Expert Systems with Applications 36 (2009) 10206–10222

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Review

A review of machine learning approaches to Spam filtering

Thiago S. Guzella *, Walmir M. Caminhas

Department of Electrical Engineering, Federal University of Minas Gerais, Ave. Antonio Carlos, 6627, Belo Horizonte (MG) 31270-910, Brazil

ARTICLE INFO

Keywords:
Spam filtering
Online learning
Bag-of-words (BoW)
Naive Bayes
Image Spam

ABSTRACT

In this paper, we present a comprehensive review of recent developments in the application of machine learning algorithms to Spam filtering, focusing on both textual- and image-based approaches. Instead of considering Spam filtering as a standard classification problem, we highlight the importance of considering specific characteristics of the problem, especially concept drift, in designing new filters. Two particularly important aspects not widely recognized in the literature are discussed: the difficulties in updating a classifier based on the bag-of-words representation and a major difference between two early naive Bayes models. Overall, we conclude that while important advancements have been made in the last years, several aspects remain to be explored, especially under more realistic evaluation settings.

1. Introduction

In recent years, the increasing use of e-mail has led to the emergence and further escalation of problems caused by unsolicited bulk e-mail messages, commonly referred to as Spam. Evolving from a minor nuisance to a major concern, given the high circulating volume and offensive content of some of these messages, Spam is beginning to diminish the reliability of e-mail (Hoanca, 2006). Personal users and companies are affected by Spam due to the network bandwidth wasted receiving these messages and the time spent by users distinguishing between Spam and normal (legitimate or ham) messages. A business model relying on Spam marketing is usually advantageous because the costs for the sender are small, so that a large number of messages can be sent, maximizing the returns, this aggressive behavior being one of the defining characteristics of *Spammers* (those that send Spam messages) (Martin-Herran, Rubel, & Zaccour, 2008). The economical impacts of Spam have led some countries to adopt legislation (e.g., Carpinter & Hunt,2006; Hoanca, 2006; Stern, 2008), although it is limited by the fact that many such messages are sent from various countries (Talbot, 2008). Besides, difficulties in tracking the actual senders of these messages can also limit the application of such laws. In addition to legislation, some authors have proposed changes in protocols and operation models (discussed in Hoanca (2006)).

Another approach adopted is the use of Spam filters, which, based on analysis of the message contents and additional information, attempt to identify Spam messages. The action to be taken once they are identified usually depends on the setting in which the filter is applied. If employed by a single user, as a client-side filter, they are usually sent to a folder which contains only Spam-labeled messages, making the identification of these messages easier. In contrast, if the filter operates in a mail server, handling messages from several users, they may either be labeled as Spam or deleted. Another possibility is a collaborative setting, in which filters running in different machines share information on the messages received, to improve their performance.

However, the use of filters has created an evolutionary scenario (Goodman, Cormack, & Heckerman, 2007; Hayes, 2007), in which *Spammers* employ tools (Stern, 2008) with various techniques specifically tailored to minimize the number of messages identified. Initially, Spam filters were based on user-defined rules, designed based on knowledge of regularities easily observed in such messages. In response, *Spammers* then began employing content "obfuscation" (or obscuring), by disguising certain terms that are very common in Spam messages (e.g., by writing "f r 3 3" instead of "free"), on an attempt to prevent the correct identification of these terms by Spam filters. Nowadays, Spam filtering is usually tackled by machine learning (ML) algorithms, aimed at discriminating between legitimate and Spam messages, providing an automated, adaptive approach, which are the focus of this review. Instead of relying on hand-coded rules, which are prone to the constantly changing nature of Spam messages, ML approaches are capable of extracting knowledge from a set of messages supplied, and using the obtained information in the classification of newly received messages. Given a collection of training documents $\mathscr{D}_{tr} \subset \mathscr{D}$ labeled as legitimate or Spam, these algorithms can be described as learning a function $f : \mathscr{D} \to \{l, s\}$, for labeling an instance (document or message) $d \in D$ as legitimate (l) or Spam (s), referred to as the classes. Another interesting feature of these algorithms is the ability to improve their performance through experience (Mitchell, 1997). Due to the fact that most practical filters employ

* Corresponding author. Present address: Instituto Gulbenkian de Ciência, Rua da Quinta Grande 6, Oeiras 2780-156, Portugal. Tel.: +351 21 4407900.
E-mail addresses: tguzella@igc.gulbenkian.pt (T.S. Guzella), caminhas@cpdee.ufmg.br (W.M. Caminhas).

0957-4174/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2009.02.037

a combination of ML and application-specific knowledge, in the form of hand-coded rules, understanding the changing characteristics of Spam is also important, and has been considered by some researchers (Gomes, Cazita, Almeida, Almeida, & Meira, 2007; Pu & Webb, 2006; Wang & Chen, 2007). Nevertheless, despite the growing research on Spam filtering, the evolution of Spam messages is still occurring, which can be seen in the development of new techniques for evading recognition, such as messages with contents embedded in images.

Text-based Spam filtering can be considered as a text categorization problem, and some works have been designed not only for Spam filtering, but also for the problem of sorting messages into folders (e-mail categorization). However, Spam filtering has several distinguishing characteristics, which should be incorporated in a system for ensuring its applicability. As discussed by Fawcett (2003), these include skewed and changing class distributions, unequal and uncertain misclassification costs of Spam and legitimate messages, complex text patterns and concept drift (a change in a target concept, such as terms indicative of Spam messages) and provide the opportunity for the development and application of new algorithms that explore these characteristics (Fawcett, 2003). Moreover, special attention should be given to the role of user feedback, in the form of immediate or delayed corrections for updating the classification model, as a way to deal with concept drift. In particular, user feedback is a growing theme not only in Spam filtering, but also in other areas of text processing (Culotta, Kristjansson, McCallum, & Viola, 2006). Spam filtering is also increasingly considered as a benchmark for testing newly developed machine learning algorithms, not specifically designed for this problem (e.g. Camastra & Verri, 2005; Gadat & Younes, 2007; Qin & Zhang, 2008).

In line with the growing concerns regarding Spam messages, there has been an increasing number of works dedicated to the problem. Wang and Cloete (2005) surveyed some approaches for e-mail classification, including Spam filtering and e-mail categorization. A relatively recent overview of approaches aimed at Spam filtering was presented by Carpinter and Hunt (2006), which focused on more general aspects of the problem. A more recent review has been conducted by Blanzieri and Bryl (2008). However, it did not discuss several of the more recent works, such as Case-Based Reasoning models and Artificial Immune Systems, which are included in this paper. Moreover, in the present review, we discuss two important aspects not widely considered in the literature: the bias imposed by the commonly used bag-of-words representation and an important difference between naive Bayes models. We also discuss the need to evaluate a filter in a realistic setting, according to some recent corpora available. Emphasis is given to recent works, minimizing the overlap with other reviews (Blanzieri & Bryl, 2008; Carpinter & Hunt, 2006; Wang & Cloete, 2005), although some early works proposing the use of some approaches are also discussed to outline the evolution of their use. Finally, although unsolicited content is current affecting not only e-mail, but also search engines (Gyongyi & Garcia-Molina, 2005) and blogs (Kolari, Java, Finin, Oates, & Joshi, 2006), this survey focuses solely on dealing with e-mail Spam.

This paper is organized in the following way: Section 2 presents an initial background of Spam filtering, discussing typical steps involved in most filters, the representation of messages, datasets used for evaluation and performance measures usually adopted. Sections 3–10 discuss different families of algorithms applied to textual-related analysis of message contents, while Section 11 presents works dedicated to comparing filters under the same experimental setup. Section 12 focuses on approaches developed for dealing with image Spam. We attempt to present only the most distinguishing characteristics of each algorithm, and focus on the application-specific aspects and experimental scenarios considered. Special attention is given to the datasets used, as different corpora have different number of messages and characteristics. Finally, Section 13 presents an overall discussion of the methods cited in this review, along with the final conclusions of this work.

2. Background and initial discussions

2.1. Structure of a usual Spam filter

The information contained in a message is divided into the header (fields containing general information on the message, such as the subject, sender and recipient) and body (the actual contents of the message). Before the available information can be used by a classifier in a filter, appropriate pre-processing steps are required. The steps involved in the extraction of data from a message are illustrated in Fig. 1, and can be grouped into:

(1) tokenization, which extracts the words in the message body;
(2) lemmatization, reducing words to their root forms (e.g., "extracting" to "extract");
(3) stop-word removal, eliminating some words that often occur in many messages (e.g., "to", "a", "for");
(4) representation, which converts the set of words present in the message to a specific format required by the machine learning algorithm used.

This formulation considers that only information contained in the message body is used by the filter, as pre-processing the

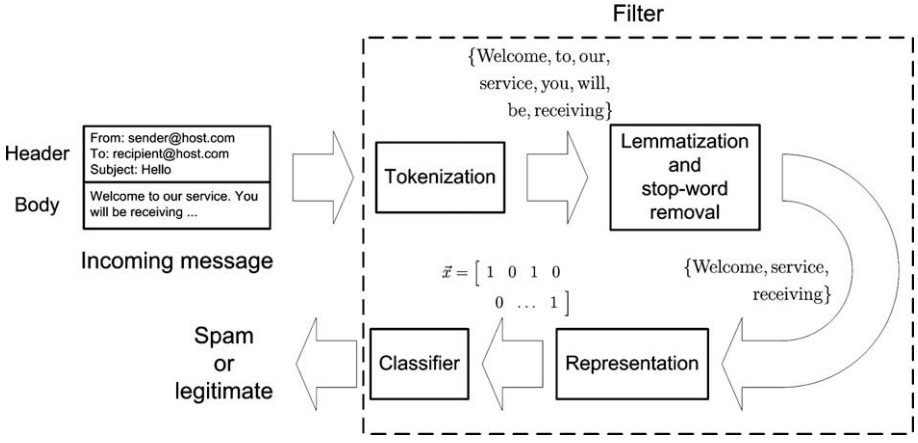

Fig. 1. An illustration of some of the main steps involved in a spam filter.

headers requires specific procedures depending on the fields considered. Moreover, it is assumed that the contents of the message have been decoded prior to analysis by the filter, as required for some messages with certain character encodings. Finally, it should be pointed out that some approaches do not require all these steps, including even methods that can operate on the raw message level, without requiring any pre-processing of the message.

2.2. Representation

A particularly important aspect in text categorization applications (Sebastiani, 2002) is the representation of text. In Spam filtering, the text is usually obtained from the body of the message, although the subject or even other header fields can be considered. One of most popular representations is commonly referred to as the bag-of-words (BoW), also known as the vector-space model. Given a set of terms $\mathscr{T} = \{t_1, t_2, \ldots, t_N\}$ selected *a priori*, it represents a document d as a N-dimensional feature vector $\vec{x} = [x_1, x_2, \ldots, x_N]$, where the value of x_i is given as a function of the occurrence of t_i in d, depending on the representation of the features adopted. The terms are generally given as words occurring in messages used for training. Some authors (e.g. Androutsopoulos, Paliouras, & Michelakis, 2004; Ciltik & Gungor, 2008; Kanaris, Kanaris, Houvardas, & Stamatatos, 2007; Sculley & Wachman, 2007a), have considered, instead, the use of character n-gram models, which are sequences of characters obtained through the application of a sliding window (e.g., for an expression "machine learning", the character 5-grams are "machi", "achin", ..., "rning"). This is particularly interesting, as tokenization is highly vulnerable to content obscuring. A further possibility is the use of n-gram word models (e.g., Zorkadis & Karras, 2006), which consider sequences of words (e.g., for an expression "machine learning methods", the word 2-gram are "machine learning" and "learning methods").

In a binary representation, a feature x_i is equal to 1 if t_i occurs in d, and 0 otherwise, while a frequency representation assigns x_i as the number of occurrences of t_i in d. Another feature representation is *tf-idf* (term frequency-inverse document frequency), which associates a value x_i to a term $t_i \in \mathscr{T}$ in a document (message) $d \in \mathscr{D}_{tr}$ according to Eq. (1), where $n_{t_i,d}$ is the number of occurrences of t_i in d and n_{t_i} denotes the number of documents in which t_i occurs:

$$x_i = n_{t_i,d} \log\left(\frac{|\mathscr{D}_{tr}|}{n_{t_i}}\right) \tag{1}$$

Finally, other representations are usually employed in text categorization problems (cf. Leopold et al., 2002; Song, Zhang, Xu, & Wang, 2007), but have not been applied to Spam filtering specifically.

Given a set $\mathscr{T}'$ containing all the terms extracted from a set of training documents $\mathscr{D}_{tr}$, a common step before representing the documents is the application of a feature selection algorithm, which selects $\mathscr{T} \subset \mathscr{T}'$ such that only the most "representative" terms are used. Some of the most commonly used methods for feature selection are shown in Table 1, where $\bar{t}_i$ denotes the absence of t_i, $c \in \{s, l\}$ is a given class and $\bar{c} = \{s, l\} \setminus c$. For the term-frequency variance, $T_f(t_i, c)$ and $T_f^{\mu}(t_i)$ represent, respectively, the term frequency of t_i in documents of class c and the mean term frequency of t_i. These methods determine a score for each term, and a given number of terms with the highest scores are selected. Other approaches considered for text categorization problems and not included herein are discussed by Sebastiani (2002). Table 1 also presents the number of works cited in this survey which have used each algorithm, either in a method proposed or in other approaches used for comparison. These algorithms have also been used in some works for weighting the features used for classification, not for selecting, but these are not included in Table 1. It can be verified that the information gain is, by far, the most commonly used, with the remaining methods being sparsely used. As pointed out by Zhang, Zhu, and Yao (2004), some authors commonly refer to the information gain as the mutual information, as the former can be seen as combining the values given by the latter considering both classes, as well as the presence and absence of the term. One method not included in Table 1 is the removal of words that appear less than $\alpha\%$ and more than $(100 - \alpha)\%$ in all the messages (used with $\alpha = 5$ in Bezerra et al. (2006) & Ruan & Tan (2007)), as it does not output a term score, being more similar to stop-word removal than feature selection.

