SpamHunting: An Instance-Based Reasoning System for Spam Labelling and Filtering

F. Fdez-Riverola[1], E. L. Iglesias[1], F. Díaz[2], J. R. Méndez[1] and J. M. Corchado[3]

[1] Dept. Informática, University of Vigo, Escuela Superior de Ingeniería Informática,
Edificio Politécnico, Campus Universitario As Lagoas s/n, 32004, Ourense, Spain
{riverola | eva | moncho.mendez}@uvigo.es
[2] Dept. Informática, University of Valladolid, Escuela Universitaria de Informática,
Plaza Santa Eulalia, 9-11, 40005, Segovia, Spain
fdiaz@infor.uva.es
[3] Dept. Informática y Automática, University of Salamanca,
Plaza de la Merced s/n, 37008, Salamanca, Spain
corchado@usal.es

Abstract. In this paper we show an instance-based reasoning e-mail filtering model that outperforms classical machine learning techniques and other successful lazy learners approaches in the domain of anti-spam filtering. The architecture of the learning-based anti-spam filter is based on a tuneable enhanced instance retrieval network able to accurately generalize e-mail representations. The reuse of similar messages is carried out by a simple unanimous voting mechanism to determine whether the target case is spam or not. Previous to the final response of the system, the revision stage is only performed when the assigned class is spam whereby the system employs general knowledge in the form of meta-rules.

Keywords. IBR system, automatic reasoning, anti-spam filtering, model comparison.

1. Introduction and Motivation

Though many of us may think of spam (UCE, *Unsolicited Commercial E-mail*) as a new problem, it goes back at least to 1975, as observed by Jon Postel [1]. Initially, spam mostly referred to Usenet newsgroup posts that got out of hand, whereby someone would mail a message to many newsgroups - a message that was unrelated to most of the newsgroups to which it was mailed. Then, social and administrative action was sufficient: the guilty party was punished privately or publicly and repeat offenders would be added to *kill lists*. As such, early spam filtering simply identified bad senders.

The huge expansion of Internet usage in recent years has increased marketing opportunities. As a result the problem of spam has grown astronomically, and the earlier techniques for keeping it under control no longer work. Unsolicited commercial communications now represent more than fifty per cent of the e-mail traffic in the European Union and around the world.

Spam is beginning to undermine the integrity of e-mail and even to discourage its use. The great majority of Internet users' mail boxes are swamped by unwanted messages over which they have no control. In large numbers, Internet users have reported that they trust e-mail less, and 29% of users even say they do not use e-mail as much as they used to because of spam [2]. Users worry that the growing volume of spam is getting in the way of their ability to safely send and receive e-mail.

In order to reduce the inconveniences continually imposed by spam on individuals, users, subscribers, consumers, companies, direct marketers, internet service providers, traders, employers, organizations, public bodies, and, in the end, the Internet itself, advances are taking place on two different fronts: (*i*) *legal measures* and (*ii*) *anti-spam filter software*. Software developments are typically based on placing filters in the mail agents, servers and clients, able to detect and to stop illegitimate mail. In this sense, two approaches exist when developing technical solutions that facilitate the control of spam: (*i*) *collaborative* applications [3] and (*ii*) *content-based* techniques.

The collaborative approach depends on the cooperation of user's groups who share information about spam. When a new spam message appears, the final user of the e-mail shares a signature (using hash codes) for the message with the rest of the group. If other users also receive the same message, they can identify it as spam based on the shared signature. Although it is a good idea, this type of technique presents two main problems. Firstly, spammers insert random characters into messages to foil signatures. Secondly, a process for sharing these signatures needs to be developed. The alternative is the content-based approach, where the classification is based on the content analysis of the features extracted from the e-mail body and header.

The success of machine learning (ML) techniques in text categorization [4] has led researchers to explore learning algorithms in anti-spam filtering. However, anti-spam filtering has a further complication because it is a cost-sensitive problem: the cost of accidentally blocking a legitimate message can be higher than letting a spam message pass the filter, and this difference must be taken into account during both training and evaluation tasks [5, 6].

Another important aspect of anti-spam filtering domain is the necessity to manage concept drift problem. While much of the research on machine learning has focused on static problems [7], a significant issue in many real-world domains is its changing environment [8]. In those situations, the target concepts (spam or legitimate e-mails) may depend on some hidden context and usually this context is dynamic. Changes in the hidden context can induce changes in the target concept, which is generally known as concept drift [9]. Concept drift in spam is particularly difficult as the spammers actively change the nature of their messages to elude spam filters.

In this paper we present our SpamHunting system, a lazy learning model based on an Instance-Based Reasoning (IBR) approach [10] to accurately solve the problem of spam labelling and filtering. Machine learning techniques usually aim at compressing available sample data into more compact representations called *models*. These models can then be used for solving different explorative (data mining) or predictive inference tasks. As opposed to traditional model-based machine learning, lazy learning [11] refers to methods that defer all essential computation until the specific prediction task is completely determined. This type of predictive inference is also known as case-based or instance-based reasoning, reflecting the fact that the methods use a set of sample instances (the case base) in a central role in the predictive inference process. The main contributions of this study to the research in the area are the definition of a new memory structure called EIRN (*Enhanced Instance Retrieval Network*) capable of storing a flexible e-mail representation and the implementation of a revision stage in the IBR process. The proposed system incorporates the EIRN model which facilitates the indexation of instances and the selection of those that are most similar to the incoming e-mails. The reuse of similar messages is carried out by a unanimous voting algorithm which generates an initial solution by creating a model with the retrieved instances. The revision stage in the proposed model is only carried out in the case of spam messages. For this purpose, the system employs general knowledge in the form of meta-rules that are extracted from the e-mail headers. Finally, the learning stage is carried out updating the knowledge structure of the whole system.

