Enhanced Topic-based Vector Space Model for Semantics-aware Spam Filtering

Igor Santos*, Carlos Laorden, Borja Sanz, Pablo G. Bringas

Laboratory for Smartness, Semantics and Security (S^3Lab), University of Deusto, Avenida de las Universidades 24, 48007 Bilbao, Spain
Telephone: +34944139003 Fax: +34944139166

Abstract

Spam has become a major issue in computer security because it is a channel for threats such as computer viruses, worms and phishing. More than 85% of received e-mails are spam. Historical approaches to combat these messages including simple techniques such as sender blacklisting or the use of e-mail signatures, are no longer completely reliable. Currently, many solutions feature machine-learning algorithms trained using statistical representations of the terms that usually appear in the e-mails. Still, these methods are merely syntactic and are unable to account for the underlying semantics of terms within the messages. In this paper, we explore the use of semantics in spam filtering by representing e-mails with a recently introduced Information Retrieval model: the enhanced Topic-based Vector Space Model (eTVSM). This model is capable of representing linguistic phenomena using a semantic ontology. Based upon this representation, we apply several well-known machine-learning models and show that the proposed method can detect the internal semantics of spam messages.

Keywords: spam detection, information retrieval, semantics, computer security, machine-learning

*Corresponding author

Email addresses: `isantos@deusto.es` (Igor Santos), `claorden@deusto.es` (Carlos Laorden), `borja.sanz@deusto.es` (Borja Sanz), `pablo.garcia.bringas@deusto.es` (Pablo G. Bringas)

1. Introduction

Electronic mail (e-mail) is a powerful communication channel. Nevertheless, as happens with all useful media, it is prone to misuse. In the past decade, *spam*, or junk mail, has become an important problem for e-mail users: a huge amount of spam arrives in people's mailboxes every day. At the time of this writing, 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 a major computer security problem that costs billions of dollars in productivity losses (Bratko et al., 2006). Moreover, it can be used as a medium for *phishing* (i.e., attacks that seek to acquire sensitive information from end-users) (Jagatic et al., 2007) and the spread of *malicious software* (e.g., computer viruses, Trojan horses, spyware and Internet worms) (Bratko et al., 2006).

Owing to the magnitude of the spam issue, several *spam filtering systems* have been proposed. The simplest methods for junk message filtering are often *blacklisting* or *signature-based* (Carpinter & Hunt, 2006). 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* (Heron, 2009) 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 (Jung & Sit, 2004; Ramachandran et al., 2006).

In contrast, signature-based systems create a unique hash value (i.e., a message digest) for each known spam message (Kołcz et al., 2004). The main advantage of these types of methods is that they rarely produce false positives. 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 technique in order to spread signatures (Carpinter & Hunt, 2006).

These simplistic methods have several shortcomings, however. First,

[1]http://www.junk-o-meter.com/stats/index.php
[2]http://www.cloudmark.com
[3]http://razor.sourceforge.net

blacklisting methods have a very high rate of false positives, making them unreliable as a *standalone* solution (Mishne et al., 2005). Second, signature-based systems are unable to detect spam until the junk message has been identified, properly registered and documented (Carpinter & Hunt, 2006).

In order to find a solution to this problem, the research community has undertaken a huge amount of work. Because *machine-learning* approaches have succeeded in text categorisation problems (Sebastiani, 2002), these techniques have been adopted in spam filtering systems. Consequently, substantial work has been dedicated to the *Naïve Bayes* classifier (Lewis, 1998), with studies in anti-spam filtering confirming it effective (Androutsopoulos et al., 2000c; Schneider, 2003; Androutsopoulos et al., 2000a,b; Seewald, 2007). Another broadly embraced machine-learning technique is *Support Vector Machines* (SVM) (Vapnik, 2000). The advantage of SVM is that its accuracy is not degraded even with many features (Drucker et al., 1999). Therefore, such approaches have been applied to spam filtering (Blanzieri & Bryl, 2007; Sculley & Wachman, 2007). Likewise, *Decision Trees* that classify by means of automatically learned rule-sets (i.e., tests) (Quinlan, 1986) have also been used for spam filtering (Carreras & Márquez, 2001). All of these machine-learning-based spam filtering approaches termed *statistical approaches* (Zhang et al., 2004).

Machine-learning approaches model e-mail messages using the *Vector Space Model* (VSM) (Salton et al., 1975), 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. Still, the VSM assumes that every term is independent, which is, at least from the linguistic point of view, not completely true. Thus, it cannot handle the existing linguistic phenomena in natural languages (Becker & Kuropka, 2003).

Because of this shortcoming of the VSM, the *Topic-based Vector Space Model* (TVSM) (Becker & Kuropka, 2003) and the *enhanced Topic-based Vector Space Model* (eTVSM) (Kuropka, 2004) 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 (Awad et al., 2008).

Against this background, we propose the first spam filtering model that uses an eTVSM to represent e-mail messages. More accurately, we use an implementation of an eTSVM that applies the *WordNet* semantic ontology (Fellbaum et al., 1998) for identifying synonym terms that share same *interpretation*. Thereafter, based on this representation, we train several supervised machine-learning algorithms for detecting and filtering junk e-mails. In summary, we advance the state of the art through the following contributions:

- We formally describe how to apply an eTVSM as a representation for e-mails in order to detect and filter spam.
- We provide an empirical validation of our method with an extensive study of several machine-learning classifiers.
- We show that the proposed method achieves high filtering rates, even on completely new, previously unseen spam; we discuss the weakness of the proposed model and explain possible enhancements.