Table 1
Some of the most common feature selection methods applied in Spam filtering.

Name	Term score	Number of works
Document frequency	$\tau(t_i) = \lvert\{d : d \in \mathscr{D}_{tr} \text{ and } t_i \in d\}\rvert$	2
Information gain	$\tau(t_i) = \sum_{c\in\{s,l\}}\sum_{t\in\{t_i,\bar{t}_i\}} P(t,c)\log\left[\frac{P(t,c)}{P(t)P(c)}\right]$	26
χ^2 statistic	$\tau(t_i,c) = \frac{\lvert\mathscr{D}_{tr}\rvert(P(t_i,c)P(\bar{t}_i,\bar{c}) - P(t_i,\bar{c})P(\bar{t}_i,c))^2}{P(t_i)P(\bar{t}_i)P(c)P(\bar{c})}$	1
Odds ratio	$\tau(t_i,c) = \frac{P(t_i\lvert c)}{1-P(t_i\lvert\bar{c})}\frac{1-P(t_i\lvert\bar{c})}{P(t_i\lvert c)}$	1
Term-frequency variance	$\tau(t_i) = \sum_{c\in\{s,l\}}(T_f(t_i,c) - T_f^{\mu}(t_i))^2$	2

One of the advantages of feature selection is that it can, in some cases, improve the classification accuracy of an algorithm, which can focus on the "relevant" features, ignoring noninformative ones. Another advantage is that subsequent processing during training and classification tends to be much more efficient, in the usual case of $N \ll |\mathscr{T}'|$, as the computational complexity of many algorithms for training and classification is typically a function of N. The combination of the BoW representation with feature selection also allows the classification model to attain a high compression rate, as it considers only a subset of the terms.

Although the BoW is very popular in approaches designed for Spam filtering, the concept drift present in this problem makes its use in a practical filter troublesome. Due to the fact that the structure of the resultant feature vectors is constant, algorithms relying on such representation may not scale well in an online setting, as it is not possible to consider a term that was not initially selected when building the model (see, e.g., Gabrilovich & Markovitch, 2007). This is illustrated in Fig. 2, in which a classifier trained using a feature vector containing four features (*money*, *buy*, *on-line* and *free*) is unable to consider either variations of these features present in incoming messages (such as *fr33* or *m0n3y*) or new features (e.g., *save*). It is important to emphasize that this inability to adapt persists even if the classification model is updated using the feature vectors originating from these new messages, as those would not contain the new features. This occurs because the BoW representation assumes that the initially selected features will always be representative for the classification of a message, which is violated when *Spammers* introduce new obscured terms.

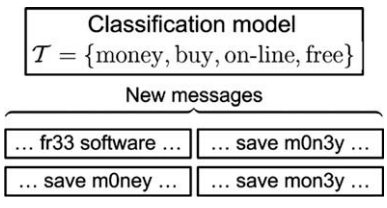

Fig. 2. Inability of a filter based on the BoW representation in adapting to newly received messages. A classifier initially trained to use a given set of features for classification is unable to consider new features which become increasingly used in new messages. To adapt to these new messages, it is necessary to re-train the classifier from scratch, so that the new features are incorporated in the classification model.

To consider a new word, it is necessary to re-train the classifier from scratch, which may be extremely inefficient if performed constantly. A possible solution for this problem is to consider all possible patterns when building the classification model, although this can lead to very high-dimensional feature vectors. This has been adopted in a character n-gram model by Sculley and Wachman (2007a), with $n \in \{3,4\}$, which leads to feasible dimensionalities (2^{24} and 2^{32}, respectively) and a good performance, as the representation of a single message is highly sparse.

It should be noted that this aspect of the BoW representation has not received much attention in the literature. This is due, in part, to the fact that most works, including recent ones, are evaluated solely in a static scenario, where a classifier is initially trained and then tested on a separate set of messages, such that the appearance of new terms is not pronounced. In contrast, the use of newly developed methods in more realistic scenarios, in which new terms appear frequently, will require a careful consideration of how this representation bias may affect the practical use of an algorithm.

2.3. Datasets

An important aspect when formulating a new Spam filter is the data used for evaluating its performance. While standard benchmark datasets have long been established for text categorization (Sebastiani, 2002), only recently have public corpora been introduced by some authors.

The public corpora used in the approaches discussed in this survey are shown in Table 2. Each corpus has very peculiar characteristics, reflected by the time of compilation, the number of users considered and the general subject of the messages included. As an example, the LingSpam corpus, one of the first publicly available datasets, compiled by Androutsopoulos, Koutsias, Chandrinos, Paliouras, and Spyropoulos (2000), contains messages collected from a mailing list on Linguistics, with legitimate messages being more topic-specific then generally expected for an user (Androutsopoulos et al., 2004). Another further important distinction is made regarding the TREC (Text REtrieval Conference) corpora, which were originally devised for online evaluations according to a well specified procedure, in which the classifier iteratively classifies a message and receives feedback. Obviously, a similar scenario could be adopted for the other corpora, although this is not common. Table 2 also includes the number of works using each corpus (note that a single work may use more than one corpus for evaluation), from which can be assessed that LingSpam and SpamAssassin are the most popular corpora, followed by PU1. This is due, in part, to the fact that these are relatively old and were used in several early works, so that several recent papers still use them for comparison. Interestingly, the TREC corpora, which are more recent and designed to mimic a real-world setting, appear to be relatively seldomly used, with the exception of TREC2005. In line with the recent efforts towards dealing with image Spam, three image datasets (Princeton Spam Image Benchmark, Dredze and Biggio) have been recently published. In summary, static corpora are, currently, more popular than those specifically designed for an online evaluation.

Due to the particularities of each corpus, comparing two filters merely on the basis of the results reported is not usually possible. For this reason, when discussing a specific work, the datasets used for evaluation are explicitly mentioned. Finally, despite the availability of several public corpora, some works use private corpora, sometimes due to the need to capture aspects (e.g., the performance for messages with mixed languages) not present in the public datasets. However, it should be noted that this compromises the reproducibility of the results reported, in addition to precluding the comparison with other filters not initially considered. Moreover, some works have considered corpora which were available some time ago, but were not accessible at the time of compiling this study, which are referred to as private corpora hereafter.

2.4. Performance measures

To evaluate the performance of a filter, performance indices typical of two distinct areas are commonly used: information retrieval (recall, precision and derived measures) and decision theory (false positives and false negatives). Independently of the indices

Table 2
Publicly available datasets.

Name	Number of messages		Description	Number of works	References
	Spam	Legitimate			
BBK	559	0	Spam messages not identified by SpamAssassin	1	Pampapathi et al. (2006)
Dornbos	20,308	0	Spam messages collected over 5 months	2	Dornbos (2002)
EM Canada	1.7 million	0	Spam messages collected since 1998, until June 2008	3	Guenter (2008)
Enron	0	20,581	Pre-processed version, divided into 7 users	6	Bekkerman (2005)
Enron-Spam	13,496	16,545	Includes Spam messages from various sources	2	Metsis et al. (2006)
GenSpam	31,196	9212	Divided into training, validation and testing	1	Medlock (2006)
Jones	6409	0	Messages collected between 1997 and 2003	1	Jones (2003)
LingSpam	481	2893	Collected from a mailing list on Linguistics	19	Androutsopoulos, Koutsias, Chandrinos, Paliouras et al. (2000)
PU1	481	618	Encrypted	12	Androutsopoulos, Koutsias, Chandrinos, and Spyropoulos (2000)
PU2	142	579	Encrypted	1	Androutsopoulos et al. (2004)
PU3	1826	2313	Encrypted	2	Androutsopoulos et al. (2004)
PUA	571	571	Encrypted	1	Androutsopoulos et al. (2004)
SpamAssassin	3793	6951	Updated in 2005	18	SpamAssassin (2005)
TREC2005	52,790	39,399	Online setting	8	Cormack and Lynam (2005)
TREC2006	24,912	12,910	Online setting	5	Cormack (2006)
TREC2007	50,199	25,220	Online setting	2	Cormack (2007)
UCI	1813	2788	Messages represented as bag-of-words with normalized frequency and 57 features	5	Asuncion and Newman (2007)
Wouters	141,329	0	Messages collected between 1997 and 2004	1	Wouters (2004)
ZH1	1205	428	Chinese corpus	1	Zhang et al. (2004)
Princeton	1071	0	Spam images, divided into 178 groups with random distortions	1	Wang et al. (2007)
Biggio	8549	0	Images collected between 2005 and 2007	1	Biggio et al. (2008)
Dredze	3297	2006	–	2	Dredze et al. (2007) and Biggio et al. (2008)

adopted, an important aspect to be considered is the asymmetry in the misclassification costs. A Spam message incorrectly classified as legitimate is a relatively minor problem, as the user is simply required to remove it. On the other hand, a legitimate message mislabeled as Spam can be unacceptable, as it implies the loss of potentially important information, particularly in configurations in which Spam messages are automatically deleted. For this reason, describing the performance of an algorithm solely in terms of the classification accuracy (the relative number of messages correctly classified) is not adequate, as it assumes equal misclassification costs for both classes. Moreover, in a highly unbalanced scenario, in which the number of Spam messages used for testing is much greater than that of legitimate messages, a classifier can attain a high accuracy by focusing only on the identification of Spam messages. In a realistic setting, in which there is a non-zero probability of incorrectly classifying a legitimate message, it is necessary to select a trade-off between the two types of errors, based on user preference and the performance indices adopted. In the subsequent presentation, we consider the application of a filter to a test dataset with n_L legitimate and n_S Spam messages, resulting in $n_{L,S}$ and $n_{S,L}$ being incorrectly classified, respectively. In this case, it clearly follows that the number of correctly classified legitimate and Spam messages are given by $n_{L,L} = n_L - n_{L,S}$ and $n_{S,S} = n_S - n_{S,L}$, respectively.

In decision theory, two classes are labeled as positive (Spam) and negative (legitimate), with the performance measures being the true positive (η_{tp}) and negative (η_{tn}) rates (Eqs. (2) and (3), respectively), corresponding to the relative number of instances of each class that have been correctly classified. From these, the false positive and negative rates can be obtained ($\eta_{fp} = 1 - \eta_{tn}$ and $\eta_{fn} = 1 - \eta_{tp}$, respectively):

$$\eta_{tp} = \frac{n_{S,S}}{n_S} \tag{2}$$

$$\eta_{tn} = \frac{n_{L,L}}{n_L} \tag{3}$$

This framework has been increasingly applied to the evaluation of machine learning algorithms which output a score which, when thresholded, determines the classification of each instance. Using the evaluation results, a ROC (Received Operating Characteristic) curve (Fawcett, 2006) can be obtained, by plotting the true positive rate as a function of the false positive rate for various threshold values. These curves allow a threshold-independent visualization of the results of the filter, and the assessment of the trade-off between false positives and negatives. Given the ROC curves of two classifiers, a single index which can be used for comparing them is the area under the ROC curve (AUC), with a higher AUC indicating a superior performance. From an statistical point of view, the AUC can be interpreted as the probability that a random Spam message will receive a score greater than a random legitimate message (Fawcett, 2006).

In an information retrieval context, it is assumed the existence of a relevant class, whose representative instances are to be retrieved from a collection of objects. A classifier is then evaluated based on the precision (the relative number of relevant instances among all those retrieved) and recall (relative number of the retrieved relevant instances that were actually retrieved). In Spam filtering, Spam is typically considered as the relevant class, so that the corresponding indices are the Spam recall (r_S) and precision (p_S), defined by Eqs. (4) and (5), respectively. It should be noted that some works also include the analogous legitimate recall and precision:

$$r_S = \frac{n_{S,S}}{n_{S,S} + n_{S,L}} \tag{4}$$

$$p_S = \frac{n_{S,S}}{n_{S,S} + n_{L,S}} \tag{5}$$

To combine recall and precision, the F_β measure is commonly employed, which assigns a weight β to the precision. Among the works in the literature using it, the value $\beta = 1$ is usually adopted, referred to as the F1 measure:

$$F_\beta = (1 + \beta^2)\frac{r_S p_S}{\beta^2 p_S + r_S} \tag{6}$$

A particular advantage of the F_β measure is that it can be used in a cost-sensitive evaluation of the classifiers, by varying the value of β. However, Androutsopoulos, Koutsias, Chandrinos, Paliouras et al. (2000) and Androutsopoulos, Koutsias, Chandrinos, and Spyropoulos (2000) argued that its use is not appropriate, as the choice of β is not easily related to the cost of misclassifying a legitimate message, and proposed a further refinement based on Spam recall and precision, to allow the evaluation of the performance of a filter based on a single measure. It considers a false positive as being λ times more costly than false negatives, with the values 1, 9 and 999 being originally proposed. Each false positive is accounted as λ mistakes, with the (weighted) overall accuracy (A_w) being given by:

$$A_w = \frac{\lambda n_{L,L} + n_{S,S}}{\lambda n_L + n_S} \tag{7}$$

which considers the total number of errors as $\lambda n_{L,S} + n_{S,L}$. In this case, the total cost ratio (T_{CR}), given by Eq. (8), is used to compare how effective the filter is for a given value of λ in comparison with a baseline setting, where no filter is employed (with, therefore, no false positives and n_S false negatives). Using this index, the use of the filter is considered advantageous if $T_{CR} > 1$:

$$T_{CR} = \frac{n_S}{\lambda n_{L,S} + n_{S,L}} \tag{8}$$

As commented by Cormack and Lynam (2007), the use of the information retrieval indices assumes that there are relatively few relevant instances, and the use of the legitimate precision and recall, which are more intuitive, considering legitimate messages as the relevant class, should be preferred. In regard to the cost-sensitive measures devised byAndroutsopoulos, Koutsias, Chandrinos, Paliouras et al. (2000) and Androutsopoulos, Koutsias, Chandrinos, and Spyropoulos (2000), it should be noted that an ideal value for λ is difficult to be determined. In particular, it depends on the message, as some legitimate messages (e.g., personal messages) are more important than others (Carpinter & Hunt, 2006), and also on the probability that the user notices its incorrect labeling. Another argument against the use of the information retrieval framework is the fact that an user is typically more interested in the relative number of legitimate and Spam messages correctly classified, namely the true negative and positive rates, respectively. While the latter is represented by the Spam recall, the Spam precision does not always allow a direct assessment of the false positive rate. To present this information to the user, it would be necessary to inform the Spam recall and legitimate recall (e.g., Cormack & Lynam, 2007). In this case, it is preferred to adhere to the decision theoretic framework, which offers more general comparison tools, such as ROC curves. In fact, true positive and negative rates are the standard basis for comparison between classifiers on the TREC corpora (see also Cormack & Lynam, 2007). However, care should be taken when summarizing the contents of a ROC curve in the form of the AUC, despite its appealing statistical interpretation, due to the fact that the contribution of the region comprising large false positive rates, which are of little interest in practice, tends to dominate the measure (Metsis, Androutsopoulos, & Paliouras, 2006). In fact, a classifier which has a better performance for low false positive rates can have a lower AUC than another classifier, even though its use is preferred. In this sense, comparing filters based solely on a single index exhibits, in all cases, potential problems.

