Neural Computing and Applications (2023) 35:4975–4992
https://doi.org/10.1007/s00521-021-06305-y

S.I. : MACHINE LEARNING APPLICATIONS FOR SECURITY

DSmishSMS-A System to Detect Smishing SMS

Sandhya Mishra[1] · Devpriya Soni[1]

Received: 4 January 2021 / Accepted: 1 July 2021 / Published online: 28 July 2021

Abstract
With the origin of smart homes, smart cities, and smart everything, smart phones came up as an area of magnificent growth and development. These devices became a part of daily activities of human life. This impact and growth have made these devices more vulnerable to attacks than other devices such as desktops or laptops. Text messages or SMS (Short Text Messages) are a part of smartphones through which attackers target the users. Smishing (SMS Phishing) is an attack targeting smartphone users through the medium of text messages. Though smishing is a type of phishing, it is different from phishing in many aspects like the amount of information available in the SMS, the strategy of attack, etc. Thus, detection of smishing is a challenge in the context of the minimum amount of information shared by the attacker. In the case of smishing, we have short text messages which are often in short forms or in symbolic forms. A single text message contains very few smishing-related features, and it consists of abbreviations and idioms which makes smishing detection more difficult. Detection of smishing is a challenge not only because of features constraint but also due to the scarcity of real smishing datasets. To differentiate spam messages from smishing messages, we are evaluating the legitimacy of the URL (Uniform Resource Locator) in the message. We have extracted the five most efficient features from the text messages to enable the machine learning classification using a limited number of features. In this paper, we have presented a smishing detection model comprising of two phases, Domain Checking Phase and SMS Classification Phase. We have examined the authenticity of the URL in the SMS which is a crucial part of SMS phishing detection. In our system, Domain Checking Phase scrutinizes the authenticity of the URL. SMS Classification Phase examines the text contents of the messages and extracts some efficient features. Finally, the system classifies the messages using Backpropagation Algorithm and compares results with three traditional classifiers. A prototype of the system has been developed and evaluated using SMS datasets. The results of the evaluation achieved an accuracy of 97.93% which shows the proposed method is very efficient for the detection of smishing messages.

Keywords Smishing · Phishing · Paytm SMS scam · Mobile security · Machine learning · Backpropagation Algorithm · Cyber security · Covid-19 SMS Scam

1 Introduction

SMISHING is a mobile security issue recently getting popular all over the world. It is a phishing attack initiated through text messaging to befool the users. Smishing is a union of 'SMS' and 'phishing' in which the attacker uses text messaging instead of email [1]. It is a fraudulent attack in which a text message pretending to be a genuine message is sent to the mobile user. This text message often contains a URL that redirects the user to malicious websites. For instance, a smishing message pretending to be from your bank might ask you for personal or financial information through a link provided in the message. The text in the SMS prompts the user to click the link in the SMS. This link in turn will ask the user to provide his/her sensitive information in consideration of unblocking the debit/credit card, getting some gifts or discounts, etc.

Mobile users are less aware of the security risks associated with smartphones. Most of them assume that mobile

✉ Sandhya Mishra
17403003@mail.jiit.ac.in

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

devices are more secure than computers, but smartphones are at high risk because of their platform offering greater flexibility for cybercriminals. As mobile users are increasing, attacks against mobile devices are also skyrocketing. Another risk factor associated with mobile devices is that people often use mobile devices on the go and read text messages when they are in a hurry. This leads users to respond to SMS and click on malicious links without thinking about its maliciousness.

Attackers use several communication mediums to connect with the victims such as text messages, email, Whatsapp messages, and phone calls. [2]. However, SMS is a medium to communicate with mobile users without the internet. The report of Statista [3] reveals that the number of smartphone users in the world was 2.9 billion in 2018, and it would be approximately 3.8 billion in 2021. According to CallHub, the response rate of text messages is much higher in comparison with email [4]. This prompts the attackers in using text messages as a medium to communicate with the users. For attackers, sending text messages is less expensive because they can send a significant number of messages to the users [5] with a single SMS package. Smishers (Smishing attacker) target to get user's personal or financial data for which they might use two methods. They might trick users into downloading malware which will in turn send sensitive information to the attacker. On the other hand, the link in the text message might redirect the user to a fake website which in turn asks for user-id, passwords, and financial details.

Recently, the Paytm smishing scam was very alarming among smartphone users for defrauding their hard-earned money. In this smishing scam, fraudsters pretending to be from Paytm send fraudulent messages to smartphone users. These smishing messages include malicious texts which give an impression to the Paytm user that their KYC (Know Your Customer) is expired and needs to be renewed. These messages contain either a phone number, email id, or URL. The text in the message prompts the user to contact the attacker for not getting their account blocked. When the user contacts the attacker, they later ask the user to download an application through which the attacker gets access to the user's device remotely. In turn, attackers get all the sensitive details entered by the Paytm users, and attackers use these details to activate fraudulent financial transactions.

Many researchers have already worked on the smishing problem and have identified several methods to detect malicious text messages. Some of them have used heuristic methods in which researchers select some features of the text messages with the aid of classification algorithms and categorize the SMS based on these features, but these methods do not inspect the URL present in the SMS and its redirections. A message might contain suspicious features, but if it does not contain a URL to redirect the user to malicious websites, then it cannot bring much harm to the user until and unless the user contacts the attacker on the phone number provided in the message. Some researchers have used blacklist methods, but this method is not efficient because blacklists are not updated frequently. Attackers change the domain name of their website more frequently. Hence, blacklists are unable to track the frequently changing patterns of attackers. Some authors have used blacklists along with various other rules to examine the maliciousness of the message. Other smishing detection approaches were using some complex methods for smishing detection. Hence, we strive to propose a less complex and less time-consuming system to check the genuineness of the URL present in the SMS.

Though smishing is considered to be a type of phishing, phishing can be detected using the information available in the phishing email or in the phishing website. In case of smishing, the strategy of attack is different and we are left with very few amount of information shared by the attacker in the Short Text Messages. Moreover, attackers use idioms, emoticons, abbreviations, and leet words in the SMS. Hence, smishing detection becomes a difficult task considering the minimum amount of information we have to design the smishing detection strategies. The URL included in the text message is also a short URL. Some of the challenges faced in the detection of smishing SMS are listed below:

- The abbreviated form of text messages makes it difficult to analyze the maliciousness of the message. This leads to limited number of features extracted from the message, and hence, the identification of malicious SMS becomes difficult.
- Leet words, idioms, and misspelled words are used in the text message which leads to hassle in the identification of smishing keywords.
- Spam messages contain a similar set of features in comparison with smishing messages. Hence, differentiating among spam messages and smishing messages is a tedious task thereof.
- The scarcity of real-time, public smishing datasets makes it a challenge to evaluate the smishing detection systems.

To address the above-mentioned challenges, we have built a smishing detection system and evaluated it using real-time datasets. The description of the dataset is given in Sect. 5 of this paper. To differentiate among spam messages and smishing messages, the legitimacy of the URL in the message is evaluated. A URL in the message that redirects the user to malicious phishing websites conforms to the maliciousness of the message. This is a crucial step in the detection of smishing within the messages. Also, leet

words and misspelled words are selected as a smishing heuristic in our system to enhance the detection capability. Leet words and misspelled words are often used by the attackers to befool the users. On the other hand, a legitimate message sent by a legitimate organization never contains leet words and misspelled words. We have extracted the five most efficient features from the text messages to enable the machine learning classification using a limited number of features. Also, we are analyzing the authenticity of the URL to avoid high false-positive rates, i.e., the number of legitimate messages categorized as smishing, in our system.

The motivation of this paper is to develop an efficient and less complex smishing detection model using some less laborious steps. At the same time, we are checking the authenticity of the URL along with analyzing the contents of the message. To address smishing, the most popular mobile security issue, we have performed a case study on the Paytm smishing scam and developed a system that detects smishing. In this system, we have used Backpropagation Algorithm to classify the text message. The system checks for URLs and extracts few features from the message. Text pre-processing is done, and nouns are selected from the message to form a signature that is provided to the search engine. The domain name of the URL is also extracted to create the signature. The system consists of two phases, Domain Checking Phase and SMS Classification Phase. A model of the proposed system is also implemented to evaluate the results. We propose the following components in this paper:

- A system to detect smishing.
- A case study is performed on the Paytm smishing scam.
- A two-step Domain Checking technique is developed to check the legitimacy of the URL.
- The Backpropagation Algorithm is studied and implemented using real-time datasets.
- A model of the system is implemented, and its evaluation using SMS datasets is presented.

The rest of the part of this paper is arranged as follows: Background study of this research work is explained in Sect. 2. The novel system suggested in this paper is elaborated in Sect. 3. Challenges faced while doing this work and comparison are discussed in Sect. 4. Experimental details and results are shown in Sect. 5, and finally, the conclusion is given in Sect. 6.

