Contributions to the study of SMS Spam Filtering: New Collection and Results

Tiago A. Almeida
School of Electrical and Computer Engineering
University of Campinas
Campinas, Sao Paulo, Brazil
tiago@dt.fee.unicamp.br

José María Gómez Hidalgo
R&D Department
Optenet
Las Rozas, Madrid, Spain
jgomez@optenet.com

Akebo Yamakami
School of Electrical and Computer Engineering
University of Campinas
Campinas, Sao Paulo, Brazil
akebo@dt.fee.unicamp.br

ABSTRACT

The growth of mobile phone users has lead to a dramatic increasing of SMS spam messages. In practice, fighting mobile phone spam is difficult by several factors, including the lower rate of SMS that has allowed many users and service providers to ignore the issue, and the limited availability of mobile phone spam-filtering software. On the other hand, in academic settings, a major handicap is the scarcity of public SMS spam datasets, that are sorely needed for validation and comparison of different classifiers. Moreover, as SMS messages are fairly short, content-based spam filters may have their performance degraded. In this paper, we offer a new real, public and non-encoded SMS spam collection that is the largest one as far as we know. Moreover, we compare the performance achieved by several established machine learning methods. The results indicate that Support Vector Machine outperforms other evaluated classifiers and, hence, it can be used as a good baseline for further comparison.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and retrieval—*information filtering*; H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*performance evaluation (efficiency and effectiveness)*

General Terms

Performance, Experimentation, Security, Standardization

Keywords

Spam filtering, mobile spam, text categorization, classification

1. INTRODUCTION

Short Message Service (SMS) is the text communication service component of phone, web or mobile communication systems, using standardized communications protocols that allow the exchange of short text messages between fixed line or mobile phone devices. According to the International Telecommunication Union (ITU)[1], SMS has become a massive commercial industry, worth over 81 billion dollars globally as of 2006.

The downside is that cell phones are becoming the latest target of electronic junk mail, with a growing number of marketers using text messages to target subscribers. SMS spam (sometimes also called mobile phone spam) is any junk message delivered to a mobile phone as text messaging. Although this practice is rare in North America, it has been very common in some parts of Asia.

Apparently, SMS spam is not as cost-prohibitive to spammers as it used to be, as the popularity of SMS has led to messaging charges dropping below US$ 0.001 in markets like China, and even free of charge in others. According to Cloudmark stats[2], the amount of mobile phone spam varies widely from region to region. For instance, in North America, much less than 1% of SMS messages were spam in 2010, while in parts of Asia up to 30% of messages were represented by spam. The same report indicates that Chinese users get up to ten mobile spam messages per day. Some operators in India are looking at high spam level, even after protocol-level filtering. And in Japan, the current spam problem is expected to get worse as operators open their networks to email-to-SMS and MMS services.

The main problem with SMS spam is that it is not only annoying, but it can also be expensive since some people pay to receive text messages. Moreover, there is a limited availability of mobile phone spam-filtering software. Other concern is that important legitimate messages as of emergency nature could be blocked. Nonetheless, many providers offer their subscribers means for mitigating unsolicited SMS messages.

In the same way that carriers are facing many problems in dealing with SMS spam, academic researchers in this field are also experiencing difficulties. For instance, the lack of real and public databases can compromise the evaluation of different approaches. So, although there has been significant

[1] See `http://www.itu.int/dms_pub/itu-s/opb/pol/S-POL-IR.DL-2-2006-R1-SUM-PDF-E.pdf`.

[2] See `http://www.cloudmark.com/en/article/mobile-operators-brace-for-global-surge-in-...mobile-messaging-abuse`

effort to generate public benchmark datasets for anti-spam filtering, unlike email spam, which has available a large variety of datasets, the mobile spam filtering still has very few corpora usually of small size. Other concern is that established email spam filters may have their performance seriously degraded when directly employed to dealing with mobile spam, since the standard SMS messaging is limited to 140 bytes, which translates to 160 characters of the English alphabet. Moreover, their text is rife with idioms and abbreviations.

To fill this important gap, in this paper, we make available a new real, public and non-encoded SMS spam corpus that is the largest one as far as we know. Moreover, we compare the performance achieved by several established machine learning methods in order to provide good baseline results for further comparison.

The remainder of this paper is organized as follows. Section 2 offers details about the newly-created SMS Spam Collection. In Section 3, we present a comprehensive performance evaluation for comparing several established machine learning approaches. Finally, Section 4 presents the main conclusions and outlines for future works.

2. THE NEW SMS SPAM COLLECTION

Reliable data are essential in any scientific research. The absence of representative data can seriously impact the processes of evaluation and comparison of methods. In this way, areas of more recent studies are generally affected by the lack of public available data.

Regarding studies of mobile spam filtering, although there are few databases of legitimate SMS messages available in the Internet, it is very hard to find real samples of mobile phone spam. Thus, to create the corpus for the purposes of this work, we use data derived from several sources.

