Content-based SMS Spam Filtering with Machine Learning and Multi-layer Perceptron Neural Network

Mengxuan Zhao
CUNY GC, Linguistics / 365 Fifth Ave, New York, NY, USA
mzhao@gradcenter.cuny.edu

Abstract

The problem of content-based Short Message Services (SMS) spam filtering has been discussed for years as a natural language processing topic. Researchers have applied machine learning algorithms and built classifiers, and several of them, including Naive Bayes, Support Vector Machine(SVM) and K-Nearest Neighbors(K-NN) have achieved satisfying results. At the same time, both Bag-of-Word(BoW) features and structural(statistical) features play an significant role in those classification tasks. However, even though being discussed comparably with email spam filtering tasks, researchers have not applied neural network on SMS spam filtering yet. In this paper, I use a corpus of 5,574 tagged text messages, build several classifiers including a multi-layer perceptron neural network, and discuss how feature extraction influences the results.

1 Introduction

Comparing to similar classification tasks such as email spam filtering, SMS spam filtering deals with its specific problems. Text messages are short. The content does not strictly respect to grammar, punctuation or spelling rules. A lack of consistency in vocabulary is observed because of a number of abbreviations, acronyms and non-alphanumeric symbols. In other words, to extract enough information from the content to feed the classifier could be troublesome. Apart from that, the definition of a 'spam' could also be subjective. It is hard to define to what extent a piece of text is 'annoying' to its receiver.

2 Related Work

The problem of SMS spam has drawn researcher's attention for long time. There are numerous researches aiming at spam message filtering with machine learning classification methods. Xiang *et. al.* (Xiang et al., 2004) propose that SVM is a effective method to be deployed in spam filtering system, based on their experiment result of an accuracy of 96.11%.

Healy *et. al.* (Healy et al., 2005) adopt word features, statistical and single character features, represented all as binary features (1 for such feature existing, 0 for not), and apply K-NN classifier. They test with a customer comments corpus and a SMS corpus, which consist of two datasets with 100 legitimate and 100 spam in each. Result show that the classifier perform the best when k=3, with Error=4.7% and FP=6.5%.

Gómez Hidalgo *et. al.* (Hidalgo et al., 2006) introduce a Spanish database which includes 199 spam messages and 1,157 legitimate messages, and an English database including 82 spam messages and 1,119 legitimate messages. They use character bi-grams and tri-grams as well as orthogonal sparse word bi-grams of lowercase alphanumeric words as features, and apply Naive Bayse, Decision Trees and SVM to build up classification models. They claim that SVM is the optimal classifier, which achieves over 95% accuracy. Moreover, they discuss how previous researches on spam email filtering and Bayesian filtering techniques help with SMS spam filtering work.

Cormack, Gómez Hidalgo and Puertas Sánz (Cormack et al., 2007a) propose that adapted email spam filters can be applied on SMS, and they test several machine learning based models (including Bayesian, Logistic Regression, SVM, and etc.) built for email filtering on Gómez Hidalgo *et. al.* (Hidalgo et al., 2006)'s datasets. They assert that the differences among filters are not clear and experiments with larger dataset are expected.

In one of the later works by the same group (Cormack et al., 2007b), experiments are applied

with SMS, blog comments and emails, with the SMS corpus including 1,002 legitimate messages and 322 spam, which is larger than their previous collection. Feature expansion greatly improved the accuracy. 1-ACC (%) score for SVM and Logistic Regression models are 0.0502 and 0.0806 [1].

Yadav *et. al.* (Yadav et al., 2011) collect their database with 2,195 spam and 2,123 legitimate with crowdsourcing. Both topical and stylistic features are taken into consideration, while to relieve computational intensity, only 20 most distinctive features are applied (count of spam words, count of '\', average word length, to name a few). The dataset is English and Hindi combined. Results show 1-ACC(%) value was 5.61 with SVM.

