Using NLP Techniques and Model Selection to Improve Performance of SMS Spam Classification

Marin Krešo, Matteo Miloš, Josip Renić

University of Zagreb, Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb, Croatia
{marin.kreso,matteo.milos,josip.renic}@fer.hr

Abstract

Recent reports clearly indicate dramatic growth in volume of SMS spam messages. SMS spam classification is a challenging problem, as this kind of messages are rife with idioms and abbreviations. Most common and baseline solution for this is using Multinomial Naive Bayes algorithm with Bag-of-words term frequencies as features. As an alternative, we propose pipeline approach that uses NLP (Natural Language Processing) techniques, extracts new features and does hyperparameter optimization for most popular machine learning classification algorithms. Our results on the SMS Spam Collection dataset show that by incorporating our proposed pipeline approach, SMS spam classification system can yield statistically significant performance gain as compared to the baseline.

1. Introduction

SMS, which stands for short message service, is one of the simplest and most popular ways of communication between people. It is a communication service component of phone, web or mobile communication systems and it uses standardized communication protocols that enable mobile devices to exchange short text messages. Although SMS messages were initially designed only to send textual content, today we can send image, video and audio content as well. While SMS already is huge market, it is still growing, although there is numerous competition, that use IP-based messaging service, such as WhatsApp, Viber, iMessage etc. One of the reasons SMS messaging is still so popular, is because of its' cheap price. For example, an actual cost of sending and SMS in Australia was approximately $0.00016 in 2015, and today it is even cheaper. However, industry earning doesn't suffer despite cheap prices, because there are, according to Portio Research, approximately 16 million SMS sent per minute. In addition to regular usage, for day-to-day informal communication between people, SMS has also become a massive commercial industry. At the end of 2016, there were about 37 million consumers that opted-in to receive business SMS, and it is predicted that by 2020 that number will reach 49 million. Despite all the benefits SMS communication offers, with years and price of messaging going down, some downsides also came up. The main downside we can highlight is emerging phenomenon of SMS spam. As SMS spam we can classify any message that wasn't wanted and solicited from user. One of the things that contributes to growth of SMS spam is already mentioned cheap price of SMS, so it is almost costless for spammers to send malicious messages. Except the main reason, which are annoying and potentially dangerous messages, one of the problems is that some carriers still require from their users to pay to receive text messages. Also, there are numerous possibilities for financial frauds, and reportedly, in 2016 Canadians more than $600000, which is about 5 times more than 2 years earlier. All mentioned problems with today's SMS communication leaded us to designing classification algorithm, that can be used in classifying the messages either to ham or spam messages. While we already briefly explained what spam messages are, ham messages we can describe as the exact opposite, because they are regular messages, either formal or informal, that user is happy to receive. Our main goal was to improve results achieved by using Multinomial Naive Bayes, which we used as baseline model for our approach. While many previous studies oriented purely on machine learning algorithms, our approach included using the NLP techniques. To produce statistically better results with our model, we incorporated NLP techniques such as stemming, stop words removal, feature extraction and tokenization This paper will be organized in following sections. In Section 2 we will discuss previous works on this matter, Section 3 offers us overview on data we are going to use. In Section 4 we will present our methodology that will be used for classification and in Section 5 we are going to conduct actual experiment. In the last section, finally we are going to discuss conclusion and potential for future work.

2. Related Work

Through the recent years, there were multiple attempts that tried to develop various techniques for successful classification of SMS spam and ham messages. Everything started in 2004, when (Xiang and Ali, 2004), decided to used Support Vector Machines for classification. Same method was used by (Gomez Hidalgo et al., 2006), with addition of selecting tokens using Information Gain. Next step forward was in 2009, when (Duan et al., 2009) used k-NN (k-nearest neighbor) algorithm along with other spam detection methods. (Almeida et al., 2011) used 13 different classification algorithms on a dataset that contained 5500 SMS messages aggregated from multiple sources. They concluded that the best performing algorithm was SVM combined with alphanumeric tokenization. Throughout the years, although the results were good, few limitations were recognized. Despite of high accuracy, there was noted also a high false-positive rate and a low efficiency, which is caused by using large number of input features. Our goal was to address these limitations and try to improve the results, either by optimizing algorithm such as SVM, or using some completely different classification algorithm.

3. Data

As in any scientific research, we had to make sure we have reliable and representative data. Because there is lack of quality SMS spam datasets, we decided to use one which was created for previous scientific work (Almeida et al., 2011). That dataset, which is publicly accessible at KAGGLE[1], is derived from multiple trustworthy sources.

