Information Sciences 277 (2014) 421–444

Contents lists available at ScienceDirect

Information Sciences

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

Study on the effectiveness of anomaly detection for spam filtering

Carlos Laorden *, Xabier Ugarte-Pedrero, Igor Santos, Borja Sanz, Javier Nieves, Pablo G. Bringas

Laboratory for Smartness, Semantics and Security (S³Lab), University of Deusto, Avenida de las Universidades 24, 48007 Bilbao, Spain

ARTICLE INFO

Article history:
Received 5 November 2011
Received in revised form 24 May 2013
Accepted 15 February 2014
Available online 25 March 2014

Keywords:
Spam filtering
Anomaly detection
Secure e-commerce
Computer security

ABSTRACT

Spam has become an important problem for computer security because it is a channel for spreading threats, including computer viruses, worms and phishing. Currently, more than 85% of received emails are spam. Historical approaches to combating these messages, including simple techniques such as sender blacklisting or using email signatures, are no longer completely reliable on their own. Many solutions utilise machine-learning approaches trained with statistical representations of the terms that usually appear in the emails. Nevertheless, these methods require a time-consuming training step with labelled data. Dealing with the limited availability of labelled training instances slows down the progress of filtering systems and offers advantages to spammers. In this paper, we present a study of the effectiveness of anomaly detection applied to spam filtering, which reduces the necessity of labelling spam messages and only employs the representation of one class of emails (i.e., legitimate or spam). This study includes a presentation of the first anomaly based spam filtering system, an enhancement of this system that applies a data reduction algorithm to the labelled dataset to reduce processing time while maintaining detection rates and an analysis of the suitability of choosing legitimate emails or spam as a representation of normality.

1. Introduction

Electronic mail is a powerful communication channel. However, as with all useful media, it is prone to misuse. Flooding inboxes with annoying and time-consuming messages, more than 85% of received emails are spam.[1] Bulk email is not only annoying to everyday email users but also constitutes a major computer security problem that costs billions of dollars in productivity losses [6]. It is also commonly used as a medium for *phishing* (i.e., attacks that seek to acquire sensitive information from end-users) [18] and the spread of malicious software (e.g., computer viruses, Trojan horses, spyware and Internet worms) [6].

Different studies have shown that spam has a notorious and prejudicial effect on the worldwide economy. Leung and Liang [25] presented an analysis of the impact of phishing on the market value of global firms, which showed that phishing alerts lead to a significant negative return on stock. In a similar vein, Mostafa Raad et al. [30] offered another study to assess

* Corresponding author. Tel.: +34 944139003; fax: +34 944139166.
E-mail addresses: claorden@deusto.es (C. Laorden), xabier.ugarte@deusto.es (X. Ugarte-Pedrero), isantos@deusto.es (I. Santos), borja.sanz@deusto.es (B. Sanz), jnieves@deusto.es (J. Nieves), pablo.garcia.bringas@deusto.es (P.G. Bringas).

[1] http://www.spam-o-meter.com/ (October 17, 2011).

http://dx.doi.org/10.1016/j.ins.2014.02.114
0020-0255/

the influence and impact of spam in several companies whose email advertisements were considered spam. Both examples clearly illustrate the necessity to detect undesired messages and, perhaps more importantly, the need to restore users confidence in their email filtering systems.

The academic community has proposed several approaches to solve the spam problem [34,9,39,10]. Among them, the *statistical approaches* [42] use machine-learning techniques to classify emails. These approaches have proven their efficiency in detecting spam and are the most utilised techniques to fight it. In particular, Bayes' theorem is widely used by anti-spam filters (e.g., SpamAssassin [27], Bogofilter [33] and Spamprobe [7]).

Statistical approaches are usually supervised, as they require a training set of previously labelled samples. These techniques perform better as more training instances are available, which requires a significant amount of previous labelling work to increase the models' accuracy. This work includes a gathering phase, in which as many emails as possible are collected. Nonetheless, the availability of labelled training instances is limited, which slows the progress of anti-spam systems.

In light of these difficulties, we propose the application of anomaly detection to spam filtering. Our approach can determine whether or not an email is spam by comparing word frequency features with a dataset composed only of what is considered normal (i.e., usually legitimate emails). If the email under inspection presents a considerable deviation from what is considered typical, it is considered an anomaly, or spam. This method does not need updated data about spam messages and thus reduces the efforts of labelling messages, working, for instance, only with a user's valid inbox folder.

By studying our method, we noticed that the number of comparisons needed to analyse each sample was considerably high (i.e., comparison against every legitimate email), resulting in a high processing overhead. We therefore present an enhancement to our approach by applying partitional clustering to reduce the number of vectors in the dataset used as normality. This improvement boosts scalability due to the reduction in processing time.

Finally, because the amount of spam within all email messages greatly exceeds the number of legitimate emails, a question regarding the suitability of choosing legitimate emails, instead of spam, as a representation of normality, may arise. Therefore, we performed a thorough study on the issue, providing comparisons between the two approaches, using both legitimate emails and spam as representations of normality.

In summary, our main findings presented in this paper include the following:

- We present an anomaly-based approach for spam filtering by proposing different deviation measures to determine whether an email is spam.
- We adapt a method for email dataset reduction based on the partitional clustering algorithm Quality Threshold (QT) and generate reduced datasets of different sizes.
- We empirically validate the reduction algorithm by testing its accuracy results and comparing them to the approach using the unreduced datasets.
- We prove that a unique, synthetically generated sample of legitimate emails is representative enough to implement an anomaly detection system without compromising accuracy results.
- We show that labelling efforts can be reduced in the industry, while still maintaining a high rate of accuracy.

The remainder of this paper is organised as follows. Section 2 provides background regarding the representation of emails based on the Vector Space Model (VSM). Section 3 details our anomaly based method. Section 4 describes the experiments and presents the results of the approach without the dataset reduction. Section 5 details the application of the dataset reduction step to our anomaly based method. Section 6 describes the experiments and presents the results of our anomaly based approach enhanced with the dataset reduction, offering a comparison with the approach that does not perform the reducing step. Section 7 compares the use of legitimate emails against spam as representations of normality. Section 9 discusses the implications of the obtained results and shows the limitations of the proposed approach. Finally, Section 10 concludes the paper and outlines avenues for future work.

2. Vector space model for spam filtering

Spam filtering software attempts to accurately classify email messages into 2 main categories: spam or legitimate messages. We thus use information found within the body and subject of an email message and discard every other piece of information (including the sender or time-stamp of the email). To represent messages, we remove the stop-words [40], which are words devoid of content (e.g., 'a', 'the', 'is'). These words do not provide any semantic information and add noise to the model [36].

We then represent the emails using an Information Retrieval (IR) model. Formally, let an IR model be defined as a 4-tuple $[\mathcal{E}, \mathcal{Q}, \mathcal{F}, R, (q_i, e_j)]$ [3] where.

- $\mathcal{E}$ is a set of representations of email.
- $\mathcal{Q}$ is a set of representations of user queries.
- $\mathcal{F}$ is a framework for modelling emails, queries and their relationships.
- $R(q_i, e_j)$ is a ranking function that associates a real number with a query $q_i,\ (q_i \in \mathcal{Q})$ and an email representation $e_j,\ (e_j \in \mathcal{E})$. This function is also called a similarity function.

Let $\mathcal{E}$ be a set of text emails $e, e : \{t_1, t_2, \ldots t_n\}$, each comprising an n number of t terms. We consider $w_{i,j}$ a weight for term t_i in an email e_j, whereas if $w_{i,j}$ is not present in e_j, then $w_{i,j} = 0$. Therefore, an email can be represented as a vector, starting from its origin, of index terms $\vec{e_j} = (w_{1,j}, w_{2,j}, \ldots, w_{n,j})$.

Using this formalisation, we can apply several IR models. Spam filtering systems commonly use the VSM. The VSM represents natural language documents in an algebraic fashion by placing the vectors in a multidimensional space. This space is formed by only positive axis intercepts. In addition, documents are represented as a term-by-document matrix, where the (i,j)th element illustrates the association between the ith term and the jth document. This association reflects the occurrence of the ith term in document j. Terms can represent different text units (e.g., a word or phrase) and can also be individually weighted, allowing terms to become more or less important within a document or the entire document collection as a whole.

Specifically, we use term frequency-inverse document frequency (tf-idf) [28] to obtain the weight of each word, whereas the weight of the ith word in the jth email, denoted by $weight(i,j)$, is defined by

$$weight(i,j) = tf_{i,j} \cdot idf_i \tag{1}$$

where the term frequency $tf_{i,j}$ [28] is defined as:

$$tf_{i,j} = \frac{m_{i,j}}{\sum_k m_{k,j}} \tag{2}$$

where $m_{i,j}$ is the number of times the word $t_{i,j}$ appears in an email e_j and $\sum_k m_{k,j}$ is the total number of words in an email e_j.

Conversely, the inverse document frequency idf_i is defined as:

$$idf_i = \frac{|\mathcal{E}|}{|\mathcal{E} : t_i \in e|} \tag{3}$$

where $|\mathcal{E}|$ is the total number of documents and $|\mathcal{E} : t_i \in e|$ is the number of documents containing the word $t_{i,j}$.

We apply relevance weights to each feature based on Information Gain (IG) [20]:

$$IG(j) = \sum_{v_j \in R} \sum_{C_i} P(v_j, C_i) \cdot \frac{P(v_j, C_i)}{P(v_j) \cdot P(C_i)} \tag{4}$$

where C_i is the i-th class, v_j is the value of the j-th interpretation, $P(v_j, C_i)$ is the probability that the j-th attribute has the value v_j in the class C_i, $P(v_j)$ is the probability that the j-th interpretation has the value v_j in the training data and $P(C_i)$ is the probability of the training dataset belonging to the class C_i. IG provides a ratio for each feature that measures its importance to consider whether or not a sample is spam. These weights help obtain a better distance rating among samples.

3. Anomaly detection

Anomaly detection approaches model normality by considering any deviation from this model to be anomalous. Using the word-frequency features of the VSM described above, our anomaly detection system analyses points in the feature space and classifies emails based on their similarity. The analysis of an email consists of 2 different phases:

- Extraction of features from the email.
- Measuring the distance from the point representing the email to the points that symbolise normality.

As a result, any point at a distance from normality that surpasses an established threshold is considered an anomaly. In this study, we considered 2 different distance measures:

- **Manhattan Distance:** The distance between two points, v and u, is the sum of the lengths of the projections of the line segments between the two points onto the coordinate axes

$$d(x,y) = \sum_{i=0}^{n} |x_i - y_i| \tag{5}$$

where x is the first point, y is the second point, and x_i and y_i are the ith components of the first and second points, respectively.

- **Euclidean Distance:** This distance is the length of the line segment connecting two points. It is calculated as

$$d(x,y) = \sum_{i=0}^{n} \sqrt{x_i^2 - y_i^2} \tag{6}$$

where x is the first point, y is the second point, and x_i and y_i are the ith components of the first and second points, respectively.

Table 1
Comparison of the used datasets. The spam ratio in all three datasets does not follow the statistics of the number of spam messages in the real world which is higher than 85%. The SpamAssassin and TREC datasets, however, contain more realistic spam emails and examples of obfuscated mails within them. Due to computational limitations, we only used a randomly extracted 30% of the TREC dataset (maintaining the spam ratio).

Feature	LingSpam	SpamAssassin	TREC
No. spam messages	480	1896	14,973
No. of ham messages	2412	4150	7653
Spam %	16.60%	31.36%	66.18%

With these measures, we can compute the deviations between two different emails. Because we must compute these measures with points representing legitimate emails, a combination metric is required to obtain a final distance value that considers every measure performed. To this end, our system employs 3 simple metrics:

- The mean value calculated from every distance value in the training dataset.
- The lowest distance value from every distance value in the training dataset.
- The highest value of the computed distances from every distance value in the training dataset.

When our method inspects an email a final distance value is acquired, which depends on both the distance measure and combination metric.

4. Empirical validation of the simple anomaly detection method

To validate our proposed method, we used the LingSpam Corpus,[2] SpamAssassin[3] public corpus and TREC 2007 Public Corpus.[4]

LingSpam contains a mixture of both spam and legitimate messages retrieved from the *Linguistic list*, an email distribution list about *linguistics*. It has 2893 different emails, of which 2412 are legitimate emails obtained by downloading digests from the list and 481 are spam emails retrieved from the inbox of one of the authors.[5] From the datasets provided in this corpus, each with different pre-processing steps, we chose the *Bare* dataset, which has no pre-processing.

The SpamAssassin corpus contains a total of 6,046 messages, of which 1896 are spam and 4150 are legitimate emails.

