SN Computer Science (2022) 3:189
https://doi.org/10.1007/s42979-022-01078-0

ORIGINAL RESEARCH

Implementation of 'Smishing Detector': An Efficient Model for Smishing Detection Using Neural Network

Sandhya Mishra[1] · **Devpriya Soni**[1]

Received: 15 November 2021 / Accepted: 19 February 2022 / Published online: 15 March 2022

Abstract

Neural network creates a neuron-based network similar to the human nervous system to solve classification problems efficiently. The smishing problem is a binary classification problem in which attackers target smartphone users through text messages. As smishing is a remarkable cybersecurity issue that is troubling researchers and smartphone users these days. Addressing this security issue using the most efficient algorithm is the need of the hour. This manuscript presented an algorithm for the model proposed by authors in 'Smishing Detector' model and implemented it using Neural Network. The result obtained proves that the neural network is much efficient in detecting smishing problem. Neural Network outperformed other machine learning algorithms with a difference of 1.11%. Neural Network performed with the final accuracy of 97.40%. In this paper, system extracted the most efficient features of smishing SMS (Short Message Service) using the Neural Network. This manuscript also reported the accuracy shown by the system for each feature selected and implemented. It is evident from the implementation that each feature selected is most effective in smishing detection and URL (Uniform Resource Locator) feature is the most effective feature with an accuracy of 94%.

Keywords Phishing · Cyber security · Mobile security · Back propagation algorithm · Information security

Introduction

Smishing is a cyber security issue faced by smartphones these days. Smishing is initiated by an attacker when he sends malicious text messages to smartphone users. These text messages [4] contain a URL, phone number, or email id and some text which prompts the user to contact the attacker. If the user is contacting the attacker through the phone number or email id, he instructs the user to download a malicious application. The user is unaware of the assailant's intention, downloads the malicious application trusting it as a genuine application. This application is downloaded into the user's device that enables the attacker to access the device and in turn he stoles the user's financial as well as personal credentials. If the user accesses the link attached in the SMS, then it redirects him/her to a malicious website which in turn downloads an illegitimate application. The website might also ask the victim to fill up his/her sensitive details in an interface displayed on the website. Recently, this smishing issue is affecting millions of users across the world. Smartphone users are losing their hard-earned money being trapped by the assailants.

The report generated by the Anti-Phishing Working Group(APWG) [1] states that phishing websites are rising every year. During the COVID-19 pandemic, there has been a huge rise in the phishing and smishing attacks related to the COVID-19 situation. Attackers were targeting the COVID-19 patients through offering blood plasma, providing COVID-19-related information claiming to be from government sources, and targeting officials doing work from home during the COVID-19 situation. The UN (United Nations) reported 350% increase in phishing attacks during the pandemic period [2].

This article is part of the topical collection "Advances in Computational Approaches for Artificial Intelligence, Image Processing, IoT and Cloud Applications" guest-edited by Bhanu Prakash K N and M. Shivakumar.

✉ Sandhya Mishra
sandhyashankar20@gmail.com

Devpriya Soni
devpriya.soni@jiit.ac.in

[1] Department of Computer Science & Engineering and Information Technology, Jaypee Institute of Information Technology, Sector-128, Noida, India

Text messages [5] are preferred by attackers for phishing attacks out of the various communication mediums like email, SMS, telephone, and others. Attackers prefer text messages due to their high response rate from users. A report presented by CallHub [3] described that the users responded to SMS at a much higher rate than email.

Neural Network is a supervised machine learning technique to solve a problem by creating a network similar to human brain neurons. It learns from the dataset and creates a Neural Network that can make intelligent decisions on its own. Neural Network needs target output to be provided along with the given set of inputs so that it can compare with the predicted output. First, it predicts an output called predicted output using an activation function. This process is called feed-forward.

A loss function is used to calculate the error by quantifying the variation between the target output and actual output. Now, the error is reduced by updating the weights and biases provided for a given set of inputs. The optimization function is used to minimize the error. To reduce the error, a partial derivative of the loss function is calculated which is called gradient descent. This derivative helps in finding the slope of the error function. A positive value of slope indicates that the weights need to be decreased and a negative value indicates that the weights should be increased.

Feedforward is predicting the output and backpropagation is calculating and minimizing the error. A complete round of feed-forward and backpropagation is known as one "epoch". This iteration is continued until a decent accuracy is attained. When the model has learned to predict the output for a given set of data with a minimum margin of error is termed as convergence. The architecture of the Neural Network is arranged in layers. One input layer, one output layer, and one or more than one hidden layer.

Figure 1 shows the iterative process of the ANN (Artificial Neural Network). Here, the process starts with the initialization of the model by applying values to weights and biases associated with the neurons. The forward propagation uses an activation function to calculate the actual output. An error function is used to calculate the error, i.e. difference between the actual output obtained by the system and the target output. In backward propagation, an optimization method is used to bring down the error to minima. Weights and biases are updated in the network to minimize the error obtained. This process is repeated for certain number of iterations till the model achieves convergence and final prediction is recorded.

In the smishing problem, messages are classified as either malicious or ham. Smishing messages are a subset of spam messages. Spam messages are messages sent in bulk for advertising purposes or for promotion of products and services. A spam message containing malicious links or attachments that are intended for stealing the user details

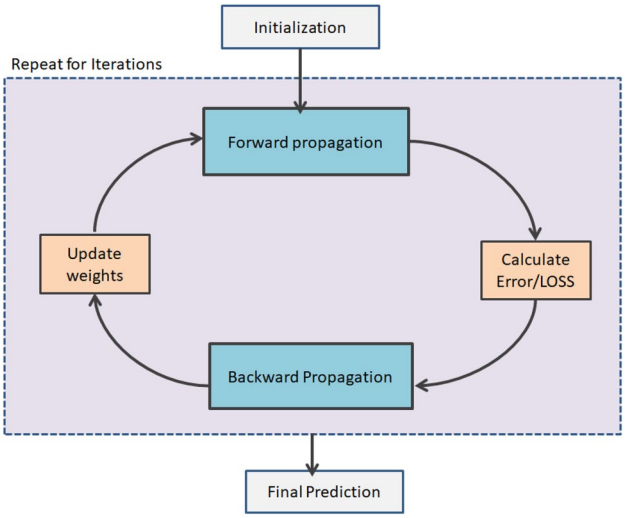

Fig. 1 Iterative process of ANN

are termed as smishing message. In the event of smishing detection, malicious messages are classified as smishing and all other messages whether it is ham or spam are categorized as HAM (legitimate). Hence, smishing is a binary classification problem in which the messages are classified either as legitimate or smishing. This research work focuses to resolve this smishing issue using Neural Networks.