The remainder of this paper is organised as follows. Section 2 describes in a technical and detailed manner the process of adapting the eTVSM for representing e-mails. Section 3 explains the machine-learning approaches used. Section 4 details the experiments performed and presents the results. Section 6 discusses the main shortcomings of the proposed method and proposes possible improvements. Finally, Section 7 concludes and outlines avenues for future work.

2. Semantics-aware e-mail message representation

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 (Cohen, 1974). Specifically, we can find several linguistic phenomena in natural languages (Polyvyanyy, 2007):

- **Synonyms:** Two or more words are interchangeable because of their similar (or identical) meanings (e.g., *'buy'* and *'purchase'*) (Carnap, 1955).
- **Hyponyms:** Specific instances of a more general word (e.g., *'spicy'* and *'salty'* are hyponyms of *'flavour'*)(Cruse, 1975).

- **Metonymy:** The substitution of one word for another with which it is associated (e.g., *'police'* instead of *'law enforcement'*) (Radden & Kövecses, 1999).
- **Homography:** Words with the same orthography but different meaning (e.g., *'bear'*: 'to support and carry' and 'an animal') (Ming-Tzu & Nation, 2004).
- **Word-groups:** Clusters of words that have particular semantic meanings when they are grouped together (e.g., *'New York City'*).

Through the study of how these phenomena affect spam-filtering systems, we are able to build a *semantics-aware* model. In this paper, we focus only on synonyms because there are attacks that evade spam filtering systems through the use of synonyms (Karlberger et al., 2007). These attacks exploit the redundancy of natural languages by substituting a word with a high spam probability with a synonym that has a lower spam probability whenever possible. However, our model is capable of defeating these attacks

Specifically, we use the information found within the body and subject of the e-mail message and discard every other information — like the sender or time-stamp of the e-mail. In order to represent messages, we start by removing *stop-words* (Wilbur & Sirotkin, 1992), which are words devoid of content (e.g., 'a','the', and 'is') that do not provide any semantic information and add noise to the model (Salton & McGill, 1983).

Afterwards, we represent the e-mails using an *Information Retrieval* (IR) model. Formally, let an IR model be defined as a 4-tuple $[E, Q, F, R(q_i, e_j)]$ (Baeza-Yates & Ribeiro-Neto, 1999) where:

- E, is a set of representations of e-mail;
- Q, is a set of representations of user queries;
- 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, $(q_i \in Q)$ and e-mail representation e_j, $(e_j \in E)$. This function is also called a *similarity function*.

Let E be a set of text e-mails e, $\{e : \{t_1, t_2, ...t_n\}\}$, each one comprises 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 $\vec{e_j} = (w_{1,j}, w_{2,j}, ...w_{n,j})$.

On the basis of this formalisation, we can apply several IR models. Commonly, spam filtering systems use *Vector Space Model* (VSM). 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.

However, the main shortcoming of the VSM is that it assumes independence between terms. In other words, it represents documents *syntactically* and cannot accommodate *synonyms* or other linguistic phenomena.

Consequently, the *Topic-based Vector Space Model* (TVSM) (Becker & Kuropka, 2003) also represents documents as vectors in an n dimensional space R (shown in equation 1) that has only positive axis intercepts (Kuropka, 2004). Nevertheless, each dimension of R represents a so-called *fundamental* topic. These axes are orthogonal (i.e., they are assumed to be inter-independent).

$$R \in \mathbb{R}^d_{\geq 0} \text{ with } e \in \mathbb{N}_{\geq 0} \tag{1}$$

Formally, a term t_i is represented in the TVSM by a term vector $\vec{t_i}$ and assigned a term weight between zero and one. A term vector direction represents how a term is associated with a fundamental topic, whereas the algebraic term vector length $|\vec{t_i}|$ corresponds to a term weight:

$$\begin{aligned} &\vec{t_i} = (t_{i,1}, t_{i,2}, ..., t_{i,n}) \\ &\text{with } |\vec{t_i}| = \sqrt{t_{i,1}^2, t_{i,2}^2, ..., t_{i,n^2}} \in [0;1] \end{aligned} \tag{2}$$

A model of the e-mail e_j, if represented via TVSM, is defined by an e-mail vector $\vec{e_j} \in R$. Further, the e-mail vector length is normalised to a length of one for convenience. In this way, a TVSM document model is obtained as the sum of the term vectors present in the document.

The *enhanced Topic-based Vector Space Model* (eTVSM) (Kuropka, 2004) is an improvement over the TVSM because the weights of the terms are based upon a defined *ontology* (a formal representation of a set of concepts from a domain and the relationships between those concepts). We chose this model to formalise e-mail messages because e-mail messages are composed of terms belonging to natural language and because the ontology is able to represent the aforementioned term-relations.

In a similar vein to a VSM (Salton et al., 1975) or a TVSM (Becker & Kuropka, 2003), an eTVSM represents its concepts as vectors in an algebraic space. However, an eTVSM focuses not only on terms, but also on interpretations. In order to achieve these formalisations, an eTVSM constructs document models from interpretation vectors (Polyvyanyy, 2007). Relations between concepts are expressed through concept vector angles, representing semantic similarity between concepts. Furthermore, eTVSM is capable of taking advantage of ontology concepts, by transforming them into vectors in an operational vector space, that maps ontological semantic relationships onto vector angles.