A collection of 425 SMS spam messages was manually extracted from the Grumbletext Web site. This is a UK forum in which cell phone users make public claims about SMS spam messages, most of them without reporting the very spam message received. The identification of the text of spam messages in the claims is a very hard and time-consuming task, and it involved carefully scanning hundreds of web pages. The Grumbletext Web site is: `http://www.grumbletext.co.uk/`.

We have also included in our corpus a subset of 3,375 SMS randomly chosen ham messages of the NUS SMS Corpus, which is a dataset of about 10,000 legitimate messages collected for research at the Department of Computer Science at the National University of Singapore. The messages largely originate from Singaporeans and mostly from students attending the University. These messages were collected from volunteers who were made aware that their contributions were going to be made publicly available. The NUS SMS Corpus is available at: `http://www.comp.nus.edu.sg/...~rpnlpir/downloads/corpora/smsCorpus/`.

We have added legitimate samples by inserting 450 SMS ham messages collected from Caroline Tag's PhD Thesis available at `http://etheses.bham.ac.uk/253/1/Tagg09PhD.pdf`.

Finally, we have incorporated the SMS Spam Corpus v.0.1 Big. It has 1,002 SMS ham messages and 322 spam messages and it is public available at: `http://www.esp.uem.es/jmgomez/smsspamcorpus/`. This corpus has been used in the following academic research efforts: [6], [7], and [14]. The sources used in this corpus are also the Grumbletext Web site and the NUS SMS Corpus.

The created corpus is composed by just one text file, where each line has the correct class followed by the raw message. We offer some examples bellow:

```
ham    What you doing?how are you?
ham    Ok lar...  Joking wif u oni...
ham    dun say so early hor...  U c already then
       say...
ham    MY NO. IN LUTON 0125698789 RING ME IF UR
       AROUND! H*
ham    Siva is in hostel aha:-.
ham    Cos i was out shopping wif darren jus now
       n i called him 2 ask wat present he wan
       lor.  Then he started guessing who i was
       wif n he finally guessed darren lor.
spam   FreeMsg:  Txt:  CALL to No:  86888 & claim
       your reward of 3 hours talk time to use
       from your phone now!  ubscribe6GBP/ mnth
       inc 3hrs 16 stop?txtStop
spam   URGENT! Your Mobile No 07808726822 was
       awarded a £2,000 Bonus Caller Prize on
       02/09/03!  This is our 2nd attempt to
       contact YOU! Call 0871-872-9758 BOX95QU
```

The SMS Spam Collection is public available at `http://www.dt.fee.unicamp.br/~tiago/smsspamcollection`.

In the following we present some statistics of the dataset. In summary, the new collection is composed by 4,827 legitimate messages and 747 mobile spam messages, a total of 5,574 short messages. To the best of our knowledge, it is the largest available SMS spam corpus that currently exists. Table 1 shows the basic statistics of the created database.

Table 1: Basic statistics

Msg	Amount	%
Hams	4,827	86.60
Spams	747	13.40
Total	5,574	100.00

Table 2 presents the statistics related to the tokens extracted from the corpus. Note that, the proposed dataset has a total of 81,175 tokens and mobile phone spam has in average ten tokens more than legitimate messages.

Table 2: Token statistics

Hams	63,632
Spams	17,543
Total	81,175
Avg per Msg	14.56
Avg in Hams	13.18
Avg in Spams	23.48

We have also performed a study regarding the occurrence frequency of tokens in each class. Tables 3 and 4 show the twenty tokens that most have appeared in ham and spam messages, respectively.

Table 3: The Twenty tokens that most appeared in ham messages

Token	Number of Hams Msg	% of Hams
i	1619	33.54
you	1264	26.19
to	1219	25.25
a	880	18.23
the	867	17.96
in	737	15.27
and	685	14.19
u	678	14.05
me	639	13.24
is	603	12.49
my	600	12.43
it	464	9.61
of	454	9.41
for	443	9.18
that	421	8.72
im	414	8.58
but	411	8.51
so	403	8.35
have	401	8.31
not	384	7.96

Table 4: The twenty tokens that most appeared in spam messages

Token	Number of spam Msg	% of Spams
to	467	62.52
call	329	44.04
a	294	39.36
your	227	30.39
you	218	29.18
for	177	23.69
or	177	23.69
the	167	22.36
free	157	21.02
txt	145	19.41
2	142	19.01
is	140	18.74
have	127	17.00
from	124	16.60
on	119	15.93
u	118	15.80
ur	114	15.26
now	112	14.99
and	108	14.46
claim	108	14.46

To complement the study regarding token frequency among each class, we also evaluated the degree of importance of each token over the full corpus. For this, we sorted all the tokens according to the information gain score (IG) [18] and present the first twenty ones in Table 5.

Table 5: The twenty tokens with highest IG score over the full corpus

Rank	IG	Token	Rank	IG	Token
1	0.099	call	11	0.036	won
2	0.066	txt	12	0.033	or
3	0.057	claim	13	0.033	now
4	0.057	free	14	0.033	&
5	0.057	to	15	0.032	stop
6	0.043	mobile	16	0.029	reply
7	0.043	www	17	0.028	win
8	0.041	i	18	0.028	text
9	0.037	prize	19	0.026	cash
10	0.036	your	20	0.025	co

2.1 Message Duplicates Analysis

As the newly collected messages in the SMS Spam Collection have been augmented with a previously existing database built using roughly the same sources (GrumbleText, NUS SMS Corpus), it is sensible to check if there are some duplicates coming from both databases.