In the previous research papers, authors proposed a model for addressing the smishing problem. The paper titled 'Smishing Detector' [11] followed a flow-based approach and used various features of the SMS and the URL included in it to classify the problem. The above-mentioned approaches used machine learning algorithms for the classification part. Authors have used three machine learning algorithms, namely Decision Tree, Random Forest, and Naive Bayes.

This paper is aimed to implement an efficient smishing detection system using ANN. Here, the Smishing detector [11] system is implemented using ANN. Also, the best 7 features of the smishing SMS are extracted using Neural Network. The best 7 features are noted and accuracy is recorded.

The following components are proposed in this paper:

- Artificial Neural Network is implemented for 'Smishing Detector' model.
- The best 7 features of the Smishing SMS are extracted using Neural Network.
- The accuracy of each feature is reported for efficient smishing detection.
- The algorithm of the 'Smishing Detector' [11]model is presented.
- The 'Smishing Detector'[11] model is implemented using Neural Network and its evaluation using real-time SMS datasets is presented.

This research paper is arranged as given below: related work study of this research work is explained in the next section. The research work presented in this paper is elaborated in the subsequent section. Implementation details and outputs are shown in the penultimate section, and finally, this research work is concluded.

Related Work Study

During the literature survey, it was observed that few research papers [9, 10, 12–18] are contributing to the study of SMS classification using Neural Network whereas other research papers [4, 8, 11, 19–22] are related to the study of smishing detection and evaluating the detection models using various techniques. Some research papers [23–28] are focused on the detection of phishing. Smishing is a type of Phishing and attackers use similar techniques in designing both the attacks.

In the research paper [9], Ankit et.al. introduced a system in which smishing messages were segregated from spam messages. Some new set of heuristics were used for identifying smishing messages from a set of spam messages. In the research paper [10], CNN (Convolutional Neural network) and LSTM (Long Short-term Memory) are combined to segregate spam messages and reported an accuracy of 98.37%. In the research paper [12], Averaged Neural Network with one hidden layer is used for the classification of spam messages and reported an accuracy of 98.8%. In the research paper [13], the Multilayer perceptron model using the backpropagation algorithm is used for the segregation of malicious emails. The accuracy shown by their model is 95%. In the research paper [14], the backpropagation algorithm is used for the classification of spam messages. 14 spam features of the messages were selected for the identification of spam messages and finally, their classification showed an accuracy of 95.81%. In the research paper [15], Deep Neural Network is used for segregation of spam messages and Restricted Boltzmann Machine is used for extraction of relevant features. Neural Network with three hidden layers provided the best accuracy for their system. In the research paper [16], spam messages are segregated with the help of ten features selected using Neural Network. The proposed approach was also proved effective for the zero-hour attack. The accuracy shown by the system is 98.74%. In paper [17], CNN and RNN (Recurrent Neural network)are combined to form a C-LSTM model which is used for the classification of sentiments and questions out of a given test dataset. CNN is used for phrase extraction and RNN is used for sentence formation using the extracted phrases. In the research paper [18], CNN and LSTM based deep learning model is used for the classification of spam messages. Their model is executed in three stages: a word matrix is created in the first stage, features were identified in the second stage and finally, classification is performed in the third stage. The reported accuracy is 99.44%.

In the research paper [4] titled 'SMS Phishing and Mitigation Approaches', various smishing detection strategies, techniques and approaches are elaborated to bring awareness among researchers, mobile users, and students. SmiDCA [19], paper implemented a correlation algorithm for the detection of smishing. 20 best features are selected 39 smishing features using dimensionality reduction technique and forwarded these features to correlation algorithm. Finally, reported an accuracy of 96.4%. In the paper titled S-detector [20], the presence of URL in the SMS and malicious file downloading is analyzed to detect smishing. Smishing keyword classification is also performed using classification algorithms. In the research paper [21], a content-based method is followed. The presence of form tag in source code and malicious file downloading through link present the SMS is inspected to detect smishing message. Kang et al. [22] proposed a research work in which they introduced a URL validation test and smishing box approach to authenticating the URL in the text message. In the research paper [11] titled 'Smishing Detector', four stages are introduced in the detection of smishing. An accuracy of 96.2% is reported after the implementation of all four modules. The main focus of this research work is to inspect the authenticity of the URL in the SMS. Lee et al. [8] implemented a cloud-based virtual environment to verify the downloading of a malicious executable file and in turn detects the maliciousness of the message.

In a model called 'PhiDMA', five stages are introduced to detect phishing. Various phishing features are verified in each stage and finally reported accuracy of 92.72%. Wu et al. [24] presented a model called ''MobiFish'' in which 'AppFish' was introduced to detect phishing in mobile applications and 'WebFish' to detect smishing through web pages in mobile interface. URL and source code of the URL is inspected to verify phishing in this model. In the research paper [25], hosted features and lexical features of the URL are inspected to detect phishing. Finally, machine learning classifiers are used to identify smishing in different datasets. Mohammad et al. [26] presented a phishing detection model in which 17 features are verified for the identification of phishing websites. 17 best features are identified with the help of the frequency of each feature in the phishing dataset. In the research paper [27], a model called CANTINA is introduced. In this, a TF-IDF (Term Frequency-Inverse Document frequency) score-based signature is formed and is provided to the search engine to identify the legitimate website and thereby differentiating the legitimate and phishing website. Sophie et al. [28] presented a paper in which 20 heuristic tests were performed to detect phishing. Phishark was introduced in this paper as a toolbar against phishing. They have proved that URL-based heuristics and source code-based heuristics are useful in the detection of phishing.