In order to construct an ontology (Floridi, 2003), an eTVSM utilises the concepts of topics, interpretations and terms, which are organised in a hierarchical, non-cyclic, directed graph structure (Gross & Yellen, 2004). The edges of the graph aim to find semantic connections between concepts of the same class as well as inter-conceptual semantic links. Whereas a topic concept is the most general semantic entity of an eTVSM ontology, other concepts refine existing topics. A directed graph with topics and nodes (called a *topic map*) is used to express inter-topic relations. Intermediate links between topics and terms correspond to interpretations, which play the role of semantic terms. Terms can be linked to an arbitrary number of topics and cannot be linked to other interpretations. Finally, terms are treated as the smallest information unit with one or several semantic interpretations. In order to express all semantic meanings, a term might be linked with an arbitrary number of interpretations.

For our purposes, we make the most of the *Themis*[4] implementation of the eTVSM, which enables the ontology model to be populated with *WordNet* (Fellbaum et al., 1998), a semantic lexicon database for the English language.

[4]http://code.google.com/p/ir-themis/

In order to proceed, the eTVSM constructs models for the target e-mails. An e-mail model, $e_j \in E$, is represented by an e-mail vector, which is defined as a weighted sum of interpretation vectors included in the document model (shown in equation 3):

$$\begin{aligned} &\forall d_j \in D : \vec{d_j} = \frac{1}{|\vec{\delta_j}|} \cdot \vec{\delta_j} \Rightarrow |\vec{d_j}| = 1 \\ &\text{and} \qquad \vec{\delta_j} = \sum_{\phi_i \in \Phi} \omega_{d_j,\phi_i} \cdot \vec{\phi_i} \end{aligned} \tag{3}$$

where ω_{e_j,ϕ_i} is the weight of the interpretation ϕ_i ($\phi_i \in \Phi$ is the set of interpretations) in the e-mail e_j. Essentially, the heuristics for the construction of interpretation vectors consider that eTVSM shows the similarity level as a vector angle trough the ontology graph structure. Because only the direction of the vector is significant for obtaining angles between vectors, an e-mail vector length is normalised (shown in equation 4).

$$\begin{aligned} |\vec{\delta_i}| &= \left| \sum_{\phi_k \in \Phi} \omega_{e_i,\phi_k} \vec{\phi_k} \right| = \sqrt{\left| \sum_{\phi_k \in \Phi} \omega_{e_i,\phi_k} \vec{\phi_k} \right|^2} \\ &= \sqrt{\left(\sum_{\phi_k \in \Phi} \omega_{e_i,\phi_k} \vec{\phi_k} \right)^2} \\ &= \sqrt{\sum_{\phi_k \in \Phi} \sum_{\phi_l \in \Phi} \omega_{e_i,\phi_k} \omega_{e_i,\phi_l} \vec{\phi_k} \vec{\phi_l}} \end{aligned} \tag{4}$$

In this way, our proposed method retrieves the weighted interpretation vectors representing email topics, and builds a model, which we use to train several machine-learning classification algorithms. In order to perform this training, we first create an *ARFF* file (Holmes et al., 1994) (i.e., attribute relation file format) that describes the shared attributes (e.g., topics) for each instance (e.g., document). Secondly, we use the *Waikato Environment for Knowledge Analysis* (WEKA) (Garner, 1995) to build the desired machine-learning classifier. Finally, we test different machine-learning classification algorithms with WEKA as described in Section 4.

3. Machine-learning Classification

Machine learning is an active research area within *Artificial Intelligence* (AI) that focuses on the design and development of new algorithms that allow computers to reason and decide based on data (i.e. computer learning) (Bishop, 2006).

Machine-learning algorithms can commonly be divided into three different types: *supervised learning*, *unsupervised learning* and *semi-supervised learning*. For supervised algorithms, the training dataset must be labelled (e.g., whether an e-mail is spam or not) (Kotsiantis, 2007). Unsupervised learning algorithms try to determine how data are organised into different groups named *clusters*. Therefore, data do not need to be labelled (Kotsiantis & Pintelas, 2004). Finally, semi-supervised machine-learning algorithms use a mixture of both labelled and unlabelled data in order to build models, improving the accuracy of unsupervised methods (Chapelle et al., 2006).

Because spam e-mails can be properly labelled, we use supervised machine-learning; however, in the future, we would also like to test unsupervised methods for spam filtering. In the remainder of this section, we review several supervised machine-learning approaches that have succeeded in similar domains (Drucker et al., 1999; Androutsopoulos et al., 2000c; Schneider, 2003; Androutsopoulos et al., 2000a,b; Seewald, 2007; Carreras & Márquez, 2001).

3.1. Bayesian Networks

Bayesian networks (Pearl, 1982), which are based on *Bayes' Theorem* (Bayes, 1763), are defined as graphical probabilistic models for multivariate analysis. Specifically, they are directed acyclic graphs that have associated probability distribution functions (Castillo et al., 1996). The nodes of the directed graph represent problem variables (either premises or conclusions), and the edges represent conditional dependencies between such variables. Moreover, the probability function illustrates the strengths of these relationships in the graph (Castillo et al., 1996).

The most important capability of Bayesian networks is their ability to determine the probability that a certain hypothesis is true (e.g., the probability that an e-mail is spam or legitimate) given a historical dataset.

3.2. Decision Trees

Decision tree classifiers are a type of machine-learning classifier usually graphically represented as a tree. The internal nodes represent conditions on the variables of a problem, whereas the final nodes or leaves represent the possible ultimate decisions of the algorithm (Quinlan, 1986).

Different training methods are typically used for learning the graph structure of these models from a labelled dataset. We use Random Forest, an ensemble (i.e., combination of weak classifiers) of different randomly built decision trees (Breiman, 2001), and J48, the WEKA (Garner, 1995) implementation of the C4.5 algorithm (Quinlan, 1993)).

