

SMSPROTECT: An automatic smishing detection mobile application

Oluwatobi Noah Akande^{a,b,*}, Oluwadara Gbenle^a, Oluwakemi Christiana Abikoye^c,
Rasheed Gbenga Jimoh^c, Hakeem Babalola Akande^d, Abdullateef O. Balogun^{c,e},
Anuoluwapo Fatokun^f

^a Landmark University SDG 09: (9) Industry, Innovation and Infrastructure, Nigeria

^b Computer Science Department, Landmark University, Kwara State, Nigeria

^c Computer Science Department, University of Ilorin, Kwara State, Nigeria

^d Department of Telecommunication, Faculty of Communication and Information Sciences University of Ilorin, Nigeria

^e Department of Computer and Information Sciences, Universiti Teknologi PETRONAS, 32610Bandar Seri Iskandar, Perak, Malaysia

^f Eastern Illinois University, IL, United States

Received 25 March 2021; received in revised form 14 December 2021; accepted 23 May 2022

Available online 31 May 2022

Abstract

Short Messaging Service (SMS) has grown to become the most widely used feature in mobile devices. The technological advancements that birthed other alternative messaging applications have not been able to phase out the use of the SMS. However, hackers have been exploiting this SMS feature to perpetrate smishing acts. Existing research has focused on how spam SMS could be detected and separated from ham messages but have not really done much at preventing the act of smishing. Therefore, this research presents a mobile application that used a rule-based SMS service to detect and prevent smishing attacks. Specifically, the developed SMS service allows the developed SMS mobile application to intercept incoming SMS to a smartphone. The intercepted messages were then forwarded through an Application Programming Interface (API) to the rule-based machine learning model. The model uses the carefully selected rules to analyze the retrieved message and asserts if it is a spam or ham. The result of the analysis is then forwarded to the mobile application through the API. However, the final decision to retain or discard the spam or ham depends on the user after receiving notification from the user.

© 2022 The Author(s). Published by Elsevier B.V. on behalf of The Korean Institute of Communications and Information Sciences. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

Keywords: Data security; Smishing; Spam SMS detection; Rule based learning; Mobile application development

1. Introduction

Deception remains the major tool used by hackers to coin the gullible into divulging sensitive information about their accounts or other highly confidential information. This could be via vishing acts — voice prompts that have been customized to mimic an individual, pharming which involves installing malicious code into the personal computer or server of the targets, thereby misdirecting them to fraudulent web sites without their knowledge or consent. Attacks could also be perpetrated via phishing acts — a fraudulent attempt aimed at obtaining sensitive information such as usernames, passwords and credit card details from targeted individuals. Smishing was coined from two words: SMS and Phishing. Smishing

is a mobile phishing attack in which hackers send SMS that encourages users to visit fake websites, apps or links. Smishing could contain links to malicious applications or websites; this is not only targeted at siphoning funds from users but to also get information such as their contact numbers, mobile phone versions, photos and other information stored on their mobile devices. With mobile phones becoming one of the most embraced products of Information and Communication Technology (ICT) advancement in the world, it has also been greatly employed by hackers to perpetrate their evil acts. With an alarming 5.22 billion unique mobile phone users in the world and a growth rate of 93 million in the past 12 months, mobile phones have come to stay as an inevitable part of our daily life that we cannot do without. Furthermore, the advent of smartphones with diverse features have contributed a great deal to the growth of mobile phones' usage. A survey carried out in [1] revealed that smartphones usage is growing at the rate of 7% annually as 1 million new smartphones are been

* Corresponding author at: Computer Science Department, Landmark University, Kwara State, Nigeria.

E-mail address: akande.noah@lmu.edu.ng (O.N. Akande).

Peer review under responsibility of The Korean Institute of Communications and Information Sciences (KICS).

acquired daily. In Africa, smartphone adoption grew from 198 million in 2016 to 498 million in 2020 with Nigeria expected to produce the highest new smartphone connections to the tune of 145 million ahead of south Africa and Kenya by 2025 [2]. Of all the functionalities of feature and smartphones, SMS is the most widely used with 9.57 and 20.92 billion messages sent and received in year 2018 [3]. The number of received SMS doubled the number of sent messages; the source of received SMS could be traceable to SMS sent for promotional and advertisement purposes. The popularity of SMS could be attributed to its ease of use, relatively cheap cost when compared to calls and the fact that it does not require internet connectivity for delivery. In addition, when compared to emails, SMS has an open rate of 98% while email has an open rate of 45%. Also, SMS has a response rate of 45% compared to email with 6% [4]. This is the reason why most organizations prefer to reach out to their customers through SMS and hackers have taken advantage of this.

Furthermore, most smartphone users easily fall victim of smishing attacks because of the small display size of mobile phones which may not allow users to critically examine the contents of malicious links. Ignorance and inadequate sensitization about smishing attacks also expose smartphone users to smishing attacks while bad security habits of entering security credentials on phishing websites have opened many users to attacks. According to Verizon's mobile security report for 2020, 85% of phishing on mobile devices does not involve emails [5]. Productivity applications (apps) are responsible for 10%, gaming apps are responsible for 11%, social media apps are responsible for 16%, 17% occurs via messaging while 31% are from news and travel apps. Most times, smartphone users are forced to grant malicious applications access to their messages, contacts, photos and other sensitive sections of their phones as a requirement for completing the installation process. Through this act, hackers easily get access to the contents of users' phones.

