A Fog-Augmented Machine Learning based SMS Spam Detection and Classification System

Sahar Bosaeed, Iyad Katib, Rashid Mehmood
Faculty and Computing and Information Technology
King Abdulaziz University
Jeddah, KSA
Email: SBusaeed0001@stu.kau.edu.sa, iakatib@kau.edu.sa, RMehmood@kau.edu.sa

Abstract—**Smart cities and societies are driving unprecedented technological and socioeconomic growth in everyday life albeit making us increasingly vulnerable to infinitely and incomprehensibly diverse threats. Short Message Service (SMS) spam is one such threat that can affect mobile security by propagating malware on mobile devices. A security breach could also cause a mobile device to send spam messages. Many works have focused on classifying incoming SMS messages. This paper proposes a tool to detect spam from outgoing SMS messages, although the work can be applied to both incoming and outgoing SMS messages. Specifically, we develop a system that comprises multiple machine learning (ML) based classifiers built by us using three classification methods -- Naïve Bayes (NB), Support Vector Machine (SVM), and Naïve Bayes Multinomial (NBM) – and five preprocessing and feature extraction methods. The system is built to allow its execution in cloud, fog or edge layers, and is evaluated using 15 datasets built by 4 widely-used public SMS datasets. The system detects spam SMSs and gives recommendations on the spam filters and classifiers to be used based on user preferences including classification accuracy, True Negatives (TN), and computational resource requirements.**

Keywords— smart cities, SMS spam, classifiers, machine learning, fog computing, edge computing, cloud computing

I. Introduction

We are witnessing unprecedented technological and socioeconomic growth in our everyday lives being carried out as part of the smart city and society developments [1]. However, the growing reliance on digital artifacts and the evolution of our physical world into the cyberphysical world are making us vulnerable to infinitely and incomprehensibly diverse threats. Various digital objects such as in smart homes, offices, buildings, factories, traffic lights, airports, and more are being connected to the digital world we live in through Internet of Things and other paradigms. These connected objects are vulnerable to being spied on, and/or taken control of, and could lead to disastrous circumstances. Spam messages in Short Message Service (SMS) is one such threat. The growing use of SMS (Short Message Service) recently have encouraged spam messages such as promotion announcements from stores, credit opportunities from banks, new tariffs and services from communication service providers, announcements about conferences and events, and more. The legitimate and useful messages have now become a small minority among the received SMS messages [2]. Moreover, spams such as in SMS, WhatsApp, and other messaging services are a well-known threat that can affect mobile security by propagating malware on mobile devices. A security breach could also cause a mobile device to send spam messages [3].

Many works have focused on classifying incoming SMS messages (the notable ones have been reviewed in Section II). Some have looked at finding the similarities between the SMS and email spamming [4], and others have look at the differences between the spamming on the two media [5].

The aim of this work is to fight against the propagation of SMS spams. We aim to detect the SMS spams generated by the malwares that illegitimately get installed on mobile devices. We develop and study the performance of different datasets, preprocessing methods, and feature extraction methods on spam and benign SMS classification in order to find the best process configurations a system should use in various layers of a distributed system (cloud, fog, and edge layers). Specifically, this paper takes the approach to search spam from outgoing SMS messages and presents a machine learning (ML) based tool for SMS spam detection. The detection and classification system that we propose here can be applied to both incoming and outgoing SMS messages. We develop a system that comprises multiple machine learning (ML) based filters. These filters are built by us using three ML classifiers -- Naïve Bayes (NB), Support Vector Machine (SVM), and Naïve Bayes Multinomial (NBM) – and five pre-processing and feature extraction methods (called filters, PF1 – PF5). The system is built such that it can be executed on both server and Android mobile platforms. The purpose of developing both mobile and server applications is to allow both platforms to be able to detect SMS spam. We will see later in the paper that the classifiers and filters that we have developed vary in their computational requirements and classification accuracies. Building a system for multiple platforms allows the proposed system to execute itself on the most suitable platform based on user preferences and/or embedded system policies. For instance, the system policy could be to execute the classifier with the highest accuracy if sufficient computational resources are available, and/or to meet the delay requirements of the classification process; and, to resort to a classifier with a lower accuracy if the system is failing to meet these requirements. The system may choose to execute a classifier on a server in the cloud or on a mobile application based on the computational and QoS requirements. The system may also resort to executing the classifier in the fog if the cloud is unable to meet the delay requirements. We will elaborate this further when we discuss the system architecture in Section III. We have evaluated our proposed system using 15 datasets built by 4 widely-used public SMS datasets and one User dataset collected by us. The system detects spam SMSs and gives recommendations on the spam filters and classifiers to be used based on user preferences including classification accuracy, True Negatives (TN), and computational resource requirements.

