Proceedings of 2018 Eleventh International Conference on Contemporary Computing (IC3), 2-4 August, 2018, Noida, India

A Comparative Study of Spam SMS Detection using Machine Learning Classifiers

Mehul Gupta, Aditya Bakliwal, Shubhangi Agarwal & Pulkit Mehndiratta
Department of Computer Science, Jaypee Institute of Information Technology, Noida, Uttar Pradesh, India
mgupta1996@gmail.com, abakliwal12@gmail.com, shubhangiagarwal46@gmail.com, pulkit.mehndiratta@jiit.ac.in

***Abstract* - With technological advancements and increment in content based advertisement, the use of Short Message Service (SMS) on phones has increased to such a significant level that devices are sometimes flooded with a number of spam SMS. These spam messages can lead to loss of private data as well. There are many content-based machine learning techniques which have proven to be effective in filtering spam emails. Modern day researchers have used some stylistic features of text messages to classify them to be ham or spam. SMS spam detection can be greatly influenced by the presence of known words, phrases, abbreviations and idioms. This paper aims to compare different classifying techniques on different datasets collected from previous research works, and evaluate them on the basis of their accuracies, precision, recall and CAP Curve. The comparison has been performed between traditional machine learning techniques and deep learning methods.**

Index Terms - Spam SMS, Detection, Machine Learning Classifiers, Neural Networks.

I. Introduction

Short Message Service (SMS) is a technique of sending short text messages from one device to another. Spam refers to the word which is generally used for message or information that is junk or unsolicited. So a spam SMS can be defined as any junk message delivered to a mobile device through text messaging. Whether spam is in the form of email or SMS, danger is equal. Spam may result in leaking personal information, invasion of privacy or accessing unauthorized data from mobile devices.

In this era of Smartphone devices, users now have personal and confidential information such as contact lists, credit card numbers, photographs, passwords and much more stored in their smartphones, making them prone to cyber attacks through spam SMS. This enables hackers involved in unethical activities to access smartphone data without the knowledge of end-user, thus compromising the user's privacy. This could also result in pecuniary as well as functional loss. Spam messages seem to be increasing and result not only in annoyance for users but critical data loss for some users. Apart from this, SMS spam can also act as a driving force for malwares and key-loggers.

The unanimous spreading of this kind of problem and less effective security control measures has inspired many researchers in the development of a set of techniques to help prevent it in a varied and efficient way. The main motive is to handle security issues in terms of protection of privacy, solidarity and accessibility. Many users are still unaware about protection mechanisms, thereby, making their mobiles prone to cyber attacks. The Government of India has set up NCPR (National Customer Preference Register) registry, which to some extent has reduced junk calls, but does not filter spam SMS.

Although various datasets are available to test email spam detection algorithms, the datasets to train and test techniques for SMS spam detection are still limited and small sized. Moreover, unlike emails, the length of text messages is short, that is, less statistically-differentiable information, due to which the availability of number of features required to detect spam SMS are less. Text messages are highly influenced by the presence of informal languages like regional words, idioms, phrases and abbreviations due to which email spam filtering methods fail in the case of SMS.

There are various approaches proposed in previous works for detecting email [6, 7, 12] and SMS [1, 2, 4, 5] spam. In this paper, our aim is to train, test and compare different traditional machine learning classifiers and deep learning techniques on different datasets. On the basis of their accuracies, precision and recall, various considered classifiers are evaluated. A comparison and analysis of all the techniques has been then carried out.

