Spam Filtering through Anomaly Detection

Igor Santos, Carlos Laorden, Xabier Ugarte-Pedrero, Borja Sanz, and
Pablo G. Bringas

S^{3}Lab, DeustoTech - Computing, Deusto Institute of Technology
University of Deusto,
Avenida de las Universidades 24, 48007
Bilbao, Spain
{isantos,claorden,xabier.ugarte,borja.sanz,
pablo.garcia.bringas}@deusto.es

Abstract. More than 85% of received e-mails are spam. Spam is an important issue for computer security because it is used to spread other threats such as computer viruses, worms or phishing. Classic techniques to fight spam, including simple techniques such as sender blacklisting or the use of e-mail signatures, are no longer completely reliable. Machine-learning techniques trained using statistical representations of the terms that usually appear in the e-mails are widely used in the literature. However, these methods demand a time-consuming training step with labelled data. Dealing with the situation where the availability of labelled training instances is limited slows down the progress of filtering systems and offers advantages to spammers. In this paper, we present the first spam filtering method based on anomaly detection that reduces the necessity of labelling spam messages and only uses the representation of legitimate e-mails. This approach represents legitimate e-mails as word frequency vectors. Thereby, an email is classified as spam or legitimate by measuring its deviation to the representation of these legitimate e-mails. This method achieves high accuracy rates detecting spam and maintains a low false positive rate, reducing the effort produced by labelling spam.

Key words: computer security, spam filtering, anomaly detection, text classification

1 Introduction

Electronic mail (e-mail) is a useful communication channel. However, as usually happens with all useful media, it is prone to misuse. In the past decade, spam, or unsolicited bulk e-mail, has become a significant problem for e-mail users: a huge amount of spam arrives in people's mailboxes every day. When this paper was written, 87.6% of all e-mail messages were spam, according to the Spam-o-meter website.[1] Spam not only is very annoying to every-day e-mail users, but also constitutes an important computer security problem that costs billions of

[1] http://www.junk-o-meter.com/stats/index.php

dollars in productivity losses [1]. Moreover, spam can be used as a medium for phishing (i.e., attacks that seek to acquire sensitive information from end-users) [2] or the spread of malicious software (e.g., computer viruses, Trojan horses, spyware and Internet worms) [1].

Because of the magnitude of the spam problem, several spam filtering systems have been proposed by both the academia and the industry. The simplest methods for junk message filtering are often based on blacklisting or signatures [3].

Blacklisting is a very simple technique that is broadly used in most filtering products. More specifically, these systems filter e-mails from certain senders, whereas whitelisting [4] delivers e-mail from specific senders in order to reduce the number of misclassified legitimate e-mails (also known as 'ham' by the spam community). Another popular approach for these so-called banishing methods is based on DNS blacklisting, in which the host address is checked against a list of networks or servers known to distribute spam [5, 6].

In contrast, signature-based systems create a unique hash value (i.e., a message digest) for each known spam message [7]. The main advantage of this type of methods is that they rarely produce false positives and they are usually very fast to compute. Examples of signature-based spam filtering systems are: Cloudmark[2], a commercial implementation of a signature-based filter that integrates with the e-mail server, and Razor[3], another filtering system that uses a distributed and collaborative approach in order to deliver signatures [3].

However, these simple methods have several shortcomings. First, blacklisting methods have a very high rate of false positives, making them unreliable as a standalone solution [8]. Second, signature-based systems are unable to detect spam until the junk message has been identified, properly registered and documented [3].

In order to find a solution to this problem, the research community has undertaken a huge amount of work. Since machine-learning approaches have succeeded in text categorisation problems [9] and spam filtering can be stated as a text categorisation problem, these techniques have been broadly adopted in spam filtering systems.

Consequently, substantial work has been dedicated to the Naïve Bayes classifier [10], with a high number of studies regarding anti-spam filtering confirming it effective [11–15].

Another broadly embraced machine-learning technique is Support Vector Machine (SVM) [16]. The advantage of SVM is that its accuracy does not degrade even when many features are present [17]. Therefore, such approaches have been adopted to junk mail filtering [18, 19]. Likewise, Decision Trees that classify by means of automatically learned rule-sets (i.e., tests) [20] have also been employed for spam filtering [21].

All of these machine-learning-based spam filtering approaches have been termed as statistical approaches because they rely on an statistical represen-

[2] http://www.cloudmark.com

[3] http://razor.sourceforge.net

tation of terms within the messages [22]. Machine-learning approaches model e-mail messages using the Vector Space Model (VSM) [23], an algebraic approach for Information Filtering (IF), Information Retrieval (IR), indexing and ranking. This model represents natural language documents in a mathematical manner through vectors in a multidimensional space formed by the words composing the message.