The novelty of our work includes the use of outgoing messages for spam detection; the development of multiple filters and classifiers using novel combinations of machine learning algorithms, preprocessing, and feature extraction methods; a mix of 15 datasets for training and evaluation of the proposed system; and the fog-augmented architecture comprising development of the proposed classifiers on server and mobile platforms.

The rest of the paper is organized as follows. Section II reviews the related work. Section III describes the methodology and design. Section IV provides the details of the implementation on the mobile and server platforms. Section V discusses the results. Section VI concludes and gives future directions.

II. Literature Review

The growing use of SMS recently have encouraged spam messages to an extent that useful messages have now become a small minority [2]. Many works have focused on classifying incoming SMS messages. Sethi and Bhootna [4] proposed to find the similarities between the SMS and email spamming. The need to consider the limited resources of a mobile phone for detecting spam and compromising on detection accuracy have been discussed in [6]. Alzahrani and Ghorbani [7] collect data from a mobile and send to a server to detect spam using behavior and signature detection techniques. Chan et al. [8] dealt with the good word attack by using the feature reweighting method and a rescaling function to classify the spam messages. Their strategy considered the weight of the classifier and the length of words using SVM in two short message dataset. Chen et al. [9] proposed TruSMS, a system based on trust management. They analyze spam detection behavior and SMS traffic data to control spam from source to destination. Abulaish [5] draw attention to the differences between the spamming on SMS and email media such as the use of variants of words and abbreviations, and propose a graph-based supervised learning method for tokenizing SMS message. Almeida et al. [10] present the largest real, public and non-encoded SMS spam dataset. They make sure that the dataset does not contain any duplication from the previous exists dataset. Sohn et al. [11] propose a new feature, called stylistic feature, that reflect the style or the manner in which the content is expressed. The stylistic features are the word and sentence length, function word count, and syntactic information. Kim, Jo, and Choi [12] propose a Frequency Ratio (FR) to measure lightness and quickness of filtering methods. A higher value of FR(j) implies that the word "j" is more frequently referred in spam messages. Zheng and Liu [13] propose a linear discrimination based key word selection with approximated logistic regression (KW-ALR). It extracts keywords using linear discrimination analysis and then trains the spam recognition model based on approximated logistic regression over the extracted keywords. Felt et al. [14] expose the incentives for writing mobile malware, which include marketing user information, premium-rate calls SMS, SMS spam, novelty and amusement, stealing user credentials, search engine optimization and ransom. A recent review of mobile SMS spam filtering techniques can be found in [15].

III. Methodology and Design

This section is divided into five subsections that describe the architectural overview of the proposed system, datasets, pre-processing and feature extraction, classification, and the numerical evaluation, respectively.

A. System Architectural Overview

Figure 1 depicts the architecture of the proposed system. The left vertical block lists the three layers of distributed systems, Cloud, Fog, and Edge Devices/IoT. The right hand block depicts the machine learning pipeline for spam classification. On the right, the bottommost block shows the datasets that undergo pre-processing, feature extraction, training, and classification into benign and spam messages. The datasets that contain benign and spam data (Section III.B) are used for training and testing purposes. Pre-processing

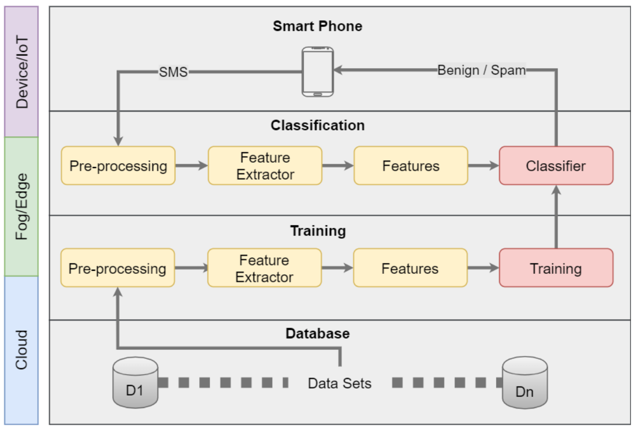


Figure 1 The System Architecture

(Section III.C) is needed to clean and prepare the data for the next stage. Feature extraction III.C) is needed to reduce the feature space by extracting important features from the dataset helping to train better and faster. Classification (Section III.D) includes the training phase and then the actual classification of the data into spam and benign messages. The numerical evaluation (Section III.E) of the classifiers is performed to evaluate the process accuracy. The right-hand system components can execute in the three layers (left-hand blocks) based on the system policies and user preferences in terms of resource availability and QoS requirements. This will become clearer in the rest of the paper as we explain and evaluate the classification system and its mobile and server implementations.