The TREC 2007 Public Corpus [12] contains all email messages delivered to a server between April 8 and July 6, 2007. The server contained accounts that had fallen into disuse but that continued receiving spam. To these accounts were added a number of 'honeypot' accounts published on the Web and used to sign up for a number of services, some legitimate and some not. The dataset contains 75,419 messages, of which 25,220 are legitimate email and 50,199 are junk messages. However, for our experiments, we randomly extracted 30% (due to computational limitations) of the full subcorpora, maintaining the spam-legitimate ratio. Our TREC dataset thus contains 7653 legitimate emails and 14,973 junk messages. Table 1 presents an overview of the dimensions of the three datasets.

We performed a *Stop Word Removal* [40] on the three datasets based on an external stop-word list[6] and removed any non alpha-numeric characters. We then used the VSM [37], an algebraic approach for Information Filtering (IF), IR, indexing and ranking, to create the model. This model mathematically represents natural language documents through vectors in a multidimensional space. Finally we extracted the top 1000 attributes using IG.

Specifically, we followed the next configuration for the empirical validation:

1. **Cross-validation.** For the LingSpam dataset, we performed a 5-fold cross-validation [21] forming 3 different divisions of 1930 emails and 2 divisions of 1929 emails to represent normality and another 3 divisions of 482 emails and 2 of 483 to measure deviations within legitimate email. Each fold thus contains 1930 or 1929 legitimate emails that represent normality and 963 or 962 testing emails, from which 483 or 482 were legitimate emails and 480 were spam. The number of legitimate emails varied in the last 2 folds because the number of legitimate emails was not divisible by 5.
   Regarding the SpamAssassin dataset, we also performed a 5-fold cross- validation to divide the dataset composed of legitimate emails (the normal behaviour) into 5 different divisions of 3320 emails to represent normality and 830 to measure deviations within legitimate emails. Each fold contains 3320 legitimate emails that represent normality and 2726 testing emails, from which 830 were legitimate emails and 1896 were spam.
   Finally, for the TREC dataset, we again performed a 5-fold cross-validation by forming 3 different divisions of 6122 emails

[2] Available at: http://nlp.cs.aueb.gr/software_and_datasets/lingspam_public.tar.gz.
[3] Available at: http://spamassassin.org/publiccorpus.
[4] Available at: http://plg.uwaterloo.ca/~gvcormac/spam.
[5] For a more detailed description of the corpus please refer to [1,35].
[6] http://www.webconfs.com/stop-words.php.

Table 2
Number of instances within each fold of the 5-fold cross-validation process. The number of spam emails within SpamAssassin and TREC varied in the folds because the number of spam emails was not divisible by 5.

	Normality	Deviations	
	# Legit.	# Legit.	# Spam
LingSpam			
Fold 1	1929	483	480
Fold 2	1929	483	480
Fold 3	1930	482	480
Fold 4	1930	482	480
Fold 5	1930	482	480
SpamAssassin			
Fold 1	3320	830	1896
Fold 2	3320	830	1896
Fold 3	3320	830	1896
Fold 4	3320	830	1896
Fold 5	3320	830	1896
TREC			
Fold 1	6122	1531	2994
Fold 2	6122	1531	2994
Fold 3	6122	1531	2994
Fold 4	6123	1530	2994
Fold 5	6123	1530	2994

and two divisions of 6123 emails to represent normality and another 3 divisions of 1531 emails and 2 of 1530 to measure deviations within legitimate email. Each fold thus contains 6122 or 6123 legitimate emails that represent normality and 4525 or 4524 testing emails, from which 1531 or 1530 were legitimate emails and 2994 were spam. The number of legitimate emails varied in the two folds because the number of legitimate emails was not divisible by 5 (Table 2).

2. **Calculating distances and combination rules.** We extracted the aforementioned characteristics and employed the 2 different measures and 3 different combination rules described in Section 3 to obtain a final measure of deviation for each piece of testing evidence. More accurately, we applied the Manhattan and Euclidean distances. For the combination rules, we tested the mean value, lowest distance and highest value.
3. **Defining thresholds.** For each measure and combination rule, we established 10 different thresholds to determine whether an email was spam. These thresholds were selected by first establishing the lowest one. This number was the highest possible value at which no spam messages were misclassified. The highest one was selected as the lowest possible value at which no legitimate spam messages were misclassified. The remaining thresholds were selected by equally dividing the range between the first and last thresholds.
   The method is thus configurable to reduce both false positives and false negatives. It is important to define whether it is better to classify spam as legitimate or legitimate as spam. In particular, one may think that it is more important to detect more spam messages than to minimise false positives. For commercial reasons, one may think just the opposite: a user can be bothered if their legitimate messages are flagged as spam. To improve these errors, we applied two techniques: (i) white and blacklisting or (ii) cost-sensitive learning. White and blacklists store a signature of an email to be flagged as either spam (blacklisting) or legitimate messages (whitelisting). Conversely, cost-sensitive learning is a machine-learning technique where one can specify the cost of each error, and the classifiers are trained to account for that consideration [15]. We can adapt cost-sensitive learning for anomaly detection with cost matrices.
4. **Testing the method.** To evaluate the results, we measured the False Negative Ratio (FNR) and the False Positive Ratio (FPR). FNR is defined as:

$$\mathrm{FNR}(\beta) = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TP}} \tag{7}$$

where TP is the number of spam emails correctly classified (true positives) and FN is the number of spam messages misclassified as legitimate (false negatives).
FPR is defined as:

$$\mathrm{FPR}(\beta) = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} \tag{8}$$

where FP is the number of legitimate emails incorrectly classified as spam while TN is the number of legitimate messages correctly classified.
Moreover, we measured the Weighted Accuracy (WA), which is defined as:

$$\mathrm{WA}(\beta) = 1 - \frac{\mathrm{FNR} + \mathrm{FPR}}{2} \tag{9}$$

This measure is calculated due to the unbalanced nature of the datasets. Calculating the average of the FNR and FPR, we attempt to "objectively" show the best results for each configuration.

Finally, we evaluated the Area Under the Receiver Operating Charac- teristic Curve (AUC). The AUC is the area under the curve formed by the union of the points representing the FPR and True Positive Ratio (TPR) for each possible threshold in a plot where the X axis represents the FPR and the Y axis represents the TPR. To calculate the AUC we used the points corresponding to the 10 thresholds selected. The area under the curve formed by these points was calculated by dividing it into 9 trapezoidal subareas and computing them independently:

$$AUC = \sum_{i=0}^{i=9} (x_{i+1} - x_i) \cdot y_i + \frac{(x_{i+1} - x_i) \cdot (y_{i+1} - y_i)}{2}$$

Table 3 shows the obtained results for the LingSpam corpus using different distances, combination rules and thresholds. Using this dataset, the best configuration was the one performed with the Euclidean Distance, the Mean combination rule and 2.59319 as the threshold, which provided a 92.27% WA, with an FNR of 8.42% and an FPR of 7.05%.

Table 4 shows the obtained results for the SpamAssassin corpus for the different distances, combination rules and thresholds. The best results were obtained with the Manhattan Distance, the Min combination rule and a 1.37525 threshold, which provided a 91.63% WA, with an FNR of 7.48% and an FPR of 9.25%.

Finally, Table 5 shows the obtained results for the TREC corpus for different distances, combination rules and thresholds. The best results were obtained with the Euclidean Distance, the Min combination rule and a 0.08532 threshold, which provided a 74.35% WA, with an FNR of 42.93% and an FPR of 8.36%.

Figs. 1–3 show different plots for the Receiver Operating Characteristic (ROC) curve for each different distance measure and selection rule.

The best results were obtained with the Min distance, which is the most conservative configuration for distance, which highlights a possible discussion topic regarding what should be called anomaly in emails. As stated above, more than 85% of emails today are spam; therefore, in terms of normality, receiving a legitimate email is an anomaly. Further study on this topic is presented in Section 7.

Table 3
Results for different combination rules and distance measures using the LingSpam corpus. The abbreviation 'Thres'. stands for the chosen threshold. The results in bold are the best for each combination rule and distance measure.

Comb.	Manhattan distance				Euclidean distance			
	Thres.	FNR (%)	FPR (%)	WA (%)	Thres.	FNR (%)	FPR (%)	WA (%)
Mean	1.86313	0.00	100.00	50.00	1.87061	0.00	99.88	50.06
	2.22637	0.21	99.13	50.33	2.11147	0.21	97.89	50.95
	2.58960	2.79	92.87	52.17	2.35233	1.67	45.23	76.55
	2.95284	5.71	73.51	60.39	**2.59319**	**8.42**	**7.05**	**92.27**
	3.31608	15.92	43.37	70.36	2.83405	47.29	1.45	75.63
	3.67931	26.46	19.24	77.15	3.07490	80.25	0.29	59.73
	4.04255	**37.71**	**5.10**	**78.60**	3.31576	92.63	0.12	53.63
	4.40579	47.88	1.12	75.50	3.55662	96.46	0.04	51.75
	4.76902	60.63	0.12	69.63	3.79748	98.96	0.04	50.50
	5.13226	70.13	0.00	64.94	4.03834	99.38	0.00	50.31
Max	3.69053	0.00	99.96	50.02	3.22709	0.00	99.79	50.10
	3.99470	0.21	99.25	50.27	3.41297	0.04	96.31	51.82
	4.29888	1.21	96.77	51.01	3.59886	0.67	86.44	56.45
	4.60305	3.21	87.89	54.45	3.78474	3.88	78.61	58.76
	4.90722	7.00	69.44	61.78	3.97063	13.38	57.55	64.54
	5.21140	16.88	44.32	69.40	**4.15651**	**20.71**	**12.89**	**83.20**
	5.51557	25.71	22.51	75.89	4.34240	41.50	1.66	78.42
	5.81974	**36.54**	**8.25**	**77.60**	4.52828	81.83	0.41	58.88
	6.12392	48.63	1.70	74.84	4.71417	93.63	0.12	53.13
	6.42809	59.46	0.00	70.27	4.90005	97.54	0.00	51.23
Min	0.09919	0.00	98.34	50.83	0.69584	0.00	96.85	51.58
	0.51575	0.62	94.61	52.38	1.00615	0.21	95.48	52.16
	0.93230	3.96	85.95	55.05	1.31645	0.21	91.83	53.98
	1.34886	8.96	69.20	60.92	1.62676	1.67	66.04	66.14
	1.76542	21.88	48.05	65.04	**1.93707**	**6.88**	**13.23**	**89.95**
	2.18197	33.29	25.83	70.44	2.24737	37.21	1.58	80.61
	2.59853	**44.58**	**10.57**	**72.42**	2.55768	79.33	0.21	60.23
	3.01509	56.67	2.40	70.46	2.86799	92.88	0.04	53.54
	3.43164	67.71	0.37	65.96	3.17829	98.67	0.04	50.65
	3.84820	76.88	0.00	61.56	3.48860	99.71	0.00	50.15

Table 4
Results for different combination rules and distance measures using the SpamAssassin corpus. The abbreviation 'Thres'. stands for the chosen threshold. The results in bold are the best for each combination rule and distance measure.

Comb.	Manhattan distance				Euclidean distance			
	Thres.	FNR (%)	FPR (%)	WA (%)	Thres.	FNR (%)	FPR (%)	WA (%)
Mean	1.15978	0.00	99.98	50.01	1.70013	0.00	99.59	50.20
	1.58697	0.14	95.23	52.32	1.91763	2.23	69.98	63.90
	2.01417	1.09	58.48	70.22	**2.13512**	**18.96**	**25.88**	**77.58**
	2.44136	**7.15**	**20.89**	**85.98**	2.35262	43.47	8.89	73.82
	2.86856	23.82	5.37	85.40	2.57011	69.37	3.73	63.45
	3.29575	49.56	1.52	74.46	2.78761	87.43	1.95	55.31
	3.72295	72.38	0.39	63.62	3.00510	93.19	0.92	52.95
	4.15014	85.16	0.27	57.29	3.22260	96.87	0.36	51.39
	4.57734	92.29	0.12	53.80	3.44009	98.59	0.10	50.66
	5.00453	95.45	0.00	52.27	3.65759	99.07	0.00	50.46
Max	3.39114	0.00	100.00	50.00	3.41015	0.00	99.45	50.28
	3.70912	0.04	99.78	50.09	3.55333	2.34	82.55	57.55
	4.02710	0.44	98.77	50.39	3.69652	17.77	36.14	73.04
	4.34509	2.33	95.08	51.29	**3.83970**	**44.58**	**8.70**	**73.36**
	4.66307	8.07	83.30	54.31	3.98288	70.14	2.94	63.46
	4.98105	23.78	46.19	65.02	4.12607	87.99	1.52	55.25
	5.29903	**48.82**	**12.14**	**69.52**	4.26925	93.54	0.72	52.87
	5.61702	68.92	2.70	64.19	4.41243	97.08	0.34	51.29
	5.93500	84.68	0.24	57.54	4.55562	98.54	0.10	50.68
	6.25298	92.51	0.00	53.74	4.69880	98.99	0.00	50.51
Min	0.04335	0.00	99.71	50.14	0.44679	0.00	99.04	50.48
	0.37633	0.11	86.70	56.60	0.76440	0.08	95.86	52.03
	0.70930	0.55	50.12	74.67	1.08201	0.22	76.31	61.73
	1.04228	2.24	20.67	88.54	**1.39962**	**6.00**	**18.41**	**87.79**
	1.37525	**7.48**	**9.25**	**91.63**	1.71723	31.39	2.00	83.30
	1.70823	17.93	4.29	88.89	2.03484	71.46	0.24	64.15
	2.04120	37.69	1.90	80.20	2.35245	93.98	0.05	52.99
	2.37418	58.95	0.75	70.15	2.67006	98.14	0.05	50.90
	2.70715	78.22	0.27	60.76	2.98767	99.15	0.02	50.42
	3.04013	88.69	0.00	55.65	3.30528	99.64	0.00	50.18

Note that to provide impartial results we present as best results, those with the higher WA, but for commercial purposes, as stated before, using configurations offering lower FNR or FPR (depending on the desired goals) is recommended.