Nonetheless, SMS has been widely used as a tool to gain access to users' devices and harvest their personal information. A survey revealed that 68% of mobile phone users are affected by SMS Spam, with teenagers being the most affected [3]. Deadly and malicious attacks such as ransomware mobile phishing, SMS spam, mobile multimedia application threat, vishing etc are all traceable to the SMS [6]. The presence of self-answering links, URL links, phone numbers and/or Email address are common pointers to smishing SMS [7]. Recently, several SMS that contains malicious links could be forwarded to redirect victims to fake websites while some could contain links that automatically install malwares on users' phone and steal saved financial information or contacts. Worrisomely, certain text may ask users to send STOP/CANCEL to terminate a transaction; this could be used to ascertain a phone number that is active. Moreover, the recent adoption of SMS for two step authentication process has endeared people to receiving codes in SMS. Certain cardless ATM implementations have been introduced by some banks in developed countries; such enables customers to receive cash via their mobile phones. Stolen customer information could be used to add new phone

numbers to customers' account thereby taking total control of their accounts. Within two weeks, \$68,000 was stolen from 17 ATMs in Illinois, Michigan, and Ohio while \$3000 was withdrawn from an individual at once using this tactic [8,9]. While some believe installing anti-malwares could be a way out, how to know if the anti-malware itself is not a malware remain an issue of concern.

Therefore, a spam filtering system is not sufficient to curtail the spread of spam SMS but coming up with ways to legally prevent its spread from the source [9]. Several techniques have been proposed to achieve this, such involve blacklisting already known smishing sources. However, new sources and methods are been developed daily which makes it difficult to keep track of malicious URLs. The idea of shortening web addresses using tiny URL, bit.ly etc have been greatly explored by these hackers. This has made it difficult for users to ascertain the authenticity of a web URL until such is clicked. Heuristic-based methods have also been widely used in detecting smishing messages. However, they do produce many false positive results as they use the content of the text messages to classify a message as malicious or not. For instance, not all URLs are malicious and valid SMS could contain account numbers or passwords. Machine learning techniques have also been massively explored to prevent smishing SMS. They are content based techniques that employ machine learning algorithms and spam SMS datasets to classify SMS into spam or ham. Classifiers such as SVM, Naïve Bayes, KNN, regression etc.; evolutionary algorithms such as artificial immune system; deep learning techniques such as Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) have been widely used for spam SMS classification. However, the accuracy of these systems heavily depends on the features extracted from the SMS and the datasets used to train the classification systems [10]. Nevertheless, the proposed research employs rule-based techniques to complement machine learning techniques for the detection and prevention of spam messages.

While email has advanced methods in place for email spams filtering, SMS still needs more standard measures to be put in place against smishing attacks [11,12]. Though many mobile applications have been developed to detect spam messages in mobile phones, yet, hackers still find a way round them [13]. Authors in [9] believed that putting a spam filtering system in place is not sufficient to curtail the spread of spam SMS but coming up with ways to legally prevent its spread from the source. This research also took a step further from proposing classification systems for determining if a SMS is spam or not but presents a mobile application that could be integrated into mobile phones to identify spam SMS from source and allow the user to determine if identified spam SMS should be deleted or ignored. RIPPER and C4.5 PART classifiers were used in the rule-based spam detection model. Then, a mobile service was developed to intercept incoming SMS to a smartphone. The intercepted SMS is then forwarded to the rule-based spam detection model for analysis and classification; this result will be forwarded to the mobile application via an Application Program Interface (API). Finally, the user is allowed to take the final decision to either delete or retain a SMS.

2. Related works

Several researches aimed at proffering solution to the menace of smishing have been carried out in the past decades [14–17]. Such include a clustering-based technique for the detection of spam SMS and thread proposed in [18]. Naïve Bayes (NB), Support Vector Machine (SVM), Latent Dirichlet Allocation (LDA) and Non-negative Matrix Factorization (NMF) binary classifiers were used to classify the SMS into spam or ham. Afterwards, K-means or NMF were used to create a cluster of non-spam SMS for easy identification of threads. SVM achieved the best accuracy among all the classifiers used. Furthermore, a spam detection technique that employs Frequent Pattern (FP) growth and Naïve Bayes algorithms were proposed in [19]. The FP growth algorithm which is an improved version of the Apriori method was used to mine frequent patterns that are present in the SMS while Naive Bayes classifier was used for the classification task. High accuracy of 98.51% was achieved when FP growth and Naive Bayes classifier were combined compared to 98.48% achieved without FP growth. This showed that FP growth improved the accuracy of Naive Bayes classifier. Authors in [16] also proposed a NB based smishing text detection model. The proposed model used a monitor to track SMS activities on smartphones by recording the logs and timestamps of incoming SMS. Afterwards, a text preprocessing module was used to identify sentence boundaries and auto word spacing. A morphological analysis module was then used to analyze text messages in unit morpheme. It also has a URL Checker that screens URLs present in the text messages. Finally, a Naïve Bayesian classifier was used to detect if noun words extracted are frequently used smishing text messages. The proposed model assumes that all SMS with a URL download link is a smishing message; this may not be always true. Authors in [9] also proposed a Bayesian based spam detection model. The emphasis of the research was to create a digital evidence that could be used in SMS forensics so as to affirm if spam SMS were actually sent by an individual. When the performance of the proposed model was evaluated, a recall, precision, accuracy and fallout values of 96.80%, 98.98%, 97.90% and 1.00% were achieved.