To address this issue, we have performed a duplicates analysis based on plagiarism detection techniques [9]. In general, plagiarism detection intends to discover cheating practices in educational environments, but it is increasingly important for enterprises with a Web site [17]. In [9], a number of techniques for plagiarism are presented, and those based on "String-of-Text", and implemented by the tool WCopyfind[3], can be considered as reasonable baseline for the purpose of detecting near-duplicated messages in our collection.

The "String-of-Text" methods involve scanning suspect texts for approximately matching character sequences. The approximation can involve transformations like case changing, separators variation (*e.g.* including more white spaces between words), etc. In other words, texts are compared searching for N-grams for relatively big sizes (*e.g.* N=6), with additional parameters (length of match in number of characters, etc.). This approach is implemented in WCopyfind, but we have simplified it to N-gram matches after text normalization involving replacing all token separators by white spaces, lowercasing all characters, and replacing digits by the character 'N' (to preserve phone numbers structure).

We searched for near-duplicates within three sub-collections: the previously existing SMS Spam Corpus v.0.1 Big (**INIT**), the additional messages from Grumbletext, the NUS SMS Corpus, and the Tag's PhD Thesis (**ADD**), and the actual new SMS Spam Collection (**FINAL**).

The goal of this process is to check if merging the first two sub-collections adds many near-duplicates to the final collection, in order to assess the overlap between both collections. Within each sub-collection, we have compared each pair of messages, stored all N size matches (N-grams with N = 5, 6, and 10), and sorted the N-grams according to their frequency, reporting the top ten ones per N.

The results of this analysis are reported in the Table 6.

[3] See: `http://plagiarism.phys.virginia.edu`

Table 6: Top N-gram matches for each sub-collection

N = 5

INIT		**ADD**		**FINAL**	
Match	*Freq.*	*Match*	*Freq.*	*Match*	*Freq.*
we are trying to contact	10	sorry i ll call later	37	sorry i ll call later	37
this is the Nnd attempt	9	private your NNNN account statement	15	private your NNNN account statement	16
urgent we are trying to	9	i cant pick the phone	12	we are trying to contact	14
prize guaranteed call NNNNNNNNNNN from	8	hope you are having a	9	prize guaranteed call NNNNNNNNNNN from	13
bonus caller prize on NN	7	text me when you re	9	you have won a guaranteed	13
draw txt music to NNNNN	7	ÃČÂć NNNN cash or a	8	a NNNN prize guaranteed call	12
prize N claim is easy	7	NNN anytime any network mins	8	draw shows that you have	12
you have won a guaranteed	7	a ÃČÂć NNNN prize guaranteed	7	i cant pick the phone	12
a N NNN bonus caller	6	have a secret admirer who	7	urgent we are trying to	11
are selected to receive a	6	u have a secret admirer	7	call NNNNNNNNNNN from land line	10

N = 6

INIT		**ADD**		**FINAL**	
Match	*Freq.*	*Match*	*Freq.*	*Match*	*Freq.*
this is the Nnd attempt to	9	private your NNNN account statement for	15	private your NNNN account statement for	16
urgent we are trying to contact	9	i cant pick the phone right	12	a NNNN prize guaranteed call NNNNNNNNNNN	12
prize guaranteed call NNNNNNNNNNN from land	7	a Ãć NNNN prize guaranteed call	7	draw shows that you have won	12
a N NNN bonus caller prize	6	have a secret admirer who is	7	i cant pick the phone right	12
bonus caller prize on NN NN	6	i am on the way to	6	prize guaranteed call NNNNNNNNNNN from land	12
cash await collection sae t cs	6	pls convey my birthday wishes to	6	urgent we are trying to contact	11
tone N ur mob every week	6	u have a secret admirer who	6	call our customer service representative on	10
you have won a guaranteed NNNN	6	Ãć NNN cash every wk txt	5	this is the Nnd attempt to	9
a NNNN prize guaranteed call NNNNNNNNNNN	5	as i entered my cabin my	5	tone N ur mob every week	9
call NNNNNNNNNNN now only NNp per	5	goodmorning today i am late for	5	we are trying to contact u	9

Due to size limitation we report only results for N = 5 and N = 6.

As it can be seen regarding 5-grams:

- 5-grams already present in the **INIT** and the **ADD** collections do not collapse to greatly increase their frequency. For instance, the 5-grams "sorry i ll call later" and "i cant pick the phone" do not change its frequency from **ADD** to **FINAL**. These 5-grams correspond to templates often present in cell phones, and used in legitimate messages. Actually, both are complete messages themselves.
- The behavior of the rest of 5-grams, which all actually nearly only occur in spam messages, is a bit different. Most of them are fuzzy duplicates that result in small frequency increases, like in "we are trying to contact" from **INIT** (10 messages) to **FINAL** (14 messages). This means that the messages in **ADD** may be duplicates of the messages in **INIT**. However, as it can be seen, the patterns of spam 5-grams within each subcollection are very regular and even overlapping, so this is not significant. In other words, these 4 messages are nor repeated, but new instances of spam probably sent by the same organization. Other messages just disappear from the top, as they keep their frequency.