2 Background study

Background study about smishing revealed that smishing-related studies fall into two categories, studies on smishing and studies on phishing. We have also included research papers which implemented neural network and Backpropagation Algorithm for SMS Classification. Moreover, the research works are also categorized based on their detection techniques used. For Smishing and phishing detection, researchers have used various methods like content-based methods, flow-based methods, heuristic-based methods, and signature-based filtering methods. The smishing-related research works are also categorized based on the machine learning algorithms used for detection of smishing. Traditional machine learning algorithms are used in some research works, whereas neural network is implemented in other research works for SMS Classification.

2.1 Smishing detection

Some researchers have conducted *studies to bring Awareness* among users and researchers about smishing. These research works include a study about SMS Phishing attacker strategies, detection techniques, approaches, and various policies that can be followed to mitigate smishing. In the research paper [1] titled 'SMS Phishing and Mitigation Approaches,' the author focused on various strategies followed by researchers to detect and prevent smishing. This paper also discusses various approaches that should be adopted by users to prevent smishing attacks. It also elaborated on various techniques used by the researchers to detect smishing messages. Their work is focused on bringing awareness among users about smishing attacks.

The heuristic-based classification system is also used by researchers to classify the SMS using machine learning algorithms. In these studies, researchers extract some set of features from the dataset and classify the SMS based on these features. Ankit et. al. [6] proposed a heuristic-based algorithm to detect smishing messages. The authors selected 10 features out of smishing messages and classified the messages based on these selected features with the help of classification algorithms. Authors have experimented with their approach on a manually processed dataset. Their evaluation results showed an accuracy of 98.74%. SmiDCA [7], and a smishing detection technique proposed by Sonowal et al. showed an accuracy of 96.4%. The authors selected 39 features of smishing messages and then, used dimensionality reduction to reduce the number of features and to select the 20 best features. They have also used the correlation algorithm and showed good accuracy in their experiment. The authors [8] proposed a smishing detection system using machine learning algorithms in a recent work proposed. They have used Support Vector Machine (SVM), Random Forest (RF), and Logistic Regression (LR) for the classification part of their system and reported a 92.7% F1-score using Support Vector Machine. In the latest research work[9] proposed for

smishing detection, the authors used four correlation algorithms, namely Pearson rank correlation, Spearman's rank correlation, Kendall rank correlation, and Point biserial rank correlation to rank the features. Finally, the best feature set is selected for smishing detection and reported an accuracy of 98.40%.

Flow-based approaches are used to build a system for smishing detection which is arranged in layers. These layers are depicted in flowcharts in the best way available. These systems are built to extract content-based features from the SMS to facilitate their classification. In another research paper titled S-detector [10], the authors followed a flow-based approach to detect smishing messages. The system flow is decided based on whether a URL exists in the SMS or not. If the URL exists, APK download criteria are analyzed. They have also done the keyword classification using classification algorithms in case the URL does not exist in SMS. In a study to recognize smishing messages [11], the authors presented a content-based approach. Whether the user is prompted to fill a form for revealing his credentials is inspected in this approach. Also, whether an executable file is downloaded into the mobile device on clicking the link is inspected. Finally, the message is categorized as smishing and legitimate. Diksha et al. [12] suggested a smishing detection model named as 'smishing classifier' in which the system analyzes the content of the message and segregates smishing keywords using the Naïve Bayes Algorithm. In this work, the author checks the existence of any link in the SMS and it also scrutinizes the phone number of the SMS sender. In addition to this, the appearance of an interface to fill credentials and APK downloading to devices is also evaluated in this model. A research work presented by Ankit et al. [13] presented a smishing detection system using some set of rules. The authors selected nine rules from the smishing dataset. Then, they have used three algorithms, namely Decision Tree, RIPPER, and PRISM to classify the messages. The implementation results of the technique presented good accuracy. In the latest work, the author proposed a research paper [14] titled 'Smishing Detector' in which they followed a content-based approach having four modules. They analyzed the content of SMS in the SMS content analyzer module and then inspected the authenticity of the URL through URL filter, Source code analyzer, and APK download detector modules. They have shown an accuracy of 96.2% in the experiment conducted.

2.2 Phishing detection

Some researchers have conducted *studies to bring Awareness* among users and researchers about Phishing attacks. These research works include a study about phishing attacker strategies, detection techniques, approaches, and various policies that can be followed to mitigate Phishing. Research work presented in paper [15] "Phishing-challenges and solutions," the author discussed phishing challenges and their solutions. The author discussed the phishing solution in three steps, Prevent Phishing, Detect Phishing, and stake holder Training. The author also discussed various challenges involved in the detection of phishing. Diksha et al. proposed a research paper [16] focusing on phishing and its related areas. This paper discusses all categories of phishing and the various techniques used by researchers in this field to detect phishing. Various categories of phishing like smishing and vishing are elaborated in this paper. It also discusses detection techniques and approaches relevant to the phishing area. Anna et al. proposed a paper [17] in which they discussed various mobile security attacks such as phishing, voice phishing, and smishing. They focused on smishing and its problems and preventive measures in this paper. Foozy et al. proposed a paper [18] discussing various phishing identification techniques on a smartphone device. They have detailed phishing attacks like Bluetooth phishing, smishing, and vishing. This paper also differentiated among several phishing detection techniques. Hossain et al. proposed a research paper [19] in which the author focused on phishing attacks and their categories. In addition to this, phishing mitigation techniques and best policies to avoid phishing attacks on android devices are also discussed in this paper. Their work is focused on bringing more awareness among users about phishing attacks. In a recent study [20] conducted by researchers, an overview of Artificial Intelligence (AI) techniques used for phishing detection is discussed. This paper conducts a comparison of different phishing detection techniques and highlights the pros and cons of these techniques. This paper also discusses the challenges of phishing detection and its future directions.

Some researchers have used *filtering techniques* to detect phishing. The phishing dataset is filtered into different categories during each stage of filtering. PhiDMA [21] proposed by Sonowal et al. is a phishing detection system using a multi-filter approach: whitelist filter to check the URL in the whitelist, URL feature filter to check the malicious features of the URL, lexical signature filter to create a signature from the words present in webpage, string matching filter for matching the URLs obtained from a search engine, and accessibility score filter to compare the accessibility score. This paper also presented an interface for visually impaired persons.

The heuristic-based classification system is also used by researchers to classify the phishing URL using machine learning algorithms. In these studies, researchers extract some set of features from the dataset and classify the URL based on these features. An anti-phishing model [22]

proposed by authors suggested a phishing detection system in which they categorized phishing features into various categories like features related to address bar, features related to HTML, and javascript, features related to the domain name of URL and listed each one of them. They have also shown the popularity of their features in detecting phishing websites. Zhang et al. [23] proposed a technique that selected some features from the phishing dataset and then, categorized them as hosted features and lexical features. A classification algorithm was also used to identify the phishing URLs based on these features. Their method has shown good accuracy while examining it on various datasets. Authors [24] proposed a phishing detection approach called CANTINA + in which they have used eight features, HTML DOM, and search engines. In addition to this, a near-duplicate filter is presented which identifies highly similar phish. A login form filter that categorizes web pages without a login form in it as legitimate is also included in this paper.

The latest research work proposed by Ankit et al. [25] implements a search engine-based technique and some heuristics to detect a phishing attack. They have extracted their search query from the URL and used this query to search for a legitimate website. They have also extracted some heuristics from the source code of the URL like login form, input tag, etc. Their experiments showed 99.05% accuracy on phishing data consisting of 2000 phishing URLs and 2000 legitimate URLs. In a recent work [26], authors proposed a phishing detection system that utilizes eight machine learning algorithms to detect phishing. They have used three different datasets for the evaluation of their work. The final evaluation and comparison of their work has shown notable performance in the detection of phishing URLs. Lokesh et al. [27] proposed a phishing detection system that uses the wrapper-based method for feature extraction, and some effective features are extracted. For the final classification, they have used various machine learning algorithms like Random Forest, K nearest neighbors, Decision Tree, and SVM. A comparative study of their work has shown that their system is efficient in the detection of phishing URLs. In a recent work [28] presented by Saravanan et al., feature extraction is done using the phishing dataset obtained from PhishTank. This feature vector is further forwarded to the module *GenFea* which conducts feature reduction, and the best feature set is obtained. This feature set is again treated by the module *PhiDec* to identify the maliciousness of the website. The evaluation results of the system proved that their system is efficient in phishing detection.

The signature-based detection system is used by some researchers for phishing classification. In this detection technique, a signature is formed using few features of the URL which is used to check the legitimacy of the webpage. Cantina [29] is another method proposed for detecting phishing URLs in which they have formed a signature using words collected from the website with the help of the TF-IDF algorithm. They have also selected a few heuristics of the URL to get better accuracy. Finally, they have presented four experiments to show the comparison of the features they have used.

Flow-based approaches are used to build a system for phishing detection which is arranged in layers. These layers are depicted in flowcharts in the best way available. Wu et al. presented MobiPhish [30], an anti-phishing model in which they have presented two interfaces, WebPhish and AppPhish for detecting phishing in webpages and mobile apps, respectively. They have used techniques like matching the domain name in the whitelist, checking whether IP address is used instead of the domain name, checking for sensitive terms in the text included in the web page.