Back Propagation Neural Network was employed for smishing SMS detection in [13]. Fourteen different features were carefully selected to detect if a message is spam or ham. Though all the features can be used to effectively identify a spam message, presence of contact numbers and money values could independently recognize illegitimate message with 94% accuracy. Among the classifiers used, back propagation yielded the highest accuracy while SVM recorded the highest true positive rate. CNN and LSTM based deep learning technique was proposed for SMS filtering in [10]. It was observed that the strength of CNN is in the extraction of hidden features present in the input SMS; however, CNN fails to capture long sequences present in SMS. CNN was also discovered to suffer from model overfitting. However, a regularization parameter (dropout) of 0.3 was used to reduce the computational overhead and the likelihood of over-fitting. An accuracy of 98.63% was achieved with the proposed CNN with

the regularization parameter while a 10-fold cross validation CNN with regularization parameter recorded an accuracy of 99.44%. The proposed LSTM model with the regularization parameter only recorded an accuracy of 96.76%. This showed that CNN still works better if a regularization parameter is added. Authors in [11] also proposed a hybrid CNN and LSTM model for SMS spam detection in Arabic and English languages. The strength of CNN in identifying common words in spam messages through its convolutional operations was explored while the inability of CNN to correlate the recent information extracted with past information was enhanced with LSTM. LSTM was believed to be good in identifying and capturing long-term dependencies and word sequences of any length. When 5574 English and 2730 Arabic SMS were used to evaluate the proposed technique, accuracies of 98.13%, 98.19% and 98.37% were achieved with CNN, LSTM and the proposed CNN-LSTM technique. It will be observed that most of the related works reviewed have proposed and employed different machine learning techniques to either classify SMS into spam or ham, none so far has demonstrated how the classification models could be used to prevent smishing acts; this is the identified gap this research intend to bridge.

3. Methodology

This research presents a mobile application that uses a rule-based SMS service to detect and prevent smishing attacks. Specifically, a mobile service was developed to intercept incoming SMS to a smartphone. The intercepted SMS is then forwarded to the rule-based machine learning model through an Application Programming Interface (API). The model uses the carefully selected rules to analyze the retrieved message and asserts if it is a spam or ham. The machine learning model then forwards the result of the analysis to the mobile application through the API. The user is then notified and allowed to decide if the spam message should be retained or discarded. The implementation stages are categorized into model development, rules formulation, model development and training using phyton-weka, evaluation of the rule-based spam detection model and mobile application development.

3.1. Data collection and pre-processing

The UCL SMS spam dataset v.1 which contains 5547 text messages (4827 ham messages and 747 Spam messages) in English language were used to train and test the developed spam detection model. However, the datasets cannot be used in its original state as it contains several abbreviations and unstructured words, therefore, the dataset was pre-processed using WEKA programming tool so as to make spam detection effective. The dataset which was gotten in CSV file format was convert to ARFF format, to enable Weka use the dataset. The preprocessing stages are:

(a) Case conversion: a SMS could contain lower- and upper-case characters. However, machine learning algorithms may treat words that are the same but with different case differently. For instance, Prize and prize may be treated as different words,

Table 1
Count vectorizer tokenization process.

Index_ no	0	1	2	3	4	5	6	7	8	9	10	11	12
Tokens	after	agent	contact	claim	once	Our	pay	Payment	prizes	To	will	you	your
SMS 1	1	1	1	1	0	1	0	1	1	1	0	0	1
SMS 2	0	1	1	0	1	1	1	0	0	0	1	2	0

therefore, the content of the SMS was converted to lower case for effective training process.

(b) Removing punctuations: This involves removing punctuation marks present in the datasets so that there will not be disparity between words followed by a punctuation mark and words without a punctuation mark.

For instance, a SMS that contains “call me after paying.” and “paying with an ATM is the best option”; “paying.” In the first SMS could be treated differently from “paying” in the second SMS because of the full-stop that follows. So, to ensure that there is no disparity among similar words, punctuations must be removed.

(c) Expansion of Abbreviations: it should not be assumed that spam SMS will be completely formal, abbreviations that arise as a result of trending slangs could also be used. Therefore, it is necessary to detect these abbreviations and replace them with the exact words. To achieve this, English language dictionary [20] and Lingo dictionary [21] were used to convert the abbreviations to their Standard English words equivalent. Therefore, abbreviations like “rotf” becomes “rolling on the floor”etc.

