Universidad de Deusto

Semantic-aware unsolicited mail filtering with reduction of labelling efforts

Dissertation submitted by
Carlos Laorden Gómez
for the degree of
Doctor of Philosophy

Supervised by
Dr. Pablo García Bringas
and
Dr. Gonzalo Álvarez Marañón

Bilbao, Enero de 2012

Universidad de Deusto

Semantic-aware unsolicited mail filtering with reduction of labelling efforts

Dissertation submitted by
Carlos Laorden Gómez
for the degree of
Doctor of Philosophy

Supervised by
Dr. Pablo García Bringas
and
Dr. Gonzalo Álvarez Marañón

Author | Co-advisor | Co-advisor

Bilbao, Enero de 2012

A mis padres.
Esfuerzo, dedicación y cariño.
Edu y Cris, soy lo que soy gracias a vosotros.

Abstract

Electronic mail is a powerful communication channel. Nevertheless, as happens with all useful media, it is prone to misuse. 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. 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) and for spreading malicious software (e.g., computer viruses, Trojan horses, spyware and Internet worms).

In order to find a solution to this problem, the research community has made a great effort, with good results in solving text categorization problems. Thus, spam filtering systems have adapted different machine-learning techniques, providing a satisfactory evaluation of the e-mails' content.

These techniques model the e-mails using the Vector Space Model (VSM), an algebraic approach for Information Filtering, Information Retrieval, indexing and "ranking". This model represents natural language documents in a mathematical way by vectors in a multidimensional space with good results. Still, the VSM assumes that all terms are independent, what, at least from the linguistic point of view, is not entirely correct. Therefore, it can not support the linguistic phenomena that can be found in natural languages. In a similar vein, the VSM is also affected by other characteristics of the text such as word sense ambiguity. Indeed, today's attacks against Bayesian spam filters attempt to keep the content of spam e-mail visible to humans, but obscured to filters. This could lead to misclassified legitimate e-mails and spammers evading filtering.

Furthermore, junk e-mails evolve at an incredible pace to adapt to the most effective classifiers and surpass the filters, hence limiting in time the validity of the spam collections and classifiers. That is why obtaining properly labelled datasets for the training phase required by the machine-learning methods employed by anti-spam filters becomes a very complex task.

In light of this background, we propose to study the application of new techniques capable of overcoming the semantic limitations of

current spam filtering systems. To this end, we propose (i) the application of a new representation based on VSM, the enhanced Topic-based Vector Space Model (eTVSM), and (ii) a disambiguation preprocess that enhances the filtering capabilities of anti-spam systems.

Moreover, it is also proposed to reduce the labelling efforts necessary for the proper performance of machine-learning methods, by applying (i) collective classification, which looks for connections between the different documents to optimise the classification, and (ii) anomaly detection, which generates a model based solely on one class and detects deviations from this model.

Resumen

El correo electrónico (e-mail) es un potente canal de comunicación. No obstante, como ocurre con toda tecnología, es vulnerable al uso malintencionado. En la última década, el "spam", o correo basura, se ha convertido en un problema importante para los usuarios de e-mail. Pero el spam no solo es algo desagradable para los usuarios de correo electrónico, sino un problema de grandes dimensiones dentro del campo de la seguridad digital, ampliamente utilizado como medio para el "phishing" (ataques que buscan adquirir información sensible del usuario) y la propagación de software malicioso (virus, troyanos, spyware o gusanos).

Con el fin de encontrar una solución a este problema, la comunidad investigadora ha realizado un gran esfuerzo, consiguiendo buenos resultados en la resolución de problemas de categorización de textos. Así, se han adaptado a los sistemas de filtrado de spam diferentes técnicas de aprendizaje automático, ofreciendo una satisfactoria evaluación del contenido de los mensajes.

Estas técnicas modelan los e-mails usando el Modelo de Espacio Vectorial (VSM), una aproximación algebraica para el Filtrado de Información, la Recuperación de Información, el indexado y el "ranking". Dicho modelo representa los documentos del lenguaje natural de modo matemático mediante vectores en un espacio multidimensional, obteniendo buenos resultados. Aun así, el VSM asume que todos los términos son independientes, lo que al menos desde el punto de vista lingüístico no es completamente correcto. Por lo tanto, no es capaz de tener en cuenta los fenómenos lingüísticos que se pueden encontrar en los lenguajes naturales. Del mismo modo, el VSM se ve también afectado por otras características del texto como la ambigüedad de las palabras. De hecho, los ataques de hoy en día contra los actuales filtros Bayesianos tratan de mantener el contenido del correo no solicitado legible para las personas, pero inidentificable para los filtros. Esto puede originar tanto e-mails legitimos mal clasificados, como spammers saltándose la protección de los filtros.

Por otro lado, los correos electrónicos no deseados evolucionan a una velocidad realmente alta con el fin de adaptarse a los clasificadores más efectivos y evitar los filtros de correo basura. De este modo

quedan limitadas en el tiempo la validez de las colecciones y clasificadores de spam. Es por ello que la obtención de conjuntos de datos correctamente etiquetados para la fase de entrenamiento que requieren los métodos de aprendizaje automático se convierte en una tarea realmente compleja.

A la luz de este entorno, planteamos estudiar la aplicación de nuevas técnicas capaces de superar las limitaciones semánticas de los actuales sistemas de filtrado de correo electrónico no deseado. Para ello, se propone (i) la aplicación de una nueva representación basada en el VSM, el VSM basado en temas mejorado, o eTVSM (enhanced Topic-based Vector Space Model), y (ii) un sistema de desambiguación que mejora la capacidad de detección de los sistemas anti-spam.

Además, se propone paliar el problema del volumen de etiquetado necesario para el buen rendimiento de los métodos de aprendizaje automático mediante (i) la clasificación colectiva, que busca las conexiones existentes entre los diferentes documentos para optimizar la clasificación, y (ii) la detección de anomalías, que genera un modelo basándose exclusivamente en una de las clases y es capaz de identificar las desviaciones sobre dicho modelo.

Agradecimientos

Este trabajo debe mucho a mucha gente que me ha ayudado y animado a lo largo de estos intensos pero reconfortantes años.

Primero de todo me gustaría dar mi más sincero agradecimiento a mis dos directores de tesis, Pablo García Bringas y Gonzalo Álvarez Marañón. Sus valiosos comentarios, recomendaciones y directrices han facilitado en gran medida el transcurso de este trabajo. No puedo tampoco olvidar la confianza depositada en mi por Pablo García Bringas cuando en mi último año de carrera me ofreció la posibilidad de formar parte del *S^3Lab (Laboratory for Smartness, Semantics and Security)*. Esta oportunidad me ha permitido formar parte de un excelente (y me quedo corto) grupo de personas, gracias al cual me encuentro actualmente redactando estas líneas. *A Pablo y Gonzalo, por todo, gracias*.

El S^3Lab cuenta como digo con un fantástico grupo humano pero especial mención, por el impacto que han tenido tanto en mi trabajo doctoral como en mi desarrollo profesional, merecen: Igor Santos Grueiro, Borja Sanz Urquijo y Javier Nieves Acedo. Excelentes ideas, excelente pensamiento crítico, excelente actitud, excelente compromiso. Su ambición y calidad les llevará lejos y, con un poco de suerte, yo estaré ahí para contarlo. *Más que compañeros de trabajo, gracias Igor, gracias Borja, gracias Javi. Gracias amigos*.

El crecimiento del S^3Lab ha traído con los años nuevas incorporaciones. Patxi Galán García y Xabier Ugarte Pedrero forman parte de esa sangre nueva que ha aportado frescura y ambición al grupo. Mi más profundo agradecimiento por el impacto que también han tenido en el desarrollo de este trabajo va dirigido a ellos. *Gracias Patxi y Xabi*.

Me gustaría también mencionar la labor realizada por Mikel Salazar González en cada uno de los diseños gráficos de este y otros trabajos. Aunque tu pensamiento sea particular, sigues siendo un artista, lo que unido a tu continua búsqueda de la perfección, se ve reflejado en cada una de tus creaciones. *Gracias Mikel*.

He de reconocer la oportunidad de colaborar con, OPTENET, una gran empresa de seguridad y filtrado de contenido y, más concretamente, con Jose María Gómez Hidalgo, al cual debo agradecer todas sus valoraciones y consejos acerca de este y otros trabajos.

Quisiera agradecer además tanto a la Universidad de Deusto como a DeustoTech la posibilidad que me han ofrecido de realizar una estancia en el extranjero. Esta oportunidad ha facilitado la consecución de esta disertación doctoral, además de haberme aportado un sinfín de positivas experiencias y conocimiento de otras formas de trabajo.

La Università degli studi di Bergamo tuvo la amabilidad de acogerme en su sede de Dalmine. Pero si a alguien debo agradecer mi placentera y fructuosa estancia en Italia es a Giuseppe Psaila y a sus hijas, Laura y Elena. *La loro gentilezza e ospitalità mi hanno fatto sentire come a casa e, difatto, trovare una nuova famiglia in Italia*. *Grazie mille Giuseppe*.

Y si en Italia me encuentro escribiendo estas líneas es gracias a María Ramírez de la Piscina Urraca. Tú me animaste a venir para acabar este trabajo y, aunque sé que ha sido duro, ha merecido la pena. Por acompañarme durante todo el proceso y apoyarme incondicionalmente, gracias. Por ayudarme a mejorar continuamente tanto en lo personal como en lo profesional, gracias. *Por hacerme más feliz cada día, gracias María*.

Por último, me gustaría dedicar unas líneas a toda mi familia, tanto a la que está como a la que no está. Por poner su granito de arena, por inspirarme o, simplemente, por creer en mí. Mi ambición y mi esfuerzo no son características heredadas, son enseñadas. *A mi familia, mi modelo a seguir, muchas gracias*.

Para acabar, como suele decirse: no están todos los que son, pero son todos los que están. *A todos los olvidados, si mi errática memoria no lo soluciona antes de mandar este documento a imprenta, gracias.*

Contents

Figure index

Table index

Monty Python's spam skit

"Spam" is a popular Monty Python sketch, first televised in 1970. In the sketch, two customers are in a greasy spoon cafeteria trying to order a breakfast from a menu that includes the processed meat product in almost every dish. The term spam (in electronic communication, and general slang) is derived from this sketch. The sketch was written by Terry Jones and Michael Palin.

—

Scene: A cafe. One table is occupied by a group of Vikings wearing horned helmets. Whenever the word spam is repeated, they begin singing and/or chanting. A man and his wife enter. The man is played by Eric Idle, the wife is played by Graham Chapman, and the waitress is played by Terry Jones.

Man: You sit here, dear.
Wife: All right.
Man: Morning!
Waitress: Morning!
Man: Well, what've you got?
Waitress: Well, there's egg and bacon; egg sausage and bacon; egg and spam; egg bacon and spam; egg bacon sausage and spam; spam bacon sausage and spam; spam egg spam spam bacon and spam; spam sausage spam spam bacon spam tomato and spam;
Vikings: Spam spam spam spam...
Waitress: ...spam spam spam egg and spam; spam spam spam spam spam spam baked beans spam spam spam...
Vikings: Spam! Lovely spam! Lovely spam!

Waitress: ...or Lobster Thermidor a Crevette with a mornay sauce served in a Provencale manner with shallots and aubergines garnished with truffle pate, brandy and with a fried egg on top and spam.
Wife: Have you got anything without spam?
Waitress: Well, there's spam egg sausage and spam, that's not got much spam in it.
Wife: I don't want ANY spam!
Man: Why can't she have egg bacon spam and sausage?
Wife: THAT'S got spam in it!
Man: Hasn't got as much spam in it as spam egg sausage and spam, has it?
Vikings: Spam spam spam spam... (Crescendo through next few lines...)
Wife: Could you do the egg bacon spam and sausage without the spam then?
Waitress: Urgghh!
Wife: What do you mean 'Urgghh'? I don't like spam!
Vikings: Lovely spam! Wonderful spam!
Waitress: Shut up!
Vikings: Lovely spam! Wonderful spam!
Waitress: Shut up! (Vikings stop) Bloody Vikings! You can't have egg bacon spam and sausage without the spam.
Wife: I don't like spam!
Man: Sshh, dear, don't cause a fuss. I'll have your spam. I love it. I'm having spam spam spam spam spam spam spam beaked beans spam spam spam and spam!
Vikings: Spam spam spam spam. Lovely spam! Wonderful spam!
Waitress: Shut up!! Baked beans are off.
Man: Well could I have her spam instead of the baked beans then?
Waitress: You mean spam spam spam spam spam spam... (but it is too late and the Vikings drown her words)
Vikings: (Singing elaborately...) Spam spam spam spam. Lovely spam! Wonderful spam! Spam spa-a-a-a-a-am spam spa-a-a-a-a-am spam. Lovely spam! Lovely spam! Lovely spam! Lovely spam! Lovely spam! Spam spam spam spam!

"Spam is all about economics. When sending junk mail costs a dollar in paper, list rental, and postage, a marketer needs a reasonable conversion rate to make the campaign worthwhile. When sending junk mail is almost free, a one in ten million conversion rate is acceptable."

Bruce Schneier
(1963 –)

1

Introduction

Most people do not know that despite the term *spam* has today come to mean network abuse, massive junk postings and unsolicited commercial mail, its true origin lies in a canned food. Much to the displeasure of Hormel Foods[1], its "Shoulder Pork and hAM" or "SPiced hAM" luncheon meat has become the nowadays most disgusting and non-desired kind of abuse. But, how did the term get this meaning? The spam skit by Monty Python's Flying Circus seems to be responsible for it. In the skit, they parody a restaurant that serves all its food with lots of spam (i.e., spiced ham), and the waitress repeats the word several times describing how much spam is in the items. Thus, the term meaning at least: something that keeps repeating and repeating to great annoyance.

Unfortunately, spam has become not only an annoying issue that floods everybody's inbox, but a serious threat to computer security. In fact, junk e-mail is a perfect channel for the spreading of further threats (Bratko et al., 2006) such as malware (i.e., any computer software designed to harm computers or networks) or phishing (i.e., posing as a trustful sender to steal the victims' credentials and/or sensitive information). Furthermore, junk e-mails evolve at an incredible pace to adapt to the most effective classifiers and surpass the filters, hence limiting in time the validity of the spam collections and classifiers.

Moreover, different studies show that the effect of spam in worldwide economy is notorious and prejudicial. According to Radicati (2010), a typical 1,000-user organisation can spend upwards of US$ 3.0 million a year to fight and

[1] http://www.spam.com

manage spam. Leung and Liang (2009) presented an analysis of the impact of phishing on the market value of global firms, which showed that phishing alerts pose a significantly negative return on stock. In a similar vein, Mostafa Raad et al. (2010) offered another study to assess the influence and impact of spam in several companies whose e-mail advertisement was considered as spam. Both examples clearly show the necessity to detect undesired messages, and, more important, the need to restore the confidence of users in their e-mail filtering systems. Therefore, spam filtering is a critical topic within Information Security that has become a topic of concern and research.

The remainder of this chapter is organised as follows. Section 1.1 briefly describes the history of spam. Section 1.2 addresses the impact of electronic undesired mail on e-commerce. Section 1.3 outlines the main challenges that remain unsolved in spam filtering. Section 1.4 formulates the main hypothesis of this dissertation and section 1.5 details the goals of this dissertation. Section 1.6 details the research methodology conducted to achieve the goals. Finally, Section 1.7 outlines the structure of the present dissertation.

1.1 Spam, a real problem in computer security

Despite e-mail spam started as an annoying issue used by aggressive marketers, it has evolved into a more serious threat for Information Security.

This section reviews the history of spam, from the first non harmful junk e-mails to the current money-making oriented threats.

1.1.1 The first junk e-mail

Einar Stefferud, an Internet pioneer, reports that DEC announced a new DEC-20 machine in 1978 by sending an invite to all ARPANET (Advanced Research Projects Agency Network) addresses on the United States west coast. ARPANET was the world's first operational packet switching network, and the predecessor of the contemporary global Internet. The e-mail, shown in Figure 1.1, was sent using the ARPANET directory to invite people to receptions in California. They were chastised for breaking the ARPANET appropriate use policy and a notice was sent out reminding others of the rule.

Einar Stefferud, who was one of the recipients of the spam, provides this note of explanation:

> *"It was sent from SNDMSG which had limited space for To and CC and Subject fields. The poor soul that typed in the announcement, also (in those days) had to type in all the addresses, and this person was not trained in the use of SNDMSG.*

> *So, she/he started typing addresses into the Subject which overflowed into the TO header, which overflowed into the CC header, and then into the Body, and then the actual message was finally typed in. So, lots of intended recipients did not receive it, including me."*

As can be appreciated, the primitive methods deployed in the beginnings of commercial junk mail sending needed an evolution to achieve the automation of the process in order to increase the number of recipients.

```
Mail-from: DEC-MARLBORO rcvd at 3-May-78 0955-PDT
Date: 1 May 1978 1233-EDT
From: THUERK at DEC-MARLBORO
Subject: {...}
To: {...}

DIGITAL WILL BE GIVING A PRODUCT PRESENTATION OF THE NEWEST MEMBERS OF
THE DECSYSTEM-20 FAMILY; THE DECSYSTEM-2020, 2020T, 2060, AND 2060T. THE
DECSYSTEM-20 FAMILY OF COMPUTERS HAS EVOLVED FROM THE TENEX OPERATING SYS-
TEM AND THE DECSYSTEM-10 <PDP-10> COMPUTER ARCHITECTURE. BOTH THE DECSYSTEM-
2060T AND 2020T OFFER FULL ARPANET SUPPORT UNDER THE TOPS-20 OPERATING SYS-
TEM. THE DECSYSTEM-2060 IS AN UPWARD EXTENSION OF THE CURRENT DECSYSTEM 2040
AND 2050 FAMILY. THE DECSYSTEM-2020 IS A NEW LOW END MEMBER OF THE DECSYSTEM-
20 FAMILY AND FULLY SOFTWARE COMPATIBLE WITH ALL OF THE OTHER DECSYSTEM-20
MODELS.

WE INVITE YOU TO COME SEE THE 2020 AND HEAR ABOUT THE DECSYSTEM-20 FAMILY AT
THE TWO PRODUCT PRESENTATIONS WE WILL BE GIVING IN CALIFORNIA THIS MONTH. THE
LOCATIONS WILL BE:

TUESDAY, MAY 9, 1978 - 2 PM HYATT HOUSE (NEAR THE L.A. AIRPORT) LOS ANGELES,
CA
THURSDAY, MAY 11, 1978 - 2 PM DUNFEY'S ROYAL COACH SAN MATEO, CA (4 MILES
SOUTH OF S.F. AIRPORT AT BAYSHORE, RT 101 AND RT 92)

A 2020 WILL BE THERE FOR YOU TO VIEW. ALSO TERMINALS ON-LINE TO OTHER
DECSYSTEM-20 SYSTEMS THROUGH THE ARPANET. IF YOU ARE UNABLE TO ATTEND, PLEASE
FEEL FREE TO CONTACT THE NEAREST DEC OFFICE FOR MORE INFORMATION ABOUT THE
EXCITING DECSYSTEM-20 FAMILY.
```

Figure 1.1: First spam message. Note that recipients have been omitted.

As the years went by, some people commenced to see a business in unsolicited e-mail sending, and started developing new tools to facilitate the distribution process.

1.1.2 The origin of spam

In April 1994, the term was not born, but it did jump a great deal in popularity when two lawyers from Phoenix, named Canter and Siegel, posted a message advertising their fairly useless services in an upcoming U.S. "green card" lottery. This was not the first such abusive posting, nor the first mass posting to be called a spam, but it was the first deliberate mass posting to commonly get that name. They had posted their message a few times before, but on April 12, they hired an mercenary programmer to write a simple script to post their ad to every single newsgroup (message board) on USENET, the world's largest on-line conferencing system.

Quickly people identified it as "spam" after the Monty Python's sketch, and the word caught on. Future multiple postings soon got the appellation. Some people also applied it to individual unwanted ads that were not posted again and again, though generally it was associated with the massive flood of the same message.

Later, some people figured they could take mass e-mailing software (which had been around for decades to handle mailing lists) and use it to send junk e-mail to large audiences who had not asked for it. The term quickly came to be used to describe these unwanted junk e-mails, and indeed that is the most common use of the term spam today.

1.1.3 **Spam evolution**

The first mass e-mails that reached the nomenclature of spam made use of rudimentary methods to maximise the spread of their sending. Over the years, with the development of new technologies, these processes have evolved up to the point where, nowadays, spam has become a real business.

The weakness of the protocol that e-mail works over, along with the zero cost involved in using this communication channel, has become easy prey for con artists and organised groups to produce junk e-mail for fraudulent purposes. Recent statistics offer a sign of the concern that the problem of spam should generate, indicating that from all electronic mail circulating on the Internet, more than 85% corresponds to spam[2]. Bank fraud attempts, chain letters, promotion of products of dubious quality (e.g., pharmaceutical, replicas) or malware distribution, is a small sample of the long list of applications offered by the massive unsolicited e-mail sending.

Figure 1.2 (based on *Keith Lynch's timeline*[3] and extended with the chronicles offered by Kaspersky's Securelist in *The Planet of the Spammers*[4]) shows the evolution that e-mail spam has suffered during its first 30 years of life.

Despite the first junk e-mail was sent back in the year 1978, the earliest known e-mail chain letter dates from 1982. Another e-mail chain letter was detected in 1985, and, in 1988, the *MAKE MONEY FAST* letter was written, a pyramid scheme created by a person who used the name Dave Rhodes, although it did not reach the news until 1993. In 1994, the "green card lawyers" newsgroup junk e-mail appeared, and the term spam commenced to be used to refer to unsolicited e-mail.

The first tools to send massive unsolicited e-mails (e.g., floodgate, spamvertise and spamware) appeared in 1995. In the same year, the black market of valid e-mail addresses was offering a list of 2 million units for sale. The next

[2] http://www.spam-o-meter.com/ (Oct. 17, 2011)
[3] http://keithlynch.net/spamline.html
[4] http://www.securelist.com/en/analysis/204792192/The_Planet_of_the_Spammers

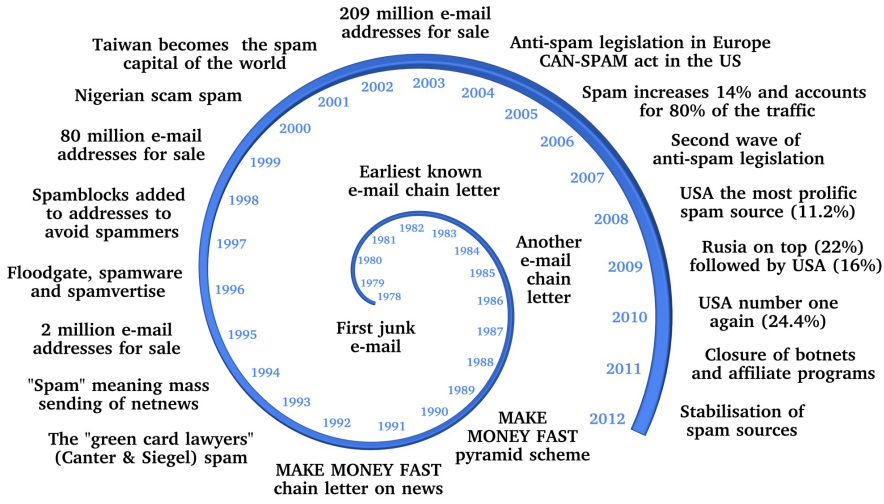


Figure 1.2: Timeline of spam.

year, 1996, spamblocks were added to addresses to avoid spammers. By the end of 1997, Paul Vixie created a *Realtime Blackhole List* of spam sites, and the list of e-mail addresses offered for sale ascended to 80 million.

It is significant that by 1995 spammers were trafficking with e-mail addresses of real users, and in the small lapse of time of 6 years, the volume of addresses offered increased in more than a 10,000% (209 million e-mail addresses offered for sale in 2001). After that, in 2003, many European countries adopted new anti-spam legislation and, at the end of the year, the United States Congress approved the Controlling the Assault of Non-Solicited Pornography And Marketing Act, usually referred to as the *CAN-SPAM Act*. Despite this changes on legislation, the amount of spam in 2004 increased by 14%, accounting for 80% of all electronic mail traffic. In fact, the US became in 2004 the most prolific source of spam with nearly 50% of the whole traffic.

In 2006-2007, a second wave of anti-spam laws was adopted by China, Pakistan, Singapore, New Zealand and Russia. Despite the new legislation, Russia and China joined the US in the top three sources of global spam. In 2007, the US topped the ranking with 11.2% of all distributed spam, closely followed by Russia with a 10.8%. In 2008, Russia became number one with a 22% and the US was responsible for 16%. In the year 2009, Russia's share of spam gradually decreased, dropping to 8.5%, while during the second half of the year the amount of spam coming from the US increased to a 24.4%. Behind them, Brazil (7.6%), India (5.9%) and South Korea (4.8%).

In the year 2010, the situation changed with several botnet command centres being shut down and the initiation of criminal proceedings against spammers and

botnet owners. This anti-botnet campaign forced cybercriminals to explore more convenient territories from which to send their spam. The amount of spam sent from the US decreased considerably and relocated in Russia, Ukraine and other eastern European countries.

Unlike the previous year, in 2011 the amount of spam sent from each source stabilised, with the only exceptions of the gradual rise of Asia and Latin America as main sources and the decrease of spam originated from western and eastern Europe. Figure 1.3 offers a current graphic of the share of spam worldwide.

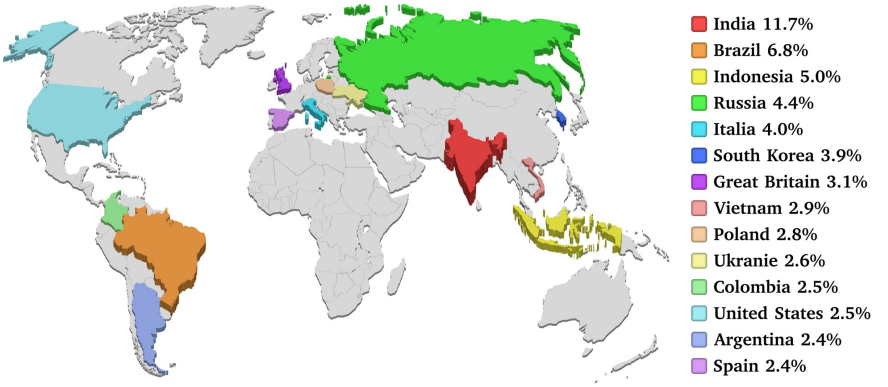


Figure 1.3: Largest producers of spam in the world in the first half of 2011.

This distribution signals there are no unsolicited e-mail free territories. Developing countries offer botnet owners protection due to the lack of anti-spam laws and developed countries are of interest because of their advanced infrastructures.

To conclude, nowadays, the growing use of the Internet has done nothing but provide the opportunity for the use of electronic mail as a channel for fraud. But technology is not the only responsible of the expansion of spam. While most of the world suffers from this problem, some countries are not aware of the importance of fighting against it. These countries do little to stop the spammers operating within their borders, becoming safe heavens from which to operate.

1.2 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. Incorrectly identifying spam can have an impact on e-commerce, since false positives result in the recipient not receiving legitimate e-mails (e.g., personal e-mails or 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 different cyber-attacks. This report includes the analysis of a British firm, called Mi2g, which published investigation from the collection of data from 7,000 hacker groups worldwide. This study provides 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, unsolicited e-mail, malicious software 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"[5]. Figure 1.4 shows the cost estimates for those digital attacks, which include hacking, malware and spam from 1996 to 2003.