3. Naive Bayes

The application of the naive Bayes classifier to Spam filtering was initially proposed by Sahami, Dumais, Heckerman, and Horvitz (1998), who considered the problem in a decision theoretic framework given the confidence in the classification of a message. A particularly appealing characteristic of a Bayesian framework is its suitability for integrating evidence from different sources. In this sense, Sahami et al. investigated the use not only of the message words, but also application-specific knowledge, in the form of rules regarding the appearance of certain phrases (e.g., "Free money"), referred to as phrasal features, and non-textual features, obtained through the analysis of the message's header (e.g., the time when the message was sent). Experiments with two private corpora, using binary features selected according to the information gain, indicated the advantage of including the non-textual features, along with the feasibility of the naive Bayes filter, due to its low false positive rates.

In the Bayesian framework, the probability that a given representation of a message, denoted as $\vec{x} = [x_1, x_2, \ldots, x_N]$, belongs to a class $c \in \{s, l\}$ is given by:

$$P(c|\vec{x}) = \frac{P(\vec{x}|c)P(c)}{P(\vec{x})} = \frac{P(\vec{x}|c)P(c)}{P(\vec{x}|s)P(s) + P(\vec{x}|l)P(l)} \tag{9}$$

where $P(\vec{x}|c)$ and $P(c)$ are the probabilities that a message classified as c is represented by $\vec{x}$ and the message belongs to class c, respectively, and $P(\vec{x})$ is the *a priori* probability of a random message represented by $\vec{x}$. The naive classifier is obtained by assuming that the components x_i, $i = 1, 2, \ldots, N$ are conditionally independent, so that $P(\vec{x}|c)$ can be written as:

$$P(\vec{x}|c) = \prod_{i=1}^{N} P(x_i|c) \tag{10}$$

Therefore, Eq. (9) is reduced to:

$$P(c|\vec{x}) = \frac{\prod_{i=1}^{N} P(x_i|c)P(c)}{\prod_{i=1}^{N} P(x_i|s)P(s) + \prod_{i=1}^{N} P(x_i|l)P(l)} \tag{11}$$

with $P(x_i|c), c \in \{s, l\}$, given by:

$$P(x_i|c) = P(X_i = x_i|c) = f(P(t_i|c, \mathscr{D}_{tr}), x_i) \tag{12}$$

where the function f depends on the representation of the message, the probabilistic event model used (cf. (Metsis et al., 2006; Schneider, 2003) and Section 11) and the documents used for training. The probability $P(t_i|c, \mathscr{D}_{tr})$ is determined based on the occurrence of the term t_i in the training data $\mathscr{D}_{tr}$, and also depends on the event model adopted.

The work of Sahami et al. led to the development and application of other machine learning algorithms to Spam filtering, and, in conjunction with the suggestions later proposed by Graham (2002), impelled the development of several practical Spam filtering systems, which commonly employ a Bayesian analysis. In particular, the classifier is very simple and computationally efficient, and attains good correct classification rates in several datasets. Among researchers, it is commonly used for comparison with a newly developed method.

Although Graham's work is usually cited as in the same line as that of Sahami et al., an important difference between these two approaches, termed hereafter Graham and Sahami models, respectively, has not been widely recognized so far in the literature. The basic functioning of these two models is illustrated in Fig. 3. The Sahami model is a straightforward application of the naive Bayes classifier to Spam filtering. It usually requires the application of a feature selection algorithm during training to select the most relevant features, with hundreds or a few thousand features typically

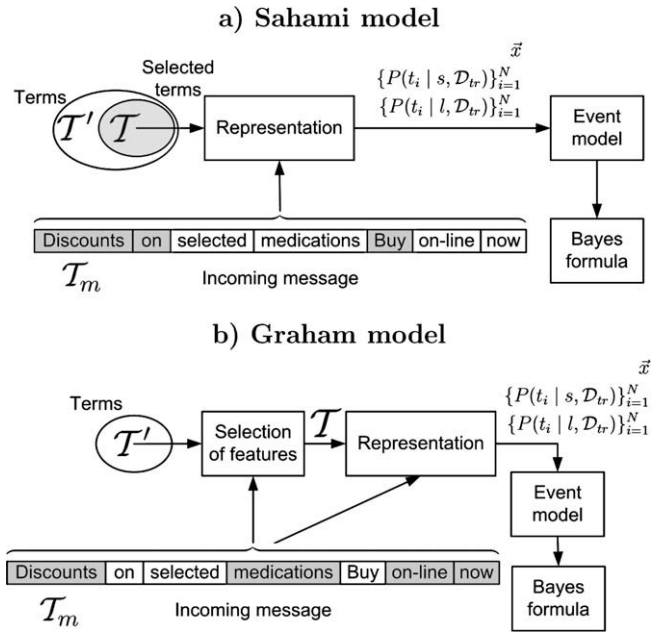

Fig. 3. A comparison between the Sahami and Graham models. The shaded words are those that are actually used for calculating the message's probability, which in some cases can be different in the two models.

leading to the best results. For the classification of an incoming message, a feature vector is generated, based on the occurrence of the selected features in the message, with the probabilities of these features being combined using the event model and Eq. (11). In contrast, Graham's model does not require the application of a feature selection algorithm during training, when only the terms are retained. To classify a message with $n_T = |\mathscr{T}_m|$ words, the first step is selecting the words that will be actually used for determining the message's probability. As proposed by Graham (2002), the N words selected are those with the most "extreme" Spam probabilities (i.e., with the largest absolute differences to 0.5) computed using the data collected during training. Once the words are selected, the remaining steps are as in the Sahami model. Moreover, if a word in the message is not present in the training data, a default probability value, suggested as 0.4 (Graham, 2002), is used. Good results are usually obtained with a small number of features, typically between 10 and 20. In the case that a message contains fewer than N words, all the n_T words are used.

Therefore, Sahami's model performs off-line feature selection, during training, while Graham's model selects the most relevant features online, and the two methods tend to perform similarly in a static setting. On the other hand, Graham's model tends to perform much better in an online fashion, as it is extremely simple for updating, requiring merely that the probabilities of the terms in the message are updated, and can easily consider new features not initially known, while Sahami's model is prone to the limitations imposed by the BoW representation, discussed in Section 2.2. For updating, the latter requires re-applying the feature selection to allow the system to adapt to newly, previously unknown patterns. Moreover, Graham's model tends to be faster for both training, as feature selection is typically the most computationally demanding step, and classification, as a much smaller number of features are used when computing the message's score. However, most of the papers that use the naive Bayes classifier employ Sahami's model. Therefore, considering this key difference between the two models may improve the performance of some algorithms in an online setting, in addition to suggesting the application of new approaches. In fact, a similar parallel can be drawn between

the Case-Based Reasoning models ECUE (Delany, Cunningham, & Coyle, 2005) and SpamHunting (Fdez-Riverola, Iglesias, Díaz, Méndez, & Corchado, 2007a), discussed in Section 7, where the former uses the terms selected according to the information gain, while the latter considers only the most relevant ones (based on their Spam and legitimate probabilities).

Since the work of Sahami et al. (1998), various studies focusing on naive Bayes models were conducted. Medlock (2006) proposed ILM (Interpolated Language Model), which considers the structure of an e-mail message, namely the subject and body, in a smoothed word *n*-gram Bayesian model. The classification of a message relies on aggregating a static (the entire training data) and a dynamic (periodically re-training with new data) component. In experiments using the GenSpam corpus, compiled by the author, ILM attained results superior or at least comparable to that of a SVM based on a *tf-idf* representation, with the use of 2-grams leading to an improved performance in comparison with 1-grams. It also outperformed multinomial naive Bayes and Bayesian logistic regression.

Segal, Markowitz, and Arnold (2006) proposed an approximation of Uncertainty Sampling (US), termed Approximated US (AUS), for building a classifier given a pool of unlabeled messages. In US, for each iteration, the classifier outputs the confidence in the classification of all messages, and then requests the true labels of some of the messages that have the smallest confidence, which are then used to update the model. In AUS, only a subset of the unlabeled messages are re-evaluated in each iteration, speeding up the training process. Experiments were conducted using the TREC2005 corpus and a modified naive Bayes classifier. It was verified that, for more than 5000 iterations, AUS had a similar performance as US, in terms of the error rate, being, in contrast, more than one order of magnitude faster. However, according to the authors, neither method appears currently suitable to handle datasets with 1 million examples or more.

Instead of considering words, Kim, Chung, and Choi (2007) focused on the URLs (links) in messages, using a naive Bayes model. The filter was periodically updated with the messages that were classified and not fed back until a certain time, besides those that were incorrectly classified. Experiments took place with a private corpus and delayed feedback, where most legitimate or Spam messages contained at least one URL. It was concluded that the filter performs similarly to others based on the analysis of URLs or keywords, being, on the other hand, automated, as it does not require the maintenance of black or white lists as in most practical URL-based filters.

In an e-mail server, Segal (2007) studied the performance of a naive Bayes classifier trained using either messages from all the users (a global filter) only with the messages of a certain user (personal filter). In experiments using the TREC2005 and 2006 corpora and a private dataset, the global filters outperformed the personal ones, due to the larger training sets. However, when the users have highly heterogeneous concepts, such as different definitions of Spam, global filters may not offer an acceptable performance, as those essentially average the statistics of each feature from all users. In this case, an approach named Dynamic Personalization (DP) was proposed, which combines, for each feature, personal and global statistics. It was observed that, in a homogeneous scenario, DP led to an improved performance, except for very low false positive rates, and considerable improvements in a heterogeneous setting.

Ciltik and Gungor (2008) applied a naive Bayes classifier based on word *n*-grams, using only some of the first words to reduce the classification time. Two classification models were considered: binary (Spam or legitimate) and instance-based (each message represented as a single class). The latter tends to show better results, but at a higher computational cost. Experiments were conducted using private messages in either English or Turkish and eight-fold cross-validation, reporting only the accuracy in some cases. It was found that a hybrid model, combining binary and instance-based classification led to better results than AdaBoost and a SVM with binary features. Overall, it was concluded that the proposed techniques allow accuracies up to 98% for Turkish and 99% for English messages.

Due to the fact that Spam filtering tends to be computationally demanding, its applicability is often problematic in certain e-mail servers where the volume of circulating messages is high. This has led some researches to consider the development of dedicated hardware for this task. Marsono, El-Kharashi, and Gebali (2008) reported the proposal of a FPGA (field programmable gate array) implementation of Graham's naive Bayes classifier. For 150 and 15 features, it is estimated that the implementation can classify up to 1 and 10 million messages per second, posing as an interesting approach to Spam control.

4. Support Vector Machines (SVM)

Support Vector Machines (SVM) (Scholkopf & Smola, 2002; Vapnik, 1998) were initially applied by Drucker, Wu, and Vapnik (1999), using the BoW representation with binary, frequency or *tf-idf* features, selected according to the information gain, and two private corpora. It was verified that boosting with decision trees achieved a slightly lower false positive rate than SVM, but the latter being more robust to different datasets and pre-processing procedures, and much more for efficient for training (in general, between one and two orders of magnitude). The best results were obtained using a binary representation for SVM and frequency-based for boosting. Finally, as boosting automatically selects the number of features, and SVM are capable of dealing with a large number of features, no feature selection needs to be performed.

In a server-side filter, an important difficulty faced are the different characteristics of Spam and legitimate messages received by different users. In this case, the use of publicly available datasets for training a single classifier for all users severely biases the classification problem. Focusing on this scenario, Bickel and Scheffer (2007) developed a framework for learning a classifier using publicly available (labeled) and user (unlabeled) messages. In the case that n users are currently subscribed, the classifier for a new user is obtained by training a linear SVM, where the misclassification cost of the messages is based on the estimated bias-correction term of each message. The experimental results were obtained using a binary representation, the Enron corpus and Spam messages from various public and private sources. It was verified that the formulation proposed decreased the $(1 - \mathrm{AUC})$ risk in up to 40%, in comparison with the user of a single classifier for all users.

Considering that many Spam and legitimate messages are generated according to a template, Haider, Brefeld, and Scheffer (2007) considered the application of a SVM-based incremental supervised clustering algorithm for identifying, based on the contents, such messages. Experiments were conducted with messages from a private and the Enron corpora, including newsletters. Using a frequency-based linear SVM, in which the batch information was incorporated through four additional features, it was verified that this additional information reduced the $(1 - \mathrm{AUC})$ risk by up to 40%. Moreover, the incremental version proposed required an execution time more than six orders of magnitude lower than that of a batch-based.

