time complexity of all the parts. Though this combination of models brings an additional cost in terms of required time, the benefit is considered to be beyond this additional cost.

4 Case study

This section discusses the datasets, features, performance evaluation metrics, machine learning algorithms, deep learning algorithms, and experimental results.

4.1 Datasets

Two datasets were used in this research. One of them is the Ebbu2017 dataset[1]. In addition to this dataset, a secondary dataset was built from several Internet resources for the experiments. The Ebbu2017 dataset contains 36,400 legitimate URLs and 37,175 phishing URLs. Therefore, it is a large dataset to perform experiments. The second one is a unique dataset containing 26,000 URLs. Half of the URLs are phishing URLs selected from the site named Phishtank[2] and the other half (i.e., 13,000) of the dataset consists of the legitimate URLs from [48]. Proposed models have been tested on these two datasets.

There are different datasets used in previous phishing URL classification studies such as UCI dataset[3]. There are also some web services that share data for researchers[4]. We aimed to select relatively recent datasets for developing our models and therefore, two dataset were selected. One of the datasets is named Ebbu2017, which is the same dataset used in a previous study [57]. As such, we were able to make our comparisons with the other models easily. The other dataset is created from URLs obtained from different sources and therefore, it is unique for this work. In future work, researchers can build new datasets as we did and evaluate the performance of the proposed model in new datasets. We did not wish to include old datasets and aimed to perform experiments in new datasets.

[1] https://github.com/ebubekirbbr/pdd/tree/master/input

[2] https://www.phishtank.com/

[3] https://archive.ics.uci.edu/ml/datasets/Website+Phishing

[4] https://phishstats.info/.

4.2 Features

The main goal of phishing attacks is to deceive the user by giving the user an impression of a legitimate site. The attacker can access valuable user information such as the password, username, and credit card number. Therefore, the attacker aims to make the phishing URL as similar to the legitimate site's URL as possible [76]. In this study, two types of features have been extracted to use in models.

4.2.1 Natural language processing (NLP) features

The attackers follow the tried-and-true patterns and experienced eyes can detect these patterns. With the help of NLP, distinctive features can be extracted from URLs. It is possible to obtain many different hand-crafted NLP features using different techniques. In this study, 40 features that were previously extracted by [57] were used to compare the results with their study. These features are shown in Table 2.

4.2.2 Character embedding

To compare different approaches for the phishing classification problem, character embedding has been used in RNN-based methods [8] in addition to traditional machine learning-based methods. Unlike other methods, RNN-based methods can learn the representation from the sequence of characters in the URL, and therefore, they do not need manual feature extraction.

There are other studies in which the character embedding technique is used and applied in a different way by using different parameter sets. For example, the data obtained by character embedding in [34] were converted into a 2D Image form that can be used with CNN. Additionally, there are some works that make feature extraction by word embedding instead of character embedding. The biggest limitation of the word embedding technique, which is likely to increase success if used with character embeddings, is that it needs very high memory to work. The reason for this is that there are many different languages and unique words [41].

4.3 Evaluation metrics

There are two types of classes in phishing detection, which are legitimate and phishing URLs. Therefore, the phishing URL detection problem is a binary classification task. In order to perform this classification, models designed to

Table 2 Natural language processing (NLP) features [57]

NLP Features	
Feature	Explanation
Raw Word Count	The number of words obtained after parsing the URL by special characters
Brand Check for Domain	Is domain of the analyzed URL in the brand name list
Average Word Length	The average length of the words in the raw word list
Longest Word Length	The length of the longest word in the raw word list
Shortest Word Length	The length of the shortest word in the raw word list
Standard Deviation	Standard deviation of word lengths in the raw word list
Adjacent Word Count	Number of adjacent words processed in the WDM module
Average Adjacent Word Length	The average length of the detected adjacent words
Separated Word Count	The number of words obtained as a result of decomposing adjacent words
Keyword Count	The number of keywords in the URL
Brand Name Count	The number of the brand name in the URL
Similar Keyword Count	The number of words in the URL that is similar to a keyword
Similar Brand Name Count	The number of words in the URL that is similar to a brand name
Random Word Count	The number of words in the URL, which is created with random characters
Target Brand Name Count	The number of target brand name in the URL
Target Keyword Count	The number of target keyword in the URL
Other Words Count	The number of words that are not in the brand name and keyword lists but are in the English dictionary (e.g., computer, pencil, notebook etc …)
Digit Count (3)	The number of digits in the URL. Calculation of numbers is calculated separately for domain, subdomain and file path
Subdomain Count	The Number of subdomains in URL
Random Domain	Is the registered domain created with random characters
Length (3)	Length is calculated separately for the domain, subdomain and path
Known TLD	[“com”, “org”, “net”, “de”, “edu”, “gov”, etc.] are the most widely used TLDs worldwide. Is the registered TLD known one
www, com (2)	The expression of “www” and “com” in domain or subdomain is a common occurrence for malicious URLs
Puny Code	Puny Code is a standard that allows the browser to decode certain special characters in the address field. Attackers may use Puny Code to avoid detecting malicious URLs
Special Character (8)	Within the URL, the components are separated from each other by dots. However, an attacker could create a malicious URL using some special characters {‘-‘, ‘.’, ‘/’, ‘@’, ‘?’, ‘&’, ‘=’, ‘_’}
Consecutive Character Repeat	Attackers can make small changes in brand names or keywords to deceive users. These slight changes can be in the form of using the same character more than once
Alexa Check (2)	Alexa is the name of a service that places frequently used websites in a certain order according to their popularity. Is the domain in Alexa Top one million list

solve the problem should apply binary classification algorithms.

