IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 9, 2022, accepted March 30, 2022, date of publication April 18, 2022, date of current version April 27, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3168235

PDGAN: Phishing Detection With Generative Adversarial Networks

SAAD AL-AHMADI, (Senior Member, IEEE), AFRAH ALOTAIBI, AND OMAR ALSALEH
Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh 11543, Saudi Arabia

Corresponding author: Saad Al-Ahmadi (salahmadi@ksu.edu.sa)

This work was supported by a grant from the ''Research Center of College of Computer and Information Sciences'', Deanship of Scientific Research, King Saud University.

ABSTRACT Phishing is a harmful online attack that could lead to identity theft and financial damages. The demand for high-accuracy phishing detection tools has risen due to the increase of online electronic services and payment systems. Most phishing detection techniques depend on features related to webpage content, which necessitates crawling the webpage and relying on third-party services. Relying on features related to webpage content could not provide high detection accuracy and leads to high false detection rates. Recently, deep learning has become a popular approach for detecting phishing websites. However, limited attention has been given to the generative adversarial network (GAN). This paper proposes a phishing detection model called PDGAN that depends only on a website's uniform resource locator (URL) to achieve reliable performance. We use a long short-term memory network (LSTM) network as a generator of synthetic phishing URLs and a convolutional neural network (CNN) as a discriminator to decide whether the URLs are phishing or legitimate. We use a dataset containing nearly two million phishing and legitimate URLs obtained through PhishTank and DomCop. The experimental results show that the PDGAN achieves a detection accuracy of 97.58% and a precision of 98.02% without depending on third-party services and with greater accuracy than the state-of-the-art models.

INDEX TERMS Convolutional Neural Network (CNN), deep learning, generative adversarial network (GAN), long short-term memory network (LSTM), phishing website detection.

I. INTRODUCTION

Phishing is a common type of social engineering attacks that tricks users into revealing their confidential information and credentials, such as passwords and credit card information. With the variety of cybersecurity attacks, phishing has received particular attention due to its powerful effects on industries as well as individuals in terms of financial and personal data [1]. A report recently published by the Anti-Phishing Working Group (APWG) [2] detected 611,877 phishing sites in the first quarter of 2021—a notable increase compared to the 164,772 sites in the first quarter of 2020. Since the number of phishing attacks has increased, there is a strong need for an efficient approach to phishing detection. Accordingly, the victims can be warned when they are a target of a phishing campaign to avoid any potential loss of sensitive data.

The associate editor coordinating the review of this manuscript and approving it for publication was Rajeeb Dey.

Phishing detection models can be categorized into human-based and software-based approaches. Human-based models aim to enhance the knowledge of end-users and help them make good decisions when faced with a phishing website. In contrast, software-based models adopt different techniques to determine whether a website is phishing or legitimate without end-user interference. The latter model is generally divided into five approaches: blacklist/whitelist, visual similarity, heuristic, machine learning, and deep learning [3].

1.1 The blacklist/whitelist-based approach relies on a list of known phishing websites which contains much information such as known phishing URLs, IP addresses, and others. This list must be updated constantly [4].

2.2 The visual similarity-based approach compares the visual similarity of the phishing webpage and its corresponding legitimate webpage using different features. If the similarity is higher than the preset threshold, a webpage is considered phishing [5].

3.3 The heuristic-based approach depends on a phishing web page's characteristics, the similarity between phishing

webpages, or experts' prior knowledge. This approach extracts many features from phishing webpages and generalizes them into a collection of heuristics [6].

4.4 The machine learning approach considers phishing detection as a binary classification problem. It typically includes two steps: first, obtaining an appropriate feature representation from the URL, and second, using this representation to train machine learning models. The first step involves acquiring useful information about the URL, which can be represented as a vector for further use by machine learning models. These features are extracted manually from different sources, such as URLs, website traffic, search engines, DNS, etc. [1]. Therefore, the training data must contain many features related to legitimate and phishing website classes. The second step involves training classification algorithms, such as k-nearest neighbor (k-NN), decision tree (DT), random forest (RF), naïve Bayes (NB), and support vector machines (SVMs) [7].

A website is classified as phishing in a machine learning approach if the tested website results match the predefined feature set. The performance of this approach depends on the feature set, training data, and classification algorithm. Using machine learning algorithms can enable unseen URLs to be easily detected. However, some machine learning approaches require high computational power to compute and obtain features from different sources [7]. Several machine learning approaches were proposed in [1], [8], and [9].

5.5 With regards to deep learning, the detection approaches typically include three steps: first, designing a deep learning model; second, selecting the model inputs; and third, analyzing a set of features that are used to classify websites. In this approach, the selection of model inputs and the construction of the model will influence its effectiveness. For example, common models used for phishing website detection include CNNs and recurrent neural networks (RNNs).

The difference between traditional machine learning and deep learning approaches can be highlighted by feature engineering. The machine learning approach needs robust knowledge to characterize the original data into specific feature collections. In contrast, deep learning models do not require feature engineering since the model obtains features set directly from the data [3].

In this paper, we propose a phishing website detection approach PDGAN, which does not depend on webpage content but rather only on a webpage's URL. PDGAN uses a deep learning model, namely a GAN, whose adversarial process allows the model to learn different variations in phishing features and produce a final model that provides better detection results. The proposed approach is domain-independent and shows an efficient computational time with a detection rate of less than 0.5 ms per URL.

The key contributions are summarized as follows:

- Proposing a deep learning model, i.e., GAN, to detect phishing websites, relying on the website URL only.
- Combining the advantages of the LSTM and CNN in processing texts. At first, we use the LSTM to generate synthetic phishing URLs, and then we use CNN to determine whether the URL is phishing or legitimate.
- Using a large-scale dataset through DomCop and PhishTank websites, which contains nearly 2 million phishing and legitimate URLs. From the experimental results, the detection accuracy reached 97.58%, and the precision reached 98.02%.
- Comparing two models with the proposed PDGAN, and according to the experimental results, the PDGAN outperforms the state-of-the-art models.

