J Internet Serv Appl (2012) 3:245–253
DOI 10.1007/s13174-012-0067-x

ORIGINAL PAPER

Occam's razor-based spam filter

Tiago A. Almeida · Akebo Yamakami

Received: 22 July 2011 / Accepted: 10 August 2012 / Published online: 2 October 2012

Abstract Nowadays e-mail spam is not a novelty, but it is still an important rising problem with a big economic impact in society. Spammers manage to circumvent current spam filters and harm the communication system by consuming sev eral resources, damaging the reliability of e-mail as a communication instrument and tricking recipients to react to spam messages. Consequently, spam filtering poses a special problem in text categorization, of which the defining characteristic is that filters face an active adversary, which constantly attempts to evade filtering. In this paper, we present a novel approach to spam filtering based on the minimum description length principle. Furthermore, we have conducted an empirical experiment on six public and real non-encoded datasets. The results indicate that the proposed filter is fast to construct, incrementally updateable and clearly outperforms the state-of-the-art spam filters.

Keywords Minimum description length · Spam filter · Text categorization · Knowledge-based system · Machine learning

1 Introduction

E-mail is one of the most popular, fastest and cheapest means of communication. It has become a part of everyday life for millions of people, changing the way we work and collabo rate. E-mail is not only used to support conversation but also as a task manager, document delivery system and archive. The downside of this success is the constantly growing volume of e-mail spam we receive. The problem of spams can be quantified in economical terms since many hours are wasted everyday by workers. It is not just the time they waste reading the spam but also the time they spend deleting those messages [33].

According to annual reports, the amount of spam is frightfully increasing. In absolute numbers, the average of spams sent per day increased from 2.4 billion in 2002[1] to 300 billion in 2010.[2] The same report indicates that more than 90 % of incoming e-mail traffic is spam. According to the US Technology Readiness Survey,[3] the cost of spam in terms of lost productivity in US has reached US$ 21.58 billion annually, while the worldwide productivity cost of spam is estimated to be US$ 50 billion. On a worldwide basis, the information technology cost of dealing with spam was estimated to rise from US$ 20.5 billion in 2003 to US$ 198 billion in 2009.

According to a report published by McAfee,[4] the cost in lost productivity per day per user is approximately equal to US$ 0.50, based on the user's having to spend 30 s for dealing with only two spam messages each day and the user's spam filter working at 95 percent accuracy (value higher than the average achieved by the majority of available anti-spam filters). Therefore, the productivity loss per employee per year due to spam is approximately equal to US$ 182.50. Supposing a company with 1,000 workers earning US$ 30

T. A. Almeida (✉)
Department of Computer Science, Federal University of São Carlos
UFSCar, 18052-780 Sorocaba, SP, Brazil
e-mail: talmeida@ufscar.br

A. Yamakami
School of Electrical and Computer Engineering, University
of Campinas, UNICAMP, 13083-970 Campinas, SP, Brazil
e-mail: akebo@dt.fee.unicamp.br

[1] See http://www.spamlaws.com/spam-stats.html.

[2] See http://www.ciscosystems.cd/en/US/prod/collateral/vpndevc/cisco_2009_asr.pdf.

[3] See http://www.rockresearch.com/news_020305.php.

[4] See http://www.mcafee.com/in/resources/reports/rp-spam-march-2009.pdf.

per hour, it would suffer about US$ 182,500 per year in lost productivity. This works out to more than US$ 41,000 per 1 percent of spam allowed into a company. Fortunately, many solutions are being proposed to avoid this "plague" and one of more promising is the use of machine learning techniques for automatically filtering e-mail messages.

Many methods have been proposed to automatic spam filtering, such as rule-based approaches, white and blacklists, collaborative spam filtering, challenge and response systems, methods which take into account the sender's domain, clustering [24], among many others. However, among all proposed techniques, machine learning algorithms have been achieved more success [22]. These methods include approaches that are considered top-performers in text categorization, like rule induction algorithm [20,21], Rocchio [31,43,10], Boosting [19,40], decision tree [48], memory-based learning [12], Naïve Bayes (NB) classifiers [42,11,50,46,3,5,7] and support vector machines (SVM) [25,34,30,39,26,1]. The two latter currently appear to be the best anti-spam filters presented in the literature [23,22,49,8].

It has been observed that compression-based techniques seem to provide a promising alternative approach to categorization tasks, as stated by Frank et al. [27]. However, the authors showed that for text categorization, such methods do not compete with the state-of-the-art techniques. They clearly state that they do not believe their results are specific to the analyzed compression models, because if the occurrence of a single word determines whether a message belongs to a category or not, any compression scheme would likely fail to classify the message correctly. According to the authors, machine learning schemes fare better because they automatically eliminate irrelevant features.