3.3. K-Nearest Neighbour

The *K-Nearest Neighbour* (KNN) (Fix & Hodges, 1952) classifier is one of the simplest supervised machine learning models. This method classifies an unknown specimen based on the class of the instances closest to it in the training space by measuring the distance between the training instances and the unknown instance.

Although several possible methods can be used to choose the class of the unknown sample, the most common technique is simply to classify the unknown instance as the most common class amongst the K-nearest neighbours.

3.4. Support Vector Machines (SVM)

SVM algorithms divide the n-dimensional space representation of the data into two regions using a hyperplane. This hyperplane always maximises the margin between the two regions or classes. The margin is defined by the longest distance between the examples of the two classes and is computed based on the distance between the closest instances of both classes to the margin, which are called *supporting vectors* (Vapnik, 2000).

Instead of using linear hyperplanes, many implementations of these algorithms use so-called *kernel functions.* These kernel functions lead to non-linear classification surfaces, such as polynomial, radial or sigmoid surfaces ((Amari & Wu, 1999).

4. Experiments and Results

In order to validate our proposed method, we used the *Ling Spam* dataset.[5] Ling Spam consists of a mixture of both spam and legitimate messages retrieved from the *Linguistic list*, an e-mail distribution list about *linguistics*.

[5]http://nlp.cs.aueb.gr/software_and_datasets/lingspam_public.tar.gz

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 2893 different e-mails, of which 2412 are legitimate e-mails obtained by downloading digests from the list and 481 are spam e-mails retrieved from one of the authors of the corpus (for a more detailed description of the corpus please refer to (Androutsopoulos et al., 2000a; Sakkis et al., 2003)). Junk messages represent approximately the 16% of the whole dataset, a rate close to the actual rate (Cranor & LaMacchia, 1998; Sahami et al., 1998; Sakkis et al., 2003). *Stop Word Removal* (Wilbur & Sirotkin, 1992) and *stemming* (Lovins, 1968) 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.
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.

Because the eTVSM model relies on identifying relations between words, the stemming process cannot be applied and, therefore, we are not able to use either *lemm* or *lemm_stop* datasets. In addition, instead of using the *stop* dataset we used the *bare* dataset and we performed a stop word removal based on an external stop-word list.[6]

Regarding the eTVSM representation of e-mail messages we used the Themis implementation (Polyvyanyy, 2007). Using this implementation we were able to populate a database with the e-mail messages from the Ling Spam dataset. In addition, we used WordNet ontology[7] to seed the model and focused only on synonymy relationships within the ontology. The reason behind focusing only on synonyms is that a previous evaluation of eTVSM showed that this kind of representation obtained the best results (Polyvyanyy, 2007) .

[6]http://www.webconfs.com/stop-words.php
[7]http://wordnet.princeton.edu/wordnet/download/

We constructed a file with the resultant vector representations of the e-mails in order to validate our method. We extracted the top 1000 attributes using *Information Gain* (Kent, 1983) (equation 5), an algorithm that evaluates the relevance of an attribute by measuring the information gain with respect to the class:

$$IG(j) = \sum_{v_j \in R} \sum_{C_i} P(v_j, C_i) \cdot \frac{P(v_j, C_i)}{P(v_j) \cdot P(C_i)} \tag{5}$$

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.

To assess the machine-learning classifiers, we used the following methodology:

- **Cross validation:** In order to evaluate the performance of machine-learning classifiers, *k-fold cross validation* (Kohavi, 1995) is commonly used in machine-learning experiments (Bishop, 2006).

  For each classifier tested, we performed a k-fold cross validation with $k = 10$. In this way, our dataset was split 10 times into 10 different sets of learning sets (90% of the total dataset) and testing sets (10% of the total data).

- **Learning the model:** For each fold, we perform the learning phase of each algorithm with each training dataset, applying different parameters or learning algorithms depending on the concrete classifier. We used four different models:

  – *Bayesian Networks:* In order to train Bayesian Networks, we used different structural learning algorithms; *K2* (Cooper & Herskovits, 1991), *Hill Climber* (Russell & Norvig, 2003) and *Tree Augmented Naïve* (TAN) (Geiger et al., 1997). We also performed experiments with *Naïve Bayes* (Lewis, 1998), a classifier that has been widely used for spam filtering (Androutsopoulos et al., 2000c; Schneider, 2003; Androutsopoulos et al., 2000a,b; Seewald, 2007).

  - *Decision Trees:* In order to train decision trees, we used *Random Forest* (Breiman, 2001) and *J48* (Weka's *C4.5* (Quinlan, 1993) implementation).
  - *K-Nearest Neighbour:* For KNN, we performed experiments with k from 1 to 5.
  - *Support Vector Machines:* We used a *Sequential Minimal Optimisation* (SMO) algorithm (Platt, 1999) with a *polynomial kernel* (Amari & Wu, 1999), a *normalised polynomial kernel* (Amari & Wu, 1999), a *Pearson VII function-based* universal kernel (Üstün et al., 2006), and a *Radial Basis Function* (RBF) based kernel (Amari & Wu, 1999). In addition, we used LibSVM[8] for the linear (i.e., hyperplane) and sigmoid kernel (Lin & Lin, 2003) implementation.

- **Testing the models:** To measure the processing overhead of the model, we measure required training and testing times:
  - *Training time:* The overhead required for building the different machine-learning algorithms.
  - *Testing time:* The total time that the models require to evaluate the testing instances in the dataset.