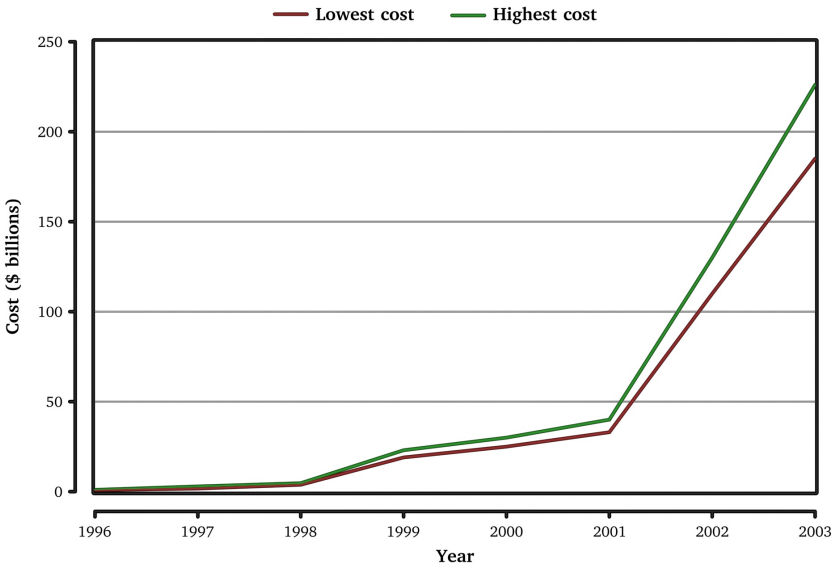


Figure 1.4: Economic damage estimates for all forms of digital attacks worldwide, based on business interruption, denial of service, data theft or deletion, loss of sensitive intelligence or intellectual property, loss of reputation, and share price declines. Source: Mi2g, Frequently Asked Questions: SIPS and EVEDA, v1.00.

[5]Mi2g defines an overt digital attack as one in which a hacker group gains unauthorised 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.

Trying to break down the numbers, in another study, Hansell (2003) showed that in 2003 the volume of unsolicited e-mails, 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 85% of received electronic mails are junk messages[6]. In this way, although the numbers correspond 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 completely 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 the costs of cybercrime go beyond the tangible issues (e.g., stolen assets, business losses or damages on company reputation), having a significant negative effect on shareholder value. The explanation to that fact resides on the worries of users about the 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 possible 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 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).

1.3 Challenges in the field of undesired electronic mail filtering

Spam is currently an unsolved problem for various technical and legal reasons. However, among the most effective measures against it, we would highlight the Bayesian filters (Lewis, 1998; Androutsopoulos et al., 2000a,b,c; Schneider, 2003; Seewald, 2007). The effectiveness of this family of filters, which in general should be called "learning-based" filters, has forced spammers to implement specific and elaborate tactics to avoid them, which proves its effectiveness. The evolution and adaptation of these spamming techniques present several problems that so far have no solution and represent the major unresolved challenges in the spam filtering field. This work will focus on the following:

- **Support for linguistic phenomena:** The approaches based on machine learning represent e-mails 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

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

language documents in a mathematical way by vectors in a multidimensional space. The VSM assumes that each word is independent, what, at least from a linguistic point of view, is not completely true. Therefore it can not support the existing linguistic phenomena in natural languages (Becker and Kuropka, 2003). In this way, and understanding that e-mails are comprised of terms belonging to natural languages, it is necessary to explore the effect of enriching filtering systems with methods capable of accommodating synonyms, homonyms and other linguistic phenomena (Awad et al., 2008).

- **Word sense disambiguation:** Like any IR system, the VSM is affected by the characteristics of the text, being one of these characteristics the ambiguity of the meanings of certain words (Sanderson, 1994). The use of ambiguous words may confuse the model, allowing spammers to avoid anti-spam filters or to cause false positives. It is therefore considered appropriate to apply the disambiguation of terms to spam filtering in order to recover the detection capabilities of the content based methods.

- **Optimise the ratio of false negatives and false positives:** The two most obvious ways to drop a classifier's usefulness is to force a substantial number of errors, which can be by default, not detecting a significant number of spam messages (i.e., incurring in many false negatives) or by excess, classifying many legitimate messages as spam (i.e., to incur in many false positives). Moreover, the costs of these errors are asymmetric, that is, the cost of a default error is different from the cost of an error by excess for several reasons. On the one hand, the cost depends on the risk posed by the error going unnoticed for a user. On the other hand, the cost of correcting an error is dependent on the task and on how is organised the infrastructure to solve the problem (e.g., delete it or send it to quarantine, retrieve a legitimate e-mail classified as spam). Finally, a mistake can be more severe in some contexts than in others (e.g., children accessing pornographic content, important work e-mails lost among spam). That is why finding a balance in the ratio of false negatives and false positives is presented as a problem to be solved in the future.

- **Reduce the labelling efforts of anti-spam systems:** Machine-learning approaches are usually supervised, i.e., they need a training set of previously labelled samples. These techniques perform better as more training instances are available. It means that a significant amount of previous labelling work is needed to increase the accuracy of the models. However, it is quite difficult to obtain this amount of labelled data for a real-world problem such as the spam filtering issue. To generate these data, a time-consuming task of analysis is mandatory and, in the process, some spam messages can avoid filtering. In this way, it is recommended to reduce the labelling tasks required by anti-spam systems.

1.4 Fundamental hypothesis

The challenges outlined mark the boundaries that exist today in the area of spam filtering and this is where the scientific community focuses its efforts. Therefore, bearing in mind these limitations we establish the fundamental hypothesis of the present doctoral dissertation:

> «*It is possible to improve spam filtering systems with semantic-aware techniques, able to overcome linguistic phenomena, and reduce the time-consuming task of sample analysis, while optimising the detection capabilities.*»

This hypothesis aims to show the possibility of creating spam filtering systems sensitive to semantics and concerned about the reduction of the labelling efforts that analysts must perform for their spam filters. This model must be able to take into account the semantics implied in e-mails in order to overcome the attacks in which spammers take advantage of linguistic phenomena to overcome the statistical filter. In addition, future anti-spam systems must reduce the hard and time-consuming task of labelling instances that every machine-learning-based spam filtering system needs.

1.5 Goals

Taking as a reference the fundamental hypothesis we establish the following main goal:

Main Goal 1 *Develop and test spam filtering techniques capable of overcoming semantic attacks and reduce the labelling efforts required to maintain optimum detecting capabilities.*

This main goal is then divided in three specific goals:

Specific Goals 1 *Develop and evaluate a spam filtering model immune to linguistic phenomena.*

Specific Goals 2 *Build and evaluate a spam filtering model immune to word ambiguity.*

Specific Goals 3 *Develop and evaluate a model to reduce the labelling efforts that filtering systems require.*

The first two goals face the problem of semantics. Firstly, we seek to solve the current problems of statistical filters against the emergence of attacks that take

advantage of linguistic phenomena. With the second goal we want to get a model with the ability to evade the ambiguity present in natural languages. Finally, the third goal seeks to build a model capable of reducing the time-consuming labelling tasks that analysts carry out to provide spam filtering systems with examined samples.

To achieve these goals, we must first meet certain operational goals:

Operational Goals 1 *Design and implement a method for the representation of e-mails taking into account the linguistic phenomena implicit in every natural language.*

Operational Goals 2 *Design and implement a method for the disambiguation of the terms that e-mails are composed of.*

Operational Goals 3 *Design and implement a method to reduce the labelled instances needed to obtain optimum results from statistical filtering systems.*

Operational Goals 4 *Optimise the ratio of false negatives and false positives of the spam filtering system.*

1.6 Research methodology

Seeking to achieve the goals marked and defined to validate the hypothesis, we propose a research strategy that includes the following steps:

1. **Acquisition of knowledge** through constant review of publications that advance the state-of-the-art techniques for spam filtering and detection, and by attending conferences that bring together scientists involved in advancing the state-of-the-art.
2. **Design and development** of models and applications that allow endorsing the validity of the partial hypotheses and open new avenues of research.
3. **Experimentation and evaluation** of the results obtained with the aforementioned models.
4. **Redesign the models** created, following the feedback obtained in the experimentation.
5. **Presentation of preliminary results** to the scientific community to obtain feedback to help validate the followed path and see if the contributions manage to offer a real advance in the state-of-the-art.
6. **Validation and diffusion** of the acquired knowledge and learned lessons to the scientific community.

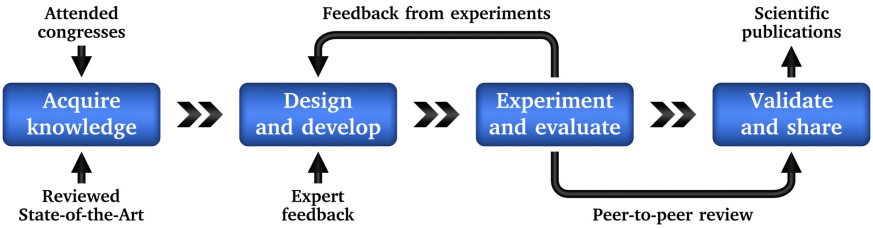


Figure 1.5: Schematic view of the research process.

Figure 1.5 shows graphically the schematic view of the research process, with the main activities that take place and the flow of input and output that feeds the process.

In addition, in order to implement and evaluate the prototypes and the partial approaches resulting from the investigation, we choose the Action Research methodology (Berg, 1989), a cyclical methodology that comprises the following steps (Susman, 1983):

1. **Diagnosis:** identify or define a problem.
2. **Planning:** consider the different alternatives of action.
3. **Action:** select the action to take.
4. **Evaluation:** study the consequences of the action.
5. **Definition:** identify and detail the overall findings.

These steps will be taken into account throughout the research process to provide scientific rigor and critical thinking to all the activity.

1.7 Thesis outline

This dissertation is organised in the following chapters:

Chapter 1. Introduction.

This chapter outlines the research presented on this dissertation. It introduces the current problems and challenges of spam filtering. It states the fundamental hypothesis this dissertation supports and presents the goals of the research. It also presents the research strategy followed throughout this dissertation.

Chapter 2. Literature review.

In this chapter, we detail the background of this research area, reviewing different proposed methods.

Chapter 3. Semantic-aware unsolicited e-mail filtering.

It introduces and details our methods to overcome semantic attacks against spam filtering systems. The chapter provides a detailed formalisation and description of two methods. First, it presents a synonym-aware solution for spam filtering and, second, an ambiguity-resilient approach to discriminate the ambiguous terms present in the e-mails. Furthermore, we evaluate these approaches and compare the results with other academical and commercial approaches.

Chapter 4. Reducing labelling efforts.

This chapter presents the proposed methods to reduce the time-consuming task of labelling, which analysts must carry out to provide their spam filtering systems with enough samples to feed the statistical algorithms. We evaluate this approaches and compare them with current solutions.

Chapter 5. Conclusions.

With this chapter we conclude our research, revisiting the hypothesis and goals presented, and providing the major contributions along with some discussion about this dissertation. It also includes open research lines and challenges for future reference, and ends with some final remarks.

"Nature has given us the seeds of knowledge, not knowledge itself."

Lucio Anneo Séneca
(2 B.C. – 65)

2

Literature review

An innumerable amount of approaches have tried to fight the problem of unsolicited electronic mail filtering. This chapter is devoted to introduce some of the most relevant techniques proposed along the years, beginning with some pioneer approaches that were penetrated as the appearance of new technologies aided criminals to improve their spamming systems, and concluding with the mostly used nowadays machine-learning-based approaches. We also present the main actors and some general concepts related to spam filtering and some extra literature review on other techniques that were used throughout this work.

The remainder of this chapter is organised as follows. Section 2.1 presents some common actors and basic concepts related to the area of spam filtering. Section 2.2 presents a classification of anti-spam methods and centres the focus of our work on content-based systems. Section 2.3 introduces the most relevant techniques within the area of spam filtering based on the analysis of the content of the messages. Section 2.4 provides other literature related to the area of spam filtering. Finally, Section 2.5 summarises the main aspects of this chapter.

2.1 Spam ecosystem

Junk electronic mail, unsolicited (commercial) e-mail, (unsolicited) bulk e-mail or, simply, unwanted e-mail, are some of the most commonly used ways to refer to the same thing: spam. But what is exactly spam?

The Spam Track at the Text Retrieval Conference (TREC) defines e-mail spam as "unsolicited, unwanted e-mail that was sent indiscriminately, directly or indirectly, by a sender having no current relationship with the recipient" (Cormack and Lynam, 2005). It can be appreciated that they are defining *e-mail* spam, not spam generally. Why? Because we can find or receive unsolicited communications through several channels besides e-mail, e.g., ordinary mail, telephone (Dantu and Kolan, 2005; Gómez Hidalgo et al., 2006; Cormack et al., 2007b), blog comments (Mishne et al., 2005; Cormack et al., 2007a), on-line social networks (Zinman and Donath, 2007; Luo et al., 2009; Sanz et al., 2010; Laorden et al., 2010) or websites (Webb et al., 2006; Castillo et al., 2008).

A few years later, Cormack (2007a) came up with a generalised and more inclusive definition for the term: "unwanted communication intended to be delivered to an indiscriminate target, directly or indirectly, notwithstanding measures to prevent its delivery". In a similar vein, Sanz et al. (2008) consider an e-mail as spam if it has the following features: (i) unsolicited, the receiver is not interested in receiving the information; (ii) unknown sender, the receiver does not know and has no link with the sender; (iii) massive, the email has been sent to a large amount of addresses.

Because this work focuses on electronic mail spam, we choose the first definition from Cormack and Lynam (2005), and future references to spam will correspond to:

Spam *"Unsolicited, unwanted e-mail that was sent indiscriminately, directly or indirectly, by a sender having no current relationship with the recipient (Cormack and Lynam, 2005)."*

If we define spam, we also have to define the non-spam. Going back to the origin of the term, we know it comes from the "Shoulder Pork and hAM" or "SPiced hAM" luncheon meat by Hormel Foods. Bearing that canned food in mind, the opposite meal, when talking about quality, would be ham. That is the reason why the term *ham* is used to define non-spam:

Ham *"Legitimate electronic mail, solicited or not, usually sent by a sender having current relationship with the recipient."*

These definitions require the adjudication of subjective terms like *unwanted* and *legitimate*, because some people may find interesting commercial campaigns while others may never want to receive some kind of communications from their own contacts.

The senders of unsolicited e-mails are commonly referred to as *spammers*. According to *Cambridge Dictionary*[1] a spammer is a "person or company that sends

[1] http://dictionary.cambridge.org

unwanted email, usually advertisements". *The Free Dictionary*[2] says a spammer is "someone who sends unwanted e-mail (often in bulk)". Being the motivation behind spam to have information delivered to the recipient that contains a payload such as advertising for a (likely worthless, illegal, or non-existent) product, bait for a fraud scheme, promotion of a cause, or computer malicious software designed to hijack the recipient's computer (Cormack, 2007a), we will define spammer as:

Spammer *"A person who massively sends unsolicited electronic mail to unknown recipients, manually or aided by automation tools, to deliver information containing a payload (e.g., advertising, fraud scheme, promotion of a cause or malicious software) to obtain a benefit from."*

The mentioned payload includes some annoying issues such as the unwanted advertising or the promotion of a (usually irrelevant for the recipient) cause, and other more serious threats such as fraud schemes or malicious software. The practice of trying to defraud users by those mentioned schemes is usually referred to as *phish* or *phishing*. According to *Dictionary*[3] phish is:

Phish *"To try to obtain financial or other confidential information from Internet users, typically by sending an e-mail that looks as if it is from a legitimate organisation, usually a financial institution, but contains a link to a fake Web site that replicates the real one."*

The Merriam-Webster defines phishing as:

Phishing *"A scam by which an e-mail user is duped into revealing personal or confidential information which the scammer can use illicitly."*

Conversely, according to McGraw and Morrisett (2000) malicious software, or malware, is:

Malware *"Any code added, changed, or removed from a software system in order to intentionally cause harm or subvert the intended function of the system (McGraw and Morrisett, 2000)."*

Electronic mail, being such a powerful communication tool, has become one of the main channels for the spreading of the aforementioned threats in the form of spam. It is clear why unsolicited e-mail has become a major issue in computer security. Therefore, the industry and the research community have focused their efforts on providing useful tools to detect and filter junk messages. These tools are commonly referred to as spam filters and defined as:

[2] http://www.thefreedictionary.com
[3] http://dictionary.reference.com

Spam filter *"An automated technique to identify spam for the purpose of preventing its delivery (Cormack, 2007a)."*

Being the main goal of the present dissertation to advance the state-of-the-art on spam filtering, our contributions will be directed toward the improvement of current techniques to identify unsolicited electronic mail.

2.2 Anti-spam methods

In recent years, multiple methods and approaches have been proposed and developed to help ease the problem of spam, both legal and technical solutions. To organise all these methods, Gansterer et al. (2005) introduced a categorisation of anti-spam methods.

Its basic distinction differs mainly between the methods that "act" *before* an e-mail has been sent and the methods that "act" *after* an e-mail has been sent. This virtually covers all existing approaches, from the intent of reducing the number of junk e-mails sent to techniques based on text analysis and classification methods applied to received e-mails.

In figure 2.1 we can see a reduced version of the classification of Gansterer et al. in which we emphasise the analysis of content as the main focus of our work.

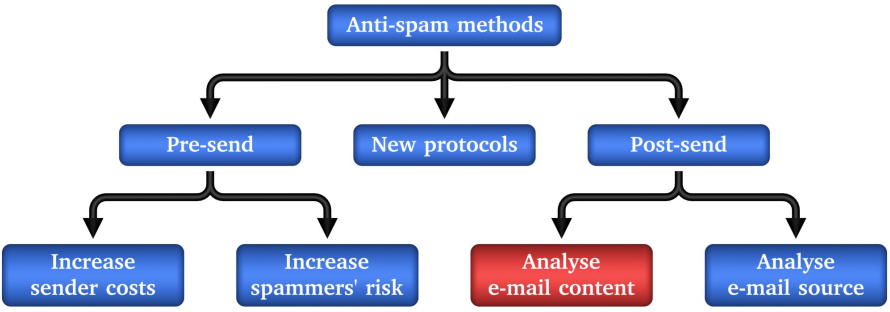


Figure 2.1: Reduced adaptation of the classification of Gansterer et ál. for anti-spam methods.

Among the methods for increasing sending costs we can find the so-called *cost functions*. These methods work by forcing the CPU of the sender to calculate a cost function, which delays sending, each time an e-mail is sent. This type of solution is insignificant to the common user, but highly detrimental to massive sending. Another method of increasing costs is to charge for each e-mail sent, a solution far more controversial and difficult to assimilate.

The methods that seek to increase the risk of spammers are essentially legal actions. Another proposed solution is called *Opt in - opt out*, where only the

ones who ask for "advertising" receive it (*opt in*) or only those who *opt out* stop receiving it. A different approach that also increases the risk for spammers the "spam traps" or "honeypotting". Spammers are known to abuse vulnerable systems like open mail relays and public open proxies. In order to discover spam activities, some administrators have created honeypot programs that simulate being such vulnerable systems, bursting hackers while protecting the real servers (Sanz et al., 2008). Moreover, spam traps can be used to collect instances of spam messages to keep a recent collection of spam to build and deploy updated filtering systems.

There are also proposals for new protocols that seek to complicate the massive sending of spam (e.g., AMTP (Weinman, 2003) or IM2000 (Bernstein, 2000)).

Among the approaches that analyse the content of the e-mail are: authentication through digital signatures; static methods based on keywords, pattern matching or rule-based; analysis of URLs using on-line search engines; *fingerprinting* methods such as checksums, CRCs (cyclic redundancy checks), hashing algorithms or message digests; and automatic classification methods, which model the e-mails and compare them with previously classified samples to assign the new class.

Finally, the methods based on the source of e-mail seek to determine if the sender is good or bad, if it is legitimate, using blacklists and whitelists, authentication tools, or checking if the message is sent by a human person with challenge-response techniques such as "quasi-Touring tests", Captcha challenges, or links.

Our goal is to address the abuse of undesired electronic mail with technical measures, by analysing the content of messages and then classifying them by means of Machine Learning (ML). ML is a branch of Artificial Intelligence (AI) concerned with the design and development of algorithms to evolve the behaviour of a system based on empirical data.

2.3 Content-based spam filtering

This section introduces different approaches to fight unsolicited e-mail that are based on the study of the content. Despite these methods are usually based on text categorisation, their task is not directed towards the understanding of the text of an e-mail, but to find significant features, such as word frequencies, to determine the nature of the e-mail. We start with the most simple techniques, heuristic approaches or url analysis, and finish with the more complex machine-learning-based methods.

2.3.1 Heuristic approaches

These approaches involve a search within the content of the e-mail of previously defined keywords and phrases that may identify an e-mail as spam (e.g., "Viagra",

"sex" or "free money"). Among them, we can find:

- **Keyword Based:** Search for exact matches of words or phrases. One problem that face this approaches is that a minimum alteration of a keyword results in the non identification of the e-mail while being readable by a human. For example, "V.i.a.g.r.a" would not be identified as a spam-related term but, still, is understandable for everyone.

- **Pattern Matching:** Using regular expressions (Friedl, 2006), this kind of methods cover simple variations of the keywords or phrases, resolving the problem of keyword alteration.

- **Rule Based:** Using more advanced regular expressions, several rules are checked. Besides, each rule receives a certain weight in order to give a value of certainty for an e-mail to be spam.

Cohen (1996) used a rule-learning algorithm to classify personal e-mails with a set of "keyword-spotting" rules. This algorithm, designed by himself and called RIPPER (Cohen, 1995), was able to provide comparable performance to "term frequency - inverse document frequency" ($tf - idf$) , a traditional weighting schema commonly used for text classification tasks. Some years later, several works presented other approaches outperforming rule-based algorithms with the application of machine learning (Provost, 1999) or (Androutsopoulos et al., 2000b).

2.3.2 URL analysis

In its simplest form URL analysis filters e-mails taking into account whitelists and/or blacklists, filtering e-mails that contain links to websites known to distribute different threats. However, the ease of URL obfuscation, via for example URL shortening services, reduces considerably the effectiveness of this approach.

In this way, more advanced approaches have been presented that analyse not only the URL but the linked website, by fetching and analysing the content from the site designated by the URL itself (Riemers, 2003; Kolesnikov et al., 2003). Thus, if an e-mail contains links to suspicious websites it should be considered as unsolicited.

2.3.3 Blacklisting and whitelisting

Blacklisting (Carpinter and Hunt, 2006) is a simple method 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. Given

the simplistic nature of these methods, it is unsurprising that they can be easily surpassed, making them unreliable as a standalone solution (Mishne et al., 2005).

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 and Sit, 2004; Ramachandran et al., 2006). Despite this method is highly effective at discarding substantial amounts of spam, it usually presents notoriously high rate of false positives, making it unusable as a standalone filtering system (Snyder, 2004).

2.3.4 Signature-based methods

Signature-based systems create a unique hash value (i.e., a message digest) for each known spam message (Kołcz et al., 2004). Signature filters compare the hash value of an incoming e-mail against all stored hash values of previously identified spam e-mails to classify the message. The main advantage of these types of methods is that they rarely produce false positives. Examples of signature-based spam filtering systems are:

- **The Distributed Checksum Clearinghouses or DCC**[4], is an anti-spam content filter, based on the idea that if e-mail recipients could compare the received messages they could recognise unsolicited e-mail. When a DCC client reports the checksums for an e-mail message to the server, receives the total number of recipients of each checksum. If the number of recipients surpasses a certain threshold, the e-mail can be considered as spam.

- **Vipul's Razor**[5], a distributed, collaborative, spam detection and filtering network. Through user contribution, it establishes a catalogue of spam consulted by e-mail clients to filter known unsolicited messages. Since the users themselves provide the information to the network, their input is validated through reputation assignments based on consensus on report and revoke assertions which in turn is used for computing confidence values associated with individual signatures.

- **Cloudmark**[6], a commercial implementation of a signature-based filter that integrates with the e-mail server. It combines fingerprinting and sender analysis techniques with corroborated feedback from trusted reporters, automated and anonymous traffic analysis and dedicated security analysts to detect and stop messaging abuse.

[4] http://www.rhyolite.com/dcc/
[5] http://razor.sourceforge.net/
[6] http://www.cloudmark.com/

The main disadvantage of signature-based systems is that they are unable to detect spam messages until they have been identified, properly registered and documented (Carpinter and Hunt, 2006).

2.3.5 **Collaborative filtering**

Collaborative filtering (Damiani et al., 2004; Gray and Haahr, 2004; Parvathaneni, 2011) is a distributed approach to filtering spam. This approach relies on knowledge sharing, instead of having each user to have his own filter, a whole community works together, sharing their judgements of what is spam and what is not with the other users. This paradigm includes a set of e-mail clients sharing their knowledge about recently received spam e-mails, providing a highly effective defence against a substantial fraction of spam attacks, also alleviating the burdens of frequent training stand-alone spam filters (Parvathaneni, 2011).

The weakness of this approach is that what is spam for somebody could be a legitimate content for another. These collaborative spam filters cannot be more accurate as a personal filter in the client side but it is an excellent option for filtering in the server side (Sanz et al., 2008).

2.3.6 **Machine-learning-based methods**

Machine learning is an active research area within 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 (Kotsiantis, 2007). Unsupervised learning algorithms try to infer how data are organised into different groups named *clusters*. Therefore, data do not need to be labelled (Kotsiantis and 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, supervised approaches have been applied successfully to anti-spam systems (Drucker et al., 1999; Androutsopoulos et al., 2000c,a,b; Carreras and Márquez, 2001; Schneider, 2003; Seewald, 2007). However, semi-supervised techniques (Pfahringer, 2006; Santos et al., 2011a,d,b,c; Ugarte-Pedrero et al., 2011) and unsupervised methods (Cormack, 2007a) have also proven their benefits in the spam problem and similar domains. The first part of our work employs supervised approaches while the second one studies some semi-supervised techniques never applied before to spam filtering. Although we would like to study the application of unsupervised learning in the future we will focus this work on supervised and semi-supervised

techniques. Therefore, we firstly present several successful supervised methods and secondly some semi-supervised approaches.

2.3.6.1 Supervised approaches

Supervised machine learning is the search for algorithms that reason from externally supplied instances to produce general hypotheses, which then make predictions about future instances (Kotsiantis, 2007). The goal of supervised learning is to build a concise model based on previously labelled data from which the class is known, and then use it to classify testing instances where class values are unknown. In this section we introduce some of the most used supervised algorithms that have been applied to the spam filtering problem.

Bayesian Networks

Bayesian inferencing is based on *Bayes' Theorem* (Bayes, 1763), a statistical reasoning method based upon a set of evidences, that is able to determine the probability that a hypothesis may be true. Therefore, Bayes' Theorem updates the previously learned probabilities when new observations appear. According to its classical formulation (shown in equation 2.1), given two events A and B, the conditional probability, $P(A|B)$, that A occurs if B occurs can be obtained if we know the probability that A occurs, $P(A)$, the probability that B occurs, $P(B)$, and the conditional probability of B given A, $P(B|A)$.

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{2.1}$$

More accurately, Bayesian Networks (Pearl, 1982) are defined as graphical probabilistic models for multivariate analysis. Specifically, they are directed acyclic graphs that have an associated probability distribution function (Castillo et al., 1996). Nodes within the directed graph represent problem variables (they can be either a premise or a conclusion) and the edges represent conditional dependencies between such variables. Moreover, the probability function illustrates the strength of these relationships in the graph (Castillo et al., 1996) (Figure 2.2).