Machine-learning classifiers require a high number of labelled e-mails for each of the classes (i.e., spam and legitimate e-mails). However, it is quite difficult to obtain this amount of labelled data for a real-world problem such as spam filtering issue. To generate these data, a time-consuming task of analysis is mandatory, and in the process, some spam messages can avoid filtering.

In light of this background, we propose here the first method that applies anomaly detection to spam filtering. This approach is able to determine whether an e-mail is spam or not by comparing word frequency features with a dataset composed only of legitimate e-mails. If the e-mail under inspection presents a considerable deviation to what it is considered as usual (legitimate e-mails), it can be considered spam. This method does not need updated data about spam messages, and thus, it reduces the efforts of labelling messages, working, for instance, only with a user's valid inbox folder.

Summarising, our main findings in this paper are:

- We present an anomaly-based approach for spam filtering.
- We propose different deviation measures to determine whether an e-mail is spam or not.
- 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 the background regarding the representation of e-mails based on the VSM. Section 3 details our anomaly-based method. Section 4 describes the experiments and presents results. Section 5 discusses the implications of the obtained results and shows its limitations. Finally, Section 6 concludes the paper and outlines avenues for future work.

2 Bag of Words to Represent E-mails

Spam filtering software attempts to accurately classify email massages into 2 main categories: spam or legitimate messages. To this end, we use the information found within the body and subject of an e-mail message and discard every other piece of information (e.g., the sender or time-stamp of the e-mail). To represent messages, we start by removing stop-words [24], 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 [25].

Afterwards, we represent the e-mails using an Information Retrieval (IR) model. Formally, let an IR model be defined as a 4-tuple $[\mathcal{E}, \mathcal{Q}, \mathcal{F}, R, (q_i, e_j)]$ [26] where:

- $\mathcal{E}$, is a set of representations of e-mail;
- $\mathcal{Q}$, is a set of representations of user queries;
- $\mathcal{F}$, is a framework for modelling e-mails, 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 e-mail representation e_j, $(e_j \in \mathcal{E})$. This function is also called a similarity function.

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

On the basis of this formalisation, we can apply several IR models. Commonly, spam filtering systems use Vector Space Model (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 i^{th} term and the j^{th} document. This association reflects the occurrence of the i^{th} 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' $(tf - idf)$ [27] to obtain the weight of each word, whereas the weight of the i^{th} word in the j^{th} e-mail, denoted by $weight(i, j)$, is defined by:

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

where the term frequency $tf_{i,j}$ [27] 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 e-mail e, and $\sum_k m_{k,j}$ is the total number of word in the e-mail e.

On the other hand, 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) [28]:

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

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 if a sample is spam or not.

These weights were calculated from two datasets: the LingSpam corpus[4]) composed of 480 spam e-mails and 2,412 legitimate messages and the SpamAssassin corpus[5] composed of 1,896 spam e-mails and 4,150 legitimate spam. These weights are useful to obtain a better distance rating among samples.

3 Anomaly Detection Techniques

Anomaly detection models what it is a normal message and every deviation to this model is considered anomalous. Through the word frequency features of the VSM described in the previous section, our method represents legitimate e-mails as points in the feature space. When an e-mail is being inspected our method starts by computing the values of the point in the feature space. This point is then compared with the previously calculated points of the legitimate e-mails.

To this end, distance measures are required. In this study, we have used the following distance measures:

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

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

  where x is the first point; y is the second point; and x_i and y_i are the ith component of the first and second point, respectively.
- **Euclidean Distance:** This distance is the length of the line segment connecting two points. It is calculated as:

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

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

By means of these measures, we are able to compute the deviations between e-mails and the legitimate e-mails. Since we have to compute this measure with the points representing legitimate e-mails, a combination metric is required in order to obtain a final distance value which considers every measure performed. To this end, our system employs 3 simple metrics:

[4] Available at: http://nlp.cs.aueb.gr/software_and_datasets/lingspam_public.tar.gz

[5] Available at: http://spamassassin.org/publiccorpus

– 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.

In this way, when our method inspects an e-mail a final distance value is acquired, which will depend on both the distance measure and the combination metric.

4 Experimental Results

In order to validate our proposed method, we used the SpamAssasin public corpus and the Ling Spam Corpus.

Table 1. Comparison of the used dataset. The spam ratio in both datasets does not follow the statistics of the number of spam messages in the real world which is higher of the 85%. SpamAssasin dataset, however, has more real spam and examples of obfuscated mails within it.

Feature	SpamAssasin	Ling Spam
No. Spam Messages	1,896	480
No. of Ham Messages	4,150	2,412
Spam %.	31,36%	16,60%

SpamAssassin corpus contains a total of 6,046 messages, of which 1,896 are spam and 4,150 are legitimate e-mails. To adequate the dataset, we performed a stop word removal based on an external stop-word list.[6] Next, we constructed a file with the resultant vector representations of the e-mails in order to validate our method. We extracted the top 1,000 words using Information Gain [28].