Kanaris et al. (2007) used character *n*-grams, when n is prespecified and a variable scheme, to train a linear SVM using a binary or frequency representation, employing the information gain

for feature selection. Experiments were based on the LingSpam and SpamAssassin datasets, 10-fold cross-validation and 3-, 4- and 5-gram models. The n-gram models achieved superior or comparable results to a word-based representation, with variable n being superior in a cost-sensitive scenario.

Sculley and Wachman (2007a) applied linear SVMs in an online setting, using a binary representation and no feature selection. The authors proposed the use of character n-gram models ($n = 3$ and $n = 4$), such that all the possible features are known *a priori*. In this formulation, the number of possible features tends to be large, but the authors pointed out (Sculley & Wachman, 2007b) that the number of unique 4-grams observed was on the order of millions in each of the TREC corpora. To further limit the number of features used, only the first 3000 characters of each message were used. Experiments conducted with the TREC2005 and 2006 corpora showed that the online SVMs obtain very competitive results, especially when $n = 4$, at a cost of a long execution time. The authors proposed a simplified formulation, termed relaxed online SVMs, which allowed between 5- and 10-fold reductions in the execution time, without considerable penalties in the classification performance. It was concluded that online SVMs perform very well, and that the simplifications proposed make their large-scale application feasible.

Sculley (2007) considered an online active learning framework, where a filter classifies a message and can then request its true label for updating the model, the objective being the maximization of the classification performance while minimizing the number of labels requested. Three hyperplane-based classifiers, namely perceptron, perceptron with margins and relaxed online SVM (Sculley & Wachman, 2007a), were studied, along with three methods for deciding to request the label. Two of these methods, *b*-Sampling (*b*S) and Logistic Margin Sampling (LMS), are probabilistic, requesting the label with a certain probability as a function of the distance to the separating hyperplane. In Fixed Margin Sampling (FMS), the label is requested if the distance is smaller than a threshold. Using the same experimental setup as in Sculley and Wachman (2007a), perceptron performed the worst, while SVMs obtained the best results, with LMS and FMS outperforming *b*S. However, when the number of requested labels was large, the three active learning methods performed similarly.

5. Artificial Neural Networks

Clark, Koprinska, and Poon (2003) proposed LINGER, which uses a multi-layer perceptron (Haykin, 1998) for e-mail categorization and Spam filtering. Messages are represented as BoW, with feature selection based on information gain or term-frequency variance. Experiments were conducted using the LingSpam and PU1 corpora, in addition to a private dataset, with 256 features and 10-fold cross-validation. Using the information gain, LINGER obtained perfect results, and outperformed a naive Bayes classifier when using term-frequency variance, but with slightly more false positives. However, when the filter was trained and tested with different datasets, it was verified that the results were unsatisfactory. It was concluded that neural networks can be successfully applied to Spam filtering and e-mail categorization, and that more experiments are needed to evaluate the portability of filters.

Luo and Zincir-Heywood (2005) used a Self-Organizing Map (SOM) (Kohonen, 2001) for sequence analysis (SBSA), which considers the order of words in a given message. The experiments employed the LingSpam corpus with the information gain for feature selection, in a cost-sensitive setting. It was verified that SBSA can properly encode and use the sequence of words in a message to classify it, outperforming the naive Bayes classifier, especially in terms of false positives. However, it should be noted that feature selection does not consider the order of words, which can negatively affect the performance of the network.

Wang, Jones, and Pan (2006) applied two online linear classifiers, Perceptron and Winnow, which rely on the determination of parameters of a separating hyperplane. Besides being fast, adapting the models with new data is simple, although no online evaluation took place in the work. Experiments were conducted using the LingSpam and PU1 corpora, 10-fold cross-validation and considering the F1 measure for performance evaluation. Initially, it was verified that using the information gain or document frequency for feature selection led to similar results, with odds ratio performing the worst. Winnow and Perceptron obtained similar results for a wide range of number of features, with the former slightly outperforming the latter, and a naive Bayes classifier performing the worst, especially for a large number of features. Altogether, the authors pointed out that both algorithms performed very well, being robust to the number of features, iterations and examples used for training. Winnow obtained marginally improved results in comparison with the Perceptron, with both considerably outperforming the naive Bayes classifier.

Tzortzis and Likas (2007) considered the application of deep belief networks, neural networks with several hidden layers, for which an efficient training algorithm was recently developed. Experiments were conducted using the LingSpam and SpamAssassin corpora, and a subset of the Enron corpus, with a frequency-based representation of the 1000 or 1500 features selected based on the information gain. It was verified that the deep belief networks with three hidden layers obtained slightly better results than a SVM using a cosine kernel, with better accuracy in all three corpora.

Instead of analyzing the message contents, Wu (2009) formulated a model considering the transaction for the delivery of each message, as much of this data tends to be faked or omitted in the case of Spam messages. This information is used to train a neural network, aimed at discriminating between behavior typical of Spam and legitimate messages. From a set of six header and four server log file fields, a total of 26 textual features were derived. Experiments were based on a personal corpus, divided by time in five groups. The system attained very low false positive and negative rates, and with better results in comparison to using the same architecture for content-based classification using 3000 features represented as *tf-idf*. Finally, in a comparison with 16 content-based methods, the proposed method attained the best results in terms of both false positive and negative rates.

6. Logistic regression

Goodman and Yih (2006) applied a logistic regression model (e.g., Hastie, Tibshirani, & Friedman, 2001), which is simple and can be easily updated. It uses binary features, distinguished based on whether they occur on the message headers or body, without the application of feature selection. It was verified that, in experiments with the TREC2005 and Enron datasets, in addition to a private corpus, the results obtained were competitive or even superior to some of the best known filters for each corpus.

Zhou (2007) designed an adaptive coding-based strategy, in which the structure of messages is stored in a Huffman tree. It uses one tree for each class, storing the words and their number of occurrences in messages, generating a feature vector encoding these terms in the tree. A logistic regression model is then trained using the features of randomly, preferentially recent, sampled messages. In experiments using the LingSpam corpus, it achieved the best overall results in comparison with several algorithms, including SVM and boosting trees, although the results presented considered only the accuracy. It was concluded that the proposed

approach is very efficient, almost three orders of magnitude faster than SVM, effective and more robust to concept drift and unbalanced data.

Jorgensen, Zhou, and Inge (2008) formulated a multiple instance learning approach for dealing with good word attacks, characterized by the inclusion in a Spam message of words that are typical of legitimate messages. An example is represented by a set of instances, and is classified depending on the classification of each instance, with four methods for generating the instances being proposed. A multiple instance logistic regression (MILR) model is used to combine the classification of each instance. Experiments were conducted using the TREC2006 corpus, chronologically divided into 11 parts, in an online setting, where the filters are trained on one part and used to classify the next. The four methods were compared with a single-instance logistic regression model, a SVM and naive Bayes, with 500 features selected based on the information gain and messages represented using *tf-idf*. It was verified that the attack noticeably degraded the true positive rates of all the filters, which was reduced by up to 50%, except for one of the MILR-based methods. When the classifiers were allowed to train on disguised Spam messages, the effects of the attack were reduced, with MIRL having the best performance. It was concluded that the developed methods are effective, even when training with the disguised messages is not possible.

7. Lazy learning

Sakkis et al. (2003) studied the performance of a lazy learning algorithm (Aha, 1997), the k-NN (nearest neighbors) classifier, on the LingSpam corpus in a cost-sensitive setting, where the similarity between two instances is given by the number of matching binary features. The experiments were conducted using the information gain for feature selection and 10-fold cross-validation. In comparison with naive Bayes, it obtained better results for $\lambda = 1$, with a similar performance otherwise, although naive Bayes only performed well in a reduced range of number of features for $\lambda = 999$.

Delany, Cunningham, and Coyle (2005) proposed ECUE (E-mail Classification Using Examples), a Case-Based Reasoning (CBR) (Aamodt & Plaza, 1994) filter which uses a case editing approach, aimed at decreasing the number of cases in the case-base. It uses some header fields (e.g. Subject, To, From), in addition to the message body, with a binary or frequency representation of the features selected according to the information gain. With a binary representation, the system can be optimized by using a Case Retrieval Net (CRN), which allows the efficient retrieval of similar cases. Using a personal corpus and 50-fold cross-validation, binary features with an edited case-base were the best overall configuration, performing similarly with naive Bayes. In a dynamic setting, where incorrectly classified messages were fed back into the classifier, ECUE achieved better results than naive Bayes, in addition to being much more efficient for updating.

Advancing further in the study of a dynamic scenario, Delany, Cunningham, Tsyambl, and Coyle (2005) focused on concept drift using ECUE. It was verified that updating the classifier with the incorrectly classified messages led to considerable reductions in the false positive rates when using an unedited case-base, and both in the number of false positives and negatives for an edited case-base. Moreover, also re-selecting the features mainly reduced the false negative rates. Altogether, it was argued that these two procedures for updating the classifier allow ECUE to properly handle concept drift.

Afterwards, Delany and Bridge (2006) compared the feature-based binary representation initially employed in ECUE with a feature-free method, in which the distance between two cases is determined based on their compression rates, which operates at the raw message level. However, it cannot be used in conjunction with the Case Retrieval Network. In experiments with private corpora and 10-fold cross-validation, the compression-based method obtained an increased overall accuracy, especially when in terms of false positive, being, however, much slower than the feature-based approach (around two orders of magnitude). Overall, it was concluded that a feature-free method leads to better results, besides not requiring pre-processing of messages. For this reason, it requires fewer actions for dealing with concept drift, as no feature re-selection and re-extraction are needed.

Veloso and Meira (2006) proposed an associative classification method, which employs an efficient algorithm for extracting patterns based on the combination of words and symbols found in the messages used for training, and then uses these patterns for inducing classification rules. In the eager classifier, the rules are generated using patterns from all the available messages, which may lead to the generation of a large number of irrelevant rules, while the lazy classifier uses only the messages with the largest number of common terms. In experiments with the TREC2005 corpus, the lazy approach was able to generate more useful rules, being faster and having a better classification performance. In comparison with open-source filters (such as BogoFilter and SpamAssassin) using the SpamAssassin corpus and 10-fold cross-validation, the lazy classifier proposed achieved the best performance in terms of true positive rates, with true negative rates similar to those of the best filter.

Fdez-Riverola et al. (2007a) developed SpamHunting, a CBR filter with a frequency representation for terms the message header and body. In contrast to ECUE, which uses a BoW representation, only the most relevant terms in each message are retrieved when adding it to the case-base, based on their occurrence in the entire case-base and in each case. To classify a message, its most relevant terms are first obtained, and the network retrieves the cases that have the greatest number of terms in common with the incoming message, using the most similar retrieved cases in a voting scheme. In experiments with the SpamAssassin corpus and 10-fold cross-validation, a SVM with 2000 terms selected according to the information gain attained the lowest false negative rates, while SpamHunting attained the best trade-off between false positives and negatives, in addition to being much faster.

Later on, Fdez-Riverola, Iglesias, Díaz, Méndez, and Corchado (2007b) proposed two approaches for tracking concept drift with SpamHunting: relevant term identification (RTI) and representative message selection (RMS). The former is based on selecting a variable number of most relevant terms, while RMS uses an adaptive sliding window to track changes in the empirical probability distributions of the terms. The evaluation was conducted by updating the model with the classified messages, and considering two dynamic scenarios, in which the concept drift happens gradually, using the SpamAssassin corpus, or abruptly, by training and testing the classifiers with the SpamAssassin and LingSpam corpora, respectively. In the first scenario, SpamHunting (ECUE) achieved lower false positives (negatives) rates. In the case of abrupt concept drift, SpamHunting achieved the highest overall accuracy, with the lowest number of false positives, while naive Bayes obtained the lowest number of false negatives.

Hsiao and Chang (2008) proposed ICBC (Incremental Cluster-Based Classification), an incremental clustering system for dealing with unbalanced datasets and concept drift. The former has a particular effect on feature selection, which tends to favor terms that occur in the dominating class. After clustering, the approach extracts a feature vector containing the *tf-idf* for each cluster, and assigns to a message the class of the nearest cluster. For handling concept drift, the cluster base is adaptively organized, repartitioning a cluster when highly heterogeneous messages are present.

Using the SpamAssassin and EM Canada corpora, a k-NN classifier obtained the best results, with ICBC being much more computationally efficient, while ICBC outperformed k-NN when highly unbalanced datasets were used. Finally, in a situation mimicking concept drift, ICBC quickly adapted once the number of new examples was sufficiently large, outperforming k-NN in terms of the F1-measure.

Mendez, Glez-Pena, Fdez-Riverola, Diaz, and Corchado (2009) proposed another modification to improve the performance of SpamHunting, by removing terms with similar Spam and legitimate probabilities, using an user-specified threshold. The threshold was selected using a personal corpus, although it was reported that the results were very similar when different values were selected. SpamHunting was compared to ECUE using the SpamAssassin corpus, 10-fold cross-validation and a dynamic scenario, where the messages were immediately added to the case-base after classification with the label assigned by the classifier. It was verified that SpamHunting attained a greater accuracy than ECUE, especially in terms of the false positive rate. However, the results appear to be very similar for different values of the threshold, especially when no terms were removed. Altogether, the authors concluded that the proposed technique shows a good accuracy, making of SpamHunting a very reliable approach to Spam filtering.

8. Artificial Immune Systems

Oda and White (2003a) proposed an Artificial Immune System (see, e.g., de Castro & Timmis, 2002) for Spam filtering, where detectors, represented as regular expressions, are used for pattern matching in a message being analyzed. It assigns a weight to each detector, which is incremented (decremented) when it recognizes an expression in a Spam (legitimate) message, with the thresholded sum of the weights of the matching detectors being used to determine the classification of a message. The system can be corrected by either incrementing or decrementing the weights of all the matching detectors. Using a personal corpus and 100 detectors, generated mostly from SpamAssassin heuristics, true positive and negative rates of 90% and 99%, respectively, were obtained. It was concluded that the proposal achieved acceptable results considering the small number of detectors used.