Research Work

Neural Network is implemented in three phases, Training, Validation, and Testing. For the implementation of each phase, we need data. So, we divide the data into 3 parts: training data, validation data, and testing data. The training data and target output is provided to the network during the training phase. The training phase includes initialization of the network, forward propagation using an activation function, and backward propagation to calculate the errors. In the training phase, the actual output is obtained.

The testing phase includes only forward propagation to obtain the actual outputs from the inputs provided as shown in Fig. 2.

Initialization is the process of assigning random values to weights and biases in the network. In this system, bias values ranging from [–1.0] to [1.0] are assigned to each neuron in the network. Random values in the range of [–1.0] to [1.0] are allocated to the weights of the connections. Neural Network consists of neurons arranged in layers. The input layer is the first layer and the output layer is the last year. There might be multiple hidden layers between the input layer and output layer of the system.

Some parameters are assumed as follows:

I represents the neurons in the input layer.

H represents the neurons in the hidden layer.

O represents the neurons in the output layer.

W_{ij} is the weight between the ith neuron of the input layer to the jth neuron of the hidden layer.

b_i is the bias corresponding to the ith neuron of the hidden layer.

x_i denotes the inputs.

Inputs are provided to each neuron in the input layer. As shown in Eq. (1), a weighted sum (s_j) is calculated. This weighted sum is provided to an activation function. Here, sigmoid activation function is used as shown in Eq. (2). a_j is the output of the hidden layer, thus obtained from Eq. (2)

$$S_j = b_j + \sum_{i=1}^{I} W_{ij} x_i, \tag{1}$$

$$a_j = \frac{1}{1 + e^{(-s_j)}}. \tag{2}$$

The weighted summation of the outputs (a_j) is provided to the output layer neurons as shown in Eq. (3) and the final output (a_k) is obtained on applying the activation function. This formulates the feed-forward phase of the neuron network. Here, the final output (a_k) is obtained by going through each layer in the network.

$$S_k = b_k + \sum_{j=1}^{J} W_{jk} a_j, \tag{3}$$

$$a_k = \frac{1}{1 + e^{(-s_k)}}. \tag{4}$$

Once the feed-forward phase is completed, backward propagation is required to calculate the error, i.e. the difference between the actual output (a_k) and target output (t_k). A mean squared error function is used to calculate the error as shown in Eq. (5)

$$E = \frac{1}{2}\left(t_k - a_k\right)^2. \tag{5}$$

The gradient of the error is calculated in Eq. (6) where a_k is the final output.

$$\frac{\partial E}{\partial a_k} = \frac{\partial\left(\frac{1}{2}\left(t_k - a_k\right)^2\right)}{\partial a_k}. \tag{6}$$

The derivative of the error is calculated in Eq. (7), which is obtained from Eq. (6).

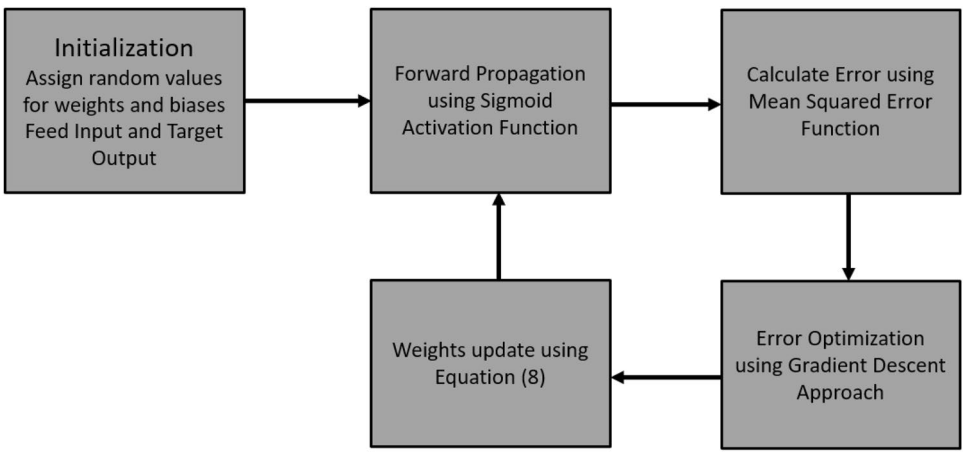

Fig. 2 Architecture of the system using neural network

$$e_k = -\left(t_k - a_k\right). \tag{7}$$

In artificial neural network, the weights are updated based on the value of the derivative. The value of the derivative indicates that the weights need to be increased or decreased. Momentum is used to maintain consistency in the weight updating equation which is given in Eq. (8).

$$W_{kj}^{new} = W_{kj}^{old} + \eta \frac{\partial E}{\partial W_{kj}} + \alpha \Delta W_{kj}^{old}. \tag{8}$$

Here, η is the learning rate and it denotes the amount of change required in the model at each iteration to reach the expected output. The value of the learning rate is set between 0 and 1. The smaller value of the learning rate causes slower convergence whereas a higher value causes rapid convergence at the local minimum. Here, a high value of learning rate has the possibility of escaping the local minimum. The value of momentum is also retained between 0 and 1.

In the validation stage, different values of hyper-parameters are tried in the network to get the best accuracy. Hyper-parameters include the number of neurons (H), learning rate (η), and momentum (α). The values of hyper-parameters which yield the greatest accuracy on our data set are shown in Table 1. In the testing phase, the trained network is proceeded using the testing data set. The results shown by the network are noted as the final accuracy of the system.