II. Related Work

The most common formulation of spam filtering is the task of classification. For a piece of text, the aim is to predict whether or not it is spam. However, past work varies in terms of what kinds of datasets are used for training and testing. For example, Delany et al. [9] created their own dataset consisting of ham messages collected from websites like GrumbleText, Who Calls Me and spam messages from SMS Spam Collection. They analysed the spam messages using content based clustering and nearly identified nine to ten clearly-defined clusters. The approach used by Kim et al. [4] proposed a method that was based on calculation of frequency which measures lightness and quickness of filtering methods. They considered Naive Bayes, J-48 and Logistic algorithms for their research. They proved that their proposed technique had a similar capability as those of others, even though it used a

978-1-5386-6835-1/18/$31.00 ©2018 IEEE

simple formula, giving them an advantage over other techniques in particular sense. Mahmoud and Mahfouz [5] developed an Artificial Immune System (AIS) for classification of SMS. The system used a set of features as an input spam filter. It was then used to categorize the text messages with the help of a trained dataset which included spam words, phone numbers etc. The results of this experiment showed better accuracy and convergence speed than the Naive Bayesian algorithm in classifying messages either as spam or non-spam. Researchers like Chan et al. [11] and Najadat et al. [1] used SVM classifier in their research, and further recruited more accurate results compared to other techniques.

Although neural networks have not been widely used for classification of SMS, they have been significantly used to detect spam emails by different researchers. The paper by Yang and Elfayoumy [6] compares feed-forward back-propagation neural network Bayesian classifiers to evaluate their effectiveness in classifying spam email. High accuracy and sensitivity was shown by the neural network algorithm making it a good competition for traditional classifiers, but it took a considerably longer training time. On the other hand, it was observed that the task to construct Bayesian classifiers was very easy, but they were not as accurate. Clark et al. [12] showed in terms of classification performance how their back-propagation based system outperformed many other algorithms. The effects of various different techniques like feature selection, weighting and normalization were also explored. The best results in email spam filtering were produced by normalization at mailbox level with Frequency and tf-idf weighting.

Convolutional Neural Networks have traditionally been used for image classification problems. The paper by Zhang et al. [13] goes against this notion by comparing character-level convolutional networks with a number of traditional and deep learning models using very big datasets in their empirical study. The results so obtain encourage the application of such neural networks for problems of text classification and a broader range of natural language processing.

Researchers like Murynets and Jover [10] have gone even further to analyse the SMS traffic. Messaging behaviour similar to a spammer was observed for some networked applications because few Machine to Machine systems transmit a huge count of texts per day. It was suggested to consider such systems when designing SMS spam detection models.

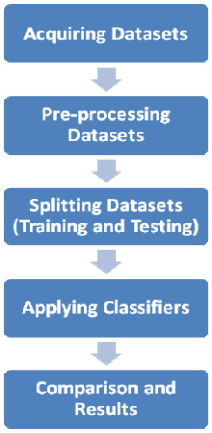

Fig. 1 Flow Diagram

III. Methodology

We begin with conversion, preprocessing and splitting of datasets as per the requirement of considered algorithms. The various models are then trained and tested, followed by evaluation and comparison based on performance metrics.

A. *Dataset Description*

Dataset#1: SMS Spam Collection V.1

This is a collection of 5574 spam and legitimate English text messages gathered from the following free research sources: National University of Singapore SMS Corpus (3,375 Ham SMS), Grumbletext Website (425 Spam SMS), Caroline Tag's PhD Theses (450 Ham SMS), and SMS Spam Corpus v.0.1 Big (1002 Ham SMS and 322 Spam SMS). With a total of 4827 legitimate messages and 747 spam messages, the corpus is hosted at the UCI Machine learning repository and also available in raw format publicly at [14].

Dataset#2: Spam SMS Dataset 2011-12

The SMS database contains 1,000 Spam and 1,000 Ham SMS. For collecting SMS spam data, Yadav et al. ran an incentivized crowd-sourcing scheme in their campus. Due to large influence of regional words, SMS with both Hindi & English words were collected from 43 participants. The dataset is available on request at [15].