Almeida, Gómez Hidalgo and Yamakami (Almeida et al., 2011) contribute a new dataset available on-line[2], which includes 4,827 legitimate and 747 spam. This dataset is what my experiments are based on. They also provide some statistics about tokens. This dataset is the largest so far. Two types of features are tried, 'tok1' for alphanumeric characters excluding dots, commas and colons from the middle, and 'tok2' for any sequence of characters separated by blanks, tabs, returns, dots, commas, colons and dashes. The first pattern keeps domain names in URLs and email addresses. They apply both features with a number of machine learning methods, and results show the highest accuracy (97.64%) in SVM with 'tok1' features.

Uysal *et. al.* (Uysal et al., 2013) focus on how feature extraction influences the performance of classifiers. They test on both Turkish and English datasets, the latter included 425 spam and 450 legitimate. They tried both BoW features and 6 different structural features including message length, number of terms, uppercase character ratio, non-alphanumeric character ratio, numeric character ratio and presence of URL. They select features with Chi-square and Gini index, and apply K-NN and SVM algorithms. Results show that 100% entire BoW features plus all structural features perform the best with SVM, with micro-f1 score achieve 0.96.

Since text messages are short and usually filled with slangs, idioms, symbols and acronyms, it is especially difficult to extract enough information from the text to feed the classifier. Almeida and Gómez Hidalgo (Almeida et al., 2016) propose several methods to normalize and expand the raw text so as to enhance the classification performance. They test on different classifiers including Decision Tree, Naive Bayes, K-NN (K=1,3,5), linear SVM and Logistic Regression. Comparing to the original text, performance of normalized text is statistically significantly improved.

Artificial Neural Network has been widely applied in email spam filtering for years, yet similar tasks done with SMS spam filtering is not found. To compare the effectiveness of Neural Network classification model with other commonly-used machine learning model, I refer the work of Sesha Rao *et. al.* (Rao et al., 2016). They build up a two-hidden-layer perceptron, with 46 nodes as input layer, 8 nodes and 6 nodes in each hidden layer, and 2 nodes as output layer. Each neuron consists of a summation function and a sigmoid function as activation function. The dataset includes 4,501 emails, and 1,813 of them are spam. Among several different Neural Network models, the highest precision and recall scores are 0.992 and 0.906 respectively.

Other papers related to SMS spam filtering are not included in this review, because they focus on corpus other than English, or they apply non-content-based methods.

3 Experiment

This paper does not focus on the problem of defining a 'spam', hence I adopt an existing corpus and follow the given tags. The corpus is released by Almeida, Gómez Hidalgo and Yamakami (Almeida et al., 2011). It consists of 5,572 text messages, in which 747 are spam, tagged as 'spam', and 4,827 are legitimate, tagged as 'ham'. The corpus is in English, while some non-English characters can be seen sparsely.

3.1 Features

Two types of features are adopted in the experiments; BoW features and structural features. For BoW features, I use uni-grams, with all the words lowercased and all the punctuation removed. Stop words are not removed, because firstly, text messages are not strictly respect to grammar rules, and in many cases functional words are omitted for typing convenience and secondly, spelling rules

[1] Although there is the percentage mark in '1-ACC (%)', they did not use percentage to represent the results. Instead they used decimal numbers.

[2] https://www.kaggle.com/uciml/sms-spam-collection-dataset

are not strictly obeyed, which means words can be spelled in simplified customs. For either reason, without normalizing the text, it is tremendously hard to shoot all targets of stop words. Word features are presented as count in a sparse word-document matrix.

The idea of structural (statistical, stylistic) features is mentioned in several works.[3] Generally, such features deal with statistical data related to text length, number of words, ratio of specific characters, etc. I try the six structural features used in Uysal *et. al.* (Uysal et al., 2013): length of text, number of terms, uppercase character ratio, non-alphanumeric character ratio, numeric character ratio and presence of URL. Among them, the first two are counts, the latter three are ratio and the last one is binary.