Ling Spam consists of a mixture of both spam and legitimate messages retrieved from the Linguistic list, an e-mail distribution list about linguistics. The dataset was preprocessed by removing HTML tags, separation tokens and duplicate e-mails: only the data of the body and the subject were kept. Ling Spam comprises 2,892 different e-mails, of which 2,412 are legitimate e-mails obtained by downloading digests from the list and 480 are spam e-mails retrieved from one of the authors of the corpus (for a more detailed description of the corpus please refer to [13, 29]). Junk messages represent approximately the 16% of the whole dataset, a rate close to the actual rate [30, 31, 29]. Stop Word Removal [24] and stemming [32] were performed on the e-mails, creating 4 different datasets:

1. **Bare:** In this dataset, the e-mail messages were pre-processed by the removal of HTML tags, separation tokens and duplicate e-mails.

[6] Available at: http://paginaspersonales.deusto.es/claorden/resources/EnglishStopWords.txt

2. **Lemm:** In addition to the removal pre-process step, a stemming phase was performed. Stemming reduces inflected or derived words to their stem, base or root form.
3. **Stop:** For this dataset, a stop word removal task was performed. This process removes all stop words (e.g., common words like 'a' or 'the').
4. **Lemm_stop:** This dataset uses the combination of both stemming and stop-word removal processes.

We used the bare dataset and we performed a stop word removal based on the same stop-word list as for SpamAssasin corpus.

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

1. **Cross validation.** For the SpamAssasin dataset, we performed a 5-fold cross-validation [33] to divide the dataset composed of legitimate e-mails (the normal behaviour) into 5 different divisions of 3,320 e-mails for representing normality and 830 for measuring deviations within legitimate e-mails. In this way, each fold is composed of 3,320 legitimate e-mails that will be used as representation of normality and 2,726 testing e-mails, from which 830 are legitimate e-mails and 1,896 are spam.
   With regards to Ling Spam dataset, we also performed a 5-fold cross-validation [33] forming 3 different divisions of 1,930 e-mails and two divisions of 1,929 e-mails for representing normality and other 3 divisions of 482 e-mails and 2 of 483 for measuring deviations within legitimate e-mail. In this way, each fold is composed of 1,930 or 1,929 legitimate e-mails that will be used as representation of normality and 963 or 962 testing e-mails, from which 483 or 482 were legitimate e-mails and 480 were spam. The number of legitimate e-mails varied in the two last folds because the number of legitimate e-mails was not divisible by 5.
2. **Calculating distances and combination rules.** We extracted the aforementioned characteristics and employed the 2 different measures and the 3 different combination rules described in Section 3 to obtain a final measure of deviation for each testing evidence. More accurately, we applied the following distances:
   - Manhattan Distance.
   - Euclidean Distance.

   For the combination rules we have tested the followings:
   - The Mean Value.
   - The Lowest Distance.
   - The Highest Value.
3. **Defining thresholds.** For each measure and combination rule, we established 10 different thresholds to determine whether an email is spam or not. These thresholds were selected by first establishing the lowest one. This number was the highest possible value with which no spam messages were misclassified. The highest one was selected as the lowest possible value with which no legitimate spam messages were misclassified.

In this way, the method is configurable in both reducing false positives or false negatives. It is important to define whether it is better to classify spam as legitimate or to classify legitimate as spam. In particular, one may think that it is more important to detect more spam messages than to minimise false positives. However, 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 can apply two techniques: (i) whitelisting and blacklisting or (ii) cost-sensitive learning. White and black lists store a signature of an e-mail in order to be flagged either as spam (blacklisting) or legitimate messages (whitelisting). On the other hand, cost-sensitive learning is a machine-learning technique where one can specify the cost of each error and the classifiers are trained taking into account that consideration [34]. We can adapt cost-sensitive learning for anomaly detection by using cost matrices.

4. **Testing the method.** To evaluate the results, we measured the most frequently used for spam: precision (Prec.), recall (Rec.) and f-measure (F-meas.). We measured the precision of the spam identification as the number of correctly classified spam e-mails divided by the number of correctly classified spam e-mails and the number of legitimate e-mails misclassified as spam:

$$Precision = \frac{N_{s \to s}}{N_{s \to s} + N_{l \to s}} \tag{7}$$

where $N_{s \to s}$ is the number of correctly classified spam and $N_{l \to s}$ is the number of legitimate e-mails misclassified as spam.