B. The Datasets

In machine learning, datasets are important and should be selected carefully for appropriate training and testing of the machine learning models. Unfortunately, there are no large and public SMS datasets and this poses a major challenge for spam SMS filtering [16]. The reason for the lack of large public datasets could be that SMS services are run by private companies and they cannot reveal their customer's data for research purposes [17]. The largest collection of SMS messages is provided by the National University of Singapore (NUS) SMS corpus. NUS has collected 45,718 English SMS and 31,465 Chinese SMS messages, and have provided these publically for research [18]. This SMS corpus is not labelled, i.e., it does not indicate that a message is spam or benign. Moreover, the NUS dataset comprises SMS in Singapore English. It has terms like "lor" or "lah", which is different from British English [17], and our aim is to focus on standard English. The dataset should be balanced in order for the trained models to be able to keep its performance across different data [19]. A study on the effects of balanced and unbalanced datasets using twelve different classifiers using Weka has been reported in [20]. They compared their work with Almeida et al. [10] and concluded that balanced datasets are a necessity for developing effective classifiers.

In this work, we have used four widely-used public datasets in SMS classification research. These are British Dataset [21], UCI [22], SMS Spam Corpus Big [23], and Spam SMS Data [24]. Additionally, we have used another dataset, called the User dataset, which comprises benign and spam SMS collected by us. We have created 15 different datasets (see TABLE 1) from these five datasets in order to study the impact of various dataset characteristics on pre-processing, feature extraction and classification methods. The dataset characteristics and variation that we have considered in this work include unbalanced dataset (in terms of the number of benign and spam SMS: see e.g. D4), balanced dataset (D10, D13, etc.), different kinds of datasets (e.g., D13, the User dataset combined with the UCI dataset), and the User dataset combined with all other four different spam datasets. The numbers in Column 2 and 3 in the table give the number of benign and spam SMS respectively.

TABLE 1 THE DATASETS

Name	Benign Dataset	Spam Dataset
D1	British Dataset:206	British Dataset:226
D2	British Dataset:206	British Dataset:425
D3	British Dataset:450	British Dataset:425
D4	SMSSpamCorpusBig:1002	SMSSpamCorpusBig:322
D5	SMSSpamCorpusBig:206	SMSSpamCorpusBig:226
D6	SpamSMSData:206	SpamSMSData:226
D7	SpamSMSData:730	SpamSMSData:721
D8	UCI:206	UCI:226
D9	UCI:4826	UCI:748
D10	User:206	British Dataset:226
D11	User:206	British Dataset:425
D12	User:206	British Dataset:112
D13	User:206	UCI:226
D14	User:206	SpamSMSData:226
D15	User:206	SMSSpamCorpusBig:226

C. Preprocessing and Feature Extraction

Pre-processing of data is an important step in machine learning. It involves steps such as stop word removal and stemming [2], cleaning, transformation, reduction, and tokenization [4]. Moreover, the efficiency and effectiveness of the content based classification methods and machine learning models depend on proper selection of a feature set. In order to investigate and find a good hybrid configuration, we have used five different combinations of various pre-processing and feature extraction methods available in Weka. These five combinations that we call filter in this paper, PF1 to PF5, are listed in TABLE 2.

The PF1 filter tokenize the string using the java.util.StringTokenizer class in Weka [25]. We did not used any stemming or stop word extraction in this filter. The number of words to keep from the whole dataset after tokenization was set to 1000. PF2 uses the lowercase pre-processing. All tokens will be in lowercase characters. It extracts the stop words that is listed in the Rainbow list [26].

Lovins Stemmer removes the longest suffix from a word to turn the stem into valid words [27]. The PF3 filter uses the same preprocessing methods of PF2 in addition to the InfoGainAttributeEval feature extractor. It is a feature selection technique used by WEKA [28]. This feature extractor evaluates the value of an attribute by measuring the information gain depending on the class (benign/spam). If the entry values is 0 then there is no information gain, and if it is 1 then it gains the maximum information. The attribute that has higher information gain value will be selected [29]. PF4 uses lower case tokens, and the number of words to be kept are set to 500. It uses the InfoGainAttributeEval feature extractor without any stemming or stop words removal. The PF5 filter is similar to PF4 with the addition of a new attribute that uses the message length values to be examined by the classifiers.

