International Journal of Intelligence Science, **2014, 4, 24-28**
Published Online January 2014 (http://www.scirp.org/journal/ijis)
http://dx.doi.org/10.4236/ijis.2014.41004

Mobile SMS Spam Filtering for Nepali Text Using Naïve Bayesian and Support Vector Machine

Tej Bahadur Shahi[1], Abhimanu Yadav[2]
[1,2]Central Department of Computer Science and Information Technology, Kathmandu, Nepal
Email: tejshahi1984@yahoo.com

Received October 16, 2013; revised November 16, 2013; accepted November 25, 2013

ABSTRACT

Spam is a universal problem with which everyone is familiar. A number of approaches are used for Spam filtering. The most common filtering technique is content-based filtering which uses the actual text of message to determine whether it is Spam or not. The content is very dynamic and it is very challenging to represent all information in a mathematical model of classification. For instance, in content-based Spam filtering, the characteristics used by the filter to identify Spam message are constantly changing over time. Naïve Bayes method represents the changing nature of message using probability theory and support vector machine (SVM) represents those using different features. These two methods of classification are efficient in different domains and the case of Nepali SMS or Text classification has not yet been in consideration; these two methods do not consider the issue and it is interesting to find out the performance of both the methods in the problem of Nepali Text classification. In this paper, the Naïve Bayes and SVM-based classification techniques are implemented to classify the Nepali SMS as Spam and non-Spam. An empirical analysis for various text cases has been done to evaluate accuracy measure of the classification methodologies used in this study. And, it is found to be 87.15% accurate in SVM and 92.74% accurate in the case of Naïve Bayes.

KEYWORDS

SMS Spam Filtering; Classification; Support Vector Machine; Naïve Bayes; Preprocessing; Feature Extraction; Nepali SMS Datasets

1. Introduction

Spam can be defined as unsolicited (unwanted, junk) email for a recipient or any email that the users do not wanted to have in their inboxes. Spam filtering is a special problem in the field of document classification and machine learning. In recent years, the technological development in mobile devices has increased in computational power, and other powerful systems have been capable to be connected to mobile phone networks. This has also increased the communication through SMS. Nobody wants the unwanted SMS on his cell phone's inbox and they want their inboxes to be free from such annoying SMS. SMS has certain characters that are different from mails. A mail consists of certain structured information such as subject, mail header, salutation, sender's address etc. but SMS lacks such structured information. These make the SMS classification task much difficult. This situation makes the necessity for developing an efficient SMS filtering method. The basic principle of Spam filtering is shown in **Figure 1**.

2. Related Work

Before 1990, some Spam prevention tools began to emerge in response to the Spammers who started to automate the process of sending Spam email. The first Spam prevention tool has used simple approach, based on language analysis by simply scanning emails for some suspicious senders or phrases like "click here to buy" and "free of charge". In late 1990s, blacklisting and whitelisting methods were implemented at the Internet Service

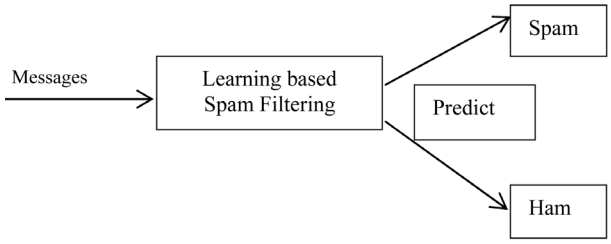

Figure 1. The basic idea of Spam filtering.

Provider (ISP) level. However, these methods suffered from some maintenance problems.