2.3 Implementation of neural network for SMS classification

SMS Classification includes both spam detection and smishing detection. Both of these studies include a similar set of features and detection techniques. Spam messages are messages sent by companies and business organizations for advertisement purposes and thereby increasing their revenue. On the other hand, smishing messages are spam messages which contain a malicious URL, attacker's phone number, or email id which redirect the user to malicious websites. Hence, smishing messages are a part of spam messages, but they are more vulnerable to mobile devices. The latest work published by the author Ankit et al. [31] proposed a system for identifying spam and smishing messages from the same dataset. First, they have segregated the spam messages and legitimate messages based on some features. Then, they have segregated the smishing messages out of the spam messages based on a new set of features. They have reported an accuracy of 96% on the neural network. In a research paper authored by Ghourabi et al. [32], the author proposed a model for the detection of spam messages. They have implemented the system combining two methods, CNN and LSTM. The combination of these two methods gave them an accuracy of 98.37%. The proposed model detects both Arabic and English spam messages. In the comparative study with other machine learning algorithms, it is shown that the CNN-LSTM model gives better accuracy. In a research work proposed by Roy et al. [33], the author implemented a deep learning-based model using CNN and LSTM along with traditional classifiers such as Naive Bayes and Random Forest. This model is intended to differentiate between spam and non-spam text messages. They have implemented CNN in three

phases, creating a word matrix, identifying the features from messages, and classifying them into predefined classes. They have shown that the CNN-based model gave an accuracy of 99.44% on tenfold cross-validation. In a novel system proposed by Sheikhi et al. [34], the model is presented in two stages: feature extraction and decision making. Some features of spam messages are extracted in the first stage; then, messages are classified in the decision-making stage. For this classification, they have used an averaged neural network. The avNNet (Averaged Neural Network) used for the classification consists of one hidden layer. The experimental analysis of the system has shown an accuracy of 98.8%. In a research paper [35] proposed by author Nandita et al., the Multilayer Perceptron model is used for the classification of spam emails. The Backpropagation Algorithm is used to train the algorithm and for calculating its gradient. They have changed the learning rate in every iteration for achieving faster convergence. The four models implemented by them have shown an average accuracy of 95%.

3 Research work

In this research work, we have focused on proposing a novel yet less complex technique for the detection of smishing messages. We have conducted a case study on various smishing messages to get better clarity about the attacker's strategy of targeting the users. To get an enhanced view of the scenario of smishing, we have conducted a detailed study about Paytm smishing scam. Paytm smishing scam is initiated by the attacker sending smishing messages to Paytm users. These attackers masquerading as Paytm official informs the user that their KYC has been expired and needs to be renewed else their account will get blocked in 24 h. Hence, this message prompts the Paytm user to contact the fraudster immediately through a link or a phone number provided in the message. When the user contacts the attacker, the credentials and sensitive financial details of the user are asked through a user interface provided which looks similar to the Paytm website. These details are visible to the attacker in plain text after they have been entered and submitted by the user. The fraudster would immediately access the user's Paytm account or connected bank account using these details. Sometimes, the attacker asks the user to download an application through which the attacker can view and access the user's device remotely. Through this method, they extract sensitive details like card number, CVV (Card Verification Value), OTP (One Time Password), etc., to access the credit card or debit card of the Paytm user. In this way, attackers can do fraudulent financial transactions and in turn, Paytm users lose their hard-earned money.

Figure 1 shows a Paytm scam message in which attackers used leet words to defraud the users. In this, some leet words, i.e., numerals looking like alphabets are used to make an illusion that it is a genuine message and at the same time they are not copying the brand name of a genuine organization. Leet words are often used by attackers to bypass word filtering and to prevent smishing messages from being discovered via keyword search. Mobile users, often in a hurry, do not notice such minor differences in brand names or URL domain names, and in turn, they click on the fake links or call on the mobile number provided.

We have noticed two observations by conducting this case study. One is attackers often use mobile numbers in the smishing messages, but these types of messages are often harmless until and unless the user contacts the attacker on the phone number provided. Hence, it is strongly advised not to contact the phone numbers and email ids provided in unknown messages. Even if the user urgently needs to contact the sender considering it as a genuine message, then the user should search for genuine phone numbers and email ids from legitimate websites. Text messages containing the phone number and email id are extracted and processed separately by our system. Our system performs the content analysis of the messages containing the phone number to assess its maliciousness. We have extracted and analyzed these features in SMS Classification Phase. The second observation is that attackers use leet words to befool the users. In the message shown in Fig. 1, they are using a numeral '2' in the brand name Paytm. Mobile users often in a hurry do not notice these minor differences in the brand names. Attackers also used the numeral '0' instead of 'O' in the word 'blocked,' which indicates a leet word. Leet words are often used by attackers to give a genuine effect to malicious texts, URLs,

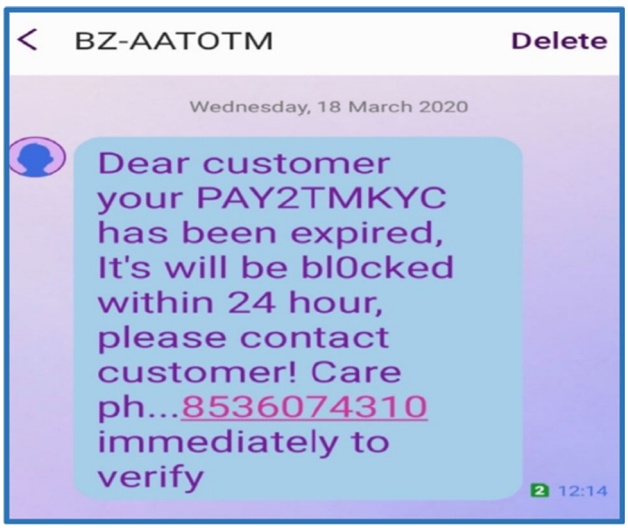

Fig. 1 A smishing message showing Paytm scam

and messages. If we feed these leet words into Google search engine, we often get zero search results. Instead, if we put genuine brand names in the Google search engine, we get the genuine website of the legitimate brand in the top search results. Hence, zero search results in Google are counted as smishing by our system. Sometimes, the Paytm smishing message contains a URL. A Paytm smishing message containing a URL to defraud the user is shown in Fig. 2. When the Paytm user clicks on the link, his/her sensitive financial details are asked through a user interface provided which looks similar to the Paytm website. The details entered and submitted by the user on this website are visible to the fraudster in plain text. To detect this type of fraudulent activity, we have processed the messages in the Domain Checking Phase which inspects the genuineness of the URL present in the message.

Smishers are exploiting the Covid-19 situation to commit fraud by using scam text messages imitating health departments, banks, and other trusted organizations. It is always advised to contact the departments on their phone number or email that is given on their official website. Attackers are trying to take advantage of people's panic and fear in the face of the COVID pandemic. A smishing message showing the Covid-19 scam [36] is shown in Fig. 3. This particular phishing SMS is prompting people to click on a link to know about new symptoms and test locations. Instead, the link leads users to a malicious website that sneaks their sensitive information such as credit card information.

Covid-19-related smishing messages often include a URL that redirects the user to fake websites. They prompt the user to provide their sensitive data in the form provided on the website. Hence, our system carefully analyzes the maliciousness of the URL provided in the SMS to assess its authenticity and thereby predicting the final results. We have inspected the messages in Domain Checking Phase in our system to deal with this issue.

Figure 4 depicts the overall working of the proposed model. The system is arranged in two phases, Domain Checking Phase and SMS Classification Phase. Domain Checking Phase inspects messages containing URL, and SMS Classification Phase focuses on messages containing email id and phone number. DsmishSMS Algorithm is presented in Algorithm 1. First, the text pre-processing is done. Text pre-processing means bringing the text into a form that is analyzable for the piece of work. Text pre-processing is a crucial step for Natural Language Processing. It includes tokenization, lowercasing the text, stop word removal, stemming, and lemmatization.

Algorithm 1 DsmishSMS Algorithm

Inputs
Incoming text messages(current_sms)
Patterns (phone_pattern, URL_pattern, and email_pattern)
Outputs
Status(Smishing SMS, Legitimate SMS)

```
1:Start
2: while current_sms.moveToNext do
3:     for Split current_sms into tokens by taking individual words as Tokenizer do
4:     call pre-processing function
5:         if current_sms matches URL_pattern then
6:         Go To Domain Checking
7:             else if current_sms matches phone_pattern or email_pattern then
8:             Go To SMS Classification
9:             end if
10:          else
11:        return legitimate SMS
12:        end if
13:   End for
14:end while
```

3.1 Domain checking phase

If a URL is detected in the message, it is inspected in Domain Checking Phase. When a URL is detected in the SMS, the system extracts the domain name of the URL. It also extracts all nouns present in the message. It forms a

signature by using all nouns and the domain name. This signature is provided to the Google search engine. The top 5 search results of the Google search engine are selected and compared with the current URL. If the domain name matches, the message is declared as legitimate else the messages are transferred for the second-level domain checking. Domain Checking Algorithm is presented in Algorithm 2.