The rest of this paper is structured as follows: Section II highlights different deep learning models for phishing detection presented in the literature. Section III introduces the basic idea of PDGAN. Section IV provides the design of the PDGAN approach in detail. Section V presents and discusses the experimental results. A conclusion is provided in section VI.

II. RELATED WORKS

Recently, various deep learning models have been used for detecting phishing websites. This section outlines some of these models.

Mohammad *et al.* [10] attempted to find an optimal solution by having the simplest model structure and avoiding network expansion. Their work attempted to increase the model's accuracy by updating the learning rate. First, the proposed model constructed a three-layer neural network, with just one neuron in the hidden layer. The hidden layer neurons were gradually increased in the training phase, but the features were manually extracted. Legitimate websites were obtained from the Yahoo! directory and starting point directory, while phishing websites were obtained from PhishTank and MillerSmiles. The results indicated that the model had good generalization for noise data and a high detection rate.

Bahnsen *et al.* [11] evaluated two approaches for URL phishing detection. The first approach combined a URL's lexical and statistical analysis with a RF classifier. The second approach was an LSTM network which directly learned a representation from the URL's character sequence instead of manually extracting the features. The phishing URLs were collected from PhishTank and the legitimate URLs from Common Crawl. The LSTM approach provided a higher accuracy rate and F1 score than the RF classifier.

Anand *et al.* [12] proposed a phishing URL detection method using GANs with an oversampling task in the data space. GANs were trained to learn the string pattern statistics of URLs in the minority class and generate synthetic URLs. Both the generator and discriminator were LSTM networks. They selected representative synthetic samples using

k-means clustering and Euclidean distance-based selection. The dataset was collected from the PhishTank and Common Crawl repositories.

Yi *et al.* [13] applied deep belief networks (DBNs) for phishing detection. The proposed system used two features for detection: original features and interaction features. The original features referred to the direct features of the URL, such as the domain age, while interacting features described interactions between websites. The model had two layers of restricted Boltzmann machines (RBMs) stacked layer by layer. The proposed model used an SVM as a binary classifier to classify the DBN features. The dataset was obtained through real IP flows from the ISP. As a result, the detection model achieved a high TP rate and a low FP rate.

Vinayakumar *et al.* [14] evaluated different deep learning models to detect phishing URLs, namely RNN, I-RNN, LSTM, CNN, and a hybrid network (CNN–LSTM). The phishing and legitimate URLs were trained at the character level by extracting features automatically. The legitimate URLs were collected from Alexa and the DMOZ directory, while the phishing URLs were collected from PhishTank and OpenPhish. The LSTM and CNN–LSTM networks performed well in distinguishing a URL as either legitimate or phishing compared with other adopted models.

Selvaganapathy *et al.* [15] proposed a model for malicious URL detection using stacked RBMs for feature selection with deep neural networks for classification. Malware URLs and advanced persistent threat URLs were collected from the malicious domain list, while spamming and phishing URLs were collected from the UCI Machine Learning Repository. Legitimate URLs were collected from the DMOZ directory. The model reduced the FP rate and improved detection accuracy compared with other adopted methods.

Shivangi *et al.* [16] proposed a tool that is deployed as an extension of the Google Chrome browser to provide the user with a safer browsing experience. The proposed tool analyzed URLs and classified them using two different deep learning models, namely artificial neural networks (ANNs) and LSTM networks. These models extracted the features from the URLs automatically, while other existing techniques required manual feature engineering, which is a computationally expensive and time-consuming task. The dataset was obtained through search engines, PhishTank, and the Twitter Streaming API. The LSTM network achieved higher accuracy than the ANN.

Peng *et al.* [17] proposed a malicious URL detection model that utilized a CNN–LSTM network to extract and filter textual features, statistical features, and WHOIS information. They aimed to identify the important role of key features in detecting malicious URLs. The phishing URLs were collected from PhishTank, while legitimate URLs were collected from popular websites. According to the results, the statistical features of the URL contributed most to detecting malicious URLs. However, deep neural network models had a better influence on detecting all features than individual ones. The proposed model achieved the highest accuracy compared with other adopted mechanisms.

Robic-Butez and Win [18] proposed a deep learning approach for phishing detection using a GAN, which consisted of two networks: a generator and a discriminator. The generator generates both legitimate features and synthetic phishing to train a discriminator, which determines whether a website is phishing or legitimate. The detection error obtained is used to improve the accuracy of the discriminator and generator networks. Both the generator and the discriminator were multilayer perceptron (MLP). The phishing and legitimate URLs were obtained from PhishTank and Amazon, respectively.

Zhang *et al.* [19] proposed a hybrid model to detect phishing websites with URL-based, abnormal-based, HTML-based, JavaScript-based, and domain-based features. The proposed model incorporated two models: an autoencoder (AE) and a CNN. The CNN was used to obtain the local feature combinations, while the AE was used to reconstruct features that explicitly enhanced the correlations among the features. Phishing URLs were collected from PhishTank, and legitimate URLs were collected from DMOZ. According to the results, the proposed model achieved the highest accuracy compared with other traditional classification algorithms.

Huang *et al.* [20] proposed a deep learning approach for phishing detection called PhishingNet that depends only on URLs. They used CNN modules to extract character-level spatial feature representations of URLs and employed attention-based hierarchical RNN modules to extract word-level temporal feature representations. They then merged these feature representations via a three-layer CNN to build accurate URL feature representations and used an MLP for classification. A large-scale dataset was built through PhishTank, OpenPhish, and Alexa. The proposed approach significantly outperformed state-of-the-art solutions, indicating the model's effectiveness.

Feng *et al.* [21] proposed a phishing website detection model based on URL features, HTML features, and third-party services. They used a stacked AE (SAE) with a Softmax classifier to detect phishing websites. The proposed model determined the optimal width of hidden layers by calculating the correlation coefficients between the weight matrices of the SAE. The legitimate webpages were obtained from Alexa, and the phishing webpages were obtained from PhishTank. The model achieved better performance than the other adopted algorithms. However, the features were manually extracted.