Following the advantages of instance-based reasoning as lazy learning technique mentioned above, we have compared our model with several artificial intelligence (AI) techniques currently used for the construction of anti-spam filters. Starting from the experiments carried out using publically available data, we show extremely promising results, with our system obtaining better indicators compared to the rest of the models analyzed.

The rest of the paper is organized as follows: Section 2 introduces the spam problem domain taking into account previous related research work; Section 3 discusses in detail our SpamHunting system, explaining the basis that conform each phase of the model; Section 4 introduces our experimental results, investigating separately the effect of several evaluation metrics with different models and corpus; Finally, Section 5 concludes and suggests new directions for further research.

2. Related Work on Spam Filtering

In recent years, a wide variety of learning approaches have been applied to the anti-spam problem. Those techniques mainly include ML algorithms as well as case and memory-based systems. Successful applications have been developed: encoding of probabilistic classifiers like Naïve Bayes [12] or its improved version called Flexible Bayes [13]; boosting of C4.5 [14]; C4.5 with PART [6]; generation of classification rules using different algorithms like Ripper [15] or Rocchio [16]; or the well-known Support Vector Machine (SVM) binary classifier [17].

Most of these algorithms try to minimize the number of errors in the classifier. However, these algorithms assign the same importance to all errors, which does not apply to the environment of spam, where the errors made by the classifier have varying significance for the end users. As such, assuming that the positive class to detect is spam, a False Positive error (FP, classifying a legitimate e-mail as spam) incurs a greater cost than a False Negative error (FN, classifying a spam message as legitimate). In order to create cost-sensitive learning methods, successful adaptations of existing algorithms have been made [18]. Another alternative is the generation of a cost-sensitive classifier starting from a learning algorithm plus a training collection and a cost distribution [19].

The Naïve Bayes learner is the most widely used algorithm. Although the independence assumption is over-simplistic, studies in anti-spam filtering have found Naïve Bayes to be effective [5, 20, 21]. At run time, Naïve Bayes requires O(p) computations for each test message.

Flexible Bayes provides an alternative approach to Naïve Bayes for continuous attributes. Instead of using a single normal distribution, the a posteriori probability of each attribute is calculated as the average of normal distributions. Despite its weaker model bias, the computational complexity of Flexible Bayes is the same as Naïve Bayes. However, there is an increase in the number of computations at run time.

SVMs constitute a family of algorithms for classification and regression developed by V. Vapnik [7] and is one of the most widely used ML techniques. SVMs can use all the terms of the messages. It is not necessary to carry out a previous selection, as in other learning algorithms, because their learning capacity does not degrade even if many characteristics exist [23].

The boosting algorithms are techniques based on the use of *weak learners*, that is to say, algorithms that learn with a next error rate of up to 50%. The classification trees C4.5 [22] are good candidates to be weak learners, although their error rates are much better that 50%. Different boosting algorithms have been developed for classification tasks, both binary and multi-class. Among them we could highlight AdaBoost [14], which boosts using C4.5 trees [23] or LogitBoost [24].

Ripper [15] induces classification rules from a set of examples. Unlike the algorithms above, it does not need a feature vector. It forms *if–then* rules which are disjunctions of conjunctions. Rocchio [16] uses normalized TF-IDF (*Term Frequency-Inverse Document Frequency*) representation of the training vectors. The advantage of this algorithm is its fast training and testing stages.

As part of the study of anti-spam filtering techniques, there has also been intensive research into the use of memory-based classifiers [5, 21, 25]. In general, using memory-based anti-spam filters are obtaining better results than ML algorithm-based approaches, mainly when the cost of the FP errors is high [25]. Moreover, case-based approaches are suitable for spam classification because they offer a natural framework to unify learning and collaboration approaches and to continually learn in the presence of new knowledge [26].

Memory-based (or instance-based) methods store all training instances in a memory structure and use them directly for classification [5]. They do not construct a unique model for each category, but simply store the training examples. In its simplest form, memory-based learning treats instances as points in a multi-dimensional space defined by the features that have been selected. Each training instance is represented as a point in that space.

Test instances are classified by estimating their similarity (distance) to the stored examples. Classification is usually performed through a variant of the basic *k-nearest-neighbour* (*k-nn*) algorithm. *k-nn* assigns to each test instance, the majority class of its *k* closest training instances.

TiMBL [27] provides an implementation of a basic memory-based classification algorithm with a variant of *k-nn*. One important difference from *k-nn* basic is in the definition of the *k*-neighbourhood. TiMBL considers that all training instances at the *k* closest *distances* form the unseen instance.

In general, memory-based approaches appear to be more appropriate than ML techniques for anti-spam filtering, especially when the assigned cost of a FP error is high. However, these systems need to configure the filter appropriately [27].

Case-based approaches outperform previous techniques to anti-spam filtering [28]. This is because spam is a disjoint concept: spam about *porn* has little in common with spam offering *rolexes*. Case-based classification works well for disjoint concepts whereas ML techniques try to learn a unified concept description. Another advantage of this approach is the ease with which it can be updated to catch the concept drift in spam.

[28] has presented a case-based system for anti-spam filtering called ECUE (*E-mail Classification Using Examples*) that can learn dynamically. In ECUE each e-mail is a case represented as a vector of binary features. The system use a similarity retrieval algorithm based on the utilization of *Case Retrieval Nets* (CRN) [29]. CRN networks are equivalent to the *k-nn* algorithm but are computationally more efficient in domains where there is feature-value redundancy and missing features in cases, such as in spam. The ECUE is a system evolved from a previously successful system [26] designed by the same authors.