Table 1 Hyperparameters used in implementing the system

Hyper parameters	Values
No. of neurons	10
Activation function	Sigmoid
Momentum	0.7
Learning rate	0.2

Learning Rate (η) denotes the amount of rectification applied at a single iteration of the algorithm. Initially, the value of the learning rate is set at 1.0 with a decrease of 0.1 at each step till the value of 0.1 is reached. For each iterated value of the learning rate, the accuracy shown by ANN is recorded and is indicated in Fig. 3. Initially, at the value of the learning rate set to 1, ANN exhibited an accuracy of approximately 83%. Each decrement in the value of the learning rate caused an increase in the accuracy shown by the network.

The behavior of error corresponding to learning rate is shown in Fig. 4. The behavior of error is contrasting to that of the accuracy when the learning rate of the algorithm is incremented from 0 to 1.

Feature Extraction

Figure 5 shows the architecture of the model. Our dataset is a collection of 5858 text messages in which 538 are smishing messages and 5320 are ham messages. This dataset is pre-processed before proceeding with the actual feature extraction and evaluation. Pre-processing is a task of preparing the text data actually fit for the analyzation and evalution by machine learning algorithms. Hence, before passing the data to machine learning algorithms, text pre-processing is done. Text pre-processing includes converting each word in the text to lower case letters, removal of punctuations and special strings included in the message and stemming. Stemming is the process of converting each word to its root form. Tokenization is done by splitting the words into tokens and finally, a word vector corpus is prepared.

Feature selection is a process that aims to select the most relevant features that identify a particular problem. Here, the 7 most relevant features of the smishing problem are