Wang *et al.* [3] proposed a deep learning approach for phishing detection called PDRCNN. The approach depended only on the website URL and combined two neural networks (a CNN and a bidirectional LSTM network). In this approach, the CNN extracted the local features while the bidirectional LSTM network extracted the global features. The model first encoded a URL into a two-dimensional tensor, then fed it into a designed model to classify the URL. The dataset was collected from Alexa and PhishTank containing legitimate and phishing URLs. The approach significantly outperformed state-of-the-art solutions, meaning that it could better

detect phishing URLs without relying on any third-party services.

Yang *et al.* [22] proposed a detection approach for URL phishing. The structure of their proposed approach was divided into three modules. The first module was a CNN combined with an LSTM network. The second module determined the multidimensional features based on three features and the URL features obtained from the first module. The third module was a dynamic category decision algorithm used for real-time detection. A large-scale dataset was built using PhishTank and DMOZ directories, which contained phishing and legitimate URLs. The CNN–LSTM with a multidimensional features approach had achieved high accuracy and a low FP rate.

Alalyan and Al-ahmadi [23] proposed a deep learning approach for phishing detection called PUCNN, which depended only on the website URL. They used a CNN to extract character-level feature representations of URLs. They proposed a large-scale dataset called MUPD that contained over two million URLs collected from PhishTank and DomCop. The proposed CNN achieved better accuracy than existing state-of-the-art models and outperformed various machine learning models based on commonly used URL features from the MUPD dataset.

As seen, most of these works increasingly used deep learning models such as CNNs and RNNs compared to the traditional classification algorithms. Moreover, they relied on third-party services in combination with their system. Also, few studies have used GAN model. So, to predefined challenges, PDGAN is proposed to detect phishing websites more efficiently. This work performs the detection task using a GAN deep learning model at a character-level feature representation. The model consists of LSTM and CNN networks and depends only on website URLs.

III. OVERVIEW OF PDGAN

A. PROBLEM DEFINITION

Networking and communication technologies have developed rapidly, subjected them to several cyberattacks. Phishing is a serious and spreading cyberattack that tricks users into disclosing sensitive personal information. It is considered a significant category of cyberattack since it is used to launch many attacks. This criticality reflects the need for efficient phishing detection techniques.

Most phishing detection techniques proposed in the past few years rely on features related to webpages, which requires crawling the content of webpages and relying on third-party services. In this paper, we propose a phishing detection approach that relies only on a website's URL rather than content-based features or third-party services.

The authors of [12] and [24] demonstrated that a URL-based approach achieved high accuracy in classifying unseen phishing URLs. In other words, a model using only features extracted from the inspection of URL strings performed similarly to content-based detection systems, thus enabling the costs and security risks associated with content-based detection systems to be discarded.

B. THE STRUCTURE OF PDGAN

The proposed PDGAN consists of an LSTM as a generator and a CNN as a discriminator. The generator's function is to produce synthetic phishing URLs similar to real ones and cannot be distinguished by the discriminator. The discriminator's function is to extract the intrinsic features in the URL to distinguish between legitimate and phishing URLs.

The PDGAN combines the advantages of LSTM networks and CNNs in processing text. First, the LSTM structure in the PDGAN generates synthetic phishing URLs whose characteristics resemble those of real phishing URLs. Next, the features are extracted from the URL string by the convolutional layer and the pooling layer with different sizes of convolution kernels. Fig. 1 shows the workflow of the proposed PDGAN model.

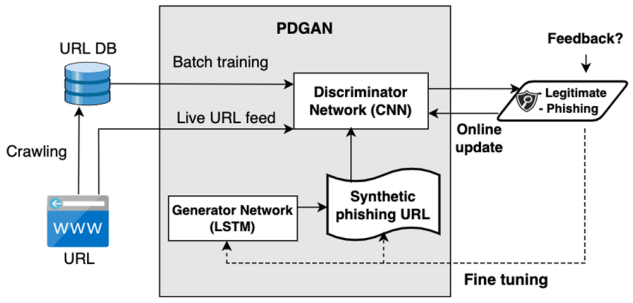

FIGURE 1. Workflow of the proposed PDGAN model.

C. SELECTION OF DEEP LEARNING MODEL

Typical deep learning models include LSTM, CNN, autoencoders, and DBN (deep belief networks). LSTM has a good performance at processing sequence and time series problems. Typically, an LSTM network remembers information for long periods— which is the key difference between LSTM networks and other neural networks. LSTM networks consist of different memory blocks called cells. These memory blocks are responsible for remembering things. The manipulations of this memory are performed through three main gates, called the forget, input, and output gates. These gates enable the LSTM network to keep and reuse relevant information within very long sequences of time series data [11]. Fig. 2 shows a single LSTM cell, where the calculation of each gate is as follows:

– Forget gate, f_t:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f), \quad (1)$$

– Input gate, i_t:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \quad (2)$$

– Cell state, C_t:

$$C_t = f_t \otimes C_{t-1} \oplus i_t \otimes tanh(W_c \cdot [h_{t-1}, x_t] + b_c), \quad (3)$$

– Output gate, o_t:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), . \quad (4)$$

where x_t is the input of the current layer, σ is the sigmoid function, h_{t-1} represents the hidden state of the t – 1 moment, b represents the bias of each gate. W_f, W_i, and W_o are the weight matrices for the connection.

The final step of the LSTM cell consists of calculating the output h_t at time t using the multiplicative operation $\otimes$ between the output gate layer and the tanh layer of the current cell state C_t.

$$h_t = o_t \otimes tanh(C_t) \quad (5)$$

As a result, the output, h_t, has passed through the network as a previous state for the next LSTM cell [25].

CNN is a particular type of deep learning network that has been widely adopted in several fields related to computer vision and natural language processing (NLP). A CNN's learning ability is largely a consequence of the use of many features in the extraction phase that can automatically learn data representations. A typical CNN architecture generally includes alternate layers of convolution and pooling followed by one or many fully connected layers [26].