Similar to the work of Frank et al. [27], Teahan and Harper [47] performed extensive experiments to evaluate the performance of different approaches for text categorization on the standard Reuters-21578 collection. They compared compression-based algorithms, such as prediction by partial matching (PPM) with Naïve Bayes classifiers and SVM. Based on the obtained results, the authors conclude that compression-based models perform better than word-based Naïve Bayes techniques and approach the performance of linear SVM.

Fortunately, Bratko et al. [18] investigated the performance achieved by data compression models in spam filtering task. They evaluate the filtering performance of two dif ferent compression algorithms: dynamic Markov compression (DMC) and prediction by partial matching (PPM). The results of their evaluation indicate that compression models are surprisingly good and their performances are comparable with Bogofilter, the best spam filter at that time. However, the most current published results [16,22,29] including TREC 2007 and CEAS 2008 Live Competition anti-spam challenges, indicate that the spam filters based on Naïve Bayes and SVM classifiers outperform such compression-based models.

A relatively recent method for inductive inference which is still rarely employed in text categorization tasks is the minimum description length (MDL) principle. It states that the best explanation, given a limited set of observed data, is the one that yields the greatest compression of the data [41,15,28].

In this paper, we present a spam filtering approach based on the MDL principle and compare its performance with seven different models of Naïve Bayes classifiers and the SVMs. Here, we carry out an evaluation with the practical purpose of filtering e-mail spams in order to compare the currently top-performer's spam filters. We have conducted an empirical experiment using six well-known, large, and public databases and the reported results indicate that our approach outperforms currently established spam filters.

A preliminary version of this work was presented at ACM SAC 2010 [4]. Here, we significantly extend the performance evaluation. First, we offer much more details about the new method, its main features and how it works. Second, and the most important, we compare the proposed approach with several established classifiers instead of only two, as presented in the mentioned paper.

The remainder of this paper is organized as follows: Sect. 2 presents the basic concepts regarding the main spam filtering techniques. In Sect. 3, we describe a new approach based on the MDL principle. Experimental results are showed in Sect. 4. Finally, Sect. 5 offers conclusions and directions for future work.

2 Basic concepts

In the setting of spam filtering, there are only two category labels: `spam` and `legitimate` (also called `ham`). Each message $m \in \mathcal{M}$ can only be assigned to one of them, but not to both.

Assuming that each message m is composed by a set of tokens $m = \{t_1, \dots, t_{|m|}\}$, where each token t_k corresponds to a word ("adult", for example), a set of words ("to be removed"), or a single character ("$"), we can represent each e-mail by a vector $\mathbf{x} = \langle x_1, \dots, x_{|m|}\rangle$, where $x_1, \dots, x_{|m|}$ are values of the attributes $X_1, \dots, X_{|m|}$ associated with the tokens $t_1, \dots, t_{|m|}$. In the simplest case, each term represents a single token and all attributes are Boolean: $X_i = 1$ if the message contains t_i or $X_i = 0$, otherwise.

Alternatively, attributes may be an integer computed by token frequencies (TF) representing how many times each token occurs in the message. A third alternative is to associate each attribute X_i to a normalized TF, $x_i = \frac{n(t_i)}{|m|}$, where $n(t_i)$ is the number of occurrences of the token represented by X_i in m, and $|m|$ is the length of m measured in token occurrences.

Normalized TF takes into account the term repetition versus the size of message [13].

3 Spam filtering based on minimum description length principle

The MDL principle is a formalization of Occam's razor in which the best hypothesis for a given set of data is the one that yields compact representations. The traditional MDL principle states that the preferred model results in the shortest description of the model and the data, given this model. In other words, the model that best compresses the data is selected. This model selection criterion naturally balances the complexity of the model and the degree to which this model fits the data. This principle was first introduced by Rissanen [41] and it becomes an important concept in information theory.

Let $\mathcal{Z}$ be a finite or countable set and let P be a probability distribution on $\mathcal{Z}$. Then there exists a prefix code C for $\mathcal{Z}$ such that for all $z \in \mathcal{Z}$, $L_C(z) = \lceil -\log_2 P(z) \rceil$. C is called the code corresponding to P. Similarly, let C be a prefix code for $\mathcal{Z}$. Then there exists a (possibly defective) probability distribution P such that for all $z \in \mathcal{Z}$, $-\log_2 P'(z) = L_{C'}(z)$. P' is called the probability distribution corresponding to C'. Thus, large probability according to P means small code length according to the code corresponding to P and vice versa [41,15,28].