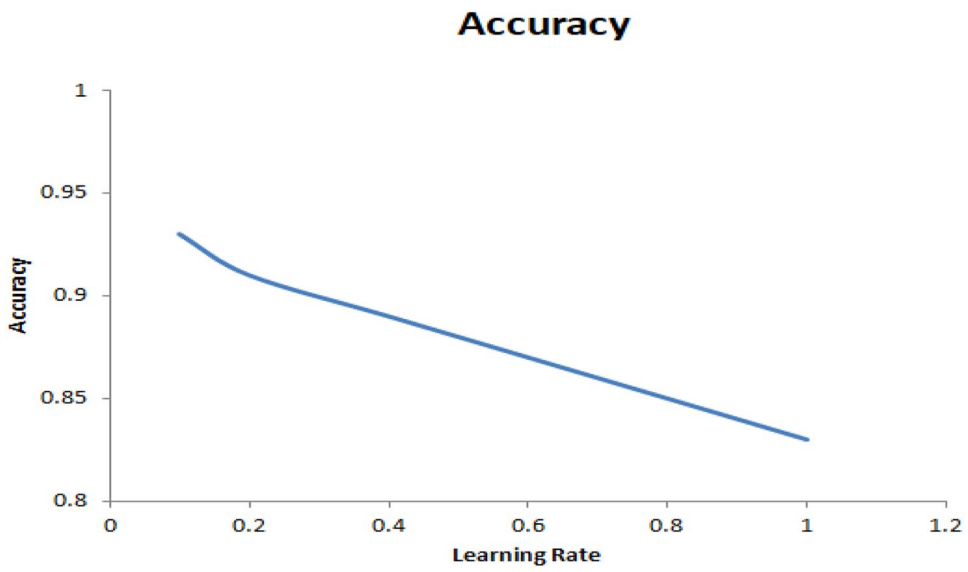

Fig. 3 Accuracy with respect to learning rate

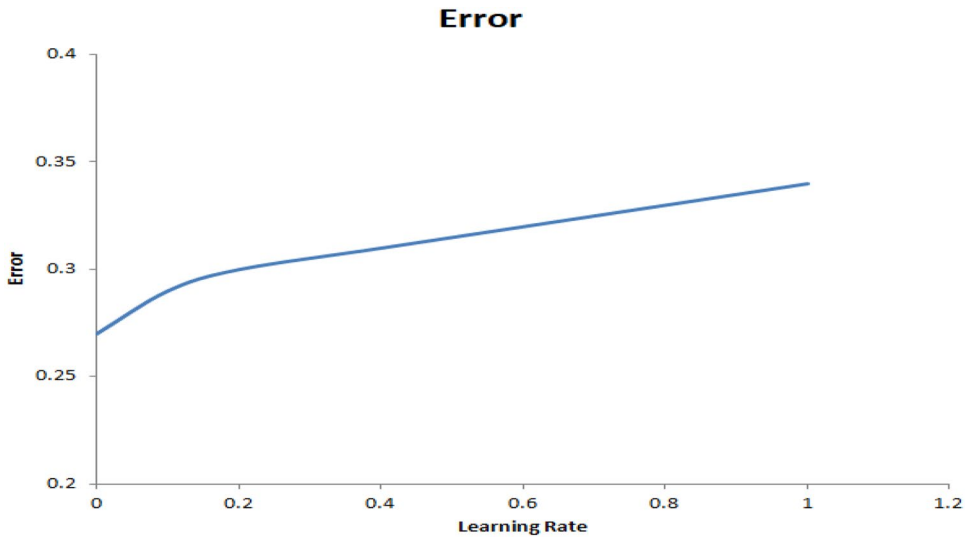

Fig. 4 Error corresponding to learning rate

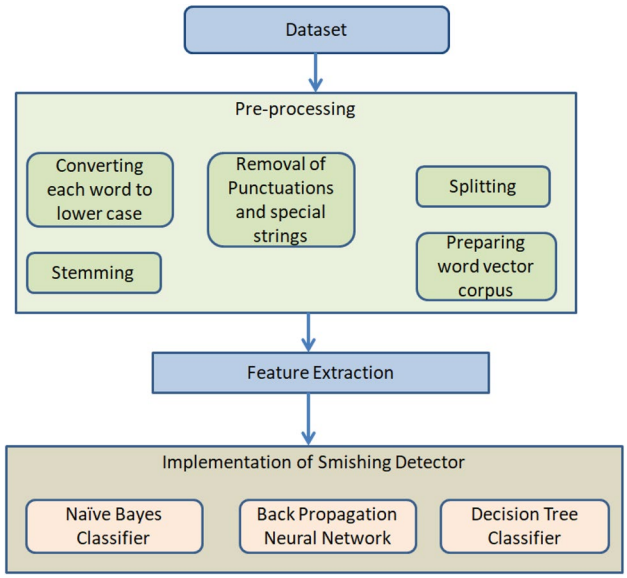

Fig. 5 Architecture of the model

extracted using ANN. In ANN, feature extraction is done using one input layer that includes all the extracted features. This is obtained by keeping track of the minimum value of error obtained. The selected feature sets are provided to the testing phase and the accuracy is recorded. The feature set which exhibited the best result is observed as the most pertinent feature set.

The Neural Network Algorithm is implemented to find out the 7 most prominent features of the smishing problem which are listed in Table 2. Results and accuracy of the above feature extraction are given in "Experimental analysis".

After feature extraction, the 'Smishing Detector' model is implemented using Back Propagation Neural Network. The results of the implementation are compared with other machine learning algorithms, namely, Decision Tree and Naive Bayes. The final accuracy, TPR (True-Positive Rate) and TNR (True Negative Rate) of the model are depicted in Experimental Analysis.

Table 2 Seven features selected using neural network algorithm

No.	Features	Description
1	url	A link present in the message redirects the user to malicious websites
2	email_id	An email id of the attacker included in the message
3	phone_no	A phone number of the attacker included in the message
4	leet_words	Words in which look-alike symbols and numerals are used in place of alphabets to give a similar look
5	smishing_keywords	Words added in the message which prompts the user to contact the attacker
6	Symbols	Symbols like %, / etc. included in the message to delude the user
7	special_characters	Special characters like $, ! etc. are added in the message to attract the user

Smishing Detector Model

The 'Smishing Detector' [11] model is presented in Algorithm 1. The smishing detector model is implemented in four modules, namely Content Analyzer Module, URL Filter Module, source Code Analyzer Module, and APK Download Detector Module. Content Analyzer Module is extracting the malicious features of the SMS text. The URL filter is extracting the malicious features of the URL. Source Code Analyzer is inspecting the source code of the website to identifying malicious activities and finally, APK Download Detector is keeping track of any executable file being downloaded into the user's device.

Algorithm 1 : SMS Content Analyzer Algorithm

Inputs
Incoming text messages(currentsms)
Signatures (SMS body)
Patterns (Phone_Number, URL, SAL, and EMail_ID)
Blacklist (Email_Blacklist, Phone_Blacklist)
Outputs
Status(Smishing SMS, Legitimate SMS)

```
1: while currentsms.moveToNext do
2: if SMS body matches URL or SAL then
3: call URL Filter Algorithm
4: else if SMS body matches Phone_number or EMail_ID then
        5: Extract Email and Phone from SMS body
        6:If Email matches Email_Blacklist or Phone matches Phone_Blacklist
        7:Return Smishing SMS

        8: else
            9: call Back Propagation Algorithm for keyword classification
            10: if prediction is malicious then
            11: return Smishing SMS
            12:else
            13:return legitimate SMS
            14:end if
        15:end if
16:end if
17: else
18: return legitimate SMS
19:end if
20:end while
```

The working of the smishing detector model is described as follows:

Step 1: **SMS Content Analyzer** (Algorithm 1) scrutinizes the appearance of a link in the SMS. In case a link is detected, then SMS proceeds for URL inspection using the URL Filter algorithm. In case the URL is not detected, the existence of an email id or telephone number in the message is scrutinized. If none of them are detected, the message is regarded as ham.

In case if the telephone number or email id is detected, corresponding blacklists are examined and messages containing blacklisted telephone numbers and email ids are declared as malicious else keyword classification is performed using the Neural Network. If the prediction appears to be malicious, the message is regarded as Smishing else Ham.

Step 2: In **URL Filter** (Algorithm 2), short URL is changed to long URL, then messages having blacklisted URLs are declared as malicious else four features of phishing URL are examined, namely, Age of Domain, presence of @tag, presence of hyphen and number of dots. The maliciousness of message decided based on the threshold ($T >= 3$) of the above heuristics else URL proceeds to Source Code Analyzer.

Step 3: In **Source Code Analyzer** (Algorithm 3), the existence of form tag in source code of URL confirms the maliciousness of the message else URL proceeded to APK download detector. If a form tag is detected, domain checking is also performed to check the similarity of SMS URL and redirection URL in source code. A difference in domain confirms the maliciousness of the message else URL proceeds to the APK download detector.

Step 4: Now the **APK Download Detector** (Algorithm 4) examines the status of file downloading with or without user consent and if without user consent is detected, it confirms the Smishing SMS. The message is declared as Ham if consent of the user is taken before downloading APK.