Additionally, we measured the recall of the spam e-mail messages, which is the number of correctly classified spam e-mails divided by the number of correctly classified spam e-mails and the number of spam e-mails misclassified as legitimate:

$$Recall = \frac{N_{s \to s}}{N_{s \to s} + N_{s \to l}} \tag{8}$$

We also computed the F-measure, which is the harmonic mean of both the precision and recall, simplified as follows:

$$F\text{-}measure = \frac{2N_{m \to m}}{2N_{m \to m} + N_{m \to l} + N_{l \to m}} \tag{9}$$

Table 2 shows the obtained results for SpamAssasin corpus for the different distances, combination rules and thresholds. The best results were obtained with the Manhattan Distance, the minimum combination rule and a 1.32493 threshold: a 95.40% of precision, a 93.86% of recall and a 94.62% of f-measure. Table 3 shows the obtained results for 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: the method achieved a 92.82% of precision, a 91.58% of recall and a 92.20% of f-measure.

Table 2. Results for different combination rules and distance measures using Spam Assasin corpus. The abbreviation 'Thres'. stands for the chosen threshold. The results in bold are the best for each combination rule and distance measure. Our method is able to detect more than the 90% of the spam messages while maintaining a high precision (a low number of legitimate messages are misclassified). In particular, the best results were obtained with minimum distance combination rule, the Manhattan distance and a 1.32493 of threshold: a 95.40% of precision, a 93.86% of recall and a 94.62% of f-Measure.

	Manhattan Distance				**Euclidean Distance**			
Combination	**Thres.**	**Prec.**	**Rec.**	**F-Meas.**	**Thres.**	**Prec.**	**Rec.**	**F-Meas.**
Mean	1.15978	69.56%	100.0%	82.05%	1.70013	69.64%	100.00%	82.10%
	1.58697	70.55%	99.86%	82.68%	**1.91763**	**76.14%**	**97.77%**	**85.61%**
	2.01417	79.44%	98.91%	88.11%	2.13512	87.74%	81.04%	84.26%
	2.44136	**91.03%**	**92.85%**	**91.93%**	2.35262	93.56%	56.53%	70.48%
	2.86856	97.01%	76.18%	85.34%	2.57011	94.93%	30.63%	46.32%
	3.29575	98.70%	50.44%	66.76%	2.78761	93.64%	12.57%	22.17%
	3.72295	99.39%	27.62%	43.22%	3.00510	94.44%	6.81%	12.71%
	4.15014	99.22%	14.84%	25.82%	3.22260	95.19%	3.13%	6.07%
	4.57734	99.32%	7.71%	14.31%	3.44009	97.10%	1.41%	2.79%
	5.00453	100.00%	4.55%	8.70%	3.65759	100.00%	0.93%	1.84%
Maximum	3.39114	69.55%	100.00%	82.04%	3.41015	69.67%	100.00%	82.12%
	3.81833	**69.61%**	**99.89%**	**82.05%**	**3.55333**	**72.99%**	**97.66%**	**83.54%**
	4.24553	69.90%	98.50%	81.77%	3.69652	83.86%	82.23%	83.04%
	4.67272	71.69%	91.74%	80.48%	3.83970	93.57%	55.42%	69.61%
	5.09992	83.51%	67.62%	74.73%	3.98288	95.87%	29.86%	45.54%
	5.52711	94.94%	35.43%	51.61%	4.26925	95.33%	6.46%	12.09%
	5.95431	99.56%	14.48%	25.29%	4.12607	94.76%	12.01%	21.33%
	6.38150	100.00%	5.84%	11.04%	4.41243	95.19%	2.92%	5.67%
	6.80870	100.00%	0.96%	1.90%	4.55562	97.18%	1.46%	2.87%
	6.25298	100.00%	7.49%	13.94%	4.69880	100.00%	1.01%	2.01%
Minimum	0.04335	69.61%	100.00%	0.44679	0.44679	69.76%	100.00%	82.18%
	0.47054	74.51%	99.85%	85.34%	0.76440	70.42%	99.92%	82.62%
	0.89774	87.75%	98.89%	92.99%	1.08201	74.92%	99.78%	85.58%
	1.32493	**95.40%**	**93.86%**	**94.62%**	**1.39962**	**92.10%**	**94.00%**	**93.04%**
	1.75213	97.91%	79.88%	87.98%	1.71723	98.74%	68.61%	80.96%
	2.17932	98.92%	54.14%	69.98%	2.03484	99.63%	28.54%	44.38%
	2.60652	99.49%	26.88%	42.32%	2.35245	99.65%	6.02%	11.36%
	3.03371	99.91%	11.43%	20.52%	2.67006	98.88%	1.86%	3.64%
	3.46091	100.00%	3.18%	6.15%	2.98767	98.78%	0.85%	1.69%
	3.04013	100.00%	11.31%	20.32%	3.30528	100.00%	0.36%	0.71%

The fact that the best results were obtained with the minimum distance, which is obviously the most conservative configuration for distance, highlights a possible topic of discussion regarding what should be called anomaly in e-mails. As we aforementioned, currently more than the 85% of the e-mails are spam and, therefore, in terms of normality, receiving a legitimate e-mail is an anomalous.