There are many efforts underway to stop the increase of Spam that plagues almost every user on the mobile network. Various techniques have been used to filter the Spam messages. Naïve Bayes [1] classifier is a simple probabilistic classifier. Its main advantage is that naïve Bayes classifiers can be trained very efficiently in a supervised learning. Naïve Bayesian classifiers are used for parameter estimation in numerous practical applications. In supervised learning, the parameters are estimated by Maximum Likelihood Estimation (MLE) method. Decision Tree [2] is one of the most famous tools of decision-making theory. Decision tree is a classifier in the form of a tree structure that shows the reasoning process. Support Vector Machines [3] is a linear maximal margin binary classifier. It can be interpreted as finding a hyper-plane in a linearly separable feature space that separates the two classes with maximum margin—the instances closest to the hyper-plane are known as the "support vectors" as they support the hyper-plane on both sides of the margin. Using these techniques, different software has been developed to filter the Spam emails. The basic concept of these techniques is the classification of SMS or email using trained classifier that can automatically predict if an incoming SMS or email is Spam or legitimate. This automatic process increases filtering performance and provides better usability than manual classification.

Some more complex approaches were also purposed against Spam problem. Most of them were implemented by using machine learning methods. A Naïve Bayes algorithm is used frequently which has shown a considerable success in filtering Spam e-mails in English [4]. Knowledge-based and rule-based systems were also used by researchers for English Spam filters [5,6]. SVM is used for text classification [7], which can also be applied for Spam filtering.

There is no work done for Nepali text SMS Spam filtering yet and it is much more necessary to start the work. The resource such as training SMS corpus is also not available for Nepali language and the corpus used in this work is created manually. The training corpus developed during this study can be made available for research proposes.

3. Methodology: A Proposed Framework for Spam SMS Filtering

Spam filtering engine flowchart is given in Figure 2.

This describes top level data flow diagram of Spam classification problem used in this research work. The proposed system framework contains three steps: preprocessing, feature extraction and classification.

3.1. Preprocessing

The purpose of pre-processing is to transform messages in SMS into a uniform format that can be understood by the learning algorithm. The first step of text mining process is text pre-processing in which the collection of documents is analysed syntactically or semantically. The text message document is considered as a bag of words because the words and their occurrences are used to represent the document. The algorithm applied in this stage are stemming and stop word removal, number removal and strip whitespaces.

3.2. TF-IDF Calculation and Feature Vector Construction

In this work, the most widely adopted feature weighting scheme known as TF-IDF scheme, in Information Retrieval (IR), TF-IDF, to represent the email as a vector in a vector space model, and it is calculated as Equation (1):

$$a_{ij} = \frac{tf_{ij} \cdot \log \frac{|D|}{DF_i}}{\sqrt{\sum_k \left(tf_{kj} \cdot \log \frac{|D|}{DF_k} \right)^2}} \quad (1)$$

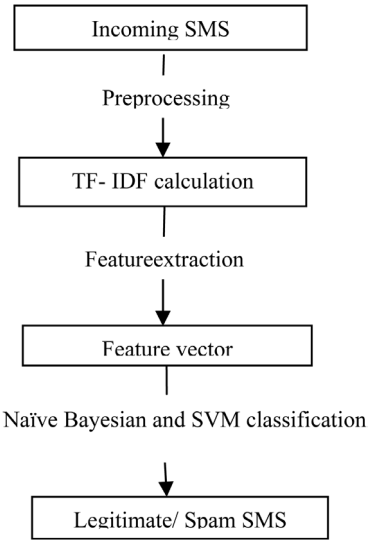

Figure 2. Framework for Spam filtering.

where tf_{ij} is SMS in the training set and DF_i is the number of SMS, containing the term i. The importance of a term in a SMS is measured by the frequency and its inverse document frequency.

3.3. Classification

Consider the problem of classifying documents or message (SMS) by their content, for example, into Spam and Non-Spam Messages. A document is drawn from set of documents (Spam and Non-Spam) which can be modeled as sets of words.

The (independent) probability that the i^{th} word of a given document occurs in a document from class C can be written as $p(w_i|C)$.