5. Improving the efficiency by reducing the normality dataset

Dataset reduction is a step that must be faced in different problems when working with large datasets. In the approach presented in Section 3 that uses non-reduced datasets, the experiments were performed with a base of over 2000 (for the LingSpam dataset), over 4000 (for the SpamAssassin dataset) and over 14,000 (for the TREC dataset) legitimate emails, which means that every sample analysed had to be compared 2000, 4000 or 14,000 times to classify it as spam or not. Therefore, we propose a data reduction algorithm based on partitional clustering.

Cluster analysis divides data into meaningful groups [22]. These techniques usually employ distance measures to compare instances in datasets to group those that appear to be similar. We can identify several types of clustering, but the most common are hierarchical and partitional clustering.

The first approach generates clusters in a nested style, which means that the dataset is divided into a set of clusters that are subdivided into other clusters related hierarchically. Conversely, partitional clustering techniques create a one-level (unnested) partitioning of the data points [22]. We are interested in this last technique to validate our initial hypothesis because it makes it possible to divide a large set of emails that represent normality (i.e., legitimate emails) into a reduced set of representations.

Heyer et al. [16] proposed the QT clustering algorithm to extract useful information from large amounts of gene expression data (Fig. 4). This clustering algorithm need not specify the number of clusters desired. It uses a similarity threshold value to determine the maximum radial distance of any cluster. It thus generates a variable number of clusters that meet a QT. Its main disadvantage is the high number of distance calculations needed. Nevertheless, this computational overhead is admissible in this case, as we must only reduce the dataset once (we employ a static representation of normality that remains invariable).

Our algorithm, shown in Fig. 5, is based on the concepts proposed by Heyer et al. [16]; it is adapted to our data reduction problem and implemented iteratively, instead of recursively.

Table 5
Results for different combination rules and distance measures using the TREC corpus. The abbreviation 'Thres'. stands for the chosen threshold. The results in bold are the best for each combination rule and distance measure.

Comb.	Manhattan distance				Euclidean distance			
	Thres.	FNR (%)	FPR (%)	WA (%)	Thres.	FNR (%)	FPR (%)	WA (%)
Mean	0.46807	0.00	100.00	50.00	1.18474	0.00	100.00	50.00
	0.61292	33.13	64.89	50.99	1.42634	27.07	71.10	50.92
	0.75777	46.24	46.66	53.55	1.66794	64.56	30.15	52.65
	0.90263	53.37	37.37	54.63	1.90954	76.18	22.07	50.87
	1.04748	57.37	28.39	57.12	2.15114	78.08	17.61	52.15
	1.19233	**64.38**	**18.48**	**58.57**	2.39273	78.60	13.04	54.18
	1.33718	70.46	14.50	57.52	**2.63433**	**78.66**	**12.41**	**54.46**
	1.48204	71.47	12.91	57.81	**2.87593**	**78.66**	**12.41**	**54.46**
	1.62689	73.08	12.45	57.23	**3.11753**	**78.66**	**12.41**	**54.46**
	1.77174	94.47	0.00	52.76	3.35913	100.00	0.00	50.00
Max	1.57424	0.00	100.00	50.00	3.70404	0.00	100.00	50.00
	1.73442	35.18	63.06	50.88	3.76495	29.24	71.97	49.39
	1.89460	47.30	46.11	53.29	3.82585	60.25	39.04	50.35
	2.05478	54.29	37.04	54.33	3.88676	68.98	27.10	51.96
	2.21496	58.80	27.92	56.64	3.94767	76.26	22.49	50.63
	2.37515	67.35	18.80	56.92	4.00857	77.92	20.79	50.65
	2.53533	71.16	15.12	56.86	4.06948	78.23	17.60	52.09
	2.69551	73.05	12.99	56.98	4.13039	78.53	16.39	52.54
	2.85569	**73.13**	**12.45**	**57.21**	**4.19129**	**78.60**	**15.71**	**52.85**
	3.01587	94.47	0.00	52.76	4.25220	99.96	0.00	50.02
Min	0.00000	0.00	100.00	50.00	0.00000	0.00	100.00	50.00
	0.08532	**46.62**	**24.85**	**64.26**	0.17927	31.74	34.71	66.78
	0.17065	60.22	15.37	62.21	0.35854	33.43	25.85	70.36
	0.25597	66.52	7.13	63.17	**0.53782**	**42.93**	**8.36**	**74.35**
	0.34129	70.25	2.59	63.58	0.71709	65.78	1.02	66.60
	0.42662	74.86	0.88	62.13	0.89636	81.25	0.26	59.25
	0.51194	76.90	0.30	61.40	1.07563	90.98	0.18	54.42
	0.59726	80.04	0.10	59.93	1.25491	93.36	0.10	53.27
	0.68259	84.51	0.03	57.73	1.43418	99.61	0.01	50.19
	0.76791	90.02	0.00	54.99	1.61345	99.92	0.00	50.04

Formally, let $\mathcal{A} = \{\mathcal{A}_0, \mathcal{A}_1, \ldots, \mathcal{A}_n\}$ be the set of potential clusters. For each vector v_i in the dataset $\mathcal{V}$, there is a potential cluster $\mathcal{A}_i \in \mathcal{A}$. A potential cluster $\mathcal{A}_i$ is the set of vectors at a distance with respect to v_i not higher than the *threshold* previously specified.

After calculating the potential clusters, we select the cluster with the highest number of vectors as a final cluster. We calculate its centroid, defined as $c = x_1 + x_2 + \cdots + x_k/k$ where $x_1, x_2, \cdots, x_k$ are points in the feature space. The resultant centroid is added to the final reduced dataset. Each vector v_j present in the selected cluster $\mathcal{A}_i$ is then removed from the original dataset $\mathcal{V}$ (as they are represented by the previously calculated centroid).

Moreover, the potential clusters $\mathcal{A}_j \in \mathcal{A}$ associated with each vector v_j previously removed are also discarded. When no available clusters remain with a number of vectors higher than the parameter *minimum vectors*, the remaining vectors in $\mathcal{V}$ are added to the final reduced dataset and the algorithm finishes and returns the resulting reduced dataset.

The final result is a dataset containing one centroid representing each cluster and all vectors that were not associated with any cluster by the QT clustering algorithm.

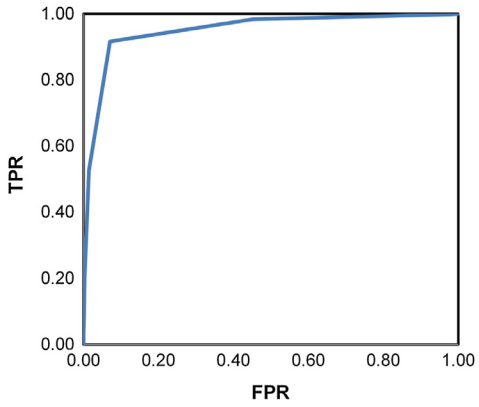
6. Empirical validation of the method with normality dataset reduction

To evaluate the performance of our method, we conducted an experiment with 2 phases: first, we reduced the set of vectors corresponding to the representation of the legitimate emails that represent normality, and second, we started the anomaly detection step to measure both accuracy and efficiency.

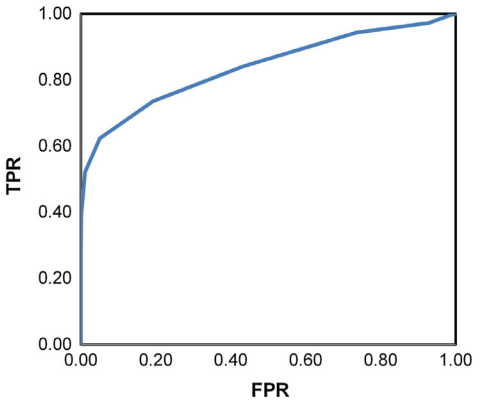
Again, we used the LingSpam, SpamAssassin and TREC 2007 datasets. To evaluate the performance of the predictors, we used k-fold cross-validation [21]. The same process was conducted for the approach without the reduction step and can be reviewed in Section 4.

To test the dataset reduction algorithm proposed, 4 experimental configurations were selected for each distance measure. The threshold parameter values for our QT clustering-based algorithm were selected by empirical observation and reference to the infinite threshold, which in practice is set to the maximum value allowed for a 64-bit double variable.

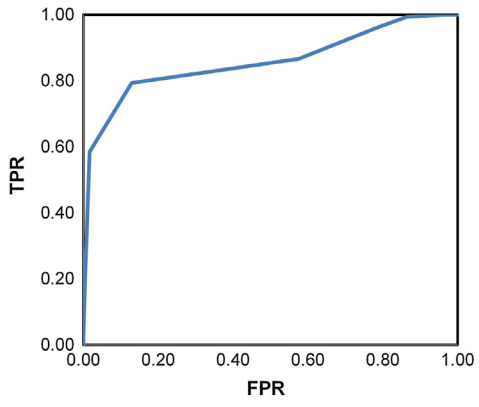
Table 6 shows the reductions obtained in the process. The result obtained for the infinity threshold is a unique centroid of the whole dataset that represents the arithmetic mean vector or a single representation of normality. In this case, selection rules did not influence the final result because the method only performed one single comparison for each sample.



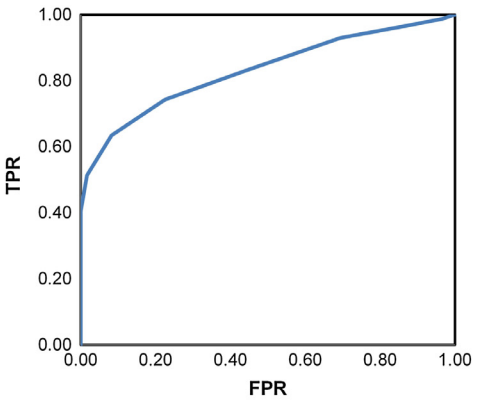
(a) ROC curve for the Euclidean distance and Mean selector.



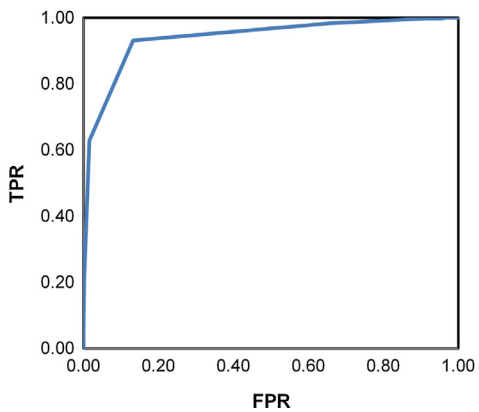
(b) ROC curve for the Manhattan distance and Mean selector.



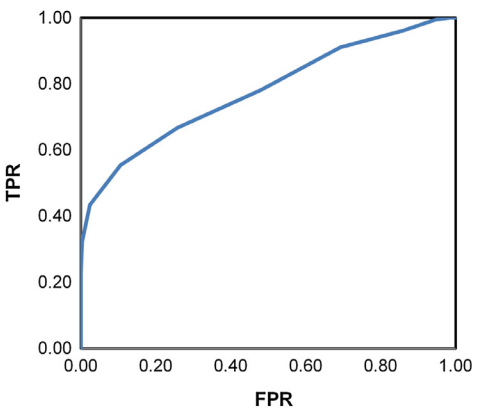
(c) ROC curve for the Euclidean distance and Max selector.



(d) ROC curve for the Manhattan distance and Max selector.