3.2 Tools and Methods

I use four different classifiers: Naive Bayes, K-NN (K=1), Linear SVM and multi-layer perceptron Neural Network. The first three are commonly-used machine learning classifiers that previous researchers have conducted. I build the neural network model to make comparison. The data is split into three parts: 70% for training, 20% for testing, and both sets are used in all of the four classifiers, and another 10% as development which is used only in Neural network model.

I use the MultinomialNB(), KNeighborsClassicier(n_neighbors=1) and SVC(kernel='linear') in Scikit-learn package and Tensorflow package for building up the classifiers. The Neural Network model includes three hidden layers, with the number of dimensions of the uni-gram word-document matrix as the number of input layer neurons, 8 neurons in each hidden layer, and 2 in output layer. In each layer, a sigmoid function is applied as shown below:

$$F(x) = W \cdot x + B \tag{1}$$

$$S_n(x) = \frac{1}{1 + e^{-F_n(x)}} \tag{2}$$

where n is the index of the layer: i for input layer, 1,2,3 for the hidden layers, and o for output layer. L2 loss is calculated as:

$$Loss = \Sigma(y - S_o(x))^2 \tag{3}$$

[3] In (Rao et al., 2016) it is called 'structural features', in (Healy et al., 2005) it is called 'statistical features', and in (Yadav et al., 2011) it is called 'stylistic features'. In this paper I adopt the term 'structural features'.

		Precision	Recall	F1 Score
NB	M1	0.951	**0.945**	0.948
	M2	0.962	0.862	0.909
	M3	0.955	0.869	0.91
	M4	0.951	0.931	0.941
K-NN	M1	0.988	0.579	0.73
	M2	0.768	0.89	0.824
	M3	0.557	0.876	0.681
	M4	0.978	0.6	0.744
SVM	M1	**0.992**	0.89	0.938
	M2	0.964	0.91	0.936
	M3	0.963	0.903	0.932
	M4	**0.992**	0.89	0.938
NN	M1	0.971	0.931	0.951
	M2	0.97	0.897	0.932
	M3	0.965	**0.945**	**0.955**
	M4	0.978	0.924	0.95

Table 1: Results

where y represents the golden standard. Tensorflow function FtrlOptimizer is applied to reduced the loss and update the weight and bias.

4 Evaluation and Discussion

Four models are built for each classifier, and results are shown in Table 1. In model 1, only unigram features are included. In model 2, both unigrams and all of the six structural features are included. In models 3 and 4, I use uni-grams and one structural feature for each: message length for M3 and numeric character ratio for M4 respectively.

Overall, the best result is seen in Neural Network M3, with the highest f1 score as 0.955, and a comparatively high precision of 0.965 and recall of 0.945. The highest precision is observed in SVM M1 and M4(0.992), and the highest recall is observed in Naive Bayes M1 and Neural Network M3(0.945).

Generally speaking, K-NN model performs the worst. Uni-gram features perform excellently in each of the four models. When all of the six structural features are added, it ruins the results. However, when the chosen ones is added separately, the results are improved in different extent.

Uysal *et. al.* (Uysal et al., 2013) report the best performance with both BoW features and all six structural features. However, the opposite is observed in my experiment. M2 performs either worse than or not quite differently as the other three models in Naive Bayes, SVM and Neural