  To evaluate each classifier's capability we measured *accuracy*, which is the total number of the classifier's hits divided by the number of messages in the whole dataset (shown in equation 6).

$$Accuracy(\%) = \frac{TP + TN}{TP + FP + TP + TN} \cdot 100 \tag{6}$$

  where TP is the amount of correctly classified spam (i.e. true positives), FN is the amount of spam misclassified as legitimate mails (false negatives), FP is the amount of legitimate mail incorrectly detected as spam, and TN is the number of legitimate mail correctly classified.

  Furthermore, we measured the *True Positive Ratio* (TPR) which is the number of spam messages correctly detected, divided by the total number of junk e-mails (shown in equation 7):

[8]http://www.csie.ntu.edu.tw/~cjlin/libsvm/

$$TPR = \frac{TP}{TP + FN} \tag{7}$$

Moreover, we measured the *False Positive Ratio* (FPR), which is the number of legitimate messages misclassified as spam divided by the total number of legitimate e-mails (shown in equation 8):

$$FPR = \frac{FP}{FP + TN} \tag{8}$$

Besides, we measured the *Area Under the ROC Curve* (AUC), which establishes the relation between false negatives and false positives (Singh et al., 2009). The ROC curve is represented by plotting the rate of true positives (TPR) against the rate of false positives (FPR).

Table 1: Time results of machine-learning classifiers.

Classifier	Training Time (s)	Testing Time (s)
DT: J48	67.75	0.00
DT: Random Forest N=10	5.98	0.01
Bayesian Network: K2	5.78	0.08
Bayesian Network: Hill Climber	421.37	0.09
Bayesian Network: TAN	436.72	0.14
Naïve Bayes	3.43	0.25
Knn K=1	0.00	1.74
Knn K=2	0.00	1.95
Knn K=3	0.00	2.00
Knn K=4	0.00	2.04
Knn K=5	0.00	2.09
SVM: Lineal	1.80	0.05
SVM: RBF Kernel	9.90	0.26
SVM: Polynomial Kernel	0.57	0.01
SVM: Normalised Polynomial Kernel	16.81	0.45
SVM: Pearson VII Kernel	20.65	0.58
SVM: Sigmoid Kernel	3.94	0.19

Table 1 shows the time results for training and testing of the supervised machine-learning models. We used a Intel Core 2 Quad CPU clocked at 2.83 GHz with 8 GB of RAM memory. The KNN algorithm did not require any training time. However, it was the slowest in terms of testing time, with results between 1.74 and 2.09 seconds. SVM with polynomial kernel was the fastest of the tested configurations for SVM, achieving a training time of 0.57 seconds and a testing time of 0.58 seconds. Naïve Bayes required 3.43 seconds for training and 0.25 second for testing. The performance of the Bayesian networks depended on the algorithm used. Overall, we found that K2 is the fastest Bayesian classifier, requiring 5.78 seconds for training and 0.08 seconds for testing. TAN had the slowest training time at 436.72 seconds; however, it only required 0.14 seconds for the testing step. Among the decision trees, Random Forest performed faster than J48, with 5.98 seconds of training time and 0.01 seconds of testing time.

Table 2: Results of the evaluation of machine-learning classifiers based on eTVSM representation spam detection.

Classifier	**Accuracy (%)**	**TPR**	**FPR**	**AUC**
2 DT: J48	95.64	0.83	0.02	0.92
DT: Random Forest N=10	98.72	0.94	0.00	1.00
Bayesian Network: K2	96.62	0.80	0.00	1.00
Bayesian Network: Hill Climber	96.62	0.80	0.00	1.00
Bayesian Network: TAN	99.26	0.97	0.00	1.00
Naïve Bayes	94.94	0.76	0.01	0.99
Knn K=1	93.48	0.68	0.02	0.83
Knn K=2	93.57	0.76	0.03	0.88
Knn K=3	93.25	0.60	0.00	0.90
Knn K=4	94.03	0.66	0.01	0.91
Knn K=5	92.19	0.52	0.00	0.92
SVM: Lineal	98.42	0.95	0.01	0.97
SVM: RBF Kernel	96.93	0.84	0.01	0.92
SVM: Polynomial Kernel	97.52	0.86	0.00	0.93
SVM: Normalised Polynomial Kernel	94.30	0.71	0.01	0.85
SVM: Pearson VII Kernel	90.07	0.40	0.00	0.70
SVM: Sigmoid Kernel	74.55	0.19	0.15	0.52

Table 2 shows the results for the evaluation of the aforementioned supervised machine-learning classifiers. Every classifier obtained an accuracy higher than 92%, with the exception of the SVM using sigmoid kernel. Specifically, Bayesian networks trained with a TAN algorithm achieved the best results, with an accuracy of 99.26%. Nevertheless, because the training dataset is clearly unbalanced (only the 16% of the messages were spam), focusing only on the percent of correctly classified messages for the evaluation of learning schemas is inconclusive. Therefore, we also evaluated TPR, FPR and the

AUC. Obtained TPR rates show a significant difference between methods. Several of the evaluated classifiers detect fewer than the 50% of the total junk messages. In particular, SVM trained with the Pearson VII Kernel obtained a 40% detection rate and the SVM trained with the Sigmoid Kernel a 19% rate. Bayes-based classifiers, SVM, and Decision trees were capable of detecting more than 80% of spam messages. Bayesian networks were the best, achieving a TPR of 97% when trained with TAN. With regards to FPR, every classifier obtained a ratio lower than 20%. Because classifiers tend to reduce overall errors rather than class-specific ones (Bishop, 2006), it is logical to think that the higher proportion of legitimate mails causes these low FPR results. Nonetheless, SVMs trained with Sigmoid Kernel performed most poorly with a 15% of FPR. Several classifiers obtained a 0% FPR. Finally, the results for the AUC showed that the best classifiers were Random Forest and Bayesian networks, with an AUC of 1, indicating that their balance between TPR and FPR is optimal.