The goal of statistical inference may be cast as trying to find regularity in the data. Regularity may be identified with ability to compress. MDL combines these two insights by viewing learning as data compression: it tells us that, for a given set of hypotheses $\mathcal{H}$ and data set $\mathcal{D}$, we should try to find the hypothesis or combination of hypotheses in $\mathcal{H}$ that compresses $\mathcal{D}$ most [41,15,28].

This idea can be applied to all sorts of inductive inference problems, but it turns out to be most fruitful in problems of model selection and, more generally, dealing with overfitting [28]. An important property of MDL methods is that they provide automatic and inherent protection against overfitting and can be used to estimate both the parameters and the structure of a model. In contrast, to avoid overfitting when estimating the structure of a model, traditional methods such as maximum likelihood must be modified and extended with additional, typically ad hoc principles [28].

In essence, compression algorithms can be applied to text categorization by building one compression model from the training documents of each class and using these models to evaluate the target document.

3.1 The MDL anti-spam filter

Given a set of classified training messages $\mathcal{M}$, the task is to assign a target e-mail m with an unknown label to one of the classes $c \in \{spam, ham\}$. So, the method measures the increase of the description length of the data set as a result of the addition of the target document. Finally, it chooses the class for which the description length increase is minimal.

We consider in this work, each class (model) c as a sequence of terms extracted from the messages and inserted into the training set. Each term t from m has a code length L_t based on the sequence of terms 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 term 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 terms of class. Let $n_c(t_i)$ the number of times that t_i appears in messages of class c, then the probability that any term belongs to c is given by the maximum likelihood estimation:

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

where n_c corresponds to the sum of $n_c(t_i)$ for all terms which appear in messages that belongs to c and $|\Omega|$ is the vocabulary size. In this work, we assume that $|\Omega| = 2^{32}$, i.e., each term 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.

The proposed MDL anti-spam filter classify a message by following these steps:

1. Tokenization: the classifier extracts all terms of the new message $m = \{t_1, \ldots, t_{|m|}\}$;
2. Compute the increase of the description length when m is assigned to each class $c \in \{spam, ham\}$:

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

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

3. if $L_m(spam) < L_m(ham)$, then

$$mdlOutput = 1 - \left(\frac{L_m(spam)}{L_m(ham)} \right)$$

   and m is classified as spam; otherwise,

$$mdlOutput = -1 \times \left[1 - \left(\frac{L_m(ham)}{L_m(spam)} \right) \right]$$

   and m is labeled as ham.
4. Training method.

In the following, we offer more details about the steps 1 and 4.

3.2 Preprocessing and tokenization

We did not perform language-specific preprocessing techniques such as word stemming, stop word removal, or case folding, since other researchers found that such techniques tend to hurt spam-filtering accuracy [51,38,22]. However, we use an e-mail-specific preprocessing before the classification task. In this way, we employ the Jaakko Hyvattis normalizemime.[5] This program converts the character set to UTF-8, decoding Base64, Quoted-Printable and URL encoding and adding warn tokens in case of encoding errors. It also appends a copy of HTML/XML message bodies with most tags removed, decodes HTML entities and limits the size of attached binary files.

Tokenization is the first stage in the classification pipeline; it involves breaking the text stream into tokens ("terms"), usually by means of a regular expression. We consider in this work that terms 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 [45]. The actual tokenization schema is defined by the following regular expression: `[^\p{Z}\p{C}][-\p{L}\p{M}\p{N}]*[^\p{Z}\p{C}]?`

These pattern use Unicode categories: `[^\p{Z}\p{C}]` means everything except whitespace and control chars (POSIX [:graph:]); \p{L}\p{M}\p{N} collectively match all alphanumerical characters ([:alnum:] in POSIX).

As proposed by Drucker et al. [25] and Metsis et al. [38], we do not consider the number of times a token appears in each message. In this way, each token is computed only once per message it appears.

3.3 Training method

The training method is basically responsible to update and store the number of times each term appears in the messages of each class. Therefore, for each message $m = \{t_1, \ldots, t_{|m|}\}$ to be trained, the MDL spam filter performs the following simple step:

1. For each term t_i of m do:
   (a) Search for t_i in the training database;
   (b) If t_i is found then update the number of messages on the class of m that t_i has appeared, otherwise insert t_i in the database.

As can be seen, a good point of the MDL classifier is that we can start with an empty training set, and according to the user feedback, the classifier builds the models for each class. Moreover, it is not necessary to keep the messages used for training since the models are incrementally building by the term frequencies. As the tokens presented in the training set are kept in a lexicographical order, so the computational complexity to train each message is of the order of $O(|m| \log n)$, where $|m|$ is the number of terms presented in the message and n is the amount of tokens in the training set. Therefore, besides the proposed approach is incrementally updateable, it is also very fast to construct, especially when compared with other established methods. Note that, for training, the Naïve-Bayes classifier has a computational complexity equivalent to $O(|m|\ n)$ [38,2,9] and the linear SVM, $O(|m|\ n^2)$ [17].