3. IBR SpamHunting Proposed Architecture

The idea behind case-based reasoning (CBR) is that people rely on concrete previous experiences when solving new problems [30]. This fact can be tested on any day to day problem by simple observation or even by psychological experimentation. A CBR system solves new problems by adapting solutions that were used to solve old problems. The case base holds a number of problems with their corresponding solutions. Once a new problem arises, the solution to it is obtained by retrieving similar cases from the case base and studying the similarity between them. A CBR system is a dynamic system in which new problems are added to the case base, redundant ones are eliminated and others are created by combining existing ones. A typical CBR system is composed of four sequential steps which are recalled every time a problem needs to be solved: retrieve the most relevant case(s), reuse the case(s) to attempt to solve the problem, revise the proposed solution if necessary, and retain the new solution as a part of a new case. Each of the steps of the CBR life cycle requires a model or method in order to perform its mission.

CBR is an incremental learning approach because every time a problem is solved a new experience can be retained and made immediately available for future retrievals. The nature of the problem and the expertise of the CBR designers determine how the CBR should be built. According to Aamodt and Plaza [30] there are five different types of CBR systems, and although they share similar features, each of them is more appropriate for a particular type of problem: exemplar-based reasoning, instance-based reasoning, memory-based reasoning, analogy-based reasoning and typical case-based reasoning. Instance-based reasoning can be considered to be a type of exemplar-based reasoning in highly syntax-dependent problem areas [31, 32, 33]. This type of CBR system focuses on problems in which there are a large number of instances which are needed to represent the whole range of the domain and where there is a lack of general background knowledge. The case representation can be made with feature vectors and the phases of the CBR cycle are normally automated as much as possible, eliminating human intervention.

An Instance-Based Reasoning Architecture has been developed for Spam classification. In order to correctly classify an incoming e-mail, a message descriptor should be generated. This message descriptor consists of a sequence of *N* features that better summarize the information contained in the e-mail. For this purpose, we use data from two main sources: (*i*) information obtained from the header of the e-mail (see Table 1) and (*ii*) those terms that are more representative of the subject, body and attachments of the message (see Table 2).

The selected features shown in Table 1 are extracted from the e-mail header and stored as is. Those features represent global knowledge about a message. SpamHunting uses those features in the revision stage, where meta-rules are generated in order to ascertain the initial solution proposed by the system.

<Table 1 goes here>

In order to obtain an accurate representation of each one of the e-mails, we calculate the frequency of each feature (term) within the actual message. This information allows us to obtain a more precise knowledge about the contents of the e-mail and gain a deeper insight into its structure.

<Table 2 goes here>

Based on the information compiled in Tables 1 and 2 and taking into account the class of the message (spam or legitimate), we construct an instance representation for each message existing in the original corpus. Figure 1 shows how we represent each one of the e-mails in the SpamHunting system.

Analyzing Figure 1, we can see that the size of the *Frequency-Term Descriptor* array is variable and depends on each e-mail. The value of the parameter *n* is stored in the *Total Terms* field of the instance-message descriptor (longest dotted arrow in Figure 1). Moreover, from the *Frequency-Term Descriptor* array we are interested in selecting the most relevant terms of each e-mail for message indexing and retrieval. As in the case of the *n* parameter, the number of relevant terms is different from e-mail to e-mail and it is stored in the *Relevant Terms* field.

<Figure 1 goes here>

Figure 2 illustrates the life-cycle of the IBR SpamHunting system as well as its integration within a typical user environment. In the upper part of Figure 2, the mail user agent (MUA) and the mail transfer agent (MTA) are in charge of dispatching the requests generated by the user. Between these two applications, SpamHunting captures all the incoming messages (using POP3 protocol) in order to identify, tag and filter spam.

<Figure 2 goes here>

Whenever SpamHunting receives a new e-mail, the system evolves through the four steps depicted in the lower part of Figure 2 as shadowed rectangles. Initially the system identifies those e-mails that best represent the new incoming message (left upper quadrant in Figure 2), only taking into account the set of messages with the highest number of terms in common. Each one of the previous selected messages contributes with one vote to the final class (left bottom quadrant in Figure 2). The revision stage is only carried out when the system proposes an e-mail as spam. For this purpose, SpamHunting uses previous encoded header meta-rules (right bottom quadrant in Figure 2). Every time the user checks his mailbox and provides feedback to the system about a previous e-mail classification, SpamHunting stores a new instance (or modifies an existing one) in the e-mail base for future use (right upper quadrant in Figure 2).

The retrieval stage is carried out using our Enhanced Instance Retrieval Network model. The EIRN network facilitates the indexation of instances and the selection of those that are most similar to the instance-message. The reuse of similar e-mails is carried out by means of the utilization of a unanimous voting mechanism (as proposed in the work of [34]), which generates an initial solution by creating a model with the retrieved instances. In the revision stage the system employs general knowledge in the form of meta-rules that are extracted from the e-mail headers. Finally, the retain (learning) stage is carried out whenever the system classifies an incoming e-mail, updating the knowledge structure of the whole system. The proposed system also takes into account the feedback of the user when it receives an incorrectly classified e-mail (dotted arrow from the user computer to the e-mail base in Figure 2).

The rest of this section is structured as follows: Subsection 3.1 details with the main aspects of defining a measure to evaluate the relevance of a term within a message. Subsection 3.2 introduces our model representation for indexing and retrieval of similar e-mails. Finally, Subsection 3.3 comments on the process of classifying new incoming e-mails by reusing similar instances and details the revise and retention stages of the proposed IBR system.

3.1. Selecting Relevant Terms for E-mail Representation

Starting from the pre-processed list of words contained in an e-mail, we are interested in selecting the most relevant terms in this e-mail. Without any other information, the only way of doing this is to base it on the frequency of each word in the message. But, if we have available a set of e-mails (a corpus), we can use information about the underlying distribution of the corpus in relation to the target concept (spam or legitimate) to modulate the relevance of each word inside a specific message.

