Word Sense Disambiguation for Spam Filtering

Carlos Laorden[a,*], Igor Santos[a,], Borja Sanz[a,], Gonzalo Alvarez[b,], Pablo G. Bringas[a,]

[a] *Laboratory for Smartness, Semantics and Security (S^3Lab), University of Deusto, Avenida de las Universidades 24, 48007 Bilbao, Spain*
Telephone: +34944139003 Fax: +34944139166
[b] *Instituto de Física Aplicada, Consejo Superior de Investigaciones Científicas (CSIC) C/Serrano 144, 28006 Madrid, Spain*
Tel.: +34915618806 Fax: +34914117651

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 86% of received e-mails are spam. Historical approaches to combating these messages, including simple techniques such as sender blacklisting or the use of e-mail signatures, are no longer completely reliable. Many current solutions feature machine-learning algorithms trained using statistical representations of the terms that most commonly appear in such e-mails. However, these methods are merely syntactic and are unable to account for the underlying semantics of terms within messages. In this paper, we explore the use of semantics in spam filtering by introducing a pre-processing step of Word Sense Disambiguation (WSD). Based upon this disambiguated 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 filtering, word sense disambiguation, secure e-commerce, computer security

*Corresponding author

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

1. Introduction

Spam has become a significant problem for e-mail users over the past decade; an enormous amount of spam arrives in peoples' mailboxes every day. At the time of writing, 86.6% of all e-mail messages are spam, according to the Spam-o-meter website[1]. Spam is also a major computer security problem: it is a medium for phishing (i.e., attacks that seek to acquire sensitive information from end-users) (Jagatic et al., 2007) and for spreading malicious software (e.g., computer viruses, Trojan horses, spyware and Internet worms) (Bratko et al., 2006).

Nevertheless, different studies show that the effect of spam in worldwide economy is notorious and prejudicial. Leung and Liang (2009) presented an analysis of the impact of phising on the market value of global firms, which showed that phising alerts pose significantly negative return on stock. In a similar vein, Mostafa Raad et al. (2010) offer another study to assess the influence and impact of spam in several companies whose email advertisement was considered as spam. Both examples clearly show the necessity to detect undesired messages, and, maybe more important, the need to restore the confidence of users in their e-mail filtering systems.

The simplest methods for filtering junk e-mail are usually blacklisting or signature-based (Carpinter and Hunt, 2006). Blacklisting is a simple technique that is broadly used in most filtering products; such systems filter out e-mails from certain senders. In contrast, whitelisting systems (Heron, 2009) deliver messages only from designated senders to reduce the number of misclassified legitimate e-mails (also known as 'ham' by the spam community). Another popular variant of these so-called banishing methods entails DNS blacklisting, in which the host address is checked against a list of networks or servers known to distribute spam (Jung and 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 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 is integrated with

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

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

the e-mail server, and Razor[3], a filtering system that uses a distributed and collaborative technique to spread signatures (Carpinter and Hunt, 2006).

However, these simplistic methods have several shortcomings. First, blacklisting methods produce a 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 messages until they have been identified, properly registered and documented (Carpinter and Hunt, 2006).

A large amount of research has been dedicated to finding better spam filtering solutions. Machine-learning approaches have been effectively applied to text categorisation problems (Sebastiani, 2002), and they have been adopted for use in spam filtering systems. Consequently, substantial work has been dedicated to naïve Bayes filtering (Lewis, 1998); several studies on its effectiveness have been published (Androutsopoulos et al., 2000c; Schneider, 2003; Androutsopoulos et al., 2000a,b; Seewald, 2007). Another broadly embraced machine-learning technique is the Support Vector Machine (SVM) method (Vapnik, 2000). The advantage of SVM is that its accuracy is not diminished when a problem involves a large number of features (Drucker et al., 1999). Several SVM approaches have been applied to spam filtering (Blanzieri and Bryl, 2007; Sculley and Wachman, 2007). Likewise, decision trees, which classify samples using automatically learned rule-sets (i.e., tests) (Quinlan, 1986), have also been used for spam filtering (Carreras and Márquez, 2001). All of these machine-learning-based spam filtering approaches are known as statistical content-based approaches (Zhang et al., 2004).

Machine-learning approaches model e-mail messages using the Vector Space Model(VSM) (Salton et al., 1975). The VSM is an algebraic approach for Information Filtering (IF), Information Retrieval (IR), indexing and ranking. This model represents natural language documents mathematically as vectors in a multidimensional space where the axes are terms within messages. As in any other IR system, the VSM is affected by the characteristics of the text, with one of those characteristics being *word sense ambiguity* (Sanderson, 1994). The use of ambiguous words can confuse the model, permitting spammers to bypass spam filters.

We propose here the application of WSD for spam filtering to recover the filtering capabilities of content-based methods. Our approach pre-processes

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

e-mails disambiguating the terms before constructing the VSM. Thereafter, based on this representation, we train several supervised machine-learning algorithms to detect and filter junk e-mails. In summary, we advance the state of the art through the following contributions:

- We present a method to disambiguate terms in e-mail messages.
- We provide an empirical validation of our method with an extensive study of several machine-learning classifiers.
- We show that the proposed method improves filtering rates; we discuss the weakness of the model and explain possible enhancements.

The remainder of this paper is organised as follows. Section 2 addresses the impact of electronic undesired mail on e-commerce. Section 3 describes the problem of WSD and the effects that ambiguity has on spam filtering systems. Section 4 introduces our method to improve detection rates by using WSD. Section 5 provides an empirical evaluation of the experiments performed and presents the results. Section 6 presents the main limitations of the proposed method and proposes possible enhancements. Finally, Section 7 presents the conclusions and outlines the avenues for future work.

2. Impact of undesired e-mail on e-commerce

Spam is a serious issue in the e-commerce arena, affecting many actors from the end users, to business offering commerce opportunities, to intermediaries. Correctly identifying spam, can have an impact on e-commerce, since false positives result the recipient not receiving legitimate e-mails (e.g., those used to conduct an advertising campaign chosen by the user itself), while false negatives can leave the recipient susceptible to spam attacks such as phishing.

On a thorough report back in 2004, Cashell et al. (2004) brought together different statistics on the economic impact of cyber-attacks. This report includes the analysis of a British firm, called Mi2g, which publishes analysis from the collection of data from 7,000 hacker groups worldwide, providing detailed monthly and year-to-date information on: digital attack hot spots, emerging threats to digital security, economic damage estimates, top hacker groups, most vulnerable operating systems and trends for vulnerabilities, spam, malware and denial of service attacks. Under the economic damage

analysis, they include the estimation of the incidence and cost of what they call "overt digital attacks"[4]. Figure 1 shows the cost estimates for those digital attacks, which include hacking, malware and spam, from 1996 to 2003.