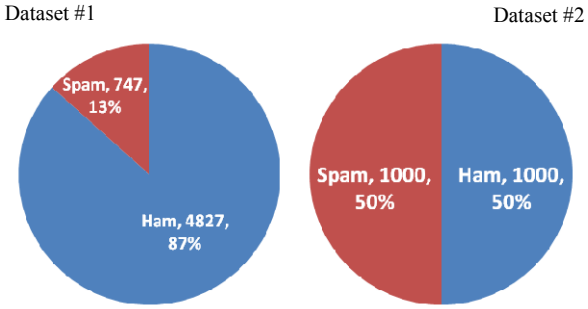

Fig. 2 Distribution of Datasets

B. *Preparing Readable Datasets*

The datasets have been prepared as comma-separated values (CSV) files. These files contain one text message per line. Each line has two columns - *v1* is the label (ham or spam) and *v2* is the raw text.

The Spam SMS Dataset 2011-12 was procured as a zipped file with numerous text files containing a message each. The name of the file indicated whether the message it contained was legitimate or spam. A script was prepared and executed to accurately label the messages and unify them into a single CSV file.

For any classifier to be able to use this data, we need to do some preprocessing.

C. *Data Preprocessing*

Different preprocessing approaches have been applied to different classifiers based on their requirement of input data. Following is a brief description of these approaches.

1) Using Term Frequency—Inverse Document Frequency (tf-idf): In a given document, the count of the number of times a word appears is called Term Frequency. In a given corpus of documents, the number of times a word appears is called Inverse Document Frequency. Words are weighed according to the importance in tf-idf. Frequently used words have a lower weight, while words used infrequently have higher weight.

We started with removal of stop words, capital letters, non alpha-numeric characters and any unnecessary punctuation. We then collected similar words (for example, desks will be transformed to desk). We then converted the cleaned text to tf-idf features (5000 features for an entry) using sklearn's TfidfVectorizer to create a bag of words i.e., a count vector, followed by tf-idf matrix.

2) Using Tokenizer: When working with text, it is always good to start with splitting the text into words. Words are known as tokens. Tokenization is the process of splitting text into words or tokens. Keras' Tokenizer is a class for vectorizing texts. It is used to turn texts into sequences. A sequence is a list of word indexes where the word of rank *i* in the dataset (starting at 1) has index *i*.

Text can be split into a list of words using the text_to_word_sequence() function provided by Keras. This function by default splits words by space, converts text to lowercase and filters out punctuation. Tokenization is restricted to the top most common words in the corpus. We defined 5000 as the maximum number of words to work with.

After applying either of the approaches on our datasets for different classifiers, the representation of first column was changed. Ham and spam were transformed to values 0 and 1 using sklearn's LabelEncoder.

D. *Training and Test Datasets*

The datasets were split into two parts - the data that will be used to train our classifiers and the data that will be used to test them. We split our datasets such that 80% of the data was used for training while 20% of the data was used for testing.

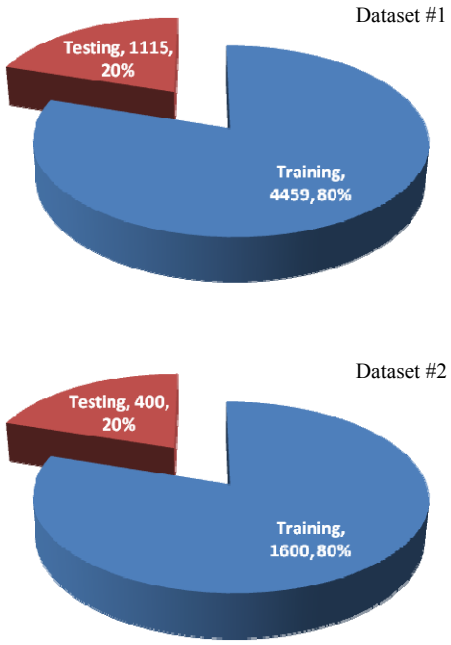

Fig. 3 Splitting of Datasets

E. *Different Classifiers*

Different preprocessing approaches have been applied to different classifiers based on their requirement of input data. Following is a brief description of these approaches.