Test data are used to measure the performance of the binary classification models, and the trained model is expected to predict the class of URLs in the test data. As a result of these predictions, each URL is categorized into four categories (i.e., True Positive, True Negative, False Positive, and False Negative). Every correctly classified phishing URL is represented as True Positive. Each legitimate URL that is correctly classified is considered True Negative. Another category is False Positive (FP). This indicates the incorrectly predicted legitimate URLs. Finally, False Negative (FN) represents the incorrectly predicted phishing URLs. These four cases form the confusion matrix. Accuracy, Area under ROC Curve (AUC), and F1-Score values are calculated based on the confusion matrix. Eqs. 1–5 show how to calculate these metrics.

In addition to the evaluation metrics applied in this study, the log loss, precision and recall can be used separately. Each of these metrics has its own advantages and disadvantages. As such, researchers prefer applying several metrics during their experiments. We selected the evaluation metrics in a way that the comparison with the previous studies is possible and also, the widely adopted metrics are utilized. To compare with previous studies, AUC, ACC and F1 Score were found to be suitable in this study. Other researchers can use different metrics, however, we do not expect any dramatic change for the final outcome of this study.

$$TruePositiveRate(Recall) = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \tag{1}$$

$$FalsePositiveRate = \frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}} \tag{2}$$

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} \tag{3}$$

$$Accuracy = \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}} \tag{4}$$

$$F1 - Score = 2 \times \frac{\mathrm{Precision} \times \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}} \tag{5}$$

4.4 Machine learning algorithms

Traditional machine learning-based methods have been used with NLP features because they need manual feature extraction [57] [2]. In this study, machine learning methods were built using the scikit-learn machine learning framework using Python programming language because this platform is flexible, robust, and easy to use. The following traditional machine learning algorithms were investigated in this research.

- *Naive Bayes:* The Naive Bayes Classifier is the simplest form of Bayesian network but it can achieve high accuracy with a kernel density estimation implementation [69].
- *k-Nearest Neighbors (k-NN):* k-NN is a non-parametric method that can be used for both classification and regression tasks [5].
- *Adaboost:* The AdaBoost classifier is a meta-estimator classifier. It fits multiple copies of the same classifier on the same dataset to achieve high accuracy on the dataset [27].
- *Decision Tree (DT):* DT is a non-parametric supervised learning method that can be used for both regression and classification like k-NNs. The main goal is to fit a model that can learn decision rules from dataset features and predict the target values [53].
- *Ridge Regression (RR):* RR is a linear classifier that uses linear least squares with L2 regularization [33].
- *Lasso:* Lasso is a linear classifier that uses L1 prior as a regularizer [38].
- *Light Gradient Boosting Machine (LightGBM):* The LightGBM classifier is a gradient boosting classifier from LightGBM framework, which uses tree-based learning algorithms [36].
- *XGBoost:* The XGBoost is an optimized distributed gradient boosting library. It has a gradient boosting-based classifier [16].
- *Random Forest (RF):* RF is a meta-estimator that runs a set of decision tree classifiers on many sub-parts of the dataset to achieve high accuracy [13].

In this study, several machine learning techniques were tested with NLP features. Some of these are kNN, Naive Bayes, Random Forest, XGBoost, and Ridge Regression. These algorithms cover all generally accepted and known machine learning algorithm types. Since there are many machine learning algorithms available, it is not possible to test them all. The SVM, which is not used in this study, is an example.

4.5 Deep learning algorithms

Deep learning [42], which has attracted great attention within the last decade, is a sub-branch of machine learning. Especially, recent developments in computing power and increasing data storage volumes have made a great contribution to the applicability of deep learning methods. In this way, deep learning-based models have provided state-of-the-art results for many different problems on large datasets. Researchers achieved the highest results in image processing [28], natural language processing [73], and machine translation [70]) tasks. Deep learning algorithms were also used for the task of phishing URL classification and promising results were achieved [8].

The following deep learning-based algorithms have been investigated in this research:

- *Deep Neural Network (DNN):* DNN-based classifiers, sometimes called MLP classifiers [51] [39], include at least two hidden layers and look very similar to the traditional multi-layer perceptrons. They consist of a layered network structure and each layer has a certain number of neurons (i.e., nodes). Node numbers and activation functions in the output layer are customized for the classification problem. It contains hidden layers in addition to the input and output layers and it can extract complex features [19]. The DNN used in this study contains two hidden layers and uses NLP attributes. This structure is shown in Fig. 2.

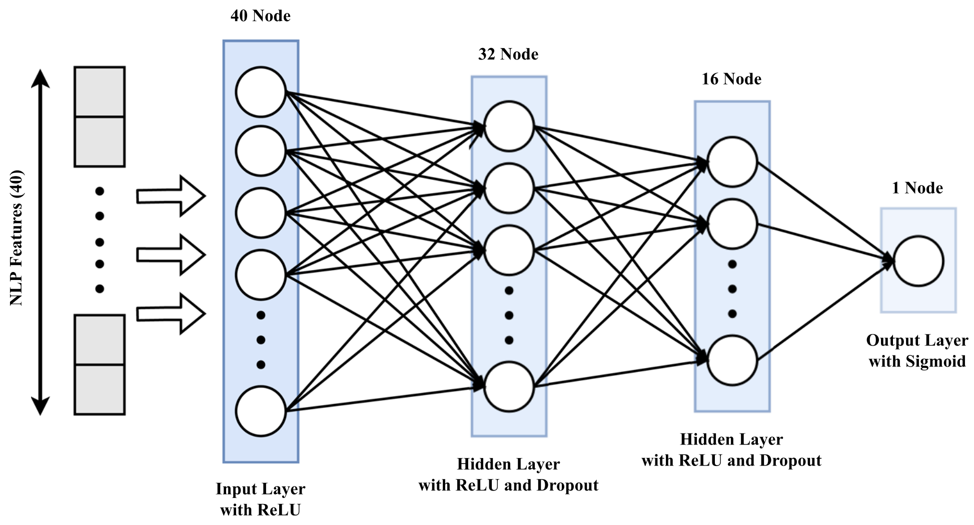
40 Node
32 Node
16 Node
1 Node
NLP Features (40)
Input Layer with ReLU
Hidden Layer with ReLU and Dropout
Hidden Layer with ReLU and Dropout
Output Layer with Sigmoid