Algorithm 2 URL Filter Algorithm

Inputs
Incoming text messages(currentsms)
current_URL (currentsms.URL)
URL_Blacklist
Outputs
Status(Smishing SMS, Legitimate SMS)

```
1: while currentsms.moveToNext do
2:convert short URL to Long URL
3:If current_URL matches URL_Blacklist
4: return smishing SMS
5:else
        6: for  URL Check do
        7:check=0
        8:If age of domain of current_URL is less than 6 months then
        9:check==check+1
       10:end if
       11:If @ tag in current_URL then
       12:check=check+1
       13:end if
       14:If hyphen in current_URL then
       15:check=check+1
       16:end if
       17:If number of dots in current_URL greater than or equal to 5 then
       18:Check=check+1
       19:end if
       20:Return check
       21:end for

    22: If check greater than or equal to 3 then
    23: return Smishing SMS
    24:else
    25:call Source code Analyzer Algorithm
    26:end if
27:end if
28:end while
```

Algorithm 3 Source Code Analyzer Algorithm

Inputs
Incoming text messages(currentsms)
current_URL (currentsms.URL)
Outputs
Status(Smishing SMS, Legitimate SMS)

```
1: while currentsms.moveToNext do
2:Extract current_domain of current_URL
3:Get HTML source code of URL
4:If form tag in source code then
        5: If domain_name in source code matches current_domain then
        6: return legitimate SMS
        7:else
        8:call APK Download Detector Algorithm
        9:end if
10:else
11:call APK Download Detector Algorithm
12:end if
13:end while
```

Algorithm 4 APK Download Detector Algorithm

Inputs
Incoming text messages(currentsms)
current_URL (currentsms.URL)
Outputs
Status(Smishing SMS, Legitimate SMS)

```
1: while currentsms.moveToNext do
2:Extract target_URL of current_URL
3:If .apk in basename of target_URL then
        4:If user consent is taken then
        5:Return legitimate SMS
        6:else
        7:return smishing SMS
        8:end if
9:else If .apk in basename of target_URL after re-direction then
        10:If user consent is taken then
        11:Return legitimate SMS
        12:else
        13:return smishing SMS
        14:end if
15:else
16:return legitimate SMS
17:end if
18:end if
19:end while
```

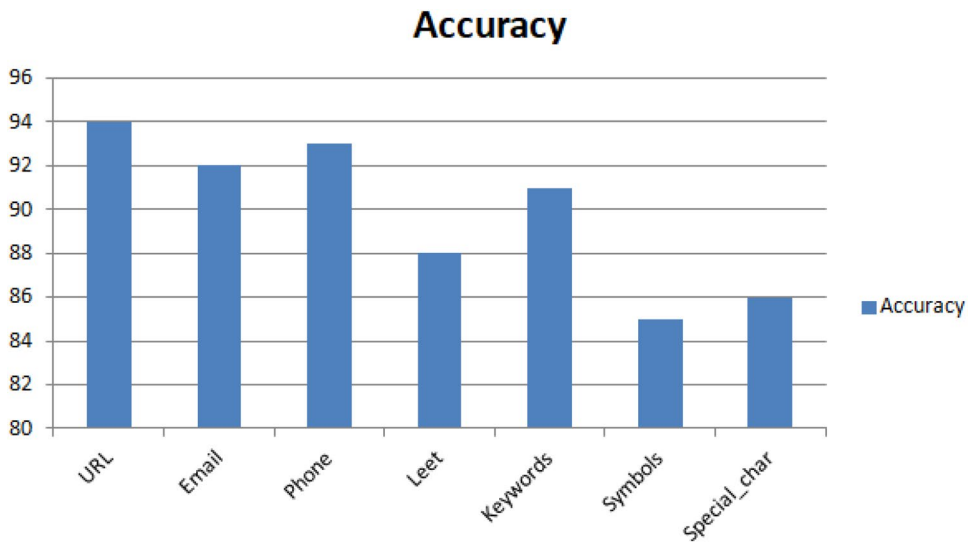

Fig. 6 Accuracy shown by the system for the best 7 features selected

Experimental Analysis

For the implementation part, real-time smishing messages collected from different sources, Almeida [6] dataset and pinterest.com [7] are used. Almeida is spam dataset from which few smishing text messages are manually extracted. Pinterest.com is a social website in which smartphone users have uploaded the screenshots of smishing text messages received in their devices for creating a dataset. These images were collected from pinterest.com and converted to text form for adding into the final dataset. Our final dataset contains a total of 5858 messages with 538 SMS labeled as Smishing and 5320 SMS labeled as Ham. The smishing detector system is implemented in Python language using Jupyter Notebook.

Smishing Detector achieved a better accuracy using Neural Network in comparison to machine learning algorithms. Although the difference is minor, Neural Network proofs itself to be the best when learning from the data is concerned.

Feature Extraction

We have extracted few features of the smishing SMS using Neural Network. The best 7 features have been selected by implementing Neural Network. These features are selected based on the minimum error shown by the network while implementation. Results shown by the Neural Network while implementing each of the seven extracted features are shown in Fig. 6. It shows that the URL feature alone can determine smishing messages with approximately 94% accuracy. URL, phone number and email id are identified as the most prominent 3 smishing features because these denotes the contact detail of the attacker. Attackers include any one of these features in the text message to divert the user through these channels and to collect his/her sensitive details. Smishing keywords and leet words are also identified as leading smishing feature included in text messages with an accuracy of 91% and 88%, respectively.

Smishing Detector

The result achieved while implementing 'Smishing Detector' [11] using Neural Network is shown in Table 3. The final accuracy achieved by the 'Smishing Detector' model using Neural Network and its comparison with the results of the same model using machine learning algorithms are also shown in Table 3. The achieved accuracy is compared with the classification results of machine learning algorithms. The comparison shows that Neural Network has achieved a better accuracy with a difference of 1.11%. fivefold cross-validation of the dataset is performed for the proposed system, i.e. Smishing Detector [11]. Table 3 depicts the cross-validation results.