1) Support Vector Machine (SVM): SVM is a discriminative classifier which is widely used for classification task. The algorithm plots each data item as a point in *n*-dimensional space assuming the value of each feature as the value of a particular coordinate. It then forms a line that splits the whole data into two differently classified groups of data. The closest points in the two groups will be the farthest away from this line.

2) Naive Bayes (NB): This classification technique is based on Bayes' theorem, assuming independence between predictors. The Bayesian classifier assumes that a particular feature in a class is not related to the presence of any other feature. Even if they do depend upon each other or upon the existence of the feature, the Bayesian classifier will consider all of the desired properties to independently contribute to the probability. The classifier holds well when the desired input's dimensionality is high. It is regarded as simple and sturdy.

An advanced version of NB is Multinomial Naive Bayes (MNB). The main improvement is the independence between document length and class. It involves multinomial

distribution which works well for countable type of data such as the words in a document or text. In simple terms, NB classifier involves conditional independence of each of the features in the model, whereas MNB classifier is a special case of a NB classifier which uses a multinomial distribution for each feature.

3) Decision Tree (DT): DT is a supervised learning algorithm which is normally preferred for classification tasks. The algorithm works well for both types of variables i.e., categorical and continuous. It starts with splitting the population into multiple homogeneous sets which is done on the basis of most significant attributes or independent variables. DT is non-parametric and hence the need for checking outlier existence or data linearity separation is not required.

4) Logistic Regression (LR): It is considered as the go-to method for classification involving binary results. It is mainly used in estimating discrete values which are based on set of variables which are independent. In more relative terms, LR outputs the probability of an event by fitting it into a logistic function which helps in prediction. The logistic function which is mostly used is sigmoid.

5) Random Forest (RF): It is a term used for an ensemble of decision trees. The Random Forest classifier is a ensemble learning method which involves collection of decision trees. Voting is done to classify a new object which is performed by each tree i.e., the trees mark their votes for that class. The class having most number of votes decides the classification label.

6) AdaBoost: AdaBoost or Adaptive Boosting is a meta-machine learning algorithm, used to increase the performance of a classifier by simply using the weak classifiers to combine them into a strong one. The final output of the boosted classifier depends upon the weighted sum of the output of all the weak classifiers. A drawback of this technique is that although it predicts more accurately, it takes more time for building the boosted model.

7) Artificial Neural Network (ANN): ANNs are nonlinear statistical data modelling techniques defined over complex relationships between inputs and outputs. They have various advantages, but among them, learning by observing datasets is the most recognised one. It is considered as tool for random function approximation, which helps in estimating the most effective methods to come to solutions. One such network is CNN.

8) Convolutional Neural Network (CNN): A convolutional neural network (CNN) is a particular type of artificial neural network which uses perceptrons for supervised learning. The supervised learning is used to analyze data. There are a wide of range of applications involving CNNs. Traditionally used for image processing, CNNs are nowadays used natural language processing as well. A CNN in relative terms is known as a ConvNet. Similar to other ANNs, a CNN also has an input layer, some hidden layers and an output layer, but it is not fully connected. Some layers are convolutional, that use a mathematical model to pass on the results to layers ahead in the network.

IV. Testing and Evaluating

In order to determine certain evaluation metrics the following parametrics were used:

a. True Positive (TP) - the number of test cases that are classified correctly;

b. True Negative (TN) - the number of test cases that are rejected from the main class correctly.

c. False Positive (FP) - the number of test cases that are rejected from the main class incorrectly;

d. False Negative (FN) - the number of test cases that are classified incorrectly to the main class.

Since accuracy is an intuitive metric and has simple interpretation: just counting correctly classified messages, we have used it as the primary evaluation criteria for our classifiers. We have also compared different classifiers on the basis of their precision and recall.