5 Discussion

The final results show that this method achieves high levels of accuracy. In addition, it can minimise the number of legitimate e-mails that are misclassified and is also able to detect a high number of spam messages. Nevertheless, several points of discussion are important regarding the suitability of the proposed method.

Table 3. Results for different combination rules and distance measures using LingSpam corpus. The abbreviation 'Thres'. means the chosen threshold. The results remarked in bold are the best for each combination rule and distance measure. Using this dataset, our method also can detect more than the 90% of the spam messages whereas maintaining a high precision (a low number of legitimate messages are misclassified). The best results were obtained with the Euclidean Distance, the mean combination rule and 2.59319 as the threshold. In particular, a 92.82% of precision, a 91.58% of recall and a 92.20% of f-measure. These results are a little lower than when using SpamAssasin.

	Manhattan Distance				**Euclidean Distance**			
Combination	**Thres.**	**Prec.**	**Rec.**	**F-Meas.**	**Thres.**	**Prec.**	**Rec.**	**F-Meas.**
Mean	1.86313	49.87%	100.00%	66.56%	1.87061	49.91%	100.00%	66.58%
	2.23637	50.04%	99.79%	66.65%	2.11147	0.50357	99.79%	66.94%
	2.58960	51.02%	97.21%	66.92%	2.35233	68.39%	98.33%	80.67%
	2.95284	56.07%	94.29%	70.32%	**2.59319**	**92.82%**	**91.58%**	**92.20%**
	3.31608	65.86%	84.08%	73.87%	2.83405	97.31%	52.71%	68.38%
	3.67931	**79.18%**	**73.54%**	**76.26%**	3.07490	98.54%	19.75%	32.91%
	4.04255	92,40%	62.29%	74.42%	3.31576	98.33%	7.38%	13.72%
	4.40579	97.89%	52.13%	68.03%	3.55662	98.84%	3.54%	6.84%
	4.76902	99.68%	39.38%	56.45%	3.79748	96.15%	1.04%	2.06%
	5.13226	100.00%	29.88%	46.01%	4.03834	100.00%	0.62%	1.24%
Maximum	3.69053	49.89%	100.00%	66.56%	3.22709	49.93%	100.00%	66.60%
	4.05377	50.08%	99.79%	66.69%	3.41297	50.80%	99.96%	67.37%
	4.41700	51.01%	98.46%	67.21%	3.59886	53.35%	99.33%	69.41%
	4.78024	54.65%	95.00%	69.39%	3.78474	54.89%	96.13%	69.88%
	5.14348	63.06%	85.13%	72.45%	3.97063	59.97%	86.63%	70.87%
	5.50671	**76.23%**	**74.29%**	**75.25%**	**4.15651**	**85.95%**	**79.29%**	**82.49%**
	5.86995	89.49%	61.38%	72.81%	4.34240	97.23%	58.50%	73.05%
	6.23319	99.03%	46.58%	63.36%	4.52828	97.76%	18.17%	30.64%
	6.59642	100.00%	33.08%	49.72%	4.71417	98.08%	6.38%	11.97%
	6.95966	100.00%	20.71%	34.31%	4.90005	100.00%	2.46%	4.80%
Minimum	0.09919	50,29%	100.00%	66.93%	0.69584	50.68%	100.00%	67.26%
	0.46243	51.02%	99.79%	67.52%	1.00615	50.98%	99.79%	67.48%
	0.82566	52.04%	96.88%	67.71%	1.31645	51.95%	99.79%	68.33%
	1.18890	55.16%	94.17%	69.57%	1.62676	59.70%	98.33%	74.30%
	1.55214	58.31%	84.17%	68.89%	**1.93707**	**87.51%**	**93.13%**	**90.23%**
	1.91537	**65.82%**	**74.38%**	**69.84%**	2.24737	97.54%	62.79%	76.40%
	2.27861	74.10%	63.54%	68.42%	2.55768	99.00%	20.67%	34.20%
	2.64185	85.06%	54.79%	66.65%	2.86799	99.42%	7.13%	13.30%
	3.00508	94.66%	43.54%	59.65%	3.17829	96.97%	1.33%	2.63%
	3.84820	100.00%	23.13%	37.56%	3.48860	100.00%	0.29%	0.58%

The VSM assumes that every term is independent, which is, at least from the linguistic point of view, not completely true. Despite the fact that e-mails are usually represented as a sequence of words, there are relationships between words on a semantic level that also affect e-mails [35]. Specifically, we can find several linguistic phenomena in natural languages[36]:

- **Synonyms:** Two or more words are interchangeable because of their similar (or identical) meaning (e.g., 'buy' and 'purchase') [37].
- **Hyponyms:** Specific instances of a more general word (e.g., 'spicy' and 'salty' are hyponyms of 'flavour')[38].
- **Metonymy:** The substitution of one word for another with which it is associated (e.g., 'police' instead of 'law enforcement') [39].
- **Homography:** Words with the same orthography but different meaning (e.g., 'bear': 'to support and carry' and 'an animal') [40].