Anti-spam classifiers generally build their predicting models by learning from examples. A basic training method is to start with an empty model, classify each new sample and train it in the right class if the classification is wrong. This is known as train on error (TOE). An improvement to this method is to train also when the classification is right, but the score is near the boundary—that is, train on near error (TONE). This method is also called thick threshold training [45].

The advantage of TONE over TOE is that it accelerates the learning process by exposing the filter to additional hard-to-classify samples in the same training period. Therefore, we employ the TONE as training method used by the proposed MDL anti-spam filter.

In our evaluations, we empirically set the uncertainty interval (TONE threshold) $\Sigma = [-0.1, 0.1]$. It means that if $mdlOutput \in \Sigma$, the training method is requested.

4 Experimental results

We performed this study on the six well-known, large, real and public Enron datasets.[6] Enron corpora tries to keep the same characteristics of a real user mailbox. It is composed by legitimate messages extracted from the mailboxes of six former employees of the Enron Corporation. The composition of each dataset is shown in Table 1. For more details and statistics, refer to [38].

Tables 2, 3, 4, 5, 6 and 7 present the performance achieved by each classifier for each Enron dataset. Bold values indi-

[5] Available at http://hyvatti.iki.fi/~jaakko/spam/.

[6] The Enron datasets are available at http://www.iit.demokritos.gr/skel/i-config/.

Table 1 Amount of messages in each Enron dataset

Dataset	No. of legitimate	No. of spam	Total
Enron 1	3,672	1,500	5,172
Enron 2	4,361	1,496	5,857
Enron 3	4,012	1,500	5,512
Enron 4	1,500	4,500	6,000
Enron 5	1,500	3,675	5,175
Enron 6	1,500	4,500	6,000
Total	16,545	17,171	33,716

cate the highest score. As pointed out by Cormack [22] and Almeida et al. [7], in order to provide a fair evaluation, we consider the most important measures, the Matthews correlation coefficient (MCC) [36] and the weighted accuracy rate ($\text{Acc}_w\%$) [13] achieved by each filter. MCC provides a balanced evaluation of the prediction, especially if the two classes are of different sizes [14,7]. Moreover, it returns a value inside a predefined range, which provides more information about the classifiers' performance. 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. In addition, we present other well-known measures as spam recall (Sre%), legitimate recall (Lre%), spam precision (Spr%), legitimate precision (Lpr%), total cost ratio (TCR) [13] and elapsed time in seconds[T (s)]. It is important to note that TCR offers an indication of the improvement provided by the filter. A greater TCR indicates better performance, and for TCR <1, it is better to use no filter. All tests were performed on a computer with an Intel Core 2 Duo 2.66 GHz, 4GB RAM and OS Linux Ubuntu 10.4.

The results achieved by the proposed MDL anti-spam filter are compared with the ones attained by methods considered the actual top performers in anti-spam filtering: seven different models of NB classifiers (Basic NB [42], multinomial term frequency NB [MN TF NB] [37], multinomial Boolean NB [MN Bool NB] [44], multivariate Bernoulli NB [MV Bern NB] [35], Boolean NB [Bool NB] [38], multivariate Gauss NB [Gauss NB] [38], and flexible Bayes [Flex Bayes] [32]) and linear SVM with Boolean attributes [25,34,30,26].

It is important to point out that all the Naïve Bayes classifiers and the proposed MDL spam filter were implemented in Matlab. For implementation details and parameters of each Naïve Bayes approach, refer [1]. On the other hand, the evaluated SVM technique is provided by the well-known LibSVM toolbox.[7] We have used the linear kernel with the default parameters, as suggested by Kolcz and Alspector [34]. Regarding the training method, it is important to note that only the MDL classifier employs the train on near error strategy with uncertainty interval $\Sigma = [-0.1\ 0.1]$.

A comprehensive set of results, including all tables and figures, is available at http://www.dt.fee.unicamp.br/~tiago/research/spam/.

Regarding the results achieved by the classifiers, the MDL spam filter outperformed the other methods for the majority e-mail collections used in our empirical evaluation. It is important to realize that in some situations, the MDL performs much better than SVM and NB classifiers. For instance, for Enron 1 (Table 2), MDL achieved spam recall rate equal to 92 % while SVM attained 83.33 %, even though MDL presented better legitimate recall. It means that for Enron 1 MDL was able to recognize over 8 % more of spam than SVM, representing an improvement of 10.40 %. In a real situation, this difference would be extremely important. Note that, the same result can be found for Enron 2 (Table 3), Enron 5 (Table 6), and Enron 6 (Table 7). Both methods, MDL and SVM, achieved similar performance with no significant statistical difference just for Enron 3 (Table 4) and Enron 4 (Table 5).