Therefore, we are interested in defining a criterion about the relevance of each term, T_i, which appears in a specific e-mail, e, of a corpus $\boldsymbol{K}$. In order to define this measure the following reasoning is carried out. First, the probability that the e-mail e is a spam message can be expressed as:

$$p(s \mid e) = \sum_{t_i \in e} p(s \mid T_i, e) p(T_i \mid e) \tag{1}$$

The expression $p(T_i \mid e)$ is known, given the e-mail e. Although the expression $p(s \mid T_i, e)$ is unknown, it can be estimated by the probability $p(s \mid T_i, \boldsymbol{K})$. That is, it can be approximated by the probability that an e-mail in the corpus $\boldsymbol{K}$ is a spam message if the term T_i is present in that e-mail. Therefore, Expression (1) can be approximated by:

$$p(s \mid e) \approx \sum_{T_i \in e} p(s \mid T_i, \boldsymbol{K}) p(T_i \mid e) \tag{2}$$

After this, and applying the Bayes' rule, the probability $p(s \mid e)$ can be expressed as:

$$p(s \mid e) \approx \sum_{T_i \in e} \frac{p(T_i \mid s, \boldsymbol{K}) p(s \mid \boldsymbol{K})}{p(T_i \mid \boldsymbol{K})} p(T_i \mid e) = p(s \mid \boldsymbol{K}) \sum_{T_i \in e} \frac{p(T_i \mid s, \boldsymbol{K}) p(T_i \mid e)}{p(T_i \mid \boldsymbol{K})} \tag{3}$$

Secondly, the probability that an e-mail is legitimate given the e-mail, $p(l \mid e)$, can be determined in a similar way:

$$p(l \mid e) \approx \sum_{T_i \in e} \frac{p(T_i \mid l, \boldsymbol{K}) p(l \mid \boldsymbol{K})}{p(T_i \mid \boldsymbol{K})} p(T_i \mid l) = p(l \mid \boldsymbol{K}) \sum_{T_i \in e} \frac{p(T_i \mid l, \boldsymbol{K}) p(T_i \mid e)}{p(T_i \mid \boldsymbol{K})} \tag{4}$$

Moreover, we are interested in truly discriminating between spam terms and legitimate terms (one term may have approximately equal probability to appear in a spam e-mail as it does in a legitimate e-mail). Therefore, the relevance measure of a term would be able to stress one term which is probably only in spam messages or only in legitimate messages, but is not equally probable in both kinds of e-mail, simultaneously. This fact can be modelled by means of the difference between the Expressions (3) and (4), and each term of the sum can be interpreted as a measure of the contribution of each term in the final result, namely, a measure of the relevance of each term. Moreover, if we are not interested in the sign of the contribution (positive if the term helps to classify an e-mail as spam or negative if it helps to classify one as legitimate), the relevance of each term of the e-mail can be defined as follows:

$$r(T_i, e) = \left\{ \frac{\left| p(s \mid \boldsymbol{K}) p(T_i \mid s, \boldsymbol{K}) - p(l \mid \boldsymbol{K}) p(T_i \mid \boldsymbol{K}) \right|}{p(T_i \mid \boldsymbol{K})} \right\} p(T_i \mid e) \tag{5}$$

The relevance term $r(T_i, e)$ tries to conjugate the local and global relevance of the term T_i. The first factor in $r(T_i, e)$ depends on the whole corpus $\boldsymbol{K}$ and expresses the utility of the term T_i in order to discriminate among spam or legitimate e-mails and therefore it evaluates the global relevance of T_i. The second factor in $r(T_i, e)$ only depends on the specific e-mail which is being processed and therefore it can be viewed as a measure of the local relevance of T_i. As consequence of this definition, firstly, the relevance of a term T_i which appears in two different e-mails only depends on the local relevance (since the first factor of Expression (5) will be the same). Secondly, the relative relevance of two terms T_i and T_j which appear in a specific e-mail e, not only depends on the local information, but also depends on the global information given by the first factor of Expression (5), which will be probably different for both terms. The last fact is relevant because we are interested in ordering (by relative relevance) different terms in a specific e-mail in order to select the most relevant ones. Finally, this formulation can be used to select the most relevant terms in two ways: (*i*) a fixed number of terms ordered with respect to $r(T_i \mid e)$ or (*ii*) a variable number of terms depending on a fixed percentage of the whole sum of individual relevancies (if the terms of an e-mail e are ordered descending by $|r(T_i, e)|$ and R is the sum of $|r(T_i, e)|$ over all the terms T_i belonging to e, then fixed a percentage α, the first k_α terms, whose partial sum of relevancies exceed the quantity of αR, will be selected as the most relevant terms). In the successive experimentation only the first way will be used for the evaluation of the system.

3.2. Instance Indexing with the EIRN model

Based on the previous formulation for selecting relevant terms of each message in a corpus $\boldsymbol{K}$, we present here our EIRN model for efficient as well as flexible instance indexing and retrieval. Our EIRN memory structure is inspired by the basic architecture and concepts of CRN networks [29], but without taking into account the similarly weighted arcs that connect information entity nodes among them.

Based on the CRNs case indexing properties, our model defines two measurements: (*i*) *Term Confidence* and (*ii*) *E-mail Confidence* used for maintaining as much information as possible concerning existing data (terms and e-mails). Figure 3 depicts an example of our EIRN model to e-mail retrieval. The EIRN network used in this work is characterized by a two-dimensional space, where the terms (cells) are connected and organized according to the probability of representing spam and legitimate messages. Each cell in the network is associated with a term confidence, *tc*, which represents a measure of how much we can trust it to classify an incoming e-mail. The value of *tc* for a given term T_i is given by Equation (6).

$$tc_i = p(T_i \mid s, K) - p(T_i \mid l, K) \qquad (6)$$

where $p(T_i \mid s, \boldsymbol{K})$ and $p(T_i \mid l, \boldsymbol{K})$ stand for the probability of the term T_i belonging to spam and legitimate e-mails, respectively.