- **Word-groups:** Clusters of words that have semantic meaning when they are grouped (e.g., ‘New York City’).

Thus, our representation cannot handle the existing linguistic phenomena in natural languages [41]. In fact, attacks exist that evade spam filtering systems through the use of synonyms [42], which our model is not capable of defeating.

As a solution, the *Topic-based Vector Space Model* (TVSM) [41] and the enhanced Topic-based Vector Space Model (eTVSM) [43] have been proposed in the last few years. The TVSM represents documents using a vector-representation where axes are topics rather than terms and, therefore, terms are weighted based upon how strongly related they are to a topic. In contrast, the eTVSM uses an ontology to represent the different relations between terms and, in this way, provides a richer natural language retrieval model that is able to accommodate synonyms, homonyms and other linguistic phenomena [44].

There is also a problem derived from IR and Natural Language Processing (NLP) when dealing with semantics: Word Sense Disambiguation (WSD). A spammer may evade our method by explicitly exchanging the key words of the mail with other polyseme terms and thus avoid detection. WSD is considered necessary in order to accomplish most natural language processing tasks [45]. We propose the study of different WSD techniques (a survey of different WSD techniques can be found in [46]) capable of providing a more semantics-aware spam filtering system. Nevertheless, a semantic approach for spam filtering will have to deal with the semantics of different languages [47] and thus be language-dependant.

Besides, our method has several limitations due to the representation of e-mails. In this way, because most of the spam filtering techniques are based on the frequencies with which terms appear within messages, spammers have started modifying their techniques to evade filters.

For example, Good Word Attack is a method that modifies the term statistics by appending a set of words that are characteristic of legitimate e-mails, thereby bypass spam filters. Nevertheless, we can adopt some of the methods that have been proposed in order to improve spam filtering, such as Multiple Instance Learning (MIL) [48]. MIL divides an instance or a vector in the traditional supervised learning methods into several sub-instances and classifies the original vector based on the sub-instances [49]. Zhou et al. [50] proposed the adoption of multiple instance learning for spam filtering by dividing an e-mail 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 feature selection of the message by splitting or modifying key message features, which renders the term-representation as no longer feasible [51].

All of these attacks, which spammers have been adopting, should be taken into account in the construction of future spam-filtering-systems.

In our experiments, we used a dataset that is very small in comparison to the real-world size. As the dataset size grows, the issue of scalability becomes a concern. This problem produces excessive storage requirements, increases time complexity and impairs the general accuracy of the models [52]. To reduce disproportionate storage and time costs, it is necessary to reduce the size of the original training set [53].

To solve this issue, data reduction is normally considered an appropriate preprocessing optimisation technique [54, 55]. This type of techniques have many potential advantages such as reducing measurement, storage and transmission; decreasing training and testing times; confronting the problem of dimensionality to improve prediction performance in terms of speed, accuracy and simplicity; and facilitating data visualisation and understanding [56, 57]. Data reduction can be implemented in two ways. Instance selection (IS) seeks to reduce the number of evidences (i.e., number of rows) in the training set by selecting the most relevant instances or by re-sampling new ones [58]. Feature selection (FS) decreases the number of attributes or features (i.e., columns) in the training set [59].

It is also important to consider efficiency and processing time. Our system compares each e-mail against a big dataset. Despite Euclidean and Manhattan distances are easy to compute, more time-consuming distance measures like Mahalanobis distance will take too much time to process every e-mail under analysis.

6 Concluding remarks

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, have a very time-consuming step of labelling e-mails.

In this paper, we presented a spam filtering system that is inspired in anomaly detection systems. Using this method, we are able to reduce the number of required labelled messages and, therefore, reduce the efforts for the filtering industry. Our experiments show that this approach provides high percentages of spam detection whilst keeping the number of misclassified legitimate messages low. Besides, this method works only with legitimate e-mails and, therefore, it can be trained using the inbox of a user.

Future versions of this spam filtering system will move in five main directions. First, we will focus on attacks against statistical spam filtering systems such as tokenisation or good word attacks. Second, we plan to include the semantics of this method with more linguistic relationships. Third, we will improve the scalability of the anomaly method in order to reduce the number of distance computations required. Fourth, we will study the feasibility of applying Word Sense Disambiguation techniques to this spam filtering method. Finally, we will deeply investigate in what has to be considered an anomaly in the e-mail filtering problem, comparing whether is better to consider spam or legitimate as an anomalous e-mail.