Algorithm 2 Domain Checking Algorithm-1

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

1:**Start**
2: while current_sms.moveToNext do
3: D<- Domain name of current_URL
4: N<- Nouns from current_sms
5: Signature← D + N
6: Command← **Signature** ***inurl*:D** ***allintitle*:**N
7: Call G <- https://google.com
8: Feed command-> G(command)
9: **for** s=1 to 5
10: d_s= domain name of s^{th} URL in search
11: **If** (d_s=D)
12: return legitimate SMS
13: **else**
14: s=s+1
15: end **if**
16: end **for**
17:end **while**

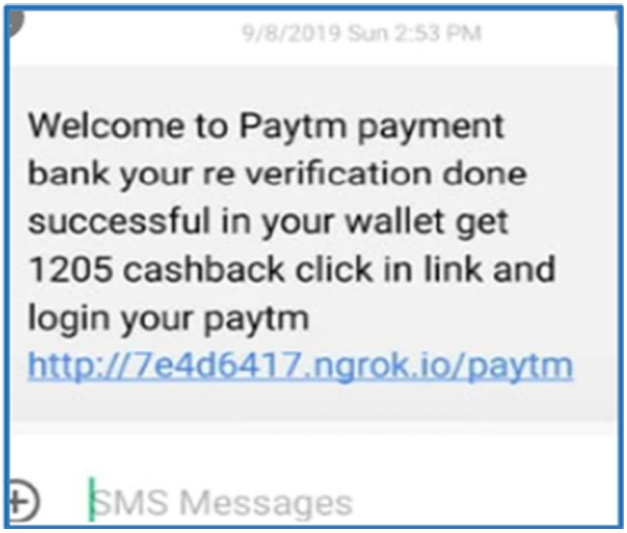

Fig. 2 A URL-based smishing message showing Paytm scam

The system extracts the value of the following features from the text message:

Noun Fraudsters send text messages masquerading as authentic, genuine organizations. Hence, these messages contain the brand names of genuine organizations. Nouns in the message represent the brand names of genuine organizations. Hence, Nouns in the message can be included in the signature to find the website of the genuine brands. These nouns and domain name will help us in finding the genuine website of the authentic organization.

Our system used the Natural Language Tool Kit (NLTK) which is a python package to extract nouns used in the text message. NLTK provides POS (parts of speech) tagging, which separates the words into parts of speech like nouns and verbs. Here, we extract the nouns from the message using the POS tagging of the NLTK package.

Domain Name The system extracts the domain name of the URL contained in the text message. If the URL is short, it is converted to a long URL and then, the domain name is extracted. Later, this domain name is used with nouns to form the signature.

Signature A signature is formed using all the nouns extracted from the text message and the domain name of the URL. This signature is fed to the Google search engine.

Google advanced search operators are special commands that can be used to perform effective retrieval. We used these retrieval operators to develop our search command. The search command used in our system is specified below:

Signature *inurl*: "domain-name" *allintitle*: nouns

In the above command, *inurl* and *allintitle* are the commands. Signature, domain name, and nouns are the values extracted from the text message.

inurl This command will return all results containing the specified word in the URL. Through this command, search results can be reduced drastically.

allintitle This command will find pages with all of the specified words in the title tag.

This command facilitates the detection of the website with the exact match of the domain name in the URL and the nouns provided in the title of the website. Most of the legitimate websites have their brand name in the title of the website.

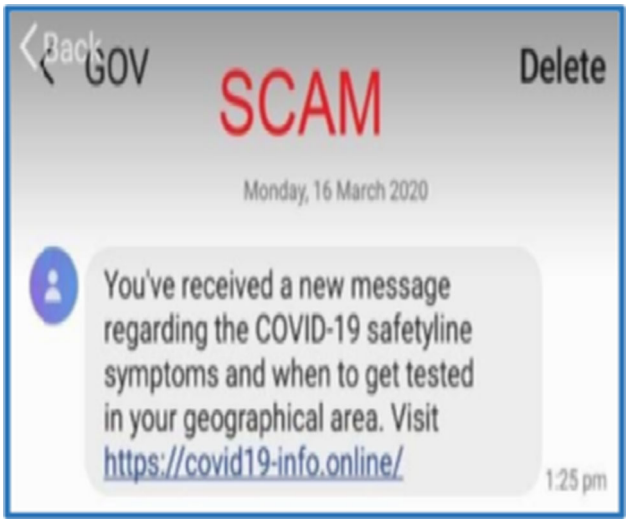

Fig. 3 A text message showing Covid-19 scam

The system matches the domain name of each search results with the domain name of the URL obtained from the text message. We repeat this step for the top 5 search results. If the string matching gives favorable results, the system declares the message as legitimate SMS else the message is forwarded for the second-level domain checking.

Phelps and Wilensky [37] proposed the idea of providing a few selected terms, called lexical signature to search engine for finding URLs. Their empirical studies suggested that it is sufficient to use about five terms to create a lexical signature for identifying a web page distinctively, out of millions of pages on the web. Hence, we are comparing the domain name with the top 5 search results. If zero search results are encountered, then the system predicts the message as smishing.

If nouns are not found in the message, we feed only the domain name to the search engine. This domain name might give zero search results in case if it is an IP address. If the Google search engine gives zero results, then we predict the message as smishing because an IP address instead of a domain name indicates smishing. If the message contains misspelled words or leet words in brand names, feeding these words into a search engine might also result in zero results. A genuine message never contains misspelled words, especially in their brand names. This leads us to assume that misspelled nouns indicate a smishing message. Domain names fed into the search engine should give genuine websites in search results if it is a domain name of a genuine website. If it is giving zero results, again it indicates a smishing message.

Our technique assumes that the Google search engine gives the majority of legitimate websites in its top search results and legitimate sites are ranked higher than phishing sites. The age of the domain of malicious websites is very short. Phishers keep on changing their domain name very frequently. The average life span of a phishing website is 4.5 days [38]. Due to the short lifespan of fake websites and lack of links pointing to them leading to have a low Google page rank, they are not displayed in top Google results.

To meet with the privacy concerns of feeding Google with the contents received from an SMS, we assure that the system is only comparing the domain names in search results but not opening the links received in search results. This way, the system protects the user from any malicious file downloading while doing search engine comparisons.

In the second-level domain checking, the system first extracts the source code of the URL included in the message. Then, all the URLs included in the source code for redirections are extracted. The domain names of these URLs are compared with the domain name of the URL contained in the message. If the domain name matches, the message is declared as legitimate else the message is transferred to SMS Classification Phase. The second-level Domain Checking Algorithm is given in Algorithm 3.

Algorithm 3 Domain Checking Algorithm-2

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

```
1:Start
2:while current_sms.moveToNext do
3:D<- Domain name of current_URL
4:Extract source code C of current_URL
5:P<-Extract all URL from C
6:p<-Domain name of each URL in P
7:      For each p in C
8:          If D matches p
9:          Print legitimate SMS
10:         Else p=p+1
11:         end if
12:     end for
13:end while
```

Second-level domain checking is a crucial part of the system. This level of checking determines whether the message needs to be transferred to the Backpropagation