<Figure 3 goes here>

The basic learning process in the EIRN network consists in topology modification and term confidence adaptations. Based on a corpus $\boldsymbol{K}$, of m^k training e-mails or training instances, learning in an EIRN network is carried out presenting all training e-mails, $\boldsymbol{K}$, in a sequential fashion. For each training instance presentation, the network performs a so-called *learning cycle*, which may result in term confidence adaptation and topology modification. Figure 3 clarifies this situation showing those cells with closest values for $p(T_i \mid l, \boldsymbol{K})$ and $p(T_i \mid s, \boldsymbol{K})$ parameters located in nearby points.

In the first step of each learning cycle, the relevant terms, *rt*, of the actual input instance e_j, are linked with the terms present in the network, adding new terms to the model if necessary. Each new connection is weighted up with a relevant value, rv_i, which represents the importance of this term to the actual instance. The value of rv_i depends on the relevant terms (rt_j) of the actual input instance e_j and the actual term T_i. rv_i is calculated using Equation (7).

$$rv_i = \frac{w_k}{2^{i-1}} \qquad (7)$$

where T_i is the actual term and w_k is a constant given by Equation (8).

$$w_k = \frac{2^{rt_j - 1}}{2^{rt_j} - 1} \qquad (8)$$

The second step consists of the adaptation of the term confidence belonging to those terms affected in the previous step and the calculation of the actual e-mail confidence, *ec*. The parameter *ec* represents a measure of the message coherence represented by its relevant terms and aids in the identification of rare e-mails. The value of *ec* for a given pair $\langle e_j, c_j \rangle$ is calculated using Equation (9).

$$ec_j = \frac{\sum_{i=1}^{rt_j} p(T_i \mid c_j, K) rv_i}{rt_j} \qquad (9)$$

where c_j represents the actual class of the e-mail e_j, rt_j stands for the number of relevant terms for e_j, $p(T_i \mid c_j, K)$ represents the probability of the term T_i belonging to a message with the same class as e_j (c_j) and rv_i is calculated using Equation (7).

3.3. E-mail Classification by Reusing Similar Instances. Retrieval, Revising and Learning

Every time a new e-mail arrives at the user mailbox, the EIRN network obtains a set *M*' composed of the instance-messages most similar to the target e-mail, *e*'. In this sense, we can conceive the EIRN memory structure as a dynamic *k-nn* mechanism able to retrieve different numbers of neighbors depending on the selected terms belonging the target e-mail, *e*'. This is done by selecting the relevant terms of the incoming message as described in Subsection 3.1 and projecting them into the network term space (see Figure 3). To perform this selection stage, the system accomplishes two sequential steps: (*i*) calculating the distance between *e'* and the set of messages that shares the greatest number of common terms (*cf*') and (*ii*) selecting those e-mails with a similarity value greater than the mean average value.

In order to calculate the similarity between two e-mails given a set of shared relevant terms, we use a weighted distance metric that takes into account the relevance of each common term. The underlying idea is to weigh those terms that are more relevant to the target email *e*', using the position that occupies each of them in the arrows coming from the target e-mail to the memory structure in Figure 3. The value of the distance between the target e-mail *e*' and a given e-mail e_j is calculated using Equation (10).

$$D(e',e_j) = \sum_{i=1}^{cf'} d(e_i^{'}, e_{ji}) rv_i \qquad (10)$$

where *cf*' is the number of common terms between *M*' and *e*', rv_i represents the importance of each term to the target e-mail *e*' and $d(e_i^{'}, e_{ji})$ measures the distance between the position assigned to the actual common term *i* in the two e-mails, calculated as the difference between the situation of this term in the arrows coming from the target e-mail *e*' and a given e-mail e_j to the memory structure in Figure 3.

Given the distance between two e-mails, the similarity is obtained by means of Equation (11), where the e-mail coherence is used to consider those messages which are most consistent with the instance base.

$$S(e',e_j) = \frac{1}{D(e',e_j)} ec_j \qquad (11)$$

Every time IBR SpamHunting executes the aforementioned instance retrieval stage by selecting those e-mails with higher values for the similarity with the target e-mail *e'*, the system assigns a class label to the incoming e-mail *e'* based on a unanimous voting algorithm. Each message in *M'* returns one vote and by means of recounting the existing votes, an initial classification is provided by the system.

Previous to the final response of the system, a revision stage is carried out when the assigned class is spam. This re-evaluation is carried out with the goal of guaranteeing the accuracy of the solution proposed [35].

In these situations, SpamHunting uses the knowledge extracted from the message header (see Table 1) in order to generate a final answer. Concretely, the system searches the instance base looking for spam e-mails written in the same language as the new one. Starting from the previous selected messages, the system compares the *from* and the *return path* fields with the incoming e-mail, selecting those messages with almost one feature in common. From the latest collection of e-mails, SpamHunting verifies the existence of messages with the same *number* and *content type* of attachments. If some e-mail exists that fulfills all these requisites, SpamHunting classifies the incoming message as spam, otherwise it labels the e-mail as legitimate. If the incoming e-mail was assigned as legitimate at the adaptation stage, the revision phase does nothing.

Before presenting the proposed classification to the user, the SpamHunting system calculates the consistency of the solution (SC). For this purpose we take into account the term coherence (*tc*) of each term belonging to the target e-mail *e'*. The sign of Expression (12) tells the user whether the assigned class is consistent with the selected terms of the target e-mail or not.

$$SC(e') = \frac{\sum_{i=1}^{rt'} tc_i rv_i}{rt'} \tag{12}$$

where *rt'* stands for the number of relevant terms selected for *e'*, tc_i represents the term coherence value of each term and rv_i is calculated using Equation (7).