Finding trustworthy SMS spam messages is a lot harder job than finding the ham ones. Therefore, as a first source, they used The Grumbletext Web site[2], which contained 425 manually classified spam messages. Grumbletext is a UK forum in which cell phone users make public claims about SMS spam messages. However, most of them do that without reporting the very spam message received, so identification of spam messages was very hard and time-consuming task. Their next source was, NUS SMS Corpus[3], dataset od 10000 ham messages collected for research at the Department of Computer Science at the National University of Singapore. A subset of 3375 SMS ham messages was randomly chosen, and all the contributors to the dataset were made aware that their contributions were going to be made publicly available. Also, a list of 450 SMS ham messages was collected from Caroline Tag's PhD Thesis[4]. As as a final source, authors incorporated the SMS Spam Corpus v.0.1 Big[5]. That is a dataset that contains 1002 ham and 322 spam messages.

Using all mentioned sources, we can conclude that our dataset contains a total of 5574 messages, of which 4827 are ham, and the rest, 747, are spam messages. As far as our knowledge goes, that dataset is the largest SMS corpus available, that contains both spam and ham messages, and therefore we determined that it is representative example for our research.

4. Proposed Methodology

Standard approach is to use Multinomial Naive Bayes with Bag-of-Words model for spam/ham detection. This model is expected to produce good results without any NLP and hyperparameter optimization. Goal of our system is to produce statistically significant better results with two different tokenization approaches. Figure 1 shows steps involved in each system.

4.1. Tokenization

Due to limited SMS length people compress their message in many forms. SMS is usually filled with colloquial and slang speech e.g., ”Nah”, ”tkts”, ”k”, ”lol”, etc. From 13654 unique tokens observed over whole dataset only 24.06% tokens appears in set of English words from

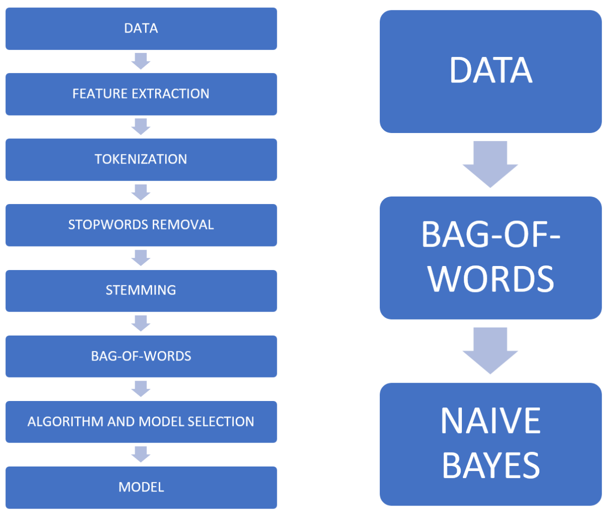

Figure 1: Proposed system pipline variations.

NLTK[6] Wordlist Corpora. People tend to lose formal grammatical structure and misspell words when writing SMS. This finding conveys us to peruse two different tokenization approaches, first standard tokenizer and second one to tackle misspelled words:

- tok1 - tokens created by white space elimination, for sentence ”Lol your” we would get tokens ”Lol” and ”your”.
- tok2 - tokens created from characters from lowercased sentence with n-gram range of (2,5), for sentence ”Lol your” we would get tokens 22 tokens, e.g. ”lo”, ”lol”, ”lol ”, ”lol y”, etc.

For tok2 n-gram range (2,5) was chosen as best performing n-gram range after extensive search.

4.2. Stop words

After tokenization with *tok1* tokenizer, we analyzed most common tokens which turned out to be stop words. As well, for spam and ham messages most common tokens are stop words. Due to nature of spam massages we would expect most common tokens to be ”call”, ”free” and ”win”. Next step in our system pipeline is stop words removal, which can only be performed with *tok1* tokenizer. We used standards NLTK corpus stop words. We can see most common tokens after stop word removal in Table 1.

4.3. Stemming

Words appear in natural language in many forms:

- Inflection - adding a suffix to a word, that doesn't change its grammatical category, such as tenses in verbs (-ing, -ed, -s), plural in nouns (s).
- Derivation - adding a suffix to a word, that changes its grammatical category, such as nation (noun) = national (adjective) = nationalize (verb).