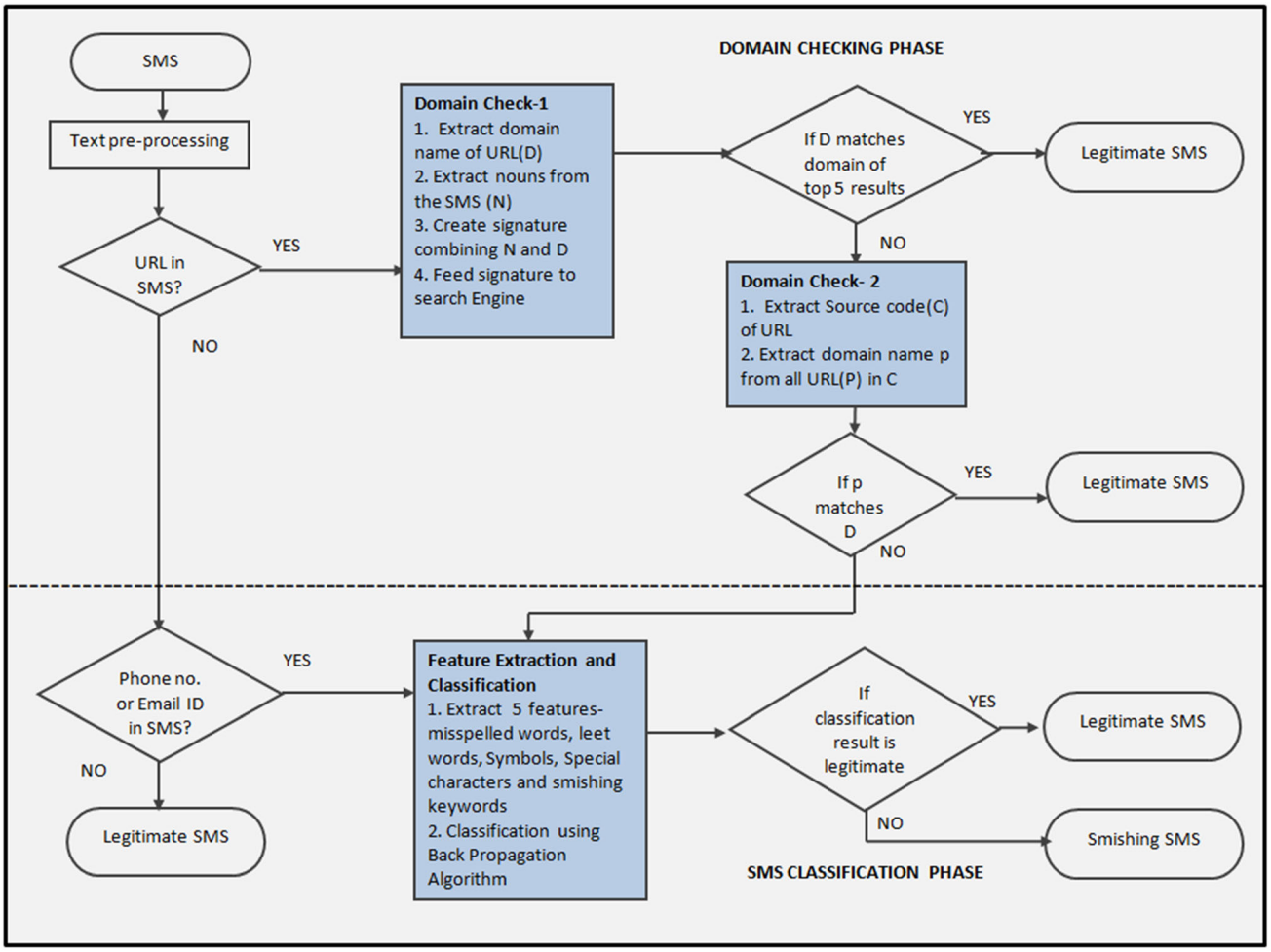

Fig. 4 Architecture of the proposed System

Algorithm for classification or it can be declared legitimate. Phishing is an activity in which attackers copy the source code of the genuine site to give a genuine effect to their malicious websites. For stealing the sensitive credentials of the user, attackers insert redirections in the source code, so that the information entered by the user gets saved in the database created by the attacker in their malicious websites. These redirections often have a different domain name provided the attacker wants to steal the user's sensitive data and needs access to it through a fake website having a different domain name. To capture this type of maliciousness, we are matching the domain name included in the source code in the second-level domain checking. The point to be noted here is that the system is accessing the source code of the URL without invoking the website to avoid any malicious file downloading during this step.

3.2 SMS classification phase

Text messages containing the phone number and email id will be processed in this phase as shown in Fig. 4. The system inspects the text message contents in this module. The values of five heuristics from the text message are extracted, namely the presence of misspelled words, leet words, symbols, special characters, and smishing keywords. Based on the values of the extracted heuristics, a feature vector is built which will be provided to the classification algorithms. A python script is created to extract and compare all the heuristics related to the text.

The presence of misspelled words, leet words, symbols, special characters, and smishing keywords in the SMS text is the five features extracted in this phase, which are elaborated as follows:

Misspelled words Misspelled words in text messages indicate smishing because attackers often send misspelled text messages to the users. Misspelled words are the result of typing errors or intentional misspellings. Attackers often

use misspelled words to befool the users. Legitimate organizations never use misspelled words in their official text message sent to customers. System checks for misspelled words if any exist. If misspelled words exist, the system returns 0 else returns 1. We found 62.12% of smishing data with misspelled words in it.

Leet Words Leet words are words in which alphabets are replaced by numerals and symbols which look similar to the alphabets. Leet words are often used by attackers to give a genuine effect to malicious texts, URLs, and messages. Hackers are using leet words in a text message to bypass word filtering and to prevent smishing messages from being discovered via keyword search. If misspelled words exist, the system returns 0 else returns 1. We found 74.22% of smishing data with leet words in it.

Symbols Attackers use mathematical symbols like +, %, -, /, ^, %, etc., to lure the victims. Using these symbols also helps them to create leet words. System checks if any symbols are used in the text message. If symbols are used, the system returns 0 else returns 1. We were able to find 51.59% of smishing data with symbols inserted in it.

Special Characters The special characters like !, $, &, #, and ~ are used by the attackers in smishing messages. The character "$" denotes currency in the smishing message claiming to award the user, and character "!" is used with smishing words like "CONGRATULATIONS!", "WINNER!", etc., to attract the victims. By reviewing our dataset, we found 26.20% of smishing data with special characters in them.

Smishing keywords The system deals with the words which are regularly used by the attackers to attract the victims. They try to attract victims by offering discount coupons, gifts, prizes, etc. We found 83.02% of smishing data with the following 20 smishing keywords in it. The following commonly used smishing keywords are extracted by the system to deal with the smishing messages:

Award, claim, gift, voucher, blocked, won, prize, winner, activate, please, account ,card, refund, due, congratulations, cash, urgent, free, happy, join

The system extracts the values of features mentioned above and classifies the messages based on that. We have used the Backpropagation Algorithm and three traditional classification algorithms, namely Decision Tree, Random Forest, and Naive Bayes to predict the results and check the effectiveness of the heuristics selected.

An Artificial Neural Network (ANN) is a machine learning technique inspired by the human brain. Hence, ANN is modeled like the human nervous system. Neural Network is comprised of neurons connected in layers. The inputs and expected outputs are fed into the system. An activation function is used to activate the neurons and calculate the output. The inputs are mapped to the outputs with the help of an activation function. This process is termed as Forward Propagation. Every neuron connection in the network has a corresponding weight. Backpropagation Algorithm (BPA) is used for training the algorithm to reach the expected outputs. An error function is used to calculate the difference between actual output and expected output. Actual output is the output given by the algorithm, and expected output is the output provided to the system. BPA adjusts the weights and biases of the network for bringing the error function to the minima. This process is termed as Backward Propagation. In BPA, both the inputs and outputs are supplied to the algorithm and the mapping function between the inputs and outputs is learned. Hyperparameters like the number of neurons, connections, number of hidden layers, etc., can be optimally set for achieving good accuracy.

Figure 5 shows the architecture of the backpropagation network. Following parameters are used in implementing the neuron-based network:

- *Activation function* is used to activate the layers and predict the output
- *Prediction error* is the difference between target output and prediction output
- *Error function* is used to calculate the error
- *Optimization method* is used to minimize error

Our dataset is trained and tested using the Backpropagation Algorithm. The best accuracy given by the algorithm is given in Sect. 5 of this paper. Also, a comparison chart depicting the results of BPA with all three traditional algorithms is shown in Fig. 7.

Messages containing URL is processed in Domain Checking Phase. On the other hand, messages containing phone number and email id are processed in SMS Classification Phase. If none of them is detected, messages are classified as legitimate. Messages which do not contain a URL, email id, or phone number are considered legitimate because such messages cannot cause any harm to the user even if it contains malicious keywords. In the proposed system, messages are segregated in an earlier stage, avoiding the processing of the same messages to different phases. Messages with email id and phone number are processed by SMS Classification Phase, and messages containing URL are processed by Domain Checking Phase. Features extracted in SMS Classification Phase are only

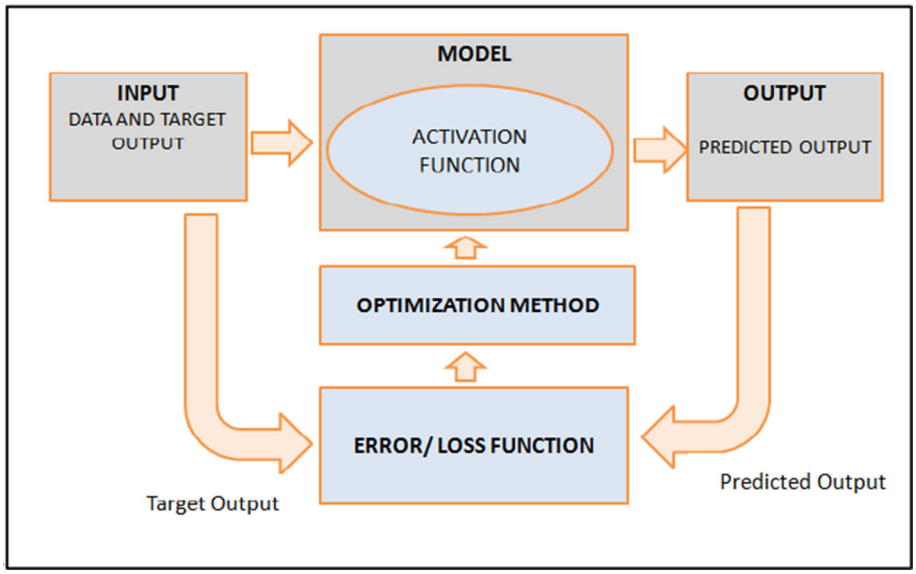

Fig. 5 Architecture of the backpropagation network

about messages having an email id or phone number in it, and thus, this system highly reduces complexity in the analysis of the text messages.

4 Challenges and comparison

This section elaborates on the various challenges we have faced during the implementation of this research work. This section also presents a comparison of the proposed model with smishing detection models proposed by other researchers.