TABLE 2 PREPROCESSING AND FEATURE EXTRACTION

Name	Description
PF1	Word Tokenization, No Stemmer or Stop word, words to keep: 1000
PF2	Lower Case Tokens, evaluator: Stop word (Rainbow), stemming (Lovins Stemmer) and words to keep: 500
PF3	Pre-processing: Lower Case Tokens, evaluator: Stop word (Rainbow), stemming (Lovins Stemmer) and words to keep: 500. Feature Extractor: InfoGainAttributeEval-search: Ranker with a threshold=0
PF4	Lower Case Tokens, and words to keep: 500. Feature Extractor: InfoGainAttributeEval-search: Ranker with a threshold=0
PF5	Adding a new Attribute msgLength which stores the message length values and then performs PF4 filter on it

D. Classification

Classifiers are methods that map input data to a specific class. The classifiers that have been used for SMS spam detection could be classified into evolutionary or machine learning classifiers [30][31]. The most effective of the machine learning classifiers are NB (Naïve Bayes), NBM (Naïve Bayes Multinomial), and SVM (Support Vector Machine) [20]. We have used these in our work. NB is one of the most well-known classifiers in the field of message classification. The classifier can give the finest results when dimensions of the input are high. It is common to use NB with text documents because it uses Bayesian theorem that assumes the attributes of the dataset are independent and computes the probability of them by calculating the frequency of values, and the relationship between them. SVM is a supervised learning algorithm and can be used for regression or classification. It classifies by building an N-dimensional hyperplane that separates the data into two classes. SVM is considered an effective classifier for text documents and SMS spam filtering. It is preferred due to its high accuracy in text classification though its memory usage could be high. NBM is a generic technique used in the text mining domain. It has been used for semi-automated classification of documents such as spam mail detection tasks [32].

Normally, the classification of the spam SMS is done on the received messages. So, the dataset that the classifier will use to classify SMSs is generated from different users. In our proposed system, we have taken the approach to classify the outgoing messages. Thus, the dataset used will consider just one user (the sender side). For each sender mobile, there will be a dataset containing the sender's text messages, and the machine learning model learns from it and classifies the SMS sent from the device as spam or benign. The classifiers are trained using the fifteen datasets that we have discussed earlier (see TABLE 1) and are evaluated for these datasets along with the five pre-processing and feature extraction filters that we have developed, PF1 – PF5 (see). These configurations allow us to study the performance of the three classifiers (NB. SVM, and NBM) for a range of variations in datasets, preprocessing and feature extraction methods. We have implemented the classifiers on two different platforms, Android mobile and server, details of these will be discussed in Section IV.

E. Numerical Evaluation Metrics

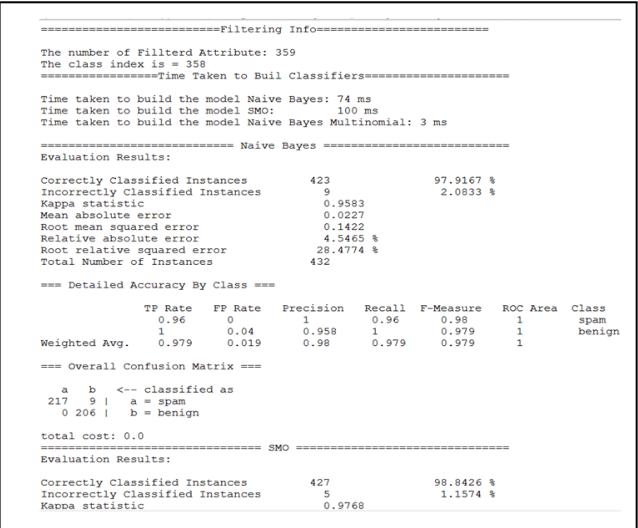
We have used 10-fold cross-validation to evaluate the three classifiers. The training dataset was divided into 66% and 34% for testing. The numerical evaluation criteria used in this work include FP, TP, FN, TN, accuracy, and precision, which are widely-used. Spam is considered as the positive class and benign as the negative. The definitions for the various evaluation metrics can be found, for instance, in [15].

IV. System Implementation

We have implemented two versions of the proposed SMS classification system. The Server App or Component could perform the training and testing/classification on the server and send the results to the mobile. The Mobile App or Component can perform the training and classification on its own. The Server Component can be useful for example if the training or classification requires larger computational resources. The Mobile version could use lighter classification methods to train or classify if the delay, privacy or other QoS measures limit the use of Server App. Moreover, a hybrid configuration could be used where the training is done on the server and actual classification is done on the smartphone. This flexibility of multiple platforms allows the proposed system to be executed, partly or fully, in the cloud, fog or edge devices, allowing optimization of resources, privacy, security, and QoS. The implementations of both mobile and server Apps are done using Weka [25].