A. Performance Metrics

It is important to choose the correct performance metrics for any experimental setup to obtain the information desired for validation of any proposal or comparison. To analyze and compare the detection capability of the considered classifiers, we have taken into account the following known metrics:

1) Accuracy: It is numerically defined as the closeness of a calculated value to the actual value. It can be formulated as

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

2) Precision: In simpler terms, it can be defined as the ratio of correct results obtained to total number of results. It can be formulated as

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

3) Recall: It is the ratio of the correct results obtained to desired number of correct results. In other words, it is the ratio of related text that is successfully retrieved. It can be formulated as

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

4) CAP Curve: Cumulative Accuracy Profile (CAP) Curve is a more robust and convenient method to compare and assist machine learning classifiers. It is being widely used to visualize the discriminative power of model. The CAP of a model represents the cumulative number of positive results on the y-axis and the corresponding cumulative number of classifying parameters on the x-axis.

The model can be evaluated by comparing our curve with perfect and random curves. The perfect CAP curve is the one in which the maximum number of positive outcomes are achieved directly, whereas the random CAP curve has the positive outcomes which are distributed equally. A model is said to be a good model if its CAP curve lies between the perfect and the random CAP curve, with the curve tending towards the perfect to be better.

There are 2 approaches to analyze the graph of the curve:

First —

• Accuracy Ratio, AR — Ratio of aR to aP

• aR — Area under the User Model Curve (the area between the model CAP and the random CAP)

• aP — Area under the Perfect Model Curve (the area between the perfect CAP and the random CAP)

As (AR)~1, model is better. As (AR)~0, model is worse.

The accuracy ratio (AR) is defined as the ratio of the area between the model CAP and the random CAP and the area between the perfect CAP and the random CAP. For a successful model the AR has values between zero and one, with a higher value for a stronger model.

Second —

• A line is drawn from the 50% point on the X- axis up to the Model's CAP Curve.

• From the intersection point, a projection is made to the Y-axis. This value on Y-axis should lie between 50% and 100% for a successful model. Higher percentage represents how good the model is.

B. *Experimental Results*

We have taken into consideration the above mentioned metrics and compared the classifiers to obtain our results. The accuracy criterion is almost always the utmost criteria to judge different techniques. The techniques we are applying over both of our datasets are all showing high accuracies with CNN showing the highest accuracy.

A high precision rate signifies how correctly the classifiers are working. A high fractional value of recall signifies that a large number of relevant instances have been regained over the total number of relevant instances.