As smishing is a fraud activity conducted through short text messages, detection of smishing involves various challenges. In the case of smishing, we have short text messages which are often in short forms or symbolic forms. We assume that attackers often use nouns in the text message to identify the legitimate brand which they are masquerading. Hence, we are extracting the nouns in the message to identify the legitimate website. Also, the URL incorporated in the message is short. Our system first converts the short URL to a long URL; then, it is processed to extract the actual domain name. Forming a signature from the short text message is a strenuous task because it includes abbreviations, text shortcuts, and acronyms used by the text sender. It also contains leet words used by the attacker which causes difficulty for the systems to recognize the actual word intended by the attacker. Leet words often lead to zero search results in Google search engines. Zero search results indicate smishing by our system. Due to the minimum amount of information shared by the attacker in the messages, we are left with a minimum set of features for classification. In this system, we are selecting the nouns used in the text message to form a signature. The nouns can be detected in Google search results if the text message is sent by a genuine organization. Short URL is converted to long URL before examining it for maliciousness. It is difficult to analyze the URL within the small display of mobile phones even if it is converted into a long URL. Abbreviated form of text messages and short URL makes it difficult to analyze the maliciousness of the text message. Hence, we analyze the presence of leet words and smishing keywords included in the text message to check its maliciousness.

Some of the other challenges faced during the evaluation of this system are discussed below:

- What if the domain name of the current URL does not match with the domain name of the results given by the search engine? If the domain name is not matching with the top 5 domain names, our system declares the message as smishing. If a message contains misspelled brand names or leet words, it might result in zero search results. A genuine message never contains misspelled words, especially in their brand names. This indicates that misspelled nouns mean a smishing message. Domain names fed into the search engine should give genuine results if it is an authenticated message.
- What if the domain name is an IP address and not identifiable by its name for comparison? If the domain name of the URL is an IP address, it indicates smishing as IP address is often used by the attackers due to their frequently changing domain names. Legitimate brands prefer to use domain names in their website URLs for effortless recognition.
- How to make a signature from the message? What are the criteria to create it from a text message? Most of the attackers are using leet words, symbols, numerals, smishing keywords, etc., in the text message which

cannot be used in a signature. Every attacker masquerades as a genuine brand; thus, they use the brand name in the text message. This brand name is often a noun or abbreviation. Hence, we have extracted the nouns from the message to form a signature.

- How to handle misspelled words in the text message? It could drastically affect search results. Misspelled words are not used in the signature to avoid false search results. To identify the smishing keywords used by the attacker, we have used some selected keywords often used by the attackers.
- What if there are zero nouns in the message? In case of the absence of nouns in the text message, the system will use the domain name of the URL to form the signature.

4.1 Comparison

A comparative analysis of our proposed system with other smishing detection systems is shown in Table 1. This comparison is made based on the security measures and techniques implemented in the system.

It is evident from the comparison table that we have used some novel techniques for the detection of smishing in our system. Search engine domain matching and source code domain matching are the two phases of domain matching conducted in our system. Search engine domain matching is used as a novel technique for the detection of smishing in our system. Also, in the case of phishing detection, signature for search engine matching can be formed by extracting website contents. But in the case of smishing, SMS has abbreviations and idioms which makes it difficult to form the signature. The existence of URL, phone number, and email id in the message is used as a heuristic in other systems too, but we have categorized the messages based on these features at the beginning of the implementation phase and thereby reducing the complexity of the system. We have extracted the best five features from our real-time dataset and thereby improving the performance and reducing the complexity. Moreover, the leet word feature used in our system is a novel heuristic used for the detection of SMS Phishing.

Search engine domain matching is a technique in which we compare the domain names of top search results with the current URL available in SMS. This technique is used for the first time in the detection of smishing. This technique helps us in finding the specific website of the legitimate brand which the attacker is masquerading. Hence, this improves our true negative rate. We have successfully developed a system to create a signature from the text messages to perform the domain checking.

Moreover, we have conducted a detailed study about the leet words used by attackers in smishing and why they are used. In this study, we have observed that leet words are different from misspelled words, and leet words are intentionally created by the attacker to give a genuine effect to the smishing messages. These leet words also help the attacker to bypass word filtering and to prevent smishing messages from being discovered via keyword search. Leet word identification is a novel technique used in our system. It provides a new direction to future research in smishing detection systems.

5 Evaluation and results

This section elaborates on the evaluation of the proposed system using SMS datasets and the evaluation results therein. The prototype of this model is built in python language. The proposed system is built into two phases, namely Domain Checking Phase and SMS Classification Phase. The two phases are finally merged to obtain a final working model of the approach. The final model is experimented with using a collection of text messages.

Table 1 Comparison of the proposed model with other proposed systems

Techniques and details	Feature based [6]	Rule based [13]	SmiDCA [7]	Smishing Detector [14]	S-detector [10]	Proposed system
Search engine domain matching	NO	NO	NO	NO	NO	YES
Source code domain matching	NO	NO	NO	YES	NO	YES
Existence of URL	YES	YES	YES	YES	YES	YES
Existence of phone number and email id in the message	YES	YES	YES	YES	NO	YES
Smishing keywords	YES	YES	YES	YES	YES	YES
Misspelled words	NO	NO	YES	NO	NO	YES
Leet words	NO	NO	NO	NO	NO	YES
Symbols	YES	YES	NO	NO	NO	YES
Special characters	NO	NO	YES	NO	NO	YES

Text messages dataset is collected from the paper Almeida [39], a contribution to the study of spam messages. It includes 5574 text messages out of which 4827 are legitimate and 747 are spam messages. According to our study, the smishing dataset is not publicly available yet. But we have observed that smishing SMS is a part of spam SMS. Hence, some smishing messages were manually extracted from the spam messages. Also, some smishing SMS was extracted from pinterest.com [40] which were added to the final dataset. So, our final dataset contains 5858 text messages of which 538 are smishing SMS and 5320 are legitimate SMS.

5.1 Evaluation metrics

The metrics estimate the performance of the system in terms of the percentage of correct instances detected and the number of misclassifications it makes. For evaluating the performance of the smishing detection system, we measured Accuracy (ACC), F1-score, precision, recall, Area Under the ROC Curve (AUC), and time complexity.

Following evaluation metrics are used in our system:

- True Positive (TP): True Positive denotes the number of smishing messages identified as smishing by the system.
- False Positive (FP): False Positive denotes the number of smishing messages identified as legitimate by the system.
- False Negative (FN): False Negative denotes the number of legitimate messages identified as smishing by the system.
- True Negative (TN): True Negative denotes the number of legitimate messages identified as legitimate by the system.
- Accuracy: Accuracy is evaluated as the proportion of True Positive and True Negative over the total number of classifications as depicted by the formula below:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}$$

- Precision: Precision is calculated by the formula:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- Recall: Recall is calculated as:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- F1-Score: It is calculated as the harmonic mean of precision and recall.

$$F1\text{Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- Area Under the Curve (AUC): The area under the curve (AUC) of a Receiver Operating Characteristic (ROC) curve is used as a metric for the performance evaluation of a binary classifier system. For plotting the ROC curve, the values of True-Positive Rate (TPR) are shown on the vertical axis and False-Positive Rate (FPR) are shown on the horizontal axis of the curve.
- Time Complexity is measured as the time consumed by the CPU to execute the code. It depends on the software as well as the hardware we are using to execute it. This value helps us to determine that the computational complexity of the proposed system is not high, and it converges to a solution within a reasonable time.

5.2 Results

The feature set used in this system is carefully selected to make sure that the smishing messages are predicted accurately. The frequency of each feature in our dataset is calculated to determine the importance of features used in our system. This frequency is calculated based on the existence of these features in 538 smishing messages available in our dataset. The presence of these features is tested by building a python code for the extraction of features. The frequency of each heuristic used in our system is depicted in Fig. 6. The results have shown that the 'smishing keywords' feature is most useful in detecting smishing SMS and it exists in 83.02% of smishing SMS, followed by 'leet words' which exist in 74.22% of smishing messages. The least important feature used in our system is special characters which are found in 26.20% of smishing data.

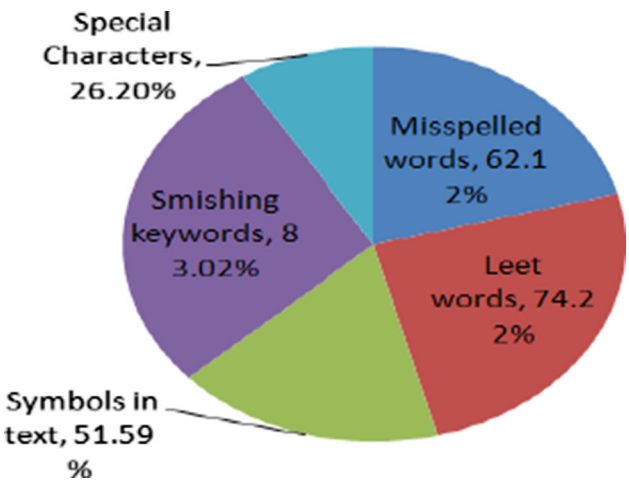

Fig. 6 Frequency of each heuristic analyzed

The above five features selected by our system have shown good accuracy while classifying the messages using Backpropagation Algorithm. The values returned by the above heuristics have been passed to machine learning algorithms. We have evaluated our approach using three machine learning algorithms, namely Decision Tree, Random Forest, and Naive Bayes. The prediction results of each algorithm based on the feature values have been recorded. The prediction result of each algorithm is shown in Fig. 7. It shows that the Random Forest Algorithm (RF) gave a decent accuracy of 97.85%, while the Backpropagation Algorithm (BPA) gave the best result with an accuracy of 97.93%. The naïve Bayes algorithm (NB) also presented a good performance with an accuracy of 97.76%. The precision and recall of BPA and RF are almost the same with a value of 84% and 94%, respectively. Thus, BPA outperformed RF in the evaluation of our system and Naïve Bayes outperformed Decision Tree in the machine learning experiment.