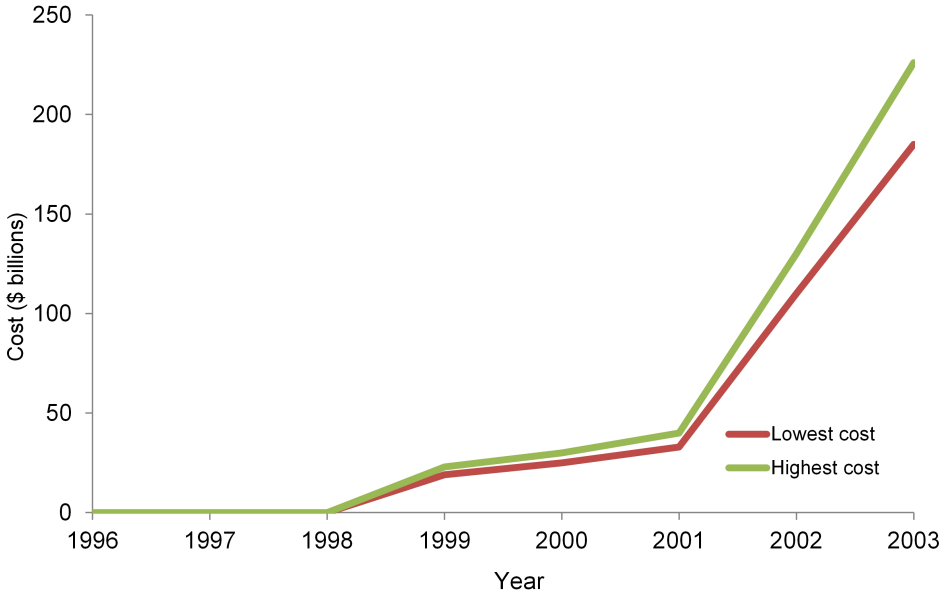

Figure 1: Economic damage estimates for all forms of digital attacks worldwide, based on business interruption, denial of service, data theft or deletion, loss of sensitive intelligence or intellectual property, loss of reputation, and share price declines. Source: Mi2g, Frequently Asked Questions: SIPS and EVEDA, v1.00.

Trying to break down the numbers, in another study, Hansell (2003) states that in 2003 the volume of spam, which was growing rapidly, implied worldwide costs exceeding 20 billion US dollars annually. And that is with "only" an estimated volume of 50% of e-mail being spam. Nowadays more than 86% of received e-mails are spam. In this way, although the numbers correspond

[4]Mi2g defines an overt digital attack as one in which a hacker group gains unauthorized access to a computer network and modifies any of its publicly visible components. Overt attacks may include either data attacks, where the confidentiality, authenticity, or integrity of data is violated, or control attacks, where network control or administrative systems are compromised. Overt attacks are those that become public knowledge, as opposed to covert attacks, which are known only to the attacker and the victim.

to some years back in time, the projections to current days, according to the increase of users with access to new technologies and the growth that electronic commerce has experienced, can be overwhelming.

Supporting that theory, in a more recent study, Smith et al. (2011) analyse the impact of cybercrime on marketing activity and shareholder value. Their results indicate that costs of cybercrime go beyond the tangible issues (e.g., stolen assets, business losses or damages on company reputation), having significant negative effect on shareholder value. The explanation to that fact, resides on the worries of users about security of their business transactions with companies that fall prey to cyber criminals. Such vulnerabilities result in a decrease of the trust from the user, causing the company to lose future business and, hence, raising the concerns of financial analysts, investors and creditors.

In a similar vein, other recent studies show the influence and impact of spam in several companies that suffered from considering their e-mail advertisement as a spam (Mostafa Raad et al., 2010) or the plague problem that the, in words of the on-line market research company e-Marketer, "killer-app of the on-line advertising world" (i.e., e-mail) is suffering as a result of spam (Gopal et al., 2011).

3. The Problem of Disambiguation

The task of disambiguating word sense is the process of identifying the most appropriate meaning of a polysemous word given a specific context. The Word Sense Disambiguation (WSD) problem has been a topic of interest and concern since the 1950s when Natural Language Processing (NLP) tasks became a reality. Indeed, it was already conceived as a fundamental task of Machine Translation (MT) in the late 1940s (Weaver, 1949). Very soon it became clear that if would be extremely difficult to solve (Bar-Hillel, 1960) and would be one of the main problems of MT. Furthermore, WSD has been described as 'AI-complete' (Mallery, 1988), that is, a problem that can be solved only by first resolving all of the difficult problems in artificial intelligence (AI), such as the representation of common sense and encyclopaedic knowledge.

The difficulty with sense disambiguation is not limited to a single cause, but arises from a variety of factors. First, the task lends itself to different formalisations due to fundamental questions, such as the approach to the representation of a word sense (ranging from an enumeration of a finite

set of senses to a rule-based generation of new ones), the granularity of sense inventories (from subtle distinctions to homonyms), the domain-oriented versus unrestricted nature of texts, and the set of target words to disambiguate (one target word per sentence vs. an 'all-words settings) (Navigli, 2009).

Second, WSD has strong dependence on previously-acquired knowledge. In fact, the skeletal procedure of any WSD system can be summarised as follows: given a set of words (e.g., a sentence or a group of words), a technique is applied that makes use of one or more sources of knowledge to associate the most appropriate senses with the words in context (Navigli, 2009).

Knowledge dependence was a serious impediment before the release of large-scale lexical resources to enable the automation of knowledge extraction systems (Wilks et al., 1990). Nowadays, this task is more attainable owing to the existence of resources such as *WordNet* (Fellbaum et al., 1998), a lexical database for the English language that groups words into sets of synonyms and records the semantic relations between the sets.

From the 1990s to the present, we have seen a large application of statistical methods for WSD systems (Ide and Véronis, 1998), and WSD has become an increasingly popular area of computational linguistics research in the past few years (Agirre and Edmonds, 2007). This is particularly due to *Senseval*[5], which has the purpose of evaluating the strengths and weaknesses of such applications with respect to different words, different varieties of language, and different languages; and provides evaluation exercises and standard datasets for the task.

Several studies have shown poor outcomes for the application of WSD to IR (Sanderson, 1994; Voorhees, 1999). Nevertheless, works such as (Krovetz, 1997; Gonzalo et al., 1999; Krovetz, 2002) and the often-cited (Krovetz and Croft, 1992), even though they have often been interpreted as saying the opposite, support the potential for improved IR performance using WSD.

The extended use of IR in combination with naïve Bayesian classifiers for spam filtering (Sahami et al., 1998; Androutsopoulos et al., 2000a,b,c; Schneider, 2003; Zhang et al., 2004), presents an ambiguity problem for the anti-spam solutions that should be taken into account. However, the problem of a term ambiguity has not reached the security industry for spam-filtering tasks.