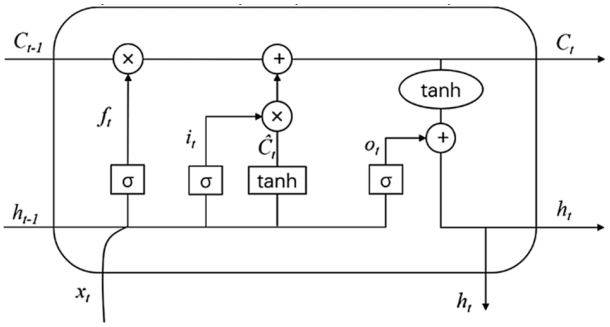

FIGURE 2. LSTM cell [25].

A convolutional layer is utilized for extracting features and consists of multiple convolutional kernels or filters that divide the input vectors into small blocks. Next, convolutional operations are applied to the input vectors with the chosen kernels to generate a series of feature maps. The pooling layer is used for reducing the dimensionality of the feature maps. The pooling layer has a two-fold purpose: accelerating the network operation and improving the performance of the entire convolutional network. A fully connected layer is a traditional neural network that performs the final classification task using the features extracted from the previous layers. Batch normalization and dropout techniques are used between CNN layers to avoid overfitting problems [27].

IV. PDGAN DESIGN

A. DATA PREPROCESSING

This step is based on the encoding process proposed in [25]. First, we suppose that the URL length is fixed to 255 characters, as HTTP standard protocol RFC2616 states there is a limit on URL length: ''Servers ought to be cautious about depending on URL lengths above 255 bytes because some older client or proxy implementations might not properly support these lengths.''[3]. Accordingly, if the URL length exceeds 255 characters, the first 255 characters are considered, and if the length is shorter than 255 characters, extra zeroes are inserted at the end.

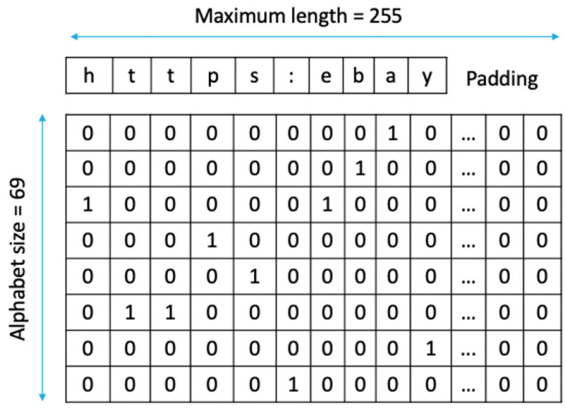

FIGURE 3. Data preprocessing.

Next, each URL character is encoded in a one-hot vector consisting of 0 and 1 since neural networks use a vector of numbers to perform any mathematical operation. The characters used include the 26 characters of the alphabet, 10 numerical digits, and the 33 special characters allowed in URLs (e.g., /, &, -, ?, and =). Finally, the encoded characters compose the tensor provided as input to the model. Fig. 3 shows the data preprocessing step.

B. CONVOLUTIONAL NEURAL NETWORK STRUCTURE: THE DISCRIMINATOR

After preprocessing, we obtain a dense representation of the URL characters. The PDGAN extracts the features of the phishing URL using a CNN to identify whether the URL is legitimate or phishing. Generally, a URL is one-dimensional; thus, we apply one-dimensional convolution, in which a filter slides in one direction.

The discriminator is nine layers deep, including six convolutional layers and three fully connected layers. The convolutional layers extract the local features of the phishing URLs. Each convolutional layer has a filter of length l; meaning filters are applied on l characters at a time where each character contains a vector of m elements. The one-dimensional convolutional layer then passes its output to a one-dimensional pooling layer. It uses the max operation to obtain the most significant features generated by the convolutional layers. Finally, the convolutional and pooling layers results are connected as a one-dimensional vector and fed to the three fully connected layers, with two dropouts between those three layers to avoid overfitting. Fig. 4 shows the architecture of the discriminator.

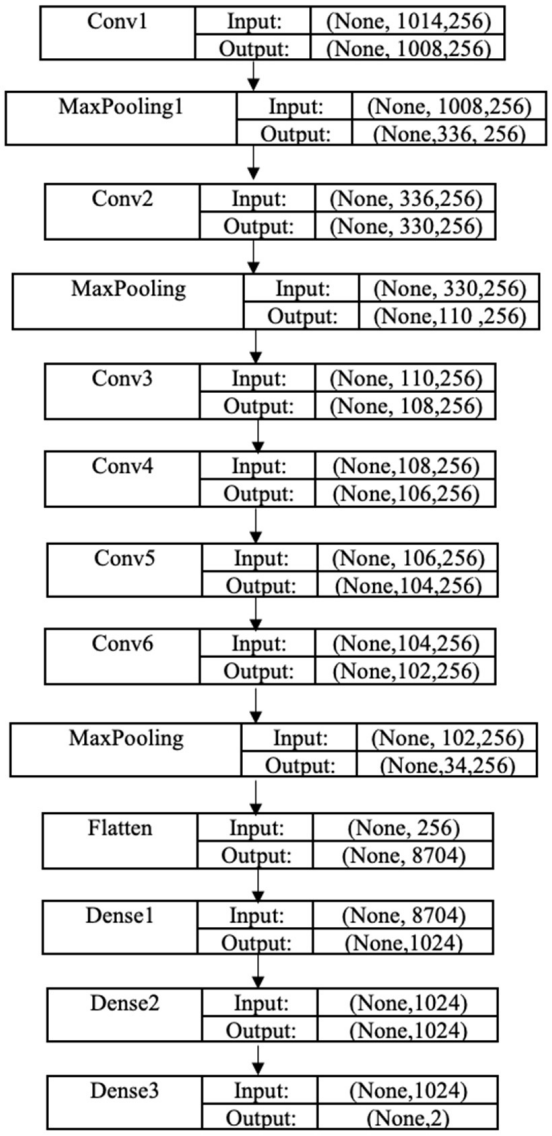

FIGURE 4. Architecture of discriminator.

C. RECURRENT STRUCTURE: THE GENERATOR

The LSTM network generates a sequence of characters. First, the embedding layer generates a representation in the form of a vector for each character that makes up the sequence. Next, each element of the sequence of the embedded characters is fed to the LSTM layer. Finally, the output of the LSTM network is passed through a dense layer to generate a URL. Fig. 5 shows the architecture of the generator.