Later on, the same authors (Oda & White, 2003b) compared two other methods for determining the classification of a message, using the SpamAssassin corpus and around 150 detectors. The first method is similar to that previously used, except that the weights are incremented only when patterns in Spam messages are recognized, while the second method weights the number of expressions recognized in Spam and all the messages. It was concluded that the second method is more appropriate, due to its greater overall classification accuracy, despite slightly higher false positive rates.

Oda and White (2005) then performed additional experiments using the SpamAssassin corpus, using also a Bayesian combination of detector weights. In addition to heuristics commonly employed in Spam filters, a dictionary of words, and patterns extracted from a set of messages were considered for generating the detectors. To determine the classification of a message, the second method proposed in Oda and White (2003b) remained as the best alternative, with the Bayes-inspired approach performing slightly worse, although the results were analyzed only in terms of the overall error rate. Finally, it was observed that using the heuristics led, indeed, to the best results, with the other two methods performing similarly.

An antibody network named SRABNET (Supervised Real-Valued Antibody Network) was presented by Bezerra et al. (2006), which evolves a population of detectors. The network size is dynamically adjusted, depending on the training data, using the total cost ratio (TCR) as stopping criterion for training. Messages are represented as BoW with binary features, with a procedure removing words that appear less than 5% and more than 95% in all the messages. Experiments were conducted with the PU1 corpus using 10-fold cross-validation, with weighted accuracies between 97% and 98.5% (for $\lambda = 9$ and $\lambda = 999$), while naive Bayes obtained very low weighted accuracies, according to the authors due to feature selection method used. For both algorithms, however, the TCR was smaller than 1 for $\lambda = 999$, indicating that the two filters may not be recommended for scenarios in which false positives are unacceptable.

Sarafijanovic and Le Boudec (2007) proposed an AIS-based collaborative filter, which attempts to learn signatures of patterns typical of Spam messages, by randomly sampling words from a message and removing those that also occur in legitimate messages. This allows the system to be robust to obfuscation based on random words. It also carefully selects the signatures that will be distributed to other agents, to prevent the use of those relating to unreliable features. In experiments with the SpamAssassin corpus, it was verified that good results can be obtained when relatively few servers collaborate, and that the proposal is robust to obfuscation.

Yue, Abraham, Chi, Hao, and Mo (2007) formulated an incremental clustering model for grouping similar Spam messages, using scores calculated from some features of each message, such as the IP (Internet Protocol) address of the computer that delivered the message, the transaction identification number for the delivery and the presence of links to e-mails or external web pages in the message body. These scores are then fed into an immune network for clustering similar Spam messages. An incremental scheme was proposed, which, when applied to the SpamAssassin corpus, improved the performance of the network, in terms of the homogeneity of the clusters and execution time when new data becomes available. However, no results in terms of the classification of Spam and legitimate messages were reported.

Abi-Haidar and Rocha (2008) designed the Immune Cross-Regulation Model (ICRM), which uses two types of detectors to process randomly sampled features from messages. These features are words and HTML (HyperText Markup Language) tags, with a score being determined for each feature-based on its specific recognition by the detectors, with the final classification given by the thresholded sum of the feature scores. Experiments were conducted with the Enron-Spam corpus in either static or dynamic (sliding window) scenarios, using the accuracy and F1 measure. In the static evaluation, naive Bayes produced the best results, with ICRM and VTT (Variable Trigonometric Threshold, previously developed by the authors for data mining problems) having a similar performance, the former being more robust to unbalanced datasets. When studying the impact of concept drift, ICRM achieved a similar accuracy as naive Bayes, but usually with more false positives.

Guzella, Mota-Santos, Uchoa, and Caminhas (2008) proposed IA-AIS (Innate Adaptive Artificial Immune System), where M detectors recognize the sender's address of Spam messages used for training. Additionally, B and T detectors analyze the subject and body of the message, the former recognizing expressions in the message and the latter determining if those will be activated, with the message being classified as Spam if at least one B detector is activated. Terms are represented in a customized binary encoding, where visually similar characters (for example, *O* and *0*, *E* and *3*) have binary values that differ in only a few bits, with the similarity between a detector and a term given by the number of matching bits, so that the use of these characters tends not to affect the recognition. In experiments conducted with a subset of the Dornbos corpus and personal legitimate messages in an online setting (with random permutations of the messages), IA-AIS correctly classified

up to 99% of Spam or legitimate messages. In comparison with a Graham naive Bayes model, it attained higher true positive rates, with, on the other hand, lower true negative rates.

9. Boosting, ensembles and related approaches

Carreras and Marquez (2001) used an AdaBoost (Hastie et al., 2001) variant, with decision trees as the base classifiers, in experiments with the PU1 corpus, 10-fold cross-validation and binary features. Given a sufficient number of training iterations, AdaBoost outperformed naive Bayes and decision trees, in terms of the F1 measure. When filters with low false positive rates were desired, AdaBoost obtained lower false positive rates, maintaining high true positive rates when base learners of a sufficient tree depth (related to classifier complexity) were used. Altogether, it was concluded that, in this corpus, AdaBoost outperforms Decision Trees and the naive Bayes classifier, having high accuracies and resistance to overfitting.

Zorkadis, Karras, and Panayoto (2005) proposed the use of information theoretical measures, inspired by concepts from communication theory, for integrating classifiers in an ensemble. The procedure is based on training classifiers that maximize either the true positive or true negative rates, and then combining these "extreme" classifiers into a system that maximizes both performance indices. Experiments were conducted with the LingSpam corpus, and selection of 100 or 500 binary or frequency features based on the information gain. The method was compared with voting, in which all the classifiers have equal weights. It was observed that the combination proposed obtained equal or better true positive and negative rates in all cases, with reductions in false positives of up to three times.

He and Thiesson (2007) proposed a boosting approach using asymmetric cost functions, termed Boosting with Different Costs (BDC). This method penalizes all legitimate messages incorrectly classified, while spam messages incorrectly classified with a large margin receive a small weight, as these tend to be outliers, thereby allowing the classifier to ignore these messages. Experiments with a private corpus with more than 200,000 messages, and either one- or two-level decision trees as base learners, were used to compare the method proposed with three logistic regression algorithms and other boosting approaches. It was observed that, for low false positive rates (below 10%), BDC attained the best results, with the use of one-level decision trees having the best performance, and in comparison with logistic regression, produced a much smaller classification model.

Koprinska, Poon, Clark, and Chan (2007) studied the application of random forests, an ensemble of decision trees, to e-mail categorization and Spam filtering, in supervised and semi-supervised (co-training) settings. In the former, the LingSpam and PU1 corpora, with 10-fold cross-validation, were used, selecting 256 features based on either information gain or the proposed term-frequency variance (TFV). Random forests produced the best overall results, with naive Bayes performing the worst, and TFV outperforming the information gain. As in Clark et al. (2003), unacceptable results were obtained in the evaluation of filter portability, especially in terms of the pronounced false positive rates (over 60%). In the semi-supervised scenario, two classifiers were simultaneously trained, with the features being split for each one. The results indicated that co-training is able to improve the performance of the classifiers, with random forests performing the best, and naive Bayes having a largely improved performance, outperforming SVM and decision trees.

Shih, Chiang, and Lin (2008) proposed an architecture for collaborative agents, in which algorithms running in different clients can interact for the classification of messages. The individual methods considered include naive Bayes, Fisher's probability combination method (Robinson, 2003), χ^2 classification, decision trees and neural networks. In the framework developed, the classification given by each method is linearly combined, with the weights of the classifiers that agree (disagree) with the overall result being increased (decreased). In experiments using the UCI corpus, the method achieved better results than the single methods in isolation, with reduced false positives. The authors argued that the proposed framework has important advantages, such as robustness to failure of single methods and easy implementation in a network.

Considering eventual difficulties in collecting legitimate messages due to privacy issues, Wei, Chen, and Cheng (2008) proposed a single-class ensemble-based approach named E2, trained using a set of Spam messages and an unlabeled corpus. It combines two approaches for single-class learning, namely PNB (Positive Naive Bayes, Denis, Gilleron, & Tommasi, 2002) and PEBL (Positive Example-Based Learning, Yu, Han, & Chang, 2004). Both methods work by first generating a model out of the labeled examples, and then selecting appropriate negative examples from the unlabeled set to train a two-class classifier. The approach uses only the negative examples selected by both algorithms, and an ensemble composed of SVM, naive Bayes and decision tree. For evaluation, the LingSpam and PU1 corpora were used along with a frequency representation of features. It was concluded that E2 usually attains better results than PNB and PEBL, being also more robust, and that high Spam precision (up to 99%) can be obtained.

10. Hybrid methods and others

In this section, we focus on Spam filters which integrate different machine learning paradigms. Models relatively unique in terms of their formulation, which cannot be easily classified according to the categories previously discussed, are also discussed.

Aiming at lower false positive rates, Zhao and Zhand (2005) applied Rough Set Theory (RST), a mathematical approach for approximate reasoning, to categorize messages into three classes: Spam, legitimate or suspicious. Features are first selected from the training set, and then, using a genetic algorithm, a set of rules is induced, which divide the universe of objects (messages) into three regions. Based on experiments using only 11 features of the UCI corpus, it was concluded that the proposed approach is effective, capable of reducing the number of legitimate messages that were eventually blocked, being superior to the naive Bayes classifier.

Bratko, Filipic, Cormack, Lynam, and Zupan (2006) proposed data compression models, operating at the raw message level. In considering each class (Spam or legitimate) as an information source, the messages used for training representative of each class are taken as a sample of the data generated by each source. A compression model for each source is then built, which is used to analyze an incoming message. The filters were evaluated in static (using the LingSpam, PU1 and PU3 datasets and 10-fold cross-validation) and online (using the TREC2005, SpamAssassin and a private dataset) settings, and compared to several other approaches. It was verified that the algorithms considered can attain high correct classification rates, superior to many approaches, being capable of correctly classifying 99% of both spam and legitimate messages in some datasets. In addition, the methods are robust to noise, represented by alterations in the spelling of words. However, as discussed by the authors, the memory requirements can be large.

Pampapathi, Mirkin, and Levene (2006) proposed an approach based on a suffix tree (ST), which stores all the suffixes found in a message (for example, for a string "cost", the suffixes are "cost", "ost", "st" and "t"), along with their corresponding frequencies. One tree is built for each class, and the similarity between a message and a tree is based on the set of substrings occurring in both,

with the message being assigned the most similar class. Experiments were conducted with the LingSpam, SpamAssassin and BBK corpora, using 10-fold cross-validation. On the LingSpam corpora, ST achieved much lower false negatives and slightly higher true positives in comparison with naive Bayes. Moreover, ST performed better in the SpamAssassin and BBK corpora, where the number of false positives of naive Bayes increased considerably.

Gordillo and Conde (2007) applied a Hidden Markov Model to the problem of identifying obscured words, assuming the availability of a blacklist containing words associated with Spam messages. Given a list of variants of each of them, one model is trained for each forbidden word. Afterwards, a word extracted from a message could be used as input to each model, determining if it is a variant of the corresponding word (by applying a threshold on the probability returned by the model). It was verified that the method is capable of identifying more than 95% of the variants of a word, but no results in terms of the classification of messages were presented.

Seeking to prevent the waste of network bandwidth for receiving Spam messages, Lam and Yeung (2007) proposed an approach based on social networks to determine if an user attempting to send a message is a Spammer. It builds a network using information on the messages sent and received by each user, from which seven features are extracted, based on the network topology and statistics, which are used in a weighted k-NN classifier to classify a user. The method was evaluated in a custom dataset with legitimate accounts from the Enron corpus, and Spam accounts generated according to previously reported statistics of the behavior of Spammers. It was observed that it was possible to correctly identify more than 99% of the Spammers, with 1% false positives. The authors concluded that, while encouraging, the results reported should be interpreted with caution, as the experimental setting did not consider the frequent changes in the addresses of Spammers, which make their identification more difficult.

Ruan and Tan (2007) combined a SVM and an AIS, using binary features and the same feature selection procedure as in Bezerra et al. (2006). The support vectors obtained after training the SVM are used to generate the initial detector set of the AIS. When the AIS is used for classification, the detector with the smallest Euclidean distance to the message is added to a Committee set, with the classification given as a majority voting of the detectors in this set, with a mutation process being applied to fine-tune these detectors. When the SVM is used for classification, the process is straightforward. To track concept drift, a sliding window of recent batches is used, with one classifier for each batch, and the overall classification being given by voting. Experiments with the PU1 and LingSpam corpora, in an online fashion, indicated that the classification based on SVM leads to considerably superior results compared to that obtained when using the AIS.

Sirisanyalak and Somit (2007) designed a hybrid model in which an AIS-based module extracts features, which are then used in a logistic regression model. An initial set of detectors, represented as regular expressions, is generated using terms extracted from the training messages, with data on the matched detectors being used in the regression model. Using the SpamAssassin corpus, the proposed method was compared with SpamAssassin and the approach developed in Oda and White (2003b). It was verified that the model proposed achieved higher true negative rates in comparison with the two other systems, with SpamAssassin having slightly higher true positive rates.

11. Comparative studies

With the increasing development of Spam filters based on various learning paradigms, and difficulties in comparing filters based solely on performance figures, several works have been dedicated to comparing different filters under the same conditions. These works can provide not only an understanding of the best performing algorithms in certain cases, but also illuminate other aspects of the problem, such as the importance of considering additional message features besides the body, such as the headers.

Schneider (2003) compared two naive Bayes models based on different event models using the LingSpam and PU1 corpora. The multivariate Bernoulli model considers only the occurrence of terms, while the multinomial model uses their frequency. Given a message to be classified, the former considers, in addition to the occurrence of the selected terms, the absence of the remaining terms. In both models, feature selection was conducted based on the information gain. Using 10-fold cross-validation, the multivariate Bernoulli model achieved lower false positives and higher false negatives, with the multinomial approach having a more balanced performance. Overall, it was concluded that the multinomial model is more robust to skewed class distributions, achieving a higher overall accuracy.