We make several observations from the experimental evaluation. First, the best overall results were obtained by Bayesian networks. These classifiers have a long history in spam filtering using a simpler VSM model and *bag of words*. They also behave well for the eTVSM-based representation. Second, the performance of Random Forest was also high and thus, may be adequate for spam filtering. Finally, some SVM kernels behave better than others: SVM with Polynomial kernels obtained high accuracy and high AUC results, while Sigmoid functions and RBF did not. These results indicate that the training space fits better to polynomial divisions than to other divisions.

5. Comparison with previous work

To evaluate the contribution of the eTVSM to spam filtering, we compare the filtering capabilities of our approach with the reported results of both real-world solutions and academic approaches see Table 3).

Cormack & Lynam (2007) tested the real-world spam filtering tools SpamAssassin, Bogofilter, SpamProbe and CRM114 against the Ling Spam corpus. Overall, these tools were unable to classify the messages correctly.

Androutsopoulos et al. (2000a) proposed a model based on the Naïve Bayes classifier. The classifier was constructed using the terms within the mail messages. The parameter λ represents the probability of an e-mail of being spam required to actually classify it as spam. In other words, a value

Table 3: Comparison of the results of both commercial or open-source solutions and academical proposals for spam filtering tools with our semantics-aware approach.

Model	Accuracy (%)	TPR	FPR	AUC
Real-world spam filtering solutions (Cormack & Lynam, 2007)				
SpamAssassin[9]	84.1	0.04	0	0.52
BogoFilter[10]	90.1	0.40	0	0.70
SpamProbe[11]	94.8	0.69	0	0.84
CRM114[12]	81.5	88.8	0.45	0.25
Androutsopoulos et al. (2000a)				
$\lambda = 1$	97.06	0.83	0.08	0.91
$\lambda = 9$	96.33	0.78	0.08	0.86
$\lambda = 999$	94.19	0.65	0	0.82
Sakkis et al. (2001)				
$k = 5$, $\lambda = 1$ and $m = 100$	98.06	0,92	0.01	0.97
$k = 3$, $\lambda = 9$ and $m = 200$	97.20	0.84	0.01	0.96
$k = 7$, $\lambda = 1$ and $m = 300$	84.89	0.90	0.16	0.75
$k = 3$, $\lambda = 9$ and $m = 100$	97.30	0.85	0.01	0.96
Schneider (2003)				
Bernoulli	98.00	0.89	0.01	0.97
mv-MI	98.86	0.96	0.01	0.98
mn-MI	98.06	0.93	0.01	0.97
dmn-MI	85.52	0.17	0.01	0.78
tf-MI	98.79	0.96	0.01	0.98
dtf-MI	98.48	0.95	0.01	0.97
tftf-MI	98.79	0.96	0.01	0.97
Our semantics-aware approach				
Bayesian Network: TAN	99.26	0.97	0.00	1.00
DT: Random Forest N=10	98.72	0.94	0.00	1.00

of lambda of 9 means that misclassifying a legitimate e-mail as spam is 9 times worse than the opposite mistake.

The approach proposed by Sakkis et al. (2001) extended the previous approach through stacked generalisation (Wolpert, 1992), which combines different classifiers. They used a memory-based classifier (Androutsopoulos et al., 2000c). The parameter k is the neighbourhood size for the memory-based learner, lambda determines the strictness of the criterion for classifying a message as spam, and m is the number of terms used for representing e-mails.

Schneider (2003) performed experiments with a Naïve Bayes text classifier using two classifier statistical event models ((McCallum & Nigam, 1998): a multivariate Bernoulli model and a multinomial model. The multivariate Bernoulli model does not take into account the number of times different words are used while the multinomial model does. Both models use only a subset of words, i.e., a fixed vocabulary. To select the words within the vocabulary, different feature selection algorithms were used for each model.

Although these research approaches generally obtained very high filtering rates (higher than 95 %), they failed to represent semantic relationships between terms. Our approach introduces a representation based on interpretations rather than terms, and therefore, it improves the results of these syntactic spam filtering systems (see Table 3).

6. Discussion

The final results show that the use of eTVSM for the representation of e-mail messages to detect spam 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. Regarding the evaluation of the different supervised machine-learning classifiers, Bayesian networks trained with Tree Augmented Naïve Bayesian algorithms outperformed the rest of the classifiers. Nevertheless, several points of discussion are important regarding the suitability of the proposed method.

First, we only used the eTVSM for representing synonymy within the terms of the messages. There are further relations between words that may be interesting for spam filtering. For instance, regarding *hyponymy*, imagine two emails composed of the following terms: $D_1 = \{$*buy a sedan*$\}$ and $D_2 = \{$*buy a car*$\}$. Using only synonymy D_1 and D_2 would be similar only by the term *'buy'* and not for the terms *'car'* and *sedan*, although sedan is a type of car and, hence, the meaning of the messages must be interpreted as the same. Likewise, the other aforementioned linguistic phenomena, especially *meronymy*, are interesting for further study to determine how it is possible to improve the performance of spam filtering techniques.

Second, with the inclusion of the concept *interpretation* into spam filtering, there is 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. In this way, WSD is considered necessary in order to accomplish most natural language processing tasks (Ide & Véronis, 1998). Therefore, we propose the study of different WSD techniques (a survey of different WSD techniques can be found in (Navigli, 2009)) 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 (Bates & Weischedel, 1993) and thus be language dependant.