With respect to 6-grams, that is the standard value used in tools like WCopyfind, we can see that the behavior is quite similar to the case of 5-grams. There are slightly different results because of two reasons:

- The fact that longer N-grams must obviously lead to lower frequencies. Actually, there is not a significant drop in the number of matches per 6-gram, as it can be seen in *e.g.* "private your NNNN account statement for", which includes the 5-gram "private your NNNN account statement" as a prefix.
- The most frequent 6-grams keep on belonging to spam messages. The 5-grams that frequently occurred on the legitimate messages have disappeared because the detected templates are, in fact, complete 5-length messages.

In 6-gram results, we can see again that there are not significant near-duplicates except for those already present in each sub-collection. Moreover, the results achieved with 10-grams are very similar to the previously presented ones.

In consequence, we believe it is safe to say that merging the sub-collections, although they have roughly the same sources, does not lead to near-duplicates that may ease the task of detecting SMS spam.

3. EXPERIMENTS

We tested several well-known machine learning methods in the task of automatic spam filtering using the created SMS Spam Collection. The main goal of this performance evaluation is to provide good baseline results for further comparison, since established email spam filters may have their performance seriously impacted when employed to classify short messages. In addition, mobile phone messages often have a lot of abbreviations and idioms that may also affect the filters accuracy.

3.1 Tokenizers

Tokenization is the first stage in the classification pipeline. It involves breaking the text stream into tokens ("words"), usually by means of a regular expression. We consider in this work two different tokenizers:

1. tok1: tokens start with a printable character, followed by any number of alphanumeric characters, excluding dots, commas and colons from the middle of the pattern. With this pattern, domain names and mail addresses will be split at dots, so the classifier can recognize a domain even if subdomains vary [16].
2. tok2: any sequence of characters separated by blanks, tabs, returns, dots, commas, colons and dashes are considered as tokens. This simple tokenizer intends to preserve other symbols that may help to separate spam and legitimate messages.

In addition, we did not perform language-specific preprocessing techniques such as stop word removal or word stemming, since other researchers found that such techniques tend to hurt spam-filtering accuracy [5, 19].

3.2 Classifiers

In the following sections, we briefly present each classifier evaluated in our experiments[4].

3.2.1 Naïve Bayes filters

The Naïve Bayes (NB) classifier is still the most employed in spam filtering because of its simplicity and high performance [2]. From Bayes' theorem and the theorem of the total probability, the probability for a message with vector $\vec{x} = \langle x_1, \ldots, x_n \rangle$ belongs to a category $c_i \in \{c_s, c_l\}$ is:

$$P(c_i|\vec{x}) = \frac{P(c_i).P(\vec{x}|c_i)}{P(\vec{x})}.$$

Since the denominator does not depend on the category, NB classifies each message in the category that maximizes $P(c_i).P(\vec{x}|c_i)$. In spam filtering domain it is equivalent to classify a message as spam (c_s) whenever

$$\frac{P(c_s).P(\vec{x}|c_s)}{P(c_s).P(\vec{x}|c_s) + P(c_l).P(\vec{x}|c_l)} > T,$$

with $T = 0.5$. By varying T, we can opt for more true negatives at the expense of fewer true positives, or vice-versa. Once the probabilities $P(\vec{x}|c_i)$ are estimated differently in each NB model, we tested eight different versions of NB classifiers.

Basic NB

Let $\mathcal{T} = \{t_1, \ldots, t_n\}$ the set of tokens collected in the training stage and $\mathcal{M}$ the set of messages in the training set, each message m is represented by a binary vector $\vec{x} = \langle x_1, \ldots, x_n \rangle$, where each x_k shows whether or not t_k will occur in m. The probabilities $P(\vec{x}|c_i)$ are calculated by [2]:

$$P(\vec{x}|c_i) = \prod_{k=1}^{n} P(t_k|c_i),$$

and the criterion for classifying a message as spam is:

$$\frac{P(c_s).\prod_{k=1}^{n} P(t_k|c_s)}{\sum_{c_i \in \{c_s,c_l\}} P(c_i).\prod_{k=1}^{n} P(t_k|c_i)} > T.$$

Here, probabilities $P(t_k|c_i)$ are estimated by:

$$P(t_k|c_i) = \frac{|\mathcal{M}_{t_k,c_i}|}{|\mathcal{M}_{c_i}|},$$

where $|\mathcal{M}_{t_k,c_i}|$ is the number of training messages of category c_i that contain the token t_k, and $|\mathcal{M}_{c_i}|$ is the total number of training messages of category c_i.