Androutsopoulos et al. (2004) compared the performance of naive Bayes, Flexible Bayes, SVMs and boosting on the four PU corpora using 10-fold cross-validation. Messages were encoded using a frequency-based representation, with the information gain for feature selection. In the case of $\lambda = 1$, all algorithms had a similar performance in terms of the weighted accuracy, with boosting and SVM being slower to train but faster when classifying, while the Bayesian classifiers had the opposite behavior. However, naive Bayes had an unacceptable performance for $\lambda = 9$. The authors then presented Filtron, a learning-based filter, which was used by one of the authors for seven months for classifying around 6700 messages, 75% being legitimate. Using a SVM as learning algorithm, trained on the PU3 corpus with $\lambda = 1$, true positive and negative rates of 89% and 99%, respectively, were obtained. The filter was not updated during the whole time, and it was verified that most of the incorrectly classified Spam messages contained either little or no text, were written in languages other than English or were encoded. When it comes to the legitimate messages incorrectly classified, half were newsletters or automatic responses.

Ozgur, Gungor, and Gurgen (2004) investigated the use of Spam filters based on Artificial Neural Networks (ANN) and naive Bayes for Turkish messages, employing a module for morphological analysis. For both algorithms, a standard BoW representation was adopted, using the information gain for feature selection. Experiments were conducted with a personal corpus of Turkish messages and six-fold cross-validation. In a first experiment, it was verified that the frequency-based representation, with 40 or 70 features, produced the best results for the ANN, with a linear network having the best trade-off between effectiveness and execution time. On the other hand, the binary representation produced the best results for the naive Bayes classifier, unless the number of features was too large. It was concluded that the methods constitute an interesting approach to analyzing Turkish messages, with acceptable correct classification rates.

Zhang et al. (2004) compared the performance of five Spam filters using the SpamAssassin, LingSpam and PU1 corpora, in addition to the Chinese corpus ZH1 compiled by the authors. In the experiments, a binary representation, along with 10-fold cross-validation, was used, with the information gain, document frequency and χ^2-test for feature selection and the TCR as performance measure. In regard to the feature selection method, it was verified that the information gain led to the best results, followed by the χ^2-test. Moreover SVM, AdaBoost and logistic regression model attained the best overall results, with naive Bayes and a lazy learning approach not being feasible in a cost-sensitive scenario with $\lambda = 999$. It was also observed that using only the message headers led to superior or at least similar results as using only the message

body, while the combination of both achieved the best performance. The authors concluded that the BoW representation of messages is effective also for classifying Chinese messages and that the good results obtained when considering the message headers highlight the importance of considering this information in a filter.

Webb and Chitti (2005) evaluated four Spam filters, namely (linear) SVM, regression-based boosting and two naive Bayes classifiers (SpamProbe and a standard version using a multivariate Bernoulli model), using a large-scale corpus assembled by the authors with more than 1 million messages. For feature selection, the information gain was used. The message headers were not used, as it was pointed out that using them tends to introduce a substantial bias on the classification, explaining in part the improvements usually observed when they are considered. It was verified that SpamProbe, SVM and boosting performed similarly, followed by naive Bayes, indicating that the filters perform very well. An attack setting was then considered, in which camouflaged Spam messages containing portions of legitimate messages were created, aimed at degrading the performance of the filters. When the classifiers were trained using "normal" Spam and legitimate messages, they are unable to identify many of the camouflaged messages, but when some of these camouflaged messages were used for training, the accuracy of the classifiers was restored. In conclusion, this attack can considerably affect the performance of the filters, but can be handled by using some of these Spam messages for training or updating the classifiers.

Metsis et al. (2006) compared the performance of five naive Bayes classifiers. For binary attributes, a multivariate Bernoulli and a multinomial model were considered. With a frequency-based representation, a multinomial model was used. Moreover, two real-valued models, multivariate Gauss and Flexible Bayes, were considered, which employ a normalized frequency representation. Experiments were conducted with messages from the Enron-Spam corpus, chronologically divided into subsets containing 100 messages, and then used in an online setting, where the classifiers were trained on all the previously seen messages and used to classify the next subset. It was verified that the best results were obtained when using 3000 features, selected on the basis of the information gain, with Flexible Bayes and the binary multinomial model, while the multinomial frequency and multivariate Bernoulli models performed the worst. The authors concluded that, due to its simpler implementation and smoother trade-off between true positive and negative rates, the multinomial binary model appears to be the best variant.

Yih, Goodman, and Hulton (2006) considered the task of training a filter for achieving low false positive rates. A two-stage approach was proposed, based on training two filters, the first one on all the data available and the second one using hard and easy Spam, and hard legitimate messages. For classification, the filter to be used is selected depending on the region in which the test message falls. Experiments were conducted with a large private corpus containing more than 1 million messages, using logistic regression and naive Bayes. The classifiers were trained with either different misclassification costs, the two-stage approach, or a combination of both. It was verified that the combination of both methods reduced the false negative rates by 20% and 40% for logistic regression and naive Bayes, respectively, with the former performing generally better. It was concluded that, to achieve low false positive rates, the two-stage method and/or training with different misclassification costs are more suitable than simply adjusting a classification threshold.

Zorkadis and Karras (2006) proposed the use of the information gain for extracting higher order features, represented by word *n*-grams. The experiments conducted used the same methodology as in Zorkadis et al. (2005), and 2-grams as higher order features, constituting 50% of all the features used. The classifiers compared included naive Bayes, Alternative Decision Trees (ADTree) and random committee machines (using neural networks as base learners), among others. However, the results were presented in terms of the relative number of realizations in which a given classifier obtained the best performance, which precludes their quantitative analysis. It was concluded that the random committee machine and ADTrees have the best true negative and positive rates, respectively, and the authors argued that the use of higher order features can improve the performance of naive Bayes.

Cormack and Lynam (2007) conducted an evaluation of six open-source filters in an online scenario, where incorrectly classified messages were immediately fed back to the filter. The filters considered were SpamAssassin, Bogofilter, CRM114, DSPAM, SpamBayes and SpamProbe. The private corpus used contained around 50,000 messages, with 18% being legitimate, collected during a period of 8 months, with each classifier used to classify the messages without any prior training. It was found that legitimate messages representing advertising, news digest, mailing lists and electronic transactions were the most difficult for correct classification by filters, while few (sometimes none) personal messages were misclassified. For Spam messages, the most difficult ones were those categorized as *backscatter*, which are delivery messages from a mail server rejecting a message forged as having been sent by the user. Altogether, the authors pointed out that the best filters correctly classify all personal legitimate messages, which tend to be the most important messages for most users, and that SpamAssassin, BogoFilter, SpamBayes and SpamProbe had similar results, with DSPAM and CRM114 having the worst performance.

Lai (2007) evaluated the performance of naive Bayes, k-NN and SVM, along with the combination of *tf-idf* and SVM, referred to as an integrated approach. The use of the message's subject, body, header or all of those was studied. Experiments were conducted using the SpamAssassin combined either with a personal dataset or with the EM Canada corpus. It was verified that the best overall method was SVM using all data of a message, with the naive Bayes performing well except when analyzing only the message's body, and the k-NN classifier obtaining generally unsatisfactory results. However, the results were presented solely in terms of the accuracies. In addition, the integrated approach achieved similar results to the SVM.

Seewald (2007) evaluated the performance of a simple naive Bayes implementation (SpamBayes), along with CRM114 and SpamAssassin, which also employ more sophisticated language models and hard-coded rules, respectively. For the initial experiments, seven private mailboxes were used. It was verified that all filters obtained similar results, with SpamBayes having slightly less false positives and more false negatives. CRM114 and SpamAssassin were, however, sensitive to outliers, when incorrectly labeled messages were used for training, due to the fact that the procedure used in the paper for training can lead to overfitting. To evaluate how the algorithms deal with concept drift, an additional mailbox, derived from a different user, was used for testing the classifiers. It was verified that the performance of CRM114 and SpamAssassin was considerably degraded, especially the former, indicating it would be necessary to re-train the classifiers at some point, while SpamBayes remained relatively stable. The author concluded that the simplest system is preferred, as the more sophisticated models do not appear to lead to improved results, not responding well to outliers and concept drift.

Sculley and Cormack (2008) evaluated the performance of several filters when incorrectly labeled messages were used for updating, as several reports highlight that the number of such messages tends to be large in real-world systems. When the true label of messages was randomly altered, independently of the message contents (synthetic noise), it was verified that, using the TREC2006 and 2007 corpora, adjusting certain parameters used for training

could restore the good performance of the filters. However, when natural noise was considered, where the labels of certain types of messages that tend to have divergent classifications by different users were altered, the performance was again degraded, indicating that this scenario is harder and requires further studies. In conclusion, the authors proposed that filters should be evaluated in this noisy setting, preferably originating from human users rather than random.

Yu and Ben Xu (2008) compared the performance of four algorithms: neural networks, naive Bayes, SVM and Relevance Vector Machine (RVM) (Tipping, 2001). In comparison with SVM, the latter does not require the adjustment of parameters and results in an usually more sparse solution, thereby resulting in faster testing. However, it involves the solution of a non-convex optimization problem, tending to be slower for training. For the neural network, binary features were used as input, while, in the SVM, a sigmoid kernel was used, although it should be noted that, in this setting, the optimization problem may become non-convex, due to the fact that this kernel is not positive definite for certain parameter values (e.g. Scholkopf & Smola, 2002, p. 45). The experimental evaluation was conducted using the SpamAssassin corpus and a private dataset, with features selected using LINGER (Clark et al., 2003). It was verified that SVM and RVM have a similar performance, with the choice between them depending on the need for faster training (SVM) or testing (RVM), followed by naive Bayes, and with the neural network performing the worst, although only the accuracies were presented.

12. Image analysis

Spam filters need not concentrate only on the textual content of messages. Given the increasing numbers of Spam messages containing images, which usually contain no text on the body or subject, or only random words aimed at biasing the classification of the message, some researchers have considered how to detect these messages based on image analysis. These approaches are discussed in this section.

Aradhye, Myers, and Herson (2005) designed a method for identifying images typical of Spam messages, based on the extraction of image regions containing text. It tends to be faster than using OCR (Optical Character Recognition), as it does not require the actual identification of the text. Five features, such as the relative portion of the message containing text, chosen to facilitate the discrimination between images typical of Spam messages and images commonly sent in legitimate messages (such as photographs), are fed to a polynomial kernel SVM, which classifies each image. The results obtained, using a personal dataset of images, indicated that the method is capable of correctly identifying between 70% and 80% of Spam-related images, and between 70% and 100% of legitimate images.

Wu, Cheng, Zhu, and Wu (2005) developed an approach for analyzing visual features of images attached to e-mails. For each message, a number of features from all the images are extracted and fed into a classifier for determining the class of the message. These features include the number of regions containing embedded text, the relative number of images containing such text, among others. The authors argued that, due to difficulties in collecting legitimate messages containing attached images, the problem should be tackled using a single-class algorithm, trained using only Spam messages. Using artificially generated legitimate messages for testing, a one-class SVM achieved reduced false positives in comparison with a two-class SVM, although the number of false negatives was increased.

Fumera, Pillai, and Roli (2006) developed a module for performing a semantic analysis of the text embedded in attached images. It was argued that this analysis is important, as the rules based on visual features as in Aradhye et al. (2005) and Wu et al. (2005) tend to have low generalization capabilities. They employed an OCR software for extracting the embedded text and then performed analysis of the contents using a linear SVM, with features selected according to the information gain. The Enron dataset, incremented with Spam messages collected by the authors, was used, and also a personal corpus, with around 10% and 5% of all the Spam messages used for testing containing image attachments, respectively. Further, legitimate messages without images were obtained from a subset of the Enron corpus. Using the personal corpus, for low false positive rates, considering also the image text led to a lower number of false negatives, with a decrease in false negative rates of up to 50% over only the Spam messages with attachments. With the SpamAssassin corpus, for low false positive rates (1%), using only the text from images produced the lowest false negative rates. Altogether, it was concluded that considering the text from images can lead to improved results.

Biggio, Fumera, Pillai, and Roli (2007) proposed two measures for detecting content obscuring, which is aimed at degrading the performance of OCR-based filters. In order to experimentally evaluate the performance of these measures, around 200 Spam messages containing image attachments were used, with nearly 50% employing some technique for content obscuring. It was verified that the two measures allow a good discrimination between images with different levels of degradation, although some overlapping occurs. However, the authors pointed out that some images cannot be identified on the basis of these measures.

Byun, Lee, Webb, and Pu (2007) considered the analysis of certain features, designed to highlight the differences between images typical of Spam or legitimate messages. The derived features are fed into a linear classifier, which determines the score of a given image relative to each class. The classifier is trained to minimize the number of training errors, and a test image is assigned to the class with the highest score. The authors argued that, due to the inherent heterogeneities among images in each class, it was interesting to train a classifier for sub-groups of each class, which contain images sharing some high-level features. Experiments conducted with a private collection of Spam and legitimate images, along with images obtained from the TREC2005 corpus indicated that the sub-group approach consistently outperformed the single-group counterpart, especially in terms of false positives.

Dredze, Gevaryahu, and Elias-Bachrach (2007) focused on fast algorithms for labeling images, considering that such classification of images attached in a message can be combined by a filter with text-related features. Several easily obtained features, which do not require checking the image contents, were defined. Experiments were conducted with the Dredze and a private corpora, using logistic regression, naive Bayes and decision tree, evaluated based on the accuracy and F1 measure. It was verified that logistic regression attained the best results, followed by decision trees and naive Bayes. The selection of 500 features based on the information gain led to a 20% reduction in the processing time, with slightly lower accuracy, while the selection based on the combination of information gain and processing time reduced the processing time by almost three orders of magnitude. An online method was also proposed, which selects the features based on their effect on the classification of the image, leading to a further two-fold speedup. The authors concluded that this highly efficient approach makes the analysis of images by Spam filters feasible.