Formally, let a Bayesian Network B be defined as a pair, $B = (D, P)$, where D is a directed acyclic graph; $P = \{p(x_1|\Psi_1), ..., p(x_n|\Psi_n)\}$ is the set composed of n conditional probability functions (one for each variable); and Ψ_i is the set of parent nodes of the node X_i in D. The set P is defined as the *joint probability density function* (Castillo et al., 1996) (equation 2.2)

$$P(x) = \prod_{i=1}^{n} p(x_i|\Psi_i) \tag{2.2}$$

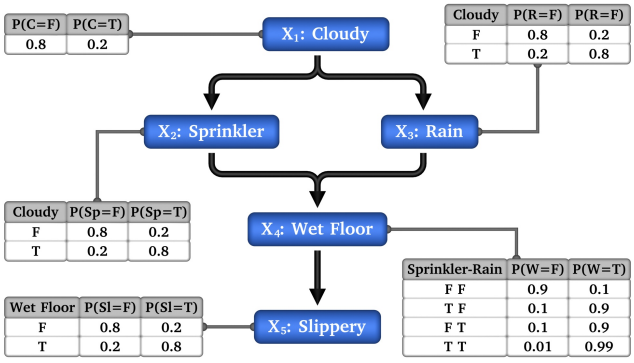


Figure 2.2: Example of a Bayesian Network, where the nodes represent problem variables and the links correspond to conditional dependencies between such variables.

The most important capability of Bayesian Networks is their ability to determine the probability that a certain hypothesis is true (e.g., the probability of an e-mail to be spam or legitimate) given a historical dataset, and is probably one of the most used algorithms for spam filtering (Sahami et al., 1998; Mason, 2002; Schneider, 2003; Burton, 2003; Meyer and Whateley, 2004; Hovold, 2005; Raymond, 2005; Metsis et al., 2006).

Decision Trees

Decision Trees that classify by means of automatically learned rule-sets have also been used for spam filtering (Carreras and Márquez, 2001; Rios and Zha, 2004). Decision Tree classifiers are a type of machine-learning classifiers that are graphically represented as trees (as shown in Figure 2.3). Internal nodes represent conditions regarding the variables of a problem, whereas final nodes or leaves represent the ultimate decision of the algorithm (Quinlan, 1986).

More formally, a decision tree graph $G = (V, E)$ is defined as a non empty set of finite *nodes* V and a set of *edges* E. If the set of edges is composed of ordered pairs, $< v, w >$, of vertices, then the graph is said to be *directed*. Moreover, a *path* is defined as an edge sequence of the form $(v_1, v_2), (v_2, v_3), ..., (v_{n-1}, v_n)$. A path is expressed by its origin, its end and the distance (i.e. the minimum number of edges from the origin to the end). Additionally, if the pair $< v, w >$ is an edge within the tree, v is said to be *parent* of w. Similarly, w can be defined as the *child node* of v. Further, nodes without parents are defined as *root nodes*. Every other node in the tree is then defined as an *internal node*. In order to build the tree representation, a set of binary questions (e.g., yes-no questions) is posed.

Different training methods are typically used for learning the graph structure of these machine-learning classifiers from a labelled dataset, such as, *Ran-*

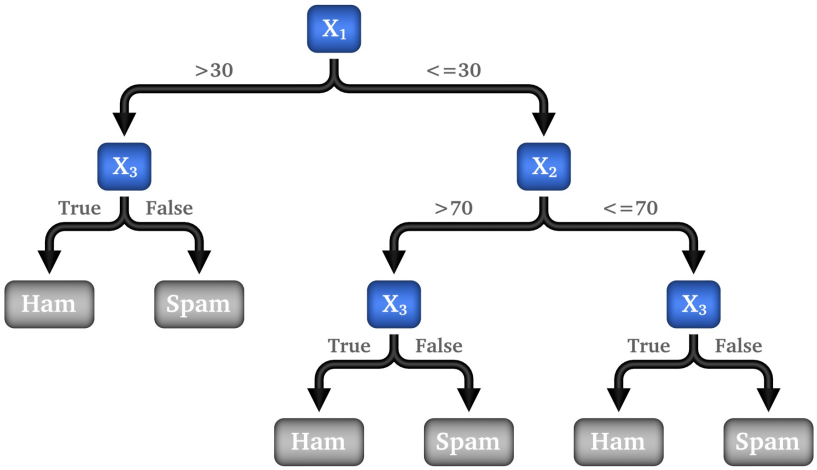


Figure 2.3: Example of a Decision Tree. The final leafs represent whether an e-mail is spam or ham (i.e. a legitimate message).

dom Forest, an ensemble (i.e., combination of other weak classifiers) of different randomly-built decision trees (Breiman, 2001), or the *C4.5* algorithm (Quinlan, 1993).

K-Nearest Neighbour

The K-Nearest Neighbour (KNN) (Fix and Hodges, 1952) classifier is one of the simplest supervised machine-learning models. This method classifies an unknown instance based on the class of the instance that is closest to itself in the training space (see Figure 2.4). Some examples of the application of KNN to the junk e-mail filtering problem are Nakov and Dobrikov (2004) and Yildiz et al. (2008).

Formally, the training phase of this algorithm represents the set of data for training $S = \{s_1, s_2, .., s_m\}$ in an n-dimensional space where n is the number of features of each instance (e.g., the frequency of occurrence of an interpretation).

The classification of an unknown instance (i.e., its class is not known beforehand) is accomplished by measuring the distance between the training instances and the unknown instance. The measure of the distance between two points X and Y in an n-dimensional space, can be performed using common metrics, for example, *Euclidean Distance* (shown in equation 2.3).

$$\sqrt{\sum_{i=0}^{n}(X_i - Y_i)^2} \tag{2.3}$$

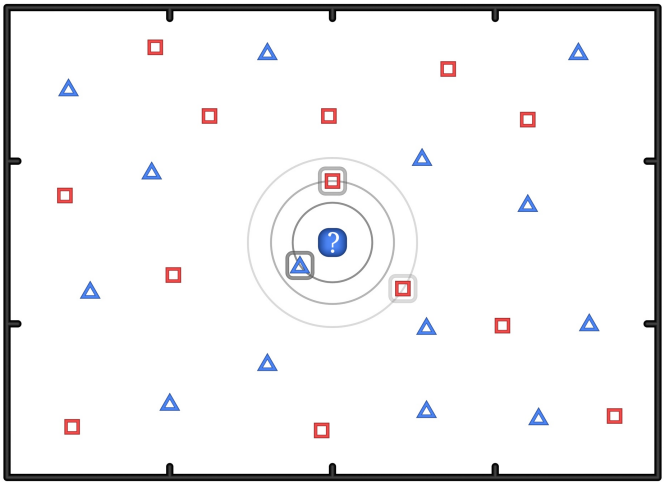


Figure 2.4: Example of a KNN classifier in a bidimensional space. A knew instance is classified based on the class of the instance closest to it in the training space.

Finally, even though several methods exist to choose the class of the unknown sample, the most common technique is to simply classify the unknown instance as the most common class amongst the K-nearest neighbours.

Support Vector Machines

Support Vector Machines (SVM) algorithms divide the n-dimensional space representation of the data into two regions using a *hyperplane* (as shown in Figure 2.5). This hyperplane always maximises the *margin* between those two regions or classes. The margin is defined by the farthest distance between the examples of the two classes and computed based on the distance between the closest instances of both classes, which are called *supporting vectors* (Vapnik, 2000).

More formally, let the optimal hyperplane be represented by a vector w and a scalar m in a way that the inner products of w with vectors $\phi(X_i)$ of the two classes are divided by an interval between -1 and $+1$ subject to m:

$$(w, \phi(X_i)) - m \geq +1 \tag{2.4}$$

for each X_i from the first class and

$$(w, \phi(X_i)) - m \leq -1 \tag{2.5}$$

for each X_i from the second class.

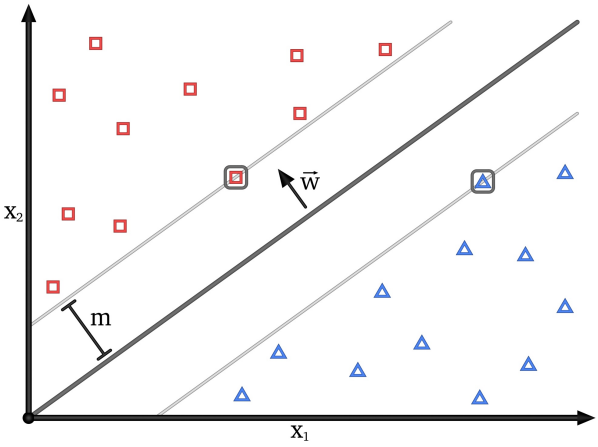


Figure 2.5: Example of a SVM classifier in a bi-dimensional space. The hyperplane that divides the space always maximises the margin between those two regions or classes.

This optimisation problem for finding w and m may be stated in terms of finding the inner products $\phi(x_i)$, $\phi(x_j)$ between data points of the training space. Instead of using inner products, it is common to use the so-called *kernel functions*. These kernel functions lead to non-linear classification surfaces, such as polynomial, radial or sigmoid surfaces (Amari and Wu, 1999). Examples of methods based on SVM for spam filtering can be found at (Drucker et al., 1999) and (Rios and Zha, 2004).

2.3.6.2 Semi-supervised approaches

Semi-Supervised Learning (SSL) is halfway between supervised and unsupervised learning (Zhu, 2005; Chapelle et al., 2006). In addition to unlabelled data, these methods use some supervision information such as the association of the targets with some of the examples. Through this section, we present literature about the use of SSL for unsolicited e-mail filtering and related domains.

Graph-based Algorithms

Graph-based semi-supervised methods define a graph where the nodes are labelled and unlabelled examples in the dataset, and edges (may be weighted) reflect the similarity of examples (Zhu, 2005). These methods usually assume label smoothness over the graph.

A standard graph-based SSL algorithm is the so-called *Learning with Local and Global Consistency* (LLGC) (Zhou et al., 2004), which tries to balance two

potentially conflicting goals: (i) nearby points are likely to have the same label and (ii) points on the same structure are likely to have the same label (Zhou et al., 2004).

Formally, the LLGC algorithm (see Figure 2.6) is stated as follows. Let $\mathscr{X} = \{x_1, x_2, ..., x_{\ell-1}, x_\ell\} \subset \mathbb{R}^m$ be the set composed of the data instances and $\mathscr{L} = \{1, ..., c\}$ the set of labels (in our case, spam and legitimate e-mails) and $x_u(\ell + 1 \leq u \leq n)$ the unlabelled instances. The goal of LLGC is to predict the class of the unlabelled instances. $\mathscr{F}$ is the set of $n \times c$ matrices with non-negative entries, composed of matrices $F = [F_1^T, ..., F_n^T]^T$ that match to the classification on the dataset $\mathscr{X}$ of each instance x_i, with the label assigned by $y_i = \text{argmax}_{j \leq c} F_{i,j}$. F can be defined as a vectorial function such as $F : \mathscr{X} \rightarrow \mathbb{R}^c$ to assign a vector F_i to the instances x_i. Y is an $n \times c$ matrix such as $Y \in F$ with $Y_{i,j} = 1$ when x_i is labelled as $y_i = j$ and $Y_{i,j} = 0$ otherwise.

if $i \neq j$ *and* $W_{i,i} = 0$ **then**
  Form the affinity matrix W defined by $W_{i,j} = \exp\left(\frac{-||x_i - x_j||^2}{2 \cdot \sigma^2}\right)$;
end
Generate the matrix $S = D^{-1/2} \cdot W \cdot D^{-1/2}$ where D is the diagonal matrix with its (i,i) element equal to the sum of the i-th row of W;
while $\neg$ *Convergence* **do**
  $F(t+1) = \alpha \cdot S \cdot F(t) + (1 - \alpha) \cdot Y$ where α is in the range $(0, 1)$;
end
F^* is the limit of the sequence $\{F(t)\}$;
Label each point x_i as $\text{argmax}_{j \leq c} F^*_{i,j}$;

Figure 2.6: LLGC algorithm.

The algorithm first defines a pairwise relationship W on the dataset $\mathscr{X}$ setting the diagonal elements to zero. Suppose that a graph $G = (V, E)$ is defined within $\mathscr{X}$, where the vertex set V is equal to $\mathscr{X}$ and the edge set $\mathscr{E}$ is weighted by the values in W. Next, the algorithm normalises symmetrically the matrix W of G. This step is mandatory to assure the convergence of the iteration. During each iteration each instance receives the information from its nearby instances while it keeps its initial information. The parameter α denotes the relative amount of the information from the nearest instances and the initial class information of each instance. The information is spread symmetrically because S is a symmetric matrix. Finally, the algorithm sets the class of each unlabelled specimen to the class of which it has received most information during the iteration process.

Pfahringer (2006) successfully applied the LLGC algorithm to the spam filtering problem within the *2006 ECML-PKDD Discovery Challenge* sharing the first rank of the *Spam Filtering Performance Award - Task A*.

Transductive Learning

Vapnik (1998) introduced the concept of transductive inference, a special case of SSL. The main characteristic is that, while inductive approaches try to create generalised models able to classify unknown instances, transductive-learning methods are designed to be used when the messages to be classified are known at the time of classifier construction Joachims (1999). In this way, a learner is transductive if it only works on the labelled and unlabelled training data, and cannot handle unseen data.

The main goal of transductive learning is to tackle the problem of inductive approaches of learning from small training samples. Some examples of their application to the spam domain are (Xu and Zhou, 2007) and (Shunli and Qingshuang, 2010).

Self-Training Approaches

In self-training a classifier is first trained with a small amount of labelled data and is then used to classify the unlabelled data (Zhu, 2005). Typically the most confident unlabelled points, together with their predicted labels, are added to the training set. The classifier is re-trained and the procedure repeated. Note the classifier uses its own predictions to teach itself.

Self-training has been applied to different problems such as word sense disambiguation (Yarowsky, 1995), subjective nouns identification (Riloff and Wiebe, 2003), dialogue classification (Maeireizo et al., 2004), object detection systems from images (Rosenberg et al., 2005) or unsolicited e-mail filtering (Cormack, 2006; Junejo et al., 2006).

Multiview Learning

Multiview learning has a long history (De Sa, 1994). It is based on the training of multiple hypotheses with different inductive biases (e.g., Decision Trees, SVMs, etc.) from the same labelled data set, and the requisite to make similar predictions on any given unlabelled instance (Zhu, 2005). By minimising the disagreement between the outputs of the different models multiview learning sensibly approximates to minimising the number of misclassification in each model, leading to similar results.

It has been applied to semi-supervised regression (Sindhwani et al., 2005), the more challenging structured output spaces (Brefeld et al., 2005; Brefeld and Scheffer, 2006) and spam filtering (Mavroeidis et al., 2006).

2.4 Other techniques

This section presents some literature that despite not corresponding to spam filtering itself is very related and important to the area of unsolicited e-mail filtering.

More accurately, we describe text categorisation, which provides the area of content-based unsolicited e-mail filtering with very useful techniques. Indeed, content-based spam filtering is usually considered as a binary text categorisation problem, in which a document belongs to one of two classes, spam or legitimate. Besides, we provide background regarding the information filtering process, a process that is followed during the experimental phases of the approaches presented in this work.

2.4.1 Text categorisation

Text categorisation, or text classification, consists in the classification of documents within previously defined categories. Manual categorisation is a common practice in multiple environments, but the number of experts needed to perform this task, and the increasingly amount of electronic information available, has lead to the interest on the automation of the process. Therefore, despite automatic text categorisation started back in the 60's (Maron, 1961), it has become a hot topic of research in recent years.

2.4.1.1 Automatic text categorisation

The task of text categorisation consists on the assignment of a boolean value to each (d_j, c_k) pair from $\mathscr{D}x\mathscr{C}$, where $\mathscr{D}$ is a collection of unclassified documents and $\mathscr{C}$ is a set of predefined categories (Sebastiani, 2002). In this way, assigning a value to (d_j, c_k) is interpreted as d_j is classified within the category c_k. Formally, automatic text categorisation consists in the approximation of an unknown function $\Phi : \mathscr{D}x\mathscr{C} \rightarrow T, F$, which describes how documents should be correctly classified by means of a function $\overline{\Phi} : \mathscr{D}x\mathscr{C} \rightarrow T, F$, called *classifier* or *model*.

Assigning a category to a document is a subjective task, hence, given a document, the number of corresponding categories may be variable. This fact is also found in manual categorisation. A clear example is the *inter-indexer inconsistency* (Cleverdon, 1984), a consequence of the discrepancy between two human experts trying to classify an unknown document within a given set of categories.

Nevertheless, this issue does not affect this work because the unsolicited e-mail filtering task is a binary classification problem, in which each document (e-mail in our case) is classified within a predefined set of known categories (i.e., spam or legitimate), not being possible for a document to belong to both categories.

2.4.1.2 Knowledge-based and learning-based categorisation

When building automatic classifiers, there are two main approaches: one based on knowledge and the other one based on learning.

The former is based on a set of rules that implement the knowledge from human experts (Hayes and Weinstein, 1990). The problems of knowledge-based approaches are, firstly, the bottleneck of knowledge acquisition and, secondly, the high costs of development and the implicit system portability restrictions. Nevertheless, the categorisation effectiveness of knowledge-based systems is satisfactorily high.

The latter, that is, learning-based approaches, have been an interesting topic of research for the academia in the past decades. These kind of methods use a general inductive process that is able to automatically generate a model by learning, from a set of preclassified documents, the characteristics of the categories of interest (Sebastiani, 2002).

The learning process tries to identify the characteristics that a document should have to be classified within one category, taking into account a collection of documents previously classified by a human expert. The collection of previously classified documents is commonly referred to as *training set*. The required effort needed by learning-based approaches for development, maintenance and adaptation to different domains is considerably reduced with respect to knowledge-based approaches (Sebastiani, 2002). Besides, learning-based approaches offer similar accuracy to that obtained by human experts, while saving considerable manpower because no intervention is needed to build the required classifiers.

2.4.1.3 Applications of automatic text categorisation

After the work of Maron (1961) on probabilistic text categorisation, several applications emerged to address different problems. In the remainder of this section we review some of these problems including: *document organisation*, *text filtering*, *word sense disambiguation* and *hierarchical categorisation of web pages*.

Document organisation

Through the automatic categorisation of text anyone can create and maintain, in a relatively inexpensive way, hierarchical structures (taxonomies) to organise information and facilitate the access to documents (Chen and Dumais, 2000; Adams, 2001).

On the one hand, taxonomies mitigate the complexities of representation in IR by contextualising the search and management of knowledge processes. On the other hand, documents taxonomies provide a common language for groups of people, allowing their members to easily relate and access concepts (Labrou

and Finin, 1999). In summary, document organisation facilitates the access to information, be it resources from physical or digital libraries, files provided within private networks or available information in the World Wide Web.

Web pages categorisation

The interest in automatic classification of web pages or sites has been of great interest since the last decade (Koller and Sahami, 1997; Mladenic, 1998; Attardi et al., 1998, 1999; Klas and Fuhr, 2000) until recent years (Chen and Hsieh, 2006; Qi and Davison, 2009; Tiun et al., 2010; Rajan et al., 2010; Savoy and Zubaryeva, 2011).

In the past, big directories such as *Yahoo!* or the *Internet Directory Project* were created and maintained by humans. This manual work was very expensive and poorly scalable to the current dimensions of information available on the Internet. Therefore, the creation and maintenance of web directories is a problem that text categorisation is successfully being applied to.

Text filtering

Text filtering is the activity of classifying a stream of incoming documents dispatched in an asynchronous way by an information producer to an information consumer (Belkin and Croft, 1996). A typical case is a news feed, where the producer is a news agency and the consumer is a newspaper (Hayes and Weinstein, 1990). Similarly, an e-mail filter may be trained to discard unsolicited messages (Drucker et al., 1999; Androutsopoulos et al., 2000b).

Web content filtering

Web filtering approaches are deployed to detect and limit general Internet abuse. Their techniques are increasingly sophisticated and effective, and their development is contributing to the advance of the state-of-the-art in a number of research fields, like text analysis and image processing (Gómez Hidalgo et al., 2009).

Word sense disambiguation

The task of sense disambiguation, or word sense disambiguation (WSD) is the process of identifying the most appropriate meaning for a polysemous word given a specific context. WSD is considered an “intermediate task” (Wilks and Stevenson, 1996), not final itself but necessary at certain levels for many natural language processing tasks. Moreover, WSD is very important for many applications, including natural language processing, and indexing documents by word senses rather than by words for IR purposes (Sebastiani, 2002).

Other applications

Many other applications exist that make use of text classification techniques. For example, text categorisation has been used to help computer users organise and manage their own information. Koprinska et al. (2007) provide a method to automatically classify e-mail messages. Other approaches have managed to automatically grade students' essays in education environments (Larkey, 1998) or to automatically identify the author of a literary text for unknown or disputed authorships (Forsyth, 1999; Teahan, 2000).

Other applications of text categorisation include: speech categorisation by means of a combination of speech recognition and text categorisation (Myers et al., 2000; Schapire and Singer, 2000), multimedia document categorisation by analysing textual captions (Sable and Hatzivassiloglou, 2000) or language identification of documents (Cavnar and Trenkle, 1994).

2.4.2 Knowledge-based automatic e-mail filtering

Information filtering is a name used to describe a variety of processes involving the delivery of information to people who need it (Belkin and Croft, 1996). As mentioned before, spam filtering is a binary text categorisation problem, in which a document belongs to one of two classes, spam or legitimate. After an anti-spam system categorises a received e-mail, it is discarded if if it is spam or offered to the user if it is legitimate, that is, every message is filtered according to its content.

Knowledge-based automatic filtering systems follow a general scheme with two main processes: *model training* and *unknown e-mail classification*. In the remainder of this section we present this scheme and the corresponding processes.

2.4.2.1 General scheme

Based on a general model of IR, Belkin and Croft (1996) presented a model of information filtering that describes the major entities and processes involved in the task. We show in Figure 2.7 an adaptation to spam filtering of that scheme.

The two main processes identified are:

- **Classifier building:** The first step is to gather as many e-mails as possible. Those e-mails must be labelled (i.e., determine if they are legitimate or spam), usually a task performed by humans (supervised approaches) or humans and machine-learning algorithms (semi-supervised approaches). Once the e-mails are correctly labelled, they must be analysed in order to find the best way to represent them so that the classifier is able to "understand" them. With the representation obtained, a classification model, or classifier, is built using machine-learning algorithms. Note that several classifiers may be built, using different algorithms, as some may behave

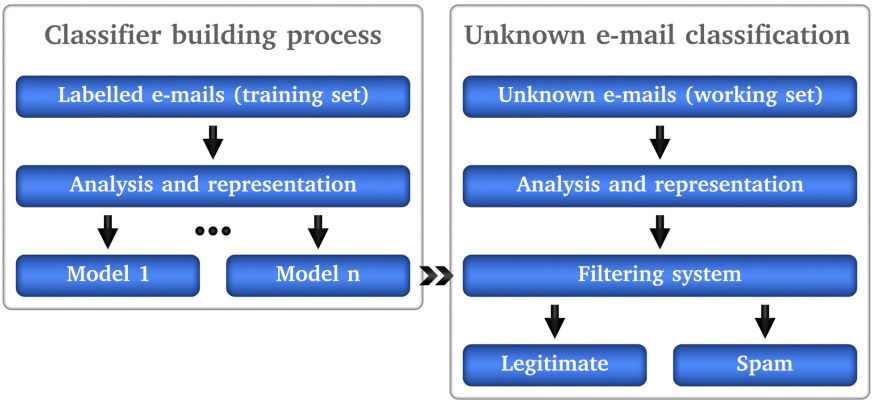


Figure 2.7: General scheme of the two main processes of a spam filtering system. This scheme is an adaptation of the scheme proposed by Belkin and Croft (1996).

better than others in different conditions or may offer different performing levels that may be more appropriate for distinct environments or situations.

- **Unknown e-mail classification:** When new unlabelled e-mails arrive, they must be analysed and represented so that the classifier is able to determine its nature, that is, spam or legitimate. Once the filtering system has the unknown e-mails' representation, it queries the model, or models, to obtain the corresponding label for each message. The anti-spam system is then able to provide the user with the legitimate messages, and discard the spam.

The unknown e-mail classification process depends on the previously created model, in this way, the task of creating the classifier is usually referred to as *training*. In a same manner, the collection of labelled e-mails employed to build the model is referred to as *training set*.

The representation of the e-mails performed during the training and before the classification of unknown documents, e-mails in our case, includes several steps (Sebastiani, 2002): e-mail representation, feature selection and the application of machine-learning techniques to build the model. We already introduced some machine-learning approaches in Section 2.3.6. Next, we introduce the other two processes.

2.4.2.2 Vector Space Model for e-mail representation

Usually, e-mails are represented using an IR model. Formally, let an IR model be defined as a 4-tuple $[\mathscr{E}, \mathscr{Q}, \mathscr{F}, R, (q_i, e_j)]$ (Baeza-Yates and Ribeiro-Neto, 1999) where

- $\mathscr{E}$ is a set of representations of e-mails.
- $\mathscr{Q}$ is a set of representations of user queries.
- $\mathscr{F}$ 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, $(q_i \in \mathscr{Q})$ and an e-mail representation e_j, $(e_j \in \mathscr{E})$. This function is also called a similarity function.

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

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

Specifically, throughout this work, we use "term frequency - inverse document frequency" $(tf - idf)$ (Salton and McGill, 1983) to obtain the weight of each word, whereas the weight of the i^{th} word in the j^{th} e-mail, denoted by $weight(i, j)$, is defined by

$$weight(i, j) = tf_{i,j} \cdot idf_i \tag{2.6}$$

where the term frequency $tf_{i,j}$ (Salton and McGill, 1983) is defined as

$$tf_{i,j} = \frac{m_{i,j}}{\sum_k m_{k,j}} \tag{2.7}$$

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

Conversely, the inverse document frequency idf_i is defined as

$$idf_i = \frac{|\mathscr{E}|}{|\mathscr{E} : t_i \in e|} \tag{2.8}$$

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

2.4.2.3 **Feature selection**

Automatic feature selection methods include the removal of non-informative terms according to corpus statistics, and the construction of new features which combine lower level features (i.e., terms) into higher-level orthogonal dimensions (Yang and Pedersen, 1997). Feature selection techniques are designed to provide a subset of the original set so that the resulting features possess a relevance for prediction equal or even greater with respect to the original. Yang and Pedersen (1997) showed that feature selection may lead to moderate increases on the effectiveness of machine-learning classifiers, in addition to the increase of the performance of the system.

There are several feature selection algorithms, e.g., *document frequency thresholding* (Yang and Pedersen, 1997; Sebastiani, 2002), *information gain* (Lewis, 1992; Larkey, 1998), *mutual information* (Lewis and Ringuette, 1994; Moulinier et al., 1996; Larkey and Croft, 1996; Dumais et al., 1998), *chi-square (χ^2)* (Galavotti et al., 2000; Caropreso et al., 2001), *term strength* (Yang and Pedersen, 1997) or *relevancy score* (Wiener et al., 1995). According to Yang and Pedersen (1997), information gain (IG), along with χ^2, is the most effective in aggressive term removal without losing categorisation accuracy, therefore along this work we use IG as the feature selection algorithm.

IG (Kent, 1983) is defined as

$$IG(j) = \sum_{v_j \in R} \sum_{C_i} P(v_j, C_i) \cdot \frac{P(v_j, C_i)}{P(v_j) \cdot P(C_i)} \tag{2.9}$$

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

2.5 **Summary**

This chapter presented current literature related to the area of spam filtering. Firstly, we introduced some general concepts and offered a classification of anti-spam methods in order to give the reader a general knowledge of the possible lines to follow. Some approaches try to fight the spam problem by "acting" before the e-mail is sent, others try to improve the channel of communication and the latest try to provide with useful tools to detect and filter undesired e-mails assuming spam has already been sent.

Secondly, we presented some of the most relevant approaches within the area of content-based unsolicited e-mail filtering, from the most simple techniques,

heuristic approaches or url analysis, to the more complex machine-learning-based methods. Finally, we included some literature that despite not corresponding to anti-spam systems itself is very related to the area of unsolicited e-mail filtering. Because content-based spam filtering is usually considered as a binary text categorisation problem, we reviewed some basic concepts of text categorisation. To conclude the chapter, we presented the general scheme of a common knowledge-based automatic unsolicited e-mail filtering system.

"As we acquire more knowledge, things do not become more comprehensible but more mysterious."