Results shown by Neural Network for smishing detection are depicted in the confusion matrix in Fig. 7. Out of 538 smishing messages, 497 were predicted as smishing, and 41 were predicted as ham messages. 5209 messages out

Table 3 Comparison of Neural Network results with other ML algorithms

Algorithms	Accuracy	True-positive Rate	True-Negative Rate
Neural network	97.40	92.37	97.91
Naive bayes	96.29	86.43	97.29
Decision tree	93.40	74.90	95.28

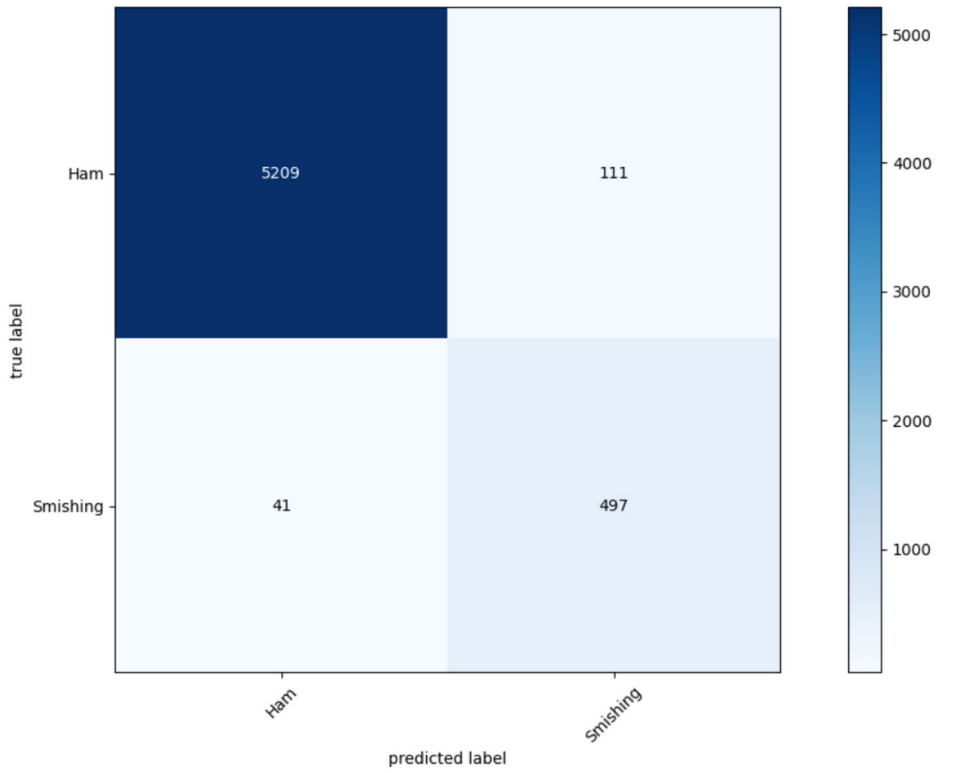

Fig. 7 Confusion matrix showing the results of neural network

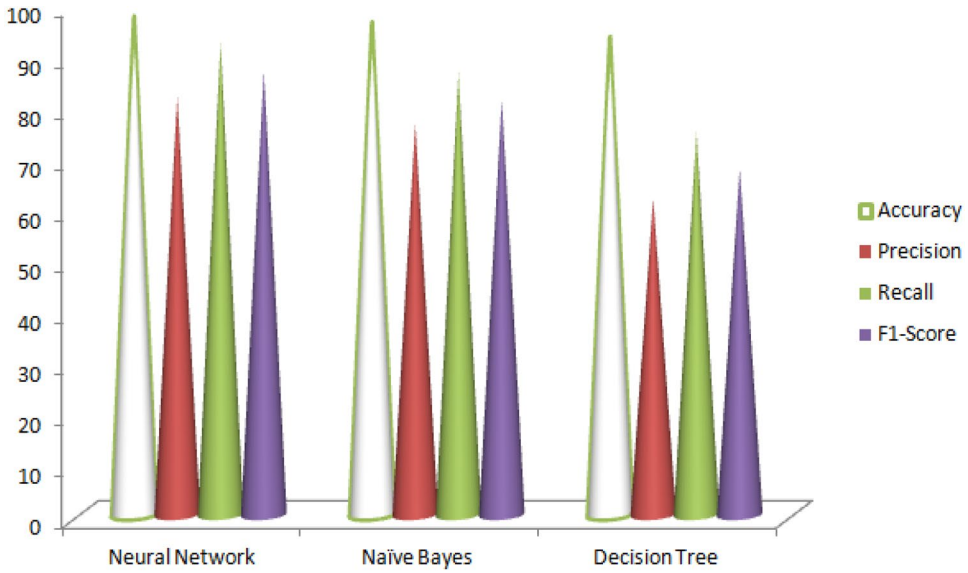

Fig. 8 Comparison of results shown by neural network with other algorithms

of 5320 messages were predicted as ham. Hence, it shows a true positive rate of 92.37% and a true negative rate of 97.91% as shown in Table 3. The comparison chart of Precision, Recall, and F1-score is depicted in Fig. 8. Neural Network has shown an F1-score of 86.72 and Naive Bayes has shown 81.07.

Conclusion and Future Scope

Smishing is a mobile security issue in which attackers target smartphone users by sending text messages. Smishing is a Phishing attack initiated through a text message. In this paper, we have aimed to classify the text messages into two

categories, Smishing or Ham using Neural Network. For the implementation of Neural Network, the ANN algorithm and its counterparts are well studied and analyzed. For the Smishing Detection approach proposed in the paper 'Smishing Detector' [11], an algorithm is presented for Smishing Detection using Neural Network. Neural Network is implemented for smishing detection and the results thereof shown that the Neural Network outperforms other classification algorithms like Naive Bayes and Random Forest. In this paper, Neural Network is also implemented for extracting the best 7 features of Smishing SMS.

As smishing is a serious cybersecurity issue that is troubling researchers and smartphone users these days, addressing this security issue using the most efficient algorithm is the need of the hour. Hence, it is targeted to prevent this issue using Artificial Neural Network. The results shown by the implementation has shown good accuracy which proved that the system is efficient in the detection of smishing SMS and thereby preventing smishing attack.

In the future scope of this research work, deep learning techniques can be used to implement SMS Phishing detection models. These deep learning models are expected to give high accuracy when compared with other neural network models. Deep learning models can be evaluated using large amount of data if available. The larger the data, higher the accuracy in case of deep learning models with more number of hidden layers.