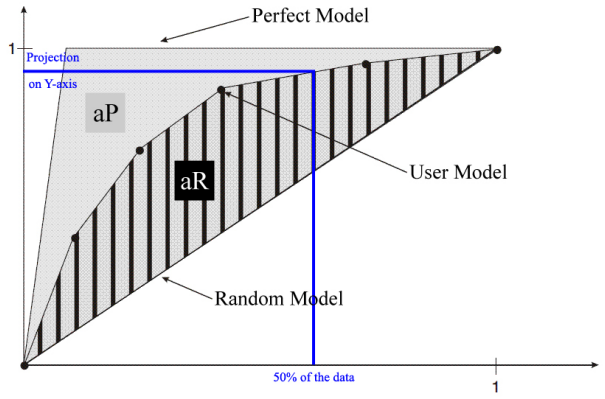

Fig. 4 Understanding CAP Curve

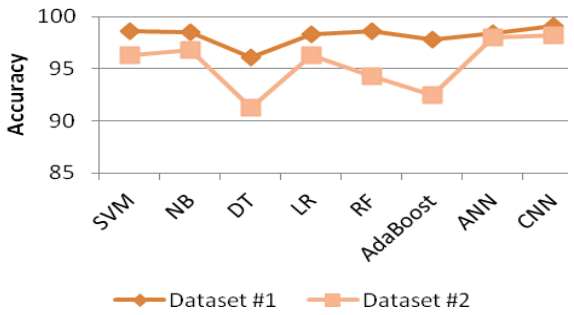

Fig. 5 Accuracy of Classifiers

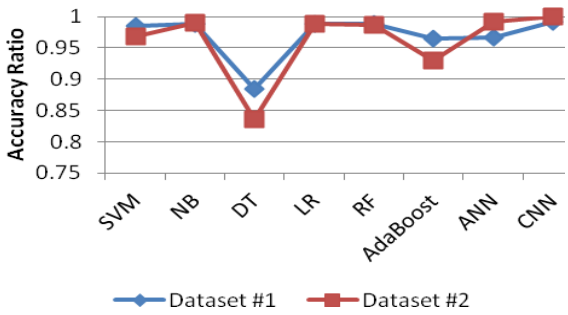

Fig. 6 Accuracy Ratio of Classifiers

TABLE I

PERFORMANCE METRICS FOR DATASET #**1**

Classifiers	Acc. (%)	Precision	Recall	AR	Positive %
Support Vector Machine	98.57	0.942	0.942	0.9846	99.280
Naive Bayes	98.48	0.904	0.993	0.9892	99.280
Decision Tree	96.05	0.797	0.913	0.8853	93.525
Logistic Regression	98.30	0.928	0.935	0.9891	99.280
Random Forest	98.65	0.992	0.898	0.9883	99.280
AdaBoost	97.85	0.952	0.869	0.9645	97.841
Artificial Neural Network	98.39	0.974	0.916	0.9667	99.393
Convolutional Neural Network	**99.10**	0.982	0.964	**0.9926**	**99.397**

TABLE II

PERFORMANCE METRICS FOR DATASET #**2**

Classifiers	Acc. (%)	Precision	Recall	AR	Positive %
Support Vector Machine	96.25	0.979	0.946	0.9689	94.6
Naive Bayes	96.75	0.975	0.961	0.9904	96.3
Decision Tree	91.25	0.943	0.883	0.8354	89.2
Logistic Regression	96.25	0.979	0.946	0.9878	95.8
Random Forest	94.25	0.984	0.902	0.9868	94.1
AdaBoost	92.50	0.953	0.897	0.9294	92.4
Artificial Neural Network	98.00	0.989	0.970	0.9918	97.0
Convolutional Neural Network	**98.25**	0.989	0.975	**0.9994**	**98.7**

For Dataset #1, Decision Tree classifier shows the least precision rate with value 0.797 and correspondingly a lower accuracy. This could be due to its greedy approach towards selecting results from the tree which is created during the process. A single data point can get a different tree which can change the classification result. On the other hand, the precision rate for Random forest is quite high at 0.992, simply because of the way in which the datasets have been split. The training been performed on 80% of the total SMS. This has been done in order to achieve full potential of the classifiers as well as maintain similar training and testing samples for all classifiers. Although the samples are selected on random basis, the ratio of splitting the dataset remains the same.

The other dataset has shown relatively linear trend for all classifiers with least precision obtained again for the Decision Tree classifier. SVM and Logistic Regression techniques have shown same precision rate at 0.979 and recall at 0.946, whereas neural networks bags the highest precision rate at 0.989 with a very good recall fraction at 0.975.

Analysis of CAP curve shows that the AR value for CNN is the highest for both datasets at 0.9926 and 0.9994, followed by Naive Bayes at 0.9892 and ANN at 0.9918 for Dataset#1 and Dataset#2 respectively. The higher value of AR corresponds to a better model which has been shown by CNN in both cases. Interestingly, the accuracy of Naive Bayes is 98.48% with AR value 0.9892, whereas the accuracy of Random Forest is higher at 98.65% with lower AR value 0.9883. The positive observations percentage at halfway mark on the X-axis is 99.28% for both. AR value is valuable to determine the discriminative power of risk models, hence preferred over accuracy. Therefore Naive Bayes should be preferred over Random Forest here.