References

1. Bratko, A., Filipič, B., Cormack, G., Lynam, T., Zupan, B.: Spam filtering using statistical data compression models. The Journal of Machine Learning Research **7** (2006) 2673–2698
2. Jagatic, T., Johnson, N., Jakobsson, M., Menczer, F.: Social phishing. Communications of the ACM **50**(10) (2007) 94–100
3. Carpinter, J., Hunt, R.: Tightening the net: A review of current and next generation spam filtering tools. Computers & security **25**(8) (2006) 566–578
4. Heron, S.: Technologies for spam detection. Network Security **2009**(1) (2009) 11–15
5. Jung, J., Sit, E.: An empirical study of spam traffic and the use of DNS black lists. In: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, ACM New York, NY, USA (2004) 370–375
6. Ramachandran, A., Dagon, D., Feamster, N.: Can DNS-based blacklists keep up with bots. In: Conference on Email and Anti-Spam, Citeseer (2006)
7. Kołcz, A., Chowdhury, A., Alspector, J.: The impact of feature selection on signature-driven spam detection. In: Proceedings of the 1^{st} Conference on Email and Anti-Spam (CEAS-2004). (2004)
8. Mishne, G., Carmel, D., Lempel, R.: Blocking blog spam with language model disagreement. In: Proceedings of the 1^{st} International Workshop on Adversarial Information Retrieval on the Web (AIRWeb). (2005) 1–6
9. Sebastiani, F.: Machine learning in automated text categorization. ACM computing surveys (CSUR) **34**(1) (2002) 1–47
10. Lewis, D.: Naive (Bayes) at forty: The independence assumption in information retrieval. Lecture Notes in Computer Science **1398** (1998) 4–18
11. Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Sakkis, G., Spyropoulos, C., Stamatopoulos, P.: Learning to filter spam e-mail: A comparison of a naive bayesian and a memory-based approach. In: Proceedings of the Machine Learning and Textual Information Access Workshop of the 4^{th} European Conference on Principles and Practice of Knowledge Discovery in Databases. (2000)
12. Schneider, K.: A comparison of event models for Naive Bayes anti-spam e-mail filtering. In: Proceedings of the 10^{th} Conference of the European Chapter of the Association for Computational Linguistics. (2003) 307–314
13. Androutsopoulos, I., Koutsias, J., Chandrinos, K., Paliouras, G., Spyropoulos, C.: An evaluation of naive bayesian anti-spam filtering. In: Proceedings of the workshop on Machine Learning in the New Information Age. (2000) 9–17
14. Androutsopoulos, I., Koutsias, J., Chandrinos, K., Spyropoulos, C.: An experimental comparison of naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages. In: Proceedings of the 23^{rd} annual international ACM SIGIR conference on Research and development in information retrieval. (2000) 160–167
15. Seewald, A.: An evaluation of naive Bayes variants in content-based learning for spam filtering. Intelligent Data Analysis **11**(5) (2007) 497–524
16. Vapnik, V.: The nature of statistical learning theory. Springer (2000)
17. Drucker, H., Wu, D., Vapnik, V.: Support vector machines for spam categorization. IEEE Transactions on Neural networks **10**(5) (1999) 1048–1054
18. Blanzieri, E., Bryl, A.: Instance-based spam filtering using SVM nearest neighbor classifier. Proceedings of FLAIRS-20 (2007) 441–442