Albert Schweitzer
(1875 – 1965)

3

Semantic-aware unsolicited e-mail filtering

REGULARLY, spam filtering approaches rely on statistical methods assuming that every term is independent, which is, at least from the linguistic point of view, not completely true. Thus, anti-spam systems cannot handle the existing linguistic phenomena in natural languages. This chapter is devoted to provide two approaches proposed to further develop spam filtering systems in order to make them sensitive to the semantics present in e-mails. First, we present a spam filtering method able of representing linguistic relationships between terms. Second, we introduce an unsolicited e-mail filtering system capable of dealing with word sense ambiguity.

The remainder of this chapter is organised as follows. Section 3.1 introduces the problem that linguistic phenomena produce in anti-spam systems. Section 3.2 presents a model capable of dealing with the linguistic phenomena that encode semantic word relations to enhance current machine-learning methods. Section 3.3 proposes the disambiguation of terms to improve filtering capabilities. Section 3.4 concludes and reviews the identified open lines for future works. Finally, section 3.5 summarises the main aspects of this chapter.

3.1 The problem of semantics

Electronic mail is a powerful communication channel. Nevertheless, as happens with all useful media, it is prone to misuse. 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, more than 85% 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).

Owing to the magnitude of the spam issue, several unsolicited e-mail filtering systems have been proposed and a large amount of research has been dedicated to finding more effective anti-spam solutions. Machine-learning approaches have been effectively applied to text categorisation problems (Sebastiani, 2002), and they have been adopted for use in junk e-mail filtering systems. Consequently, substantial work has been dedicated to naïve Bayes filtering (Lewis, 1998), with several published studies supporting its effectiveness (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 unsolicited e-mail 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). This model represents natural language documents mathematically as vectors in a multidimensional space where the axes are terms within messages. 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 and Kuropka, 2003).

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):

- **Synonymity:** Two or more words are interchangeable because of their similar (or identical) meanings (e.g., *buy* and *purchase*) (Carnap, 1955).

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

- **Inflection:** A word is modified to express different grammatical categories such as gender, aspect, number or case (e.g., *buy*, *bought*) (Stump, 2001).
- **Compounding:** Word formation by combining or putting together two or more words (e.g., *football*, *blackboard* or *highlight*) (Plag, 2003).
- **Derivation:** Process of creating new words on the basis of an existing word (e.g., *recreation - recreational* or *do - undo*) (Crystal, 1999).
- **Hyponymy:** Specific instances of a more general word (e.g., *spicy* and *salty* are hyponyms of *flavour*)(Cruse, 1975).
- **Meronymy:** A semantic relation denoting a word constituent part of or member of something (e.g., *wheel* is a meronym of *automobile*) (Bunt, 1985).
- **Homography:** Words with the same orthography but different meaning (e.g., *bear*: "to support and carry" and "an animal") (Ming-Tzu and Nation, 2004).
- **Metonymy:** The substitution of one word for another with which it is associated (e.g., *police* instead of *law enforcement*) (Radden and Kövecses, 1999).
- **Word-groups:** Clusters of words that have particular semantic meanings when they are grouped together (e.g., *New York City*).

In a similar vein, the VSM is also affected by other characteristics of the text such as *word sense ambiguity* (Sanderson, 1994), which can confuse the model. Indeed, today's attacks against Bayesian spam filters attempt to keep the content of junk messages visible to humans, but obscured to filters. For instance, attackers circumvent these anti-spam systems by replacing suspicious words by innocuous terms with the same meaning (Karlberger et al., 2007; Nelson et al., 2009).

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).

The following sections introduce our two proposed approaches to overcome the first linguistic phenomena (i.e, synonymity), employing a representation of e-mail messages that uses *interpretations* rather than *terms*; and to reduce the impact of word sense ambiguity, by applying a pre-processing procedure that is able to disambiguate confusing terms, to improve the capabilities of anti-spam systems.

3.2 Enhanced Topic-based Vector Space Model for semantics-aware spam filtering

As stated before, the VSM cannot handle the existing linguistic phenomena in natural languages (Becker and Kuropka, 2003). Because of this shortcoming, the *Topic-based Vector Space Model* (TVSM) (Becker and 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 the 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 section is organised as follows. Section 3.2.1 describes in a technical and detailed manner the process of adapting the eTVSM for representing e-mails. Section 3.2.2 details the experiments performed and presents the results. Finally, Section 3.2.4 discusses the main shortcomings and outlines avenues for future work.

3.2.1 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). Through the study of how these phenomena affect spam-filtering systems, we are able to build a *semantics-aware* model.

In this work, 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 and 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 and 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 and 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; and $R(q_i, e_j)$ is a ranking function that associates a real number with a query q_i, $(q_i \in Q)$ and e-mail representation e_j, $(e_j \in E)$ (a *similarity function*).

As seen in Section 2.4.2.2 an e-mail can be represented as a vector, starting from its origin, of index terms $\vec{e_j} = (w_{1,j}, w_{2,j}, ...w_{n,j})$. Then we can apply the *Vector Space Model* (VSM) to represent the e-mail in an algebraic fashion by placing the vectors in a multidimensional space (see Section 2.4.2.2). 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.

The *Topic-based Vector Space Model* (TVSM) (Becker and Kuropka, 2003) also represents documents as vectors in an n dimensional space R (shown in equation 3.1) 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).

$$R \in \mathbb{R}^d_{\geq 0} \text{ with } e \in \mathbb{N}_{\geq 0} \tag{3.1}$$

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:

$$\begin{aligned} &\vec{t_i} = \left(t_{i,1}, t_{i,2}, ..., t_{i,n}\right) \\ &\text{with } |\vec{t_i}| = \sqrt{t_{i,1}^2 + t_{i,2}^2 + ... + t_{i,n^2}} \in [0;1] \end{aligned} \tag{3.2}$$

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 and 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). The relations between the given concepts are expressed through concept vector angles, representing semantic similarity between concepts. Furthermore, eTVSM is capable of taking advantage of the ontology concepts, by transforming them into vectors in an operational vector space, that maps the 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 and 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 the inter-topic relations. Intermediate links between topics and terms correspond to interpretations, which play the role of semantic terms. The terms can be linked to an arbitrary number of topics and cannot be linked to other interpretations. Finally, the terms are treated as the smallest information unit with one or several semantic interpretations. In order to express all the existing semantic meanings, a term might be linked with an arbitrary number of interpretations.

For instance, on the basis of Figure 3.1, “Pornography” and “Biology” represent two different topics, where the former topic is commonly observed in spam e-mail messages, and “Intercourse”, “Sex?” and “Gender” are three interpretations of the leaf concepts in the ontology extract (i.e., the corresponding terms). The term “Sex” can have two clear interpretations, namely “Intercourse” or “Gender”. Therefore, “Sex?” is designed for cases when it is impossible to distinguish the meaning clearly, resulting in a *Word Sense Disambiguation* (WSD) problem.

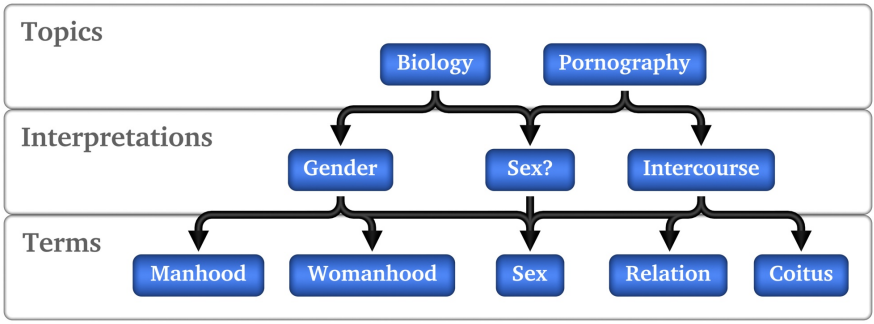


Figure 3.1: Example of ontology extract.

Although the eTVSM does not resolve this issue, elucidating the context of an ambiguous word is a prerequisite for complete semantic awareness. The term "Sex" used in a pornographic context presents a high probability that the e-mail is spam, while the same term in a biological context does not. We discuss this problem in more detail in Section 3.2.4.

For our purposes, we make the most of the *Themis*[2] 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.

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:

$$\begin{aligned} &\forall e_j \in E : \vec{e_j} = \frac{1}{|\vec{\delta_j}|} \cdot \vec{\delta_j} \Rightarrow |\vec{e_j}| = 1 \\ &\text{and} \qquad \vec{\delta_j} = \sum_{\phi_i \in \Phi} \omega_{e_j,\phi_i} \cdot \vec{\phi_i} \end{aligned} \tag{3.3}$$

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:

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

$$
\begin{aligned}
|\vec{\delta_i}| &= \left| \sum_{\phi_k \in \Phi} \omega_{e_i,\phi_k} \vec{\phi_k} \right| = \sqrt{\left| \sum_{\phi_k \in \Phi} \omega_{e_i,\phi_k} \vec{\phi_k} \right|^2} \\
&= \sqrt{\left(\sum_{\phi_k \in \Phi} \omega_{e_i,\phi_k} \vec{\phi_k} \right)^2} \\
&= \sqrt{\sum_{\phi_k \in \Phi} \sum_{\phi_l \in \Phi} \omega_{e_i,\phi_k} \omega_{e_i,\phi_l} \vec{\phi_k} \vec{\phi_l}}
\end{aligned}
\tag{3.4}
$$

In this way, our proposed method retrieves the weighted interpretation vectors representing e-mail 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) file (Holmes et al., 1994) (Garner, 1995a) to build the desired machine-learning classifier. Finally, we test different machine-learning classification algorithms with WEKA as described in Section 3.2.2.

3.2.2 **Empirical validation**

In order to validate our proposed method, we used the *Ling Spam* dataset (see Appendix A). 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.[3]

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 the WordNet ontology[4] 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).

We constructed a file with the resultant vector representations of the e-mails in order to validate our method. We extracted the top 1,000 attributes using IG

[3] http://www.webconfs.com/stop-words.php

[4] http://wordnet.princeton.edu/wordnet/download/

(see Section 2.4.2.3), an algorithm that evaluates the relevance of an attribute by measuring the information gain with respect to the class.

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 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 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., 2000a,b; Schneider, 2003; 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* (Breiman, 2001) (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[5] 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 times by each configuration to perform the training and testing phases:

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

- *Training time:* The overhead required for building the different 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:

$$Accuracy(\%) = \frac{TP + TN}{TP + FP + TP + TN} \cdot 100 \tag{3.5}$$

where TP is the amount of correctly classified unsolicited messages (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 junk e-mails, 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:

$$TPR = \frac{TP}{TP + FN} \tag{3.6}$$

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:

$$FPR = \frac{FP}{FP + TN} \tag{3.7}$$

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 3.1 shows the time results for training and testing of the supervised machine-learning models. 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.

Table 3.1: Time results of machine-learning classifiers based on eTVSM representation spam detection.

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

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 3.2 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. Two 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

Table 3.2: Results of the evaluation of machine-learning classifiers based on eTVSM representation spam detection.

Classifier	Accuracy (%)	TPR	FPR	AUC
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

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.

3.2.3 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 (commercial or open-source) and academic approaches (see Table 3.3).

Table 3.3: 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 and Lynam, 2007)				
SpamAssassin	84.1	0.04	0	0.52
BogoFilter	90.1	0.40	0	0.70
SpamProbe	94.8	0.69	0	0.84
CRM114	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

Cormack and Lynam (2007) tested the real-world spam filtering tools SpamAssassin[6], Bogofilter[7], SpamProbe[8] and CRM114[9] against the Ling Spam corpus. Overall, these anti-spam tools were unable to classify electronic mails correctly.

Androutsopoulos et al. (2000a) proposed a model based on the Naïve Bayes classifier. The classifier was constructed using the terms within the e-mails. The parameter λ represents the cost of misclassifying a legitimate e-mail as spam compared to misclassifying a spam message as legitimate. In other words, a value of lambda of 9 means that misclassifying a legitimate e-mail as spam is 9 times worse than the opposite mistake.

The method proposed by Sakkis et al. (2001) extended the previous approach

[6] http://spamassassin.apache.org
[7] http://bogofilter.sourceforge.net
[8] spamprobe.sourceforge.net
[9] http://crm114.sourceforge.net

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 electronic mails.

Schneider (2003) performed experiments with a Naïve Bayes text classifier using two classifier statistical event models (McCallum and 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 3.3).

3.2.4 **Discussion**

The final results obtained with the use of eTVSM for the representation of e-mail messages to detect spam show that our approach 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 e-mails 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 nec-

essary in order to accomplish most natural language processing tasks (Ide and 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 and 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 and 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 and 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 we think semantics will be a topic of research and concern in the coming years.

3.3 Word Sense Disambiguation for spam filtering

As stated before, current unsolicited e-mail filtering systems feature machine-learning, which model e-mail messages using the VSM (Salton et al., 1975). 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 the application of WSD for spam filtering to recover the filtering capabilities of content-based methods. Our approach pre-processes 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 section is organised as follows. Section 3.3.1 describes the problem of WSD and the effects that ambiguity has on spam filtering systems. Section 3.3.2 introduces our method to improve detection rates by using WSD. Section 3.3.3 provides an empirical evaluation of the experiments performed and presents the results. Finally, Section 3.3.4 discusses the main shortcomings and outlines the avenues for future work.

3.3.1 The problem of disambiguation

The task of disambiguating word senses is the process of identifying the most appropriate meaning of a polysemous word given a specific context. The 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, 1955). Very soon it became clear that it 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 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*[10], 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.

3.3.2 Our Word Sense Disambiguation approach

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 3.3.3) 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 (Miller et al., 1993). We used for our exper-

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

iments the models provided with the distribution of SenseLearner, which were trained on *SemCor* (Miller et al., 1993), 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 3.3.4.

In this way, we formally define an e-mail $\mathscr{M}$ as a set composed of n terms t_i, $\mathscr{M} = \{t_1, t_2, \ldots, t_{n-1}, t_n\}$, 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, 1995b) to build the desired classifiers. Finally, we test different machine-learning classification algorithms with WEKA as described in Section 3.3.3.

3.3.3 Empirical validation

We employed the *Ling Spam* dataset[11] and the *TREC 2007 Public Corpus*[12] (see Appendix A) separately, as the spam corpus in two different experiments, applying our proposed approach to both of them.

For our experiments with TREC, 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. Table 3.4 presents an overview of the dimensions of the datasets.

Table 3.4: Comparison of the used datasets in the disambiguation approach. The spam ratio in the datasets does not follow the statistics of the number of spam messages in the real world which is higher than 85%. The TREC dataset, however, contains more realistic spam e-mails and examples of obfuscated mails within them.

Feature	Ling Spam	TREC
No. Spam Messages	480	14,973
No. of Ham Messages	2,412	7,653
Spam %.	16.60%	66.18%

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 3.3.2) 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 (see Section 2.4.2).

We also extracted from the model the top 1,000 attributes using IG (Kent, 1983), an algorithm that evaluates the relevance of an attribute by measuring the information gain with respect to the class (see Section 2.4.2). Figures 3.2 and 3.3 show the frequency of senses for the 1,000 attributes selected with IG for each of the datasets.

[11] http://nlp.cs.aueb.gr/software_and_datasets/lingspam_public.tar.gz
[12] http://plg.uwaterloo.ca/~gvcormac/spam

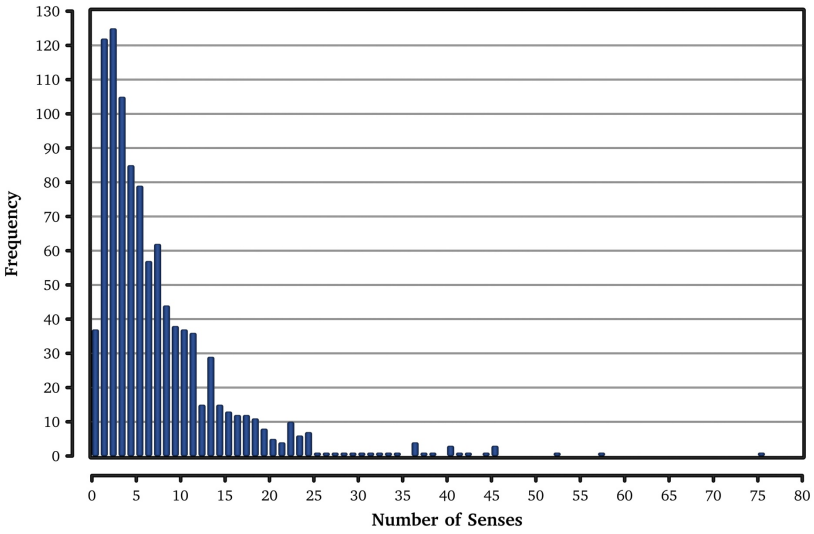


Figure 3.2: Frequency of senses for all selected top IG scoring attributes for Ling Spam dataset in WSD approach.

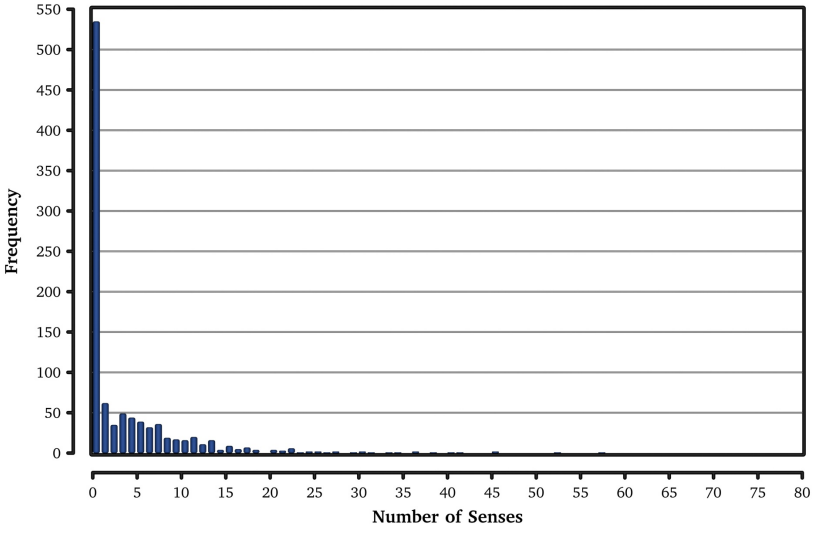


Figure 3.3: Frequency of senses for all selected top IG scoring attributes for TREC dataset in WSD approach.

It 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:

  - *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., 2000a,b; Schneider, 2003; 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* (Breiman, 2001) (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[13] 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:
  - *Training time:* The overhead required to build 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 defined and 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:

$$S_P = \frac{N_{s\to s}}{N_{s\to s} + N_{l\to s}} \tag{3.8}$$

  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:

$$S_R = \frac{N_{s\to s}}{N_{s\to s} + N_{s\to l}} \tag{3.9}$$

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

$$F\text{-}measure = \frac{2N_{s\to s}}{2N_{s\to s} + N_{s\to l} + N_{l\to s}} \tag{3.10}$$

  In addition, we measured the accuracy, which is the number of the classifier's hits divided by the total number of classified instances, and the *Area Under the ROC Curve* (AUC), which establishes the relation between false negatives and false positives (see Section 3.2.2 for a more detailed definition of accuracy and AUC).

Tables 3.5 and 3.6 show training and testing times for the different machine-learning classifiers. The kNN algorithm needs almost no time for training but is

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

Table 3.5: 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

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 Bayesian networks depended on the algorithm that is used. Overall, we found that K2 is the fastest Bayesian classifier to train, while testing times are similar for all. Among decision trees, Random Forest with 10, 50 and 100 trees trained faster than J48 only when using the Ling Spam dataset, while in testing, the fastest one was J48.

Tables 3.7 and 3.8 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

Table 3.6: 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

kNN algorithm improves, in most of the 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 disambiguation process 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, for 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

Table 3.7: Precision evaluation of machine-learning classifiers for the WSD approach. 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).

default option set on, also improving significantly for TREC, with no signs of significant variation in recall for either dataset. Lastly, the naïve Bayes classifier presents a vast improvement in terms of 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 3.9 offers the results for the 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

Table 3.8: Recall evaluation of machine-learning classifiers for the WSD approach. 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).

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

Table 3.9: Area under the ROC curve (AUC) evaluation of the machine-learning classifiers for WSD approach. 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).

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.

3.3.4 **Discussion**

The results obtained during the evaluation of our approach show that the preprocessing step of WSD, 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 WSD in spam filtering, there is a problem derived from IR and NLP 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.

3.4 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 section, we have firstly 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.

Secondly, we have presented the application of WSD for spam filtering to improve the detection capabilities of content-based methods. Our approach preprocesses the e-mail messages, disambiguating the terms before constructing the VSM. 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.

According to the limitations found in our proposed methods, we identified several open lines for future work that we next summarise:

- Multiple Instance Learning (MIL) (Dietterich et al., 1997) to fight attacks that modify term statistics (e.g., *Good Word Attack*).
- A solution against *tokenisation* attacks, which work against the feature selection of the message by splitting or modifying key message features, rendering the term representation as no longer feasible (Wittel and Wu, 2004).
- Improvement of our ambiguity-resilient approach by enriching the disambiguation process with a more complex PoS labelling format such as SemCor (Miller et al., 1993).
- The study of the problem of language dependency that semantic approaches suffer from (Bates and Weischedel, 1993).
- The study of the Topic Detection and Tracking (TDT) (Allan et al., 1998) to examine its applicability to spam filtering systems.

3.5 Summary

This chapter has introduced a problem that current unsolicited e-mail filtering systems have to face, that is, linguistic phenomena. We have presented two

approaches to face two different issues affecting these systems: synonymity and word sense ambiguity.

On the one hand, we have described how to improve the VSM representation by using the eTVSM, which by using topics rather than terms to create the models deals with the problem of synonymity. On the other hand, we have proposed a pre-processing procedure of WSD that is able to disambiguate confusing terms.

In summary, the empirical validation shows that both approaches improve the filtering capabilities of anti-spam systems. In this way, these approaches may be considered to complement current models in order to provide them with a semantic layer.

"No matter how complicated a problem is, it usually can be reduced to a simple, comprehensible form which is often the best solution."

An Wang
(1920 – 1990)

4

Reducing labelling efforts

In most cases, the solutions to fight the spam problem feature machine-learning algorithms, usually supervised, needing a training set of previously labelled samples. Dealing with situations in which the availability of labelled training instances is limited slows the filtering systems' progress and offers advantages to spammers. This chapter is devoted to provide two approaches to reduce the labelling efforts of the anti-spam filtering systems. First, we introduce Collective Classification algorithms for spam filtering, an interesting method for optimising the classification of partially-labelled data. Second, we present a study of the effectiveness of anomaly detection applied to spam filtering, which reduces the necessity of labelling spam messages.

The remainder of this chapter is organised as follows. Section 4.1 introduces the problem that actual spam filtering systems face due to the amount of unclassified messages. Section 4.2 presents Collective Classification, a semi-supervised approach that reduces the labelling efforts. Section 4.3 offers the study of the effectiveness of anomaly detection for spam filtering. Section 4.4 concludes and reviews the identified open lines for future works. Finally, section 4.5 summarises the main aspects of this chapter.

4.1 The problem of labelling

According to Radicati (2010) the number of worldwide e-mail accounts is projected to increase from over 2.9 billion in 2010, to over 3.8 billion by 2014. In a similar report, Radicati and Khmartseva (2009) estimated worldwide e-mail traffic totalled 247 billion messages per day in 2009, and that this figure would almost double to 507 billion messages per day by 2013. Moreover, as stated before, flooding inboxes with annoying and time-consuming messages, from the total amount of e-mail traffic, more than 85% of received e-mails are spam[1].

The academic community has proposed several approaches to solve the spam problem Robinson (2003); Chirita et al. (2005); Schryen (2006); Chiu et al. (2007). Among them, the statistical approaches Zhang et al. (2004) use machine-learning techniques to classify e-mails. These approaches have proven their efficiency in detecting spam and are the most utilised techniques to fight it.

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

In light of this background, several approaches have tried to overcome the problem of labelling in order to reduce the efforts required by spam filtering systems. The most common approach is the use of semi-supervised machine-learning algorithms Pfahringer (2006); Cormack (2006); Junejo et al. (2006); Mavroeidis et al. (2006); Xu and Zhou (2007); Shunli and Qingshuang (2010), which use both labelled and unlabelled data to build their training models.

The following sections introduce our two proposed approaches to reduce the labelling efforts by (i) adapting a semi-supervised text categorisation approach, i.e., Collective Classification, to spam filtering and (ii) applying anomaly detection techniques to detect and filter undesired e-mails.

4.2 A Collective Classification approach to reduce the labelling efforts

The approaches referred to as *statistical approaches* Zhang et al. (2004) use machine-learning techniques, commonly supervised, to classify e-mails. Because statistical techniques perform better when more training instances are available, a significant amount of previous labelling effort is needed to increase the models' accuracy.

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

A similarly arduous task is usually performed for text categorisation, or text classification. Because text categorisation uses the contents of documents and external sources to build accurate document classifiers, the scientific community has focused its attention on the link structure between documents (Dengel and Dubiel, 1995; Denoyer and Gallinari, 2004), in order to improve the performance of document classification.

The connections between documents vary from the common citation graph formed when papers cite other papers or when websites link to other websites, to the links constructed between relationships (e.g., co-authors, co-citations, or co-appearances at a conference venue). Combinations of these connections can be used to create interlinked collections of text documents.

It can be interesting not only to determine the topic of a single document but also to infer a topic for a collection of unlabelled documents. Collective classification attempts to collectively optimise the problem of topic determination by taking connections present among the documents into account. This type of classification is a semi-supervised technique, i.e., it uses both labelled and unlabelled data (typically, a small amount of labelled data and a large amount of unlabelled data); this reduces the work involved in labelling.

Given this background, we propose the first spam filtering system that uses Collective Classification to optimise classification performance. This approach minimises the necessity of labelled e-mails without compromising the accuracy of the filter.

In this section, we (i) describe a method for adopting Collective Classification in spam filtering; (ii) attempt to determine the optimal size of the labelled dataset for collective-classification-based spam filtering; (iii) compare our collective approach with commonly used supervised algorithms; and (iv) show that this approach can reduce the effort of labelling e-mails while maintaining a high accuracy rate.

The reminder of this section is organised as follows. Section 4.2.1 describes the process of using collective classification applied to the spam filtering problem. Section 4.2.2 details the experiments performed and presents the results. Finally, Section 4.2.4 discusses the main shortcomings and outlines avenues for future work.

4.2.1 **Collective Classification**

Collective Classification is a direct consequence of the great effort in the scientific community directed towards the link structure among documents Dengel and Dubiel (1995); Denoyer and Gallinari (2004), to improve the performance of document classification. To our knowledge, the use of collective algorithms in anti-spam systems has not been addressed yet, so our work presents the first application of collective classification techniques to the spam filtering problem.

Collective classification is a combinatorial optimisation problem, in which we are given a set of documents or nodes, $\mathscr{D} = \{d_1, ..., d_n\}$ and a neighbourhood function N, where $N_i \subseteq \mathscr{D} \setminus \{\mathscr{D}_i\}$, which describes the underlying network structure Namata et al. (2009). $\mathscr{D}$, a random collection of documents, is divided into two sets $\mathscr{X}$ and $\mathscr{Y}$, where $\mathscr{X}$ corresponds to the set of documents for which we know the correct class, and $\mathscr{Y}$ represents the set of documents whose class values need to be determined. Therefore, the task is to label the nodes $\mathscr{Y}_i \in \mathscr{Y}$ with one of a small number of labels $\mathscr{L} = \{l_1, ..., l_q\}$.

The Waikato Environment for Knowledge Analysis (WEKA) Garner (1995a) provides a Semi-Supervised Learning and Collective Classification plugin[2] with several algorithms.