Fig. 2 Architecture of the DNN model

- *Convolutional Neural Network (CNN):* CNN [43] is an algorithm that uses convolution layers in addition to the fully connected layers in different layers of the network. Its main goal is to extract features by applying convolution to the data. It passes these extracted features to the next layers and classifies the data this way. They have been mostly used in image processing [47] tasks. In addition, CNN can also be used on one-dimensional data [31]. In this study, the CNN algorithm was used on one-dimensional character embedding vectors. In addition to the convolutional and fully connected layers, pooling layers, dropout layers, and batch normalization layers can also be used in CNN models.
- *Recurrent Neural Networks (RNN):* RNN is a type of deep learning algorithm developed for time-series data [59]. RNN has directional connections between its inner nodes. With the help of these connections, it is able to calculate the next time step by using the information in the previous time step. As such, it can process sequential data. RNNs are widely used in tasks such as natural language processing [22] and speech to text tasks [66].
- *Long Short-Term Memory (LSTM):* Long-term short-term memory (LSTM) is a variation of the RNN [32] algorithm. The main problem with standard RNNs is that they cannot find meaningful connections easily between data pieces that have 10 or more time steps. LSTM is able to determine the importance of information and, if necessary, it can keep it for a very long time steps. It can establish a relationship even between data with more than 1000 time steps between them. Eqs. 6–11 are mathematical representations of the LSTM architecture. Every LSTM node is feed with h_{t-1} (output of the previous node), x_t (input), c_{t-1} (cell state of previous node). The cell state contains useful hidden long-term information. The data is passed from three different gates in LSTM, which are i_t (input gate), f_t (forget gate) and o_t (output gate). With the help of these gates, the LSTM node can keep or forget the old cell state and calculate the next outputs. σ represents the sigmoid function and $\odot$ denotes the element-wise multiplication. W and b are weight and bias values.

$$i_t = \sigma(x_t W_{xi} + h_{t-1} W_{hi} + b_i) \tag{6}$$

$$f_t = \sigma\left(x_t W_{xf} + h_{t-1} W_{hf} + b_f\right) \tag{7}$$

$$o_t = \sigma(x_t W_{xo} + h_{t-1} W_{ho} + b_o) \tag{8}$$

$$\tilde{c_t} = \tanh(x_t W_{xc} + h_{t-1} W_{hc} + b_c) \tag{9}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c_t} \tag{10}$$

$$h_t = o_t \odot \tanh(c_t) \tag{11}$$

- *Bi-directional LSTM (BiLSTM):* The BiLSTM algorithm has two LSTM cells instead of one for a single time step. One of these cells connects forward while the other connects backward. The outputs of these cells that take the same input are then combined. Therefore, it is able to find both forward and backward connections between time steps [30].

4.6 Experimental results

In order to evaluate the proposed approach, a widely used dataset, known as the Ebbu2017 dataset[5], was preferred. As the second dataset, a balanced dataset was built using the Phishtank Phishing dataset[6] and legitimate [48] URLs. After the datasets were obtained, the algorithms mentioned in the previous subsection were used for modeling. The

[5] https://github.com/ebubekirbbr/pdd/tree/master/input.

[6] https://www.phishtank.com/.

processing power required for training is provided by Google Colab, which is a free to use platform. Graphics processing unit (GPU) support was not needed because of the preferred models and datasets.

In the following subsections, first results related to the machine learning-based models are presented. In the next subsection, the performance of deep learning-based models is provided.

4.6.1 Experimental results based on machine learning algorithms

Machine learning algorithms were first evaluated. During the experiments, boosting algorithms were tested with the help of machine learning frameworks such as LightGBM as well as some frequently used machine learning algorithms. All experiments were implemented using Python language and additional frameworks developed for Python such as XGBoost, and scikit-learn machine learning library. NLP features were used as input in all designed models.

10-fold cross-validation results on the Ebbu2017 Phishing dataset and Phishtank dataset are presented in Tables 3 and 4. As seen in these tables, among the machine learning methods using only NLP features, the boosting classifier that is set up with the LightGBM algorithm provided the best results. LightGBM algorithm was more successful (i.e., 98.19% accuracy on Ebbu2017 dataset and 93.92% accuracy on Phishtank dataset) than the Random Forest classifier that presented the best result in previous studies. LightGBM is followed by the RandomForest algorithm, achieving a 98.09% accuracy, 0.9807 AUC value, and 0.9803 F1-Score on the Ebbu2017 dataset and a 93.30% accuracy, 0.9331 AUC value, and 0.9334 F1-Score on the PhishTank dataset. On the other hand, Naive Bayes was the worst classification model with an accuracy rate of 67.06% and 73.69% in both the Ebbu2017 dataset and the PhishTank dataset, respectively. The reason is that Naive Bayes assumes that the features are independent.