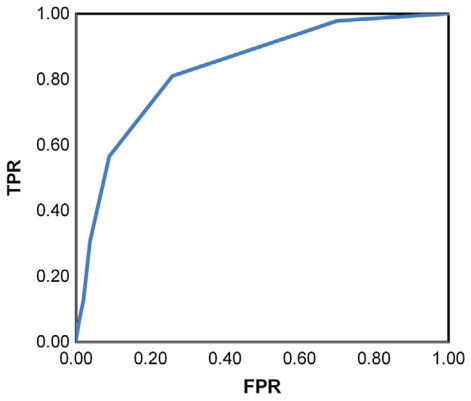


(e) ROC curve for the Euclidean distance and Min selector.

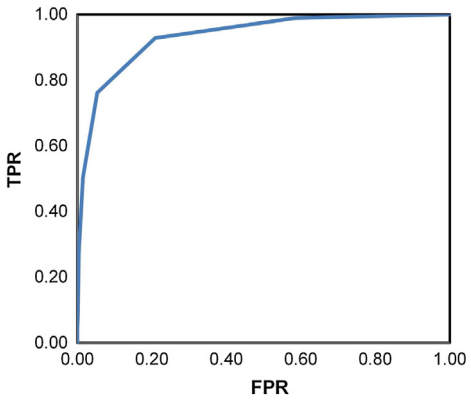


(f) ROC curve for the Manhattan distance and Min selector.

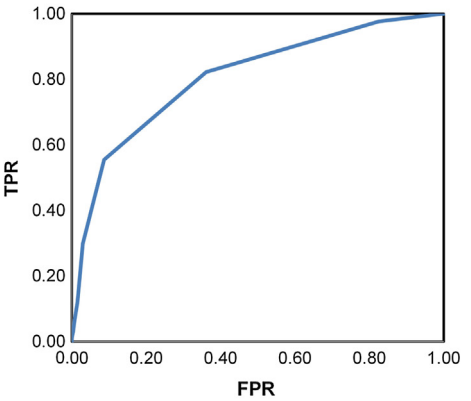
Fig. 1. ROC curves for the different experimental configurations applied to LingSpam.



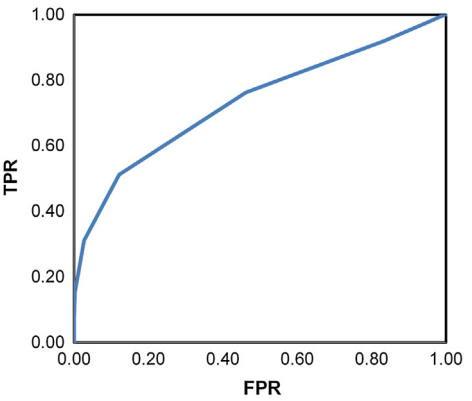
(a) ROC curve for the Euclidean distance and Mean selector.



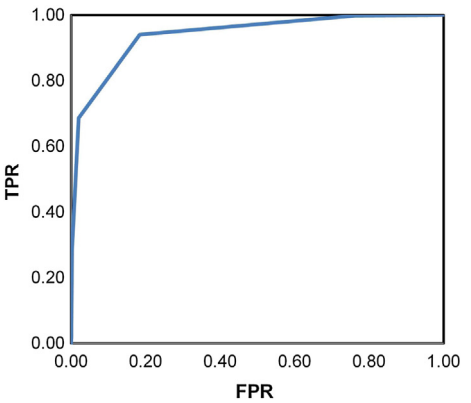
(b) ROC curve for the Manhattan distance and Mean selector.



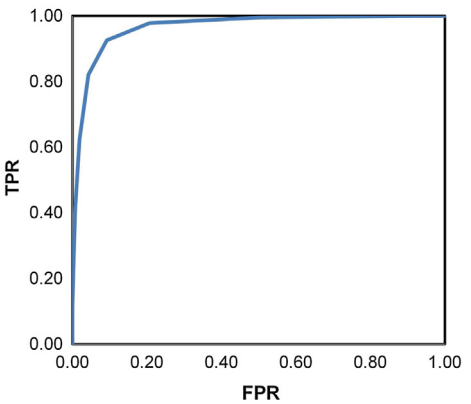
(c) ROC curve for the Euclidean distance and Max selector.



(d) ROC curve for the Manhattan distance and Max selector.

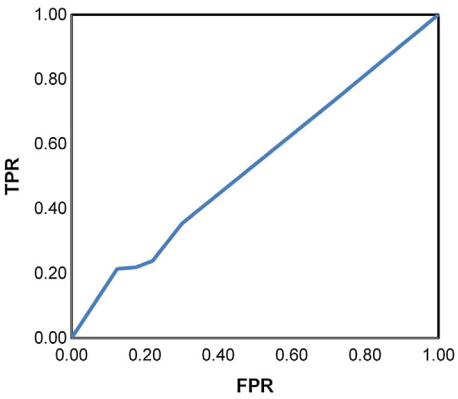


(e) ROC curve for the Euclidean distance and Min selector.

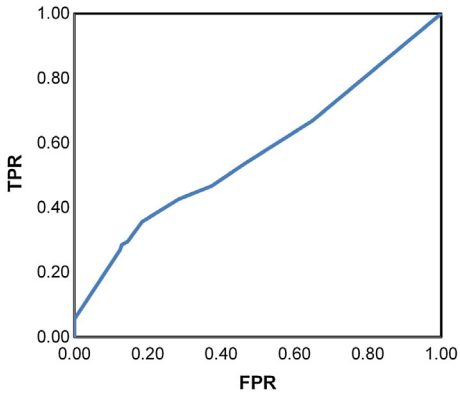


(f) ROC curve for the Manhattan distance and Min selector.

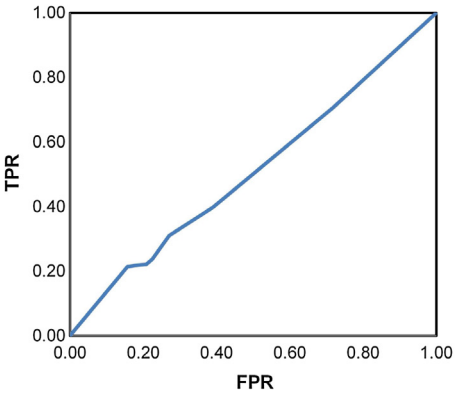
Fig. 2. ROC curves for the different experimental configurations applied to SpamAssassin.



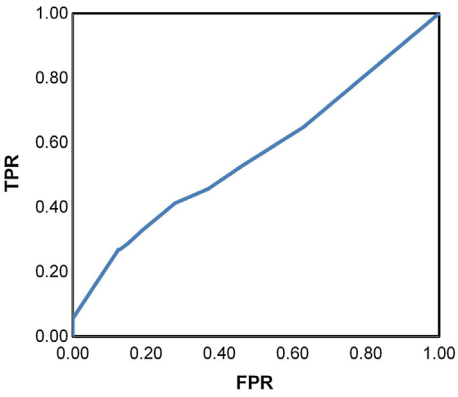
(a) ROC curve for the Euclidean distance and Mean selector.



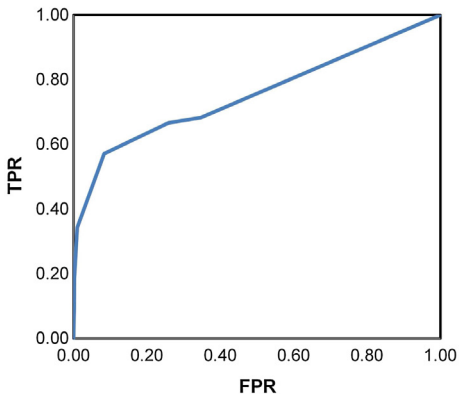
(b) ROC curve for the Manhattan distance and Mean selector.



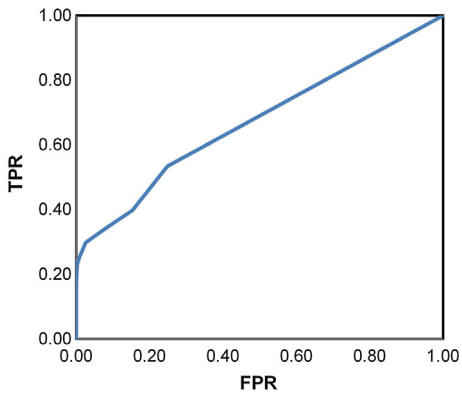
(c) ROC curve for the Euclidean distance and Max selector.



(d) ROC curve for the Manhattan distance and Max selector.



(e) ROC curve for the Euclidean distance and Min selector.



(f) ROC curve for the Manhattan distance and Min selector.

Fig. 3. ROC curves for the different experimental configurations applied to TREC.

Furthermore, during our experimental evaluation, we measured the times employed in both data reduction and anomaly detection:

- **Data reduction.** In this phase, we reduced the original datasets for each fold. We used 8 different configurations to reduce each different dataset: Euclidean distance (1.50, 1.75, 2.00 and ∞) and Manhattan distance (1.50, 1.75, 2.00 and ∞) for LingSpam and SpamAssassin; and Euclidean distance (0.25, 0.50, 1.00 and ∞) and Manhattan distance (0.50, 1.00, 1.25 and ∞) for TREC.

```
input  : The original dataset 𝒢, the distance threshold for each
         cluster d
output: The generated clusters
Procedure QT_Clust(G,d)
    // Base case.
    if |𝒢| ≤ 1 then
        output 𝒢
    else
        foreach {i|i ∈ 𝒢} do
            flag ⟵ TRUE // 𝒜_i is the cluster started by i
            𝒜_i ⟵ {i}
            while flag = TRUE, 𝒜_i ≠ 𝒢 do
                // Find j such that diameter(𝒜_i ∪ {j}) is
                   minimum.
                ∃j ∈ (𝒢 − 𝒜_i) : ∀k ∈ (𝒢 − 𝒜_i − j) : diameter(𝒜_i ∪ {j})
                <diameter(𝒜_i ∪ {k})
                if diameter(𝒜_i ∪ {j} > d) then
                    flag ⟵ FALSE
                else
                    // Add j to cluster 𝒜_i.
                    𝒜_i ⟵ 𝒜_i ∪ {j}
        // Obtain 𝒞 ∈ 𝒜 with maximum cardinality.
        output 𝒞 ∈ 𝒜 : ∀𝒜_i ∈ 𝒜 : |𝒞| ≥ |𝒜_i|
        QT_Clust(𝒢 − 𝒞,d)
```

Fig. 4. QT algorithm.

```
input  : The original dataset 𝒱, the distance threshold for each
         cluster threshold and the minimum number of vectors in each
         cluster minimumvectors
output: The reduced dataset ℛ
// Calculate the distance from each vector (set of email
   features) to the rest of vectors in the dataset.
foreach {v_i|v_i ∈ 𝒱} do
    foreach {v_j|v_j ∈ 𝒱} do
        // If a vector v_j's distance to v_i is lower than the
           specified threshold, then v_j is added to the
           potential cluster 𝒜_i, associated to the v_i vector
        if distance(v_i,v_j) ≥ threshold then
            𝒜_i.add(v_j)
// In each loop, select the potential cluster with the
   highest number of vectors
while ∃𝒜_i ∈ 𝒜 : |𝒜_i| ≥ minimumvectors and ∀𝒜_j ∈ 𝒜 : |𝒜_i| ≥ |𝒜_j|
and i ≠ j do
    // Add the centroid vector for the cluster to the
       result set
    ℛ.add(centroid(𝒜_i))
    // Discard potential clusters associated to vectors
       v_j ∈ 𝒜_i
    foreach {v_j|v_j ∈ 𝒜_i} do
        𝒜.remove(𝒜_j)
        𝒱.remove(v_j)
    // Remove vectors v_j ∈ 𝒜_i from the clusters 𝒜_k remaining
       in 𝒜
    foreach {𝒜_k|𝒜_k ∈ 𝒜} do
        foreach {v_j|v_j ∈ 𝒜_k and v_j ∈ 𝒜_i} do
            𝒜_k.remove(v_j)
// Add the remaining vectors to the final reduced dataset
foreach {v_j|v_j ∈ 𝒱} do
    ℛ.add(v_j)
```

Fig. 5. QT Clustering based dataset reduction algorithm.

The average processing time consumed to reduce the datasets (Fig. 6) for each configuration is 1107 s for LingSpam, 3302 s for SpamAssassin and 15,235 s for TREC when using Euclidean distance and 751 s for LingSpam, 2179 s for SpamAssassin and 10,236 s for TREC when using Manhattan distance. This process, despite being time consuming, is executed only once and does not interfere with system performance.

Table 6
Number of vectors conforming the reduced datasets for the different reduction thresholds.