Finally, 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 bypassing 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) (Dietterich et al., 1997). 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 (Maron & Lozano-Pérez, 1998). (Zhou et al., 2007) 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 (Wittel & Wu, 2004). All of these attacks, which spammers have been adopting, should be taken into account in the construction of future spam filtering systems, an area in which where we think semantics will be a topic of research and concern in the coming years.

7. 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, lack an underlying semantic comprehension of the e-mail messages.

In this paper, we presented a spam filtering system that further develops filters sensitive to the semantics present in e-mails by applying eTVSM, a recent IR model that is capable of representing linguistic relationships between terms. Using this representation, we are able to deal with synonyms in the messages and, therefore, enhance current machine-learning methods. Our experiments show that this approach provides high percentages of spam detection whilst keeping the number of misclassified legitimate messages low.

Future versions of this spam filtering system will move in three main directions. First, we will focus on attacks against statistical spam filtering systems such as tokenisation or good word attacks. Second, we plan to enhance the semantics of this method with more linguistic relationships. Finally, we will study the feasibility of applying Word Sense Disambiguation techniques to this spam filtering method.

References

Amari, S., & Wu, S. (1999). Improving support vector machine classifiers by modifying kernel functions. *Neural Networks*, *12*, 783–789.

Androutsopoulos, I., Koutsias, J., Chandrinos, K., Paliouras, G., & Spyropoulos, C. (2000a). An evaluation of naive bayesian anti-spam filtering. In *Proceedings of the workshop on Machine Learning in the New Information Age* (pp. 9–17).

Androutsopoulos, I., Koutsias, J., Chandrinos, K., & Spyropoulos, C. (2000b). An experimental comparison of naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 160–167).

Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Sakkis, G., Spyropoulos, C., & Stamatopoulos, P. (2000c). 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 4th European Conference on Principles and Practice of Knowledge Discovery in Databases*.

Awad, A., Polyvyanyy, A., & Weske, M. (2008). Semantic querying of business process models. In *IEEE International Conference on Enterprise Distributed Object Computing Conference (EDOC 2008)* (pp. 85–94).

Baeza-Yates, R. A., & Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Bates, M., & Weischedel, R. (1993). *Challenges in natural language processing*. Cambridge Univ Pr.

Bayes, T. (1763). An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society*, *53*, 370–418.

Becker, J., & Kuropka, D. (2003). Topic-based vector space model. In *Proceedings of the 6th International Conference on Business Information Systems* (pp. 7–12).

Bishop, C. (2006). *Pattern recognition and machine learning*. Springer New York.

Blanzieri, E., & Bryl, A. (2007). Instance-based spam filtering using SVM nearest neighbor classifier. *Proceedings of FLAIRS-20*, (pp. 441–442).

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

Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5–32.

Carnap, R. (1955). Meaning and synonymy in natural languages. *Philosophical Studies*, *6*, 33–47.

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

Carreras, X., & Márquez, L. (2001). Boosting trees for anti-spam email filtering. In *Proceedings of RANLP-01, 4th international conference on recent advances in natural language processing* (pp. 58–64). Citeseer.

Castillo, E., Gutiérrez, J. M., & Hadi, A. S. (1996). *Expert Systems and Probabilistic Network Models*. (Erste ed.). New York, NY, USA.

Chapelle, O., Schölkopf, B., & Zien, A. (2006). *Semi-supervised learning*. MIT Press.

Cohen, D. (1974). *Explaining linguistic phenomena*. Halsted Press.

Cooper, G. F., & Herskovits, E. (1991). A bayesian method for constructing bayesian belief networks from databases. In *Proceedings of the 7^{th} conference on Uncertainty in artificial intelligence*.

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

Cranor, L., & LaMacchia, B. (1998). Spam! *Communications of the ACM*, *41*, 74–83.

Cruse, D. (1975). Hyponymy and lexical hierarchies. *Archivum Linguisticum*, *6*, 26–31.

Dietterich, T., Lathrop, R., & Lozano-Pérez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, *89*, 31–71.

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

Fellbaum, C. et al. (1998). *WordNet: An electronic lexical database*. MIT press Cambridge, MA.

Fix, E., & Hodges, J. L. (1952). *Discriminatory analysis: Nonparametric discrimination: Small sample performance. Technical Report Project 21-49-004, Report Number 11*. Technical Report USAF School of Aviation Medicine, Randolf Field, Texas.

Floridi, L. (Ed.) (2003). *Blackwell Guide to the Philosophy of Computing and Information*. Cambridge, MA, USA: Blackwell Publishers, Inc.

Garner, S. (1995). Weka: The Waikato environment for knowledge analysis. In *Proceedings of the New Zealand Computer Science Research Students Conference* (pp. 57–64).

Geiger, D., Goldszmidt, M., Provan, G., Langley, P., & Smyth, P. (1997). Bayesian network classifiers. In *Machine Learning* (pp. 131–163).

Gross, J. L., & Yellen, J. (2004). Handbook of graph theory. In *Series Discrete Mathematics and its Applications*. CRC press.

Heron, S. (2009). Technologies for spam detection. *Network Security*, *2009*, 11–15.

Holmes, G., Donkin, A., & Witten, I. H. (1994). Weka: a machine learning workbench. (pp. 357–361).

Ide, N., & Véronis, J. (1998). Introduction to the special issue on word sense disambiguation: the state of the art. *Computational linguistics*, *24*, 2–40.

Jagatic, T., Johnson, N., Jakobsson, M., & Menczer, F. (2007). Social phishing. *Communications of the ACM*, *50*, 94–100.

Jung, J., & Sit, E. (2004). An empirical study of spam traffic and the use of DNS black lists. In *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement* (pp. 370–375). ACM New York, NY, USA.