4.6.2 Cross-validation results of deep learning-based models

After the machine learning algorithms were investigated, deep learning algorithms and the proposed models based on these algorithms were analyzed. The Google Colab environment was used for the experiments. The Pytorch library was preferred for the DNN implementation. The Keras library was selected to implement the proposed methods and other RNN-based methods. While only hand-crafted NLP features were used as input for the model established with the DNN algorithm, character embedding-based features were used as input for the RNN-based models. For the proposed hybrid models, both NLP features and character embedding-based features were used as input, and different hyperparameters were investigated to obtain the optimal classifier model.

Table 3 Cross-validation results of the machine learning algorithms on Ebbu2017 phishing dataset

Algorithm	Accuracy	AUC	F1-Score
Naive Bayes	67.06%	0.6716	0.7476
kNN ($k = 3$)	93.80%	0.9377	0.9358
Adaboost	95.36%	0.9534	0.9521
Decision Tree	96.87%	0.9685	0.9678
Ridge regression	91.24%	0.9110	0.9043
LASSO	89.97%	0.8997	0.8994
LightGBM	**98.19%**	**0.9817**	**0.9813**
XGBoost	97.75%	0.9774	0.9769
Random Forest	98.09%	0.9807	0.9803

Bold values indicate the best overall result for the corresponding algorithm

In addition to the Keras library used in the study, it is possible to build a similar hybrid architecture with Pytorch and Tensorflow as well. Keras was preferred by the authors because it is easier to use, has syntactic simplicity and several APIs for implementation, and different libraries do not affect the experiment results.

Table 5 shows the search space of the hyperparameters with the best hyperparameter values. Adam was chosen as the optimizer function. ReLU $R(x) = max(0, x)$ was used as the activation function of the hidden layers because ReLU can break linearity between layers and prevents non-saturation of gradients [49], and the sigmoid function was applied for the output layer. The dropout rate was set to 0.3 for all dropout layers. During the training, every fold was trained for 40 epochs with 128 batch sizes. All remaining parameters are left by default.

According to our experimental results, the adam optimizer provided better performance than the other optimizers in terms of speed and accuracy parameters. There are also other optimizers that can be investigated in future work such as AdaBelief [77]. Adagrad [24] and Adamax [12] are also feasible alternatives. Future work can address these different optimizers that have not been evaluated yet. It is also possible for researchers to develop new optimization algorithms to improve the overall performance. However, our objective is not to develop a new optimizer, and therefore, we applied the available optimization algorithms in this study. In addition, experiments on activation functions show that the relu was more successful than the other activation functions. Last but not the least, for the epoch number, there was no increase in accuracy at values of 40 and above. As a future work, researchers can aim to

Table 4 Cross-validation results of the machine learning algorithms on phishtank dataset

Algorithm	Accuracy	AUC	F1-Score
Naive Bayes	73.69%	0.7280	0.7909
kNN ($k = 3$)	88.03%	0.8801	0.8841
Adaboost	89.38%	0.8934	0.8980
Decision Tree	90.23%	0.9026	0.8994
Ridge regression	85.30%	0.8521	0.8611
LASSO	73.26%	0.7327	0.7342
LightGBM	**93.92%**	**0.9391**	**0.9415**
XGBoost	92.61%	0.9259	0.9269
Random Forest	93.30%	0.9331	0.9334

Bold values indicate the best overall result for the corresponding algorithm

apply different algorithms and optimize the hyper parameters better than this study.

In addition to the hyperparameters used in Table 5, some of the other parameters in the models were changed during the experiments and the results were evaluated. These experiments were made to improve the overall performance during the experiments. Furthermore, some parameters other than the hyperparameters shown in Table 5 were used with their default values or the values used in previous studies. Further research can address the optimization of all parameters in the proposed model, we focused on the hyperparameters specified in Table 5

In order to test the statistical significance of the results, the Wilcoxon Signed Rank Test [26] is applied using a 0.05 significance level. Instead of Wilcoxon Signed Rank Test, t-test can also be used as a suitable alternative. However, as stated in [21], t test is conceptually inappropriate and statistically unsafe. Wilcoxon signed rank test is recommended over t test. Therefore, we applied the Wilcoxon signed rank test during our experiments to test the statistical significance of the results.

According to the 10-fold cross-validation results shown in Tables 6 and 7, the Accuracy, F1-Score and AUC values of the proposed models utilizing the properties of deep-level features are significantly larger than the baseline methods.

The hybrid model that integrates DNN and BiLSTM algorithms provided 98.79% accuracy, 0.9878 AUC, and

Table 6 Cross-validation results of the deep learning algorithms on Ebbu2017 phishing dataset

Algorithm	Accuracy	AUC	F1-Score
DNN	96.43%	0.9644	0.9627
CNN	93.25%	0.9326	0.9339
RNN	97.17%	0.9718	0.9720
LSTM	98.24%	0.9826	0.9828
BiLSTM	97.58%	0.9759	0.9761
DNN+LSTM	98.62%	0.9864	0.9865
DNN+BiLSTM	**98.79%**	**0.9878**	**0.9881**

Bold values indicate the best overall result for the corresponding algorithm

0.9881 F1-score on the Ebbu2017 phishing dataset when cross-validation was used for the evaluation. On the Phishtank dataset, the DNN and BiLSTM algorithm-based model provided 99.21% accuracy, 0.9934 AUC, and 0.9941 F1-score. The DNN-BiLSTM model is followed by the DNN–LSTM hybrid model with a 98.62% accuracy in the Ebbu2017 dataset and a 98.98% accuracy in the PhishTank dataset. The model that provided the worst performance on the Ebbu2017 dataset was the CNN-based classifier. The CNN-based classification model achieved a 93.25% accuracy, 0.9326 AUC value, and 0.9339 F1-Score. On the other hand, the worst classifier for the PhishTank dataset was the DNN-based classifier, achieving 91.13% accuracy.