(d) Tokenization: a set of words is generated from the SMS when punctuations have been removed. However, tokenization helps to further break these set of words into the smallest indivisible form. Tokenization can be characters, words or sub-words. For instance, if spaces are used to break the sentence “the smarter way to claim your prizes” into tokens, then, the sentence contains seven tokens. However, “smarter” can be further broken down to “smart-er” (sub-word tokens) or “s-m-a-r-t-e-r” (character tokens). Since, standard SMS are not more than 160 characters and spam SMS are not usually more than a page, word tokenization using count vectorizer algorithm was employed for the tokenization. Count Vectorizer algorithm generates a sparse matrix where the rows of the matrix contain the content of each SMS while the column contains each unique word present in the SMS. The tokens are re-arranged in alphabetical order while the number of instances of each word that appears in the SMS being considered will be documented in each cell of the matrix as shown in Table 1. For instance,

SMS 1: “Contact our agent to claim your prizes after payment”
SMS 2: “Our agent will contact you once you pay”

After case conversion and removal of punctuation, the count vectorizer algorithm generates the tokens as illustrated in Table 1. The essence of tokenization is to generate a vocabulary of unique set of tokens that are present in the SMS. Table 1 shows that the two SMS have 13 unique words. However, these tokens will not be stored as strings but will

be assigned a unique index number. For instance, the index number of the tokens will be 0 to 13.

(e) Stemming: stemming is another preprocessing step that makes the spam detection task easier. The refined and extracted tokens serve as input to the stemming stage. However, the tokens are further reduced to their root words for easy identification as there could be many variants that arise from a root word.

For instance, a stemming algorithm will be used to reduce “profitable”, “profiting” and “profits” to the root word “profit”. There are different algorithms that could be used for stemming but this research employed Lovins stemmer for the stemming task as it is fast and works well with pluralized words which may not be regular. It works by removing suffix present in tokens and then converts the remaining words to a valid word. For instance, “profiting” becomes “profit + ing” while “winning” becomes “winn” then “win”.

(f) Stop words removal: stop words are words used in meaningful sentence construction, however, they are not needed to classify a SMS into spam or ham. Therefore, they can be removed so as to have the barest minimum words to work with. For instance, a SMS that contains the text “Contact our agent to claim your prizes after payment” becomes SMS 1: “Contact agent claim prizes payment” after stop words removal. Examples of stop words are:

“you, yours, our, we, I, them, they, his, her, is, few, some, all, then, their, ourselves, yourself, that, above, very, will, shall, can, could, should, being, because, such, no, now, me, when, where, myself, he, him, such, above, below, and, in, on, own, does, theirs, just, only, during, be, too, such, these, up, all, why etc.”

3.2. Rules formulation

Rule-based technique that uses if ...then command was used to formulate what should be looked for in respective SMS in order to classify them as either ham or spam. It searches for and extracts features that met a particular condition from the text being considered. Generally, rule-based techniques can be classified into two: direct and indirect approach. Direct method extracts the rules directly from the text under consideration. There is no need to generate a decision tree as each of the rule in a given class covers many of the tuples of that class. Examples of direct method algorithms are Ripper, CN2, Holte’s 1R etc. Ripper was employed in this research. For the indirect method, the rules are extracted from classification models such as: neural networks and decision trees. Examples of indirect method algorithms are: C4.5 and C5.0; C4.5 was used in this research. Therefore, RIPPER and C4.5 were used to build rules used for feature extraction in this research.

3.2.1. Rules formulation using RIPPER algorithm

Repeated Incremental Pruning to Produce Error Reduction (RIPPER) algorithm is a rule-based classifier that formulates its rule from the training set. Three phases are involved in the rule formulation process: growing phase, pruning phase and optimizing phase. With emphasis on SMS dataset which has two classes: SPAM or HAM, the RIPPER classifier uses a divide and conquer strategy to divide the dataset into two. The class with the highest feature is made the majority class while RIPPER tries to learn features that differentiates the two classes by growing features from the minority class to the majority class. This increases the rule's specificity and reduces its entropy; at any point in time when an increase in specificity does not lead to a reduction of the entropy the rule is pruned. The growing and pruning of the rules are iterated until the rule has been optimized to effectively identify the features that classifies the SMS into spam or ham. Minimum description length principle was used as the termination criteria. The Pseudocode for the RIPPER algorithm is adopted is provided in Appendix A.

3.2.2. Rules formulation using C4.5 algorithm

C4.5 algorithm is an extension of ID3 algorithm that is good for classifying discrete and continuous variables. It is also good for pruning tress and handling missing values after tree construction. This is the reason why it was used to complement any missing values that RIPPER algorithm could miss. So, the output of RIPPER algorithm serves as input to C4.5 algorithm. C4.5 has additional features such as managing missing values, continuous attribute categorization, decision trees pruning, rule derivation and others. C4.5 constructs a very large tree by considering all the attribute values and finalizes by pruning the decision rule. It makes use of a heuristic pruning approach based on the statistical significance of splits. C4.5 PART algorithm is a Weka variation of the C4.5 algorithm, it builds a partial C4.5 decision tree in each iteration and makes the “best” leaf into a rule. PART is a learner of laws separate from and conquer. The algorithm that produces sets of rules called “decision lists” that are set of rules planned. In addition, a new data is compared to each rule in the list and the object is assigned the first corresponding rule class. After training the dataset with