Multinomial term frequency NB

The multinomial term frequency NB (MN TF NB) represents each message as a set of tokens $m = \{t_1, \ldots, t_n\}$, computing each one of t_k as how many times it appears in m. In this sense, m can be represented by a vector

[4] Some of the implementations of the described classifiers are provided by the Machine Learning library WEKA, available at `http://www.cs.waikato.ac.nz/ml/weka/`. The algorithms have been used with their default parameters except when otherwise is specified.

$\vec{x} = \langle x_1, \dots, x_n \rangle$, where each x_k corresponds to the number of occurrences of t_k in m. $P(\vec{x}|c_i)$ is the multinomial distribution [2]:

$$P(\vec{x}|c_i) = P(|m|).|m|!.\prod_{k=1}^{n} \frac{P(t_k|c_i)^{x_k}}{x_k!}.$$

Thus, the criterion for classifying a message as spam becomes:

$$\frac{P(c_s).\prod_{k=1}^{n} P(t_k|c_s)^{x_k}}{\sum_{c_i \in \{c_s,c_l\}} P(c_i).\prod_{k=1}^{n} P(t_k|c_i)^{x_k}} > T,$$

and the probabilities $P(t_k|c_i)$ are estimated as a Laplacian prior:

$$P(t_k|c_i) = \frac{1 + N_{t_k,c_i}}{n + N_{c_i}},$$

where N_{t_k,c_i} is the number of occurrences of token t_k in the training messages of category c_i, and $N_{c_i} = \sum_{k=i}^{n} N_{t_k,c_i}$.

Multinomial Boolean NB

The multinomial Boolean NB (MN Bool NB) is similar to the MN TF NB, including the estimates of $P(t_k|c_i)$, except that each attribute x_k is Boolean. Note that these approaches do not take into account the absence of tokens ($x_k = 0$) from the messages.

Multivariate Bernoulli NB

The multivariate Bernoulli NB (Bern NB) represents each message m by computing the presence or absence of each token. Hence, m can be represented as a binary vector $\vec{x} = \langle x_1, \dots, x_n \rangle$, where each x_k shows whether or not t_k will occur in m. The probabilities $P(\vec{x}|c_i)$ are computed by [2]:

$$P(\vec{x}|c_i) = \prod_{k=1}^{n} P(t_k|c_i)^{x_k}.(1 - P(t_k|c_i))^{(1-x_k)}.$$

The criterion for classifying a message as spam becomes:

$$\frac{P(c_s).\prod_{k=1}^{n} P(t_k|c_s)^{x_k}.(1 - P(t_k|c_s))^{(1-x_k)}}{\sum_{c_i \in \{c_s,c_l\}} P(c_i).\prod_{k=1}^{n} P(t_k|c_i)^{x_k}.(1 - P(t_k|c_i))^{(1-x_k)}} > T,$$

and probabilities $P(t_k|c_i)$ are estimated as a Laplacian prior:

$$P(t_k|c_i) = \frac{1 + |\mathcal{M}_{t_k,c_i}|}{2 + |\mathcal{M}_{c_i}|},$$

where $|\mathcal{M}_{t_k,c_i}|$ is the number of training messages of category c_i that comprise the token t_k, and $|\mathcal{M}_{c_i}|$ is the total number of training messages of category c_i.

Boolean NB

Boolean NB (Bool NB) is similar to Bern NB with the difference that this does not take into account the absence of tokens. The probabilities $P(\vec{x}|c_i)$ are estimated only by:

$$P(\vec{x}|c_i) = \prod_{k=1}^{n} P(t_k|c_i),$$

and the criterion for classifying a message as spam becomes:

$$\frac{P(c_s).\prod_{k=1}^{n} P(t_k|c_s)}{\sum_{c_i \in \{c_s,c_l\}} P(c_i).\prod_{k=1}^{n} P(t_k|c_i)} > T,$$

where probabilities $P(t_k|c_i)$ are estimated in the same way as used in the Bern NB.

Multivariate Gauss NB

Multivariate Gauss NB (Gauss NB) uses real-valued attributes by assuming that each attribute follows a Gaussian distribution $g(x_k; \mu_{k,c_i}, \sigma_{k,c_i})$ for each category c_i, where the μ_{k,c_i} and σ_{k,c_i} of each distribution are estimated from the training set. The probabilities $P(\vec{x}|c_i)$ are calculated by [2]:

$$P(\vec{x}|c_i) = \prod_{k=1}^{n} g(x_k; \mu_{k,c_i}, \sigma_{k,c_i}),$$

and the criterion for classifying a message as spam becomes:

$$\frac{P(c_s).\prod_{k=1}^{n} g(x_k; \mu_{k,c_s}, \sigma_{k,c_s})}{\sum_{c_i \in \{c_s,c_l\}} P(c_i).\prod_{k=1}^{n} g(x_k; \mu_{k,c_i}, \sigma_{k,c_i})} > T.$$