5 Discussion

Phishing attacks are one of the major cyber-crimes threatening internet users today. Especially, with the increasing use of the internet, the detection of phishing attacks is becoming more and more important. Therefore, many methods have been proposed to detect phishing attacks so far. Blacklist-based methods that are proposed by [52] are the simplest ones among all other methods, however, they need a database that needs to be constantly updated. Machine learning-based methods, which provide better performance compared to the blacklist-based methods, are more complex because they require a feature extraction step that must be carried out beforehand.

Table 5 Hyper parameter search space and best hyper parameters

Hyper parameter	Search space	Value
Optimizer	adam, adadelta, rmsprop, sgd	adam
Activation functions (Hidden layers)	relu, tanh, elu	relu
Dropout rate	0.1–0.5	0.3
Epoch	10, 20, 40, 60	40
Batch size	16, 32, 64, 128	128

Table 7 Cross-validation results of the deep learning algorithms on phishtank dataset

Algorithm	Accuracy	AUC	F1-Score
DNN	91.13%	0.9125	0.9139
CNN	93.33%	0.9332	0.9349
RNN	97.22%	0.9720	0.9730
LSTM	98.23%	0.9822	0.9827
BiLSTM	98.27%	0.9826	0.9831
DNN+LSTM	98.98%	0.9901	0.9910
DNN+BiLSTM	**99.21%**	**0.9934**	**0.9941**

Bold values indicate the best overall result for the corresponding algorithm

In this study, it was observed that boosting-based machine learning classifiers provided better results than other machine learning methods. As Nielsen [50] stated, boosting-based classifiers are highly adaptive methods and carefully take the bias-variance trade-off into account in every aspect of the learning process. As such, they provide better results than the other traditional machine learning algorithms.

In addition to the machine learning algorithms, the deep learning algorithms were also investigated in this study. Although traditional machine learning methods can provide acceptable performance in some cases, their success is highly dependent on the quality of the extracted features (i.e., feature engineering). In contrast, deep learning algorithms discover features automatically and identify deep connections that are not easily recognized by domain experts. Some phishing detection models in literature preferred the use of very complicated hand-crafted features. However, some other approaches focused on the use of automatically discovered features. Due to no consensus on the feature sets of the prediction models, many models resulted in complicated architectures. In this study, to avoid this complexity, the power of these two strategies were combined and a novel prediction model utilizing the required features were presented.

Manually extracted features from URLs compress indeed important information and reduce the data size. As such, machine learning-based methods can learn high-level connections in URLs. However, some important connections between characters are lost during this reduction process. For this reason, recurrent neural network-based models are able to discover different deep connections when they use character embedding-based features.

The proposed hybrid architecture utilizes the advantages of hand-crafted features and character embedding-based features, which build novel phishing detection models. The novel architecture is able to extract all the information from both characters and hand-crafted features. Since the DNN and LSTM-based parts join toward the end and create a single model, back-propagation during the training is based on the same error value. With the help of this architecture, the DNN and the LSTM-based parts are optimized in harmony.

In this study, two datasets were used, however, additional experiments can be performed on other phishing detection datasets as well. Although there might be some changes in the performance of the proposed models, acceptable performance was expected on the other datasets. In addition, the phishing methods used by attackers are changing very rapidly. Therefore, the proposed models might need to be adapted for new kinds of phishing types.

To improve the performance of the proposed models, different optimization algorithms such as Adam have been investigated; however, there are also other parameters that have been used with their default values in our models. Therefore, as a future work, researchers can also consider optimizing the parameters used by their default parameters.

There are different LSTM models suggested in literature such as simple LSTM, bi-LSTM, stacked LSTM, CNN-LSTM, encoder–decoder LSTM, and generative LSTM. In this study, we evaluated the performance of bi-LSTM algorithm and demonstrated its effectiveness in our hybrid model. There is also possibility that other researchers can investigate the other LSTM-based algorithms in a hybrid model, we preferred the bi-LSTM algorithm due to its superior performance compared to the simple LSTM algorithm. We also discussed why we did not prefer the other LSTM-based algorithms in the relevant section. Bi-LSTM algorithm requires more computational power compared to the simple LSTM algorithm, however, our main objective is to achieve better performance.

For the implementation of the models, we used Keras Library because it easier to use, has syntactic simplicity and several APIs for implementation. The same models can be implemented in different deep learning frameworks such as PyTorch. We do not expect much change for the evaluation of our experiments if the same implementation of the algorithms is preferred. Nowadays, most of the deep learning researchers prefer Keras platform for implementation; however, it is also possible to implement these algorithms in different platforms.

We built a new dataset in this study and also, used a recent dataset for our experiments. We plan to create new datasets using the same approach and therefore, we will be able to continue our experiments. Building a new dataset was another challenge and we were able to use the dataset easily during our experiments. Different researchers can perform similar experiments in several datasets. The performance might be slightly different in different datasets, however, we expect acceptable results even if we run our models on new datasets.