Distance measure	Quality threshold	% Average reduction	Vectors per fold				
			1	2	3	4	5
LingSpam							
Euclidean	1.50	13.21%	1647	1646	1674	1688	1718
	1.75	57.10%	800	802	817	848	871
	2.00	89.72%	184	184	191	212	220
	∞	99.94%	1	1	1	1	1
Manhattan	1.50	33.75%	1318	1322	1296	1223	1232
	1.75	46.78%	1079	1047	1051	979	978
	2.00	62.47%	769	749	750	673	679
	∞	99.94%	1	1	1	1	1
SpamAssassin							
Euclidean	1.50	89.78%	302	342	431	297	324
	1.75	97.63%	66	79	102	66	79
	2.00	99.34%	16	18	33	20	21
	∞	99.96%	1	1	1	1	1
Manhattan	1.50	93.59%	119	230	251	221	242
	1.75	96.81%	50	117	132	109	121
	2.00	98.57%	17	53	60	52	54
	∞	99.96%	1	1	1	1	1
TREC							
Euclidean	0.25	56.16%	2673	2671	2663	2685	2729
	0.50	74.81%	1531	1504	1528	1577	1570
	1.00	98.28%	108	104	105	108	103
	∞	99.98%	1	1	1	1	1
Manhattan	0.50	98.06%	125	124	117	107	121
	1.00	99.95%	3	3	3	3	3
	1.25	99.94%	3	3	4	3	4
	∞	99.98%	1	1	1	1	1

The times do not vary considerably among the different thresholds used for each distance measure because the operations that take a higher processing overhead are the distance measure calculations, and the algorithm proposed in Fig. 5 calculates all distances between points before starting the clustering step. Consequently, the data reduction algorithm performs the same heavy calculations independent of the threshold specified.

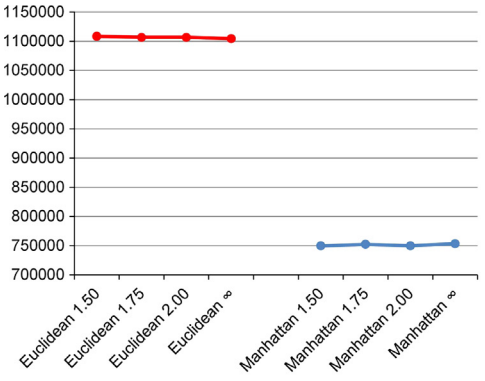
- **Sample comparison.** In this phase, for each experimental configuration employed in the data reduction stage, the samples were compared to the reduced dataset.
  The number of comparisons depends exclusively on the number of vectors present in the resulting datasets, so the time employed in this step is inversely proportional to the threshold value used in the clustering algorithm.
  Fig. 7 shows the average time employed by the comparison step for each testing sample. The time required for comparison is lower when utilising fewer vectors. For Euclidean distance the average comparison time varies from 494.53 ms for a 1.50 clustering threshold value, to 0.46 ms for an ∞ threshold (comparison against a single vector representation) with LingSpam, from 121.07 ms for a 1.50 threshold to 0.35 for an ∞ threshold with SpamAssassin and from 1063.56 ms for a 0.25 threshold to 0.41 for an ∞ threshold with TREC. In Manhattan distance, times are lower due to the simplicity of the calculations needed, varying from 257.55 ms, 56.22 ms and 32.11 ms to 0.30 ms, 0.24 and 0.34 with LingSpam, SpamAssassin and TREC respectively. Compared to LingSpam, SpamAssassin and TREC present lower comparison times despite their larger sizes. The increased reduction suffered by the datasets causes this difference (as in Table 6).

Hereafter, we obtained the representation of the emails from all three datasets, reduced the dataset using the 2 different distance measures and 4 different threshold values (resulting into 16 different reduced datasets) and employed the same 2 different measures and the 3 combination rules described in Section to test the datasets and obtain a final measure of deviation for each testing sample.

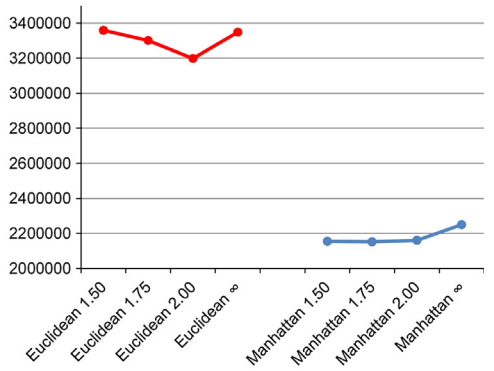
For each measure and combination rule, we established 10 different thresholds to determine whether an email was spam and selected the one that resulted in the best results in each case.

We evaluated the results by measuring the FNR, FPR, WA and AUC. Tables 7–9 show the obtained results. To simplify the results presented, we only show the performance associated with the best threshold for each configuration.

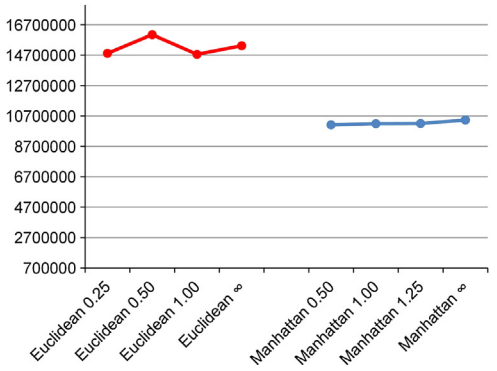
Our anomaly-based spam filtering system is able to correctly detect over 92% of junk mails while maintaining a rate of misclassified legitimate emails below 6% with the best configuration tested with LingSpam (Euclidean distance, 1.75 threshold and Mean rule).



(a) Time required (in ms) to reduce the original dataset of LingSpam.



(b) Time required (in ms) to reduce the original dataset of SpamAssassin



(c) Time required (in ms) to reduce the original dataset of TREC

Fig. 6. The *X* axis shows the different experimental configurations selected for the data reduction step. The *Y* axis shows the time required by each clustering process performed, expressed in ms.

For SpamAssassin, the WA approaches 90% with fewer than 7% misclassified legitimate emails (Manhattan distance, 2.00 threshold and Min rule). Finally, for the TREC dataset, we obtain the worst results with only a 74.82% WA and an improvable 9.19% misclassified legitimate emails (Euclidean distance, 1.75 threshold and Min rule).

The Min combination rule achieved the best results for SpamAssassin and TREC, while the Mean combination rule was the best for LingSpam (though this can be discussed, as the Min combination rule produces fewer misclassified legitimate emails while maintaining a similar WA).

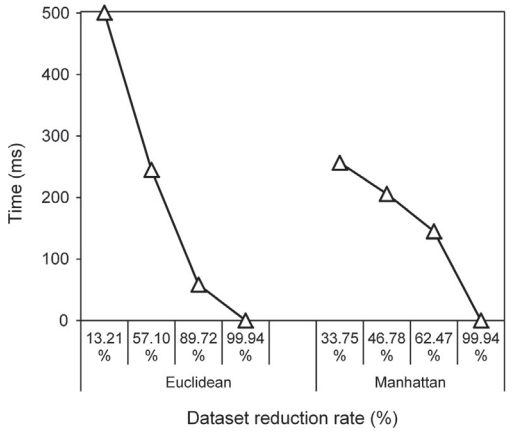
Figs. 8–10 show different plots for each different distance measure and selection rule. In each plot are 4 ROC curves corresponding to the 4 different reduced datasets. In some cases, the ROC curve shows better results when the threshold employed for reduction is ∞ (thus, the number of vectors to compare is only 1).

Fig. 11(a)–(c) represents the evolution of the configurations for the different reduction rates. Regarding Min and Max selection rules, as the number of vectors diminishes, the system loses accuracy for SpamAssassin and TREC but maintains it for LingSpam. Nevertheless, when the samples were compared to the mean vector, the results improve for all three datasets.

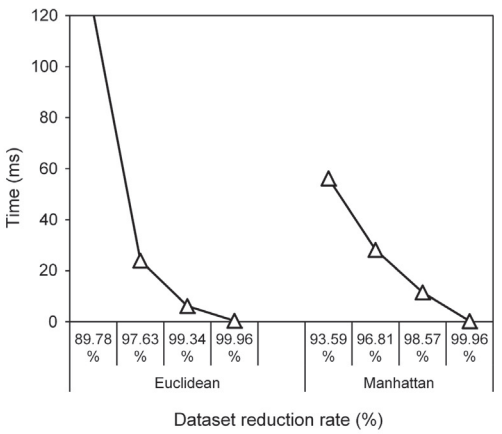
This behaviour is more noticeable for Max selector because it is more sensitive to groups of vectors distant from the normality representation, which can have a negative effect because it alters the distance value. Conversely, the Min selector achieved the best results in almost all cases. This fact, as stated above, highlights a possible discussion topic regarding what should be called an anomaly in emails.

7. Representation of normality: legitimate vs. spam

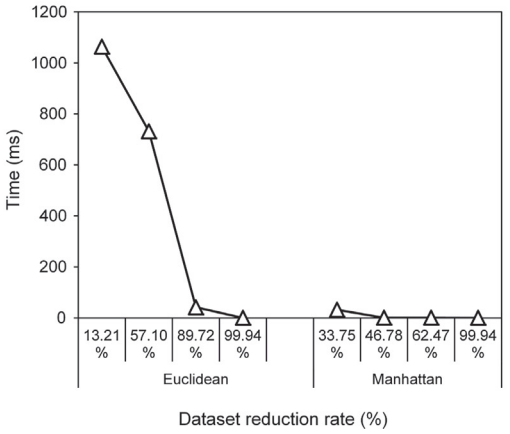
To evaluate the suitability of choosing spam as anomaly, we performed several experiments using spam to represent normality to compare against the results obtained with the approach that represents normality using legitimate messages. The



(a) Time required by the comparison phase for each reduced dataset of LingSpam



(b) Time required by the comparison phase for each reduced dataset of SpamAssassin



(c) Time required by the comparison phase for each reduced dataset of TREC

Fig. 7. The *X* axis represents the resulting reduction rate for each dataset after applying the clustering step. The bigger the reduction rate, the lower the number of vectors utilised. The *Y* axis represents the average comparison time for each e-mail, expressed in ms.

experiments were conducted following the methods described in Sections 3 and 5, only this time, spam was the normal behaviour and legitimate emails were the anomaly.

Again, we used the LingSpam, SpamAssassin and TREC 2007 Public Corpus datasets and followed the same configuration detailed above (Section 4).

For the LingSpam dataset, we performed a 5-fold cross-validation [21] by dividing the dataset of spam emails (the normal behaviour) into 5 different divisions of 96 messages, using 4 to represent normality and 1 to measure deviations. Each fold thus contains 384 spam emails that represent normality and 2508 testing emails, from which 96 were spam and 2412 were legitimate emails.

For the SpamAssassin dataset, we also performed a 5-fold cross-validation. Each fold contains 1517 or 1516 spam emails that represent normality and 4530 or 4529 testing emails, from which 380 or 379 were spam emails and 4150 were legitimate emails.

In the TREC dataset, the 5-fold cross-validation divided spam emails into 3 different divisions of 11,978 emails and 2 divisions of 11,979 emails to represent normality. Each fold has 11,979 or 11,978 spam emails that represent normality and 10,647 or 10,648 testing emails, from which 2994 or 2995 were spam emails and 7653 were legitimate emails (Table 10).

Calculation of the different distances and combination rules, the definition of the thresholds and the way to test the method are described in Section 4.

Table 7
Results for the different reduced datasets of LingSpam, combination rules and distance measures.

LingSpam						
Distance	QT	Selection rule	Threshold	FNR (%)	FPR (%)	WA (%)
Euclidean	Without reduction	Mean	2.59319	8.42	7.05	92.27
		Max	4.15651	20.71	12.89	83.20
		Min	1.93707	6.88	13.23	89.95
	1.50	Mean	2.61855	9.04	6.80	92.08
		Max	4.15651	20.71	12.89	83.20
		Min	2.01180	9.71	6.47	91.91
	1.75	Mean	2.72416	9.75	5.80	92.22
		Max	4.15651	20.71	12.89	83.20
		Min	2.05017	11.71	5.02	91.64
	2.00	Mean	2.91093	11.21	5.14	91.83
		Max	4.15647	20.71	12.89	83.20
		Min	2.08618	14.75	4.10	90.57
	∞	Mean	2.11057	16.33	3.65	90.01
		Max	2.11057	16.33	3.65	90.01
		Min	2.11057	16.33	3.65	90.01
Manhattan	Without reduction	Mean	4.04255	37.71	5.10	78.60
		Max	5.81974	36.54	8.25	77.60
		Min	2.59853	44.58	10.57	72.42
	1.50	Mean	3.97401	26.17	13.06	80.39
		Max	5.81974	36.54	8.25	77.60
		Min	2.91339	29.00	9.91	80.55
	1.75	Mean	4.08539	26.08	10.41	81.76
		Max	5.81974	36.54	8.25	77.60
		Min	3.05884	28.83	6.67	82.25
	2.00	Mean	4.22296	25.50	7.92	83.29
		Max	5.81974	36.54	8.25	77.60
		Min	3.20269	27.67	5.14	83.60
	∞	Mean	3.58608	24.33	10.95	82.36
		Max	3.58608	24.33	10.95	82.36
		Min	3.58608	24.33	10.95	82.36

Table 8
Results for the different reduced datasets of SpamAssassin, combination rules and distance measures.