[5]http://www.senseval.org

4. Our Word Sense Disambiguation Approach

Today's attacks against Bayesian spam filters attempt to keep the content of spam mail visible to humans, but obscured to filters. For instance, attackers circumvent these filters by replacing suspicious words by innocuous terms with the same meaning (Karlberger et al., 2007; Nelson et al., 2009; Santos et al., 2012). In a similar vein, these spam-filtering systems do not take into account the possible existence of ambiguous terms within e-mail messages. This could lead to misclassified legitimate e-mails and spammers evading filtering, since it is expected that incorrectly disambiguated words may entail noise (Mavroeidis et al., 2005) and decrease the classification accuracy (Xu and Yu, 2010). To solve this issue, we apply WSD to spam filtering a pre-processing procedure that is able to disambiguate confusing terms, to improve the capabilities of anti-spam systems.

Our approach utilises *SenseLearner* (Mihalcea and Csomai, 2005), a state-of-the-art minimally supervised WSD system that attempts to disambiguate all content words in a text using *WordNet* senses. Because *SenseLearner* needs a pre-processing stage in which the text is annotated with part-of-speech (PoS) tags, our e-mail message dataset was previously tagged using *Freeling* (Carreras et al., 2004), a suite of analysis tools based on the architecture of (Carreras and Padró, 2002).

While the PoS tagging had no special parameters worthy of comment, the WSD task offered several options that should be mentioned. First, in cases where the system was unable to make a prediction, we chose to mark the word with the most frequent sense from *WordNet* (sense 1) by activating the *default* option. This option improved the results (see Section 5) by generalising non-clear terms' meanings, avoiding the loss of their sense.

Second, the training data consisted of sense annotated texts, formatted by following the SemCor XML format. We used for our experiments the models provided with the distribution of *SenseLearner*, which were trained on SemCor, and a separate training instance base was built for each model. These models implement the following features:

1. For **nouns**:
    - A contextual model that relies on the first noun, verb, or adjective before the target noun and the corresponding PoS.
    - A collocation model that implements collocation-like features based on the first word to the left and the first word to the right of the

target word.

2. For **verbs**:
    - A contextual model that relies on the first word before and the first word after the target verb and its PoS.
    - A collocation model that implements collocation-like features based on the first word to the left and the first word to the right of the target word.
3. For **adjectives**:
    - Contextual model 1, which relies on the first noun after the target adjective.
    - Contextual model 2, which relies on the first word before and the first word after the target adjective and its PoS.
    - A collocation model that implements collocation-like features based on the first word to the left and the first word to the right of the target word.

Finally, although *SenseLearner* offers two different input methods, SemCor (Miller et al., 1993) and PoS tagging, we chose the second due to its simplicity. However, the use of SemCor for future experiments is discussed in Section 6.

In this way, we formally define an e-mail $\mathcal{M}$ as a set composed of n terms t_i, $\mathcal{M} = \{t_1, t_2, \ldots, t_{n-1}, t_n\}$, where each term corresponds to a word (although we are aware of the possibility of applying WSD to collocations, we decided to leave this strength to future improvements of our system). Each t_i has a set of n senses s_i, $s = \{s_1, s_2, \ldots, s_{n-1}, s_n\}$. WSD selects the corresponding s_i for each term and generates a new relation of term-sense $t_{i,j}$, where i indicates the term and j denotes its corresponding sense.

Our method builds a model with term-sense relations, which we use to train several machine-learning classification algorithms. In order to perform this training, we first create an *ARFF* file (attribute relation file format) (Holmes et al., 1994) that describes the shared attributes (e.g., term-sense) for each instance (e.g., document). Secondly, we use the *Waikato Environment for Knowledge Analysis* (WEKA) (Garner, 1995) to build the desired classifiers. Finally, we test different machine-learning classification algorithms with WEKA as described in Section 5.

5. Empirical Evaluation

We employed the *Ling Spam* dataset[6] and the *TREC 2007 Public Corpus*[7] separately, as the spam corpus in two different experiments, applying our proposed approach to both of them. Ling Spam comprises both spam and legitimate messages retrieved from the *Linguistic list*, an e-mail distribution list focusing on *linguistics*. The dataset consists of 2,893 different e-mails, of which 2,412 are legitimate e-mails obtained by downloading digests from the linguistic list and 481 are spam e-mails retrieved from one of the authors' inbox (a more detailed description of the corpus is provided in (Androutsopoulos et al., 2000a; Sakkis et al., 2003)). *Stop Word Removal* (Wilbur and Sirotkin, 1992) and *stemming* (Lovins, 1968) were performed on the e-mails, generating the following four different datasets:

1. **Bare:** In this dataset, HTML tags, separation tokens, and duplicate e-mails were removed from messages.
2. **Lemm:** In addition to the removal pre-processing 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 a combination of both stemming and stop-word removal processes.

We did not use the *lemm* or *lemm_stop* datasets. Additionally, instead of using the *stop* dataset we used the *bare* dataset and performed a stop word removal based on an external stop-word list[8].

TREC 2007 Public Corpus (Cormack, 2007) contains all e-mail messages delivered to a server from April 8 through July 6, 2007. The server contained many accounts that had fallen into disuse but that continued receiving a lot of spam. To these accounts were added a number of 'honeypot' accounts published on the web and used to sign up for a number of services, some

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

[7] http://plg.uwaterloo.ca/~gvcormac/spam

[8] http://www.webconfs.com/stop-words.php
Alternative link:
http://paginaspersonales.deusto.es/claorden/resources/
EnglishStopWords.txt

legitimate and some not. The dataset contains 75,419 messages, of which 25,220 are legitimate e-mail and 50,199 are junk messages, divided into three subcorpora (Cormack, 2007):

- **trec07p/full/** - immediate, full feedback
- **trec07p/delay/** - feedback only for the first 10,000 messages
- **trec07p/partial/** - feedback only for 30,388 messages corresponding to one recipient

For our experiments, we randomly extracted 30% (due to computational limitations) of the full subcorpora, maintaining the spam-legitimate ratio. In this way, our TREC dataset comprises 7,653 legitimate e-mails and 14,973 junk messages.

In both experiments, with Ling Spam and TREC, we modelled three different datasets using the VSM (Salton et al., 1975). The first dataset corresponded to the raw e-mails with no modification except for the stop word removal. The second dataset had a pre-processing step of WSD without the *default* option (see Section 4) that marked unpredictable senses for the word with the most frequent sense from *WordNet*. Finally, the third dataset had a WSD pre-processing step but with the *default* option activated. We also used the *Term Frequency – Inverse Document Frequency* (TF–IDF) (Salton and McGill, 1983) weighting schema, where the weight of the i^{th} term in the j^{th} document, denoted by $weight(i, j)$, is defined by:

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

The *term frequency* $tf_{i,j}$ is defined as:

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

where $n_{i,j}$ is the number of times the term $t_{i,j}$ appears in a document d, and $\sum_k n_{k,j}$ is the total number of terms in the document d. The inverse term frequency idf_i is defined as:

$$idf_i = \frac{|\mathcal{D}|}{|\mathcal{D} : t_i \in d|} \tag{3}$$

where $|\mathcal{D}|$ is the total number of documents and $|\mathcal{D} : t_i \in d|$ is the number of documents containing the term t_i.