A. The Server App

The Server App has been implemented on an HP OMEN device, Intel core i7 2.5 GH processor, 16 GB RAM, and Windows 10 OS. It is composed of a Datasets Menu (15 datasets, see TABLE 1), Filters Menu (PF1 – PF5), and Classifiers Menu (NB, SVM, NBM). The User dataset has been converted to an .arff file so it could be combines with other datasets. Figure 2 shows a sample of the command line output. Note the three classifiers in the figure; SMO (Sequential minimal optimization) represents SVM. The sample shows results for NB and some results for SMO. The results include time to build, correctly classified instances, and various classification accuracy statistics (correctly classified instances, TP rate, MAE, Kappa, etc.).

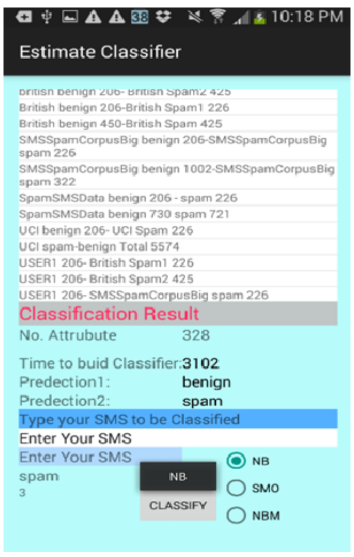


```
=========================Filtering Info=========================

The number of Fillterd Attribute: 359
The class index is = 358
=================Time Taken to Buil Classifiers====================

Time taken to build the model Naive Bayes: 74 ms
Time taken to build the model SMO:         100 ms
Time taken to build the model Naive Bayes Multinomial: 3 ms

============================ Naive Bayes ===========================
Evaluation Results:

Correctly Classified Instances         423               97.9167 %
Incorrectly Classified Instances         9                2.0833 %
Kappa statistic                          0.9583
Mean absolute error                      0.0227
Root mean squared error                  0.1422
Relative absolute error                  4.5465 %
Root relative squared error             28.4774 %
Total Number of Instances              432

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.96      0         1          0.96     0.98        1         spam
                 1         0.04      0.958      1        0.979       1         benign
Weighted Avg.    0.979     0.019     0.98       0.979    0.979       1

=== Overall Confusion Matrix ===

   a   b   <-- classified as
 217   9 |   a = spam
   0 206 |   b = benign

total cost: 0.0
================================ SMO ===============================
Evaluation Results:

Correctly Classified Instances         427               98.8426 %
Incorrectly Classified Instances         5                1.1574 %
Kappa statistic                          0.9768
```

Figure 2 Classification on the Server App

B. The Mobile App

The Mobile App has been implemented on an Android device, Samsung Galaxy S3-GT I9300, quad-core 1.4 GHz Cortex-A9-Chipset Exynos 4412 quad processor, Android version 4.3 (Jelly Bean) operating system and 1 GB RAM (see Figure 3). Similar to the Server App, the Mobile App has various menus, which the user could use to select a specific dataset, filter, and classifier to get the various statistics (classified instances, time to build, Kappa, etc.). Also, the user can type a text message in the Mobile App and classify it according to a specific dataset, filter, and classifier. The results are displayed in the middle of the GUI.



10:18 PM

Estimate Classifier

British benign 206- British Spam2 425
British benign 206-British Spam1 226
British benign 450-British Spam 425
SMSSpamCorpusBig benign 206-SMSSpamCorpusBig spam 226
SMSSpamCorpusBig benign 1002-SMSSpamCorpusBig spam 322
SpamSMSData benign 206 - spam 226
SpamSMSData benign 730 spam 721
UCI benign 206- UCI Spam 226
UCI spam-benign Total 5574
USER1 206- British Spam1 226
USER1 206- British Spam2 425
USER1 206- SMSSpamCorpusBig spam 226

Classification Result

No. Attrubute	328
Time to buid Classifier:	3102
Predection1:	benign
Predection2:	spam

Type your SMS to be Classified

Enter Your SMS

Enter Your SMS

spam

3

[✓ NB] SMO NBM

Figure 3 Classification on Mobile App

V. Results And Analysis

We have collected results for a range of metrics including accuracy, precision, TPR, FPR, TNR, FNR, and computational complexity to build the classifier. We have limited the discussion of the results in this paper due to the 6-page limit.

A. Preprocessing and Feature Extraction (PF1 – PF5)

We have collected various performance metrics for the 15 datasets, 5 filters, and 3 classifiers. We would have liked to indulge in presenting and discussing the results for all 15 x 5 x 3 combinations of the datasets, filters and classifiers, however, due to lack of space we have decided to focus on one filter that produces the best results. We found that PF5 produces the best results, overall, among the 5 filters as, for example, can be seen in Figure 4 that shows the accuracy results of all 5 filters for the SVM classifier. The highest accuracy achieved by D10 with 98.8%.