We have tried different values for hyperparameters like the number of hidden nodes, the number of epochs, and learning rate for achieving the best accuracy and achieved the final accuracy of 97.93% using the Backpropagation Algorithm. Hyperparameters used in achieving this accuracy are shown in Table 2. We have tried values ranging from 1 to 12 for the number of hidden nodes and achieved the best accuracy for 10 hidden nodes. The maximum number of iterations was set to 100 for evaluating the model, but time complexity was high on the execution of the system. Hence, the best accuracy achieved in a remarkable time complexity was noted for which the number of epochs was 10. The sigmoid activation function was also tried for BPA, but we have achieved the best accuracy using ReLU Function for the BPA implementation of our system. Different values of learning rate were tested ranging from 0.1 to 1.0. The resultant graph and response to the error rate are depicted in Fig. 8. The figure shows that the error rate is rising in response to an increase in the learning rate.

Table 2 Hyperparameters used in Backpropagation Algorithm

Hyperparameters	Values
No. of hidden nodes	10
No. of Epochs	10
Activation function	ReLU
Solver	Adam
Learning rate	0.01

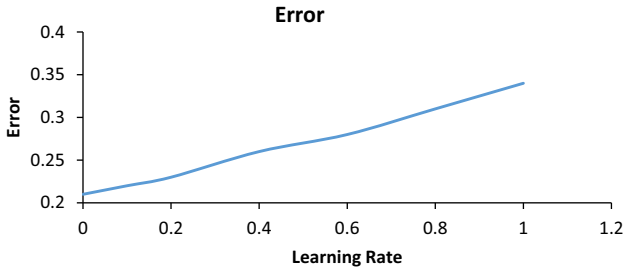

Fig. 8 Behavior of error to change in learning rate

The prediction error is calculated by finding the difference between the actual output and the target output. A Mean Squared Error (MSE) function is used to calculate the error as shown in Eq. (1), where a_k is the actual output, t_k is the target output, and E is the prediction error.

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

The behavior of Error corresponding to learning rate is shown in Fig. 8. The error is gradually increasing when the

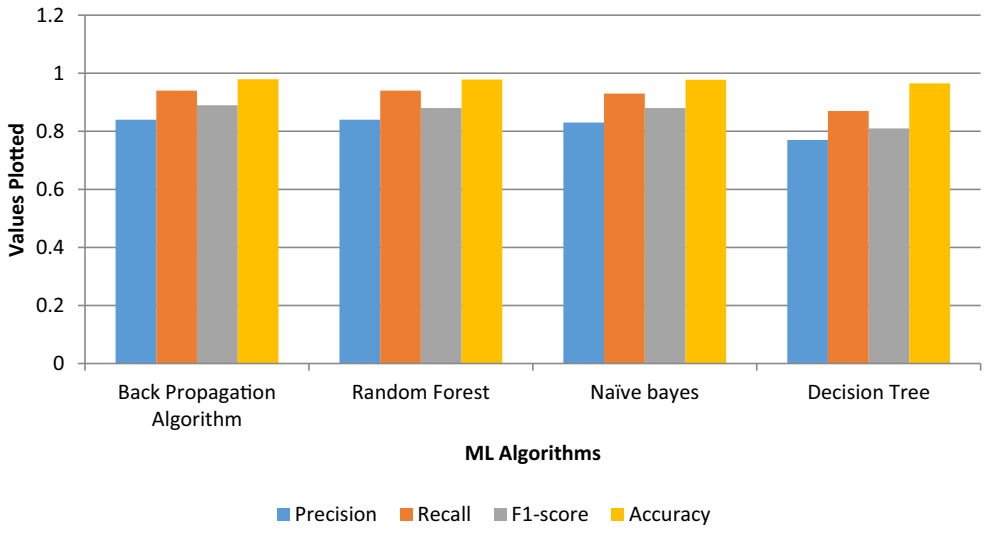

Fig. 7 Performance of algorithms on our system

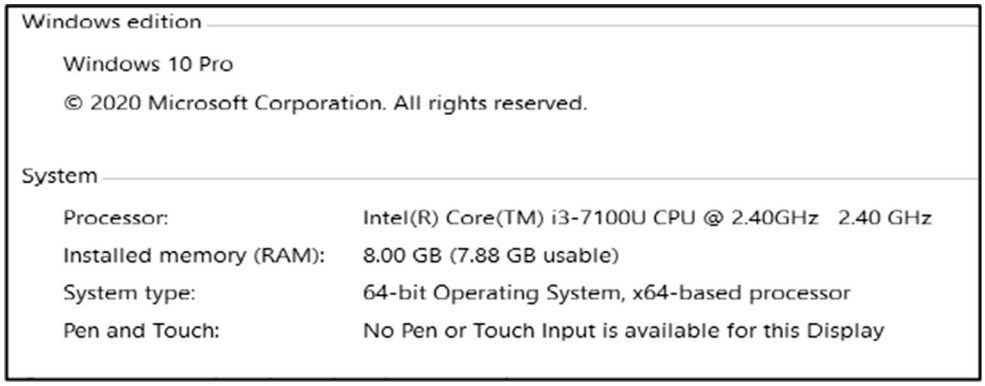

Fig. 9 The software and hardware configurations used for evaluation of the system

Table 3 Evaluation results of the system

Algorithm	Accuracy	AUC	Execution time in seconds
Backpropagation Algorithm	97.93	0.988	33.32
Random Forest	97.85	0.985	20.41
Naive Bayes	97.76	0.983	17.24
Decision Tree	96.48	0.974	16.32

Table 4 Performance of the proposed model on Backpropagation Algorithm

	Confusion matrix	
	Smishing messages	Legitimate messages
Classified as smishing	TP = 509	FP = 92
Classified as legitimate	FN = 29	TN = 5228

Table 5 Outcome of the proposed model after cross-validation

Iterations	Accuracy
Iteration 1	97.65
Iteration 2	97.97
Iteration 3	98.29
Iteration 4	97.52
Iteration 5	98.25
Average	97.93

learning rate is incremented from 0.1 to 1. This means that the algorithm achieves convergence at a slower pace when the learning rate is set to low values, but a higher value of learning rate causes fast convergence. But the fast convergence has a risk of escaping the local minimum. Hence, the error rate rises with increase in the learning rate.

Execution time of the system is estimated using Begin Time() and End Time() functions in the python environment. Hardware and software configurations used for the execution of the system are shown in Fig. 9. A 64-bit operating system with 8 GB RAM is used for the evaluation of the system. The time taken by the Backpropagation Algorithm is 33.32 s, whereas random Forest took 20.14 s to evaluate the system. Hence, the time complexity is high in the case of the Backpropagation Algorithm, but it gave us a higher accuracy in comparison with Random Forest. Therefore, BPA has shown the best performance with an accuracy of 97.93% and execution time of 33.32 s. AUC values are plotted using the ROC curve which is shown in Table 3. AUC value 0.988 is achieved for BPA and 0.985 for Random Forest. Hence, it is evident from the AUC values that our system is persistent in the performance for smishing detection.

The performance of the Backpropagation Algorithm is depicted in Table 4. It shows that this approach can identify smishing messages with accuracy of 97.93%. The false-positive rate is very less which shows that very few ham messages are classified as smishing. Twenty-nine smishing messages out of 538 smishing messages in our dataset were wrongly classified as legitimate by the system. This shows that the false-negative rate is very low, and the system is efficient in detecting smishing messages. Only 92 legitimate messages out of 5320 legitimate messages were classified as smishing which concludes 1.7% FNR (False Negative Rate). Hence, our system is efficient in identifying legitimate messages.

We have experimented our approach using fivefold cross-validation in which we break up the dataset into five

equal subsets, i.e., 80% data for training and 20% data for testing. One subset is used as the testing set and the rest four subsets are used as the training set. Iterations are repeated until all the subsets are used as the testing set. In fivefold cross-validation, a final accuracy of 97.93% is achieved. The result of the cross-validation experiment is shown in Table 5.

The above results show that this system is efficient in the detection of smishing messages. We have tested the performance of our system using evaluation metrics like Accuracy, Precision, recall, and F1-Score. A confusion matrix depicting the performance of the Backpropagation Algorithm is also presented. System is also evaluated using AUC (Area Under the Curve). Thus, the results are persistent when evaluating with both the evaluation metrics.