Flexible Bayes

Flexible Bayes (Flex NB) works similar to Multivariate Gauss NB. However, instead of using a single normal distribution for each attribute X_k per category c_i, FB represents the probabilities $P(\vec{x}|c_i)$ as the average of L_{k,c_i} normal distributions with different values for μ_{k,c_i}, but the same one for σ_{k,c_i} [2]:

$$P(x_k|c_i) = \frac{1}{L_{k,c_i}} \sum_{l=1}^{L_{k,c_i}} g(x_k; \mu_{k,c_i,l}, \sigma_{c_i}),$$

where L_{k,c_i} is the amount of different values that the attribute X_k has in the training set Tr of category c_i. Each of these values is used as $\mu_{k,c_i,l}$ of a normal distribution of the category c_i. However, all distributions of a category c_i are taken to have the same $\sigma_{c_i} = \frac{1}{\sqrt{|Tr_{c_i}|}}$.

Boosted NB

Boosting is a meta-classifier based on the idea that relatively low accuracy classifiers can be improved by sequentially training them on the examples (messages) they wrongly classify, and combining the sequence of generated classifiers. Starting with a Multivariate Gauss NB, we have applied the AdaBoost M1 boosting algorithm [12] with ten iterations.

3.2.2 Support Vector Machines

Support vector machine (SVM) [10, 13] is one of the most recent techniques used in text classification. In this method a data point is viewed as a p-dimensional vector and the approach aims to separate such points with a $(p-1)$-dimensional hyperplane. This is called a linear classifier. There are many hyperplanes that might classify the data. One reasonable choice as the best hyperplane is the one that represents the largest separation, or margin, between the two classes. Therefore, SVM chooses the hyperplane so that the distance from it to the nearest data point on each side is maximized. If such a hyperplane exists, it is known as the maximum-margin hyperplane and the linear classifier it defines is known as a maximum margin classifier.

3.2.3 Minimum Description Length

Minimum Description Length (MDL) classifier considers each class c as a sequence of tokens extracted from the messages and inserted into the training set. Each token t from m has a code length L_t based on the sequence of tokens presented in the messages of the training set of c. The length of m when assigned to the class c corresponds to the sum of all code lengths associated with each token of m, $Lm = \sum_{i=1}^{|m|} L_{t_i}$. We calculate $L_{t_i} = \lceil -log_2 P_{t_i} \rceil$, where P is a probability distribution related with the tokens of class. Let $n_c(t_i)$ the amount of messages of class c that has t_i, then the probability that any token belongs to c is given by the maximum likelihood estimation:

$$P_{t_i} = \frac{n_c(t_i) + \frac{1}{|\chi|}}{n_c + 1}$$

where n_c corresponds to the sum of $n_c(t_i)$ for all tokens that appear in messages that belongs to c and $|\Omega|$ is the vocabulary size. We assume that $|\Omega| = 2^{32}$, that is, each token in an uncompress mode is a symbol with 32 bits. This estimation reserves a "portion" of probability to words which the classifier has never seen before [4].

Briefly, the MDL spam filter classify a message by computing the increase of the description length when m is assigned to each class $c \in \{spam, legitimate\}$:

$$L_m(spam) = \sum_{i=1}^{|m|} \left\lceil -log_2 \left(\frac{n_{spam}(t_i) + \frac{1}{|\Omega|}}{n_{spam} + 1} \right) \right\rceil$$

$$L_m(ham) = \sum_{i=1}^{|m|} \left\lceil -log_2 \left(\frac{n_{ham}(t_i) + \frac{1}{|\Omega|}}{n_{ham} + 1} \right) \right\rceil$$

If $L_m(spam) < L_m(ham)$, then m is classified as spam; otherwise, m is labeled as legitimate.

3.2.4 K-Nearest Neighbors

K-Nearest Neighbors (KNN) [1] is a lazy method that actually does not build a model on the training data, but classifies new instances according to the classes of the samples in the training collection that are closest to the target instance in the attribute vector space. We have used 1, 3 and 5 closest messages (neighbors) according to the Euclidean Distance in our experiments, selecting the majority class among the neighbors. KNN was previously used in SMS spam detection [14].

3.2.5 C4.5

C4.5 [15] is a decision tree learner in which the nodes are labeled with tests on attributes, and the branches with the possible values of the attribute they are connected. The leaves of the trees are labeled with the classes, and the algorithm for building a tree works in a top-down fashion, in which attributes are selected according to their Gain Ratio score on each step. On this basic strategy, some optimizations like pruning the resulting tree are performed, in order to avoid overfitting. C4.5 has also been previously tested in mobile phone spam classification [14].

3.2.6 PART

PART [11] is a rule learner that is based on an algorithm for inferring rules by repeatedly generating partial decision trees with C4.5. A major property of this algorithm is that rule sets are learned one rule at a time, without any need for global optimization. Despite of avoiding this optimization present in other rule learners like RIPPER, or decision tree learners like C4.5, it achieves very good performance in terms of accuracy and efficiency. This method was evaluated in SMS spam filtering task by [14].

3.3 Baselines

Given that the collection is clearly biased to the legitimate class, an obvious baseline is the trivial rejector (TR) for the spam class.