Then the probability that a given document D contains all of the words w_i, given a class C,

$$p(D|C) = \prod_i p(w_i|C) \quad (2)$$

Now by definition

$$p(D|C) = \frac{p(D \cap C)}{p(C)} \quad (3)$$

And

$$p(C|D) = \frac{p(D \cap C)}{p(D)} \quad (4)$$

Bayes' theorem manipulates these into a statement of probability in terms of likelihood

$$p(C|D) = \frac{p(C)}{p(D)} p(D|C) \quad (5)$$

Assume for the moment that there are only two mutually exclusive classes, S and $\neg S$ (i.e. Spam and not Spam), such that every element (message) is in either one or the other:

$$p(D|S) = \prod_i p(w_i|S) \quad (6)$$

And

$$p(D|\neg S) = \prod_i p(w_i|\neg S) \quad (7)$$

Using the Bayesian result above

$$p(S|D) = \frac{p(S)}{p(D)} \prod_i p(w_i|S) \quad (8)$$

And

$$p(\neg S|D) = \frac{p(\neg S)}{p(D)} \prod_i p(w_i|\neg S) \quad (9)$$

Finally the document can be classified as follows. It is Spam if $p(S|D) > p(\neg S|D)$

In their basic form shown in **Figure 3**, SVM construct

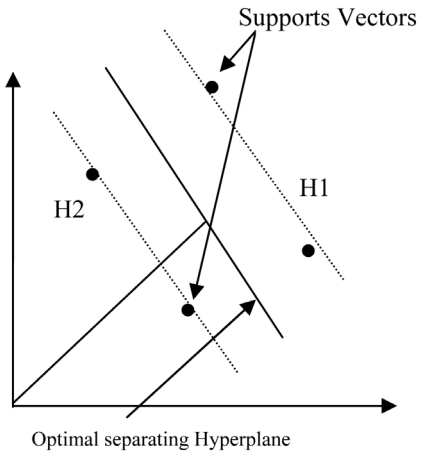

Figure 3. Support vector machine.

the hyper-plane in input space that correctly separate the example data into two classes. Hence, SVM is a binary classifier. This hyper-plane can be used to make the prediction of class for unseen data. The hyper-plane always exists for the linearly separable data [8]. Each SMS is converted into feature vector on the Bag of word basis and the length of feature vector is equal to number of words in the Dictionary. The Dictionary consists of feature word from the training corpus. Some frequent Spam words are also included in dictionary.

4. Experimental Setup and Results

Java programming language is used for the implementation of the proposed framework. SVM light [9] is used for as classification tool for SVM and Naïve Bayes is implemented in Java.

Naïve Bayes and Support Vector Machine algorithms have been implemented for the Spam filtering task. The study has gone through the empirical analysis of the performance of both the Spam filters (SVM and Naïve Bayes) for Nepali SMS. It is observed from the experiment that the Spam Filter based on Naïve Bayes outperforms the Spam Filter based on SVM. Extensive tests have been performed with varying numbers of data set sizes. The success rates reach their maximum using all the messages and all the words in training corpus.

Tables 1-3 show the results of experiment and it is shown that the learning methods perform well when they are trained using more examples.

5. Conclusions and Future Work

The main concern for this study was to examine the efficiency of Naïve Bayesian and SVM Spam filters. The comparison of efficiency between these Spam filters was done on the basis of the accuracy, precision and recall. This comparison helps to find the best algorithm for Spam filtering.

Table 1. SVM classification results.

No. of test	Messages (Spam/Non-Spam)	SVM			
		(Correct/Incorrect) SMS	Accuracy	Precision	Recall
1	10	(8/2)	80%	77.78%	100%
2	30	(26/4)	86.67%	83.33%	100%
3	50	(42/8)	84%	78.95%	100%
4	68	(59/9)	86.76%	80.85%	100%
5	90	(81/9)	90%	86.96%	100%
6	110	(99/11)	90%	86.42%	100%
7	150	(139/11)	92.67%	89.11%	100%

Table 2. Naïve Bayes

No. of test	Messages (Spam/Non-Spam)	Naïve Bayes			
		(Correct/Incorrect) SMS	Accuracy	Precision	Recall
1	10	(9/1)	90%	100%	75%
2	30	(28/2)	93.33%	100%	83.33%
3	50	(45/5)	90%	90%	85.71%
4	68	(63/5)	92.65%	93.33%	90.32%
5	90	(84/6)	93.33%	93.33%	87.5%
6	110	(104/6)	94.54%	95%	90.47%
7	150	(143/7)	95.33%	96.67%	92.06%

Table 3. Comparative results of SVM and Naïve Bayes.