CollectiveIBk

CollectiveIBk uses internally WEKA's classic IBk algorithm, its implementation of the K-Nearest Neighbour (KNN) algorithm, to determine the best k in the training set. The classifier then builds a neighbourhood consisting of k instances for all instances in the test set from the pool of train and test sets. Either a naïve search of the complete set of instances or a k-dimensional tree is used to determine neighbours. All neighbours in such a neighbourhood are sorted according to their distances from the test instances to which they belong. The neighbourhoods are sorted according to "rank", where "rank" indicates the different occurrences of the two classes (i.e., spam and legitimate) in the neighbourhood.

For every unlabelled test instance ranked highest, class label is determined by majority vote or, in the case of a tie, by the first class. This is performed until no unlabelled test instances remain. Classification terminates by returning the class label of the instance that is about to be classified.

CollectiveForest

CollectiveForest uses WEKA's implementation of RandomTree as a base classifier to divide the test set into folds each of which contain the same number of elements. The first iteration trains using the original training set and generates the distribution for all the instances in the test set. The best instances are then added to the original training set (being the number of instances chosen the same as the number in a fold).

The next iterations train with the new training set and then generate the distributions for the remaining instances in the test set.

[2] http://www.scms.waikato.ac.nz/~fracpete/projects/collective-classification

CollectiveWoods and CollectiveTree

CollectiveWoods works in a similar manner as CollectiveForest but uses CollectiveTree instead of RandomTree.

CollectiveTree is similar to WEKA's original RandomTree classifier. It splits each attribute at the position that divides the current subset of instances (training and test instances) into two halves. The process is finished when one of the following conditions is met:

- Only training instances are covered (the labels for these instances are already known).
- Only test instances remain in the leaf, in which case distribution is taken from the parent node.
- Training instances of only one class remain. In this case, all test instances are considered to belong to this class.

To calculate the class distribution of a complete set or a subset, weights are summed according to the weights in the training set, and then normalised. The nominal attribute distribution corresponds to the normalised sum of weights for each distinct value. For a numeric attribute, the distribution of the binary split is calculated based on the median. The weights are summed for the two bins and finally normalised.

RandomWoods

RandomWoods works like WEKA's classic RandomForest but uses CollectiveBagging (a machine learning ensemble meta-algorithm for improving stability and classification accuracy that has been extended so as to be available to collective classifiers) in combination with CollectiveTree. RandomForest, in contrast, uses Bagging and RandomTree.

4.2.2 Empirical validation

To evaluate the collective algorithms we used the Ling Spam[3], SpamAssassin[4] and TREC 2007 Public Corpus[5] datasets (see Appendix A).

Unfortunately, due to computational restrictions we were obliged to reduce the SpamAssassin and TREC datasets while maintaining the spam-legitimate ratio. We extracted 50% from SpamAssassin, which finally consists of 3,023 e-mails, of which 964 are spam and 2,059 are legitimate messages. For TREC, we

[3] http://nlp.cs.aueb.gr/software_and_datasets/lingspam_public.tar.gz
[4] http://spamassassin.apache.org/publiccorpus/
[5] http://plg.uwaterloo.ca/~gvcormac/spam

randomly extracted 20% of the full subcorpora. As a result, our TREC dataset comprises 5,063 legitimate e-mails and 10,021 junk messages. Table 4.1 presents an overview of the dimensions of the three datasets.

Table 4.1: Comparison of the used datasets in the collective approach. The spam ratio in all three datasets does not follow the statistics of the number of spam messages in the real world which is higher than 85%. The SpamAssassin and TREC datasets, however, contain more realistic spam e-mails and examples of obfuscated mails within them.

Feature	Ling Spam	SpamAssassin	TREC
No. Spam Messages	480	964	10,021
No. of Ham Messages	2,412	2,059	5,063
Spam %.	16.60%	31.36%	66.43%

We also performed a *Stop Word Removal* Wilbur and Sirotkin (1992) for all datasets based on an external stop-word list[6] and removed any non alphanumeric characters.

We then used the VSM, described in Section 2.4.2.2, to create a model. This model represents natural language documents in a mathematical manner through vectors in a multidimensional space.

We constructed an Attribute Relation File Format (ARFF) file Holmes et al. (1994) with the e-mails' resultant vector representations to build the aforementioned WEKA classifiers using the default parameters.

To evaluate the results, we applied the most frequently used measures for spam; these are: precision, recall and Area Under the ROC Curve (AUC) (see Sections 3.2.2 and 3.3.3 for more detailed definitions).

In our experiments we tested various configurations of the collective algorithms with different sizes of the $\mathscr{X}$ set of known instances; the latter varied from 10% to 90% of the instances used for training (i.e., instances known during the test).

It must be noted that, due to unknown issues regarding the implementation of some of the collective algorithms, the testing of TREC dataset with Collective Woods and Random Woods resulted in errors and therefore could not be evaluated.

Figure 4.1 shows the precision of identifying unsolicited e-mail in individual datasets using the different algorithms. Collective KNN shows significant improvements with Ling Spam when the number of known instances increases (from 0.60 with 10% to 0.84 with 90%), but remains constant with SpamAssassin and TREC. When precision was evaluated, Collective Forest was the best

[6] http://www.webconfs.com/stop-words.php

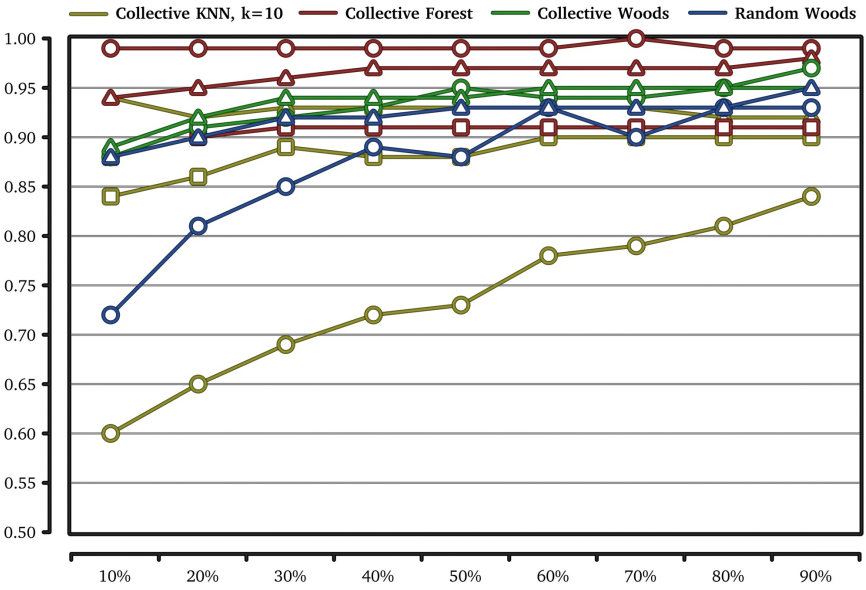


Figure 4.1: Precision of the evaluation of collective algorithms of spam filtering with different sizes for the $\mathscr{X}$ set of known instances. Circle line markers correspond to Ling Spam, triangle line markers correspond to SpamAssassin and square line markers correspond to TREC.

collective algorithm, it achieved a precision of between 0.99 and 1.00 for Ling Spam, of no less than 0.94 for SpamAssassin and of no less than 0.88 for TREC. In testing with Ling Spam and SpamAssassin, Collective Woods and Random Woods showed some improvement when the number of known instances was increased.

Figure 4.2 shows the recall of the various algorithms. Again, Collective KNN shows better results (although still not sufficiently accurate) when the number of known instances increases. With Collective KNN, recall increases from 0.34 with 10% to 0.78 with 90% for Ling Spam, from 0.18 with 10% to 0.84 with 90% for SpamAssassin and from 0.19 to 0.91 for TREC. Collective Forest presents poor recall of 0.77 for 10% with Ling Spam but behaves better with the remaining configurations in all datasets, showing a minimum of 0.89 and a maximum of 0.97 recall for Ling Spam and SpamAssassin and recall between 0.97 and 0.98 for TREC. Collective Woods and Random Woods demonstrate similarly poor recall, achieving maxima with 90% of 0.16 and 0.22, respectively, for Ling Spam and 0.74 and 0.74 for SpamAssassin.

Finally, Figure 4.3 shows the Area Under the ROC Curve (AUC) corresponding to the results obtained with the different algorithms. Once more, the performance of Collective KNN increases with more known instances; the AUC increases from 0.64 with 10% to 0.87 with 90% for Ling Spam, from 0.58 to 0.90

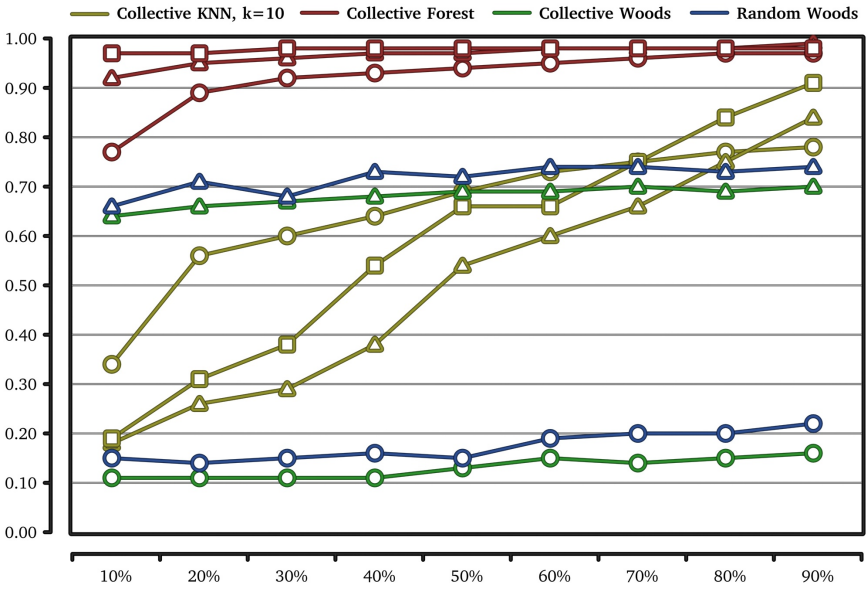


Figure 4.2: Recall of the evaluation of collective algorithms for spam filtering with different sizes of the $\mathcal{X}$ set of known instances. Circle line markers correspond to Ling Spam, triangle line markers correspond to SpamAssassin and square line markers correspond to TREC.

for SpamAssassin and from 0.56 to 0.85 for TREC. Collective Forest offers a perfect 1.00 for every configuration with Ling Spam, a minimum of 0.99 with SpamAssassin and a minimum of 0.95 for TREC, supporting Collective Forest as a suitable choice for the application of collective classification to unsolicited e-mail filtering. Collective Woods and Random Woods achieve similar results, increasing from 0.86 both to 0.92 and 0.91 respectively with the Ling Spam dataset and increasing from 0.93 both to 0.97 and 0.96 respectively with SpamAssassin.

4.2.3 Comparison with supervised approaches

To evaluate the contribution of Collective Classification to spam filtering, we compare the filtering capabilities of our approach with those of commonly used machine-learning algorithms Drucker et al. (1999); Carreras and Márquez (2001); Schneider (2003); Seewald (2007).

To assess the machine-learning classifiers, we used the same datasets as for Collective Classification (i.e., Ling Spam, SpamAssassin and TREC) applying the following methodology:

- **Cross-validation:** To evaluate the performance of machine-learning clas-

sifiers, *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 and apply different parameters or learning algorithms, depending on the concrete classifier. We used three different models:

  - *Bayesian Networks:* To train Bayesian Networks, we used *K2* (Cooper and Herskovits, 1991) and *Tree Augmented Naïve* (TAN) structural learning algorithms.

    We also conducted experiments with *Naïve Bayes* (Lewis, 1998), a classifier that has been used widely for spam filtering Schneider (2003); Seewald (2007).

  - *Decision Trees:* To train decision trees, we used *Random Forest* and *J48* (Weka's *C4.5* implementation).

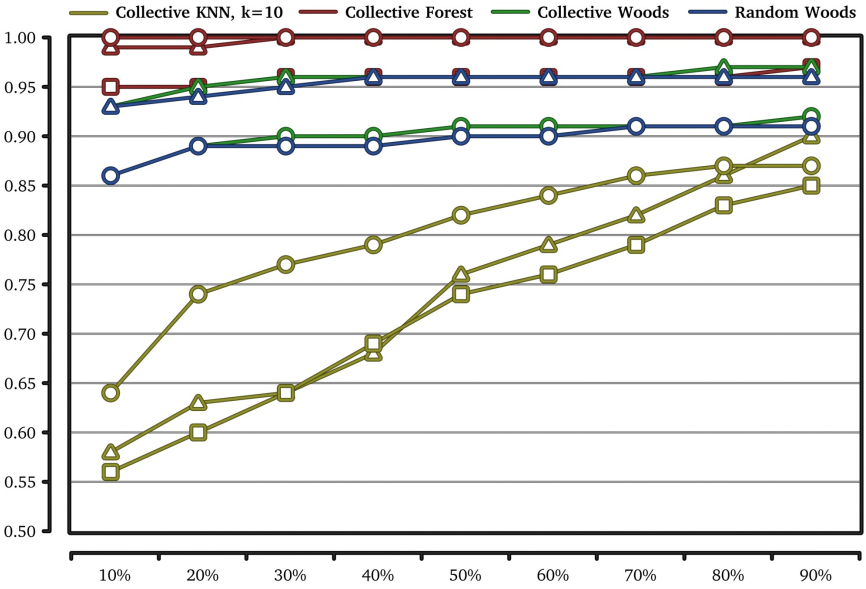


Figure 4.3: Area under the ROC curve (AUC) evaluation of collective algorithms for spam filtering with different sizes of the $\mathcal{X}$ set of known instances. Circle line markers correspond to Ling Spam, triangle line markers correspond to SpamAssassin and square line markers correspond to TREC.

– *Support Vector Machines:* We used a *Sequential Minimal Optimisation* (SMO) algorithm with a *polynomial kernel*, a *normalised polynomial kernel* and a *Radial Basis Function* (RBF) based kernel.

  In addition, we used LibSVM[7] for the linear (i.e., hyperplane) and sigmoid kernel implementation.

Table 4.2 shows a comparison between the results obtained using the best collective algorithm of our approach, Collective Forest, and some of the most commonly used supervised machine-learning algorithms. The results of the comparison show that using only 10% of the labelled data in the training phase, Collective Forest offers sound results with the single drawback of a recall of 0.78 for Ling Spam that could be improved. When 20% of the labelled data is used for training, recall for the Ling Spam dataset is recovered, and the rest of the results improve slightly. Finally, with 50% of the labelled instances, Collective Forest outperforms most of the supervised configurations, while considerably reducing the labelling efforts.

Table 4.2: Comparison of results for the best collective algorithm of our approach, Collective Forest, with results obtained applying commonly used supervised machine-learning algorithms.

	LingSpam			*SpamAssassin*			*TREC*		
Model	**Prec.**	**Rec.**	**AUC**	**Prec.**	**Rec.**	**AUC**	**Prec.**	**Rec.**	**AUC**
BN: K2	0.91	0.98	1.00	0.91	0.89	0.98	1.00	0.40	0.96
BN: TAN	0.92	0.98	1.00	0.94	0.96	0.99	0.90	0.97	0.95
Naïve Bayes	0.97	0.94	1.00	0.83	0.92	0.96	0.92	0.39	0.88
SVM: Polynomial	0.97	0.97	0.98	0.97	0.97	0.98	0.91	0.97	0.90
SVM: Norm Polynom	1.00	0.94	0.97	0.98	0.98	0.99	0.92	0.98	0.90
SVM: RBF	0.99	0.93	0.96	0.98	0.97	0.98	0.81	1.00	0.76
SVM: Lineal	0.97	0.97	0.98	0.97	0.97	0.98	0.91	0.97	0.89
SVM: Sigmoid	1.00	0.47	0.73	0.96	0.89	0.93	0.86	0.99	0.84
DT: J48	0.88	0.84	0.93	0.92	0.93	0.95	0.86	0.98	0.88
DT: RF N=10	0.97	0.95	1.00	0.95	0.98	1.00	0.91	0.98	0.97
Collect. Forest (10%)	0.99	0.77	1.00	0.94	0.92	0.99	0.88	0.97	0.95
Collect. Forest (20%)	0.99	0.89	1.00	0.95	0.95	0.99	0.90	0.97	0.95
Collect. Forest (50%)	0.99	0.94	1.00	0.97	0.97	1.00	0.91	0.98	0.96

4.2.4 **Discussion**

Collective Classification algorithms for spam filtering present a suitable approach to optimising the classification of partially labelled data, thereby overcoming the massive number of unclassified spam that are created every day.

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

Collective Forest, in particular, shows strong results for every configuration of known instances (i.e., different sizes of the $\mathscr{X}$ set of known instances), with precision values above 0.93, recall values above 0.90 (although offering only a poor recall of 0.78 with 10% of $\mathscr{X}$) and approaching 1.00 of AUC for all configurations.

Because precision and AUC are slightly affected by the variations of known instances, recall should be taken into account in determining the optimal size of labelled data, assuming that Collective Forest is the chosen algorithm. Moreover, knowing that a high recall implies a low false positive rate, and that classification costs in spam filtering are asymmetric (i.e., it is worse to block legitimate messages, than to allow spam), reinforces the importance of recall to measure the suitability of the different configurations. In our case, for a value of $\mathscr{X} =$ 50%, Collective Forest achieves the most balanced results between accuracy and amount of labelled data, with only minimal improvements in some configurations when compared to Collective Forest with $\mathscr{X} = 90\%$.

4.3 Anomaly-based spam filtering

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

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

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

Finally, because the amount of spam within all e-mail messages greatly exceeds the number of legitimate e-mails, a question regarding the suitability of choosing legitimate e-mails, instead of spam, as a representation of normality,

may arise. Therefore, we performed a thorough study on the issue, providing comparisons between the two approaches, using both legitimate e-mails and spam as representations of normality.

In summary, our main findings presented in this section include:

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

The remainder of this section is organised as follows. Section 4.3.1 details our anomaly based method. Section 4.3.2 describes the experiments and presents the results of the approach without the dataset reduction. Section 4.3.3 details the application of the dataset reduction step to our anomaly based method. Section 4.3.4 describes the experiments and presents the results of our anomaly based approach enhanced with the dataset reduction, offering a comparison with the approach that does not perform the reducing step. Section 4.3.5 compares the use of legitimate e-mails against spam as representations of normality. Finally, Section 4.3.6 discusses the main shortcomings and outlines avenues for future work.

4.3.1 **Anomaly detection**

Anomaly detection approaches model normality by considering any deviation from this model to be anomalous. Using the word-frequency features of the VSM described in Section 2.4.2.2, our anomaly detection system analyses points in the feature space and classifies e-mails based on their similarity. The analysis of an e-mail consists of 2 different phases:

- Extraction of word-frequency features from the e-mail.
- Measuring the distance from the point representing the e-mail to the points that symbolise normality.

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

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

$$d(x,y) = \sum_{i=0}^{n} |x_i - y_i| \tag{4.1}$$

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

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

$$d(x,y) = \sum_{i=0}^{n} \sqrt{x_i^2 - y_i^2} \tag{4.2}$$

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

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

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

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

4.3.2 Empirical validation of the anomaly detection method

To validate our proposed method, we used the Ling Spam Corpus, SpamAssassin public corpus and TREC 2007 Public Corpus.

However, for our experiments with the TREC dataset, we randomly extracted 30% (due to computational limitations) of the full subcorpora, maintaining the spam-legitimate ratio. Our TREC dataset thus contains 7,653 legitimate e-mails and 14,973 junk messages. Table 4.3 presents an overview of the dimensions of the three datasets.

Table 4.3: Comparison of the used datasets in the anomaly-based approach. The spam ratio in all three datasets does not follow the statistics of the number of spam messages in the real world which is higher than 85%. The SpamAssassin and TREC datasets, however, contain more realistic spam e-mails and examples of obfuscated mails within them.

Feature	Ling Spam	SpamAssassin	TREC
No. Spam Messages	480	1,896	14,973
No. of Ham Messages	2,412	4,150	7,653
Spam %.	16.60%	31.36%	66.18%

We performed a *Stop Word Removal* Wilbur and Sirotkin (1992) on the three datasets based on an external stop-word list[8] and removed every non alpha-numeric characters. We then used the VSM Salton et al. (1975) to create the model. Finally we extracted the top 1,000 attributes using IG .

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

1. **Cross-validation.** For the Ling Spam dataset, we performed a 5-fold cross-validation Kohavi (1995) forming 3 different divisions of 1,930 e-mails and 2 divisions of 1,929 e-mails to represent normality and another 3 divisions of 482 e-mails and 2 of 483 to measure deviations within legitimate e-mail. Each fold thus contains 1,930 or 1,929 legitimate e-mails that represent normality and 963 or 962 testing e-mails, from which 483 or 482 were legitimate e-mails and 480 were spam. The number of legitimate e-mails varied in the last 2 folds because the number of legitimate e-mails was not divisible by 5.

   Regarding the SpamAssassin dataset, we also performed a 5-fold cross-validation to divide the dataset composed of legitimate e-mails (the normal behaviour) into 5 different divisions of 3,320 e-mails to represent normality and 830 to measure deviations within legitimate e-mails. Each fold contains 3,320 legitimate e-mails that represent normality and 2,726 testing e-mails, from which 830 were legitimate e-mails and 1,896 were unsolicited messages.

   Finally, for the TREC dataset, we again performed a 5-fold cross-validation by forming 3 different divisions of 6,122 e-mails and two divisions of 6,123

[8] http://www.webconfs.com/stop-words.php

e-mails to represent normality and another 3 divisions of 1,531 e-mails and 2 of 1,530 to measure deviations within legitimate e-mail. Each fold thus contains 6,122 or 6,123 legitimate e-mails that represent normality and 4525 or 4524 testing e-mails, from which 1,531 or 1,530 were legitimate e-mails and 2,994 were spam. The number of legitimate e-mails varied in the two folds because the number of legitimate e-mails was not divisible by 5 (Table 4.4).

Table 4.4: Number of instances within each fold of the 5-fold cross-validation process in the anomaly-based approach. The number of spam e-mails within Ling Spam and TREC varied in the folds because the number of spam e-mails was not divisible by 5.

Ling Spam

	Normality	Deviations	
	# Legit.	# Legit.	# Spam
Fold 1	1,929	483	480
Fold 2	1,929	483	480
Fold 3	1,930	482	480
Fold 4	1,930	482	480
Fold 5	1,930	482	480

SpamAssassin

	Normality	Deviations	
	# Legit.	# Legit.	# Spam
Fold 1	3,320	830	1,896
Fold 2	3,320	830	1,896
Fold 3	3,320	830	1,896
Fold 4	3,320	830	1,896
Fold 5	3,320	830	1,896

TREC

	Normality	Deviations	
	# Legit.	# Legit.	# Spam
Fold 1	6,122	1,531	2,994
Fold 2	6,122	1,531	2,994
Fold 3	6,122	1,531	2,994
Fold 4	6,123	1,530	2,994
Fold 5	6,123	1,530	2,994

2. **Calculating distances and combination rules.** We extracted the aforementioned word-frequency features from the e-mail and employed the 2 different measures and 3 different combination rules described previously in Section 4.3.1 to obtain a final measure of deviation for each piece of testing evidence. More accurately, we applied the Manhattan and Euclidean distances. For the combination rules, we tested the mean value, lowest distance and highest value.

3. **Defining thresholds.** For each measure and combination rule, we established 10 different thresholds to determine whether an e-mail was spam or not.

   These thresholds were selected by first establishing the lowest one. This number was the highest possible value at which no spam messages were misclassified. The highest one was selected as the lowest possible value at which no legitimate spam messages were misclassified. The remaining thresholds were selected by equally dividing the range between the first and last thresholds.

   The method is thus configurable to reduce both false positives and false negatives. It is important to define whether it is better to classify an unsolicited message as legitimate or legitimate as spam. In particular, one may think that it is more important to detect more spam messages than to minimise false positives. For commercial reasons, one may think just the opposite: a user can be bothered if their legitimate messages are flagged as undesired e-mail.

   To improve these kind of errors, we could apply two different techniques: (i) white and blacklisting or (ii) cost-sensitive learning. As mentioned in Section 2.3, white and blacklists store a signature of an electronic mail to be flagged as either spam (blacklisting) or legitimate messages (whitelisting). Conversely, cost-sensitive learning is a machine-learning technique where one can specify the cost of each error, and the classifiers are trained to account for that consideration Elkan (2001). We could adapt cost-sensitive learning for anomaly detection with cost matrices.

4. **Testing the method.** To evaluate the results, we measured the False Negative Ratio (FNR) and the False Positive Ratio (FPR). FNR is defined as:

$$FNR = \frac{FN}{FN + TP} \tag{4.3}$$

   where TP is the number of spam e-mails correctly classified (true positives) and FN is the number of spam messages misclassified as legitimate (false negatives).

   FPR is defined as:

$$FPR = \frac{FP}{FP + TN} \tag{4.4}$$

   where FP is the number of legitimate e-mails incorrectly classified as spam while TN is the number of legitimate messages correctly classified.

   Moreover, we measured the weighted accuracy (WA), defined as:

$$WA = 1 - \frac{FNR + FPR}{2} \tag{4.5}$$

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

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

$$AUC = \sum_{i=0}^{i=9} (x_{i+1} - x_i) \cdot y_i + \frac{(x_{i+1} - x_i) \cdot (y_{i+1} - y_i)}{2} \tag{4.6}$$

Table 4.5 shows the obtained results for the LingSpam corpus using different distances, combination rules and thresholds. Using this dataset, the best configuration was the one performed with the Euclidean Distance, the Mean combination rule and 2.59319 as the threshold, which provided a 92.27% weighted accuracy, with an FNR of 8.42% and an FPR of 7.05%.

Table 4.6 shows the obtained results for the SpamAssassin corpus for the different distances, combination rules and thresholds. The best results were obtained with the Manhattan Distance, the Min combination rule and a 1.37525 threshold, which provided a 91.63% weighted accuracy, with an FNR of 7.48% and an FPR of 9.25%.

Finally, Table 4.7 shows the obtained results for TREC for different distances, combination rules and thresholds. The best results were obtained with the Euclidean Distance, the Min combination rule and a 0.08532 threshold, which provided a 74.35% weighted accuracy, with an FNR of 42.93% and an FPR of 8.36%.

The best results were obtained with the Min distance, which is the most conservative configuration for distance, highlighting a possible discussion topic regarding what should be called anomaly in e-mails. As stated above, more than 85% of electronic mails today are spam; therefore, in terms of normality, receiving a legitimate e-mail could be considered as an anomalous behaviour. Further study on this topic is presented in Section 4.3.5.

Figures 4.4, 4.5 and 4.6 show different plots for each different distance measure and selection rule.

Table 4.5: Results for different combination rules and distance measures using the Ling Spam corpus in the anomaly-based approach. The abbreviation 'Thres'. stands for the chosen threshold, FPR for False Positive Rate, FNR for False Negative Rate and WA for Weighted Accuracy. The results in bold are the best for each combination rule and distance measure.

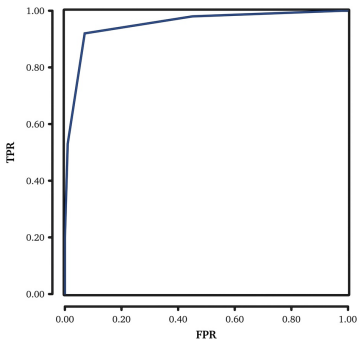
	Manhattan Distance				Euclidean Distance			
Comb.	**Thres.**	**FNR**	**FPR**	**WA**	**Thres.**	**FNR**	**FPR**	**WA**
Mean	1.86313	0.00%	100.00%	50.00%	1.87061	0.00%	99.88%	50.06%
	2.22637	0.21%	99.13%	50.33%	2.11147	0.21%	97.89%	50.95%
	2.58960	2.79%	92.87%	52.17%	2.35233	1.67%	45.23%	76.55%
	2.95284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

Table 4.6: Results for different combination rules and distance measures using the SpamAssassin corpus in the anomaly-based approach. The abbreviation 'Thres'. stands for the chosen threshold, FPR for False Positive Rate, FNR for False Negative Rate and WA for Weighted Accuracy. The results in bold are the best for each combination rule and distance measure.