Karlberger, C., Bayler, G., Kruegel, C., & Kirda, E. (2007). Exploiting redundancy in natural language to penetrate bayesian spam filters. In *Proceedings of the 1st USENIX workshop on Offensive Technologies (WOOT)* (pp. 1–7). USENIX Association.

Kent, J. (1983). Information gain and a general measure of correlation. *Biometrika*, *70*, 163.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International Joint Conference on Artificial Intelligence* (pp. 1137–1145). volume 14.

Kołcz, A., Chowdhury, A., & Alspector, J. (2004). The impact of feature selection on signature-driven spam detection. In *Proceedings of the 1st Conference on Email and Anti-Spam (CEAS-2004)*.

Kotsiantis, S. (2007). Supervised Machine Learning: A Review of Classification Techniques. In *Proceeding of the 2007 conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies* (pp. 3–24).

Kotsiantis, S., & Pintelas, P. (2004). Recent advances in clustering: A brief survey. *WSEAS Transactions on Information Science and Applications*, *1*, 73–81.

Kuropka, D. (2004). Modelle zur Repräsentation natürlichsprachlicher Dokumente-Information-Filtering und-Retrieval mit relationalen Datenbanken. *Advances in Information Systems and Management Science*, *10*.

Lewis, D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. *Lecture Notes in Computer Science*, *1398*, 4–18.

Lin, H., & Lin, C. (2003). *A study on sigmoid kernels for SVM and the training of non-PSD kernels by SMO-type methods*. Technical Report Nat'l Taiwan University.

Lovins, J. (1968). Development of a Stemming Algorithm. *Mechanical Translation and Computational Linguistics*, *11*, 22–31.

Maron, O., & Lozano-Pérez, T. (1998). A framework for multiple-instance learning. *Advances in neural information processing systems*, (pp. 570–576).

McCallum, A., & Nigam, K. (1998). A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*. Citeseer volume 752.

Ming-Tzu, K., & Nation, P. (2004). Word meaning in academic English: Homography in the academic word list. *Applied linguistics*, *25*, 291–314.

Mishne, G., Carmel, D., & Lempel, R. (2005). Blocking blog spam with language model disagreement. In *Proceedings of the 1st International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)* (pp. 1–6).

Navigli, R. (2009). Word sense disambiguation: a survey. *ACM Computing Surveys (CSUR)*, *41*, 10.

Pearl, J. (1982). Reverend bayes on inference engines: a distributed hierarchical approach. In *Proceedings of the National Conference on Artificial Intelligence* (pp. 133–136).

Platt, J. (1999). Sequential minimal optimization: A fast algorithm for training support vector machines. *Advances in Kernel Methods-Support Vector Learning*, *208*.

Polyvyanyy, A. (2007). Evaluation of a novel information retrieval model: eTVSM. MSc Dissertation.

Quinlan, J. (1986). Induction of decision trees. *Machine learning*, *1*, 81–106.

Quinlan, J. (1993). *C4. 5 programs for machine learning*. Morgan Kaufmann Publishers.

Radden, G., & Kövecses, Z. (1999). Towards a theory of metonymy. *Metonymy in language and thought*, (pp. 17–59).

Ramachandran, A., Dagon, D., & Feamster, N. (2006). Can DNS-based blacklists keep up with bots. In *Conference on Email and Anti-Spam*. Citeseer.

Russell, S. J., & Norvig (2003). *Artificial Intelligence: A Modern Approach (Second Edition)*. Prentice Hall.

Sahami, M., Dumais, S., Heckerman, D., & Horvitz, E. (1998). A Bayesian approach to filtering junk e-mail. In *Learning for Text Categorization: Papers from the 1998 workshop* (pp. 98–05). Madison, Wisconsin: AAAI Technical Report WS-98-05 volume 62.

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

Sakkis, G., Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Spyropoulos, C. D., & Stamatopoulos, P. (2001). Stacking classifiers for anti-spam filtering of e-mail. In *Proceedings of the 6th Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 44–50).

Salton, G., & McGill, M. (1983). *Introduction to modern information retrieval*. McGraw-Hill New York.

Salton, G., Wong, A., & Yang, C. (1975). A vector space model for automatic indexing. *Communications of the ACM*, *18*, 613–620.

Schneider, K. (2003). A comparison of event models for Naive Bayes anti-spam e-mail filtering. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics* (pp. 307–314).

Sculley, D., & Wachman, G. (2007). Relaxed online SVMs for spam filtering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 415–422).

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, *34*, 1–47.

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

Singh, Y., Kaur, A., & Malhotra, R. (2009). Comparative analysis of regression and machine learning methods for predicting fault proneness models. *International Journal of Computer Applications in Technology*, *35*, 183–193.

Üstün, B., Melssen, W., & Buydens, L. (2006). Facilitating the application of Support Vector Regression by using a universal Pearson VII function based kernel. *Chemometrics and Intelligent Laboratory Systems*, *81*, 29–40.

Vapnik, V. (2000). *The nature of statistical learning theory*. Springer.

Wilbur, W., & Sirotkin, K. (1992). The automatic identification of stop words. *Journal of information science*, *18*, 45–55.

Wittel, G., & Wu, S. (2004). On attacking statistical spam filters. In *Proceedings of the 1st Conference on Email and Anti-Spam (CEAS)*.

Wolpert, D. (1992). Stacked generalization*. *Neural networks*, *5*, 241–259.

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

Zhou, Y., Jorgensen, Z., & Inge, M. (2007). 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* (pp. 298–305). IEEE Computer Society.