The results indicate that the data compression model is more efficient to distinguish messages as spams or hams than other compared spam filters. It achieved an impressive average accuracy rate higher than 97 % and high precision × recall rates for all datasets indicating that the MDL classifier makes few mistakes. We also verify that the MDL classifier achieved an average MCC score higher than 0.925 for all tested e-mail collections. It clearly indicates that the proposed filter almost accomplished a perfect prediction (MCC = 1.000) and it is much better than not using a filter (MCC = 0.000).

Among the evaluated NB classifiers, the results indicate that all of them achieved similar performance with no significant statistical difference. However, they achieved lower results than MDL and SVM, which attained accuracy rate higher than 90 % for the most of Enron datasets.

Nevertheless, it is important to note that TCR is really not an informative measurement, as previously argued by Metsis et al. [38] and Cormack [22]. For instance, for Enron 4 (Table 5), MDL and SVM achieved similar performances ($\text{MCC}_{\text{MDL}} = 0.945$ and $\text{MCC}_{\text{SVM}} = 0.978$). However, their TCR are very different ($\text{TCR}_{\text{MDL}} = 34.615$ and $\text{TCR}_{\text{SVM}} = 90.000$), besides their precision × recall rates are very close.

Regarding time performance, in all tests, the MDL spam filter spent less or equal time than other compared techniques. Note that, the time consumed by the methods to process Enron 2 (Table 3) and 6 (Table 7) is higher than other Enron

[7] The LibSVM toolbox for Matlab is free available to download at http://www.csie.ntu.edu.tw/~cjlin/libsvm/.

Table 2 Enron 1: results achieved by each filter

Classifiers	Sre (%)	Spr (%)	Lre (%)	Lpr (%)	Acc$_w$ (%)	TCR	MCC	*T* (s)
Basic NB	91.33	85.09	93.48	96.36	92.86	4.054	0.831	84
MN TF NB	82.00	73.21	87.77	92.29	86.10	2.083	0.676	94
MN Bool NB	82.67	60.19	77.72	91.67	79.15	1.389	0.560	84
MV Bern NB	72.67	60.56	80.71	87.87	78.38	1.339	0.508	84
Bool NB	96.00	52.55	64.67	97.54	73.75	1.103	0.551	84
Gauss NB	85.33	89.51	95.92	94.13	92.86	4.054	0.824	126
Flex Bayes	86.67	88.44	95.38	94.61	92.86	4.054	0.825	132
SVM	83.33	87.41	95.11	93.33	91.70	3.488	0.796	248
MDL	92.00	92.62	97.01	96.75	**95.56**	6.552	**0.892**	83

Table 3 Enron 2: results achieved by each filter

Classifiers	Sre (%)	Spr (%)	Lre (%)	Lpr (%)	Acc$_w$ (%)	TCR	MCC	*T* (s)
Basic NB	80.00	97.56	99.31	93.53	94.38	4.545	0.850	150
MN TF NB	75.33	96.58	99.08	92.13	93.02	3.659	0.812	166
MN Bool NB	76.00	98.28	99.54	92.36	93.53	3.947	0.827	150
MV Bern NB	66.00	81.82	94.97	89.06	87.56	2.055	0.657	152
Bool NB	95.33	81.25	92.45	98.30	93.19	3.750	0.836	150
Gauss NB	75.33	98.26	99.54	92.16	93.36	3.846	0.823	208
Flex Bayes	64.00	97.96	99.54	88.96	90.46	2.679	0.743	220
SVM	90.67	90.67	96.80	96.80	95.23	5.357	0.875	401
MDL	91.33	99.28	99.77	97.10	**97.31**	10.714	**0.937**	148

Table 4 Enron 3: results achieved by each filter

Classifiers	Sre (%)	*Spr*(%)	*Lre*(%)	*Lpr*(%)	Acc$_w$ (%)	*TCR*	MCC	T(*s*)
Basic NB	58.00	100.00	100.00	86.45	88.59	2.381	0.708	52
MN TF NB	62.00	100.00	100.00	87.58	89.67	2.632	0.737	60
MN Bool NB	60.00	100.00	100.00	87.01	89.13	2.500	0.723	52
MV Bern NB	100.00	85.23	93.53	100.00	95.29	5.769	0.893	52
Bool NB	95.33	87.73	95.02	98.20	95.11	5.556	0.881	52
Gauss NB	55.33	97.65	99.50	85.65	87.50	2.174	0.676	82
Flex Bayes	52.67	97.53	99.50	86.78	72.83	2.055	0.656	90
SVM	91.33	96.48	98.76	96.83	96.74	8.333	0.917	159
MDL	90.00	100.00	100.00	96.40	**97.28**	10.000	**0.931**	52