We assume that the dataset used from literature is precise, complete, and has no noisy instances. However, as in many cases, there might be a few noisy instances that can affect the performance of our models. In this study, we did not apply noise detection techniques to remove the noisy instances from the datasets, however, as a future work researchers can also investigate the effect of noisy instances during the experiments

In addition, we assume that the manual NLP features used in this study are extracted correctly in previous studies. We used these features as-is and did not investigate whether the previous researchers created these features correctly. If there is any minor problem during the extraction of these NLP features, this might have affected the performance of our models.

For benchmarking purposes, we used different traditional machine learning algorithms. We assume that the implementation of these algorithms in Keras platform includes feasible parameter values; therefore, during the experiments we used the default parameter values for traditional machine learning algorithms. Future work can investigate the effect of different parameter values on the proposed models.

6 Conclusion

This paper proposed two hybrid deep learning-based models, namely DNN–LSTM and DNN-BiLSTM models, that utilize both hand-crafted NLP features and character embedding-based features for the detection of phishing URLs. The proposed models are able to integrate the properties of high-level NLP features while finding deep connections between characters. Two datasets were used to evaluate the performance of the proposed models and comparison was performed with several machine learning and deep learning-based models. One of these datasets is the same dataset used in a previous study by other researchers. In addition, not only RNN-based models but also a CNN-based model and traditional machine learning models were evaluated during the experiments.

It was concluded that the proposed models provide better performance than the other models and the results are very promising. In the first dataset, which was used in a previous study, the proposed DNN-BiLSTM model is able to reach 98.79% accuracy. In the second dataset created specifically for this study, again the DNN-BiLSTM model achieves the highest accuracy rate with 99.21%. The DNN-BiLSTM model was followed by the DNN–LSTM model in both datasets in terms of the accuracy metric. The most important reason for this success is that the proposed models use different features at the same time because of the hybrid network structure. In addition, the Bi-directional LSTM model provided better results than the traditional LSTM model. The reason for this result is that the BiLSTM structure, unlike the traditional LSTM structure, can find the connections between characters both in the forward and backward direction.

The most important reason why hybrid architecture models provide better results is that they can use both NLP features and character embedding features simultaneously compared to the other models. This combination of features lets the hybrid model discover more deep features in URLs. Later, these relations were used to classify URLs. In addition, the reason why bi-LSTM is more successful than LSTM is that the BiLSTM, unlike the LSTM, can find the connections between characters both in the forward and backward direction. For the current prediction, bi-LSTM can use both left and right context and therefore, it provides better performance mostly in NLP tasks that include a lot of information in the right context. This lets the BiLSTM utilize more deep features than LSTM. There are also some studies, which state that the bi-LSTM algorithm provides better performance than the LSTM algorithm [60]

The following limitations exist for this study:

- We built a new dataset in this study and also, used a recent dataset for our experiments. We plan to create new datasets using the same approach and therefore, we will be able to continue our experiments. Building a new dataset was another challenge, and we were able to use the dataset easily during our experiments. Different researchers can perform similar experiments in several datasets. The performance might be slightly different in different datasets, however, we expect acceptable results even if we run our models on new datasets.
- We assume that the dataset used from literature is precise, complete, and has no noisy instances. However, as in many cases, there might be a few noisy instances that can affect the performance of our models.

In order to further improve the success of the results, the features obtained by word embedding can be added to the existing hybrid model. This means that the hybrid model processes three different feature sets simultaneously. Also, the recently popular self-attention technique can be used with character embedding attributes. Studies in other fields have shown that the RNN-based model with self-attention technique is better than standard RNN-based models (Jing 2019).

Declarations

Conflict of interest The authors have no conflicts of interest to declare that are relevant to the content of this article.