Every time SpamHunting revises the proposed solution for an incoming e-mail, a new instance-message containing the instance-descriptor and the solution (assigned class) is constructed and stored in the instance base for future reuse. Whenever this happens, the relevant terms of the new instance-message are selected and projected into the network term space. For those terms, the EIRN network updates their tc_i values and recalculates the $p(T_i \mid l, \boldsymbol{K})$ and $p(T_i \mid s, \boldsymbol{K})$ parameters belonging to the affected nodes. Once the new message has been correctly saved and the EIRN network architecture updated, the system computes the value of the e-mail confidence for the new solved instance-message.

4. System Evaluation

This section introduces our evaluation of the SpamHunting system. We carry out an evaluation of the performance of the system from a classical static point of view, where new messages are simply classified and no update of the model occurs. The experiments carried out were offline evaluations using e-mails collected over an extended period of time.

For this purpose, we use six well-known metrics to evaluate the performance (efficacy) of our IBR SpamHunting system: percentage of correct classifications (%OK), percentage of false positives (%FP), percentage of false negatives (%FN), spam recall, spam precision and total cost ratio (TCR) at three different usage scenarios. Moreover, we introduce a measure of the computational complexity (efficiency) of each algorithm by measuring the time spent for each technique in completing the test bed experiments. Finally, we show how our model generalizes the information existing in the training sets of e-mails indicating the mean value of message and model terms for several configurations of the EIRN network.

4.1. Experimental Setup

A key objective was to evaluate the performance of our instance-based reasoning system over other well-known approaches in the domain of spam filtering. For this reason, we used 10-fold stratified cross-validation [36] in all of our experiments, a technique that increases the confidence of experimental findings when using small datasets. In other words, SpamAssassin corpus was partitioned into 10 parts, with each part maintaining the same ratio of legitimate and spam messages as the entire corpus. Each experiment was repeated 10 times, each time reserving a different part as the testing corpus and using the remaining 9 parts as the training corpus. Previous performance scores were then averaged over the 10 iterations.

Table 3 describes the SpamAssassin corpus (created by Justin Mason of Network Associates, and public available for download at http://www.spamassassin.org/publiccorpus/) employed in our experiments. The selected messages were received from January 2002 up to and including December 2003. We append to the subject of each one of the e-mails, the body of the message as well as the attachments. In order to process the diverse formats of the attached files, we use different techniques in each case taking into account the “content-type” header information.

<Table 3 goes here>

At this point, we only applied stop word removal. The e-mails were not altered to carry out stemming of words. In addition, we do not store each word with its part-of-speech tag and we disregard the discourse information of spoken material.

4.2. Static Evaluation

In order to exhaustively evaluate our IBR SpamHunting system and to verify previously published results [37], we compare our model against five well-known widely referenced techniques in the domain literature. These techniques are Naïve Bayes classifier, Adaboost, SVM and two successful case-based reasoning systems: ECUE [34] and a previously developed CBR system belonging to the same authors [26], which we call *Cunn Odds Rate*.

All these models except *Cunn Odds Rate* use IG [22] to select the most predictive features as it has been shown to be an effective technique in aggressive feature removal in text classification [38]. For our comparisons, we have selected the best performance model of each technique varying between 100 and 2000 features. For *Cunn Odds Rate* model, we have maintained the original technique of selecting 30 words representing spam e-mails plus 30 words representing legitimate messages. The algorithm employed for sorting the vocabulary is based on the odds-ratio described in [26].

<Figure 4 goes here>

Figure 4 shows the percentage of correct classifications (%OK), percentage of false positives (%FP) and percentage of false negatives (%FN) belonging to the six models analyzed. From Figure 4 we can surmise that the model producing a higher percentage of correct answers is SVM working with 2000 terms (features). Nevertheless, the model that presents a better ratio between FP and FN errors is the IBR SpamHunting system, which outperforms the rest of the systems. Table 4 clarifies this situation.

<Table 4 goes here>

As we can see from Table 4 when comparing *Cunn Odds Rate* against the IBR SpamHunting system, our system significantly reduces the mean value of FNs while maintaining the same statistical performance in FPs. The next model, SVM working with 2000 terms, reduces the mean value of FNs but significantly increases the number of FPs.

In order to obtain a deeper insight into the operation of the different models, we calculate the recall (Figure 5a) and precision (Figure 5b) scores for the six techniques analyzed. From Figure 5a (filter effectiveness) it can be seen that the worst model is *Cunn Odds Rate*, while the best result was obtained by the SVM algorithm. The rest of the models are within the same interval of correctly classified spam messages, following the best technique closely and with a high value of spam recall.

<Figure 5 goes here>

From Figure 5b (filter safety) it can be seen that the technique that best classifies spam messages is the IBR SpamHunting system that envelops the EIRN model. In this case, the model with worst precision is Naïve Bayes working with 1000 terms. As in the case of recall, the rest of the models closely follow the best technique.

In general, a high spam recall value indicates low FN error rate, and a high spam precision value implies low FP error rate. These two metrics are straightforward to understand, but do not reflect differential treatment of the two types of errors. The TCR score is introduced for this reason, where higher TCR values indicate better performance of the models [21].

Now, let us assume that FP errors are λ times more costly than FN errors, where λ depends on the usage scenario. Three different usage scenarios are used in our experiments. In the first one, the filter flags messages when it suspects them to be spam, without removing them. In this case, $\lambda = 1$. The second scenario assumes that messages classified as spam are returned to the sender. In this scenario $\lambda = 9$ is considered. That is, mistakenly blocking a legitimate message was taken to be as bad as letting 9 spam messages pass the filter. In the third scenario messages classified as spam are deleted automatically without further processing. Now $\lambda = 999$ is used. Figure 6 shows the results taking into account the TCR score and varying the λ parameter as commented above.