We also extracted from the model the top 1,000 attributes using *Information Gain* (Kent, 1983), an algorithm that evaluates the relevance of an attribute by measuring the information gain with respect to the class:

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

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 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 that the training dataset belongs to class C_i. Figures 2 and 3 show the frequency of senses for the 1,000 attributes selected with IG for each of the datasets. Must be noted that some terms have no sense because they correspond to abbreviations, URLs, brand names or even to words that have suffered a transformation and can't be identified by WordNet. In the case of the TREC dataset, the number of terms without a sense increases considerably, what leads us to believe that a further study on the retrieval of the corpus, identifying errors or modifications in the words, could improve significantly our obtained results.

After removing the less significant attributes, the resultant files are used as training datasets for the classifiers. In this way, we obtained three datasets corresponding to: a non-disambiguated set of e-mails, a disambiguated set of e-mails with the default option set off, and a disambiguated set of e-mails with the default option set on.

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

- **Cross validation:** 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 datasets were 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:

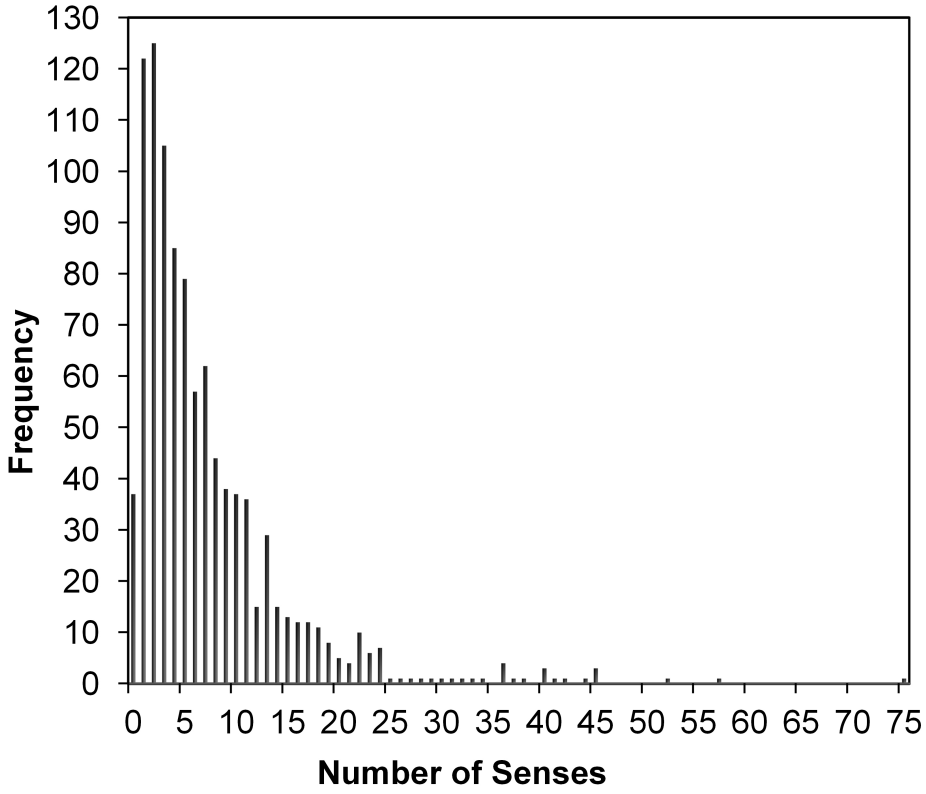

Figure 2: Frequency of senses for all selected top IG scoring attributes for Ling Spam dataset.

- *Bayesian Networks:* In order to train Bayesian networks, we used different structural learning algorithms; *K2* (Cooper and Herskovits, 1991), *Hill Climber* (Russell and 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.

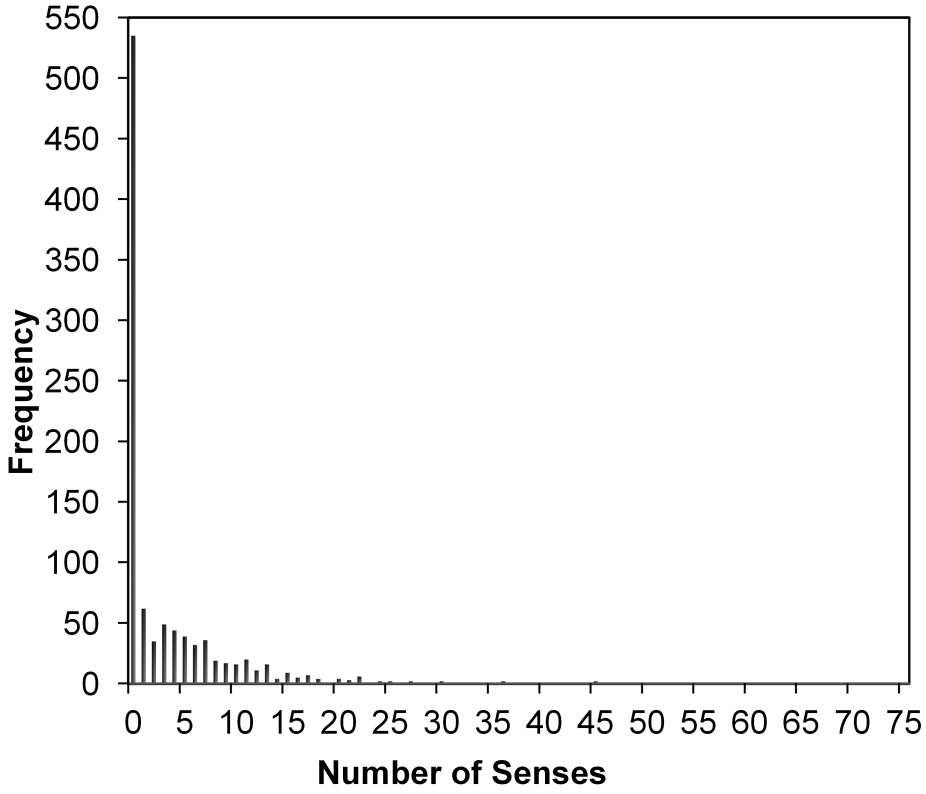

Figure 3: Frequency of senses for all selected top IG scoring attributes for TREC dataset.

– *Support Vector Machines:* We used a *Sequential Minimal Optimisation* (SMO) algorithm (Platt, 1999) with a *polynomial kernel* (Amari and Wu, 1999), a *normalised polynomial kernel* (Amari and Wu, 1999), a *Pearson VII function-based* universal kernel (Üstün et al., 2006), and a *Radial Basis Function* (RBF) based kernel (Amari and Wu, 1999). In addition, we used LibSVM[9] for the linear (i.e., hyperplane) and sigmoid kernel (Lin and Lin, 2003) implementation.