References

1. Adebowale MA, Lwin KT, Hossain MA (2020) Intelligent phishing detection scheme using deep learning algorithms. J Enterprise Inf Manag
2. Akinyelu AA (2019) Machine learning and nature inspired based phishing detection: a literature survey. Int J Artif Intell Tools 28(05):1930002
3. Aleroud A, Zhou L (2017) Phishing environments, techniques, and countermeasures: a survey. Comput Secur 68:160–196
4. Aljofey A, Jiang Q, Qu Q, Huang M, Niyigena JP (2020) An effective phishing detection model based on character level convolutional neural network from url. Electronics 9:9
5. Altman NS (1992) An introduction to kernel and nearestneighbor nonparametric regression. Am Stat 46(3):175–185
6. APWG GA, Manning R (2020) Apwg phishing reports
7. Ardabili SF, Najafi B, Shamshirband S, Bidgoli BM, Deo RC, Chau KW (2018) Computational intelligence approach for modeling hydrogen production: a review. Eng Appl Comput Fluid Mech 12(1):438–458
8. Bahnsen AC, Bohorquez EC, Villegas S, Vargas J, González FA (2017) Classifying phishing urls using recurrent neural networks. In: 2017 APWG symposium on electronic crime research (eCrime), pp 1–8
9. Banan A, Nasiri A, Taheri-Garavand A (2020) Deep learning-based appearance features extraction for automated carp species identification. Aquacult Eng 89:102053
10. Basit A, Zafar M, Liu X, Javed AR, Jalil Z, Kifayat K (2020) A comprehensive survey of ai-enabled phishing attacks detection techniques. Telecommun Syst, pp 1–16
11. Benavides E, Fuertes W, Sanchez S, Sanchez M (2020) Classification of phishing attack solutions by employing deep learning techniques: A systematic literature review. In: Developments and advances in defense and security. Springer, pp 51–64
12. Bengio Y, LeCun Y (2015) Adam: a method for stochastic optimization. In: 3rd International conference on learning representations, ICLR'15, San Diego, CA, USA
13. Breiman L (2001) Random forests. Mach Learn 45(1):5–32
14. Brownlee J (2017) Deep learning for natural language processing: develop deep learning models for your natural language problems. Mach Learn Mastery
15. Brownlee J (2017) Long short-term memory networks with python develop sequence prediction models with deep learning. Mach Learn Mastery
16. Chen T, Guestrin C (2016) Xgboost. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining
17. Chiew KL, Yong KSC, Tan CL (2018) A survey of phishing attacks: their types, vectors and technical approaches. Expert Syst Appl 106:1–20
18. da Silva CMR, Feitosa EL, Garcia VC (2020) Heuristic-based strategy for phishing prediction: a survey of url-based approach. Comput Secur 88:101613
19. Dargan S, Kumar M, Ayyagari MR, Kumar G (2020) A survey of deep learning and its applications: a new paradigm to machine learning. Arch Comput Methods Eng 27(4):1071–1092
20. De La Torre Parra G, Rad P, Choo KKR, Beebe N (2020) Detecting internet of things attacks using distributed deep learning. J Netw Comput Appl 163:102662
21. Dem J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7(1):1–30
22. Deng H, Zhang L, Shu X (2018) Feature memory-based deep recurrent neural network for language modeling. Appl Soft Comput 68:432–446
23. Dou Z, Khalil I, Khreishah A, Al-Fuqaha A, Guizani M (2017) Systematization of knowledge (sok): a systematic review of software-based web phishing detection. IEEE Commun Surv Tutor 19(4):2797–2819
24. Duchi J, Hazan E, Singer Y (2011) Adaptive subgradient methods for online learning and stochastic optimization. J Mach Learn Res 12(61):2121–2159
25. Fan Y, Xu K, Wu H, Zheng Y, Tao B (2020) Spatiotemporal modeling for nonlinear distributed thermal processes based on KL decomposition, MLP and LSTM network. IEEE Access 8:25111–25121
26. Fix E, Hodges Jr J (1955) Significance probabilities of the wilcoxon test. In: The Annals of Mathematical Statistics, pp 301–312
27. Freund Y, Schapire RE (1997) A decision-theoretic generalization of on-line learning and an application to boosting. J Comput Syst Sci 55(1):119–139
28. Garcia-Garcia A, Orts-Escolano S, Oprea S, Villena-Martinez V, Martinez-Gonzalez P, Garcia-Rodriguez J (2018) A survey on deep learning techniques for image and video semantic segmentation. Appl Soft Comput 70:41–65
29. Goel D, Jain AK (2018) Mobile phishing attacks and defence mechanisms: state of art and open research challenges. Comput Secur 73:519–544
30. Graves A, Schmidhuber J (2005) Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Netw 18(5–6):602–610
31. Hao S, Ge FX, Li Y, Jiang J (2020) Multisensor bearing fault diagnosis based on one-dimensional convolutional long short-term memory networks. Measurement 159:107802
32. Hochreiter S, Schmidhuber J (1997) Long short-term memory. *Neural Comput.* **9**(8):1735Ã¢â'¬â€œ1780
33. Hoerl AE, Kennard RW (2000) Ridge regression: biased estimation for nonorthogonal problems. Technometrics 42(1):80–86
34. Huang Y, Yang Q, Qin J, Wen W (2019) Phishing URL Detection via CNN and Attention-Based Hierarchical RNN. In: 18th IEEE International conference on trust, security and privacy in computing and communications/13th IEEE international conference on big data science and engineering, (TrustCom/BigDataSE)'19
35. Jing R (2019) A self-attention based LSTM network for text classification. J Phys Conf Ser 1207:012008
36. Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, Ye Q, Liu TY (2017) Lightgbm: a highly efficient gradient boosting decision tree. In: Proceedings of the 31st international conference on neural information processing systems, NIPS'17, pp. 3149–Ã'â'¬Ã'Å"3157, Red Hook, NY, USA
37. Khonji M, Iraqi Y, Jones A (2013) Phishing detection: a literature survey. IEEE Commun Surv Tutor 15(4):2091–2121
38. Kim S, Koh K, Lustig M, Boyd S, Gorinevsky D (2007) An interior-point method for large-scale '1-regularized least squares. IEEE J Sel Top Signal Process 1(4):606–617
39. Kussul N, Lavreniuk M, Skakun S, Shelestov A (2017) Deep learning classification of land cover and crop types using remote sensing data. IEEE Geosci Remote Sens Lett 14(5):778–782
40. Lastdrager EE (2014) Achieving a consensual definition of phishing based on a systematic review of the literature. Crime Sci 3(1):9
41. Le H, Pham Q, Sahoo D, Hoi SCH (2018) URLNet: learning a URL representation with deep learning for malicious URL detection
42. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444
43. Lecun Y, Jackel L, Bottou L, Brunot A, Cortes C, Denker J, Drucker H, Guyon I, Muller U, Sackinger E, Simard P, Vapnik V (1995) Comparison of learning algorithms for handwritten digit recognition