D. MODEL TRAINING

The generator and discriminator are trained to create synthetic phishing URLs, and decide whether the URLs were phishing or legitimate, respectively. At each step, the generator and discriminator are trained separately. Thus these system components are working together to improve the overall performance.

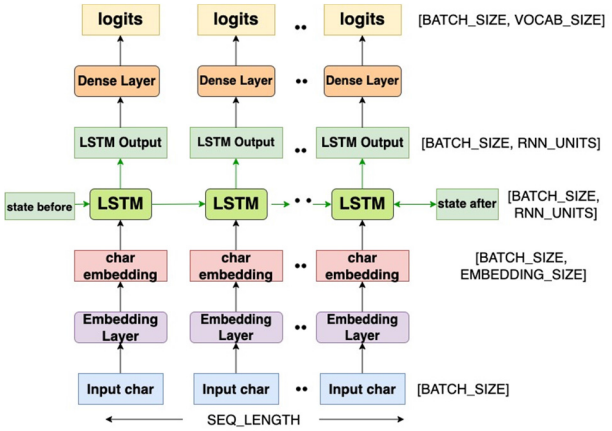

FIGURE 5. Architecture of generator.

We use binary cross-entropy as the generator and discriminator's loss function. The networks' parameters were tuned to reduce the loss. The loss function was optimized by continuously updating the weights in the network through an iterative process. We use the Adam optimizer, a common optimization strategy that reduces loss and leads the model to converge quickly.

In the Adam optimizer, the learning rate for each parameter is adjusted dynamically depending on the gradient's first-order and second-order moment estimates. We selected the Adam optimizer because the size of the learning step of each iteration has a specific range and does not lead to a large learning step when the gradient is large, and the value of the parameter is relatively stable. After testing different learning rates, we set the learning rate to 0.001. The model converges, and the training ends when the loss value is sufficiently reduced.

V. RESULTS AND DISCUSSION

In this section, we introduce our dataset optimization parameters and discuss the experimental results to show the effectiveness of the proposed model.

A. DATASET

In our experiments, we used the MUPD dataset [23], which contains 2,220,853 legitimate URLs and 2,353,933 phishing URLs. The source for phishing URLs was PhishTank, which was similarly used by most of the works we reviewed in the related works section. MUPD dataset only considered phishing websites which were verified as phishing on PhishTank. The source for the legitimate websites was the top 4 million domains list from DomCop. MUPD dataset has the index page (if it existed) for each of the top 4 million domains and a random internal URL. Furthermore, the MUPD dataset has been published, which eases the process of training and evaluating our proposed model.

After preprocessing, the final dataset contained 1,167,201 phishing URLs and 1,140,599 legitimate URLs. The following pre-processing steps were performed to generate the

published datasets: sampling to guarantee that the dataset was balanced, removing duplicate data, and splitting the dataset into three subsets (training, validation, and testing).

The collected datasets contained different URLs from the same host or many repeated URLs. For example, many pages from the same phishing website were frequently reported as phishing pages. Similarly, their collection process, which used top-level domains, may have resulted in repeated hosts for various reasons, such as HTTP redirects. Therefore, URLs with repeated hosts and duplicate URLs were removed from the proposed MUPD dataset. This step aimed to enhance the evaluation decision and prevent models from memorizing the host.

A balanced dataset is preferable in binary classification, particularly when an accuracy metric is used. Although the dataset was balanced before removing duplicate URLs, phishing URLs represented only one-third of the dataset when duplicate URLs were removed. A random sampling of 1,200,000 legitimate URLs was used to fix this issue. Through this step, a balanced dataset of 1,140,599 legitimate URLs and 1,167,201 phishing URLs was obtained. The dataset was split randomly into three subsets: 0.6 training, 0.2 validation, and 0.2 testing. Table 1 summarizes dataset size before and after preprocessing.

TABLE 1. Sizes of datasets used in our experiments.

Dataset	Phishing	Legitimate
MUPD dataset (before preprocessing)	2,353,933	2,220,853
MUPD dataset (after preprocessing)	1,167,201	1,140,599

B. EVALUATION INDICATORS

We use the following performance measurements to evaluate the proposed model and other models: accuracy, precision, recall, and F-measure.

The accuracy is the number of legitimate URLs correctly labeled as legitimate plus the number of phishing URLs correctly labeled as phishing over the total number of test set samples. The equation for calculating accuracy is given in Eq. (6).

$$accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{6}$$

The precision is the number of phishing URLs correctly labeled as phishing over the total number of URLs labeled as phishing. The equation for calculating precision is given in Eq. (7).

$$precision = \frac{TP}{TP + FP} \tag{7}$$

The recall (also known as TPR and sensitivity) is the number of phishing URLs correctly labeled as phishing over the total number of actual phishing URLs. The equation for calculating recall is given in Eq. (8).

$$recall = \frac{TP}{TP + FN} \tag{8}$$

TABLE 2. Effect of hidden layers on the generator.

Hidden layers	Loss
32	1.41
64	1.34
128	1.31
256	**1.28**

The F-measure is the weighted harmonic mean of the precision and recall rate. Methods with a higher F-measure are more effective. The equation for calculating F-measure is given in Eq. (9).

$$F = \frac{2 \times precision \times recall}{precision + recall} \tag{9}$$

Among them, true positives (TPs) indicate that the phishing URLs are correctly labeled as phishing URLs, false positives (FPs) indicate that the legitimate URLs are incorrectly labeled as phishing URLs, true negatives (TNs) indicate that the legitimate URLs are correctly labeled as legitimate URLs, and false negatives (FNs) indicate that the phishing URLs are incorrectly labeled as legitimate URLs.

C. PDGAN PARAMETERS SETUP

The tuning of hyperparameters in a neural network plays an essential role in the classification process. The CNN's convolution kernel size and the number of hidden units in the LSTM network are essential in estimating the loss and classification accuracy. We set the convolution kernel size in the range of 3 to 7 and selected the appropriate convolution kernel based on accuracy and loss functions. The number of hidden layers for the generator was selected from the set {32, 64, 128, 256}.