Table 5 Enron 4: results achieved by each filter

Classifiers	Sre (%)	Spr (%)	Lre (%)	Lpr (%)	Acc$_w$ (%)	TCR	MCC	*T* (s)
Basic NB	95.33	100.00	100.00	87.72	96.50	21.429	0.914	52
MN TF NB	94.00	100.00	100.00	84.75	95.50	16.667	0.893	60
MN Bool NB	97.11	100.00	100.00	92.02	97.83	34.615	0.945	52
MV Bern NB	98.22	100.00	100.00	94.94	98.67	56.250	0.966	52
Bool NB	98.00	100.00	100.00	94.34	98.50	50.000	0.962	52
Gauss NB	94.22	100.00	100.00	85.23	95.67	17.308	0.896	84
Flex Bayes	95.78	100.00	100.00	88.76	96.83	23.684	0.922	91
SVM	98.89	100.00	100.00	96.77	**99.17**	90.000	**0.978**	161
MDL	97.11	100.00	100.00	92.02	97.83	34.615	0.945	51

Table 6 Enron 5: results achieved by each filter

Classifiers	Sre (%)	Spr (%)	Lre(%)	Lpr (%)	Acc$_w$ (%)	TCR	MCC	T(s)
Basic NB	90.22	98.81	97.33	80.22	92.28	9.200	0.832	48
MN TF NB	88.59	100.00	100.00	78.12	91.89	8.762	0.832	54
MN Bool NB	95.11	100.00	100.00	89.29	96.53	20.444	0.922	48
MV Bern NB	98.10	91.86	78.67	94.40	92.47	9.436	0.814	48
Bool NB	85.87	100.00	100.00	74.26	89.96	7.077	0.799	48
Gauss NB	88.59	99.39	98.67	77.89	91.51	8.364	0.821	86
Flex Bayes	91.58	98.54	96.67	82.39	93.05	10.222	0.845	92
SVM	89.40	99.70	99.33	79.26	92.28	9.200	0.837	166
MDL	99.73	98.39	96.00	99.31	**98.65**	52.571	**0.967**	48

Table 7 Enron 6: results achieved by each filter

Classifiers	Sre (%)	Spr (%)	Lre (%)	Lpr (%)	Acc$_w$ (%)	TCR	MCC	T(s)
Basic NB	87.78	99.00	97.33	72.64	90.17	7.627	0.781	102
MN TF NB	75.78	99.42	98.67	57.59	81.50	4.054	0.651	118
MN Bool NB	93.33	97.45	92.67	82.25	93.17	10.976	0.828	102
MV Bern NB	96.00	92.31	76.00	86.36	91.00	8.333	0.753	106
Bool NB	66.67	99.67	99.33	49.83	74.83	2.980	0.572	102
Gauss NB	89.56	98.05	94.67	75.13	90.83	8.182	0.785	162
Flex Bayes	94.22	97.03	91.33	84.05	93.50	11.538	0.833	180
SVM	89.78	95.28	86.67	73.86	89.00	6.818	0.727	316
MDL	98.67	95.48	86.00	95.56	**95.50**	16.667	**0.878**	99

databases. It is because such datasets are composed by a lot of large messages, with big content.

5 Conclusions and further work

In this paper, we have presented a new spam filtering approach based on the MDL principle that has proved to be very fast to construct and incrementally updateable. We have also compared its performance with the well-known lin ear SVM and seven different models of Naïve Bayes classifiers, something the spam literature does not always acknowledge.

To evaluate the proposed approach we have conducted empirical experiments using well-known, real, large and pub lic databases and the reported results indicate that the proposed classifier outperforms currently established spam filters. It is important to emphasize that MDL spam filter has obtained the best average performance for all analyzed collections presenting an average accuracy rate higher than 97 % for all e-mail datasets.

Actually, we are conducting more experiments to compare our approach with other compression-based methods, along with commercial and open-source anti-spam filters, such as DMC, PPM, Bogofilter, SpamAssassin and ProcMail, among others.

Future works include evaluating the MDL spam filter to classify messages in environments where the text have rigid restriction in length, such as SMS spam [6], blog spam and social network spam, among others.

Acknowledgments This work is supported by the Brazilian funding agencies CNPq, CAPES and FAPESP.