- **Testing the models:** To measure the processing overhead of the model, we measure the required training and testing times:

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

– *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 the results, 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:

$$S_P = \frac{N_{s\to s}}{N_{s\to s} + N_{l\to s}} \tag{5}$$

where $N_{s\to s}$ is the number of correctly classified spam messages 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:

$$S_R = \frac{N_{s\to s}}{N_{s\to s} + N_{s\to l}} \tag{6}$$

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

$$\textit{F-measure} = \frac{2N_{s\to s}}{2N_{s\to s} + N_{s\to l} + N_{l\to s}} \tag{7}$$

In addition, we measured the accuracy, which is the number of the classifier's hits divided by the total number of classified instances:

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

Finally, 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), where

the TPR is the number of spam messages correctly detected divided by the total number of junk e-mails:

$$TPR = \frac{TP}{TP + FN} \tag{9}$$

and the FPR is the number of legitimate messages misclassified as spam divided by the total number of legitimate e-mails:

$$FPR = \frac{FP}{FP + TN} \tag{10}$$

Tables 1 and 2 show training and testing times for the different machine-learning classifiers. The kNN algorithm needs almost no time for training but is the slowest classifier in the testing phase. SVM lineal, SVM sigmoid and SVM with polynomial kernel configurations for SVM were the fastest in both the training and testing phases. Naïve Bayes performed well in both phases, being the second fastest classifier after kNN in the training phase for TREC and offering testing times of 0.34-0.36 nanoseconds for Ling Spam and 0.98-0.99 nanoseconds for TREC. The performance of the Bayesian networks depended on the algorithm used. Overall, we found that K2 is the fastest Bayesian classifier to train, while the testing times are quite similar for all of them. Among the decision trees, Random Forest with 10, 50 and 100 trees trained faster than J48 only when using the Ling Spam dataset, while in the testing phase, the fastest one was J48 for both datasets.

Tables 3 and 4 show the results for the classifiers in terms of precision and recall. The kNN algorithm showed generally similar behaviour regarding precision with both the original model and the disambiguated models when testing with Ling Spam, never reaching our approach in the accuracy obtained with the original one, but showing statistically significant improvements for each kNN configuration when testing with TREC. In terms of

recall, it is noteworthy that the kNN algorithm improves in most cases with Ling Spam when using the disambiguated models and always improves significantly, when using the disambiguated model with the default option activated, maintaining the same values as the TREC dataset. The configurations tested with the SVM algorithm for Ling Spam show the same precision for all models, and significantly improve when testing with TREC and using the disambiguated model with the default option activated. The recall significantly improves in four of the six configurations tested for Ling Spam, again with the default option selected, but has significant degradation with TREC. Decision trees show similar behaviour with all models in both accuracy and recall when testing the Ling Spam dataset. However, when testing with TREC, experiments for each configuration show a significant improvement in terms of precision while maintaining the same recall. Bayesian networks trained with K2 and Hill Climber show significant degradation with the Ling Spam dataset when applying disambiguation, for both precision and recall, only maintaining the precision values for TREC. Instead, when training with the TAN algorithm, the precision for Ling Spam is preserved, or improved with the default option set on, also improving significantly for TREC, with the recall suffering no significant variation for either dataset. Finally, naïve Bayes presents a vast improvement in precision when testing Ling Spam and maintains the improvement for TREC, using the model with the pre-processing step of disambiguation and the default option activated, almost preserving the same recall levels for Ling Spam but with significant degradation with the TREC dataset.

Finally, Table 5 offers the results for the area under the ROC curve (AUC), which indicates the classifiers with the best balance between correct positives and false positives. The best balance is achieved by decision trees, showing no significant variation between the different models when testing Ling Spam, but with a significant improvement in the disambiguated model with the default option activated when testing the TREC dataset. The kNN algorithm shows significant improvements for both datasets tested with the disambiguated model with the default option activated. SVM significantly improved when compared to the original model, also with the default option selected, except for the Pearson VII kernel configuration for Ling Spam, which suffered significant degradation. The Bayesian networks again significantly improved with the disambiguated model with the default option set

on when testing the TREC dataset, but only maintaining the proper balance of the original model with the TAN algorithm for Ling Spam. Finally, naïve Bayes again shows an improvement, significant for TREC, when using the disambiguated model.

We can make several observations from the experimental evaluation. First, almost every classifier experienced an improvement for both datasets when testing the disambiguated model with the default option activated, most of them with statistically significant improvements. However, when testing the model disambiguated with the default option deactivated, the results suffered significant degradation. In this way, comparing the original model with the disambiguated one with the default option activated, the results show an overall improvement in the detection capabilities. In particular, the most widespread among spam filtering systems, the naïve Bayes classifier, experienced a substantial improvement for Ling Spam, when applied the disambiguation pre-processing. The Bayesian networks with the TAN learning algorithm also offer an improvement when applying our model but this was only significant for the TREC dataset. Furthermore, all of the SVM configurations obtained good results, which again show as statistically significant improvement when applied the disambiguation of terms, with the only exception being SVM with Pearson VII for Ling Spam. Finally, the decision trees, which show good results especially with the Random Forest algorithm implementation, are only influenced by the use of our model when testing with the TREC dataset.

6. Discussion

The results obtained during the evaluation of our approach show that the pre-processing step of Word Sense Disambiguation, applied to a model that represents electronic mail for anti-spam systems, improves filtering rates. In addition, we keep the false positives (legitimate e-mails incorrectly classified as spam) to a minimum, sometimes even reducing them, while detecting a large number of junk e-mails. Regarding the results of each classifier, it is noteworthy that the recall of the naïve Bayes classifier decreases substantially for the TREC dataset, showing a weakness against larger and less domain-oriented datasets than Ling Spam. On the other hand, decision trees with the Random Forest algorithm implementation show themselves as the most suitable to address the problem of spam both because of the level of spam detection and, above all, their low levels of false positives. However, there

are several important points to be discussed referring to the appropriateness of using our proposed method.

First, we employ a very basic input format, PoS labelling, for the disambiguation process. There are more complex formats such as SemCor (Miller et al., 1993) that can provide more information for this step and can result in a richer disambiguation of terms. On the other hand, based on our results, we see that labelling all the words with at least the most frequent WordNet sense (default option enabled), when the system is not able to make a prediction, offers better filtering rates.

Second, by including Word Sense Disambiguation in spam filtering, there is a problem derived from Information Retrieval and Natural Language Processing when dealing with semantics: the dependence of language (Bates and Weischedel, 1993). This language dependency complicates the acquisition of training datasets to feed the learning models. However, this problem is enhanced by the continuous evolution and changing nature of spam. It is almost impossible for a system with global aspirations to obtain current samples that cover all natural languages used by spammers.