SpamAssassin						
Distance	QT	Selection rule	Threshold	FNR (%)	FPR (%)	WA (%)
Euclidean	Without reduction	Mean	2.13512	18.96	25.88	77.58
		Max	3.83970	44.58	8.70	73.36
		Min	1.39962	6.00	18.41	87.79
	1.50	Mean	2.27873	22.34	22.60	77.53
		Max	3.66095	34.07	20.48	72.72
		Min	1.47257	12.45	9.04	89.26
	1.75	Mean	2.43757	24.67	22.51	76.41
		Max	3.64738	32.34	22.51	72.58
		Min	1.50019	13.70	10.02	88.14
	2.00	Mean	2.54008	19.75	34.80	72.73
		Max	3.58818	22.95	34.24	71.40
		Min	1.51793	14.12	14.87	85.50
	∞	Mean	1.55007	15.63	30.43	76.97
		Max	1.55007	15.63	30.43	76.97
		Min	1.55007	15.63	30.43	76.97
Manhattan	Without reduction	Mean	2.44136	7.15	20.89	85.98
		Max	5.29903	48.82	12.14	69.52
		Min	1.37525	7.48	9.25	91.63
	1.50	Mean	3.01706	8.21	18.94	86.43
		Max	5.29349	51.31	11.08	68.80
		Min	2.07288	17.03	8.65	87.16
	1.75	Mean	3.14866	8.14	24.05	83.90
		Max	5.29349	56.17	10.99	66.42
		Min	2.07288	10.52	13.04	88.22
	2.00	Mean	3.52854	19.18	13.49	83.66
		Max	4.97643	31.71	40.58	63.86
		Min	2.33412	13.86	6.70	89.72
	∞	Mean	2.37407	6.73	19.08	87.09
		Max	2.37407	6.73	19.08	87.09
		Min	2.37407	6.73	19.08	87.09

Table 9
Results for the different reduced datasets of TREC, combination rules and distance measures.

TREC						
Distance	QT	Selection rule	Threshold	FNR (%)	FPR (%)	WA (%)
Euclidean	Without reduction	Mean	2.63433	78.66	12.41	54.46
		Max	4.19129	78.60	15.71	52.85
		Min	0.53782	42.93	8.36	74.35
	1.50	Mean	2.47451	78.66	12.41	54.46
		Max	4.19129	78.60	15.71	52.85
		Min	0.53782	42.65	8.49	74.43
	1.75	Mean	2.49432	78.66	12.41	54.46
		Max	4.19129	78.60	15.71	52.85
		Min	0.53782	41.17	9.19	74.82
	2.00	Mean	2.43508	78.62	12.41	54.48
		Max	4.13039	78.53	16.39	52.54
		Min	0.76720	65.10	14.05	60.43
	∞	Mean	2.33679	78.66	12.41	54.46
		Max	2.33679	78.66	12.41	54.46
		Min	2.33679	78.66	12.41	54.46
Manhattan	Without reduction	Mean	1.19233	64.38	18.48	58.57
		Max	2.85569	73.13	12.45	57.21
		Min	0.08532	46.62	24.85	64.26
	1.50	Mean	1.53588	71.45	12.70	57.92
		Max	2.69551	73.05	12.99	56.98
		Min	0.46470	68.50	8.35	61.58
	1.75	Mean	1.51117	55.66	24.62	59.86
		Max	2.37515	67.35	18.80	56.92
		Min	0.96223	83.03	0.80	58.09
	2.00	Mean	1.54777	56.32	28.83	57.43
		Max	2.69551	73.05	12.99	56.98
		Min	1.02516	85.68	0.54	56.89
	∞	Mean	1.18801	64.43	17.30	59.14
		Max	1.18801	64.43	17.30	59.14
		Min	1.18801	64.43	17.30	59.14

7.1. Results of anomaly-based approach without dataset reduction and spam as normality

Table 11 shows the best results obtained for each testing dataset (i.e., LingSpam, SpamAssassin and TREC) using different distances, combination rules and thresholds. The best results were offered for all three datasets by the Euclidean Distance with the Min combination rule: 84.58% FNR, 0.43% FPR and 57.49% WA for LingSpam; 59.49% FNR, 1.47% FPR and 69.52% WA for SpamAssassin; and 10.53% FNR, 32.01% FPR and 78.73% WA for TREC.

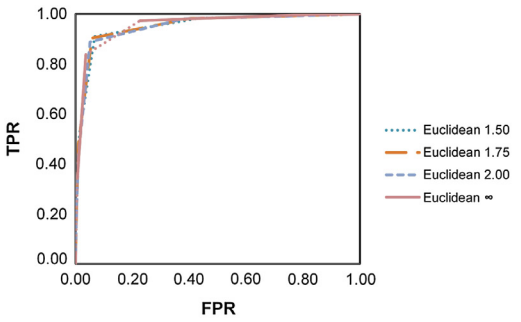
7.2. Results of anomaly-based approach with dataset reduction and spam as normality

Table 12 shows the best results obtained for each testing dataset (i.e., LingSpam, SpamAssassin and TREC). Only configurations with the Min selection rule are shown, as they outperformed both the Mean and Max rules in all cases.

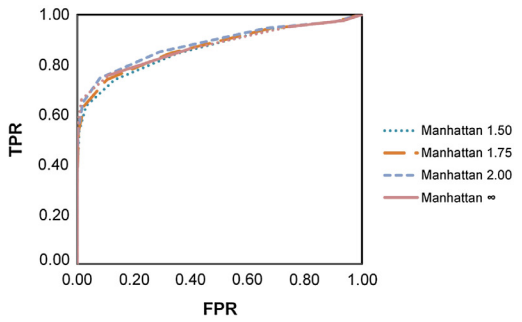
In the table, we also include the best results of the approach without the clustering step for comparison purposes. These results confirm that using our proposed dataset reduction does not diminish detecting capabilities but, in fact, could increase them (at least one configuration for all three datasets outperforms the results obtained without reduction). The best results were obtained with Manhattan distance for LingSpam and SpamAssassin and with Euclidean distance for TREC: 71.04% FNR, 8.70% FPR and 60.13% WA for LingSpam; 37.29% FNR, 22.69% FPR and 70.01% WA for SpamAssassin; and 9.60% FNR, 30.69% FPR and 79.85% WA for TREC.

8. Comparison of the approaches

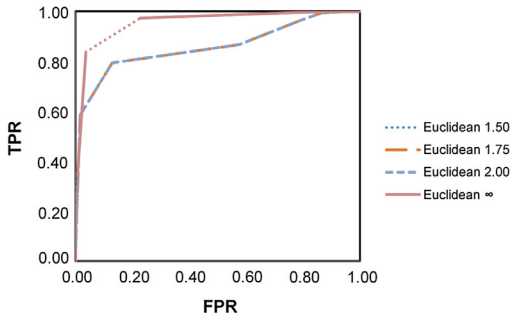
Table 13 compares the best results obtained for each approach, considering spam or legitimate emails as anomaly, and the different tested configurations. In summary, the Min selection rule always provided the best results, Euclidean distance is the best in our experiments, and finally, regarding the suitability of using legitimate emails or spam to represent normality, we obtain different readings. On one hand, for the LingSpam and SpamAssassin datasets, the best approach uses legitimate emails to represent normality, thus considering spam as anomaly. On the other hand, for the TREC dataset, we obtain better results when considering legitimate emails as anomaly. This is a consequence of the different types of emails contained within each dataset, with TREC the most current, complete and heterogeneous dataset of the three, clearly showing the importance of the nature of the normality dataset.



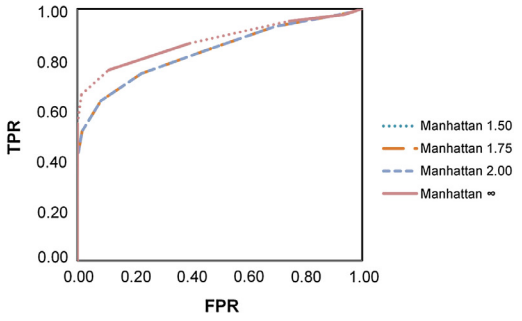
(a) ROC curve for the Euclidean distance and Mean selector.



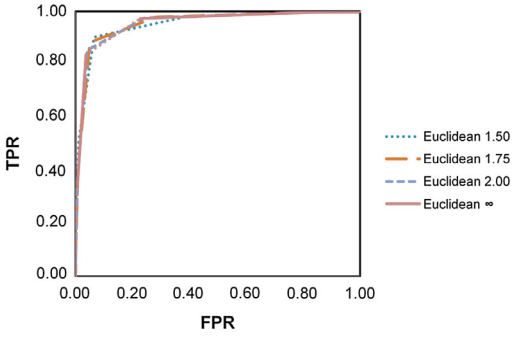
(b) ROC curve for the Manhattan distance and Mean selector.



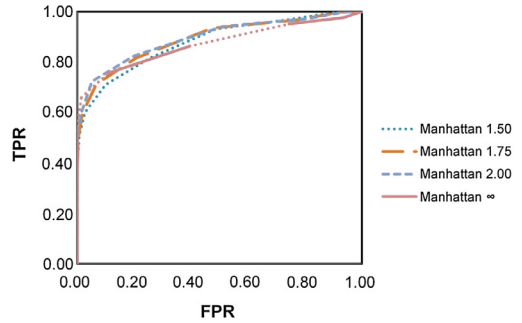
(c) ROC curve for the Euclidean distance and Max selector.



(d) ROC curve for the Manhattan distance and Max selector.



(e) ROC curve for the Euclidean distance and Min selector.



(f) ROC curve for the Manhattan distance and Min selector.

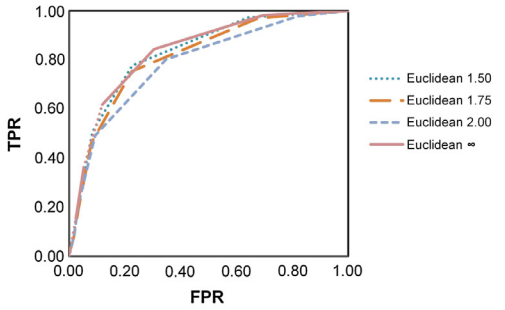
Fig. 8. ROC curves for the different experimental configurations applied to LingSpam. Each figure shows 4 ROC curves corresponding to the different reduced datasets.

9. Discussion

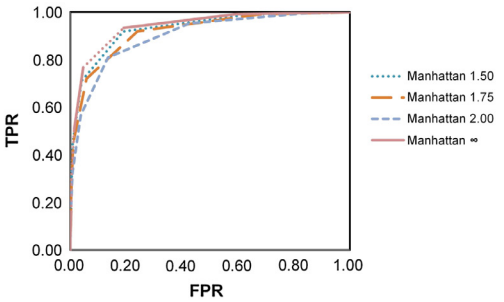
The final results show that this method achieves high accuracy levels, minimising the labelling efforts with a dataset of only one class of emails: legitimate or spam. Nevertheless, several discussion points arise regarding the suitability of the proposed method.

The VSM on which this method relies assumes that every term is independent, which is, at least from a linguistic point of view, not completely true. Though emails are usually represented as a sequence of words, there are relationships between words on a semantic level that also affect emails [11]. Specifically, we can find several linguistic phenomena in natural languages [31]:

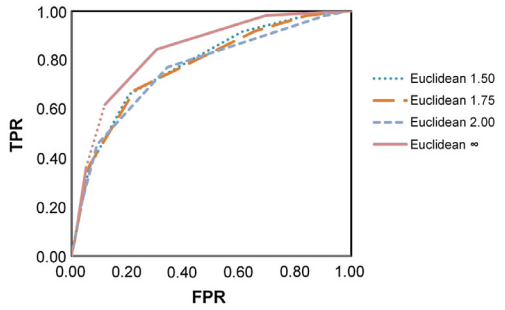
- **Synonyms:** Two or more words are interchangeable because of their similar (or identical) meaning (e.g., 'buy' and 'purchase') [8].



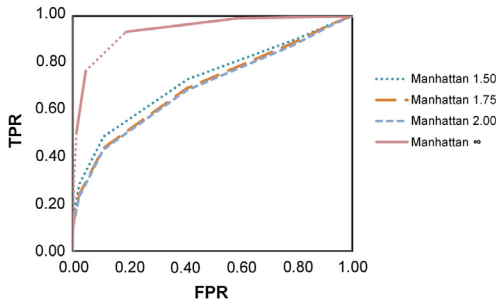
(a) ROC curve for the Euclidean distance and Mean selector.