<Figure 6 goes here>

From Figure 6 we can see that when FP errors are assigned the same importance as FN errors (a non realistic point of view for a final user) the best model is the SVM algorithm, following by the IBR SpamHunting and the ECUE systems. As soon as one increases the importance of classifying legitimate mail correctly (considering a FP error to be more costly than a FN error) the situation changes drastically and the IBR SpamHunting System produces much better results. This circumstance is supported by the high precision score obtained by the IBR SpamHunting system shown in Figure 5b and the better ratio between FP and FN errors demonstrated in Table 4 and Figure 4.

Moreover, another relevant factor in the design of on-line anti-spam filters is the speed of the algorithms in both training and run time stages. Regarding the computational complexity of the analyzed models, Table 5 shows the time spent (measured in minutes) in the execution of the 10 fold-cross validation. All the models were created and executed in a 2.8 GHz. Pentium IV processor with hyper-threading technology. The SpamHunting, ECUE and *Cunn Odds Rate* systems have been implemented using Java language (JDK 1.4.x) whereas Naïve Bayes, Adaboost and SVM algorithms have been tested with WEKA 3 [39].

<Table 5 goes here>

From Table 5 we can surmise that the model producing the fastest answer is the IBR SpamHunting system, and from a practical point of view, our filter overcomes the advantages typically granted to the Naïve Bayes classifier.

Once we have compared both efficiency and efficacy of our IBR SpamHunting system against other well-know techniques for anti-spam filtering, it is interesting to show how our EIRN network stores the information that it captures from the training messages. Columns in Table 6 represent the total of terms of frequency selected for each e-mail. Note that each column corresponds to a different EIRN configuration.

<Table 6 goes here>

The first row of Table 6 shows for each EIRN configuration, the mean value for the number of selected terms for a randomly chosen e-mail in the model. The second row of Table 6 indicates the number of different terms indexed by the corresponding EIRN network. For the experiments carried out in this paper, the best performance of the EIRN network was achieved storing only the 30% of the total terms of frequency of each email. This led to indexing 13021 terms of the whole corpus and to representing each e-mail using an average of 16 terms.

5 Conclusions and Further Work

In this paper we have presented a successful implementation of an IBR system for anti-spam filtering. The system employs an Enhanced Instance Retrieval Network for e-mail indexing and retrieval. The reuse of similar messages is carried out by means of a simple unanimous voting mechanism to determine whether the target case is spam or not. The revision stage is only carried out in the case of spam messages where the system employs general knowledge in the form of meta-rules that were extracted from the e-mail headers to assign a final class.

Our thorough investigation examined various performance aspects of several well-known machine learning techniques and documented successful anti-spam filters. For this purpose we studied and used a variety of measurements in our experiments to report performance. In this sense, the preliminary results obtained from a static evaluation of the analyzed models showed how our IBR SpamHunting system obtains a better ratio between FP and FN errors as well as in the precision score.

Another issue in anti-spam filtering is the cost of different misclassification errors in normal operation. In this sense we tested the performance of our model against other well-known classifiers in three different cost scenarios. With the exception of assigning the same importance to FP errors than FN errors (first unrealistic scenario), the SpamHunting system obtained significantly better results. Our experiments also showed that the real-life computational cost of running a SpamHunting system is always lower than other approaches.

As the core of the SpamHunting system is the EIRN network, we investigated how our memory structure stores the information that it captures from the training messages. The experiments carried out in this sense showed how the SpamHunting system attained the better performance storing only the terms belonging to each e-mail and representing 30% of the message frequency.

The results obtained from the experiments carried out are very promising and they back up the idea that instance-based reasoning systems can offer a number of advantages in the spam filtering domain. Spam is a disjoint concept and IBR classification works well in this domain. In addition IBR systems can learn over time simply by updating their memory with new instances of spam or legitimate e-mail. Moreover, it provides seamless learning capabilities without the need for a separate learning process and facilitates extending the learning process over different levels of learning.

Since SpamHunting is a long-life IBR spam filtering software, a key challenge for us in order to improve our obtained successful results is the development of a policy for instance-base maintenance. Instance editing techniques involve reducing an instance-base or training set to a smaller number of instances while endeavouring to maintain or even improve the generalization accuracy of the system. In this sense, we are working on the definition of an evolving sliding window in order to better model continuous updating of the system. Further work in this area will also include the comparison of our SpamHunting system with the more common ensemble approach to handling concept drift.

Acknowledgements

This work has been supported by the Spanish Council for Science and Technology (MEC) in projects TIC2003-07369-C02-02. The authors want to thank the work carried out by the referees for their reviews and comments.