The setting of the epoch size and batch size is crucial: If the number of epochs is very small, PDGAN cannot achieve the highest accuracy, while if the number of epochs is very high, overfitting may occur. For the generator, we set the epoch to 80 and selected batch size from the set {32, 64, 128, 256}. For the discriminator, we set the epoch to 20 and selected batch size from the set {64, 128, 256}.

Several experiments were conducted; in each experiment, we evaluated the effects of various numbers of hidden layers within the generator using the validation set. Table 2 shows the loss with different numbers of hidden layers.

As shown in Table 2, as the number of layers in the generator increases from 32, the loss continuously decreases. Also, Table 3 shows the effect of batch size on the generator.

As we can see from Table 3 when the batch size increases from 32 to 64, the loss is decreased, but when the size increases from 64, the loss continuously increases. After setting the number of hidden layers to 256 and the batch size

TABLE 3. Effect of batch size on the generator.

Batch size	Loss
32	1.32
64	**1.28**
128	1.35
256	1.47

TABLE 4. Effect of convolution kernel size on discriminator.

Convolution kernel	Accuracy	Loss
7, 6, 5, 4, 3	96.34%	0.16
7, 7, 4, 4, 3	96.42%	0.12
7, 7, 3, 3, 3, 3	**97.56%**	**0.09**

TABLE 5. Effect of batch size on discriminator.

Batch size	Accuracy	Loss
64	96.28%	0.18
128	**97.56%**	**0.09**
256	96.22%	0.20

to 64, we evaluated the effect of the convolution kernel size, as shown in Table 4.

Table 4 shows that the convolution kernel size {7, 7, 3, 3, 3, 3} generated the best results. We then evaluated the effect of batch size on the discriminator, as shown in Table 5.

Table 5 shows that when the batch size is adjusted to 128, the discriminator has the least loss and the highest accuracy. After adjusting hyperparameters, the optimal values of the hyperparameters for the PDGAN model were as follows: the number of hidden units in the LSTM was 256; the size of the convolution kernel in the CNN was {7, 7, 3, 3, 3, 3}; the epoch size of the LSTM was 80; the epoch size of the CNN was 20; the batch size in the LSTM was 64, and the batch size in the CNN was 128. We use dropout to avoid overfitting. The dropout is assigned 0.2 value in the LSTM and 0.5 value in the CNN.

D. BASELINE MODELS

To verify the ability of the PDGAN model to determine phishing URLs, we compared the performance of the proposed PDGAN with the PUCNN [23] and PDRCNN models [3].

PUCNN is a phishing detection approach that depends only on the website URL. It used a CNN to extract character-level feature representations of URLs. PUCNN used a dataset named MUPD which contains 2,220,853 legitimate URLs and 2,353,933 phishing URLs. The source for phishing website URLs was PhishTank, while the source for the legitimate websites was the top 4 million domains list from DomCop. We used PUCNN as our main baseline model due to its various similarities to our PDGAN model. First, it relies on URLs only, making it similar to our scenario. It also used the same dataset for training, validation, and testing, which raises the confidence in the comparison results.

TABLE 6. Results of proposed PDGAN.

Dataset	Accuracy	Precision	Recall	F-measure
MUPD dataset	96.48%	96.09%	96.82%	96.43%
MUPD dataset + 5,000 synthetic phishing URLs	96.50%	95.83%	96.75%	96.25%
MUPD dataset + 10,000 synthetic phishing URLs	96.51%	96.89%	96.22%	96.55%
MUPD dataset + 50,000 synthetic phishing URLs	**97.58%**	**98.02%**	**97.27%**	**97.64%**

PDRCNN is a phishing website detection approach that relies only on the URL of the website. It encodes the information of an URL into a two-dimensional tensor and feeds the tensor into a deep learning neural network to classify the original URL. A bidirectional LSTM network is used to extract the global features of the constructed tensor, while a CNN is used to extract the local features. PDRCNN used a dataset containing nearly 500,000 URLs obtained through Alexa and PhishTank. We used a PDRCNN as a second baseline model because its workflow is also based on URLs and is similar to our discriminator model, as it used a character-level CNN.

Generated Phishing URLs

http://login.paypal.com-casepp-96616.11qnt.info/login.htm
http://clientform.ref13560903351.bbt.com.dllsoro.cn/clients/data/proc.jsp
http://businessbanking.53.com.session0690244.tenpost.cn/clientbase/form.asp
http://business-eb.client86825907-form.bbt.com.tenipp.cn/clients/form/b_form.jsp

Real Phishing URLs

http://login.paypal.com-casepp-845788.11qnt.info/login.htm
http://clientform.ref498124.bbt.com.posterst.cn/clients/data/proc.jsp
http://businessbanking.53.com.session149648933.orety.hk/clientbase/form.asp
http://business-eb.client912217-form.bbt.com.postidi.cn/clients/form/b_form.jsp

FIGURE 6. Synthetic and real phishing URLs.

E. REPRESENTATIVE URLS

We show some examples of real and synthetic phishing URLs in Fig. 6. The generated phishing URLs in the top list have similar characteristics to the real phishing URLs in the bottom list. We can see that our PDGAN model correctly captured the common structure of URLs, such as hostname, domains, etc. However, some details of the URLs still contain semantically incorrect information and are not entirely understood. This might result from using the dropout technique that prevents overfitting. However, the proposed PDGAN still successfully identified phishing URLs.

After completing the generator's training phase, it was also used to obtain different amounts of representative phishing URLs. We generated 10,000, 50,000, and 100,000 synthetic

TABLE 7. Confusion matrix.