C4.5 PART model, the following are some of the rules that were extracted by the algorithm:

- 1: Create a root node N;
- 2: If (T belongs to same category C)
 - { leaf node = N;
 - Mark N as class C;
 - Return N;
 - }
- 3: For i = 1 to n
 - {Calculate Information_gain (Ai);}
- 4: ta = testing attribute;
- 5: N.ta = attribute having highest information_gain;
- 6: if (N.ta == continuous)
 - { find threshold;}
- 7: For (Each T in splitting of T)

- 8: if (T is empty)
 - {child of N is a leaf node;}
 - else
 - {
 - child of N = dtree T
 - }
- 10: calculate classification error rate of node N;
- 11: return N

3.3. Feature extraction using RIPPER and C4.5 rule-based approach algorithm

The challenge with spam SMS detection is the insufficiency of features that could be accurately used to classify them as ham or spam. A valid SMS could contain URL, phone numbers, account numbers, words like call, text, prize etc., yet, these are common features of spam SMS. Therefore, this research created dictionaries of frequently used spam and ham words, Smishing URL and domains, and Smishing contact numbers. These were used to validate URLs, domains and contact numbers seen in the SMS. The Smishing URL and domain dictionary was created from publicly available phishing URL repositories [22–24]; these are valid phishing URL repositories that are updated hourly. For instance, as at 26th of February, 2021, repository [22] contains 2,383,360 phishing domains, [23] contains 235,961 phishing domains and 485,091 phishing links while [24] contains 147,129 fake URLs that are updated daily.

So, when a URL is detected in a SMS, the URL is validated against these phishing URL repositories before it is adjudged to be a phishing URL. Likewise, a dictionary of spam and ham words were extracted from publicly available UCL SMS spam dataset [25]. The dataset contains 4,827 hams and 747 spams with 63,632 tokens for hams and 17,543 tokens for spam respectively. Spam words include: elected, lottery, prize, immediately, offer, confirm, claim, testing, vaccine, charity, free, gift, call, now, debt, relief, credit, cards, interest, wire, transfer, money, award, username, password, bonus, award etc. A spam SMS is expected to contain a high proportion of these words why hams are expected to contain fewer proportions of these. Also, a dictionary of Smishing contact number was also created from known Smishing contact number repository available in [26]. It is a repository of contact numbers that have been used for several 419 scams, fake electronics store, unpaid contractor scams, dead customer scams, cheque frauds, fake lottery scams, fake job scams and other general scams. However, contact numbers alone cannot be used to classify SMS as spam as contact numbers are easily recycled. Therefore, a combination of one or more feature can be used to convincingly detect spam SMS and this is best achieved using rule-based approach. Some of the rules that were formulated using if ... then commands are:

- i If SMS contains URL link, then check against phishing URL dictionary; if search is positive then it is a spam else ham

- ii If SMS contains contact numbers, then check against contact number dictionary; if search is positive then it is a spam else ham
- iii If number of spam words in SMS is greater than or equal to 2 then it is a spam SMS
- iv If number of ham words in SMS is greater than or equal to 4 then it is a ham SMS
- v If the SMS contains any sign like “\$”, “£”, “#”, “&” and number of spam words in SMS is greater than or equal to 2 then it is a spam SMS.
- vi If the SMS contains any mathematical symbol like +, -, <, >, / etc and number of spam words in SMS is greater than or equal to 2 then it is a spam SMS.
- vii If the number of emoticons in the SMS is greater than or equal to 2 then it is a ham
- viii If the SMS is a self-answering type, then it is a spam
- ix If the SMS contains abbreviations, then it is a ham.

3.4. Model development and training using Python-Weka

Weka is an open-source platform that provides data processing methods, multiple machine-learning algorithms and visualization tools that could be used to build and apply machine learning techniques to real-world data mining problems. However, Python-Weka-Wrapper was employed to build the rule-based RIPPER and C4.5 SMS detection model proposed in this research. It provides an interface that permits Weka libraries to be used with python programs; most importantly, it uses Javabridge to link Weka libraries with python. 10-folds cross validation was employed to test the model after training.

3.5. Performance evaluation of the developed model

To evaluate the performance of the developed model, true positive rate, true negative rate, false positive rate, false negative rate, precision and recall were used. They were calculated using the following:

(a) True Positive Rate (TPR) - This measure the percentage of spam messages that were correctly classified by the algorithm. If the total spam messages were denoted by SM and the correctly identified spam messages were denoted by CSM, then TPR was calculated in Eq. (1):

$$TPR = CSM/SM \quad (1)$$

(b) True Negative Rate (TNR)- This measure the proportion of ham that were correctly classified as ham. If the total ham messages were denoted by HM and the correctly identified ham messages were denoted by CHM, then TNR was calculated using Eq. (2):

$$TNR = CHM/HM \quad (2)$$

(c) False Positive Rate (FPR)- This compute the total number of ham that were wrongly classified. If the wrongly classified ham were denoted by WCH, then, FPR was calculated using Eq. (3):

$$FPR = WCH/HM \quad (3)$$

(d) False Negative Rate (FN) - This calculates the percentage of spam messages that were wrongly classified as ham. This was calculated using Eq. (4):

$$FNR = WCS/SM \quad (4)$$

(e) Precision — This reflects the proportion of SMS that were spam and which the classification algorithm correctly classified as spam. It was computed using Eq. (5):

$$\text{Precision} = TP/(TP + FP) \quad (5)$$

(f) Recall — This reflects the percentage of classified spam messages that were actually classified as spam. It is used to ascertain if all the spam SMS were actually recognized and classified as spam. Eq. (6) was used for the calculation:

$$\text{Recall} = TP/(TP + FN) \quad (6)$$

3.6. Mobile application development

Based on the proposed rule-based SMS spam detection model, an android based mobile application was developed to employ the model in preventing smishing attacks. The mobile application was specifically developed to perform the following tasks:

- i Intercepts incoming SMS to a mobile phone
- ii Send the SMS through an API to the rule-based SMS spam detection model.
- iii Analyze the intercepted message to assert if it is a spam or not
- iv Label the message as spam or not and alert the user via the developed mobile application

The mobile application was developed using java programming language; a mobile service was developed to intercept incoming SMS through *Okhttp*. *Okhttp* is a java application that allows developer to make Http request. The intercepted SMS is then saved in json format and forwarded to the rule-based SMS spam detection model through an API that was developed using Flask Restful. Flask is a light-weight Web Server Gateway Interface (WSGI) that serves as an interface between web applications and web servers while Flask Restful is an extension of flask that provides supports that permits programmers to easily build REST Application Programming Interface (APIs). REST determines the look and feel of the API. It stands for “Representational State Transfer”. It is a set of rules which developers follow when building their API. One of those rules states that when connection request is made to a specific URL, a piece of data called a resource should be forwarded in return. Each URL called makes a request for the resources available on a particular server linked to that address. So, the API was used to establish a communication mode between the SMS spam detection model and the mobile application developed. So, when the SMS has been analyzed, the classification result (spam or not spam) will be forwarded to the mobile app through the *Okhttp* post request. The user can then decide to either delete or retain the SMS.

Table 2
Evaluated JRip classifier.

Total Number of Instances	5574
Correctly Classified Instances	5467(98.08%)
Incorrectly Classified Instances	107(1.92%)
Kappa Statistic	0.914
Mean Absolute Error	0.0369
Root Mean Squared Error	0.1358
Relative Absolute Error	15.8813%
Root Relative Squared Error	39.8596%

Table 3
Detailed accuracies of JRip classifier.

	TP Rate	FP Rate	Precision	Recall	F-Measure
HAM	0.996	0.116	0.982	0.996	0.989
SPAM	0.884	0.004	0.971	0.884	0.925
Weighted average	0.981	0.101	0.981	0.981	0.980

Table 4
Evaluated PART classifier.

Total Number of Instances	5574
Correctly Classified Instances	5486(98.42%)
Incorrectly Classified Instances	88(1.58%)
Kappa Statistic	0.9385
Mean Absolute Error	0.0287
Root Mean Squared Error	0.1198
Relative Absolute Error	12.3617%
Root Relative Squared Error	35.1665%

4. Results and discussion

The performance of the proposed rule-based SMS spam detection models was evaluated using TP rate, FP rate, Precision, Recall, F-Measure etc. Also, the developed mobile application was simulated in real time to see how the incoming SMS were intercepted, analyzed with the proposed models and classified as spam or not spam.

4.1. Evaluating the performance of the rule based SMS spam detection models

The performance of the RIPPER classifier was carried out using TP rate, FP rate, Precision, Recall, F-Measure etc. The results obtained are shown in Tables 2 and 3:

It will be observed that the percentage of the correctly classified instances was 98.08% (5467 instances) while the percentage of the incorrectly classified instances was 1.92% (107 instances). This achieved a TP rate of 0.996 and a FP rate of 0.116 with precision and recall values 0.982 and 0.989 respectively. The incorrectly classified instances of 1.92% was as a result of some missing values; this influenced our decision to use C4.5 algorithm to capture the missing values. The results obtained when C4.5 PART algorithm was evaluated are presented in Tables 4 and 5:

It was observed that the percentage of the correctly classified instances was 98.42% (5486 instances) which was higher than 98.08% (5467 instances) achieved with RIPPER; this shows that RIPPER achieved a better accuracy. Also, the

Table 5
Detailed accuracies of PART classifier.

	TP Rate	FP Rate	Precision	Recall	F-Measure
HAM	0.995	0.884	0.987	0.995	0.991
SPAM	0.916	0.005	0.965	0.916	0.940
Weighted average	0.984	0.074	0.984	0.984	0.984

percentage of the incorrectly classified instances was 1.58% compared to 1.92% obtained when RIPPER was used. A TP rate of 0.995, FP rate of 0.084, with precision and recall values 0.987 and 0.995 respectively were also achieved with the C4.5 algorithm. This showed that the C4.5 PART algorithm used significantly improved the performance of the rule-based technique.