Krasser, Tang, Gould, Alperovitch, and Judge (2007) developed a fast method, that does not require decoding an image, for identifying those typical of Spam messages, based on image header features fed into a suitable classifier. For a complete evaluation, a total of seven features were employed, using a private corpus and 10-fold cross-validation, with decision trees and SVM as classifiers. It was

verified that the latter achieved the best performance in terms of the AUC. The SVM-based fast classification model was then compared to a more computationally demanding model (around 200 times slower) using a SVM, where features are derived from the spatial and frequency domain characteristics of the decoded image. It was found that the latter can attain much higher true positive rates, although at a computational cost that was not interesting in the scope of the work. It was concluded that 60% of the Spam images can be detected, with 0.5% false positives, and that the low-cost system can be used as a first level in a multi-module filter.

As Spam images are usually sent in batches, generated from a single template and then subjected to random small distortions to prevent recognition by signature-based filters, Wang, Josephson, Lv, Charikar, and Li (2007) tackled the problem of detecting these distorted messages. For this, it is assumed that some images have been classified as Spam by other techniques, so that the problem reduces to detecting variations of known images. This is based on storing feature vectors extracted from the known images, and then checking the similarity with the feature vectors derived from an incoming message. Three different filters, tailored to specific features sensitive to certain types of distortions, were proposed, in addition to three methods for aggregating the outputs of each filter. Experiments conducted using the Princeton Image Spam dataset, compiled by the authors, indicated the high performance of the filters, with aggregation based on voting having the best compromise between false positives and negatives. The approach is fast and storage-efficient, so that the distribution of feature vectors between different servers, in a collaborative setting, is possible.

Biggio, Fumera, Pillai, and Roli (2008) compared the obfuscation-sensitive features proposed in Biggio et al. (2007) with the methods proposed in Aradhye et al. (2005) and Dredze et al. (2007), along with four "generic" features (relying on low-level image characteristics). The experiments were based on the Dredze and Biggio corpora, using five-fold cross-validation and ROC analysis. It was verified that the features proposed in Aradhye et al. (2005) and Dredze et al. (2007) along with the generic ones, led to better results than the obfuscation-sensitive, as the latter tends be reactive to noise typical of legitimate images. Furthermore, the combination of the obfuscation-sensitive features with any of the others, either at the feature level (where a feature vector is formed by concatenating data from the two different methods) or at the score level (where one classifier is trained for each method, and the scores are combined using a linear SVM) arises as a new possibility, leading to improved results in most of the cases.

13. Discussion and conclusions

In this paper, a comprehensive review of recent machine learning approaches to Spam filters was presented. A quantitative analysis of the use of feature selection algorithms and datasets was conducted. It was verified that the information gain is the most commonly used method for feature selection, although it has been suggested that others (e.g., the term-frequency variance, in Koprinska et al. (2007)) may lead to improved results when used with certain machine learning algorithms. Among the several publicly available datasets, the LingSpam and SpamAssassin corpora stand as the most popular, while the recent TREC corpora, which attempt to reproduce a realistic, online, setting, are moderately popular at present. In terms of evaluation measures, the true positive and negative rates, which are given, respectively, by the relative number of Spam and legitimate messages correctly classified, are suggested as the preferred indices for evaluating filters, especially in the form of ROC curves (Fawcett, 2006).

Two important aspects not widely recognized in the literature were discussed. Although most algorithms represent messages as bag-of-words (BoW), it should be carefully used, as it imposes a severe bias in the problem. This is due to that fact that updating a model to consider new terms, which were not initially available, can be a weak point, as it usually requires re-building the classifier from scratch. This is also necessary when features that become noninformative need to be discarded. An interesting line of research would be to design methods to allow the incremental addition or removal of features, without re-building the entire model, in a similar way as updating using new information on the occurrence of known terms is performed. Another option would be the approach used by Sculley and Wachman (2007a), where all possible terms are known *a priori*, although the resulting large dimensionalities may impose limitations in the types of algorithms that can be used. An important difference between the naive Bayes models proposed by Sahami et al. (1998) and Graham (2002) was also highlighted. While the former adopts a BoW representation, relying on *a priori* feature selection, the latter selects the features that will be used for classification in an online fashion, depending on the current statistics of the terms present in the message. This online selection makes updating the filter with new terms much easier, in addition to being faster, as the number of features actually used tends to be much lower.

It is interesting to note that the bias imposed by the BoW representation and the difference between the two naive Bayes models are relevant in an online setting, in which periodically updating the model is necessary, as discussed in Section 2. The importance of model update is highlighted by works studying the performance of filters (Jorgensen et al., 2008; Webb & Chitti, 2005) when dealing with disguised messages, where training using these messages improved the performance of the filters. The fact that many recent works conduct experiments in a static scenario is likely to be the explanation for not considering these two aspects. However, in order to have an adequate assessment of the performance of filters, it is necessary to adopt more realistic evaluation settings (e.g., the TREC corpora, Cormack, 2006, 2007; Cormack & Lynam, 2005), that better mimic the scenario faced by a filter deployed for practical operation. In particular, the argument raised by Cormack and Lynam (2007), and further reinforced by Seewald (2007), regarding the still unproven potential of more advanced machine learning algorithms to Spam filtering, can be associated to the evaluation scenarios considered. More then simply affecting the experimental results obtained when reporting the development of a new filter, this may inspire the development of customized filters, tailored to the characteristics of the problem (cf. Fawcett, 2003). In addition, the recent results reported by Sculley and Cormack (2008) regarding the sensitivity of most filters to noisy feedback indicate that the design and evaluation of new filters should also consider the possibility of unreliable feedback. Finally, the advantages of using message headers remains uncertain when evaluating filters, with some authors pointing out that this additional information is beneficial (Lai, 2007; Zhang et al., 2004), while others argue that its use may lead to biased results (Webb & Chitti, 2005).

When it comes to approaches for dealing with image Spam, an important distinction can be made between them. While the use of Optical Character Recognition (e.g., Fumera et al., 2006) to allow text-based algorithms to analyze the embedded text can be an effective way to handle these messages, the typically high computational costs, in addition to the increasing use of image obfuscation techniques, may limit its applicability. In this sense, methods based on extracting relatively simple image features (e.g., Aradhye et al., 2005; Biggio et al., 2007; Wu et al., 2005), with varying degrees of complexity and discrimination capabilities, can have positive contributions. Therefore, for dealing with image Spam, a multi-stage system, combining algorithms based on these two approaches and a careful selection of the methods to be applied in each step, in terms of the desired confidence and process-

ing time, as in Dredze et al. (2007), is likely to be the best approach. This is can be easily performed, due to the modular design of most filters (Fumera et al., 2006). Another aspect to be considered is how, given a method for analyzing single images (e.g., Aradhye et al., 2005; Biggio et al., 2007; Byun et al., 2007), determine the classification of a message containing text and several images (cf. Dredze et al., 2007; Wu et al., 2005).

In summary, we conclude that while important advancements have been made in the last years, several aspects remain to be explored, especially in more realistic settings. Using an immunological metaphor, as in Hayes (2007), it is expected that more reliable systems can be obtained by combining several algorithms with different characteristics, in the same way as the immune system has several mechanisms for dealing with invading pathogens. For this reason, and the likely emergence of new techniques for disguising Spam messages, Spam filtering is likely to remain an active area of research in the next years.

Acknowledgements

This work was supported by grants from UOL, through its Bolsa Pesquisa program (process number 20060519110414a), FAPEMIG and CNPq.

References

Aamodt, A., & Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *Artificial Intelligence Communication, 7*(1), 39–59.

Abi-Haidar, A., & Rocha, L. M. (2008). Adaptive spam detection inspired by a cross-regulation model of immune dynamics: A study of concept drift. *Lecture Notes in Computer Science*, 5132.

Aha, D. W. (1997). Lazy learning. *Artificial Intelligence Review, 11*(1–5), 7–10.

Androutsopoulos, I., Koutsias, J., Chandrinos, K. V., Paliouras, G., & Spyropoulos, C. (2000). An evaluation of naive Bayesian anti-spam filtering. In G. Potamias, V. Moustakis, & M. van Someren, M. (Eds.), *Proc of the 11th Eur conf on mach learn*.

Androutsopoulos, I., Koutsias, J., Chandrinos, K. V., & Spyropoulos, C. D. (2000). An experimental comparison of naïve Bayesian and keyword-based anti-spam filtering with personal e-mail messages. In *Proc of the ann int ACM SIGIR conf on res and devel in inform retrieval*.

Androutsopoulos, I., Paliouras, G., & Michelakis, E. (2004). *Learning to filter unsolicited commercial e-mail*. Tech. rep. 2004/2, NCSR "Demokritos".

Aradhye, H., Myers, G., & Herson, J. (2005). Image analysis for efficient categorization of image-based spam e-mail. In *Proc int conf doc analysis and recog* (Vol. 2).

Asuncion, A., & Newman, D. (2007). *UCI machine learning repository*. <http://www.ics.uci.edu/mlearn/MLRepository.html>.

Bekkerman, R. (2005). *Email classification on enron dataset*. <http://www.cs.umass.edu/ronb/enron_dataset.html> (visited on June 2008).

Bezerra, G. B., Barra, T. V., Ferreira, H. M., Knidel, H., de Castro, L. N., & Zuben, F. J. V. (2006). An immunological filter for spam. *Lecture Notes in Computer Science, 4163*, 446–458.

Bickel, S., & Scheffer, T. (2007). Dirichlet-enhanced spam filtering based on biased samples. *Advances in Neural Information Processing System, 19*, 161–168.

Biggio, B., Fumera, G., Pillai, I., & Roli, F. (2007). Image spam filtering using visual information. In *Proc int conf on image analysis and proc*.

Biggio, B., Fumera, G., Pillai, I., & Roli, F. (2008). Improving image spam filtering using image text features. In *Proc of the fifth conf on email and anti-spam*.

Blanzieri, E., & Bryl, A. (2008). *A survey of learning-based techniques of email spam filtering*. Tech. rep. DIT-06-056, University of Trento, Information Engineering and Computer Science Department.

Bratko, A., Filipic, B., Cormack, G. V., Lynam, T. R., & Zupan, B. (2006). Spam filtering using statistical data compression models. *Journal of Machine Learning Research, 7*, 2673–2698.

Byun, B., Lee, C.-H., Webb, S., & Pu, C. (2007). A discriminative classifier learning approach to image modeling and spam image identification. In *Proc of the fourth conf on email and anti-spam*.

Camastra, F., & Verri, A. (2005). A novel kernel method for clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 27*(5), 801–805.

Carpinter, J., & Hunt, R. (2006). Tightening the net: A review of current and next generation spam filtering tools. *Computers and Security, 25*(8), 566–578.

Carreras, X., & Marquez, L. (2001). Boosting trees for anti-spam email filtering. In *Proc of the fourth int conf on recent adv in nat lang proc*.

Ciltik, A., & Gungor, T. (2008). Time-efficient spam e-mail filtering using *n*-gram models. *Pattern Recognition Letters, 29*(1), 19–33.

Clark, J., Koprinska, I., & Poon, J. (2003). A neural network based approach to automated e-mail classification. In *Proc of the IEEE/WIC int conf on web intell*.

Cormack, G. V. (2006). TREC 2006 spam track overview. In *Proc of TREC 2006: The 15th text retrieval conf*.

Cormack, G. V. (2007). TREC 2007 spam track overview. In *Proc of TREC 2007: The 16th text retrieval conf*.

Cormack, G. V., & Lynam, T. (2005). TREC 2005 spam track overview. In: *Proc of TREC 2005: The 14th text retrieval conf*.

Cormack, G. V., & Lynam, T. R. (2007). Online supervised spam filter evaluation. *ACM Transactions on Information Systems, 25*(3), 11.

Culotta, A., Kristjansson, T., McCallum, A., & Viola, P. (2006). Corrective feedback and persistent learning for information extraction. *Artificial Intelligence, 170*(14–15), 1101–1122.

de Castro, L. N., & Timmis, J. (2002). *Artificial immune systems: A new computational intelligence approach* (1st ed.). Springer.

Delany, S. J., & Bridge, D. (2006). Textual case-based reasoning for spam filtering: A comparison of feature-based and feature-free approaches. *Artificial Intelligence Review, 26*(1–2), 75–87.

Delany, S. J., Cunningham, P., & Coyle, L. (2005). An assessment of case-based reasoning for spam filtering. *Artificial Intelligence Review, 24*(3–4), 359–378.

Delany, S. J., Cunningham, P., Tsymbal, A., & Coyle, L. (2005). A case-based technique for tracking concept drift in spam filtering. *Knowledge-Based Systems, 18*(4–5), 187–195.

Denis, F., Gilleron, R., & Tommasi, M. (2002). Text classification from positive and unlabeled examples. In *Proc of the int conf on inform proc and manag of uncertainty in knowl-based syst*.

Dornbos, J. (2002). *Spam: What can you do about it?* <http://www.dornbos.com/spam01.shtml> (visited on June 2008).

Dredze, M., Gevaryahu, R., & Elias-Bachrach, A. (2007). Learning fast classifiers for image spam. In *Proc of the fourth conf on email and anti-spam*.

Drucker, H., Wu, D., & Vapnik, V. N. (1999). Support vector machines for spam categorization. *IEEE Transactions on Neural Networks, 10*(5), 1048–1054.

Fawcett, T. (2003). "In vivo" spam filtering: A challenge problem for KDD. *SIGKDD Explorations, 5*(2), 140–148.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters, 27*(8), 861–874.

Fdez-Riverola, F., Iglesias, E., Díaz, F., Méndez, J. R., & Corchado, J. M. (2007a). SpamHunting: An instance-based reasoning system for spam labelling and filtering. *Decision Support Systems, 43*(3), 722–736.