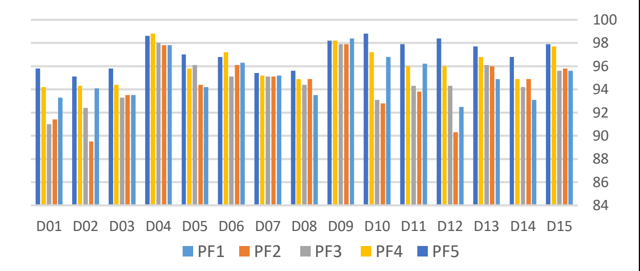


Figure 4 SVM accuracy (all filters)

B. *Computational Complexity*

Figure 5 and Figure 6 show the time taken to build the three classifiers using PF5 on server and mobile platforms, respectively. Note the differences in time for the various datasets and classifiers for the two Apps. Mobile App is significantly slower than Server App (D9 on the mobile is approx. 30,000 vs 1,000 on the server). Looking at the 15 datasets, Dataset 9 (it is the largest dataset, see TABLE 1) has taken much higher times than the other datasets for SVM (Server app: 922ms, mobile app: 30457ms) and NB (Server app: 919ms, mobile app: 30349ms); and the time for NBM (Server app: 11ms, mobile app: 109ms) is much smaller. We will see later that SVM and NB provide higher accuracies, overall, compared to NBM but the differences in accuracies could be ignored to use NBM subject to the user preferences if computational resources are limited. Note that NB requires the highest time to build, overall, followed by SVM and NBM.

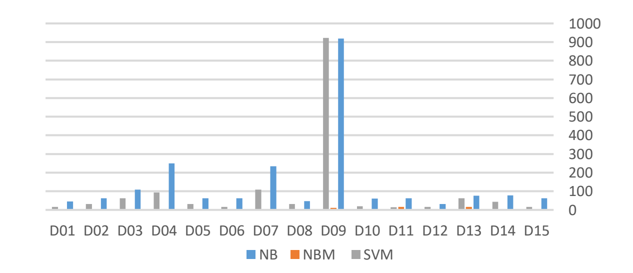


Figure 5 Server App: Time to Build Classifier (PF5) (in ms)

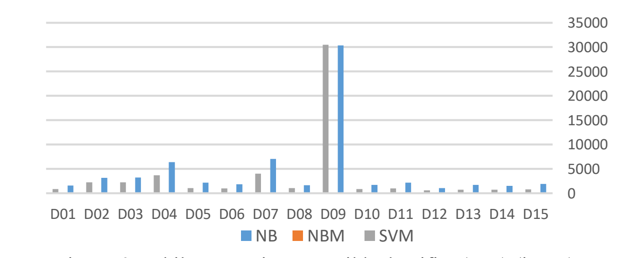


Figure 6 Mobile App: Time to Build Classifier (PF5) (in ms)

C. *Accuracy*

Figure 7 depicts the accuracy results for the 15 datasets using PF5. The overall trend is that SVM performs the best i.e.

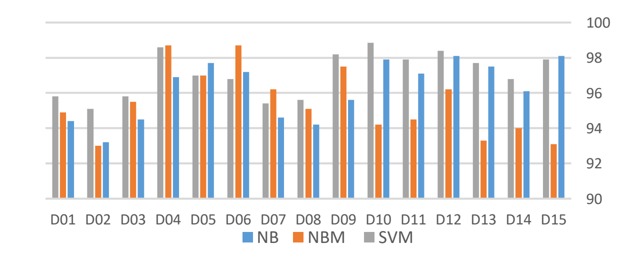


Figure 7 Accuracy using PF5 (3 Classifiers)

D10 (98.84) followed by NB and NBM, with some variations such as NBM performs the best for D6 (98.7) and D4(98.6), and NB performs best for D12(98.1) and D15 (98.1).

D. *Spam and Benign Precision*

Figure 8 and Figure 9 show the spam and benign precision using PF5 filter. We have considered spam as positive and benign as negative because we are detecting spams. Note that precision reaches 100 or near 100 for some of the datasets i.e. D6. There is a clear difference between the classifiers performance for the two classes, spam and benign. NBM is the best performing classifier for spam detection, overall, followed by a mixed behavior by SVM and NB. It appears that this behavior is due to the specific unbalanced datasets which provide erratic performance. Future work will look further into these aspects including the possibility for using a different classifier for spam and benign detection.