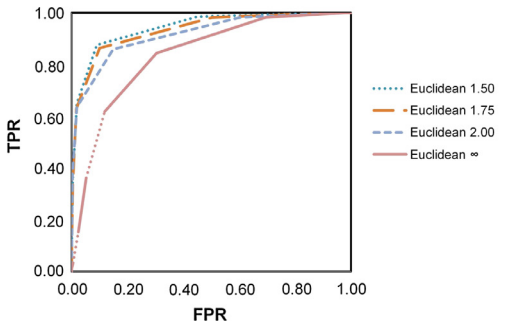
(b) ROC curve for the Manhattan distance and Mean selector.



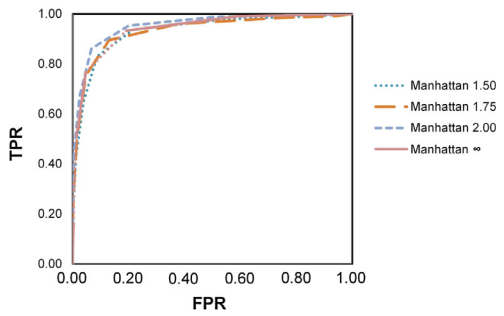
(c) ROC curve for the Euclidean distance and Max selector.



(d) ROC curve for the Manhattan distance and Max selector.



(e) ROC curve for the Euclidean distance and Min selector.



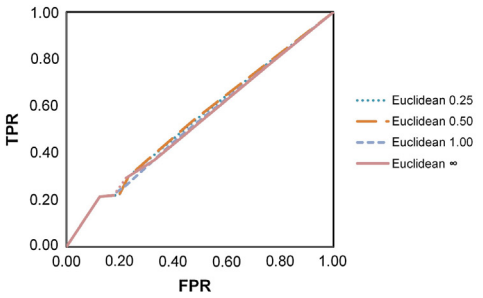
(f) ROC curve for the Manhattan distance and Min selector.

Fig. 9. ROC curves for the different experimental configurations applied to SpamAssassin. Each figure shows 4 ROC curves corresponding to the different reduced datasets.

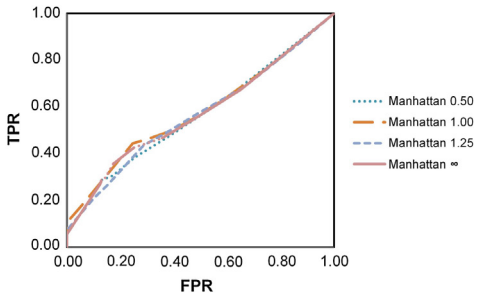
- **Hyponyms:** Specific instances of a more general word (e.g., 'spicy' and 'salty' are hyponyms of 'flavour')[13].
- **Metonymy:** The substitution of one word for another with which it is associated (e.g., 'police' instead of 'law enforcement') [32].
- **Homography:** Words with the same orthography but different meaning (e.g., 'bear': 'to support and carry' and 'an animal') [29].
- **Word-groups:** Clusters of words that have semantic meaning when they are grouped (e.g., 'New York City').

Therefore, our representation cannot handle the existing linguistic phenomena in natural languages [5]. In fact, attacks exist that evade spam filtering systems with synonyms [19], which our model cannot defeat.

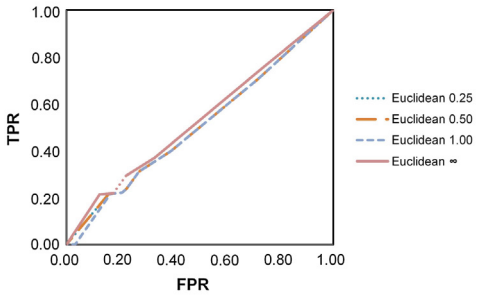
As a solution, researchers have recently proposed the Topic-based Vec- tor Space Model (TVSM) [5] and the enhanced Topic-based Vector Space Model (eTVSM) [23]. The TVSM represents documents using a vector-representation where axes are topics rather than terms and terms are therefore weighted based upon how strongly related they are to a topic. Conversely, the eTVSM uses ontology to represent the different relations between terms and, in this way, provides a richer nat-



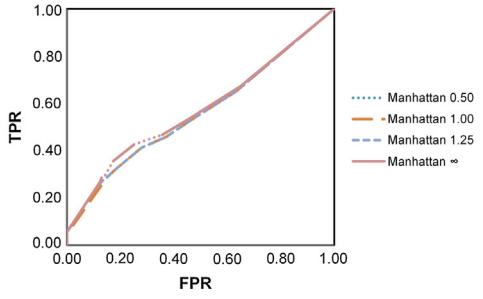
(a) ROC curve for the Euclidean distance and Mean selector.



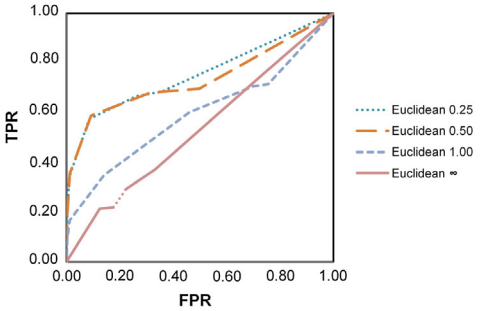
(b) ROC curve for the Manhattan distance and Mean selector.



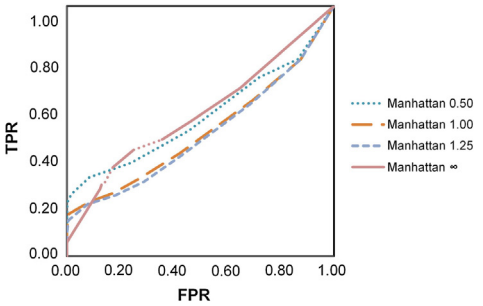
(c) ROC curve for the Euclidean distance and Max selector.



(d) ROC curve for the Manhattan distance and Max selector.



(e) ROC curve for the Euclidean distance and Min selector.



(f) ROC curve for the Manhattan distance and Min selector.

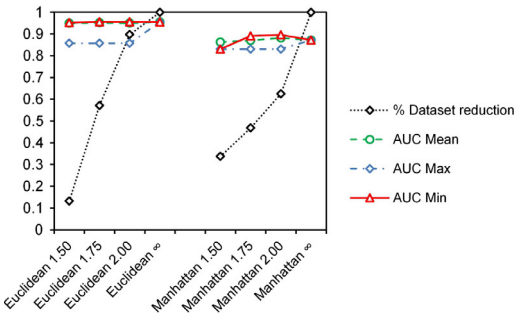
Fig. 10. ROC curves for the different experimental configurations applied to TREC. Each figure shows 4 ROC curves corresponding to the different reduced datasets.

ural language retrieval model that can accommodate synonyms, homonyms and other linguistic phenomena [2]. Using eTVSM for spam filtering has already been proposed [38], but further study on combining anomaly detection and eTVSM could offer interesting results.

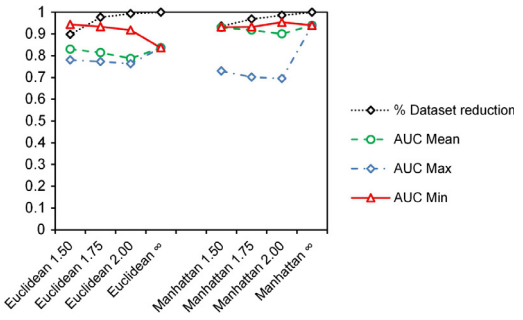
A problem also derives from IR and Natural Language Processing (NLP) when dealing with semantics: Word Sense Disambiguation (WSD), which is considered necessary to accomplish most NLP tasks [17]. A spammer may evade our method by explicitly exchanging key words in an email with other polyseme terms. In a previous work we explored the use of semantics in spam filtering by introducing a pre-processing step of WSD [24], but, as happens with eTVSM, combining both techniques could be interesting. Nevertheless, a semantic approach for spam filtering must handle the semantics of different languages [4] and thus be language-dependant.

Our method also has several limitations due to the representation of emails. Because most spam filtering techniques are based on the frequencies that terms appear within messages, spammers have started modifying their techniques to evade filters.

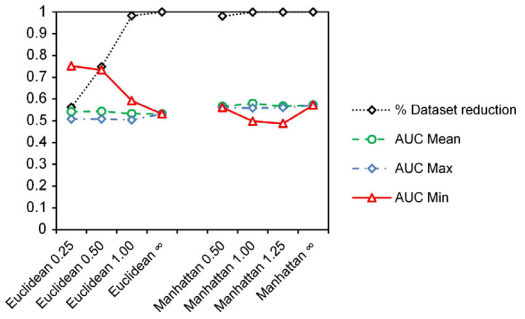
For example, Good Word Attack is a method that modifies term statistics by appending a set of words that are characteristic of legitimate emails, thereby bypassing spam filters. To overcome this problem, we could adopt some proposed



(a) Dataset reduction rate and the accuracy achieved with each reduced dataset of LingSpam.



(b) Dataset reduction rate and the accuracy achieved with each reduced dataset of SpamAssassin.



(c) Dataset reduction rate and the accuracy achieved with each reduced dataset of TREC.

Fig. 11. The Dataset reduction line represents the increasing reduction rate (the higher the rate, the lower the number of samples in the reduced dataset), while the other lines represent the AUC obtained with each reduced dataset.

Table 10
Number of instances within each fold of the 5-fold cross-validation process. The number of spam emails within SpamAssassin and TREC varied in the folds because the number of spam emails was not divisible by 5.

	Normality	Deviations	
	# Spam	# Spam	# Legit.
LingSpam			
Fold 1	384	96	2412
Fold 2	384	96	2412
Fold 3	384	96	2412
Fold 4	384	96	2412
Fold 5	384	96	2412
SpamAssassin			
Fold 1	1516	380	4150
Fold 2	1517	379	4150
Fold 3	1517	379	4150
Fold 4	1517	379	4150
Fold 5	1517	379	4150
TREC			
Fold 1	11,979	2994	7653
Fold 2	11,979	2994	7653
Fold 3	11,978	2995	7653
Fold 4	11,978	2995	7653
Fold 5	11,978	2995	7653

Table 11
Best results obtained for different combination rules and distance measures with our anomaly-based approach without the clustering step when using spam to represent normality, thus considering legitimate emails as anomalous behaviour. 'Thres.' stands for the chosen threshold.

	Manhattan distance				Euclidean distance			
Comb.	Thres.	FNR (%)	FPR (%)	WA (%)	Thres.	FNR (%)	FPR (%)	WA (%)
LingSpam								
Mean	2.96003	100.00	0.00	50.00	2.34311	100.00	0.00	50.00
Max	9.04422	32.08	65.15	51.38	4.72080	64.38	25.68	54.97
Min	0.70134	87.50	2.52	54.99	1.28991	84.58	0.43	57.49
Mean	2.44163	100.00	0.00	50.00	1.97276	100.00	0.00	50.00
Max	4.91811	100.00	0.00	50.00	4.43060	43.41	19.75	68.42
Min	0.40784	76.90	1.60	60.75	1.27859	59.49	1.47	69.52
Mean	0.69536	100.00	0.00	50.00	1.73610	44.18	52.15	51.46
Max	2.69006	99.93	0.00	50.04	3.77008	66.15	32.59	50.63
Min	0.00000	29.45	20.99	74.78	0.43639	10.53	32.01	78.73

Table 12
Best results obtained for different combination rules and distance measures with our anomaly-based approach applying the dataset reduction step when using spam to represent normality, hence considering legitimate emails as an anomalous behaviour. 'Thres.' stands for the chosen threshold.