As the most of the tokens with the highest Information Gain score often occur in the spam class, it is sensible to expect that messages may get automatically segregated into two classes on the basis of those tokens. In consequence, we provide an additional baseline in the form of the results of the Expectation-Maximization (EM) clustering algorithm [8], over a vector representation based on the tokenizer tok2. EM is an iterative soft clusterer that estimates cluster densities. Basically, cluster membership is a hidden latent variable that the maximum likelihood EM method estimates.

EM clustering works in the following way. To start with, a random assignment of instances to clusters is used. Initial distributions for each cluster are learned from this starting point, and then the E and M step of the algorithm are executed in subsequent iterations. The E step estimates the cluster membership of each instance given the current model – this is a soft, probabilistic membership where the predicted density/probability distribution is used to weight each instance. Then the M step re-estimates the parameters of the normal and discrete distributions for each cluster using the weights computed by the E step. Iteration stops when the likelihood of the training data with respect to the model does not increase enough from one iteration to the next, or the maximum number of iterations have been performed.

In our experiments, we have set up a maximum of 20 iterations and used the rest of the default values for EM parameters in WEKA.

3.4 Protocol

We carried out this study using the following experiment protocol. We divided the corpus in two parts: the first 30% of the messages were separated for training (1,674 messages) and the remainder ones for testing (3,900 messages). As all the messages are fairly short, we did not use any kind of method to reduce the dimensionality of the training space, *e.g.*, terms selection techniques.

To compare the results achieved by the filters we employed the following well-known performance measures:

- Spam Caught (%) – SC;
- Blocked Hams (%) – BH;
- Accuracy (%) – Acc;

- Matthews Correlation Coefficient – MCC [3].

MCC is used in machine learning as a measure of the quality of binary classifications. It returns a real value between -1 and $+1$. A coefficient equals to $+1$ indicates a perfect prediction; 0, an average random prediction; and -1, an inverse prediction [4].

$$MCC = \frac{(|\mathcal{TP}|.|\mathcal{TN}|) - (|\mathcal{FP}|.|\mathcal{FN}|)}{\sqrt{(|\mathcal{TP}| + |\mathcal{FP}|).(|\mathcal{TP}| + |\mathcal{FN}|).(|\mathcal{TN}| + |\mathcal{FP}|).(|\mathcal{TN}| + |\mathcal{FN}|)}},$$

where $|\mathcal{TP}|$ corresponds to the amount of true positives, $|\mathcal{TN}|$ is the number of true negatives, $|\mathcal{FP}|$ is the amount of false positives, and $|\mathcal{FN}|$ is the number of false negatives.

3.5 Results

Table 7 presents the best fifteen results achieved by each evaluated classifier and tokenizer. Note that the results are sorted in descending order of MCC.

Table 7: The fifteen best results achieved by combinations of classifiers + tokenizers and the baselines Expectation-Maximization (EM) and trivial rejection (TR)

Classifier	$SC\%$	$BH\%$	$Acc\%$	MCC
SVM + tok1	83.10	0.18	97.64	0.893
Boosted NB + tok2	84.48	0.53	97.50	0.887
Boosted C4.5 + tok2	82.91	0.29	97.50	0.887
PART + tok2	82.91	0.29	97.50	0.887
MDL + tok1	75.44	0.35	96.26	0.826
C4.5 + tok2	75.25	2.03	95.00	0.770
Bern NB + tok1	54.03	0.00	94.00	0.711
MN TF NB + tok1	52.06	0.00	93.74	0.697
MN Bool NB + tok1	51.87	0.00	93.72	0.695
1NN + tok2	43.81	0.00	92.70	0.636
Basic NB + tok1	48.53	1.42	92.05	0.600
Gauss NB + tok1	47.54	1.39	91.95	0.594
Flex NB + tok1	47.35	2.77	90.72	0.536
Boolean NB + tok1	98.04	26.01	77.13	0.507
3NN + tok2	23.77	0.00	90.10	0.462
EM + tok2	17.09	4.18	85.54	0.185
TR	0.00	0.00	86.95	–

It is notable that the linear SVM achieved the best results and outperformed the other evaluated methods. It caught 83.10% of all spams with the cost of blocking only 0.18% of legitimate messages, acquiring an accuracy rate higher than 97.5%. It is a remarkable performance considering the EM and TR baselines and the high difficulty of classifying mobile phone messages. However, the results also indicate that the best four algorithms achieved similar performance with no statistical difference. All of them accomplished an accuracy rate superior than 97%, that can be considered as a very good baseline in a such context.

It is important to point out that MDL and C4.5 techniques also achieved good results since they found a good balance between false and true positive rates. On the other hand, the remainder evaluated approaches had an unsatisfying performance. Note that, although the most of them have obtained accuracy rate superior than 90%, they have correctly filtered about only 50% of spams or even less.

Therefore, based on the achieved results, we can certainly conclude that the linear SVM offers the best baseline performance for further comparison.