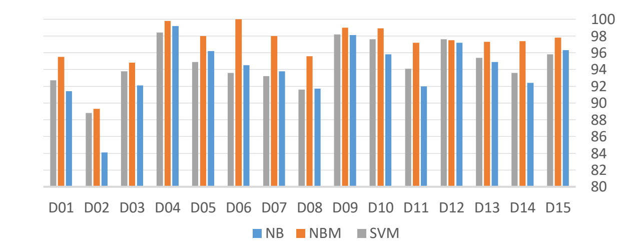


Figure 8 Spam Precision Using PF5

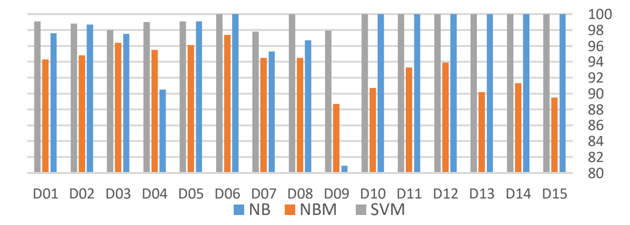


Figure 9 Benign Precision Using PF5

E. *TNR*

Figure 10 shows TNR (True Negative Rate) results using the PF5 filter for the three classifiers. TN means that any benign message is classified correctly as benign. The purpose of considering this metric is that we are classifying the sent messages, so it is important that we classify the benign messages correctly otherwise these will be blocked. Note that SVM provides the best performance (most datasets with 100 or near-100 TNR), overall, with NB providing similar performance except for the datasets, D8, D7, D4, D12 and D10. NBM does not perform well for all datasets.

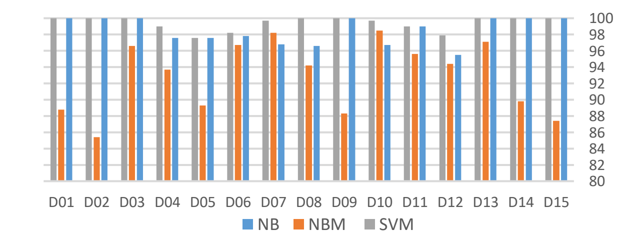


Figure 10 TNR using PF5

VI. CONCLUSION & FUTURE WORK

Smart cities and societies are driving unprecedented technological and socioeconomic growth in everyday life albeit making us increasingly vulnerable to infinitely and incomprehensibly diverse threats. Short Message Service (SMS) spam is one such threat that can affect mobile security by propagating malware on mobile devices. A security breach could also cause a mobile device to send spam messages. We have developed and studied the performance of different datasets, preprocessing methods, and feature extraction methods on spam and benign SMS classification in order to find the best process configurations a system should use in various layers of a distributed system; cloud, fog, and edge layers. We have found that the PF5 filter and SVM perform the best, overall, in classifying SMS messages. Future work will focus on deeper investigations into the classification, preprocessing and feature extraction methods, dataset characteristics, and the optimization of the software in fog, edge and cloud layers.

REFERENCES

[1] R. Mehmood, S. See, I. Katib, and I. Chlamtac, Eds., Smart Infrastructure and Applications: foundations for smarter cities and societies. EAI/Springer Innovations in Communication and Computing, Springer International Publishing, Springer Nature Switzerland AG, 2020.

[2] A. K. Uysal, S. Gunal, S. Ergin, and E. S. Gunal, "A novel framework for SMS spam filtering," INISTA 2012 - Int. Symp. Innov. Intell. Syst. Appl., 2012.

[3] "Android mobiles hit by spamming computer virus - BBC News," 2012. [Online]. Available: https://www.bbc.com/news/technology-20768996.

[4] G. Sethi and V. Bhootna, "SMS Spam Filtering Application Using Android," Int. J. Comput. Sci. Inf. Technol., vol. 5, no. 3, pp. 4624–4626, 2014.

[5] M. Abulaish, "Graph-Based Learning Model for Detection of SMS Spam on Smart Phones," pp. 1046–1051, 2012.

[6] G. Mujtaba and M. Yasin, "SMS Spam Detection Using Simple Message Content Features," J. Basic Appl. Sci. Res., vol. 4, no. 4, pp. 275–279, 2014.

[7] A. J. Alzahrani and A. A. Ghorbani, "SMS mobile botnet detection using a multi-agent system: Research in progress," ACM Int. Conf. Proceeding Ser., 2014.

[8] P. P. K. Chan, C. Yang, D. S. Yeung, and W. W. Y. Ng, "Spam filtering for short messages in adversarial environment," Neurocomputing, vol. 155, pp. 167–176, 2015.

[9] L. Chen, Z. Yan, W. Zhang, and R. Kantola, "TruSMS: A trustworthy SMS spam control system based on trust management," Futur. Gener. Comput. Syst., vol. 49, pp. 77–93, 2015.