References

1. Almeida T, Yamakami A (2010) Content-based spam filtering. In: Proceedings of the 23rd IEEE International Joint Conference on Neural Networks. Barcelona, Spain, pp 1–7
2. Almeida T, Yamakami A (2011) Redução de Dimensionalidade Aplicada na Classificação de Spams Usando Filtros Bayesianos. Revista Brasileira de Computação Aplicada 3(1):16–29
3. Almeida T, Yamakami A, Almeida J (2009) 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. Miami, FL, USA, pp 517–522
4. Almeida T, Yamakami A, Almeida J (2010a) Filtering spams using the minimum description length principle. In: Proceedings of the 25th ACM Symposium on Applied Computing. Sierre, Switzerland, pp 1856–1860
5. Almeida T, Yamakami A, Almeida J (2010b) Probabilistic antispam filtering with dimensionality reduction. In: Proceedings of the 25th ACM Symposium On Applied Computing. Sierre, Switzerland, pp 1804–1808
6. Almeida T, Hidalgo JG, Yamakami A (2011a) Contributions to the study of SMS spam filtering: new collection and results. In:

Proceedings of the 2011 ACM Symposium on Document Engineering. Mountain View, CA, USA, pp 259–262

7. Almeida T, Almeida J, Yamakami A (2011b) Spam filtering: how the dimensionality reduction affects the accuracy of Naive Bayes classifiers. J Internet Serv Appl 1(3):183–200
8. Almeida TA, Yamakami A (2012a) Advances in spam filtering techniques. In: Elizondo D, Solanas A, Martinez-Balleste A (eds) Com putational intelligence for privacy and security. Studies in computational intelligence. vol 394. Springer, Berlin, pp 199–214
9. Almeida TA, Yamakami A (2012b) Facing the spammers: a very effective approach to avoid junk e-mails. Expert Syst Appl: 1–5
10. Anagnostopoulos A, Broder A, Punera K (2008) Effective and efficient classification on a search-engine model. Knowl Inf Syst 16(2):129–154
11. Androutsopoulos I, Koutsias J, Chandrinos K, Paliouras G, Spyropoulos C (2000a) An evalutation of Naive Bayesian anti-spam filtering. In: Proceedings of the 11th European Conference on Machine Learning. Barcelona, Spain, pp 9–17
12. Androutsopoulos I, Paliouras G, Karkaletsis V, Sakkis G, Spyropoulos C, Stamatopoulos P (2000b) Learning to filter spam e-mail: a comparison of a Naive Bayesian and a memory-based approach. In: Proceedings of the 4th European Conference on Principles and Practice of Knowledge Discovery in Databases. Lyon, France, pp 1–13
13. Androutsopoulos I, Paliouras G, Michelakis E (2004) Learning to filter unsolicited commercial e-mail. Technical Report 2004/2, National Centre for Scientific Research "Demokritos", Athens, Greece
14. Baldi P, Brunak S, Chauvin Y, Andersen C, Nielsen H (2000) Assessing the accuracy of prediction algorithms for classification: an overview. Bioinformatics 16(5):412–424
15. Barron A, Rissanen J, Yu B (1998) The minimum description length principle in coding and modeling. IEEE Trans Inf Theory 44(6):2743–2760
16. Blanzieri E, Bryl A (2008) A survey of learning-based techniques of email spam filtering. Artif Intell Rev 29(1):335–455
17. Bordes A, Ertekin S, Weston J, Bottou L (2005) Fast kernel classifiers with online and active learning. J Mach Learn Res 6:1579–1619
18. Bratko A, Cormack G, Filipic B, Lynam T, Zupan B (2006) Spam filtering using statistical data compression models. J Mach Learn Res 7:2673–2698
19. Carreras X, Marquez L (2001) Boosting trees for anti-spam email filtering. In: Proceedings of the 4th International Conference on Recent Advances in Natural Language Processing. Tzigov Chark, Bulgaria, pp 58–64
20. Cohen W (1995) Fast effective rule induction. In: Proceedings of 12th International Conference on Machine Learning. Tahoe City, CA, USA, pp 115–123
21. Cohen W (1996) Learning rules that classify e-mail. In: Proceedings of the AAAI Spring Symposium on Machine Learning in Information Access. CA, USA, Stanford, pp 18–25
22. Cormack G (2008) Email spam filtering: a systematic review. Found Trends Inf Retr 1(4):335–455
23. Cormack G, Lynam T (2007) Online supervised spam filter evaluation. ACM Trans Inf Syst 25(3):1–11
24. Czarnowski I (2011) Cluster-based instance selection for machine classification. Knowl Inf Syst
25. Drucker H, Wu D, Vapnik V (1999) Support vector machines for spam categorization. IEEE Trans Neural Netw 10(5):1048–1054