IV. CONCLUSION AND FUTURE SCOPE

In this paper, we mainly concentrated on discussing and evaluating machine learning techniques for spam SMS detection. We ran out comparisons among 8 different classifiers. The results obtained from our evaluation of the classifiers show that Convolutional Neural Network Classifier achieves the highest accuracy of 99.19% and 98.25% and AR value of 0.9926 and 0.9994 for the two datasets. Although CNN has been generally used in the classifications of image related data, it has shown significant improvement against the traditional classifiers and achieves the highest accuracy among them for textual data as well. This achievement of CNN has wide opened the research aspect of its application over text related classification problems which involve review classification and sentiment prediction. As was expected, among traditional classifiers, SVM and NB show good results, very close to CNN for both the datasets. Significant results have been obtained from this work, corresponding to which this research can be taken to real world application level for detection of spam SMS.

REFERENCES

[1] H. Najadat, N. Abdulla, R. Abooraig, and S. Nawasrah, "Mobile SMS Spam Filtering based on Mixing Classifiers," International Journal of Advanced Computing Research, vol. 1, 2014.

[2] J. Deng, H. Xia, Y. Fu, J. Zhou, and Q. Xia, "Intelligent spam filtering for massive short message stream," COMPEL-The international journal for computation and mathematics in electrical and electronic engineering, vol. 32, pp. 586-596, 2013.

[3] K. Yadav, P. Kumaraguru, A. Goyal, A. Gupta, and V. Naik, "SMSAssassin: Crowdsourcing Driven Mobile-based System for SMS Spam Filtering", HotMobile'11 Proceedings of the 12th Workshop on Mobile Computing Systems and Applications, pp. 1-6, 2011.

[4] B. S.-E. Kim, J.-T. Jo, and S.-H. Choi, "SMS Spam Filtering Using Keyword Frequency Ratio," International Journal of Security and Its Applications, vol. 9, pp. 329-336, 2015.

[5] T. M. Mahmoud and A. M. Mahfouz, "SMS Spam Filtering Technique Based on Artificial Immune," IJCSI International Journal of Computer Science Issues, vol. 9, 2012.

[6] Y. Yang and S. Elfayoumy, "Anti-Spam Filtering Using Neural Networks and Bayesian Classifiers", Proceedings of the 2007 IEEE International Symposium on Computational Intelligence in Robotics and Automation, Jacksonville, FL, USA, June 20-23, 2007.

[7] David Ndumiyana, Munyaradzi Magomelo, and Lucy Sakala, "Spam Detection using a Neural Network Classifier," Online Journal of Physical and Environmental Science Research, vol. 2, issue 2, pp. 28-37, April 2013.

[8] T. A. Almeida, J. M. G. Hidalgo, and A. Yamakami, "Contributions to the Study of SMS Spam Filtering: New Collection and Results (preprint)," Proceedings of the 11th ACM symposium on Document engineering, Mountain View, California, USA, pp. 259-262, 2011.

[9] S. J. Delany, M. Buckley, and D. Greene, "SMS spam filtering: Methods and data," Expert Systems with Applications, vol. 39, pp. 9899-9908, 2012.

[10] I. Murynets and R. P. Jover, "Analysis of SMS Spam in Mobility Networks," International Journal of Advanced Computer Science, vol. 3, 2013.

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

[12] J. Clark, I. Koprinska, and J. Poon, "A Neural Network Based Approach to Automated E-mail Classification," Proceedings of the IEEE/WIC International Conference on Web Intelligence, Canada, 2003.

[13] X. Zhang, J. Zhao, Y. LeCun, "Character-level Convolutional Networks for Text Classification," Proceedings of the 28th International Conference on Neural Information Processing Systems, vol. 1, pp. 649- 657, 2015.

[14] SMS Spam Collection V.1 Dataset. Publicly available online at: http://www.dt.fee.unicamp.br/~tiago/smsspamcollection/.

[15] Spam SMS Dataset 2011-12. Available online on request at: http://precog.iiitd.edu.in/requester.php?dataset=smsspam.