No. of test	Messages (Spam and non-Spam)	Accuracy	
		SVM	Naïve Bayes
1	10	80%	90%
2	30	86.67%	93.33%
3	50	84%	90%
4	68	86.76%	92.65%
5	90	90%	93.33%
6	110	90%	94.54%
7	150	92.67%	95.33%

The classification accuracy of 92.74% was obtained for the Naïve Bayes classifier and 87.15% accuracy was obtained for SVM classifier on Nepali Spam dataset. On the basis of accuracy, Naïve Bayes is a better classification technique than SVM-based classifier.

No hundred percent filtering Spam system is invented till now. The classification accuracy of Naïve Bayesian and SVM proposed in this research work, however, can be further improved. Here, the TF-IDF scheme was used to make feature vector, which did not consider the individual word in SMS. *i.e.*, it only considers the weighted words. Some techniques that use context base features can be used.

The features used to convert given Spam into vector can be enriched so that the higher accuracy can be achieved. Due to the small SMS corpus size, there is the

unknown word problem in Naïve Bayes classifier. Hence, some other techniques to handle the unknown word can be used. The size of SMS corpus can be increased by collecting more real SMS in the future.

Acknowledgements

Authors would like to thank Dr. Shashidhar Ram Joshi, Professor of Computer Science at institute of engineering, Pulchwok and Asst. Prof. Dr. Sanjib Pandey for their supervision during the completion of this work.

REFERENCES

[1] M. Sahami, S. Dumais, D. Heckerman and E. Horvitz, "A Bayesian Approach to Filtering Junk E-mail," Learning for Text Categorization Papers from the AAAI Workshop, 1998, pp. 55-62.

[2] S. Carrerasx. and L. Marquez, "Boosting Trees for Anti-Spam Email Filtering," *Proceeding of RANLP*, Tizigov-chark, 5-7 September 2001, pp. 58-64.

[3] H. Drucker, D. H. Wu and V. N. Vapnik, "Support Vector Machines for Spam Categorization," *IEEE Transaction on Neural Networks*, Vol. 10, No. 5, 1999, pp. 1048-1054. http://dx.doi.org/10.1109/72.788645

[4] I. Androutsopoulos and J. Koutsias, "An Evaluation of Naive Bayesian Networks," Machine Learning in the New Information Age, Barcelona, 2000, pp. 9-17.

[5] W. Cohen, "Learning Rules That Classify E-Mail," *AAAI Spring Symposium on Machine Learning in Information Access*, Stanford, 25-27 March 1996, pp. 18-25.

[6] C. Apte, F. Damerau and S. M. Weiss, "Automated Learning of Decision Rules for Text Categorization," *ACM Transactions on Information Systems*, Vol. 12, No. 3, 1994, pp. 233-251. http://dx.doi.org/10.1145/183422.183423

[7] T. A. Almeida, J. M. G. Hidalgo and A. Yamakami, "Contributions to the Study of SMS Spam Filtering: New Collection and Results," *ACM Transection on Information System*, Mountain View, 19-22 September 2011, pp. 20-25.

[8] P. Graham, "A Plan for Spam," 2002. http://www.paulgraham.com/Spam.html

[9] V. T. Joachims, "Making Large-Scale SVM Learning Practical," In: B. Schölkopf, C. Burges and A. Smola, Eds., *Advances in Kernel Methods Support Vector Learning*, MIT-Press, Cambridge, 1999.