26. Forman G, Scholz M, Rajaram S (2009) Feature shaping for linear SVM classifiers. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. France, Paris, pp 299–308
27. Frank E, Chui C, Witten I (2000) Text categorization using compression models. In: Proceedings of the 10th Data Compression Conference. Snowbird, UT, USA, pp 555–565
28. Grünwald P (2005) A tutorial introduction to the minimum description length principle. In: Grünwald P, Myung I, Pitt M (eds) Advances in minimum description length: theory and applications. MIT Press, Cambridge, pp 3–81
29. Guzella T, Caminhas W (2009) A review of machine learning approaches to spam filtering. Expert Syst Appl 36(7):10206–10222
30. Hidalgo J (2002) Evaluating cost-sensitive unsolicited bulk mail categorization. In: Proceedings of the 17th ACM Symposium on Applied Computing. Madrid, Spain, pp 615–620
31. Joachims T (1997) A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In: Proceedings of 14th International Conference on Machine Learning. Nashville, TN, USA, pp 143–151
32. John G, Langley P (1995) Estimating continuous distributions in Bayesian classifiers. In: Proceedings of the 11th International Conference on Uncertainty in Artificial Intelligence. Montreal, Canada, pp 338–345
33. Katakis I, Tsoumakas G, Vlahavas I (2009) Tracking recurring contexts using ensemble classifiers: an application to email filtering. Knowl Inf Syst 22(3):371–391
34. Kolcz A, Alspector J (2001) SVM-based filtering of e-mail spam with content-specific misclassification costs. In: Proceedings of the 1st International Conference on Data Mining. San Jose, CA, USA, pp 1–14
35. Losada D, Azzopardi L (2008) Assessing multivariate Bernoulli models for information retrieval. ACM Trans Inf Syst 26(3):1–46
36. Matthews B (1975) Comparison of the predicted and observed secondary structure of T4 phage lysozyme. Biochimica et Biophysica Acta 405(2):442–451
37. McCallum A, Nigam K (1998) A comparison of event models for Naive Bayes text classication. In: Proceedings of the 15th AAAI Workshop on Learning for Text Categorization. Menlo Park, CA, USA, pp 41–48
38. Metsis V, Androutsopoulos I, Paliouras G (2006) Spam filtering with Naive Bayes—which Naive Bayes? In: Proceedings of the 3rd International Conference on Email and Anti-Spam. Mountain View, CA, USA, pp 1–5
39. Peng T, Zuo W, He F (2008) SVM based adaptive learning method for text classification from positive and unlabeled documents. Knowl Inf Syst 16(3):281–301
40. Reddy C, Park J-H (2010) Multi-resolution boosting for classification and regression problems. Knowl Inf Syst
41. Rissanen J (1978) Modeling by shortest data description. Automatica 14:465–471
42. Sahami M, Dumais S, Hecherman D, Horvitz E (1998) A Bayesian approach to filtering junk e-mail. In: Proceedings of the 15th National Conference on Artificial Intelligence. Madison, WI, USA, pp 55–62
43. Schapire R, Singer Y, Singhal A (1998) Boosting and Rocchio applied to text filtering. In: Proceedings of the 21st Annual International Conference on Information Retrieval. Melbourne, Australia, pp 215–223
44. Schneider K (2004) On word frequency information and negative evidence in Naive Bayes text classification. In: Proceedings of the 4th International Conference on Advances in Natural Language Processing. Alicante, Spain, pp 474–485
45. Siefkes C, Assis F, Chhabra S, Yerazunis W (2004) 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. Pisa, Italy, pp 410–421

46. Song Y, Kolcz A, Gilez C (2009) Better Naive Bayes classification for high-precision spam detection. Softw Pract Experience 39(11):1003–1024

47. Teahan W, Harper D (2001) Using compression-based language models for text categorization. In: Proceedings of the 2001 Workshop on Language Modeling and Information Retrieval. Pittsburgh, PA, USA, pp 1–5

48. Wozniak M (2010) A hybrid decision tree training method using data streams. Knowl Inf Syst

49. Wu X, Kumar V, Quinlan J, Ghosh J, Yang Q, Motoda H, McLachlan G, Ng A, Liu B, Yu P, Zhou Z, Steinbach M, Hand D, Steinberg D (2008) Top 10 algorithms in data mining. Knowl Inf Syst 14(1):1–37

50. Zhang J, Kang D, Silvescu A, Honavar V (2006) Learning accurate and concise Naive Bayes classifiers from attribute value taxonomies and data. Knowl Inf Syst 9(2):157–179

51. Zhang L, Zhu J, Yao T (2004) An evaluation of statistical spam filtering techniques. ACM Trans Asian Lang Inf Process 3(4):243–269