[1]`http://www.kaggle.com`
[2]`http://www.grumbletext.co.uk/`
[3]`http://www.comp.nus.edu.sg/~rpnlpir/downloads/corpora/smsCorpus/`
[4]`http://etheses.bham.ac.uk/253/1/Tagg09P hD.pdf/`
[5]`http://www.esp.uem.es/jmgomez/smsspamcorpus/`
[6]`http://www.nltk.org`

Table 1: 10 most common tokens

all data	spam pruned	ham pruned
.	call	im
to	free	im
I	2	2
you	txt	get
	u	ltgt
?	ur	ok
!	mobile	go
a	text	ur
...	4	got
the	stop	ill

The goal of stemming is to reduce such forms of words to their common base form, which could be useful for our task of classification of spam in SMS messages. Most of the current spam filters use keywords to detect spams. These keywords can be misspelled, even on purpose to avoid detection. It is impossible to predict all possible misspellings for some keywords, so that is where our stemmer comes in handy. It is important to notice that stem of some word doesn't need to be identical to the morphological root of the word. It is usually sufficient that related words map to the same stem, even if the stem itself is not a morphologically valid root. Stemming is typically done by suffix stripping plus some extra steps and checks.
Algorithms for stemming have been studied in computer science since the 1960s. For our paper, we are going to use on the most popular stemming algorithms - Porter Stemmer. Porter's algorithm consists of 5 phases of word reductions :

- Phase 1 deals with plurals and past participles.
- Phase 2-4 deal with derivation of words.
- Final phase for tidying up.

Another approach to solve the issue with many forms of words is lemmatization, but we are not considering this in our paper.

4.4. Feature extraction

Due to SMS limited size and average length being 80 characters, given that largest observed SMS has 910 characters, we could benefit from additional information in form of hand-crafted features. In next sections we will explain proposed features. Besides proposed features we explored additional ideas for features as: number of punctuations, number of capital words, number of lowercase words and ratio of correct English words. These features did not improve performance and their presence in ham and spam was equal.

4.4.1. Length

Spammers usually use longer messages average spam length is 139 while average ham 72 characters. Our explanation is that spam messages are context free, there is no previous conversation, so they need more words to convey

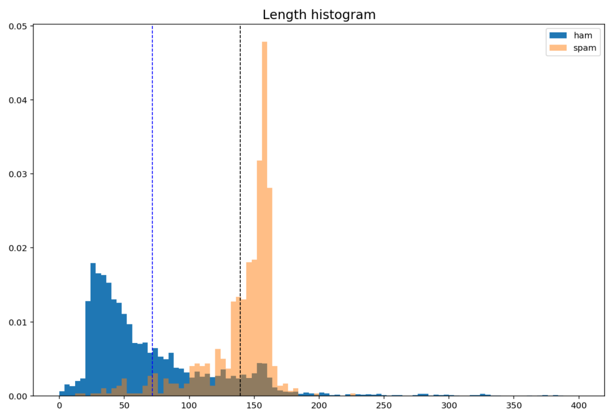

Figure 2: Spam and ham character length histogram with corresponding averages.

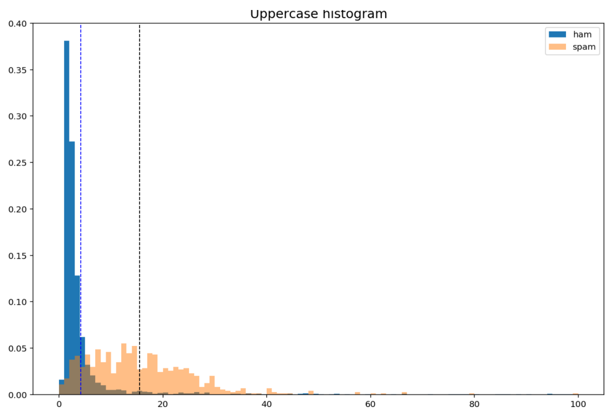

Figure 3: Spam and ham uppercase character histogram with corresponding averages.

their message. While ham message does not suffer from lack of context. Figure 2 shows us histogram of spam and ham length histogram. We can see clear separation between ham and spam in length, which proves that length would be good feature candidate.

4.4.2. Number of Upper Characters

After observing spam messages, we found one typical spam characteristics, usage of large amounts or uppercase characters. Spam messages tend to include all capital words like: "WINNER", "WIN", "FREE","CALL" and "URGENT". There are exceptions in ham messages that contain all capital words which tends to be common in SMS when people express urgency or excitement. Average upper characters is 4.17 and ham average is 15.48. Figure 3 shows us histogram of spam and ham upper characters histogram. We can see distinction for large amounts of spam messages while ham messages tend to contain very small to none number of upper characters.