These limitations to our approach imply the need to find alternative methods for spam filtering. The Topic Detection and Tracking (TDT) method is a technique that should be considered. The TDT method assumes multiple sources of information and assumes that the information flowing from each source is divided into a sequence of stories, which may provide information on one or more topics (or events) (Allan et al., 1998). The general task is to identify the events being discussed in these stories, in terms of the stories that describe them. Stories that describe unexpected events will of course follow the event, whereas stories on expected events can both precede and follow the event. The application of this technique to spam filtering is clear if we expect the evolution of spam to be cyclical in many cases (false Christmas greetings in December) and to adapt well to different popular events of great impact (spam over the World Cup in South Africa). For these reasons, we believe it would be interesting to study the TDT method in detail, to examine its applicability to future unsolicited bulk e-mail filtering systems.

7. Concluding Remarks

Electronic mail (e-mail) is a powerful communication channel. However, as with any technology, electronic mail is vulnerable to malicious use. Spam is not only unpleasant for e-mail users, but is a major problem for digital

security and worldwide economy, having a great negative impact on market value of global firms and influencing marketing campaigns of legitimate companies that suffer from considering their e-mail advertisement as spam.

There is a great need to improve the undesired e-mail detection, in order to restore the confidence of users in electronic commerce, since it is demonstrated that the use of e-mail marketing is risky due to the spam problem (Mostafa Raad et al., 2010). Despite their ability to detect spam, traditional methods of spam filtering based on machine learning are not able to take into account the semantic layer of e-mails.

In this paper, we have presented the application of Word Sense Disambiguation for spam filtering to improve the detection capabilities of content-based methods. Our approach pre-processes the e-mail messages, disambiguating the terms before constructing the Vector Space Model. Our experiments show that this approach provides high rates of spam filtering while maintaining a low number of legitimate e-mails that are incorrectly classified.

Future versions of this filtering system will follow three main directions. First, in the future, we will analyse a variety of existing disambiguation techniques and study their advantages in applications of our model. Second, we plan to improve the process of disambiguation using the SemCor format to feed the disambiguation system. Finally, we will consider adding techniques that detect and track events for spam filtering.

References

Agirre, E., Edmonds, P., 2007. Word Sense Disambiguation: Algorithms and Applications. Springer Publishing Company, Incorporated.

Allan, J., Carbonell, J., Doddington, G., Yamron, J., Yang, Y., et al., 1998. Topic detection and tracking pilot study: Final report. In: Proceedings of the DARPA broadcast news transcription and understanding workshop. Vol. 1998. Citeseer.

Amari, S., Wu, S., 1999. Improving support vector machine classifiers by modifying kernel functions. Neural Networks 12 (6), 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 23^{rd} 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 4^{th} European Conference on Principles and Practice of Knowledge Discovery in Databases.

Bar-Hillel, Y., 1960. The Present Status of Automatic Translation of Languages. Vol. 1. pp. 91–163.

Bates, M., Weischedel, R., 1993. Challenges in natural language processing. Cambridge Univ Pr.

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, 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 (1), 5–32.

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

Carreras, X., Chao, I., Padró, L., Padró, M., 2004. Freeling: An open-source suite of language analyzers. In: Proceedings of the 4th LREC. Vol. 4.

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. Citeseer, pp. 58–64.

Carreras, X., Padró, L., 2002. A flexible distributed architecture for natural language analyzers. In: Proceedings of the LREC. Vol. 2.

Cashell, B., Jackson, W. D., Jickling, M., Webel, B., 2004. The economic impact of cyber-attacks. Congressional Research Service, Library of Congress.

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., 2007. TREC 2007 spam track overview. In: Sixteenth Text REtrieval Conference (TREC-2007).

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

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

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.

Gonzalo, J., Penas, A., Verdejo, F., 1999. Lexical ambiguity and information retrieval revisited. In: Proceedings of EMNLP/VLC. Vol. 99.

Gopal, R., Tripathi, A., Walter, Z., 2011. Economic issues in advertising via e-mail: Role for a trusted third party? Marketing, 1pp.

Hansell, S., 2003. Diverging estimates of the costs of spam. New York Times.

Heron, S., 2009. Technologies for spam detection. Network Security 2009 (1), 11–15.

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

Ide, N., Véronis, J., 1998. Word sense disambiguation: The state of the art. Computational Linguistics 24, 1–40.

Jagatic, T., Johnson, N., Jakobsson, M., Menczer, F., 2007. Social phishing. Communications of the ACM 50 (10), 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. ACM New York, NY, USA, pp. 370–375.

Karlberger, C., Bayler, G., Kruegel, C., Kirda, E., 2007. Exploiting redundancy in natural language to penetrate bayesian spam filters. In: WOOT '07: Proceedings of the first USENIX workshop on Offensive Technologies. USENIX Association, Berkeley, CA, USA, pp. 1–7.

Kent, J., 1983. Information gain and a general measure of correlation. Biometrika 70 (1), 163.

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

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

Krovetz, B., 2002. On the importance of word sense disambiguation for information retrieval. In: Proc. of LREC Workshop on Creating and Using Semantics for Information Retrieval and Filtering, Las Palmas. Citeseer.

Krovetz, R., 1997. Homonymy and polysemy in information retrieval. In: Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics. Association for Computational Linguistics, pp. 72–79.

Krovetz, R., Croft, W., 1992. Lexical ambiguity and information retrieval. ACM Transactions on Information Systems (TOIS) 10 (2), 115–141.

Leung, C., Liang, Z., 2009. An analysis of the impact of phishing and anti-phishing related announcements on market value of global firms. HKU Theses Online (HKUTO).

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. Tech. rep., Nat'l Taiwan University.

Lovins, J., 1968. Development of a Stemming Algorithm. Mechanical Translation and Computational Linguistics 11 (1), 22–31.

Mallery, J. C., 1988. Thinking about foreign policy: Finding an appropriate role for artificially intelligent computers. In: Master's thesis, M.I.T. Political Science Department.

Mavroeidis, D., Tsatsaronis, G., Vazirgiannis, M., Theobald, M., Weikum, G., 2005. Word sense disambiguation for exploiting hierarchical thesauri in text classification. Knowledge Discovery in Databases: PKDD 2005, 181–192.

Mihalcea, R., Csomai, A., 2005. Senselearner: Word sense disambiguation for all words in unrestricted text. In: Proceedings of the ACL 2005 on Interactive poster and demonstration sessions. Association for Computational Linguistics, p. 56.

Miller, G., Leacock, C., Tengi, R., Bunker, R., 1993. A semantic concordance. In: Proceedings of the workshop on Human Language Technology. Association for Computational Linguistics, pp. 303–308.

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

Mostafa Raad, N., Alam, G., Zaidan, B., Zaidan, A., 2010. Impact of spam advertisement through e-mail: A study to assess the influence of the anti-spam on the e-mail marketing. African Journal of Business Management 4 (11), 2362–2367.