4.2. Evaluating the performance of the mobile application

An android emulator that comes with android studio was used to simulate the performance of the proposed mobile application in real life. As presented in Fig. 1, once a message is sent to a user, the model intercepts and forwards the message to the rule-based spam detection model for analysis. The result of the analysis is then forwarded to the mobile application where the user can see if the SMS is spam or not. A video demonstration of how the developed mobile application works is available at:

<https://www.youtube.com/watch?v=WTwANeq3hUs>

5. Conclusion

Smartphones have become a tool that makes our daily activity complete as many individuals cannot do without them. Of all the features available on a smartphone, SMS is the most widely used feature. Despite different alternative messaging platforms that have been introduced, SMS still remain the cheapest, the most widely used and the feature with the highest response rate. Therefore, hackers have seen SMS as a tool that can be used to perpetrate smishing acts. Though, several techniques have been proposed to identify and separate spam from ham, most works reported so far have not demonstrated how the proposed models can be used to actually prevent smishing acts in practical terms. Therefore, this research has illustrated how a rule-based classification technique can be used to detect and prevent smishing. RIPPER and C4.5 classification algorithms were used as the rule-based spam detection application. Afterwards, a mobile service was developed to intercept and forward incoming SMS to the rule-based spam detection application. The result of the analysis was then forwarded to the user through the developed mobile application. The user can now make the final decision whether to accept or reject the result of the analysis. The proposed mobile application can be deployed on mobile devices for smishing acts detection and prevention. Also, further works can be carried out to use more feature extraction techniques to get a better accuracy and also the model can be embedded into the mobile application to make it faster for the mobile app to detect the smishing messages and delete it from the mobile phone [27,28].

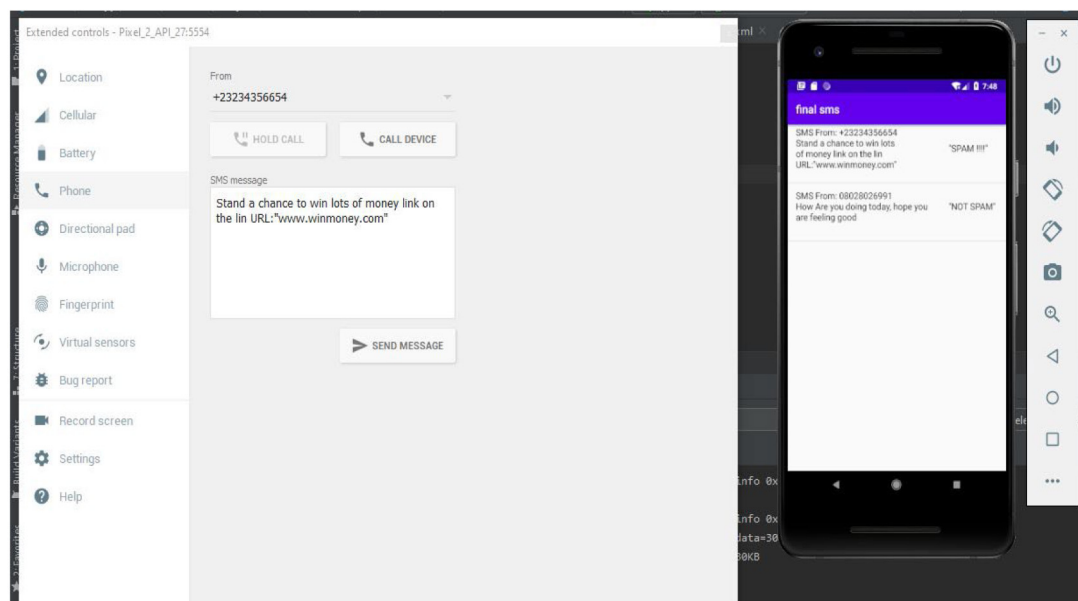


Fig. 1. Mobile Application with SMS labeled as SPAM.

Declaration of competing 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.

Acknowledgments

Authors appreciate Landmark University Centre for Innovation and Development for sponsoring the publication of this article. All authors want to appreciate the reviewer for accepting to push out our article for review. We also want to appreciate all the reviewers for their positive and constructive criticism of the work.