4. CONCLUSIONS

The task of automatic filtering SMS spam still is a real challenge nowadays. There are three main problems hindering the development of algorithms in this specific field of research: the lack of public and real datasets, the low number of features that can be extracted per message, and the fact that the text is rife with idioms and abbreviations.

To fill some of those gaps, in this paper we presented a new mobile phone spam collection that is the largest one as far as we know. Besides being large, it is also publicly available and composed by only non-encoded and real messages.

Moreover, we offered statistics relating to the proposed corpus, as tokens frequencies and the most relevant words in terms of information gain scores. Since the corpus is composed by subsets of messages extracted from the same sources, we also presented a study regarding the message duplicates and the found results indicate that the proposed collection is reliable because there are no more duplicates than those ones already presented within the used subsets.

Finally, we compared the performance achieved by several established machine learning methods and the results indicate that Support Vector Machine outperforms other evaluated classifiers and, hence, it can be used as a good baseline for further comparison.

Future work should consider to use different strategies to increase the dimensionality of the feature space. Well-known techniques, such as orthogonal sparse bigrams (OSB), 2-grams, 3-grams, among many others could be employed with the standard tokenizers to produce a larger number of tokens and patterns which can assist the classifier to separate ham messages from spams.

5. ACKNOWLEDGMENTS

The authors would like to thank the financial support of Brazilian agencies FAPESP and CAPES/PRODOC.

6. REFERENCES

[1] D. Aha and D. Kibler. Instance-based learning algorithms. *Machine Learning*, 6:37–66, 1991.

[2] T. A. Almeida, J. Almeida, and A. Yamakami. Spam Filtering: How the Dimensionality Reduction Affects the Accuracy of Naive Bayes Classifiers . *Journal of Internet Services and Applications*, 1(3):183–200, 2011.

[3] T. A. Almeida, A. Yamakami, and J. Almeida. Evaluation of Approaches for Dimensionality Reduction Applied with Naive Bayes Anti-Spam Filters. In *Proceedings of the 8th IEEE International Conference on Machine Learning and Applications*, pages 517–522, Miami, FL, USA, 2009.

[4] T. A. Almeida, A. Yamakami, and J. Almeida. Filtering Spams using the Minimum Description Length Principle. In *Proceedings of the 25th ACM Symposium On Applied Computing*, pages 1856–1860, Sierre, Switzerland, 2010.

[5] G. Cormack. Email Spam Filtering: A Systematic Review. *Foundations and Trends in Information Retrieval*, 1(4):335–455, 2008.

[6] G. V. Cormack, J. M. Gómez Hidalgo, and E. Puertas Sanz. 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, New York, NY, USA, 2007.

[7] G. V. Cormack, J. M. Gómez Hidalgo, and E. Puertas Sanz. Spam Filtering for Short Messages. In *Proceedings of the 16th ACM Conference on Conference on information and Knowledge Management*, pages 313–320, Lisbon, Portugal, 2007.

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[9] H. Dreher. Automatic Conceptual Analysis for Plagiarism Detection. *Issues in Informing Science and Information Technology*, 4:601–628, 2007.

[10] G. Forman, M. Scholz, and S. Rajaram. Feature Shaping for Linear SVM Classifiers. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 299–308, Paris, France, 2009.

[11] E. Frank and I. H. Witten. Generating Accurate Rule Sets Without Global Optimization. In *Proceedings of the 15th International Conference on Machine Learning*, pages 144–151, Madison, WI, USA, 1998.

[12] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Thirteenth International Conference on Machine Learning*, pages 148–156, San Francisco, 1996. Morgan Kaufmann.

[13] J. M. Gómez Hidalgo. Evaluating Cost-Sensitive Unsolicited Bulk Email Categorization. In *Proceedings of the 17th ACM Symposium on Applied Computing*, pages 615–620, Madrid, Spain, 2002.

[14] J. M. Gómez Hidalgo, G. Cajigas Bringas, E. Puertas Sanz, and F. Carrero García. Content Based SMS Spam Filtering. In *Proceedings of the 2006 ACM Symposium on Document Engineering*, pages 107–114, Amsterdam, The Netherlands, 2006.

[15] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., 1993.

[16] C. Siefkes, F. Assis, S. Chhabra, and W. Yerazunis. Combining Winnow and Orthogonal Sparse Bigrams for Incremental Spam Filtering. In *Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 410–421, Pisa, Italy, 2004.

[17] E. Vallés Balaguer. Putting Ourselves in SME's Shoes: Automatic Detection of Plagiarism by the WCopyFind Tool. In *Proceedings of the SEPLN'09 Workshop on Uncovering Plagiarism, Authorship and Social Software Misuse*, pages 29–33, San Sebastian, Spain, 2009.

[18] Y. Yang and J. Pedersen. A Comparative Study on Feature Selection in Text Categorization. In *Proceedings of the 14th International Conference on Machine Learning*, pages 412–420, Nashville, TN, USA, 1997.

[19] L. Zhang, J. Zhu, and T. Yao. An Evaluation of Statistical Spam Filtering Techniques. *ACM Transactions on Asian Language Information Processing*, 3(4):243–269, 2004.