Navigli, R., 2009. Word sense disambiguation: A survey. ACM Comput. Surv. 41 (2), 1–69.

Nelson, B., Barreno, M., Jack Chi, F., Joseph, A., Rubinstein, B., Saini, U., Sutton, C., Tygar, J., Xia, K., 2009. Misleading learners: Co-opting your spam filter. Machine Learning in Cyber Trust, 17–51.

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

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

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

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. Vol. 62. Madison, Wisconsin: AAAI Technical Report WS-98-05, pp. 98–05.

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 (1), 49–73.

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 (11), 613–620.

Sanderson, M., 1994. Word sense disambiguation and information retrieval. In: SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval. Springer-Verlag New York, Inc., New York, NY, USA, pp. 142–151.

Santos, I., Laorden, C., Sanz, B., Bringas, P. G., 2012. Enhanced topic-based vector space model for semantics-aware spam filtering. Expert Systems With Applications 39 (1), 437–444, doi:10.1016/j.eswa.2011.07.034.

Schneider, K., 2003. 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. pp. 307–314.

Sculley, D., Wachman, G., 2007. Relaxed online SVMs for spam filtering. In: Proceedings of the 30^{th} 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), 1–47.

Seewald, A., 2007. An evaluation of naive Bayes variants in content-based learning for spam filtering. Intelligent Data Analysis 11 (5), 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 (2), 183–193.

Smith, K. T., Smith, M., Smith, J. L., 2011. Case studies of cybercrime and their impact on marketing activity and shareholder value. Academy of Marketing Studies Journal.

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 (1), 29–40.

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

Voorhees, E., 1999. Natural language processing and information retrieval. Information Extraction, 724–724.

Weaver, W., 1949. Translation. W. N. Locke and A. D. Booth, Eds. Technology Press of MIT, Cambridge, MA, and John Wiley & Sons, New York, NY, pp. 15–23.

Wilbur, W., Sirotkin, K., 1992. The automatic identification of stop words. Journal of information science 18 (1), 45–55.

Wilks, Y., Fass, D., ming Guo, C., McDonald1, J. E., Plate, T., Slator, B. M., 1990. Providing machine tractable dictionary tools. Machine Translation 5 (2), 99–154.

Xu, H., Yu, B., 2010. Automatic thesaurus construction for spam filtering using revised back propagation neural network. Expert Systems with Applications 37 (1), 18–23.

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

Table 1: Training time results of machine-learning classifiers with and without disambiguation. The VSM columns correspond to the non-modified dataset, the WSD columns correspond to the dataset pre-processed with WSD with the default option set off, and the WSD_def columns correspond to the pre-processed dataset with WSD and the default option set on.

Training Time (ns)						
Classifier	Ling Spam			TREC		
	VSM	WSD	WSD_def	VSM	WSD	WSD_def
DT: J48	116.37	107.24	118.02	88.78	88.21	90.35
DT: RF N=10	7.99	8.04	7.38	303.23	321.06	251.88
DT: RF N=50	39.81	40.49	37.10	1468.00	1650.09	1348.24
DT: RF N=100	80.36	82.10	74.22	3232.53	3591.30	2127.36
DT: RF N=150	120.70	122.99	112.28	3922.16	4861.16	2824.28
DT: RF N=200	162.26	166.36	151.98	5057.17	5879.32	3550.75
Naïve Bayes	8.21	10.60	8.58	38.03	36.83	36.49
BN: K2	14.98	15.69	19.01	81.58	79.49	79.64
BN: Hill Climber	1531.41	1439.85	1417.50	3502.05	3524.48	3519.06
BN: TAN	1330.79	1573.70	1544.56	3333.98	3282.99	3206.47
Knn K=1	0.00	0.00	0.00	0.03	0.04	0.03
Knn K=2	0.00	0.00	0.00	0.03	0.03	0.03
Knn K=3	0.00	0.00	0.00	0.03	0.03	0.03
Knn K=4	0.00	0.00	0.00	0.03	0.03	0.03
Knn K=5	0.00	0.00	0.00	0.03	0.03	0.03
SVM: Lineal	2.76	3.02	3.94	96.71	72.31	73.75
SVM: Sigmoid	3.89	3.85	3.87	72.67	70.92	66.62
SVM: Polynomial	1.36	1.65	1.54	145.88	149.89	153.92
SVM: Norm Polynom	42.98	43.47	41.02	1038.56	1010.57	1022.76
SVM: Pearson VII	87.19	89.01	101.85	1070.08	993.51	1009.98
SVM: RBF	14.81	14.37	15.85	1451.73	1443.26	1244.30

Table 2: Testing time results of machine-learning classifiers with and without disambiguation. The VSM columns correspond to the non-modified dataset, the WSD columns correspond to the dataset pre-processed with WSD with the default option set off, and the WSD_def columns correspond to the pre-processed dataset with WSD and the default option set on.

Testing Time (ns)						
Classifier	Ling Spam			TREC		
	VSM	WSD	WSD_def	VSM	WSD	WSD_def
DT: J48	0.00	0.00	0.00	0.05	0.11	0.08
DT: RF N=10	0.01	0.01	0.01	0.15	0.14	0.11
DT: RF N=50	0.04	0.04	0.04	1.03	1.08	0.95
DT: RF N=100	0.08	0.09	0.09	3.05	3.18	2.01
DT: RF N=150	0.12	0.13	0.14	4.27	4.33	3.57
DT: RF N=200	0.18	0.20	0.19	5.34	5.39	4.56
Naïve Bayes	0.36	0.33	0.34	0.99	0.98	0.98
BN: K2	0.23	0.23	0.25	0.68	0.71	0.69
BN: Hill Climber	0.19	0.17	0.18	0.65	0.67	0.67
BN: TAN	0.23	0.26	0.24	1.00	0.99	0.93
Knn K=1	4.34	4.41	5.48	23.61	22.02	22.75
Knn K=1	4.34	4.41	5.48	25.53	24.03	24.77
Knn K=2	5.25	5.45	6.32	26.93	24.99	26.20
Knn K=4	5.17	5.96	7.05	27.89	26.09	27.13
Knn K=5	5.96	6.35	7.63	28.87	26.86	27.94
SVM: Lineal	0.15	0.16	0.18	0.96	0.86	0.81
SVM: Sigmoid	0.24	0.26	0.29	6.63	6.67	6.43
SVM: Polynomial	0.02	0.02	0.02	0.08	0.07	0.08
SVM: Norm Polynom	1.02	1.10	1.12	16.54	16.34	16.65
SVM: Pearson VII	2.01	2.04	2.44	20.07	19.37	19.54
SVM: RBF	0.42	0.44	0.49	19.43	18.95	17.16

Table 3: Precision evaluation of machine-learning classifiers. The VSM columns correspond to the non-modified dataset, the WSD columns correspond to the dataset pre-processed with WSD with the default option set off, and the WSD_def columns correspond to the pre-processed dataset with WSD and the default option set on.