[10] T. A. Almeida, J. María, G. Hidalgo, and T. P. Silva, "Towards SMS Spam Filtering: Results under a New Dataset," Int. J. Inf. Secur. Sci., vol. 2, no. 1, pp. 1–18, 2016.

[11] D. N. Sohn, J. T. Lee, K. S. Han, and H. C. Rim, "Content-based mobile spam classification using stylistically motivated features," Pattern Recognit. Lett., vol. 33, no. 3, pp. 364–369, 2012.

[12] S. E. Kim, J. T. Jo, and S. H. Choi, "SMS Spam filterinig using keyword frequency ratio," Int. J. Secur. its Appl., vol. 9, no. 1, pp. 329–336, 2015.

[13] Y. Zheng and F. Liu, "Filtering Network Spam Message using Approximated Logistic Regression," J. Networks, vol. 9, no. 9, pp. 2462–2467, 2014.

[14] A. P. Felt, M. Finifter, E. Chin, S. Hanna, and D. Wagner, "A survey of mobile malware in the wild," Proc. ACM Conf. Comput. Commun. Secur., no. October, pp. 3–14, 2011.

[15] S. M. Abdulhamid et al., "A Review on Mobile SMS Spam Filtering Techniques," IEEE Access, vol. 5, pp. 15650–15666, 2017.

[16] J. M. G. Hidalgo, T. A. Almeida, and A. Yamakami, "On the validity of a new SMS spam collection," in Proceedings - 2012 11th International Conference on Machine Learning and Applications, ICMLA 2012, 2012, vol. 2, pp. 240–245.

[17] S. J. Delany, M. Buckley, and D. Greene, "SMS spam filtering: Methods and data," Expert Syst. Appl., vol. 39, no. 10, pp. 9899–9908, 2012.

[18] T. Chen and M.-Y. Kan, "Creating a Live, Public Short Message Service Corpus: The NUS SMS Corpus," Dec. 2011.

[19] A. A. Hammad and A. El-Halees, "An Approach for Detecting Spam in Arabic Opinion Reviews," 2015.

[20] H. Najadat, N. Abdulla, R. Abooraig, and S. Nawasrah, "Mobile SMS Spam Filtering based on Mixing Classifiers," Int. J. Adv. Comput. Res., vol. 1, no. March, pp. 1–7, 2014.

[21] M. T. Nuruzzaman, C. Lee, and D. Choi, "Independent and personal SMS spam filtering," in Proceedings - 11th IEEE International Conference on Computer and Information Technology, CIT 2011, 2011, pp. 429–435.

[22] "UCI Machine Learning Repository." [Online]. Available: https://archive.ics.uci.edu/ml/index.php. [Accessed: 25-Jan-2020].

[23] "SMS Spam Collection." [Online]. Available: http://www.dt.fee.unicamp.br/~tiago/smsspamcollection/. [Accessed: 27-Jan-2020].

[24] A. Narayan and P. Saxena, "The curse of 140 characters: Evaluating the efficacy of SMS spam detection on Android," in Proceedings of the ACM Conference on Computer and Communications Security, 2013, pp. 33–41.

[25] "Weka 3 - Data Mining with Open Source Machine Learning Software in Java." [Online]. Available: https://www.cs.waikato.ac.nz/ml/weka/. [Accessed: 25-Jan-2020].

[26] "StopWords list based on Rainbow statistical text · GitHub." [Online]. Available: https://gist.github.com/shuson/b3051fae05b312360a18. [Accessed: 08-Mar-2020].

[27] "The Lovins stemming algorithm." [Online]. Available: http://snowball.tartarus.org/algorithms/lovins/stemmer.html. [Accessed: 08-Mar-2020].

[28] "Class InfoGainAttributeEval." [Online]. Available: https://weka.sourceforge.io/doc.dev/index.html?weka/attributeSelection/InfoGainAttributeEval.html.

[29] "How to Perform Feature Selection With Machine Learning Data in Weka." [Online]. Available: https://machinelearningmastery.com/perform-feature-selection-machine-learning-data-weka/. [Accessed: 08-Mar-2020].

[30] M. B. Junaid and M. Farooq, "Using evolutionary learning classifiers to do mobile spam (SMS) filtering," Genet. Evol. Comput. Conf. GECCO'11, pp. 1795–1801, 2011.

[31] M. Z. Rafique, N. Alrayes, and M. K. Khan, "Application of evolutionary algorithms in detecting SMS spam at access layer," Genet. Evol. Comput. Conf. GECCO'11, pp. 1787–1794, 2011.

[32] D. Herrmann, R. Wendolsky, and H. Federrath, "Website Fingerprinting: Attacking Popular Privacy Enhancing Technologies with the Multinomial Naïve-Bayes Classifier," 2009.