4.4.3. Number of Numeric Characters

When we think of SMS spam we think of typical prize money award message that wants us to call some number. To cover those messages, we introduce number of numeric

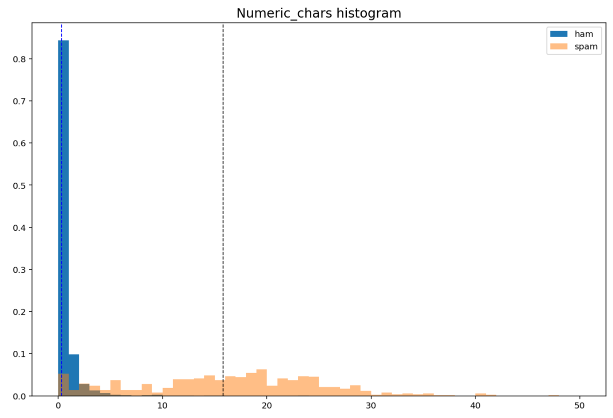

Figure 4: Spam and ham numeric character histogram with corresponding averages.

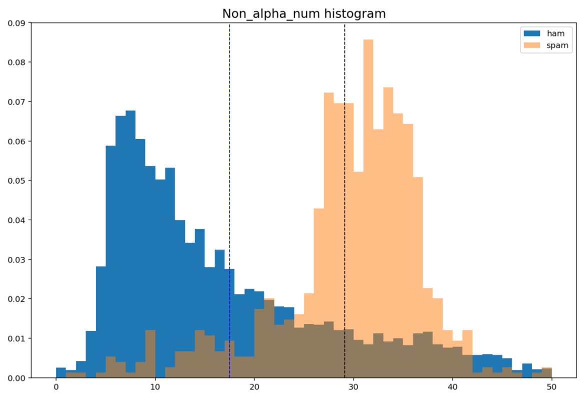

Figure 5: Spam and ham non-alphanumeric character ratio histogram with corresponding averages.

characters feature. Average numeric characters in spam messages is 15.81 and for ham messages is 0.3. We can see clear distinction between ham and spam messages. Figure 4 shows us that almost all ham messages contain some to none numeric characters while most of spam messages contain a lot more.

4.4.4. Number of Non-Alphanumeric Characters

Spam messages tend to use more special characters. One explanation is that they need to be more interesting and flashy. On the other hand, people sending ham messages need to convey their message in limited amount of space and do not use non-alphanumeric characters a lot. As well some mobile phones have limited support for special characters. Spam average number of non-alphanumeric characters is 17.5 and for ham is 23.1. Figure 5 shows clear bimodality in non-alphanumeric characters distribution which supports our claim as a good feature.

4.5. Bag-of-words

Bag-of-words is simplifying unstructured representation of text as an unordered set of terms (text is considered as a 'bag' containing words). While disregarding grammar and word order, this model considers the words and their frequency of occurrence in text.
In practice, the Bag-of-words model is mainly used as a tool of feature generation. Machine learning algorithms can't work with text directly, and this model converts text into vectors of numbers representation which is suitable for algorithms we are going to use. These vectors are sparse (most elements are zero), especially for our problem that has huge vocabulary across messages in collection. It is beneficial and often necessary to use specialized algorithms and data structures that take advantage of the sparse structure of such vectors. We are going to use such vectors as features in our models.

4.6. Machine Learning Algorithms

4.6.1. Baseline

We are going to create simple but effective Naive Bayes baseline model that will be used for comparing with other, more advanced algorithms that use NLP techniques and are optimized by using grid search and cross-validation (model selection). For proof that some other classifier is better than baseline we will perform statistical significance test

Naive Bayes is a classifying algorithm which uses data about prior events to estimate the probability of future events based on applying Bayes' theorem. It has a strong (naive) assumption that every feature is independent of others. Probabilistic classifiers like this are one of the oldest solutions for spam filtering. They are typically used with bag-of-words model features to identify spam.

It is often used as baseline model due to its simplicity and often good performance. It is simple and fast to train, it has no hyperparameters that need to be tuned, it has probabilistic output, they have been demonstrated to be reliable and accurate in number of applications of NLP, so there is no confusion about popularity of this algorithm.