Distance	QT	Selection rule	Thres.	FNR (%)	FPR (%)	WA (%)
LingSpam						
Euclidean	Without reduction	Min	1.28991	84.58	0.43	57.49
	0.25	Min	1.28991	84.58	0.43	57.49
	1.00	Min	1.28991	84.58	0.43	57.49
	2.00	Min	1.44940	83.33	2.84	56.91
	∞	Min	1.28257	99.79	0.20	50.01
Manhattan	Without reduction	Min	0.70134	87.50	2.52	54.99
	0.25	Min	0.70134	87.50	2.52	54.99
	1.00	Min	1.69034	71.46	12.91	57.82
	2.00	Min	2.21186	71.04	8.70	60.13
	∞	Min	2.96003	100.00	0.00	50.00
SpamAssassin						
Euclidean	Without reduction	Min	1.27859	59.49	1.47	69.52
	0.25	Min	1.27859	59.49	1.47	69.52
	1.00	Min	1.27859	59.55	1.81	69.32
	2.00	Min	1.07816	86.71	2.33	55.48
	∞	Min	0.83543	100.00	0.00	50.00
Manhattan	Without reduction	Min	0.40784	76.90	1.60	60.75
	0.25	Min	0.40784	77.37	1.60	60.51
	1.00	Min	0.82545	69.41	1.46	64.57
	2.00	Min	2.06302	37.29	22.69	70.01
	∞	Min	0.69476	100.00	0.00	50.00
TREC						
Euclidean	Without reduction	Min	0.43639	10.53	32.01	78.73
	0.25	Min	0.46411	8.86	33.18	78.98
	0.50	Min	0.46416	9.60	30.69	79.85
	0.75	Min	0.69692	4.05	51.70	72.12
	∞	Min	1.03371	68.18	27.30	52.26
Manhattan	Without reduction	Min	0.00000	29.45	20.99	74.78
	0.25	Min	0.31199	3.41	61.78	67.40
	0.50	Min	0.49392	2.83	68.73	64.22
	0.75	Min	0.70688	2.53	76.74	60.37
	∞	Min	0.69476	100.00	0.00	50.00

methods to improve spam filtering, including Multiple In- stance Learning (MIL) [14]. MIL divides an instance or vector in traditional supervised learning methods into several sub-instances and classifies the original vector based on the sub-instances [26]. Zhou et al. [43] proposed adopting MIL for spam filtering by dividing an email into a bag of multiple segments and classifying it as spam if at least one instance in the corresponding bag was spam. Another attack, known as tokenisation, works against the message's feature selection by splitting or modifying key message features, rendering the term-representation no longer feasible [41]. All of these attacks, which spammers have been adopting, should be accounted for when constructing future spam-filtering systems.

Table 13
Best results obtained for the two approaches (i.e., considering spam or legitimate as anomaly) applying both methods (i.e., anomaly detection with and without clustering) for different combination rules and distance measures.

Dataset	Approach	FNR (%)	FPR (%)	WA (%)	Configuration
LingSpam	Spam is anomaly without clustering	8.42	7.05	92.27	Euclidean, Mean and 2.59319 thres.
	Spam is anomaly with clustering	9.75	5.80	92.22	Euclidean, QT of 1.75, Mean and 2.72416 thres.
	Legitimate is anomaly without clustering	84.58	0.43	57.49	Euclidean, Min and 1.28991 thres.
	Legitimate is anomaly with clustering	71.04	8.70	60.13	Manhattan, QT of 2.00, Min and 2.21186 thres.
SpamAssassin	Spam is anomaly without clustering	7.48	9.25	91.63	Manhattan, Min and 1.37525 thres.
	Spam is anomaly with clustering	13.86	6.70	89.72	Manhattan, QT of 2.00, Min and 2.33412 thres.
	Legitimate is anomaly without clustering	59.49	1.47	69.52	Euclidean, Min and 1.27859 thres.
	Legitimate is anomaly with clustering	37.29	22.69	70.01	Manhattan, QT of 2.00, Min and 2.06302 thres.
TREC	Spam is anomaly without clustering	42.93	8.36	74.35	Euclidean, Min and 0.53782 thres.
	Spam is anomaly with clustering	41.17	9.19	74.82	Euclidean, QT of 1.75, Min and 0.53782 thres.
	Legitimate is anomaly without clustering	10.53	32.01	78.73	Euclidean, Min and 0.43639 thres.
	Legitimate is anomaly with clustering	9.60	30.69	79.85	Euclidean, QT of 0.50, Min and 0.46416 thres.

Moreover, the QTs and each selection rule's thresholds were selected through empirical observation. An extensive analysis is mandatory to detect possible optimisations. An automated process could select the best threshold combinations to improve the results of our filtering system.

Finally, despite the behaviour of anomaly based spam filtering, using spam messages to represent normality behaves poorly; the improvements that show this behaviour when testing the TREC dataset, the most heterogeneous dataset of the three used, lead to the conclusion that, in real world environments, legitimate emails should be considered the anomaly. Further research on this assumption should be performed with other recent datasets.

10. Conclusions

Spam is a serious computer security issue that is not only annoying for end-users, but also financially damaging and dangerous to computer security because of the possible spread of other threats like malware or phishing. The classic machine-learning-based spam filtering methods, despite their ability to detect spam, require a time-consuming step of labelling emails.

In this paper, we presented a spam filtering system inspired by anomaly detection systems. Using this method, we can reduce the number of required labelled messages and therefore reduce effort for the filtering industry. To improve the scalability of this method, we also provide an optimisation that, through clustering techniques, reduces the normality dataset, thus reducing the number of comparisons performed when analysing new samples; it also improves the efficiency of the approach while maintaining its efficacy.

Future versions of this spam filtering system will move in four main directions. First, we will focus on attacks against statistical spam filtering systems, including tokenisation or Good Word Attacks. Second, we plan to include a semantic-aware layer to this method by supporting several linguistic phenomena and by being able to tackle ambiguity attacks. Third, we will automate the threshold selection process to improve the results of our filtering system. Finally, we will expand the study of what must be considered an anomaly in the email filtering problem by testing with other recent datasets and studying the nature of both approaches to find possible optimisations.

References

[1] I. Androutsopoulos, J. Koutsias, K. Chandrinos, G. Paliouras, C. Spyropoulos, An evaluation of naive bayesian anti-spam filtering, in: Proceedings of the workshop on Machine Learning in the New Information Age, 2000, pp. 9–17.
[2] A. Awad, A. Polyvyanyy, M. Weske, Semantic querying of business process models, in: IEEE International Conference on Enterprise Distributed Object Computing Conference (EDOC 2008), 2008, pp. 85–94.
[3] R.A. Baeza-Yates, B. Ribeiro-Neto, Modern Information Retrieval, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.
[4] M. Bates, R. Weischedel, Challenges in Natural Language Processing, Cambridge Univ Pr., 1993.
[5] J. Becker, D. Kuropka, Topic-based vector space model, in: Proceedings of the 6th International Conference on Business Information Systems, 2003, pp. 7–12.
[6] A. Bratko, B. Filipič, G. Cormack, T. Lynam, B. Zupan, Spam filtering using statistical data compression models, J. Mach. Learn. Res. 7 (2006) 2673–2698.
[7] B. Burton, Spamprobe-bayesian spam filtering tweaks, in: Proceedings of the Spam Conference, 2003.
[8] R. Carnap, Meaning and synonymy in natural languages, Philos. Stud. 6 (1955) 33–47.
[9] P. hirita, J. Diederich, W. Nejdl, MailRank: using ranking for spam detection, in: Proceedings of the 14th ACM International Conference on Information and Knowledge Management, ACM, 2005, pp. 373–380.
[10] Y. Chiu, C. Chen, B. Jeng, H. Lin, An alliance-based anti-spam approach, in: Third International Conference on Natural Computation, 2007, ICNC 2007, IEEE, 2007, pp. 203–207.
[11] D. Cohen, Explaining Linguistic Phenomena, Halsted Press, 1974.
[12] G. Cormack, TREC 2007 spam track overview, in: Sixteenth Text REtrieval Conference (TREC-2007), 2007.
[13] D. Cruse, Hyponymy and lexical hierarchies, Arch. Linguist. 6 (1975) 26–31.
[14] T. Dietterich, R. Lathrop, T. Lozano-Pérez, Solving the multiple instance problem with axis-parallel rectangles, Artif. Intell. 89 (1997) 31–71.

[15] C. Elkan, The foundations of cost-sensitive learning, in: Proceedings of the 2001 International Joint Conference on Artificial Intelligence, 2001, pp. 973–978.
[16] L. Heyer, S. Kruglyak, S. Yooseph, Exploring expression data: identification and analysis of coexpressed genes, Gen. Res. 9 (1999) 1106–1115.
[17] N. Ide, J. Véronis, Introduction to the special issue on word sense disambiguation: the state of the art, Comput. Linguist. 24 (1998) 2–40.
[18] T. Jagatic, N. Johnson, M. Jakobsson, F. Menczer, Social phishing, Commun. ACM 50 (2007) 94–100.
[19] C. Karlberger, G. Bayler, C. Kruegel, E. Kirda, Exploiting redundancy in natural language to penetrate bayesian spam filters, in: Proceedings of the 1st USENIX Workshop on Offensive Technologies (WOOT), USENIX Association, 2007, pp. 1–7.
[20] J. Kent, Information gain and a general measure of correlation, Biometrika 70 (1983) 163.
[21] R. Kohavi, A study of cross-validation and bootstrap for accuracy estimation and model selection, in: International Joint Conference on Artificial Intelligence, 1995, pp. 1137–1145.
[22] V. Kumar, An Introduction to Cluster Analysis for Data Mining, Computer Science Department, University of Minnesota, USA, 2000.
[23] D. Kuropka, Modelle zur Repräsentation natürlichsprachlicher Dokumente-Information-Filtering und-Retrieval mit relationalen Datenbanken, Advan. Inform. Syst. Manage. Sci. 10 (2004).
[24] C. Laorden, I. Santos, B. Sanz, G. Alvarez, P.G. Bringas, Word sense disambiguation for spam filtering, in: Electronic Commerce Research and Applications, 2012.
[25] C. Leung, Z. Liang, An Analysis of the Impact of Phishing and Anti-Phishing Related Announcements on Market Value of Global Firms, HKU Theses Online (HKUTO), 2009.
[26] O. Maron, T. Lozano-Pérez, A framework for multiple-instance learning, Advan. Neural Inform. Process. Syst. (1998) 570–576.
[27] J. Mason, Filtering spam with spamassassin, in: HEANet Annual Conference, 2002.
[28] M. McGill, G. Salton, Introduction to Modern Information Retrieval, McGraw-Hill, 1983.
[29] K. Ming-Tzu, P. Nation, Word meaning in academic English: homography in the academic word list, Appl. Linguist. 25 (2004) 291–314.
[30] N. Mostafa Raad, G. Alam, B. Zaidan, A. Zaidan, Impact of spam advertisement through e-mail: a study to assess the influence of the anti-spam on the e-mail marketing, Afri. J. Bus. Manage. 4 (2010) 2362–2367.
[31] A. Polyvyanyy, Evaluation of a Novel Information Retrieval Model: eTVSM, MSc Dissertation, 2007.
[32] G. Radden, Z. Kövecses, Towards a theory of metonymy, Meton. Lang. Thought (1999) 17–59.
[33] E. Raymond, Bogofilter: A Fast Open Source Bayesian Spam Filters, 2005.
[34] G. Robinson, A statistical approach to the spam problem, Linux J. 2003 (2003) 3.
[35] G. Sakkis, I. Androutsopoulos, G. Paliouras, V. Karkaletsis, C. Spyropoulos, P. Stamatopoulos, A memory-based approach to anti-spam filtering for mailing lists, Inform. Ret. 6 (2003) 49–73.
[36] G. Salton, M. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, New York, 1983.
[37] G. Salton, A. Wong, C. Yang, A vector space model for automatic indexing, Commun. ACM 18 (1975) 613–620.
[38] I. Santos, C. Laorden, B. Sanz, P.G. Bringas, Enhanced topic-based vector space model for semantics-aware spam filtering, Exp. Syst. Appl. 39 (2012) 437–444.
[39] G. Schryen, A formal approach towards assessing the effectiveness of anti-spam procedures, in: Proceedings of the 39th Annual Hawaii International Conference on System Sciences, 2006, HICSS'06, IEEE, 2006, pp. 129–138.
[40] W. Wilbur, K. Sirotkin, The automatic identification of stop words, J. Inform. Sci. 18 (1992) 45–55.
[41] G. Wittel, S. Wu, On attacking statistical spam filters, in: Proceedings of the 1st Conference on Email and Anti-Spam (CEAS), 2004.
[42] L. Zhang, J. Zhu, T. Yao, An evaluation of statistical spam filtering techniques, ACM Trans. Asian Lang. Inform. Process. (TALIP) 3 (2004) 243–269.
[43] Y. Zhou, Z. Jorgensen, M. Inge, Combating good word attacks on statistical spam filters with multiple instance learning, Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence, vol. 02, IEEE Computer Society, 2007, pp. 298–305.

Glossary

AUC: Area Under the Receiver Operating Characteristic Curve
eTVSM: enhanced Topic-based Vector Space Model
FNR: False Negative Ratio
FPR: False Positive Ratio
IF: Information Filtering
IG: Information Gain
IR: Information Retrieval
MIL: Multiple Instance Learning
NLP: Natural Language Processing
QT: Quality Threshold
ROC: Receiver Operating Characteristic
tf-idf: term frequency – inverse document frequency
TPR: True Positive Ratio
TVSM: Topic-based Vector Space Model
VSM: Vector Space Model
WA: Weighted Accuracy
WSD: Word Sense Disambiguation