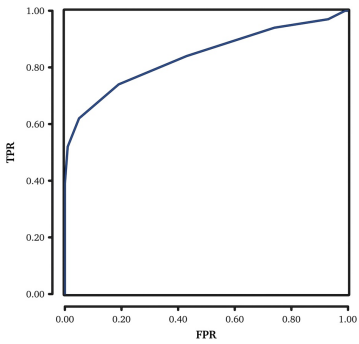
	Manhattan Distance				Euclidean Distance			
Comb.	**Thres.**	**FNR**	**FPR**	**WA**	**Thres.**	**FNR**	**FPR**	**WA**
Mean	1.15978	0.00%	99.98%	50.01%	1.70013	0.00%	99.59%	50.20%
	1.58697	0.14%	95.23%	52.32%	1.91763	2.23%	69.98%	63.90%
	2.01417	1.09%	58.48%	70.22%	**2.13512**	**18.96%**	**25.88%**	**77.58%**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

Table 4.7: Results for different combination rules and distance measures using the TREC corpus in the anomaly-based approach. The abbreviation 'Thres'. stands for the chosen threshold, FPR for False Positive Rate, FNR for False Negative Rate and WA for Weighted Accuracy. The results in bold are the best for each combination rule and distance measure.

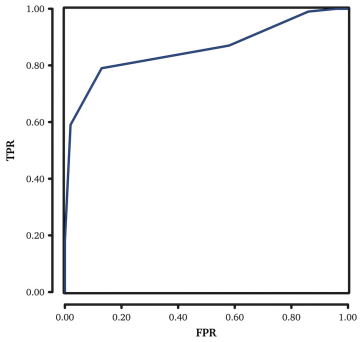
	Manhattan Distance				Euclidean Distance			
Comb.	**Thres.**	**FNR**	**FPR**	**WA**	**Thres.**	**FNR**	**FPR**	**WA**
Mean	0.46807	0.00%	100.00%	50.00%	1.18474	0.00%	100.00%	50.00%
	0.61292	33.13%	64.89%	50.99%	1.42634	27.07%	71.10%	50.92%
	0.75777	46.24%	46.66%	53.55%	1.66794	64.56%	30.15%	52.65%
	0.90263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



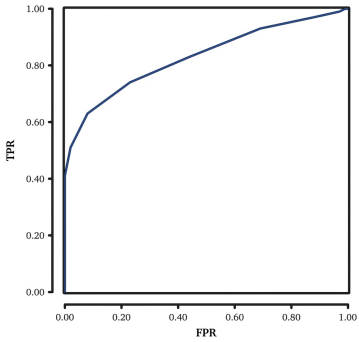
(a) ROC curve for the Euclidean distance and Mean selector.



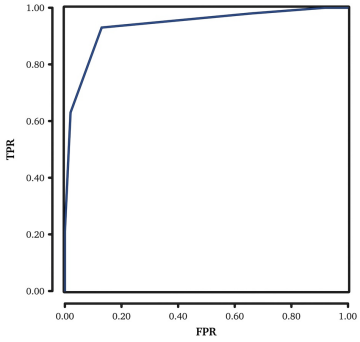
(b) ROC curve for the Manhattan distance and Mean selector.



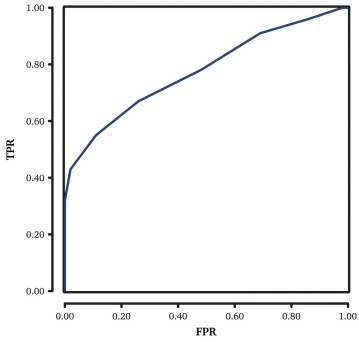
(c) ROC curve for the Euclidean distance and Max selector.



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

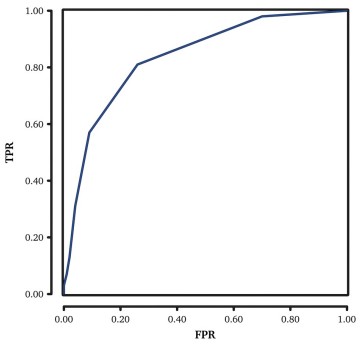


(e) ROC curve for the Euclidean distance and Min selector.

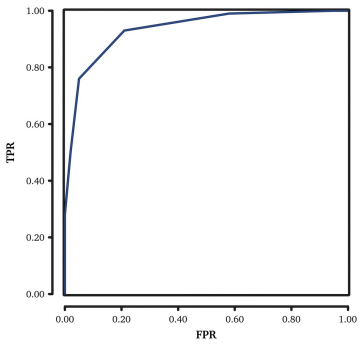


(f) ROC curve for the Manhattan distance and Min selector.

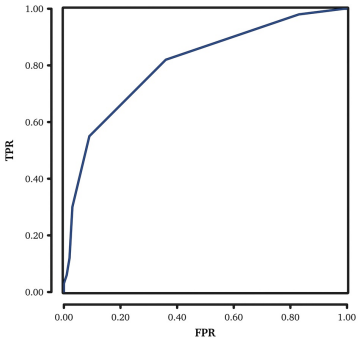
Figure 4.4: ROC curves for the different experimental configurations applied to LingSpam in the anomaly-based approach.



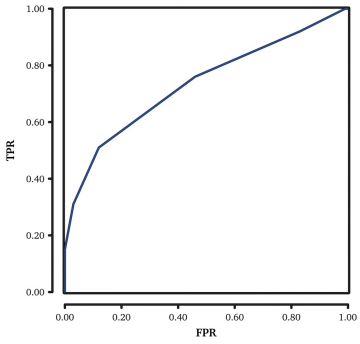
(a) ROC curve for the Euclidean distance and Mean selector.



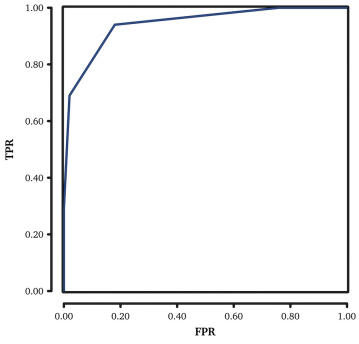
(b) ROC curve for the Manhattan distance and Mean selector.



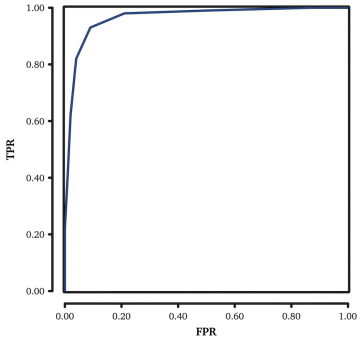
(c) ROC curve for the Euclidean distance and Max selector.



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

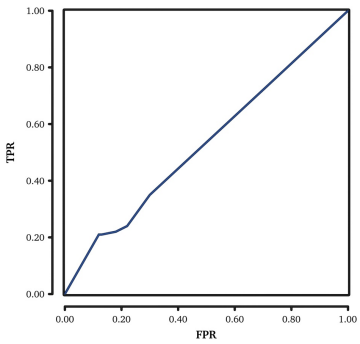


(e) ROC curve for the Euclidean distance and Min selector.

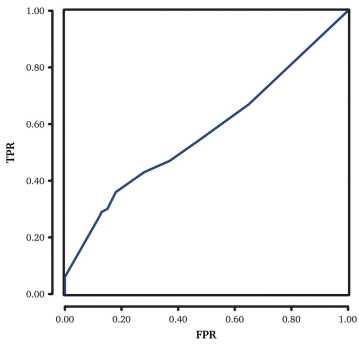


(f) ROC curve for the Manhattan distance and Min selector.

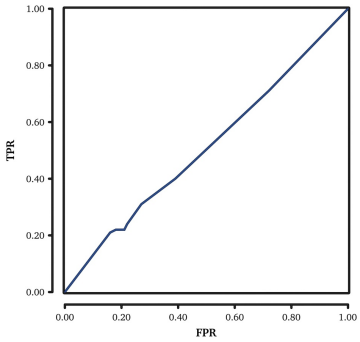
Figure 4.5: ROC curves for the different experimental configurations applied to SpamAssassin in the anomaly-based approach.



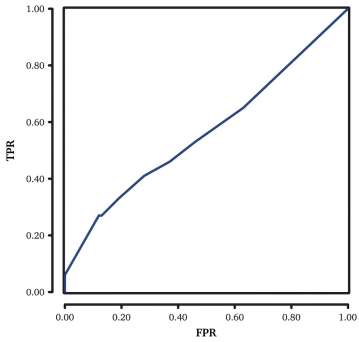
(a) ROC curve for the Euclidean distance and Mean selector.



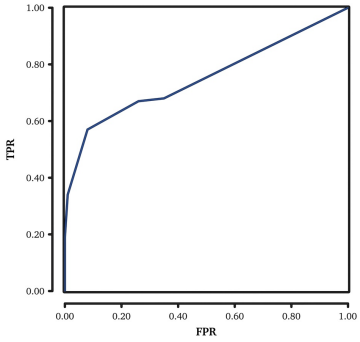
(b) ROC curve for the Manhattan distance and Mean selector.



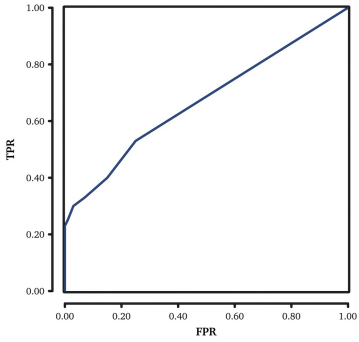
(c) ROC curve for the Euclidean distance and Max selector.



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



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



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

Figure 4.6: ROC curves for the different experimental configurations applied to TREC in the anomaly-based approach.

4.3.3 Improving the efficiency with dataset reduction

Dataset reduction is a step that must be faced in different problems when working with large datasets. In the approach presented in Section 4.3.1 that uses non-reduced datasets, the experiments were performed with a base of over 2,000 (for the LingSpam dataset), over 4,000 (for the SpamAssassin dataset) and over 14,000 (for the TREC dataset) legitimate e-mails, which means that every sample analysed had to be compared 2,000, 4,000 or 14,000 times to classify it as spam or not. Therefore, we propose a data reduction algorithm based on partitional clustering.

Cluster analysis divides data into meaningful groups Kumar (2000). These techniques usually employ distance measures to compare instances in datasets to group those that appear to be similar. We can identify several types of clustering, but the most common are hierarchical and partitional clustering.

The first approach generates clusters in a nested style, which means that the dataset is divided into a set of clusters that are subdivided into other clusters related hierarchically. Conversely, partitional clustering techniques create a one-level (unnested) partitioning of the data points Kumar (2000). We are interested in this last technique to validate our initial hypothesis because it makes it possible to divide a large set of e-mails that represent normality (i.e., legitimate e-mails) into a reduced set of representations.

Heyer et al. (1999) proposed the QT clustering algorithm to extract useful information from large amounts of gene expression data (Figure 4.7). This clustering algorithm need not specify the number of clusters desired. It uses a similarity threshold value to determine the maximum radial distance of any cluster. It thus generates a variable number of clusters that meet a quality threshold. Its main disadvantage is the high number of distance calculations needed. Nevertheless, this computational overhead is admissible in this case, as we must only reduce the dataset once (we employ a static representation of normality that remains invariable).

Our algorithm, shown in Figure 4.8, is based on the concepts proposed by Heyer et al. (1999); it is adapted to our data reduction problem and implemented iteratively, instead of recursively.

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

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

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

Figure 4.7: Quality Threshold algorithm.

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

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

4.3.4 Empirical validation of the method with the normality dataset reduction

To evaluate the performance of our method, we conducted an experiment with two phases: first, we reduced the set of vectors corresponding to the representation of the legitimate e-mails that represent normality, and second, we started the anomaly detection (i.e., spam detection) step to measure both accuracy and efficiency.

input : The original dataset $\mathscr{V}$, the distance threshold for each cluster $threshold$ and the minimum number of vectors in each cluster $minimumvectors$
output: The reduced dataset $\mathscr{R}$

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

Figure 4.8: Dataset reduction algorithm based on QT Clustering.

Again, we used the Ling Spam, SpamAssassin and TREC 2007 datasets. To evaluate the performance of the predictors, we conducted the same process as for the approach without the reduction step that can be reviewed in Section 4.3.2, including: a k-fold cross-validation Kohavi (1995), stop word removal, representation with VSM, feature selection with IG , distances and combination rules calculation and threshold defining.

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

Table 4.8 shows the reductions obtained in the process. The result obtained for the infinity threshold is a unique centroid of the whole dataset that represents the arithmetic mean vector or a single representation of normality. In this case, selection rules did not influence the final result because the method only performed one single comparison for each sample.

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

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

  The average processing time consumed to reduce the datasets (Figure 4.9) for each configuration is 1,107 seconds for LingSpam, 3,302 seconds for SpamAssassin and 15,235 seconds for TREC when using Euclidean distance and 751 seconds for LingSpam, 2,179 seconds for SpamAssassin and 10,236 seconds for TREC when using Manhattan distance. This process, despite being time consuming, is executed only once and does not interfere with system performance.

  The times do not vary considerably among the different thresholds used for each distance measure because the operations that take a higher processing overhead are the distance measure calculations, and the algorithm proposed in Figure 4.8 calculates all distances between points before starting the clustering step. Consequently, the data reduction algorithm performs the same heavy calculations independently of the previously selected threshold.

- **Sample comparison.** In this phase, for each experimental configuration employed in the data reduction stage, the samples were compared to the reduced dataset. The number of comparisons depends exclusively on the

Table 4.8: Number of vectors conforming the reduced datasets for the different reduction thresholds of the anomaly-based approach with clustering.

LingSpam

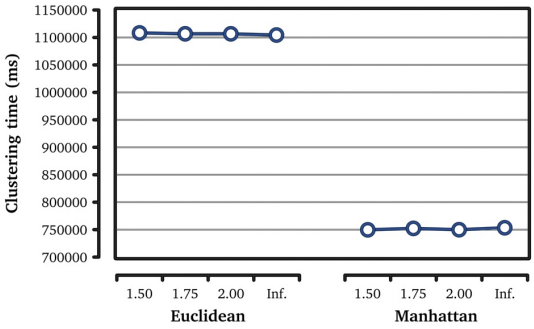
Distance measure	Quality threshold	% Average reduction	Vectors per fold				
			1	2	3	4	5
Euclidean	1.50	13.21%	1,647	1,646	1,674	1,688	1,718
	1.75	57.10%	800	802	817	848	871
	2.00	89.72%	184	184	191	212	220
	∞	99.94%	1	1	1	1	1
Manhattan	1.50	33.75%	1,318	1,322	1,296	1,223	1,232
	1.75	46.78%	1,079	1,047	1,051	979	978
	2.00	62.47%	769	749	750	673	679
	∞	99.94%	1	1	1	1	1

SpamAssassin

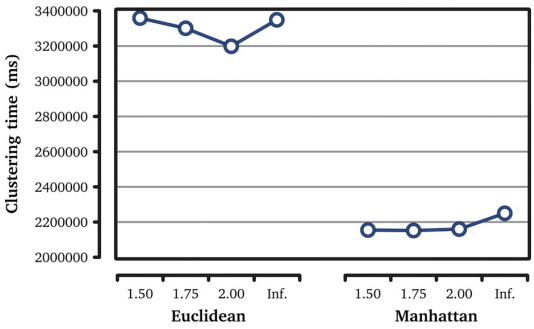
Distance measure	Quality threshold	% Average reduction	Vectors per fold				
			1	2	3	4	5
Euclidean	1.50	89.78%	302	342	431	297	324
	1.75	97.63%	66	79	102	66	79
	2.00	99.34%	16	18	33	20	21
	∞	99.96%	1	1	1	1	1
Manhattan	1.50	93.59%	119	230	251	221	242
	1.75	96.81%	50	117	132	109	121
	2.00	98.57%	17	53	60	52	54
	∞	99.96%	1	1	1	1	1

TREC

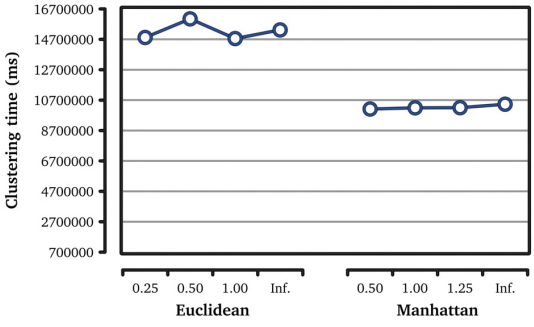
Distance measure	Quality threshold	% Average reduction	Vectors per fold				
			1	2	3	4	5
Euclidean	0.25	56.16%	2,673	2,671	2,663	2,685	2,729
	0.50	74.81%	1,531	1,504	1,528	1,577	1,570
	1.00	98.28%	108	104	105	108	103
	∞	99.98%	1	1	1	1	1
Manhattan	0.50	98.06%	125	124	117	107	121
	1.00	99.95%	3	3	3	3	3
	1.25	99.94%	3	3	4	3	4
	∞	99.98%	1	1	1	1	1



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



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



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

Figure 4.9: Dataset reduction times of the anomaly-based approach with clustering. The X axis shows the different experimental configurations selected for the data reduction step. The Y axis shows the time required by each clustering process performed, expressed in milliseconds.

number of vectors present in the resulting datasets, so the time employed in this step is inversely proportional to the threshold value used in the clustering algorithm.

Figure 4.10 shows the average time employed by the comparison step for each testing sample. As expected, the time required for comparison is lower when utilising fewer vectors.

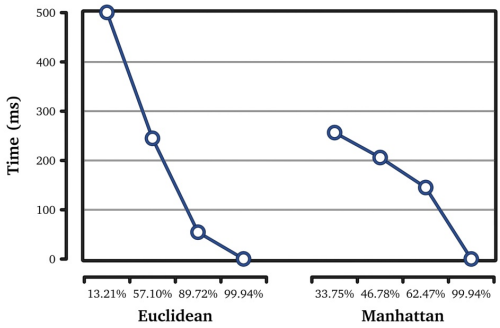
For Euclidean distance the average comparison time varies from 494.53 ms for a 1.50 clustering threshold value, to 0.46 ms for an ∞ threshold (comparison against a single vector representation) with LingSpam, from 121.07 ms for a 1.50 threshold to 0.35 for an ∞ threshold with SpamAssassin and from 1,063.56 ms for a 0.25 threshold to 0.41 for an ∞ threshold with TREC. In Manhattan distance, times are lower due to the simplicity of the calculations needed, varying from 257.55 ms, 56.22 ms and 32.11 ms to 0.30 ms, 0.24 and 0.34 with LingSpam, SpamAssassin and TREC respectively. Compared to Ling Spam, SpamAssassin and TREC present lower comparison times, despite their larger sizes, because of the increased reduction suffered (as in Table 4.8).

Hereafter, we obtained the representation of the messages from all three datasets, reduced the dataset using the two different distance measures and four different threshold values and employed the same two different measures and the three combination rules described in Section 4.3.1 to test the datasets and obtain a final measure of deviation for each testing sample.

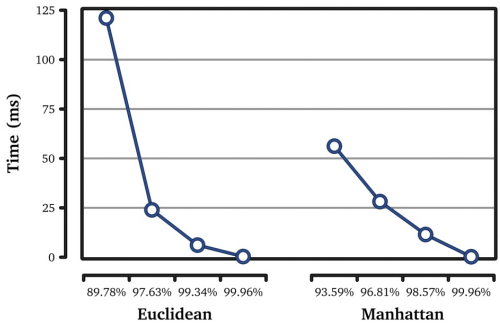
For each measure and combination rule, we established 10 different thresholds to determine whether an e-mail was spam and selected the one that offered the best results in each case. We evaluated the results by measuring the FNR, FPR, WA and AUC.

FNR is defined as $FNR = FN/FN + TP$, where TP is the number of spam e-mails correctly classified (true positives) and FN is the number of spam messages misclassified as legitimate (false negatives). FPR is $FPR = FP/FP + TN$, where FP is the number of legitimate e-mails incorrectly classified as spam while TN is the number of legitimate messages correctly classified. WA is defined as $WA = 1-(FNR+FPR/2)$. Finally, AUC is the area under the curve formed by the union of the points representing the FPR and TPR for each possible threshold in a plot where the X axis represents the FPR and the Y axis represents the TPR.

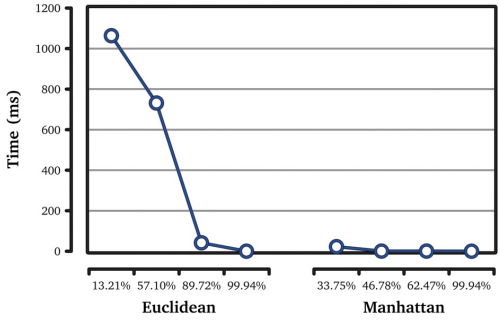
Tables 4.9, 4.10 and 4.11 show the obtained results. To simplify the empirical evaluation presented, we only show the performance associated with the best threshold for each configuration. Our anomaly-based unsolicited e-mail filtering system is able to correctly detect over 92% of junk electronic mails while maintaining a rate of misclassified legitimate messages below 6% with the best configuration tested with Ling Spam (Euclidean distance, 1.75 threshold and Mean rule). For SpamAssassin, the WA approaches 90% with fewer than 7% misclassified legitimate e-mails (Manhattan distance, 2.00 threshold and Min rule).



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



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



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

Figure 4.10: Comparison times for the anomaly-based approach with dataset reduction. The X axis represents the resulting reduction rate for each dataset after applying the clustering step. The bigger the reduction rate, the lower the number of vectors utilised. The Y axis represents the average comparison time for each e-mail, expressed in milliseconds.

Finally, for the TREC dataset, we obtain the worst results with only a 74.82% WA and an improvable 9.19% misclassified legitimate e-mails (Euclidean distance, 1.75 threshold and Min rule).

The Min combination rule achieved the best results for SpamAssassin and TREC, while the Mean combination rule was the best for LingSpam (though this can be discussed, as the Min combination rule produces fewer misclassified legitimate e-mails while maintaining a similar WA). To provide impartial results, we present as best results those that received the higher weighted accuracy, but, as stated before, for commercial purposes, using configurations offering lower FNR or FPR (depending on the desired goals) is recommended.

Figures 4.11, 4.12 and 4.13 show different plots for each different distance measure and selection rule. Each plot offers 4 ROC curves corresponding to the 6 different reduced versions of each datasets. In some cases, the ROC curve shows better results when the threshold employed for reduction is ∞ (thus, the number of vectors to compare is only 1).

Figures 4.14(a), 4.14(b) and 4.14(c) represent the evolution of the configurations for the different reduction rates. Regarding Min and Max selection rules, as the number of vectors diminishes, the system loses accuracy for SpamAssassin and TREC but maintains it for LingSpam. Nevertheless, when the samples were compared to the mean vector, the results improve for all datasets.

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

Table 4.9: Results for the different reduced datasets of LingSpam, combination rules and distance measures.

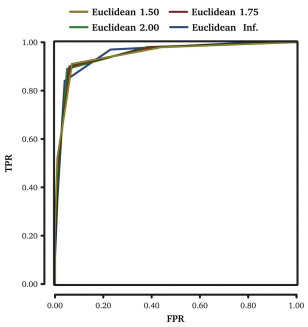
Ling Spam						
Distance	Quality Threshold	**Selection rule**	**Threshold**	**FNR**	**FPR**	**WA**
Euclidean	Without reduction	Mean	2.59319	8.42%	7.05%	92.27%
		Max	4.15651	20.71%	12.89%	83.20%
		Min	1.93707	6.88%	13.23%	89.95%
	1.50	Mean	2.61855	9.04%	6.80%	92.08%
		Max	4.15651	20.71%	12.89%	83.20%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

Table 4.10: Results for the different reduced datasets of SpamAssassin, combination rules and distance measures.

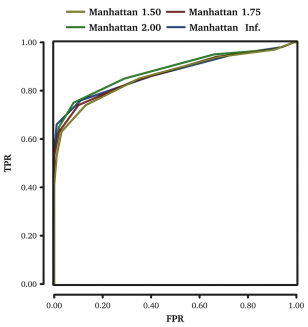
SpamAssassin						
Distance	Quality Threshold	**Selection rule**	**Threshold**	**FNR**	**FPR**	**WA**
Euclidean	Without reduction	Mean	2.13512	18.96%	25.88%	77.58%
		Max	3.83970	44.58%	8.70%	73.36%
		Min	1.39962	6.00%	18.41%	87.79%
	1.50	Mean	2.27873	22.34%	22.60%	77.53%
		Max	3.66095	34.07%	20.48%	72.72%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

Table 4.11: Results for the different reduced datasets of TREC, combination rules and distance measures.

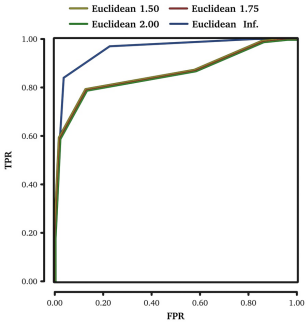
TREC						
Distance	Quality Threshold	**Selection rule**	**Threshold**	**FNR**	**FPR**	**WA**
Euclidean	Without reduction	Mean	2.63433	78.66%	12.41%	54.46%
		Max	4.19129	78.60%	15.71%	52.85%
		Min	0.53782	42.93%	8.36%	74.35%
	1.50	Mean	2.47451	78.66%	12.41%	54.46%
		Max	4.19129	78.60%	15.71%	52.85%
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



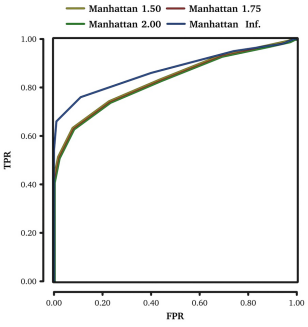
(a) ROC curve for the Euclidean distance and Mean selector.



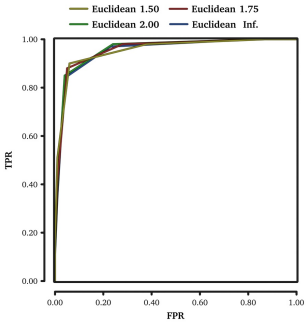
(b) ROC curve for the Manhattan distance and Mean selector.



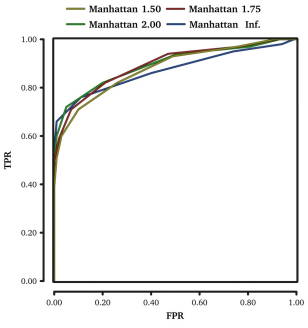
(c) ROC curve for the Euclidean distance and Max selector.



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

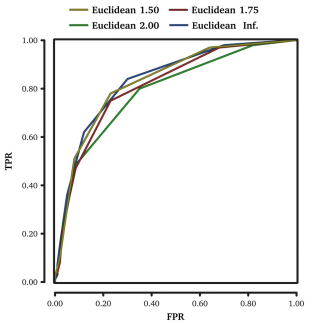


(e) ROC curve for the Euclidean distance and Min selector.

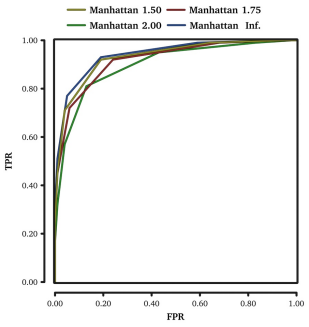


(f) ROC curve for the Manhattan distance and Min selector.