Precision						
Classifier	Ling Spam			TREC		
	VSM	WSD	WSD_def	VSM	WSD	WSD_def
DT: J48	0.88	0.86	0.88	0.86	0.86	✓0.89
DT: RF N=10	0.98	0.98	0.98	0.91	x 0.91	✓0.93
DT: RF N=50	0.99	0.99	0.99	0.91	x 0.91	✓0.93
DT: RF N=100	0.99	1.00	1.00	0.91	x 0.91	✓0.93
DT: RF N=150	1.00	1.00	1.00	0.91	x 0.91	✓0.93
DT: RF N=200	1.00	1.00	1.00	0.91	x 0.91	✓0.93
Naïve Bayes	0.64	0.63	✓0.72	0.97	0.96	0.97
BN: K2	0.99	0.98	x 0.96	1.00	1.00	1.00
BN: Hill Climber	0.99	0.98	x 0.96	1.00	1.00	1.00
BN: TAN	0.94	0.94	0.96	0.88	x 0.88	✓0.90
Knn K=1	1.00	0.99	0.97	0.91	x 0.91	✓0.92
Knn K=2	0.98	0.91	x 0.64	0.90	0.90	✓0.92
Knn K=3	1.00	0.99	0.99	0.90	0.90	✓0.92
Knn K=4	0.99	0.94	0.97	0.90	0.90	✓0.92
Knn K=5	1.00	0.99	1.00	0.90	0.90	✓0.92
SVM: Lineal	0.99	0.99	0.99	0.89	x 0.88	✓0.91
SVM: Sigmoid	1.00	1.00	1.00	0.87	x 0.86	✓0.88
SVM: Polynomial	0.99	0.99	0.99	0.88	x 0.88	✓0.90
SVM: Norm Polynom	1.00	1.00	1.00	0.89	x 0.89	✓0.91
SVM: Pearson VII	1.00	1.00	1.00	0.89	x 0.89	✓0.91
SVM: RBF	1.00	1.00	1.00	0.82	0.82	✓0.85

✓, x, statistically significant improvement or degradation (for a statistical significance of 0.05).

Table 4: Recall evaluation of machine-learning classifiers. The VSM columns correspond to the non-modified dataset, the WSD columns correspond to the dataset pre-processed with WSD with the default option set off, and the WSD_def columns correspond to the pre-processed dataset with WSD and the default option set on.

Recall						
Classifier	Ling Spam			TREC		
	VSM	WSD	WSD_def	VSM	WSD	WSD_def
DT: J48	0.85	0.83	0.83	0.98	0.97	x 0.98
DT: RF N=10	0.92	0.91	0.91	0.98	0.98	0.98
DT: RF N=50	0.93	0.91	0.91	0.98	0.98	0.98
DT: RF N=100	0.93	0.92	0.91	0.98	0.98	0.98
DT: RF N=150	0.93	0.92	0.91	0.98	0.98	0.98
DT: RF N=200	0.93	0.92	0.91	0.98	0.98	0.98
Naïve Bayes	0.99	0.98	0.98	0.35	x 0.34	x 0.34
BN: K2	0.90	x 0.84	x 0.77	0.39	x 0.38	x 0.37
BN: Hill Climber	0.90	x 0.84	x 0.77	0.39	x 0.38	x 0.37
BN: TAN	0.98	0.99	0.98	0.98	0.98	0.97
Knn K=1	0.41	0.41	✓0.45	0.97	0.97	0.97
Knn K=2	0.44	0.46	✓0.58	0.98	0.98	0.98
Knn K=3	0.31	0.31	✓0.42	0.97	0.97	0.97
Knn K=4	0.28	✓0.39	✓0.47	0.97	0.97	0.97
Knn K=5	0.24	0.30	✓0.36	0.97	0.97	0.97
SVM: Lineal	0.95	0.95	✓0.97	0.98	0.98	0.98
SVM: Sigmoid	0.91	0.90	✓0.93	0.99	0.99	0.99
SVM: Polynomial	0.97	0.96	0.98	0.98	✓0.99	0.98
SVM: Norm Polynom	0.85	0.86	✓0.90	0.99	0.99	x 0.98
SVM: Pearson VII	0.20	x 0.18	0.18	0.99	0.99	x 0.99
SVM: RBF	0.92	0.92	✓0.96	0.99	0.99	x 0.99

✓, x, statistically significant improvement or degradation (for a statistical significance of 0.05).

Table 5: Area under de ROC curve (AUC) evaluation of the machine-learning classifiers. The VSM column correspond to the non-modified dataset, the WSD column correspond to the dataset pre-processed with WSD with the default option set off, and the WSD_def column correspond to the pre-processed dataset with WSD and the default option set on.

Area under de ROC curve (AUC)						
Classifier	Ling Spam			TREC		
	VSM	WSD	WSD_def	VSM	WSD	WSD_def
DT: J48	0.92	0.91	0.90	0.90	✓0.93	0.92
DT: RF N=10	1.00	1.00	1.00	0.96	x 0.96	✓0.96
DT: RF N=50	1.00	1.00	1.00	0.96	x 0.96	✓0.97
DT: RF N=100	1.00	1.00	1.00	0.96	x 0.96	✓0.97
DT: RF N=150	1.00	1.00	1.00	0.96	x 0.96	✓0.97
DT: RF N=200	1.00	1.00	1.00	0.96	x 0.96	✓0.97
Naïve Bayes	0.94	0.94	✓0.95	0.91	0.92	0.92
BN: K2	1.00	1.00	x 0.99	0.95	x 0.95	✓0.96
BN: Hill Climber	1.00	1.00	x 0.99	0.95	x 0.95	✓0.96
BN: TAN	1.00	1.00	1.00	0.95	x 0.94	✓0.95
Knn K=1	0.70	0.71	✓0.72	0.95	0.95	0.96
Knn K=2	0.64	✓0.72	✓0.77	0.95	0.95	✓0.96
Knn K=3	0.75	0.74	✓0.80	0.95	0.95	✓0.96
Knn K=4	0.78	0.77	✓0.84	0.95	0.95	✓0.96
Knn K=5	0.81	0.82	✓0.86	0.95	0.95	✓0.96
SVM: Lineal	0.98	0.98	✓0.99	0.87	x 0.86	✓0.89
SVM: Sigmoid	0.95	0.95	✓0.97	0.84	x 0.84	✓0.86
SVM: Polynomial	0.98	0.98	✓0.99	0.86	x 0.86	✓0.89
SVM: Norm Polynom	0.93	0.93	✓0.95	0.88	x 0.87	✓0.90
SVM: Pearson VII	0.60	x 0.59	0.59	0.88	x 0.87	✓0.90
SVM: RBF	0.96	0.96	✓0.98	0.78	0.78	✓0.82

✓, x, statistically significant improvement or degradation (for a statistical significance of 0.05).