44. Lee S, Kim J (2013) Warningbird: a near real-time detection system for suspicious urls in twitter stream. IEEE Trans Depend Secure Comput 10(3):183–195
45. Li Q, Cheng M, Wang J, Sun B (2020a) Lstm based phishing detection for big email data. In: IEEE transactions on big data
46. Li T, Kou G, Peng Y (2020b) Improving malicious urls detection via feature engineering: Linear and nonlinear space transformation methods. Inf Syst 91:101494
47. Li Z, Yang W, Peng S, Liu F (2020c) A survey of convolutional neural networks: analysis, applications, and prospects
48. Marchal S, Francois J, State R, Engel T (2014) Phishstorm: detecting phishing with streaming analytics. IEEE Trans Netw Serv Manag 11(4):458–471
49. Rectified Linear Units Improve Restricted Boltzmann Machines Vinod Nair. In: Proceedings of ICML 27:807–814
50. Nielsen D (2016) Tree boosting with xgboost-why does xgboost win" every" machine learning competition? Master's thesis, NTNU
51. Pal SK, Mitra S (1992) Multilayer perceptron, fuzzy sets, and classification. IEEE Trans Neural Netw 3(5):683–697
52. Prakash P, Kumar M, Kompella RR, Gupta M (2010) Predictive blacklisting to detect phishing attacks. In: 2010 Proceedings IEEE INFOCOM, pp 1–5
53. Quinlan JR (1986) Induction of decision trees. Mach Learn 1(1):81–106
54. Ramzan Z, Wuest C, (2007) Phishing attacks: analyzing trends in 2006. In CEAS, Citeseer
55. Rao RS, Vaishnavi T, Pais AR (2019) Phishdump: a multimodel ensemble based technique for the detection of phishing sites in mobile devices. Pervasive Mobile Comput 60:101084
56. Er S, Ravi R (2020) A performance analysis of software defined network based prevention on phishing attack in cyberspace using a deep machine learning with cantina approach (dmlca). Comput Commun 153:375–381
57. Sahingoz OK, Buber E, Demir O, Diri B (2019) Machine learning based phishing detection from urls. Expert Syst Appl 117:345–357
58. Shamshirband S, Rabczuk T, Chau K-W (2019) A survey of deep learning techniques: application in wind and solar energy resources. IEEE Access 7:164650–164666
59. Sherstinsky A (2020) Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. Phys D Nonlinear Phenom 404:132306
60. Siami-Namini S, Tavakoli N, Siami-Namin A (2019) The Performance of LSTM and BiLSTM in Forecasting Time Series. In: 2019 IEEE international conference on big data, (Big Data)'19
61. Somesha M, Pais AR, Rao RS, Rathour VS (2020) Efficient deep learning techniques for the detection of phishing websites. Sādhanā 45(1):165
62. Subasi A, Kremic E (2020) Comparison of adaboost with multiboosting for phishing website detection. Proc Comput Sci 168:272–278
63. Sullins LL (2006) Phishing for a solution: domestic and international approaches to decreasing online identity theft. Emory Int'l L. Rev. 20:397
64. Tan CL, Chiew KL, Yong KS, Sze SN, Abdullah J, Sebastian Y (2020) A graph-theoretic approach for the detection of phishing webpages. Comput Secur 95:101793
65. Varshney G, Misra M, Atrey PK (2016) A survey and classification of web phishing detection schemes. Secur Commun Netw 9(18):6266–6284
66. Wang WJ, Liao YF, Chen SH (2002) Rnn-based prosodic modeling for mandarin speech and its application to speechto- text conversion. Speech Commun 36(3):247–265
67. Wei B, Hamad RA, Yang L, He X, Wang H, Gao B, Woo WL (2019) A deep-learning-driven light-weight phishing detection sensor. Sensors (Basel, Switzerland) 19(19):4258
68. Wei W, Ke Q, Nowak J, Korytkowski M, Scherer R (2020) Accurate and fast url phishing detector: a convolutional neural network approach. Comput Netw 178:107275
69. Wu X, Kumar V, Ross Quinlan J, Ghosh J, Yang Q, Motoda H, McLachlan GJ, Ng A, Liu B, Yu PS, Zhou ZH, Steinbach M, Hand DJ, Steinberg D (2008) Top 10 algorithms in data mining. Knowl Inf Syst 14(1):1–37
70. Wu Y, Schuster M, Chen Z, Le QV, Norouzi M, Macherey W, Krikun M, Cao Y, Gao Q, Macherey K, Klingner J, Shah A, Johnson M, Liu X, Lukasz Kaiser, Gouws S, Kato Y, Kudo T, Kazawa H, Stevens K, Kurian G, Patil N, Wang W, Young C, Smith J, Riesa J, Rudnick A, Vinyals O, Corrado G, Hughes M, Dean J (2016) Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*
71. Yang L, Zhang J, Wang X, Li Z, Li Z, He Y (2021) An improved elmbased and data preprocessing integrated approach for phishing detection considering comprehensive features. Expert Syst Appl 165:11113863
72. Yi P, Guan Y, Zou F, Yao Y, Wang W, Zhu T (2018) Web phishing detection using a deep learning framework. Wireless Commun Mobile Comput 2018:4678746
73. Young T, Hazarika D, Poria S, Cambria E (2018) Recent trends in deep learning based natural language processing [review article]. IEEE Comput Intell Mag 13(3):55–75
74. Zhang Q, Bu Y, Chen B, Zhang S, Lu X (2021) Research on phishing webpage detection technology based on CNN-BiLSTM algorithm. J Phys Conf Ser 1738:012131
75. Zhang H, Liu G, Chow TWS, Liu W (2011) Textual and visual content-based anti-phishing: a approach. IEEE Trans Neural Netw 22(10):1532–1546
76. Zhu E, Ju Y, Chen Z, Liu F, Fang X (2020) Dtof-ann: An artificial neural network phishing detection model based on decision tree and optimal features. Applied Soft Computing 95:106505
77. Zhuang J, Tang T, Ding Y, Tatikonda S, Dvornek N, Papademetris X, Duncan JS (2020) AdaBelief optimizer: adapting stepsizes by the belief in observed gradients. In: 34th conference on neural information processing systems, NeurIPS'20, Vancouver, Canada