Fdez-Riverola, F., Iglesias, E. L., Díaz, F., Méndez, J. R., & Corchado, J. M. (2007b). Applying lazy learning algorithms to tackle concept drift in spam filtering. *Expert Systems with Applications, 33*(3), 36–48.

Fumera, G., Pillai, I., & Roli, F. (2006). Spam filtering based on the analysis of text information embedded into images. *Journal of Machine Learning Research, 7*, 2699–2720.

Gabrilovich, E., & Markovitch, S. (2007). Harnessing the expertise of 70,000 human editors: Knowledge-based feature generation for text categorization. *Journal of Machine Learning Research, 8*, 2297–2345.

Gadat, S., & Younes, L. (2007). A stochastic algorithm for feature selection in pattern recognition. *Journal of Machine Learning Research, 8*, 509–547.

Gomes, L. H., Cazita, C., Almeida, J. M., Almeida, V., & Meira, W. Jr., (2007). Workload models of spam and legitimate e-mails. *Performance Evaluation, 64*(7–8), 690–714.

Goodman, J., & Yih, W. (2006). Online discriminative spam filter training. In *Proc of the third conf on email and anti-spam*.

Goodman, J., Cormack, G. V., & Heckerman, D. (2007). Spam and the ongoing battle for the inbox. *Communications of the ACM, 50*(2), 24–33.

Gordillo, J., & Conde, E. (2007). An HMM for detecting spam mail. *Expert Systems with Applications, 33*(3), 667–682.

Graham, P. (2002). *A plan for spam*. <http://www.paulgraham.com/spam.html> (visited on April 2008).

Guenter, B. (2008). SPAM archive. <http://untroubled.org/spam/> (visited on June 2008).

Guzella, T. S., Mota-Santos, T. A., Uchoa, J. Q., & Caminhas, W. M. (2008). Identification of spam messages using an approach inspired on the immune system. *Biosystems, 92*(3), 215–225.

Gyongyi, Z., & Garcia-Molina, H. (2005). Web spam taxonomy. In *Proceedings of the first international workshop on adversarial information retrieval on the web (AIRWeb)*.

Haider, P., Brefeld, U., & Scheffer, T. (2007). Supervised clustering of streaming data for email batch detection. In *Proc of the int conf on mach learn*.

Hastie, T., Tibshirani, R., & Friedman, J. (2001). *The elements of statistical learning*. Springer.

Hayes, B. (2007). How many ways can you spell V1@gra? *Scientific American, 95*(4), 298–302.

Haykin, S. (1998). *Neural networks: A comprehensive foundation* (2nd ed.). Prentice Hall.

He, J., & Thiesson, B. (2007). Asymmetric gradient boosting with application to spam filtering. In *Proc of the fourth conf on email and anti-spam*.

Hoanca, B. (2006). How good are our weapons in the spam wars? *IEEE Technology and Society Magazine, 25*(1), 22–30.

Hsiao, W.-F., & Chang, T.-M. (2008). An incremental cluster-based approach to spam filtering. *Expert Systems with Applications, 34*(3), 1599–1608.

Jones, R. (2003). *Spam*. <http://www.annexia.org/spam> (visited on July 2008).

Jorgensen, Z., Zhou, Y., & Inge, M. (2008). A multiple instance learning strategy for combating good word attacks on spam filters. *Journal of Machine Learning Research, 8*, 993–1019.

Kanaris, I., Kanaris, K., Houvardas, I., & Stamatatos, E. (2007). Words versus character *N*-grams for anti-spam filtering. *International Journal of Artificial Intelligence Tools, 16*(6), 1047–1067.

Kim, J., Chung, K., & Choi, K. (2007). Spam filtering with dynamically updated URL statistics. *IEEE Security and Privacy, 5*(4), 33–39.

Kohonen, T. (2001). *Self-organizing maps* (3rd ed.). Springer.

Kolari, P., Java, A., Finin, T., Oates, T., & Joshi, A. (2006). Detecting spam blogs: A machine learning approach. In *Proc of the 21st nat conf on artif intell*.

Koprinska, I., Poon, J., Clark, J., & Chan, J. (2007). Learning to classify e-mail. *Information Sciences, 177*(10), 2167–2187.

Krasser, S., Tang, Y., Gould, J., Alperovitch, D., & Judge, P. (2007). Identifying image spam based on header and file properties using C4.5 decision trees and support vector machine learning. In *IEEE SMC inf assur and sec workshop*.

Lai, C.-C. (2007). An empirical study of three machine learning methods for spam filtering. *Knowledge-Based Systems, 20*(3), 249–254.

Lam, H.-Y., & Yeung, D.-Y. (2007). A learning approach to spam detection based on social networks. In *Proc of the fourth conf on email and anti-spam*.

Leopold, E., & Kindermann, J. (2002). Text categorization with support vector machines. How to represent texts in input space? *Machine Learning, 46*(1–3), 423–444.

Luo, X., & Zincir-Heywood, N. (2005). Comparison of a SOM based sequence analysis system and naive Bayesian classifier for spam filtering. In *Proc of the int conf on neural networks* (Vol. 4).

Marsono, M., El-Kharashi, M. W., & Gebali, F. (2008). Binary LNS-based naïve Bayes inference engine for spam control: Noise analysis and FPGA implementation. *IET Computers and Digitial Techniques, 2*(1), 56–62.

Martin-Herran, G., Rubel, O., & Zaccour, G. (2008). Competing for consumer's attention. *Automatica, 44*(2), 361–370.

Medlock, B. (2006). An adaptive, semi-structured language model approach to spam filtering on a new corpus. In *Proc of the third conf on email and anti-spam*.

Mendez, J. R., Glez-Pena, D., Fdez-Riverola, F., Diaz, F., & Corchado, J. (2009). Managing irrelevant knowledge in CBR models for unsolicited e-mail classification. *Expert Systems with Applications, 36*(2), 1601–1614.

Metsis, V., Androutsopoulos, I., & Paliouras, G. (2006). Spam filtering with naive Bayes – Which naive Bayes? In *Proc conf on email and anti-spam*.

Mitchell, T. M. (1997). *Machine learning* (1st ed.). McGraw-Hill.

Oda, T., & White, T. (2003b). Increasing the accuracy of a spam-detecting artificial immune system. In *Proc of the IEEE cong on evol comput* (Vol. 1).

Oda, T., & White, T. (2003a). Developing an immunity to spam. *Lecture Notes in Computer Science, 2723*, 231–242.

Oda, T., & White, T. (2005). Immunity from spam: An analysis of an artificial immune system for junk email detection. *Lecture Notes in Computer Science, 3627*, 276–289.

Ozgur, L., Gungor, T., & Gurgen, F. (2004). Adaptive anti-spam filtering for agglutinative languages: A special case for Turkish. *Pattern Recognition Letters, 25*(16), 1819–1831.

Pampapathi, R., Mirkin, B., & Levene, M. (2006). A suffix tree approach to anti-spam email filtering. *Machine Learning, 65*(1), 309–338.

Pu, C., & Webb, S. (2006). Observed trends in spam construction techniques: A case study of spam evolution. In *Proc of the third conf on email and anti-spam*.

Qin, Y., & Zhang, S. (2008). Empirical likelihood confidence intervals for differences between two datasets with missing data. *Pattern Recognition Letters, 29*(6), 803–812.

Robinson, G. (2003). A statistical approach to the spam problem. *Linux Journal, 107*, 6467.

Ruan, G., & Tan, Y. (2007). Intelligent detection approaches for spam. In *Proc Int Conf on Nat Comput* (Vol. 3).

Sahami, M., Dumais, S., Heckerman, D., & Horvitz, E. (1998). *A Bayesian approach to filtering junk E-mail*. Tech. rep. WS-98-05. AAI Press.

Sakkis, G., Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Spyropoulos, C. D., & Stamatopoulos, P. (2003). A memory-based approach to anti-spam filtering for mailing lists. *Information Retrieval, 6*(1), 49–73.

Sarafijanovic, S., & Le Boudec, J.-Y. (2007). *Artificial immune system for collaborative spam filtering*. Tech. rep. LCA-REPORT-2007-008, Ecole Polytechnique Federale de Lausanne.

Schneider, K.-M. (2003). A comparison of event models for naive Bayes anti-spam e-mail filtering. In *Proc of the 10th conf of the Eur chapter of the assoc for comput ling*.

Scholkopf, B., & Smola, A. J. (2002). *Learning with kernels* (1st ed.). MIT Press.

Sculley, D. (2007). Online active learning methods for fast label efficient spam filtering. In *Proc of CEAS*.

Sculley, D., & Cormack, G. V. (2008). Filtering email spam in the presence of noisy user feedback. In *Proc of the fifth conf on email and anti-spam*.

Sculley, D., & Wachman, G. M. (2007a). Relaxed online SVMs for spam filtering. In *Proc of the ann int ACM SIGIR conf on res and devel in inform retrieval*.

Sculley, D., & Wachman, G. M. (2007b). Relaxed online SVMs in the TREC spam filtering track. In *Proc of TREC 2007: The 16th text retrieval conf*.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys, 34*(1), 1–47.

Seewald, A. K. (2007). An evaluation of naive Bayes variants in content-based learning for spam filtering. *Intelligent Data Analysis, 11*(5), 497–524.

Segal, R. (2007). Combining global and personal anti-spam filtering. In *Proc of the fourth conf on email and anti-spam*.

Segal, R., Markowitz, T., & Arnold, W. (2006). Fast uncertainty sampling for labeling large e-mail corpora. In *Proc of the third conf on email and anti-spam*.

Shih, D.-H., Chiang, H.-S., & Lin, B. (2008). Collaborative spam filtering with heterogeneous agents. *Expert Systems with Applications, 34*(4), 1555–1566.

Sirisanyalak, B., & Somit, O. (2007). An artificial immunity-based spam detection system. In *IEEE cong on evol comput*.

Song, F., Zhang, D., Xu, Y., & Wang, J. (2007). Five new feature selection metrics in text categorization. *International Journal of Pattern Recognition, 21*(6), 1085–1101.

SpamAssassin. (2005). *Spamassassin public corpus*. <http://spamassassin.apache.org/publiccorpus/> (visited on June 2008).

Stern, H. (2008). A survey of modern spam tools. In *Proc of the fifth conf on email and anti-spam*.

Talbot, D. (2008). Where SPAM is born. *Technology Review, 111*(3), 28.

Tipping, M. E. (2001). Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research, 1*, 211–244.

Tzortzis, G., & Likas, A. (2007). Deep belief networks for spam filtering. In *Proc of the IEEE int conf on tools with art intel* (Vol. 2).

Vapnik, V. N. (1998). *Statistical learning theory*. Wiley–Interscience.

Veloso, A., Meira Jr., W. (2006). Lazy associative classification for content-based spam detection. In *Proc of the Latin Amer web cong*.

Wang, X.-L., & Cloete, I. (2005). Learning to classify email: A survey. In *Proc of the int conf on mach learn and cybernetics* (Vol. 9).

Wang, C.-C., & Chen, S.-Y. (2007). Using header session messages to anti-spamming. *Computers and Security, 26*(5), 381–390.

Wang, Z., Josephson, W., Lv, Q., Charikar, M., & Li, K. (2007). Filtering image spam with near-duplicate detection. In *Proc of the fourth conf on email and anti-spam*.

Wang, B., Jones, G. J. F., & Pan, W. (2006). Using online linear classifiers to filter spam emails. *Pattern Analysis and Applications, 9*(4), 339–351.

Webb, S., Chitti, S., & Pu, C. (2005). An experimental evaluation of spam filter performance and robustness against attack. In *Proc of the int conf on collab comput: Networking, appl and worksharing*.

Wei, C.-P., Chen, H.-C., & Cheng, T.-H. (2008). Effective spam filtering: A single-class learning and ensemble approach. *Decision Support Systems, 45*(3), 491–503.

Wouters, P. (2004). *Why spam is bad*. <http://www.xtdnet.nl/paul/spam/> (visited on July 2008).

Wu, C.-T., Cheng, K.-T., Zhu, Q., & Wu, Y.-L. (2005). Using visual features for anti-spam filtering. In *Proc of the IEEE int conf on image proc* (Vol. 3).

Wu, C.-H. (2009). Behavior-based spam detection using a hybrid method of rule-based techniques and neural networks. *Expert Systems with Applications, 36*(3), 4321–4330.

Yih, W.-T., Goodman, J., & Hulton, G. (2006). Learning at low false positive rates. In *Proc of the third conf on email and anti-spam*.

Yu, B., & Ben Xu, Z. (2008). A comparative study for content-based dynamic spam classification using four machine learning algorithms. *Knowledge-Based Systems, 21*(4), 355–362.

Yue, X., Abraham, A., Chi, Z.-X., Hao, Y.-Y., & Mo, H. (2007). Artificial immune system inspired behavior-based anti-spam filter. *Soft Computing, 11*, 729–740.

Yu, H., Han, J., & Chang, K. C.-C. (2004). PEBL: Web page classification without negative examples. *IEEE Transactions on Knowledge and Data Engineering, 16*(1), 70–81.

Zhang, L., Zhu, J., & Yao, T. (2004). An evaluation of statistical spam filtering techniques. *ACM Transactions on Asian Language Information Processing, 3*(4), 243–269.

Zhao, W., & Zhang, Z. (2005). An email classification model based on rough set theory. In *Proc of the int conf on active media technology*.

Zhou, Y. (2007). Adaptive spam filtering using dynamic feature spaces. *International Journal of Artificial Intelligence Tools, 16*(4), 627–646.

Zorkadis, V., & Karras, D. A. (2006). Efficient information theoretic extraction of higher order features for improving neural network-based spam e-mail categorization. *Journal of Experimental and Theoretical Artificial Intelligence, 18*(4), 523–534.

Zorkadis, V., Karras, D. A., & Panayoto, M. (2005). Efficient information theoretic strategies for classifier combination, feature extraction and performance evaluation in improving false positives and false negatives for spam e-mail filtering. *Neural Networks, 18*(5–6), 799–807.