Naive Bayes model assumes that each of the features it uses are conditionally independent of one another given some class. More formally, if we want to calculate the probability of observing features *f1* through *fn*, given some class *c*, under the Naive Bayes assumption the following holds:

$$p(f1, ..., fn|c) \propto p(c)p(f1|c)...p(fn|c)$$

We have said nothing about the distribution of each feature. In other words, we have left $p(fi|c)$ undefined. Multinomial Naive Bayes is a specialized version of Naive Bayes that is designed more for text documents. It simply lets us know that each $p(fi|c)$ is a multinomial distribution, rather than some other distribution. This distribution works well for data which can easily be turned into counts, such as word counts in text, so we can apply it in our problem.

4.6.2. Candidates

The "No Free Lunch" theorem states that there is no one model that works best for every problem. Maybe there are better solutions than our baseline model which sometimes isn't powerful enough due to its independence assumption, and it has problems with high dimensionality of the feature

	bow	tok1	tok2
MultinomialNB	0.9714*	0.8945	0.9733
LogisticRegression	**0.9733**	0.9733	0.9820
SVC	0.9586	0.9664	0.9702
KNeighborsClassifier	0.8092	0.9681	0.9799
RandomForestClassifier	0.9608	**0.9873**	**0.9849**
XGBoost	0.9443	0.9864	0.9840

Table 2: Experiment results with $F_{0.5}$ scores

space. To test if there is some better machine learning algorithm than baseline, we are going to use some of the most common machine learning algorithms for classification:

- Logistic Regression
- Support Vector Machine
- k-NN
- Random Forest
- XGBoost

5. Experiment

For our experiment we evaluated three different approaches. One originally proposed system with Bag-of-Word only and two our systems with *tok1* and *tok2* tokenizers. When using *tok2* tokenizer stop words removal and stemming steps were omitted. For each model we tuned their hyperparameters with 10-fold cross validation method.

5.1. Evaluation

Due to specific nature of spam and ham messages, emphasis was put on classifying ham messages correctly. We concluded that it would be fine if we had some spam messages misclassified but much worse if ham message was labeled as spam. Spam messages present positive examples and ham messages are negative examples. Due to that fact we want precision measure as high as possible. As a result, we choose $F_{0.5}$ score as a performance measure to put emphasis on precision.

5.2. Results

First column in Table 2. represents results with Bag-of-Words approach which was implemented as CountVectorizer. We can see that tuned Logistic Regression produces slightly better performance than our baseline model.

Second column represents results of our proposed system with *tok1* tokenizer. We can see increase in performance in most of our models. Worth mentioning is sharp decline in baseline performance which shows us that generative models do not benefit from additional features.

Third column represents results of our proposed system with *tok2* tokenizer. In this approach we gain 20 times more features from text. Most models had increase in performance slightly, but our best performing models Random Forest Classifier had slight decrease. We concluded that tradeoff between performance increase and explosion in feature dimensionality goes in favor of *tok1* tokenizer.

5.3. Significance Testing

Lastly, we tested our baseline model against best performing model within our system which is Random Forest classifier. Significance testing was done with one sided paired t-test on 99% confidence level, and alternative hypothesis that our approach with Random Forest classifier performs better then baseline model. Performance was measured with nested cross validation with 10-fold both on inner and outer loop. Observed p-value was 0.0025 or 0.25%, which indeed proves that our system has statistically significant increase in performance over baseline model.

6. Conclusion and Future Work

In this paper, we propose a pipeline approach that combines NLP techniques such as stemming and stop words removal, with model selection of most popular machine learning algorithms for classification. We used SMS Spam Collection dataset which is the largest SMS corpus as we know. We conduct experiments on this dataset of comparing baseline model (Multinomial Naive Bayes) with best solution from our pipeline approach. We showed that our pipeline approach can yield statistically significant performance gain as compared to the baseline. In future, we would like to explore the deep learning models, especially LSTM.

References

T.A. Almeida, J.M.G. Hidalgo, and A. Yamakami. 2011. Contributions to the study of sms spam filtering: New collection and results.

H. W. Brown, A. S. Forbes, and S. D. Smith. 1900. Title title title title title title title title title title. *Journal journal journal*.

L. Duan, N. Li, and L. Huang. 2009. A new spam short message classification. 2:168–171, March.

Jose Gomez Hidalgo, Guillermo Cajigas Bringas, Enrique Sanz, and Francisco García. 2006. Content based sms spam filtering. 2006:107–114, 01.

Chowdhury Morshed Xiang, Yang and Shawkat Ali. 2004. Filtering mobile spam by support vector machine. *CSITeA'04: Third International Conference on Computer Sciences, Software Engineering, Information Technology, E-Business and Applications, International Society for Computers and Their Applications (ISCA)*, pages 1–4.