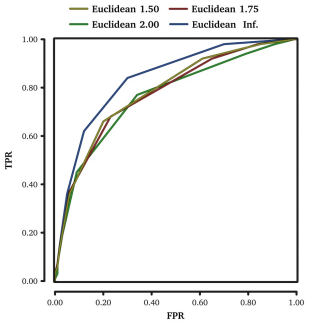
Figure 4.11: ROC curves for the different experimental configurations applied to LingSpam in the anomaly-based approach with clustering. Each figure shows 4 ROC curves corresponding to the different reduced datasets.



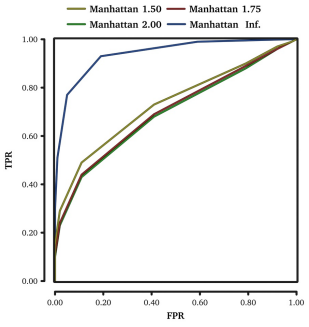
(a) ROC curve for the Euclidean distance and Mean selector.



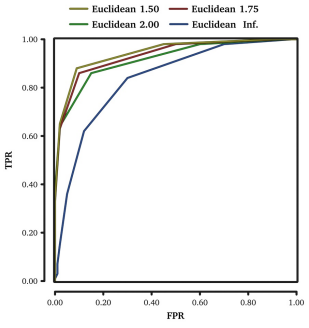
(b) ROC curve for the Manhattan distance and Mean selector.



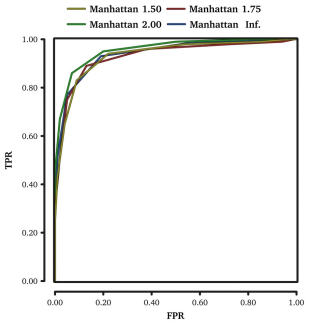
(c) ROC curve for the Euclidean distance and Max selector.



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

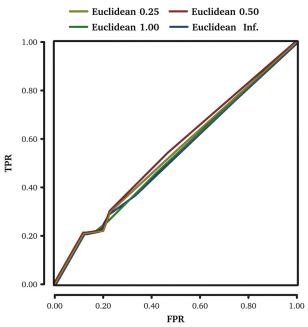


(e) ROC curve for the Euclidean distance and Min selector.

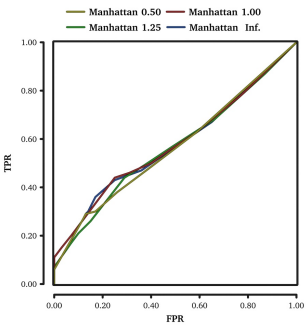


(f) ROC curve for the Manhattan distance and Min selector.

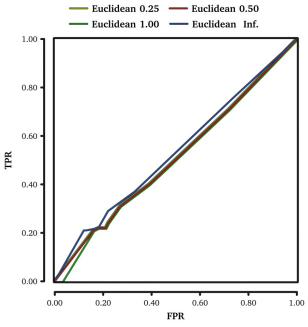
Figure 4.12: ROC curves for the different experimental configurations applied to SpamAssassin in the anomaly-based approach with clustering. Each figure shows 4 ROC curves corresponding to the different reduced datasets.



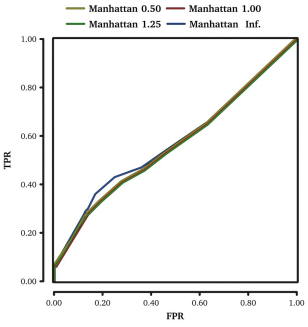
(a) ROC curve for the Euclidean distance and Mean selector.



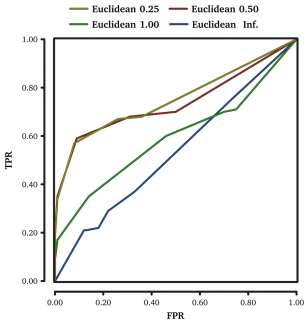
(b) ROC curve for the Manhattan distance and Mean selector.



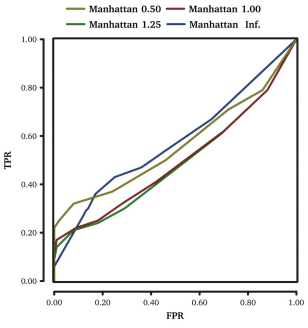
(c) ROC curve for the Euclidean distance and Max selector.



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

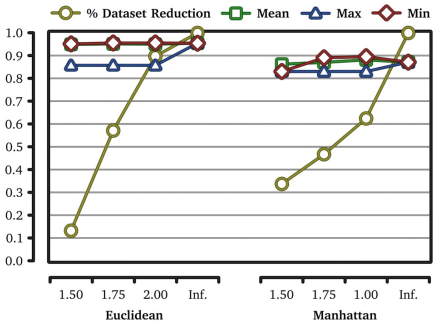


(e) ROC curve for the Euclidean distance and Min selector.

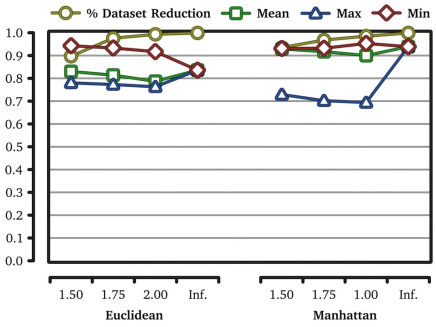


(f) ROC curve for the Manhattan distance and Min selector.

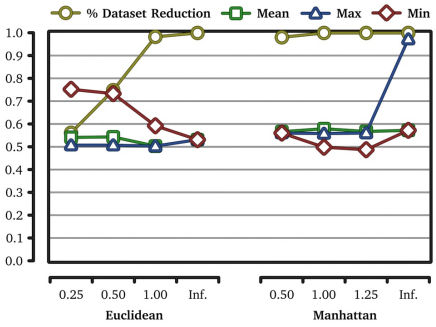
Figure 4.13: ROC curves for the different experimental configurations applied to TREC in the anomaly-based approach with clustering. Each figure shows 4 ROC curves corresponding to the different reduced datasets.



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



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



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

Figure 4.14: Relation between dataset reduction rate and accuracy in the anomaly-based approach with clustering. The Dataset reduction line represents the increasing reduction rate (the higher the rate, the lower the number of samples in the reduced dataset), while the other lines represent the area under the ROC curve (AUC) obtained with each reduced dataset.

4.3.5 Representation of normality: legitimate vs. spam

To evaluate the suitability of choosing spam as anomaly, we performed several experiments using spam to represent normality to compare against the results obtained with the approach that represents normality using legitimate messages. The experiments were conducted following the methods described in Sections 4.3.1 and 4.3.3, only this time, spam was the normal behaviour and legitimate e-mails were the anomaly.

Again, we used the Ling Spam, SpamAssassin and TREC 2007 Public Corpus datasets and followed the same configuration detailed in Section 4.3.2.

For Ling Spam, we performed a 5-fold cross-validation by dividing the dataset of spam e-mails (the normal behaviour) into 5 different divisions of 96 messages, using 4 to represent normality and 1 to measure deviations. Each fold thus contains 384 spam e-mails that represent normality and 2,508 testing e-mails, from which 96 were spam and 2,412 were legitimate e-mails. For SpamAssassin, we also performed a 5-fold cross-validation. Each fold contains 1,517 or 1,516 spam e-mails that represent normality and 4,530 or 4,529 testing e-mails, from which

Table 4.12: Number of instances within each fold of the 5-fold cross-validation process when legitimate e-mails are considered as the anomaly. The number of spam e-mails within SpamAssassin and TREC varied in the folds because the number of spam e-mails was not divisible by 5.

Ling Spam

	Normality	Deviations	
	# Spam	# Spam	# Legit.
Fold 1	384	96	2,412
Fold 2	384	96	2,412
Fold 3	384	96	2,412
Fold 4	384	96	2,412
Fold 5	384	96	2,412

SpamAssassin

	Normality	Deviations	
	# Spam	# Spam	# Legit.
Fold 1	1,516	380	4,150
Fold 2	1,517	379	4,150
Fold 3	1,517	379	4,150
Fold 4	1,517	379	4,150
Fold 5	1,517	379	4,150

TREC

	Normality	Deviations	
	# Spam	# Spam	# Legit.
Fold 1	11,979	2,994	7,653
Fold 2	11,979	2,994	7,653
Fold 3	11,978	2,995	7,653
Fold 4	11,978	2,995	7,653
Fold 5	11,978	2,995	7,653

380 or 379 were spam e-mails and 4,150 were legitimate e-mails. Finally, for TREC, the 5-fold cross-validation divided spam e-mails into 3 different divisions of 11,978 e-mails and 2 divisions of 11,979 e-mails to represent normality. Each fold has 11,979 or 11,978 spam e-mails that represent normality and 10,647 or 10,648 testing e-mails, from which 2,994 or 2,995 were spam e-mails and 7,653 were legitimate e-mails (Table 4.12). For more information on how to test the method refer to Section 4.3.2.

4.3.5.1 **Results of the anomaly-based approach without the dataset reduction and spam as normality**

Table 4.13 shows the best results obtained for each testing dataset using different distances, combination rules and thresholds. The best results were offered for all three datasets by the Euclidean Distance with the Min combination rule: 84.58% FNR, 0.43% FPR and 57.49% WA for Ling Spam; 59.49% FNR, 1.47% FPR and 69.52% WA for SpamAssassin; and 10.53% FNR, 32.01% FPR and 78.73% WA for TREC.

Table 4.13: Best results obtained for different combination rules and distance measures with our anomaly-based approach without the clustering step when using spam to represent normality, thus considering legitimate e-mails as anomalous behaviour. 'Thres.' stands for the chosen threshold.

Ling Spam

	Manhattan Distance				Euclidean Distance			
Comb.	**Thres.**	**FNR**	**FPR**	**WA**	**Thres.**	**FNR**	**FPR**	**WA**
Mean	2.96003	100.00%	0.00%	50.00%	2.34311	100.00%	0.00%	50.00%
Max	9.04422	32.08%	65.15%	51.38%	4.72080	64.38%	25.68%	54.97%
Min	0.70134	87.50%	2.52%	54.99%	1.28991	84.58%	0.43%	57.49%

SpamAssassin

	Manhattan Distance				Euclidean Distance			
Comb.	**Thres.**	**FNR**	**FPR**	**WA**	**Thres.**	**FNR**	**FPR**	**WA**
Mean	2.44163	100.00%	0.00%	50.00%	1.97276	100.00%	0.00%	50.00%
Max	4.91811	100.00%	0.00%	50.00%	4.43060	43.41%	19.75%	68.42%
Min	0.40784	76.90%	1.60%	60.75%	1.27859	59.49%	1.47%	69.52%

TREC

	Manhattan Distance				Euclidean Distance			
Comb.	**Thres.**	**FNR**	**FPR**	**WA**	**Thres.**	**FNR**	**FPR**	**WA**
Mean	0.69536	100.00%	0.00%	50.00%	1.73610	44.18%	52.15%	51.46%
Max	2.69006	99.93%	0.00%	50.04%	3.77008	66.15%	32.59%	50.63%
Min	0.00000	29.45%	20.99%	74.78%	0.43639	10.53%	32.01%	78.73%

To provide impartial results, we present as best results those with higher weighted accuracy values, but for commercial purposes, using configurations offering lower FNR or FPR (depending on the desired goals) is recommended.

Table 4.14: Best results obtained for different combination rules and distance measures with our anomaly-based approach applying the dataset reduction step when using spam to represent normality, hence considering legitimate e-mails as an anomalous behaviour. 'Thres.' stands for the chosen threshold.

Ling Spam

Distance	Quality Threshold	Selection rule	Threshold	FNR	FPR	WA
Euclidean	Without reduction	Min	1.28991	84.58%	0.43%	57.49%
	0.25	Min	1.28991	84.58%	0.43%	57.49%
	1.00	Min	1.28991	84.58%	0.43%	57.49%
	2.00	Min	1.44940	83.33%	2.84%	56.91%
	∞	Min	1.28257	99.79%	0.20%	50.01%
Manhattan	Without reduction	Min	0.70134	87.50%	2.52%	54.99%
	0.25	Min	0.70134	87.50%	2.52%	54.99%
	1.00	Min	1.69034	71.46%	12.91%	57.82%
	2.00	Min	2.21186	71.04%	8.70%	60.13%
	∞	Min	2.96003	100.00%	0.00%	50.00%

SpamAssassin

Distance	Quality Threshold	Selection rule	Threshold	FNR	FPR	WA
Euclidean	Without reduction	Min	1.27859	59.49%	1.47%	69.52%
	0.25	Min	1.27859	59.49%	1.47%	69.52%
	1.00	Min	1.27859	59.55%	1.81%	69.32%
	2.00	Min	1.07816	86.71%	2.33%	55.48%
	∞	Min	0.83543	100.00%	0.00%	50.00%
Manhattan	Without reduction	Min	0.40784	76.90%	1.60%	60.75%
	0.25	Min	0.40784	77.37%	1.60%	60.51%
	1.00	Min	0.82545	69.41%	1.46%	64.57%
	2.00	Min	2.06302	37.29%	22.69%	70.01%
	∞	Min	0.69476	100.00%	0.00%	50.00%

TREC

Distance	Quality Threshold	Selection rule	Threshold	FNR	FPR	WA
Euclidean	Without reduction	Min	0.43639	10.53%	32.01%	78.73%
	0.25	Min	0.46411	8.86%	33.18%	78.98%
	0.50	Min	0.46416	9.60%	30.69%	79.85%
	0.75	Min	0.69692	4.05%	51.70%	72.12%
	∞	Min	1.03371	68.18%	27.30%	52.26%
Manhattan	Without reduction	Min	0.00000	29.45%	20.99%	74.78%
	0.25	Min	0.31199	3.41%	61.78%	67.40%
	0.50	Min	0.49392	2.83%	68.73%	64.22%
	0.75	Min	0.70688	2.53%	76.74%	60.37%
	∞	Min	0.69476	100.00%	0.00%	50.00%

4.3.5.2 **Results of the anomaly-based approach with the dataset reduction and spam as normality**

Table 4.14 shows the best results obtained for each testing dataset (i.e., LingSpam, SpamAssassin and TREC). Only configurations with the Min selection rule are shown, as they outperformed both the Mean and Max rules in all cases.

In the table, we also include the best results of the approach without the clustering step for comparison purposes. These results confirm that using our proposed dataset reduction does not diminish detecting capabilities but, in fact,

could increase them (at least one configuration for all three datasets outperforms the results obtained without reduction). The best results were obtained with Manhattan distance for LingSpam and SpamAssassin and with Euclidean distance for TREC: 71.04% FNR, 8.70% FPR and 60.13% WA for LingSpam; 37.29% FNR, 22.69% FPR and 70.01% WA for SpamAssassin; and 9.60% FNR, 30.69% FPR and 79.85% WA for TREC.

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

4.3.5.3 **Comparison of the approaches**

Table 4.15 compares the best results obtained for each approach, considering spam or legitimate e-mails as anomaly, and the different tested configurations. In summary, the Min selection rule always provided the best results, Euclidean distance is the best in our experiments, and finally, regarding the suitability of using legitimate e-mails or spam to represent normality, we obtain different readings. On one hand, for the LingSpam and SpamAssassin datasets, the best approach uses legitimate e-mails to represent normality, thus considering spam as anomaly. On the other hand, for the TREC dataset, we obtain better results when considering legitimate e-mails as anomaly. This is a consequence of the different types of e-mails contained within each dataset, with TREC the most current, complete and heterogeneous dataset of the three, clearly showing the importance of the nature of the normality dataset.

4.3.6 **Discussion**

The final results show that the use of anomaly detection techniques for spam filtering achieves high accuracy levels, minimising the labelling efforts with a dataset of only one class of e-mails: legitimate or spam. Nevertheless, several discussion points arise regarding the suitability of the proposed method.

The VSM on which this method relies assumes that every term is independent, which is, at least from a linguistic point of view, not completely true. Though 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). Therefore, our representation cannot handle the existing linguistic phenomena in natural languages Becker and Kuropka (2003). To address this problem, we presented in Section 3.2 an approach capable of detecting the internal semantics of spam messages.

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

Table 4.15: Best results for each approach, considering spam or legitimate e-mails as anomaly, and the different tested configurations.

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

evade filters. As happens with our previously presented approaches, Good Word Attack or tokenisation are some methods that would allow spammers to bypass the filter.

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

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

4.4 Concluding remarks

As the number of unsolicited bulk messages increases, the classification and labelling steps used by common supervised methods become increasingly unattainable. To reverse this situation, we propose the first spam filtering system that uses collective classification to optimise classification performance. The algorithms in-

troduced minimise the necessity of using labelled e-mails by 50% without compromising detection capability.

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

According to the limitations found in our proposed methods, we identified several open lines for future work that we next summarise:

- An extensive analysis on the selection of the Quality Thresholds and the selection rule's for each thresholds for the anomaly-based approach.
- In real world environments, legitimate e-mails could be considered the anomaly. Further research on this assumption should be performed with other recent datasets.
- Unsupervised methods have proven their benefits in spam filtering and similar domains. Therefore, the study of recent unsupervised techniques (Cormack, 2007a) in order to apply them to the unsolicited e-mail filtering problem should be taken into account.

4.5 Summary

This chapter has introduced the labelling problem that most anti-spam systems suffer. We have presented two different approaches to face this problem: collective algorithms and anomaly detection.

Firstly, we have described how to adapt collective classification to spam filtering, a semi-supervised text categorisation technique that reduces the labelling efforts. Secondly, we have proposed an anomaly-based filtering approach that is able to detect unsolicited e-mails by training the models with only one class, i.e., spam or legitimate messages.

In summary, our results show that both approaches are able to maintain the filtering capabilities while considerably reducing the number of labelled instances needed to train the models. In this way, these approaches may be considered to diminish the manpower used for labelling on which current models depend.

"All truths are easy to understand once they are discovered; the point is to discover them."

Galileo Galilei (1564–1642)

5

Conclusions

ALONG this work we have described in detail and empirically evaluated the main contributions of the present dissertation. Now is time to measure the level of accomplishment that this work has achieved according to the established fundamental hypothesis and goals. The remainder of this chapter is organised as follows. Section 5.1 revisits the fundamental hypothesis and reviews the major contributions to the area of spam filtering. Section 5.2 discusses the main shortcomings of the proposed approaches. Section 5.3 presents open research lines and challenges for future reference. Finally, Section 5.4 concludes this dissertation.

5.1 Main contributions

After presenting our proposed approaches and the evaluation results, it is now time to revisit once again our initial hypothesis:

> «*It is possible to improve spam filtering systems with semantic-aware techniques, able to overcome linguistic phenomena, and reduce the time-consuming task of sample analysis, while optimising the detection capabilities.*»

Next we summarise the main contributions, result of the study of the suitability of providing unsolicited e-mail filtering systems with a semantic layer and the reduction of the labelling needs of these systems:

- **A model capable of dealing with the linguistic phenomena that encode semantic word relations to enhance current machine-learning methods.** We have proposed the first spam filtering model that uses an enhanced Topic-based Vector Space Model (eTVSM) to represent e-mail messages. More accurately, we use an implementation of an eTSVM that applies the WordNet semantic ontology for identifying synonym terms that share the same interpretation. The final results obtained with the use of eTVSM for the representation of e-mail messages to detect spam show that this approach achieves high levels of accuracy, while being able to keep the number of false positives (legitimate e-mails incorrectly classified as spam) to a minimum.

- **An ambiguity-resilient approach that improves filtering capabilities by adding a term disambiguation pre-processing step before the e-mail representation.** We have proposed the application of Word Sense Disambiguation (WSD) for spam filtering to recover the filtering capabilities of content-based methods. Our approach pre-processes e-mails disambiguating the terms before constructing the Vector Space Model (VSM) that represents the e-mail messages. The results obtained during the evaluation of this approach show that the pre-processing step of WSD, applied to a model that represents electronic mail for anti-spam systems, improves filtering rates. In addition, this approach maintains a minimum false positive rate, sometimes even reducing it, while detecting a large number of junk e-mails.

- **Collective Classification for bulk e-mail filtering, a semi-supervised approach that reduces the time-consuming task of labelling.** We have proposed the first spam filtering system that uses Collective Classification to optimise classification performance. This approach minimises the necessity of labelled e-mails without compromising the accuracy of the filter. Collective Classification algorithms for spam filtering have presented a suitable approach to optimising the classification of partially labelled data, thereby overcoming the massive number of unclassified spam e-mails that are created every day.

- **An anomaly-based filtering approach that is able to detect unsolicited e-mails by training the models with only one class.** We have presented a study of the effectiveness of anomaly detection applied to spam filtering, which reduces the necessity of labelling spam messages and only employs the representation of one class of e-mails (i.e., legitimate or spam). This study includes a presentation of the first anomaly-based spam filtering system, an enhancement of this system that applies a data reduction algorithm to the labelled dataset to reduce processing time while maintaining detection rates and an analysis of the suitability of choosing legitimate e-mails or spam as representation of normality. The final results show that the use of anomaly detection techniques for spam filtering achieves high accuracy

levels, minimising the labelling efforts with a dataset of only one class of e-mails: legitimate or spam.

Through the development of these main contributions, we have managed to accomplish the specific goals presented in Section 1:

Specific Goals 1 *Develop and evaluate a spam filtering model immune to linguistic phenomena.*

Specific Goals 2 *Build and evaluate a spam filtering model immune to word ambiguity.*

Specific Goals 3 *Develop and evaluate a model to reduce the labelling efforts that filtering systems require.*

In order to fulfil these specific goals we have also met certain operational goals:

Operational Goals 1 *Design and implement a method for the representation of e-mails taking into account the linguistic phenomena implicit in every natural language.*

Operational Goals 2 *Design and implement a method for the disambiguation of the terms that e-mails are composed of.*

Operational Goals 3 *Design and implement a method to reduce the labelled instances needed to obtain optimum results from statistical filtering systems.*

Operational Goals 4 *Optimise the ratio of false negatives and false positives of the spam filtering system.*

With the achievement of the established specific and operational goals we believe we have been able to *"develop and test spam filtering techniques capable of overcoming semantic attacks and reduce the labelling efforts required to maintain optimum detecting capabilities."*, the main goal. Therefore, we consider that the present work validates the fundamental hypothesis of this dissertation.

5.2 Discussion of the contributions

There are several important points to be discussed referring to the appropriateness of our proposed methods. Next we introduce some of them.

The first main shortcoming is related to the representation of the e-mails. The first proposed approach in this work employs the enhanced Topic-based

Vector Space Model (eTVSM) representation while the other three approaches (i.e., ambiguity-resilient, collective-classification-based and anomaly-based) employ the common Vector Space Model (VSM). All four approaches suffer from the impossibility of the VSM-based representation to fight attacks that modify term statistics (e.g., *Good Word Attack*). Besides, other attacks try to exploit the feature selection step of the filtering process, an example, *tokenisation*, which splits or modifies key message features. These kind of attacks, which spammers have been adopting, should be accounted for when constructing future spam-filtering systems.

The second problem identified is inherent to the main purpose of this work: to provide spam filtering systems with semantics-aware capabilities. The inclusion of semantics introduces a problem derived from Information Retrieval and Natural Language Processing, *language dependency* (Bates and Weischedel, 1993). Therefore, semantic approaches for spam filtering have to deal with the semantics of different languages.

5.3 Future lines of research

Here we present some possible solutions to the main shortcomings discussed in the previous section, along with the future open lines of research identified in the present dissertation.

5.3.1 Statistical and tokenisation attacks

As stated before, future unsolicited e-mail filtering research should take into account how statistical and tokenisation attacks endanger the filtering capabilities of anti-spam systems. The study of Multiple Instance Learning (MIL) (Dietterich et al., 1997) is proposed to fight attacks that modify term statistics. To fight tokenisation attacks a preprocessing step on the dataset could be used to identify tokenised words and "clean" them using for example the *Levenshtein distance* (Levenshtein, 1966). Particularly, is taking shape the field of *adversarial classification* Dalvi et al. (2004), where the analysis and learning systems should be aware of the existence of an adversary (the spammer), whose objective is to degrade the classification effectiveness of the systems built with these techniques.

5.3.2 Improving the semantics

Regarding our proposed word-sense-disambiguation-based approach, we employed a very basic input format for the disambiguation process. An improvement of the method could be achieved by enriching the disambiguation process with a more complex part-of-speech labelling format such as SemCor (Miller

et al., 1993). Furthermore, by using topics rather than terms to create the models (eTVSM approach) we were able to deal with the problem of synonymity, but other linguistic phenomena exist (Becker and Kuropka, 2003). Enhancing the semantics of this method with support for more linguistic relationships could be tackled in future research.

5.3.3 Enhancing the anomaly-based approach

Our proposed technique to reduce the labelling efforts based on anomaly detection is very dependent on the correct selection of the different thresholds that parametrise the filtering process. Because, these thresholds were chosen through empirical observation, an extensive analysis is mandatory to detect possible optimisations. An automated process could select the best threshold combinations to improve the results of our filtering system maybe using *genetic algorithms* (Koza and Poli, 2005). Besides, despite the behaviour using spam messages to represent normality showed poorly results, we obtained improvements when testing the TREC dataset, the most heterogeneous dataset of the three used. This fact leads to the conclusion that, in real world environments, legitimate e-mails could be considered the anomaly. Further research on this assumption should be performed with other recent datasets, studying the nature of both approaches to find possible optimisations.

5.3.4 Unsupervised learning

Unsupervised methods have proven their benefits in spam filtering and similar domains. Given the good results obtained in the reduction of the labelling efforts with our proposed approaches, the study of recent unsupervised techniques (Cormack, 2007a) in order to apply them to the unsolicited e-mail filtering problem should be taken into account.

5.3.5 Evolving nature of spam

The evolving nature of spam, which often undergoes cyclical changes (e.g., false Christmas greetings in December), forces the anti-spam community to look for techniques able to identify these variations. Besides, popular worldwide events of great impact are commonly used by spammers to create campaigns (e.g., spam over the World Cup in South Africa). The Topic Detection and Tracking (TDT) method is a technique that 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. 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.

5.3.6 Language dependency

In order to improve semantic filters it is mandatory to study the problem of language dependency. Different natural languages behave different, have different relations and interpretations, and all of them are affected with spam. Moreover, new technologies are changing the way of communication of new generations. On the one hand, the appearance of the Short Message Service (SMS) and its limited length in the messages created "alternative languages" with word contractions and initialisms (i.e., abbreviations formed from the initial letters of a sequence of words). Common filters were not able to detect spam in this channel because of their changed nature, and several approaches tried to address the problem (Gómez Hidalgo et al., 2006; Cormack et al., 2007a). Nowadays, this situation has been extended to on-line social networks, where the spam is becoming more and more a real serious problem (Zinman and Donath, 2007; Luo et al., 2009; Sanz et al., 2010; Laorden et al., 2010). Finding a solution to language dependency could lead to better semantic filters applicable in different channels of communication.

5.3.7 Joining all the approaches

In this work we have tested the different approaches separately. It would be interesting to validate a model with disambiguation capabilities and enhanced with the eTVSM representation of the e-mails. Moreover, the final model could include one of the proposed approaches to reduce the labelling efforts (i.e., collective approach and anomaly-based approach). Figure 5.1 offers the proposed schema to join all the approaches.

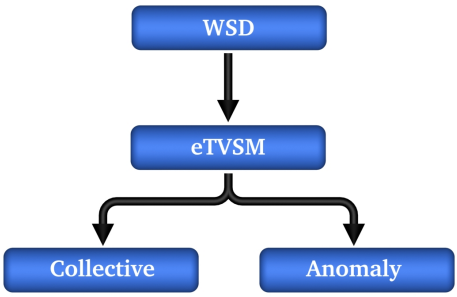


Figure 5.1: Proposed schema to join all the approaches.

The first step would be to disambiguate the content of the messages, to latter represent them using the enhanced version of the VSM, i.e., eTVSM. Finally, collective classification algorithms or the anomaly-based approach could be applied to the resultant model to reduce the labelling efforts.