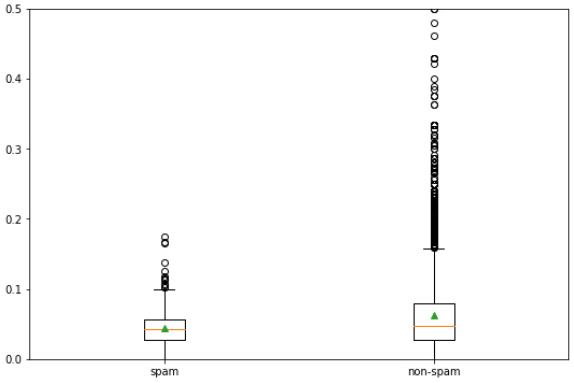

Figure 1: non-alphanumeric character ratio for spam and non-spam

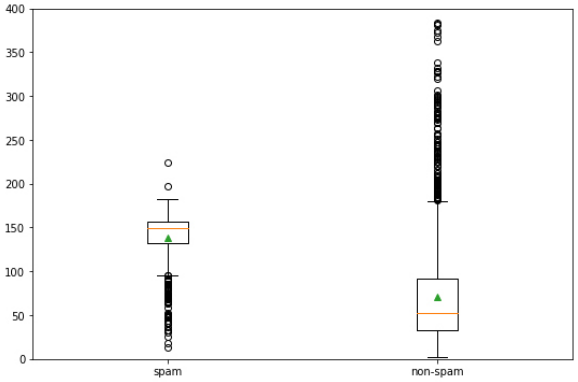

Figure 2: message length for spam and non-spam

Network models. Even thought K-NN M2 outperforms other K-NN models, it does not beat any other classifiers.

The corpus Uysal *et. al.* adopt is much smaller than the one in my experiments. More diversity is observed in latter one. Not all of the six features are distinctive enough (see Figure 1). At the same time, those distinctive features can be very noisy (see Figure 2). That explains why the results are corroded with all six features all-together yet improved with only the distinctive ones.

5 Conclusion and Future Work

Uni-gram features perform excellently in the classification of spam and non-spam text messages. Structural features, if cautiously chosen, help improve the performance of classification. In addition, comparing to other commonly-used machine learning classification models, multi-layer perceptron Neural Network model performs optimally under same conditions.

Text normalization is not discussed in this paper. In terms of noise reduction and dimension reduction, text normalization should hopefully bring even better classification accuracy.

References

T. A. Almeida, J. M. Gómez Hidalgo, and A. Yamakami. 2011. Contributions to the study of sms spam filtering: New collection and results. In *Proceedings of the 11th ACM symposium on Document engineering*, pages 259–262.

T. A. Almeida, T. P. Silvaa, I. Santos, and J. M. Gmez Hidalgo. 2016. Text normalization and semantic indexing to enhance instant messaging and sms spam filtering. *Knowledge-Based Systems*, 108:25–32.

G. V. Cormack, J. M. Gómez Hidalgo, and E. Puertas Sánz. 2007a. Feature engineering for mobile (sms) spam filtering. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 871–872.

G. V. Cormack, J. M. Gómez Hidalgo, and E. Puertas Sánz. 2007b. Spam filtering for short messages. In *Proceedings of the 16th ACM Conference on Conference on Information and Knowledge Management*, pages 313–320.

M. Healy, S. J. Delany, and A. Zamolotskikh. 2005. An assessment of case-based reasoning for short text message classification. In *Proceedings of 16th Irish Conference on Artificial Intelligence and Cognitive Science*, pages 257–266.

J. M. Gómez Hidalgo, G. Cajigas Bringas, E. Puertas S/'anz, and F. Carrero Garc/'ia. 2006. Content based sms spam filtering. In *Proceedings of the 2006 ACM Symposium on Document Engineering*, pages 107–114.

A.Sesha Rao, P.S.Avadhani, and Nandita Bhanja Chaudhuri. 2016. A content-based spam e-mail filtering approach using multilayer percepton neural networks. *International Journal of Engineering Trends and Technology*, 41(1):44–55.

A. K. Uysal, S. Gunal, S. Ergin, and E. Sora Gunal. 2013. The impact of feature extraction and selection on sms spam filtering. *ELEKTRONIKA IR ELEKTROTECHNIKA*, 19(5):67–72.

Y. Xiang, M. U. Chowdhury, and S. Ali. 2004. Filtering mobile spam by support vector machine. In *Proceedings of the Third International Conference on Computer Science, Software Engineering, Information Technology, E-business and Applications*, pages 1–4.

K. Yadav, P. Kumaraguru, A. Goyal, A. Gupta, and V. Naik. 2011. SMSAssassin: Crowdsourcing driven mobile-based system for sms spam filtering. In *Proceedings of the 12th Workshop on Mobile Computing Systems and Applications*, pages 1–6.