19. Sculley, D., Wachman, G.: Relaxed online SVMs for spam filtering. In: Proceedings of the 30^{th} annual international ACM SIGIR conference on Research and development in information retrieval. (2007) 415–422
20. Quinlan, J.: Induction of decision trees. Machine learning **1**(1) (1986) 81–106
21. Carreras, X., Márquez, L.: Boosting trees for anti-spam email filtering. In: Proceedings of RANLP-01, 4th international conference on recent advances in natural language processing, Citeseer (2001) 58–64
22. Zhang, L., Zhu, J., Yao, T.: An evaluation of statistical spam filtering techniques. ACM Transactions on Asian Language Information Processing (TALIP) **3**(4) (2004) 243–269
23. Salton, G., Wong, A., Yang, C.: A vector space model for automatic indexing. Communications of the ACM **18**(11) (1975) 613–620
24. Wilbur, W., Sirotkin, K.: The automatic identification of stop words. Journal of information science **18**(1) (1992) 45–55
25. Salton, G., McGill, M.: Introduction to modern information retrieval. McGraw-Hill New York (1983)
26. Baeza-Yates, R.A., Ribeiro-Neto, B.: Modern Information Retrieval. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1999)
27. McGill, M., Salton, G.: Introduction to modern information retrieval. McGraw-Hill (1983)
28. Kent, J.: Information gain and a general measure of correlation. Biometrika **70**(1) (1983) 163–173
29. Sakkis, G., Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Spyropoulos, C., Stamatopoulos, P.: A memory-based approach to anti-spam filtering for mailing lists. Information Retrieval **6**(1) (2003) 49–73
30. Cranor, L., LaMacchia, B.: Spam! Communications of the ACM **41**(8) (1998) 74–83
31. Sahami, M., Dumais, S., Heckerman, D., Horvitz, E.: A Bayesian approach to filtering junk e-mail. In: Learning for Text Categorization: Papers from the 1998 workshop. Volume 62., Madison, Wisconsin: AAAI Technical Report WS-98-05 (1998) 98–05
32. Lovins, J.: Development of a Stemming Algorithm. Mechanical Translation and Computational Linguistics **11**(1) (1968) 22–31
33. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. In: International Joint Conference on Artificial Intelligence. Volume 14. (1995) 1137–1145
34. Elkan, C.: The foundations of cost-sensitive learning. In: Proceedings of the 2001 International Joint Conference on Artificial Intelligence. (2001) 973–978
35. Cohen, D.: Explaining linguistic phenomena. Halsted Press (1974)
36. Polyvyanyy, A.: Evaluation of a novel information retrieval model: eTVSM (2007) MSc Dissertation.
37. Carnap, R.: Meaning and synonymy in natural languages. Philosophical Studies **6**(3) (1955) 33–47
38. Cruse, D.: Hyponymy and lexical hierarchies. Archivum Linguisticum **6** (1975) 26–31
39. Radden, G., Kövecses, Z.: Towards a theory of metonymy. Metonymy in language and thought (1999) 17–59
40. Ming-Tzu, K., Nation, P.: Word meaning in academic English: Homography in the academic word list. Applied linguistics **25**(3) (2004) 291–314
41. Becker, J., Kuropka, D.: Topic-based vector space model. In: Proceedings of the 6th International Conference on Business Information Systems. (2003) 7–12

42. Karlberger, C., Bayler, G., Kruegel, C., Kirda, E.: Exploiting redundancy in natural language to penetrate bayesian spam filters. In: Proceedings of the 1st USENIX workshop on Offensive Technologies (WOOT), USENIX Association (2007) 1–7
43. Kuropka, D.: Modelle zur Repräsentation natürlichsprachlicher Dokumente-Information-Filtering und-Retrieval mit relationalen Datenbanken. Advances in Information Systems and Management Science **10** (2004)
44. Awad, A., Polyvyanyy, A., Weske, M.: Semantic querying of business process models. In: IEEE International Conference on Enterprise Distributed Object Computing Conference (EDOC 2008). (2008) 85–94
45. Ide, N., Véronis, J.: Introduction to the special issue on word sense disambiguation: the state of the art. Computational linguistics **24**(1) (1998) 2–40
46. Navigli, R.: Word sense disambiguation: a survey. ACM Computing Surveys (CSUR) **41**(2) (2009) 10
47. Bates, M., Weischedel, R.: Challenges in natural language processing. Cambridge Univ Pr (1993)
48. Dietterich, T., Lathrop, R., Lozano-Pérez, T.: Solving the multiple instance problem with axis-parallel rectangles. Artificial Intelligence **89**(1-2) (1997) 31–71
49. Maron, O., Lozano-Pérez, T.: A framework for multiple-instance learning. Advances in neural information processing systems (1998) 570–576
50. Zhou, Y., Jorgensen, Z., Inge, M.: Combating Good Word Attacks on Statistical Spam Filters with Multiple Instance Learning. In: Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence-Volume 02, IEEE Computer Society (2007) 298–305
51. Wittel, G., Wu, S.: On attacking statistical spam filters. In: Proceedings of the 1st Conference on Email and Anti-Spam (CEAS). (2004)
52. Cano, J., Herrera, F., Lozano, M.: On the combination of evolutionary algorithms and stratified strategies for training set selection in data mining. Applied Soft Computing Journal **6**(3) (2006) 323–332
53. Czarnowski, I., Jedrzejowicz, P.: Instance reduction approach to machine learning and multi-database mining. In: Proceedings of the Scientific Session organized during XXI Fall Meeting of the Polish Information Processing Society, Informatica, ANNALES Universitatis Mariae Curie-Skłodowska, Lublin. (2006) 60–71
54. Pyle, D.: Data preparation for data mining. Morgan Kaufmann (1999)
55. Tsang, E., Yeung, D., Wang, X.: OFFSS: optimal fuzzy-valued feature subset selection. IEEE transactions on fuzzy systems **11**(2) (2003) 202–213
56. Torkkola, K.: Feature extraction by non parametric mutual information maximization. The Journal of Machine Learning Research **3** (2003) 1415–1438
57. Dash, M., Liu, H.: Consistency-based search in feature selection. Artificial Intelligence **151**(1-2) (2003) 155–176
58. Liu, H., Motoda, H.: Instance selection and construction for data mining. Kluwer Academic Pub (2001)
59. Liu, H., Motoda, H.: Computational methods of feature selection. Chapman & Hall/CRC (2008)