5.4 Final remarks

The world moves fast. The appearance of new technologies such as on-line social networks facilitate the way people communicates. However, new channels for communication create new opportunities for spammers to spread their threats. In addition, unwanted communications are not limited to text. The number of undesired communications through other media resources, such as images or videos, is increasing everyday. This facts clearly show, that spam is not going away, at least not in a near future.

In a recent study from the Universities of Berkeley and Budapest, they detected that 95% of worldwide undesired e-mail is monetised by only 3 banks (Levchenko et al., 2011). This study could lead to the thought that, to end the global spam, the best solution is to cut the spam network billing systems. However, the problem of different legal jurisdictions, makes it impossible nowadays.

Will we see the end of spam? Legislative and technical measures may help reducing the amount of spam, but, the creativity and efforts of those who choose to ignore the law will always provide a great challenge to filtering systems. In light of this background, we choose to focus our efforts towards improving content-based filtering systems, with the internal hope to provide the spam filtering community with useful tools, techniques or, simply, ideas to fight it. Because, after all:

> «*Ideas are the spark of science. Find a good one and some working material, and you will get your fire.*»

Publications

Aspects of the work described in this dissertation feature in the following publications:

Journal Articles

1. C. Laorden, B. Sanz, I. Santos, P. Galán-García and P.G. Bringas. *Collective Classification for Spam Filtering*. Logic Journal of the IGPL. ISSN (online): 1368-9894, ISSN (print) 1367-0751. Impact Factor: 0.458 (JCR 2010) - 3rd quartile.

2. C. Laorden, I. Santos, B. Sanz, G. Alvarez and P.G. Bringas. *Word Sense Disambiguation for Spam Filtering*. Electronic Commerce Research and Applications. ISSN: 1567-4223. DOI: 10.1016/j.elerap.2011.11.004. Impact Factor: 1.946 (JCR 2010) - 1st quartile.

3. I. Santos, C. Laorden, B. Sanz and P.G. Bringas. *Enhanced Topic-based Vector Space Model for Semantics-aware Spam Filtering*. Expert Systems With Applications, 39(1). ISSN: 0957-4174. DOI: 10.1016/j.eswa.2011.07.034. Impact Factor: 1.924 (JCR 2010) - 1st quartile.

International Conference Proceedings

4. C. Laorden, X. Ugarte-Pedrero, I. Santos, B. Sanz, J. Nieves and P.G. Bringas. *On the Study of Anomaly-based Spam Filtering Using Spam as Representation of Normality*. In Proceedings of 3rd Consumer Communications and Network Conference (CCNC) Research Student Workshop, Las Vegas, Nevada (USA), 14-17 January 2012.

5. C. Laorden, X. Ugarte-Pedrero, I. Santos, B. Sanz and P.G. Bringas. *Enhancing Scalability in Anomaly-based Email Spam Filtering*. In Proceedings of the 8^th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference (CEAS), Perth (Australia), 1-2 September 2011, pp. 13-22.

6. I. Santos, C. Laorden, X. Ugarte-Pedrero, B. Sanz y P.G. Bringas. *Anomaly-based Spam Filtering*. In Proceedings of the 6^th International Conference on Security and Cryptography (SECRYPT), Sevilla (España), 18-21 de Julio 2011, pp. 5-14.

7. C. Laorden, B. Sanz, I. Santos, P. Galán-García and P.G. Bringas. *Collective Classification for Spam Filtering*. In Proceedings of the 4^th International Conference on Computational Intelligence in Security for Information Systems (CISIS). Torremolinos (Spain), 8-10th June, 2011, CISIS 2011, LNCS 6694, Á. Herrero and E. Corchado (Eds.), ISBN: 978-3-642-21323-6, pp. 1-8. Springer, Heidelberg (2011).

Bibliography

Adams, K. (2001). Representing knowledge in enterprise portals. *Knowledge Management World*.

Agirre, E. and 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*, volume 1998.

Amari, S. and 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., and Spyropoulos, C. (2000a). An evaluation of naive bayesian anti-spam filtering. In *Proceedings of the workshop on Machine Learning in the New Information Age*, pages 9–17.

Androutsopoulos, I., Koutsias, J., Chandrinos, K., and 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*, pages 160–167. ACM.

Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Sakkis, G., Spyropoulos, C., and 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*.

Attardi, G., Di Marco, S., and Salvi, D. (1998). Categorization by context. *Journal of Universal Computer Science*, 4(9):719–736.

Attardi, G., Gullí, A., and Sebastiani, F. (1999). Automatic web page categorization by link and context analysis. In *Proceedings of THAI-99, 1st European Symposium on Telematics, Hypermedia and Artificial Intelligence*, pages 105–119.

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

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

Bar-Hillel, Y. (1960). *The Present Status of Automatic Translation of Languages*, volume 1, pages 91–163.

Bates, M. and 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. and Kuropka, D. (2003). Topic-based vector space model. In *Proceedings of the 6th International Conference on Business Information Systems*, pages 7–12.

Belkin, N. and Croft, W. (1996). Information filtering and information retrieval: two sides of the same coin? *Communications of the ACM*, 35(12):29–38.

Berg, B. (1989). *Qualitative research methods for the social sciences*. Allyn and Bacon Boston.

Bernstein, D. (2000). Internet mail 2000.

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

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

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

Brefeld, U., Büscher, C., and Scheffer, T. (2005). Multi-view discriminative sequential learning. *Machine Learning: ECML 2005*, pages 60–71.

Brefeld, U. and Scheffer, T. (2006). Semi-supervised learning for structured output variables. In *Proceedings of the 23rd international conference on Machine learning*, pages 145–152. ACM.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Bunt, H. (1985). *Mass terms and model-theoretic semantics*, volume 295. Cambridge University Press.

Burton, B. (2003). Spamprobe-bayesian spam filtering tweaks. In *Proceedings of the Spam Conference*.

Carnap, R. (1955). Meaning and synonymy in natural languages. *Philosophical Studies*, 6(3):33–47.

Caropreso, M., Matwin, S., and Sebastiani, F. (2001). A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. *Text Databases and Document Management: Theory and Practice*, pages 78–102.

Carpinter, J. and 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., and Padró, M. (2004). Freeling: An open-source suite of language analyzers. In *Proceedings of the 4th LREC*, volume 4.

Carreras, X. and 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*, pages 58–64.

Carreras, X. and Padró, L. (2002). A flexible distributed architecture for natural language analyzers. In *Proceedings of the LREC*, volume 2.

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

Castillo, C., Chellapilla, K., and Denoyer, L. (2008). Web spam challenge 2008. In *Proceedings of the 4th International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*.

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

Cavnar, W. and Trenkle, J. (1994). N-gram-based text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175.

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

Chen, H. and Dumais, S. (2000). Bringing order to the web: automatically categorizing search results. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 145–152. ACM.

Chen, R. and Hsieh, C. (2006). Web page classification based on a support vector machine using a weighted vote schema. *Expert Systems with Applications*, 31(2):427–435.

Chirita, P., Diederich, J., and Nejdl, W. (2005). MailRank: using ranking for spam detection. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 373–380. ACM.

Chiu, Y., Chen, C., Jeng, B., and Lin, H. (2007). An Alliance-Based Anti-spam Approach. In *Natural Computation, 2007. ICNC 2007. Third International Conference on*, volume 4, pages 203–207. IEEE.

Cleverdon, C. (1984). Optimizing convenient online access to bibliographic databases. *Information services and Use*, 4(1):37–47.

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

Cohen, W. (1995). Fast effective rule induction. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 115–123.

Cohen, W. (1996). Learning rules that classify e-mail. In *AAAI Spring Symposium on Machine Learning in Information Access*, volume 18.

Cooper, G. F. and 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. (2006). Harnessing unlabeled examples through iterative application of dynamic markov modeling. In *Proceedings of the ECML/PKDD 2006 Discovery Challenge Workshop*, pages 10–15.

Cormack, G. (2007a). Email spam filtering: A systematic review. *Foundations and Trends in Information Retrieval*, 1(4):335–455.

Cormack, G. (2007b). TREC 2007 spam track overview. In *Sixteenth Text REtrieval Conference (TREC-2007)*.

Cormack, G., Gómez Hidalgo, J., and Sánz, E. (2007a). Spam filtering for short messages. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 313–320. ACM.

Cormack, G., Hidalgo, J., and Sánz, E. (2007b). Feature engineering for mobile (sms) spam filtering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 871–872. ACM.

Cormack, G. and Lynam, T. (2005). Spam corpus creation for trec. In *Proceedings of Second Conference on Email and Anti-Spam CEAS*.

Cormack, G. and Lynam, T. (2007). Online supervised spam filter evaluation. *ACM Transactions on Information Systems (TOIS)*, 25(3):11.

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

Crystal, D. (1999). *The Penguin dictionary of language*. Penguin.

Dalvi, N., Domingos, P., et al. (2004). Adversarial classification. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, page 108. ACM.

Damiani, E., De Capitani di Vimercati, S., Paraboschi, S., and Samarati, P. (2004). P2p-based collaborative spam detection and filtering. In *Peer-to-Peer Computing, 2004. Proceedings. Proceedings. Fourth International Conference on*, pages 176–183. IEEE.

Dantu, R. and Kolan, P. (2005). Detecting spam in voip networks. In *Proceedings of the Steps to Reducing Unwanted Traffic on the Internet on Steps to Reducing Unwanted Traffic on the Internet Workshop*, pages 5–5.

De Sa, V. (1994). Learning classification with unlabeled data. *Advances in neural information processing systems*, pages 112–112.

Dengel, A. and Dubiel, F. (1995). Clustering and classification of document structure-a machine learning approach. *Document Analysis and Recognition, International Conference on*, 2:587.

Denoyer, L. and Gallinari, P. (2004). Bayesian network model for semi-structured document classification. *Information Processing & Management*, 40(5):807–827.

Dietterich, T., Lathrop, R., and Lozano-Pérez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71.

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

Dumais, S., Platt, J., Heckerman, D., and Sahami, M. (1998). Inductive learning algorithms and representations for text categorization. In *Proceedings of the seventh international conference on Information and knowledge management*, pages 148–155. ACM.

Elkan, C. (2001). The foundations of cost-sensitive learning. In *Proceedings of the 2001 International Joint Conference on Artificial Intelligence*, pages 973–978.

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

Fix, E. and 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., editor (2003). *Blackwell Guide to the Philosophy of Computing and Information*. Blackwell Publishers, Inc., Cambridge, MA, USA.

Forsyth, R. (1999). New directions in text categorization. *Causal models and intelligent data management*, pages 151–185.

Friedl, J. (2006). *Mastering regular expressions*. O'Reilly Media, Inc.

Galavotti, L., Sebastiani, F., and Simi, M. (2000). Experiments on the use of feature selection and negative evidence in automated text categorization. *Research and Advanced Technology for Digital Libraries*, pages 59–68.

Gansterer, W., Ilger, M., Lechner, P., Neumayer, R., and Strauß, J. (2005). Anti-spam methods-state of the art. *Institute of Distributed and Multimedia Systems, University of Vienna*.

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

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

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

Gómez Hidalgo, J., Bringas, G., Sánz, E., and García, F. (2006). Content based sms spam filtering. In *Proceedings of the 2006 ACM symposium on Document engineering*, pages 107–114. ACM.

Gómez Hidalgo, J., Sanz, E., García, F., and Rodríguez, M. (2009). Web content filtering. *Advances in Computers*, 76:257–306.

Gonzalo, J., Penas, A., and Verdejo, F. (1999). Lexical ambiguity and information retrieval revisited. In *Proceedings of EMNLP/VLC*, volume 99.

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

Gray, A. and Haahr, M. (2004). Personalised, collaborative spam filtering. In *Proceedings of 1st conference on email and anti-spam*.

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

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

Hayes, P. and Weinstein, S. (1990). Construe/tis: a system for content-based indexing of a database of news stories. In *Second Annual Conference on Innovative Applications of Artificial Intelligence*, pages 49–66.

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

Heyer, L., Kruglyak, S., and Yooseph, S. (1999). Exploring expression data: identification and analysis of coexpressed genes. *Genome research*, 9(11):1106–1115.

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

Hovold, J. (2005). Naive bayes spam filtering using word-position-based attributes. In *Proceedings of the Second Conference on Email and Anti-Spam (CEAS 2005)*.

Ide, N. and Véronis, J. (1998). Introduction to the special issue on word sense disambiguation: the state of the art. *Computational linguistics*, 24(1):2–40.

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

Jagatic, T., Johnson, N., Jakobsson, M., and Menczer, F. (2007). Social phishing. *Communications of the ACM*, 50(10):94–100.

Joachims, T. (1999). Transductive inference for text classification using support vector machines. In *Proceedings of ICML-99, 16th International Conference on Machine Learning*, pages 200–209.

Junejo, K., Yousaf, M., and Karim, A. (2006). A two-pass statistical approach for automatic personalized spam filtering. In *Proceedings of the ECML/PKDD 2006 Discovery Challenge Workshop*, pages 16–27.

Jung, J. and 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*, pages 370–375. ACM New York, NY, USA.

Karlberger, C., Bayler, G., Kruegel, C., and 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*, pages 1–7, Berkeley, CA, USA. USENIX Association.

Kent, J. (1983). Information gain and a general measure of correlation. *Biometrika*, 70(1):163–173.

Klas, C. and Fuhr, N. (2000). A new effective approach for categorizing web documents. In *Proceedings of BCSIRSG-00, the 22nd Annual Colloquium of the British Computer Society Information Retrieval Specialist Group*.

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

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

Kolesnikov, O., Lee, W., and Lipton, R. (2003). Filtering spam using search engines. Technical report, Georgia Tech., College of Computing, Georgia Institute of Technology, Atlanta, GA 30332 (2004-2005).

Koller, D. and Sahami, M. (1997). Hierarchically classifying documents using very few words. In *Proceedings of ICML-97, 14th International Conference on Machine Learning*, pages 170–178.

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

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*, pages 3–24.

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

Koza, J. and Poli, R. (2005). Genetic programming. *Search Methodologies*, pages 127–164.

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

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

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

Kumar, V. (2000). An introduction to cluster analysis for data mining. *Computer Science Department, University of Minnesota, USA*.

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

Labrou, Y. and Finin, T. (1999). Yahoo! as an ontology: using yahoo! categories to describe documents. In *Proceedings of the eighth international conference on Information and knowledge management*, pages 180–187. ACM.

Laorden, C., Sanz, B., Alvarez, G., and Bringas, P. G. (2010). A threat model approach to threats and vulnerabilities in on-line social networks. In *Computational Intelligence in Security for Information Systems 2010*, volume 85 of *Advances in Intelligent and Soft Computing*, pages 135–142.

Larkey, L. (1998). Automatic essay grading using text categorization techniques. In *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*, pages 90–95.

Larkey, L. and Croft, W. (1996). Combining classifiers in text categorization. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 289–297. ACM.

Leung, C. and 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)*.

Levchenko, K., Pitsillidis, A., Chachra, N., Enright, B., Grier, C., Halvorson, T., Kanich, C., Kreibich, C., Liu, H., McCoy, D., et al. (2011). Click trajectories: End-to-end analysis of the spam value chain. In *Security and Privacy (SP), 2011 IEEE Symposium on*, pages 431–446. IEEE.

Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710.

Lewis, D. (1992). *Representation and learning in information retrieval*. PhD thesis, University of Massachusetts.

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

Lewis, D. and Ringuette, M. (1994). A comparison of two learning algorithms for text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 81–93.

Lin, H. and 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(1):22–31.

Luo, W., Liu, J., Liu, J., and Fan, C. (2009). An analysis of security in social networks. In *Dependable, Autonomic and Secure Computing, 2009. DASC'09. Eighth IEEE International Conference on*, pages 648–651. IEEE.

Maeireizo, B., Litman, D., and Hwa, R. (2004). Co-training for predicting emotions with spoken dialogue data. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, page 28. Association for Computational Linguistics.

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*.

Maron, M. (1961). Automatic indexing: an experimental inquiry. *Journal of the ACM (JACM)*, 8(3):404–417.

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

Mason, J. (2002). Filtering spam with spamassassin. In *HEANet Annual Conference*.

Mavroeidis, D., Chaidos, K., Pirillos, S., Christopoulos, D., and Vazirgiannis, M. (2006). Using tri-training and support vector machines for addressing the ecml-pkdd 2006 discovery challenge. In *Proceedings of the ECML/PKDD 2006 Discovery Challenge Workshop*, pages 39–47.

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

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

McGraw, G. and Morrisett, G. (2000). Attacking malicious code: A report to the infosec research council. *Software, IEEE*, 17(5):33–41.

Metsis, V., Androutsopoulos, I., and Paliouras, G. (2006). Spam filtering with naive bayes-which naive bayes. In *Proceedings of the Third Conference on Email and Anti-Spam (CEAS)*, volume 17, pages 28–69.

Meyer, T. and Whateley, B. (2004). Spambayes: Effective open-source, bayesian based, email classification system. In *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*, volume 98.

Mihalcea, R. and 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*, pages 53–56. Association for Computational Linguistics.

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

Ming-Tzu, K. and Nation, P. (2004). Word meaning in academic English: Homography in the academic word list. *Applied linguistics*, 25(3):291–314.

Mishne, G., Carmel, D., and Lempel, R. (2005). Blocking blog spam with language model disagreement. In *Proceedings of the first international workshop on adversarial information retrieval on the Web (AIRWeb)*, pages 1–6.

Mladenic, D. (1998). Turning yahoo into an automatic web-page classifier. In *Proceedings of the 13th European Conference on Artificial Intelligence*, pages 473–474.

Mostafa Raad, N., Alam, G., Zaidan, B., and 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.

Moulinier, I., Raskinis, G., and Ganascia, J. (1996). Text categorization: a symbolic approach. In *Proceedings of SDAIR-96, 5th Annual Symposium on Document Analysis and Information Retrieval*, pages 87–99.

Myers, K., Kearns, M., Singh, S., and Walker, M. (2000). A boosting approach to topic spotting on subdialogues. In *Proceedings of ICML-00, 17th International Conference on Machine Learning*.

Nakov, P. and Dobrikov, P. (2004). Non-parametric spam filtering based on knn and lsa. In *Proceedings of the 33th National Spring Conference*.

Namata, G., Sen, P., Bilgic, M., and Getoor, L. (2009). Collective classification for text classification. *Text Mining*, pages 51–69.

Navigli, R. (2009). Word sense disambiguation: a survey. *ACM Computing Surveys*, 41(2):1–69.

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

Parvathaneni, S. (2011). Collaborative spam filtering. *International Journal of Engineering Research and Applications*, 1(3):427–431.

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

Pfahringer, B. (2006). A semi-supervised spam mail detector. In *Proceedings of the ECML/PKDD 2006 Discovery Challenge Workshop*, pages 18–22.

Plag, I. (2003). *Word-formation in English*. Cambridge University Press.

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.

Provost, J. (1999). Naıve-bayes vs. rule-learning in classification of email. *University of Texas at Austin*.

Qi, X. and Davison, B. (2009). Web page classification: Features and algorithms. *ACM Computing Surveys (CSUR)*, 41(2):1–31.

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.

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

Radicati, S. (2010). Email statistics report, 2010. Technical report, The Radicati Group, Inc.

Radicati, S. and Khmartseva, M. (2009). Email statistics report, 2009-2013. Technical report, The Radicati Group, Inc.

Rajan, S., Yankov, D., Gaffney, S., and Ratnaparkhi, A. (2010). A large-scale active learning system for topical categorization on the web. In *Proceedings of the 19th international conference on World wide web*, pages 791–800. ACM.

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

Raymond, E. (2005). Bogofilter: A fast open source bayesian spam filters.

Riemers, B. (2003). Automatic uniform resource locator-based message filter. US Patent 6,615,242.

Riloff, E. and Wiebe, J. (2003). Learning extraction patterns for subjective expressions. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 105–112. Association for Computational Linguistics.

Rios, G. and Zha, H. (2004). Exploring support vector machines and random forests for spam detection. In *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*, pages 284–292.

Robinson, G. (2003). A statistical approach to the spam problem. *Linux J.*, 2003:3.

Rosenberg, C., Hebert, M., and Schneiderman, H. (2005). Semi-supervised self-training of object detection models. In *Seventh IEEE Workshop on Applications of Computer Vision*. IEEE Press.

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

Sable, C. and Hatzivassiloglou, V. (2000). Text-based approaches for non-topical image categorization. *International Journal on Digital Libraries*, 3(3):261–275.

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

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

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

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

Salton, G., Wong, A., and 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*, pages 142–151, New York, NY, USA. Springer-Verlag New York, Inc.

Santos, I., Laorden, C., and Bringas, P. G. (2011a). Collective classification for unknown malware detection. In *Proceedings of the 6^{th} International Conference on Security and Cryptography (SECRYPT)*, pages 251–256.

Santos, I., Nieves, J., and Bringas, P. G. (2011b). Semi-supervised learning for unknown malware detection. In *Proceedings of the 4^{th} International Symposium on Distributed Computing and Artificial Intelligence (DCAI). 9th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS)*, pages 415–422.

Santos, I., Sanz, B., Laorden, C., Brezo, F., and Bringas, P. G. (2011c). Opcode-sequence-based semi-supervised unknown malware detection. In *Proceedings of the 4^{th} International Conference on Computational Intelligence in Security for Information Systems (CISIS)*, pages 50–57.

Santos, I., Ugarte-Pedrero, X., Sanz, B., Laorden, C., and Bringas, P. G. (2011d). Collective classification for packed executable identification. In *Proceedings of*

the 8^{th} Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference (CEAS), pages 23–30.

Sanz, B., Laorden, C., Alvarez, G., and Bringas, P. G. (2010). A threat model approach to attacks and countermeasures in on-line social networks. In *In Proceedings of the 11th Reunion Española de Criptografía y Seguridad de la Información (RECSI), 7-10th September, Tarragona (Spain), in press.*, pages 343–348.

Sanz, E., Gómez Hidalgo, J., and Cortizo Pérez, J. (2008). Email spam filtering. *Advances in Computers*, 74:45–114.

Savoy, J. and Zubaryeva, O. (2011). Classification based on specific vocabulary. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2011 IEEE/WIC/ACM International Conference on*, volume 1, pages 120–123. IEEE.

Schapire, R. and Singer, Y. (2000). Boostexter: A boosting-based system for text categorization. *Machine learning*, 39(2):135–168.

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*, pages 307–314.

Schryen, G. (2006). A formal approach towards assessing the effectiveness of anti-spam procedures. In *System Sciences, 2006. HICSS'06. Proceedings of the 39th Annual Hawaii International Conference on*, volume 6, pages 129–138. IEEE.

Sculley, D. and 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*, pages 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.

Shunli, Z. and Qingshuang, Y. (2010). Personal spam filter by semi-supervised learning. In *Proceedings of The Third International Symposium on Computer Science and Computational Technology (ISCSCT)*, pages 171–174.

Sindhwani, V., Niyogi, P., and Belkin, M. (2005). Beyond the point cloud: from transductive to semi-supervised learning. In *Proceedings of the 22nd international conference on Machine learning*, pages 824–831. ACM.

Singh, Y., Kaur, A., and 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., and Smith, J. L. (2011). Case studies of cybercrime and their impact on marketing activity and shareholder value. *Academy of Marketing Studies Journal*.

Snyder, J. (2004). Spam in the wild, the sequel. *Network World 12/20*, 4.

Stump, G. (2001). *Inflectional morphology: A theory of paradigm structure*, volume 93. Cambridge University Press.

Susman, G. (1983). Action research: a sociotechnical systems perspective. *Beyond method: Strategies for social research*, pages 95–113.

Teahan, W. (2000). Text classification and segmentation using minimum cross-entropy. In *Proceeding of RIAO-00, 6th International Conference "Recherche d'Information Assistee par Ordinateur"*.

Tiun, S., Abdullah, R., and Kong, T. (2010). Automatic topic identification using ontology hierarchy. *Computational Linguistics and Intelligent Text Processing*, pages 444–453.

Ugarte-Pedrero, X., Santos, I., Bringas, P., Gastesi, M., and Esparza, J. (2011). Semi-supervised learning for packed executable detection. In *In Proceedings of the 5th International Conference on Network and System Security (NSS)*, pages 342–346.

Üstün, B., Melssen, W., and 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. (1998). Statistical learning theory. 1998.

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

Voorhees, E. (1999). Natural language processing and information retrieval. *Information Extraction*, pages 724–724.

Weaver, W. (1955). Translation. *Machine translation of languages*, 14:15–23.

Webb, S., Caverlee, J., and Pu, C. (2006). Introducing the webb spam corpus: Using email spam to identify web spam automatically. In *Proceedings of the Third Conference on Email and Anti-Spam (CEAS)*.

Weinman, W. (2003). Authenticated mail transfer protocol.

Wiener, E., Pedersen, J., and Weigend, A. (1995). A neural network approach to topic spotting. In *Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval*, volume 332. Las Vegas, NV, USA: Univ. of Nevada.

Wilbur, W. and 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., and Slator, B. M. (1990). Providing machine tractable dictionary tools. *Machine Translation*, 5(2):99–154.

Wilks, Y. and Stevenson, M. (1996). The grammar of sense: Is word sense tagging much more than part-of speech tagging. Technical report, University of Sheffield, Sheffield, UK.

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

Wolpert, D. (1992). Stacked generalization*. *Neural networks*, 5(2):241–259.

Xu, C. and Zhou, Y. (2007). Transductive support vector machine for personal inboxes spam categorization. In *Computational Intelligence and Security Workshops, 2007. CISW 2007. International Conference on*, pages 459–463. IEEE.

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

Yang, Y. and Pedersen, J. (1997). A comparative study on feature selection in text categorization. In *Proceedings of ICML-97, 14th International Conference on Machine Learning*, pages 412–420.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196. Association for Computational Linguistics.

Yildiz, T., Yildirim, S., and Altilar, D. (2008). Spam filtering with paralellized knn algorithm. *Akademik Bilişim*.

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

Zhou, D., Bousquet, O., Lal, T., Weston, J., and Schölkopf, B. (2004). Learning with local and global consistency. In *Advances in Neural Information Processing Systems 16: Proceedings of the 2003 Conference*, pages 595–602.

Zhou, Y., Jorgensen, Z., and Inge, M. (2007). Combating Good Word Attacks on Statistical Spam Filters with Multiple Instance Learning. In *Proceedings of the 19^{th} IEEE International Conference on Tools with Artificial Intelligence-Volume 02*, pages 298–305. IEEE Computer Society.

Zhu, X. (2005). Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison.

Zinman, A. and Donath, J. (2007). Is britney spears spam. In *Proceedings of the Fourth Conference on Email and Anti-Spam (CEAS)*.

Appendix

APPENDIX

A

Datasets used in the empirical evaluation

In order to validate our proposed methods, we used the Ling Spam[1] dataset, the SpamAssassin[2] public mail corpus and the TREC 2007 Public Corpus (Cormack, 2007b).

Ling Spam consists of a mixture of both spam and legitimate messages retrieved from the *Linguistic list*, an e-mail distribution list about *linguistics*. The dataset was preprocessed by removing HTML tags, separation tokens and duplicate e-mails: only the data of the body and the subject were kept. Ling Spam comprises 2,893 different e-mails, of which 2,412 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)). *Stop Word Removal* (Wilbur and 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.

[1] http://nlp.cs.aueb.gr/software_and_datasets/lingspam_public.tar.gz
[2] http:// spamassassin.apache.org

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.

The SpamAssassin corpus contains messages that were posted to public fora, originated as newsletters from public news web sites, or sent by anonymous people. It's split into three parts, 500 spam messages received from non-spam-trap sources, 2,500 legitimate messages typically quite easy to differentiate from spam, 250 legitimate e-mails closer in many aspects to typical spam (e.g., use of HTML, spam-related phrases, coloured text), a more recent addition of 1,400 legitimate messages and another more recent set of 1,396 spam messages. A total of 6,046 messages, of which 1,896 are spam and 4,150 are legitimate e-mails.

Finally, the TREC 2007 Public Corpus (Cormack, 2007b) 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 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, 2007b):

- **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

Index