References

- [1] Smartphone users in Nigeria, <https://www.statista.com/statistics/467187/forecast-of-smartphone-users-in-nigeria/>.
- [2] GSMA intelligence, 2021, Retrieved January 12, 2021 from <https://www.gsmainelligence.com/>.
- [3] The real value of SMS to businesses, 2021, Retrieved January 12, 2021 from <https://www.smscomparison.co.uk/sms-gateway-uk/2018-statistics/>.
- [4] Chris. Pemberton, Tap into the marketing power of SMS, 2016, accessed on 15th February, 2021 available at: <https://www.gartner.com/en/marketing/insights/articles/tap-into-the-marketing-power-of-sms>.
- [5] Verizon mobile security index 2020 report <https://enterprise.verizon.com/resources/reports/2020-msi-report.pdf>.
- [6] A.K. Jain, B.B. Gupta, Feature based approach for detection of smishing messages in the mobile environment, 12, (2) 2019, pp. 17–35, <http://dx.doi.org/10.4018/JITR.2019040102>.
- [7] S. Mishra, D. Soni, A content-based approach for detecting smishing in mobile environment, in: International Conference on Sustainable Computing in Science, Technology & Management (SUSCOM-2019), 2019, pp. 986–993.
- [8] SMS phishing cardless ATM=profit, 2020, <https://krebsonsecurity.com/2018/11/sms-phishing-cardless-atm-profit/> accessed on 30th 2020.
- [9] Abikoye Oluwakemi Christiana, Abubakar Abdullahi, Ahmed Haruna Dokoro, Akande Noah Oluwatobi, Kayode Anthonia Aderonke, A novel technique to prevent SQL-injection and cross-site scripting attacks using knuth-morris-pratt string matching algorithm, EURASIP J. Inf. Secur. 14 (2020) 1–14, <http://dx.doi.org/10.1186/s13635-020-00113-y>.
- [10] P. Kumar, J. Prakash, S. Banerjee, Deep learning to filter SMS spam, Future Gener. Comput. Syst. 102 (2020) 524–533, <http://dx.doi.org/10.1016/j.future.2019.09.001>.
- [11] A. Ghourabi, M.A. Mahmood, A hybrid CNN-LSTM model for SMS spam detection in arabic and english messages, Future Internet 12 (156) (2020) 1–16, <http://dx.doi.org/10.3390/fi12090156>.
- [12] S. Mishra, D. Soni, Smishing detector: A security model to detect smishing through SMS content analysis and URL behavior analysis, Future Gener. Comput. Syst. 108 (2020) 803–815, <http://dx.doi.org/10.1016/j.future.2020.03.021>.
- [13] A.K. Jain, D. Goel, S. Agarwal, Y. Singh, G. Bajaj, Predicting spam messages using back propagation neural network, Wirel. Pers. Commun. 110 (2020) 403–422, <http://dx.doi.org/10.1007/s11277-019-06734-y>.
- [14] N. Bhushana Samyuel, Benjamin A. Shimray, Securing IoT device communication against network flow attacks with recursive internet-working architecture (RINA), ICT Express (ISSN: 2405-9595) 7 (1) (2021) 110–114, <http://dx.doi.org/10.1016/j.ict.2020.08.001>.
- [15] Alex. Shenfield, David. Day, Aladdin. Ayesh, Intelligent intrusion detection systems using artificial neural networks, ICT Express (ISSN: 2405-9595) 4 (2) (2018) 95–99, <http://dx.doi.org/10.1016/j.ict.2018.04.003>.
- [16] Jun. Zheng, Sai Krishna Chigurupati, M-pattern: A novel scheme for improving the security of android pattern unlock against smudge attacks, ICT Express (ISSN: 2405-9595) 5 (3) (2019) 192–195, <http://dx.doi.org/10.1016/j.ict.2018.11.003>.
- [17] Ban Mohammed Khammas, Ransomware detection using random forest technique, ICT Express 6 (4) (2020) 325–331, <http://dx.doi.org/10.1016/j.ict.2020.11.001>.
- [18] Abdullateef O. Balogun, Kayode S. Adewole, Muiz O. Raheem, Oluwatobi N. Akande, Fatima E. Usman-Hamza, Improving the Phishing Website Detection using Empirical Analysis of Function Tree and Its Variants, Heliyon, (ISSN: 2405-440) 2021, <http://dx.doi.org/10.1016/j.heliyon.2021.e07437>.
- [19] Victor E. Adeyemo, Abdullateef O. Balogun, Hamed A. Mojeed, Noah O. Akande, Kayode S. Adewole, Ensemble-based logistic model

- trees for website phishing detection, in: *Advances in Cyber Security*, Chapter: 41, Springer, 2021, http://dx.doi.org/10.1007/978-981-33-6835-4_41.
- [20] Freeling english language dictionary, 2021, Retrieved January 12, 2021 from <http://devel.cpl.upc.edu/freeling/>.
- [21] Internet & text slang dictionary, 2021, Retrieved January 12, 2021 from www.noslang.com/dictionary/.
- [22] The malwareurl team, 2021, Retrieved January 12, 2021 from <https://www.malwareurl.com/>.
- [23] Phishing domain database, 2021, Retrieved January 14, 2021 from <https://awesomeopensource.com/project/mitchellkrogza/Phishing-Database>.
- [24] Fake sites database, 2021, Retrieved January 14, 2021 from <https://db.aa419.org/fakebankslist.php?start=20>.
- [25] T.A. Almeida, J.M. GÃmez Hidalgo, A. Yamakami, Contributions to the study of SMS spam filtering: New collection and results, in: *Proceedings of the 2011 ACM Symposium on Document Engineering (DOCENG'11)*, Mountain View, CA, USA, 2011.
- [26] Nigerian scam phone numbers, 2021, Retrieved January 14, 2021 from https://www.watchforscams.com/nigerian_scam_phone_numbers.html.
- [27] The number of mobile phone users in the world, <https://datareportal.com/global-digital-overview#:text=There%20are%205.22%20billion%20unique,of%201.8%20percent%20per%20year>.
- [28] J. Woong, J. Seo, Y. Moon, S. Singh, J. Hyuk, S-detector: An enhanced security model for detecting smishing attack for mobile computing, *Telecommun. Syst.* 66 (2017) 29–38, <http://dx.doi.org/10.1007/s11235-016-0269-9>.