	Predicted as phishing	Predicted as legitimate	Total
Phishing	236,732 (97.3%)	6,634 (2.7%)	243,366
Legitimate	4,771 (2.1%)	223,423 (97.9%)	228,194
Total	241,503	230,057	471,560

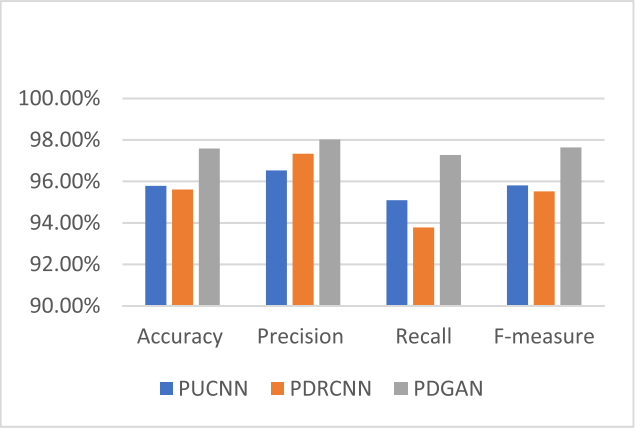

FIGURE 7. Results of proposed PDGAN and the baseline model.

phishing URLs. Each set was split into the proportions 0.8 training and 0.2 testing and added to the original training and testing sets. Table 6 shows all the performance measures of the proposed PDGAN for the original dataset and the three generated sets.

The PDGAN model showed significant results in all experiments, although it achieved the lowest accuracy with the original MUPD dataset. The proposed PDGAN model detected phishing websites with the highest accuracy, precision, recall, and F-measure scores on the MUPD dataset plus 50,000 synthetic URLs compared with the other experiments. This demonstrates how the generator model can enhance the classification results of the discriminator because the generator has explored other phishing URLs not learned by the discriminator.

F. COMPARISON OF DIFFERENT MODELS

To evaluate the performance of the PDGAN, we used the test dataset to compare the PDGAN approach and the two selected baseline models. Table 7 presents the results of the PDGAN on the original test dataset and 10,000 test synthetic phishing URLs.

As can be seen from the confusion matrix, 236,732 phishing URLs were correctly classified as phishing URLs, and 4,771 legitimate URLs were incorrectly classified as phishing URLs, for a FPR of only 2.1%. We mainly focused on the accuracy metric for comparisons, as accuracy was a popular metric through the literature review. However, we also provide the other performance metrics for the proposed PDGAN and the baseline models in Fig. 7.

In Fig. 7, we can see that the proposed PDGAN could detect phishing URLs with higher accuracy, precision, recall, and F-measure scores than the two baseline models. The precision value of the PDGAN model is around 98%, which indicates the superior performance of the proposed PDGAN, where the generator learns different variations in the phishing features to generate other URLs not learned by the discriminator. PDGAN achieved 97.56% accuracy, thus outperforming the other two baseline models.

According to the results, the proposed PDGAN is effective and has the highest accuracy among other compared systems. This can be interpreted from the ability of the generator to explore URLs other than those in the original dataset and the discriminator decidability to discover phishing URLs. The proposed PDGAN model can effectively integrate the advantages of both the LSTM and CNN models. In addition, the proposed PDGAN model can produce good results when considering only the URL in detecting phishing websites.

VI. CONCLUSION

In this work, we presented a new phishing detection model using GAN called PDGAN. Our model analyzes a URL and classifies the relevant webpage as phishing or legitimate. The proposed PDGAN model consists of a generator and a discriminator trained in adversarial processes. The generator is an LSTM model which generates synthetic phishing URLs, and the discriminator is a CNN model which decides whether a URL is phishing or legitimate.

PDGAN model does not depend on webpage content or third-party services; rather, it depends only on a website's URL to achieve a better phishing detection rate. Moreover, the adversarial process in the PDGAN model enhances the ability of the discriminator to distinguish phishing URLs by exploring other different phishing URLs that are not involved in the training dataset. Although some URL details still contain incorrect semantic information and are not fully understood, the PDGAN still successfully identified phishing URLs.

Several experiments were conducted on a large dataset containing two million legitimate and phishing URLs, split into training, validation, and testing datasets. PDGAN achieved 97.58% accuracy and 98.02% precision without depending on third-party services and greater accuracy than other compared models. These results demonstrate that the PDGAN model can detect phishing URLs with an enhanced classification result.

For future work, we intend to calculate the model's complexity to enrich the comparison between different models. We also decided to expand the scope of the proposed PDGAN to cover visual similarity–based approaches. In addition, we will analyze the effect of character-level similarity between various URL components to generate well representative synthetic phishing URLs.

ACKNOWLEDGMENT

The authors would like to thank the Research Center of College of Computer and Information Sciences, Deanship of Scientific Research, King Saud University for their grant and support.