Authors' Contributions SM: conceptualization, methodology, software, data curation, writing—original draft, visualization, and investigation. DS: conceptualization, supervision, validation, and writing—review and editing.

Data Availability The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Anti-Phishing Working Group (APWG). Phishing activity trends report. 2020. https://apwg.org/. Accessed 2020.
2. The New Indian Express. Increasing cybercrime: UN reports 350 per cent rise in phishing websites during pandemic. 2020. https://www.newindianexpress.com/business/2020/aug/08/increasing-cybercrime-un-reports-350-per-cent-rise-in-phishing-websites-during-pandemic-2180777.html. Accessed 2020.
3. CallHub. 6 reasons why sms is more efective than email marketing—callhub. 2016. https://callhub.io/6-reasons-sms-efectiveemail-marketing/. Accessed 2018.
4. Mishra S, Soni D. SMS phishing and mitigation approaches. In: Twelfth International Conference on Contemporary Computing (IC3), Noida, India, 2019, pp. 1–5, https://doi.org/10.1109/IC3.2019.8844920.
5. McAfee. Protect yourself from smishing. 2012. https://securingtomorrow.mcafee.com/consumer/family-safety/protect-yourselffrom-smishing/. Accessed 2017.
6. Almeida TA, Hidalgo JMG, Yamakami A. Contributions to the study of SMS spam filtering: New collection and results, in: 11th ACM Symposium on Document Engineering, 2011, pp. 259–262
7. Pinterest, Smishing message images, November 20 2018, Retrieved from https://in.pinterest.com/seceduau/smishing-dataset/?lp=true.
8. Lee A, Kim K, Lee H, Jun M. A study on realtime detecting smishing on cloud computing environments. In: Park JJJH, Chao HC, Arabnia H, Yen NY, editors. Advanced multimedia and ubiquitous engineering. Berlin: Springer; 2016. p. 495–501.
9. Jain A. A novel approach to detect spam and smishing SMS using machine learning techniques. Int J E-Serv Mobile Appl. 2019;2:2. https://doi.org/10.4018/IJESMA.2020010102.
10. Ghourabi A, Mahmood MA, Alzubi QM. A hybrid CNN-LSTM model for SMS spam detection in Arabic and English messages. Fut Internet. 2020;12:156.
11. Mishra S, Soni D. Smishing detector: a security model to detect smishing through sms content analysis and url behavior analysis. Future Gener Comput Syst 2020;108:803–15. https://doi.org/10.1016/j.future.2020.03.021, http://www.sciencedirect.com/science/article/pii/S0167739X19318758.
12. Sheikhi S, Kheirabadi MT, Bazzazi A. An effective model for SMS spam detection using content-based features and averaged neural network. Int J Eng (IJE) IJE Trans B Appl. 2020;33(2):221–8.
13. Rao AS, Avadhani PS, Chaudhuri N. A content-based spam e-mail filtering approach using multilayer perceptron neural networks. Int J Eng Trends Technol. 2016;41:44–5. https://doi.org/10.14445/22315381/IJETT-V41P210.
14. Jain AK, Goel D, Agarwal S, et al. Predicting spam messages using back propagation neural network. Wireless Pers Commun. 2020;110:403–22. https://doi.org/10.1007/s11277-019-06734-y.
15. Mathappan N, Rs S, Angamuthu K, Thangaraj P. SMS spam detection using deep neural network. Int J Pure Appl Math. 2018;119:2425–36.
16. Jain A, Gupta BB. Feature based approach for detection of smishing messages in the mobile environment. J Inf Technol Res. 2019;12:17–35. https://doi.org/10.4018/JITR.2019040102.
17. Zhou C, Sun C, Liu Z, Lau F (2015) A C-LSTM neural network for text classification.
18. Roy PK, Singh JP, Banerjee S. Deep learning to filter SMS spam. Fut Gener Comput Syst. 2020;102:524–33.
19. GunikhanSonowal K, Kuppusamy S. SmiDCA: An anti-smishing model with machine learning approach. Comput J. 2018;61(8):1143–57.
20. Joo JW, Moon SY, Singh S, Park JH. S-detector: an enhanced security model for detecting smishing attack for mobile computing. Telecommun Syst. 2017;66:1–10.
21. Sandhya M, Soni D. A content-based approach for detecting smishing in mobile environment. Suscom. 2019. https://doi.org/10.2139/ssrn.3356256.
22. Kang A, Lee JD, Kang WM, Barolli L, Park JH. Security considerations for smart phone smishing attacks. Adv Comput Sci Appl. 2014. https://doi.org/10.1007/978-3-642-41674-3_66.
23. Sonowal G, Kuppusamy K. Phidma—a phishing detection model with multi-filter approach. J King Saud Univ Comput Inf Sci. 2017;29:1–15.
24. Wu L, Du X, Wu J. MobiFish: A lightweight antiphishing scheme for mobile phones. In: 23rd International Conference on Computer Communication and Networks, ICCCN, 2014, pp. 1–8.

25. Zhang J, Wang Y. A real-time automatic detection of phishing URLs. In: 2nd International Conference on Computer Science and Network Technology, ICCSNT, IEEE, 2012, pp. 1212–1216.
26. Mohammad RM, Thabtah F, McCluskey L. Intelligent rule-based phishing websites classification. IET Inf Secur. 2014;8:153–60.
27. Yue Z, Jason H, Lorrie C. Cantina: a content-based approach to detecting phishing web sites. 2007;639–648, https://doi.org/10.1145/1242572.1242659.
28. Sophie G-P, Gustavo GG, Maryline L. Decisive heuristics to differentiate legitimate from phishing sites. In: Conference on Network and Information Systems Security, La Rochelle, 2011, pp. 1–9, https://doi.org/10.1109/SAR-SSI.2011.5931389