6 Conclusion

This paper focused on developing a system for the detection of smishing. We successfully designed and evaluated a smishing detection system comprising of two phases, namely Domain Checking Phase and SMS Classification Phase. Each phase focused on different aspects of the SMS to determine its maliciousness. We mainly focused on scrutinizing the authenticity of the URL in SMS while reducing the complexity of the system. We have also discussed the Paytm smishing scam and its counterparts in this paper.

Finally, a model of the system has been developed and experimented on SMS data comprising of 5858 messages. We have selected the most efficient features for the identification of smishing messages. We have implemented our system using the Backpropagation Approach and obtained the final accuracy of 97.93%. The existing smishing detection approaches focused on SMS contents only, but our method focused on monitoring the authenticity of the URL in the detection of smishing messages.

A comparison of our system with other smishing detection systems has been presented. It displayed that some novel techniques have been used in this research work to detect smishing. Google search engine has been used to aid in the detection of legitimate websites.

In this study, the phone number and email id included in a message are not checked for their maliciousness. In a few cases, messages might appear legitimate, but the phone number and email id included in the message might belong to an attacker. Blacklists of phone number and email id are not publicly available yet. These blacklists can be developed and utilized through a publicly available database which might be used for future directions of this work. Also, different techniques can be developed for creating the signature for search engine domain check conducted in this work. Creating a signature from the minimum amount of information shared by the attacker in a message is a difficult task. But, researchers are motivated to find out the best technique for creating a signature using the minimum available information and thereby taking it as a challenge. Other deep learning algorithms can be used for the classification and comparison part of this work. Results of the Backpropagation Algorithms can be compared with other deep learning models.

Declarations

Conflict of interest The authors have no conflicts of interest to declare that are relevant to the content of this article. All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

References

1. S Mishra, D Soni, (2019) SMS phishing and mitigation approaches. In: Twelfth International Conference on Contemporary Computing (IC3), Noida, India pp. 1–5, doi: https://doi.org/10.1109/IC3.2019.8844920
2. Arab M, Sohrabi MK (2017) Proposing a new clustering method to detect phishing websites. Turk J Electr Eng Comput Sci 25(6):4757–4767
3. Statista , "Number of smartphone users worldwide from 2016 to 2021". (2020) URL https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/, accessed on 2020
4. CallHub , "6 reasons why sms is more effective than email marketing - callhub." (2020) URL https://callhub.io/6-reasons-sms-effective-email-marketing/, accessed on 2020
5. Delany SJ, Buckley M, Greene D (2012) Sms spam filtering: methods and data. Expert Syst Appl 39(10):9899–9908
6. Jain A, Gupta BB (2019) Feature based approach for detection of smishing messages in the mobile environment. J Inf Technol Res 12:17–35. https://doi.org/10.4018/JITR.2019040102
7. Sonowal G, Kuppusamy KS (2018) SmiDCA: an anti-smishing model with machine learning approach. Comput J 61(8):1143–1157
8. C. Balim and E. S. Gunal, (2019) Automatic detection of smishing attacks by machine learning methods. In: 1st International Informatics and Software Engineering Conference (UBMYK), Ankara, Turkey, pp. 1–3, doi: https://doi.org/10.1109/UBMYK48245.2019.8965429
9. Sonowal G (2020) Detecting phishing SMS based on multiple correlation algorithms. SN Comput Sci 1(6):361. https://doi.org/10.1007/s42979-020-00377-8
10. Joo JW, Moon SY, Singh S, Park JH (2017) S-detector: an enhanced security model for detecting smishing attack for mobile computing. Telecommun Syst 66:1–10
11. Mishra S, Soni D (2019) A content-based approach for detecting smishing in mobile environment. Suscom. https://doi.org/10.2139/ssrn.3356256
12. D Goel, AK Jain, (2018) Smishing-classifier: a novel framework for detection of smishing attack in mobile environment. In: NGCT, CCIS 828, pp. 502–512

13. Jain Ak, Gupta BB (2018) Rule based framework for detection of smishing messages in mobile environment. Procedia Comput Sci 125:617–623
14. Mishra S, Soni D (2020) Smishing detector: a security model to detect smishing through sms content analysis and url behavior analysis. Futur Gener Comput Syst. https://doi.org/10.1016/j.future.2020.03.021
15. Vayansky I, Kumar S (2018) Phishing – challenges and solutions. Comput Fraud Secur. https://doi.org/10.1016/S1361-3723(18)30007-1
16. Goel D, Jain AK (2017) Mobile phishing attacks and defence mechanisms: state of art and open research challenges. Comput Secur. https://doi.org/10.1016/j.cose.2017.12.006
17. Kang A, Lee JD, Kang WM, Barolli L, Park JH (2014) Security considerations for smart phone smishing attacks. Springer, Berlin
18. Foozy CFM, Ahmad R, Abdollah MF (2013) Phishing detection taxonomy for mobile device. Int J Comput Sci 10(1):338–344
19. Shahriar H, Klintic T, Clincy V (2015) Mobile phishing attacks and mitigation techniques. J Inf Secur 06:206–212. https://doi.org/10.4236/jis.2015.63021
20. Basit A, Zafar M, Liu X et al (2021) A comprehensive survey of AI-enabled phishing attacks detection techniques. Telecommun Syst 76:139–154. https://doi.org/10.1007/s11235-020-00733-2
21. Sonowal G, Kuppusamy K (2017) Phidma—a phishing detection model with multi-filter approach. J King Saud Univ Comput Inf Sci 29:1–15
22. Mohammad RM, Thabtah F, McCluskey L (2014) Intelligent rule-based phishing websites classification. IET Inf Secur 8:153–160
23. J Zhang, Y Wang, (2012) A real-time automatic detection of phishing URLs. In: 2nd International Conference on Computer Science and Network Technology, ICCSNT, IEEE, pp. 1212–1216
24. Xiang G, Hong J, Rosé C, Cranor L (2011) CANTINA+: a feature-rich machine learning framework for detecting phishing web sites. ACM Trans Inf Syst Secur. https://doi.org/10.1145/20195992019606
25. Gupta BB, Ankit J (2020) Phishing attack detection using a search engine and heuristics-based technique. J Inf Technol Res 13:94–109. https://doi.org/10.4018/JITR.2020040106
26. M. Korkmaz, O. K. Sahingoz and B. Diri, (2020) Detection of phishing websites by using machine learning-based URL analysis. In: 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kharagpur, India, pp. 1–7, doi: https://doi.org/10.1109/ICCCNT49239.2020.9225561.
27. Harinahalli Lokesh G, BoreGowda G (2021) Phishing website detection based on effective machine learning approach. J Cyber Sec Tech 5(1):1–14. https://doi.org/10.1080/23742917.2020.1813396
28. Saravanan P, Subramanian S (2020) A framework for detecting phishing websites using GA based feature selection and ART-MAP based website classification. Procedia Comput Sci 171:1083–1092. https://doi.org/10.1016/j.procs.2020.04.116
29. Y Zhang, J Hong, L Cranor, (2007) Cantina: a content-based approach to detecting phishing web sites. In: Proceedings of the 16th International Conference on World Wide Web pp. 639–648, doi: https://doi.org/10.1145/1242572.1242659
30. L. Wu, X. Du, J. Wu, (2014) MobiFish: a lightweight antiphishing scheme for mobile phones. In: 23rd International Conference on Computer Communication and Networks, ICCCN, pp. 1–8
31. Ankit J (2019) A novel approach to detect spam and smishing SMS using machine learning techniques. Int J E-Services Mob Appl. https://doi.org/10.4018/IJESMA.2020010102
32. Ghourabi A, Mahmood MA, Alzubi QM (2020) A hybrid CNN-LSTM model for SMS spam detection in Arabic and english messages. Future Internet 12:156
33. Roy PK, Singh JP, Banerjee S (2020) Deep learning to filter SMS spam. Future Gener Comput Syst 102:524–533
34. Sheikhi S, Kheirabadi MT, Bazzazi A (2020) An effective model for SMS spam detection using content-based features and averaged neural network. Int J Eng (IJE) IJE Trans B Appl 33(2):221–228
35. Sesha RA, Avadhani PS, C Nandita., (2019) A content-based spam e-mail filtering approach using multilayer percepton neural networks. Int J Eng Trends Technol 41:44–45. https://doi.org/10.14445/22315381/IJETT-V41P210
36. MessageMedia, "6 COVID-19 (Coronavirus) SMS scams to look out for", (2020) URL https://messagemedia.com/au/blog/covid-19-coronavirus-sms-scams-to-look-out-for/, accessed on 2020
37. Phelps TA, Wilensky R (2000) Robust hyperlinks and locations. D-Lib Mag 6:7–8
38. Wu L, Du X, Wu J (2016) Effective defense schemes for phishing attacks on mobile computing platforms. IEEE Trans Veh Technol 65(8):6678–6691. https://doi.org/10.1109/TVT.2015.2472993
39. TA Almeida, JMG Hidalgo, A Yamakami, (2011) Contributions to the study of SMS spam filtering: new collection and results. In: 11th ACM Symposium on Document Engineering, pp. 259–262
40. Pinterest, "Smishing Dataset", November 20 2018, Retrieved from https://in.pinterest.com/seceduau/smishing-dataset/?lp=true.