REFERENCES

[1] A. K. Jain and B. B. Gupta, "Towards detection of phishing websites on client-side using machine learning based approach," *Telecommun. Syst.*, vol. 68, no. 4, pp. 687–700, Aug. 2018, doi: 10.1007/s11235-017-0414-0.
[2] (2021). APWG. *Anti Phishing Work Group*. Accessed: Aug. 30, 2021. [Online]. Available: https://apwg.org/trendsreports/
[3] W. Wang, F. Zhang, X. Luo, and S. Zhang, "PDRCNN: Precise phishing detection with recurrent convolutional neural networks," *Secur. Commun. Netw.*, vol. 2019, pp. 1–15, Oct. 2019, doi: 10.1155/2019/2595794.
[4] Y. Cao, W. Han, and Y. Le, "Anti-phishing based on automated individual white-list," in *Proc. 4th ACM Workshop Digit. Identity Manage. (DIM)*, 2008, pp. 51–59, doi: 10.1145/1456424.1456434.
[5] L. Wenyin, G. Huang, L. Xiaoyue, Z. Min, and X. Deng, "Detection of phishing webpages based on visual similarity," in *Proc. Special Interest Tracks Posters 14th Int. Conf. World Wide Web (WWW)*, 2005, pp. 1060–1061, doi: 10.1145/1062745.1062868.
[6] Y. Zhang, J. I. Hong, and L. F. Cranor, "Cantina: A content-based approach to detecting phishing web sites," in *Proc. 16th Int. Conf. World Wide Web (WWW)*, 2007, pp. 639–648, doi: 10.1145/1242572.1242659.
[7] O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine learning based phishing detection from URLs," *Expert Syst. Appl.*, vol. 117, pp. 345–357, Mar. 2019, doi: 10.1016/j.eswa.2018.09.029.
[8] H. Abutair, A. Belghith, and S. Alahmadi, "CBR-PDS: A case-based reasoning phishing detection system," *J. Ambient Intell. Hum. Comput.*, vol. 10, no. 7, pp. 2593–2606, Jul. 2019, doi: 10.1007/s12652-018-0736-0.
[9] M. Zouina and B. Outtaj, "A novel lightweight URL phishing detection system using SVM and similarity index," *Hum.-Centric Comput. Inf. Sci.*, vol. 7, no. 1, p. 17, Jun. 2017.
[10] R. M. Mohammad, F. Thabtah, and L. Mccluskey, "Predicting phishing websites based on self-structuring neural network," *Neural Comput. Appl.*, vol. 25, no. 2, pp. 443–458, Aug. 2014, doi: 10.1007/s00521-013-1490-z.
[11] A. C. Bahnsen, E. C. Bohorquez, S. Villegas, J. Vargas, and F. A. Gonzalez, "Classifying phishing URLs using recurrent neural networks," in *Proc. APWG Symp. Electron. Crime Res. (eCrime)*, Apr. 2017, pp. 1–8, doi: 10.1109/ECRIME.2017.7945048.
[12] A. Anand, K. Gorde, J. R. A. Moniz, N. Park, T. Chakraborty, and B.-T. Chu, "Phishing URL detection with oversampling based on text generative adversarial networks," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2018, pp. 1168–1177.
[13] P. Yi, Y. Guan, F. Zou, Y. Yao, W. Wang, and T. Zhu, "Web phishing detection using a deep learning framework," *Wireless Commun. Mobile Comput.*, vol. 2018, pp. 1–9, Sep. 2018, doi: 10.1155/2018/4678746.
[14] R. Vinayakumar, K. P. Soman, and P. Poornachandran, "Evaluating deep learning approaches to characterize and classify malicious URL's," *J. Intell. Fuzzy Syst.*, vol. 34, no. 3, pp. 1333–1343, Mar. 2018, doi: 10.3233/JIFS-169429.
[15] S. G. Selvaganapathy, M. Nivaashini, and H. P. Natarajan, "Deep belief network based detection and categorization of malicious URLs," *Inf. Secur. J., Global Perspective*, vol. 27, no. 3, pp. 145–161, Apr. 2018, doi: 10.1080/19393555.2018.1456577.
[16] S. Shivangi, P. Debnath, K. Sajeevan, and D. Annapurna, "Chrome extension for malicious URLs detection in social media applications using artificial neural networks and long short term memory networks," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2018, pp. 1993–1997, doi: 10.1109/ICACCI.2018.8554647.
[17] Y. Peng, S. Tian, L. Yu, Y. Lv, and R. Wang, "Malicious URL recognition and detection using attention-based CNN-LSTM," *KSII Trans. Internet Inf. Syst.*, vol. 13, no. 11, pp. 5580–5593, 2019, doi: 10.3837/tiis.2019.11.017.
[18] P. Robic-Butez and T. Y. Win, "Detection of phishing websites using generative adversarial network," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2019, pp. 3216–3221.
[19] X. Zhang, D. Shi, H. Zhang, W. Liu, and R. Li, "Efficient detection of phishing attacks with hybrid neural networks," in *Proc. IEEE 18th Int. Conf. Commun. Technol. (ICCT)*, Oct. 2018, pp. 844–848, doi: 10.1109/ICCT.2018.8600018.
[20] Y. Huang, Q. Yang, J. Qin, and W. Wen, "Phishing URL detection via CNN and attention-based hierarchical RNN," in *Proc. 18th IEEE Int. Conf. Trust, Secur. Privacy Comput. Commun./13th IEEE Int. Conf. Big Data Sci. Eng. (TrustCom/BigDataSE)*, Aug. 2019, pp. 112–119, doi: 10.1109/TrustCom/BigDataSE.2019.00024.
[21] J. Feng, L. Y. Zou, and T. Z. Nan, "A phishing webpage detection method based on stacked autoencoder and correlation coefficients," *J. Comput. Inf. Technol.*, vol. 27, no. 2, pp. 41–54, Nov. 2019, doi: 10.20532/cit.2019.1004702.
[22] P. Yang, G. Zhao, and P. Zeng, "Phishing website detection based on multidimensional features driven by deep learning," *IEEE Access*, vol. 7, pp. 15196–15209, 2019, doi: 10.1109/ACCESS.2019.2892066.
[23] A. Al-alyan and S. Al-ahmadi, "Robust URL phishing detection based on deep learning," *KSII Trans. Internet Inf. Syst.*, vol. 14, no. 7, pp. 2752–2768, 2020, doi: 10.3837/tiis.2020.07.001.
[24] M. Darling, G. Heileman, G. Gressel, A. Ashok, and P. Poornachandran, "A lexical approach for classifying malicious URLs," in *Proc. Int. Conf. High Perform. Comput. Simul. (HPCS)*, Jul. 2015, pp. 195–202, doi: 10.1109/HPCSim.2015.7237040.
[25] W. Chen, W. Zhang, and Y. Su, "Phishing detection research based on LSTM recurrent neural network," *Data Sci.*, vol. 6, pp. 638–645, Sep. 2018, doi: 10.1007/978-981-13-2203-7_52.
[26] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 649–657.
[27] N. Q. Do, A. Selamat, O. Krejcar, T. Yokoi, and H. Fujita, "Phishing webpage classification via deep learning-based algorithms: An empirical study," *Appl. Sci.*, vol. 11, no. 19, p. 9210, Oct. 2021, doi: 10.3390/app11199210.

SAAD AL-AHMADI (Senior Member, IEEE) is currently an Associate Professor with the Department of Computer Science, King Saud University, Saudi Arabia. He has published many articles in highly cited journals and worked as a part-time Consultant at several government organizations and the private sector. His current research interests include the IoT security, machine learning for cybersecurity, and future generation networks.

AFRAH ALOTAIBI received the B.S. and M.S. degrees in computer science, King Saud University, Saudi Arabia, in 2018 and 2021, respectively. Her current research interests include web security, machine learning, and computer networks.

OMAR ALSALEH, photograph and biography not available at the time of publication.