References

[1] J. Postel, On the Junk Mail Problem, RFC 706, Internet Engineering Task Force, (http://www.faqs.org/rfcs/rfc706.html/) (1975).
[2] D. Fallows, Internet Users and Spam: What the Attitudes and Behavior of Internet Users can tell Us about Fighting Spam, In Proc. of the First Conference on Email and Anti-Spam (Mountain View, CA, 2004).
[3] A. Gray and M. Haahr, Personalised Collaborative Spam Filtering, In Proc. of the First Conference on Email and Anti-Spam (Mountain View, CA, 2004).
[4] F. Sebastiani, Machine Learning in Automated Text Categorization, ACM Computing Surveys 34(1) 1–47 (2002).
[5] I. Androutsopoulos, G. Paliouras, V. Karkaletsis, G. Sakkis, C.D. Spyropoulos and P. Stamatopoulos, Learning to Filter Spam E-mail: A Comparison of a Naïve Bayesian and a Memory-Based Approach, In Proc. of the Workshop on Machine Learning and Textual Information Access, 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (Lyon, France, 2000).
[6] J.G. Hidalgo, M.M. López and E.P. Sanz, Combining Text and Heuristics for Cost-Sensitive Spam Filtering, In Proc. of the 4th Computational Natural Language Learning Workshop (Lisbon, Portugal, 2000).
[7] V. Vapnik, The Nature of Statistical Learning Theory, 2nd Ed. Statistics for Engineering and Information Science (New York, US, 1999).
[8] M.G. Kelly, D.J. Hand and N.M. Adams, The Impact of Changing Populations on Classifier Performance, In Proc. of the 5th International Conference on Knowledge Discovery and Data Mining (San Diego, California, US, 1999).
[9] G. Widmer and M. Kubat, Learning in the Presence of Concept Drift and Hidden Contexts, Machine Learning 23(1) 69–101 (1996).
[10]I. Watson, Applying Case-based Reasoning: Techniques for Enterprise Systems (San Mateo, CA, 1997).
[11]D. Aha, Lazy learning, editor (Dordrecht, 1997).
[12]M. Sahami, S. Dumais, D. Heckerman and E. Horvitz, A Bayesian Approach to Filtering Junk E-mail, In Learning for Text Categorization – Papers from the AAAI Workshop (Madison, WI, 1998).
[13]G. John and P. Langley, Estimating Continuous Distributions in Bayesian Classifiers, In Proc. of the 11th Conference on Uncertainty in Artificial Intelligence (Montreal, Quebec, Canada, 1995).
[14]R.E. Schapire and Y. Singer, BoosTexter: A Boosting-Based System for Text Categorization, Machine Learning 39(2/3) 135–168 (2000).
[15]W. Cohen and Y. Singer, Context-Sensitive Learning Methods for Text Categorization, ACM Transactions on Information Systems 17(2) 141-173 (1999).
[16]T. Joachims, A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization, In Proc. of the 14th International Conference on Machine Learning (Nashville, TN, 1997).
[17]N. Cristianini and J. Shawe-Taylor, An Introduction to Support Vector Machines and other Kernel-Based Learning Methods (Cambridge, UK, 2000).
[18]K.M. Ting, Inducing Cost-Sensitive Trees Via Instance Weighting, In Proc. of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery (Nantes, France, 1998).

[19]J.M. Gómez, E. Puertas, F. Carrero and M. De Buenaga, Categorización de Texto Sensible al Coste Para el Filtrado de Contenidos Inapropiados en Internet, Procesamiento del Lenguaje Natural 31 13–20 (2003).
[20]I. Androutsopoulos, J. Koutsias, K.V. Chandrinos, G. Paliouras and C. Spyropoulos, An Evaluation of Naïve Bayesian Anti-Spam Filtering, In Proc. of the Workshop on Machine Learning in the New Information Age, 11th European Conference on Machine Learning (Barcelona, Spain, 2000).
[21]I. Androutsopoulos, G. Paliouras and E. Michelakis, Learning to Filter Unsolicited Commercial E-Mail, Technical Report 2004/2, NCSR "Demokritos". http://www.iit.demokritos.gr/skel/i-config/publications/ (2004).
[22]J.R. Quinlan, C4.5 Programs for Machine Learning (San Mateo, CA, 1993).
[23]H.D. Drucker, D. Wu and V. Vapnik, Support Vector Machines for Spam Categorization, IEEE Transactions on Neural Networks 10(5) 1048–1054 (1999).
[24]J. Friedman, T. Hastie and R. Tibshirani, Additive Logistic Regression: A Statistical View of Boosting, Annals of Statistics 28(2) 337–374 (2000).
[25]G. Sakkis, I. Androutsopoulos, G. Paliouras, V. Karkaletsis, C. Spyropoulos and P. Stamatopoulos, A Memory-Based Approach to Anti-Spam Filtering for Mailing Lists, Information Retrieval 6 (1) 49–73 (2003).
[26]P. Cunningham, N. Nowlan, S.J. Delany and M. Haahr, A Case-Based Approach to Spam Filtering that Can Track Concept Drift, In Proc. of the ICCBR'03 Workshop on Long-Lived CBR Systems (Trondheim, Norway, 2003).
[27]W. Daelemans, Z. Jakub, K. van der Sloot and A. van den Bosch, TiMBL: Tilburg Memory Based Learner, version 2.0, Reference Guide, ILK, Computational Linguistics, Tilburg University, (http://ilk.kub.nl/~ilk/papers/ilk9901.ps.gz) (1999).
[28]S.J. Delany, P. Cunningham and L. Coyle, An Assessment of Case-Based Reasoning for Spam Filtering, In Proc. of Fifteenth Irish Conference on Artificial Intelligence and Cognitive Science (Co. Mayo, Ireland, 2004).
[29]M. Lenz, E. Auriol and M. Manago, Diagnosis and Decision Support, Lecture Notes in Artificial Intelligence 1400 51–90 (1998).
[30]A. Aamodt and E. Plaza, Case-based reasoning: Foundational issues, methodological variations, and system approaches, Artificial Intelligence Communications 7(1) 39–59 (1994).
[31]F. Fdez-Riverola and J.M. Corchado, CBR Based System for Forecasting Red Tides, Knowledge-Based Systems 16(5–6) 321–328 (2003).
[32]F. Fdez-Riverola and J.M. Corchado, FSfRT: Forecasting System for Red Tides, Applied Intelligence 21(3) 251–264 (2004).
[33]E.S. Corchado, J.M. Corchado and J. Aiken, IBR Retrieval Method Based on Topology Preserving Mappings, Journal of Experimental & Theoretical Artificial Intelligence 6 (3) 145–160 (2004).
[34]S.J. Delany, P. Cunningham, A. Tsymbal and L. Coyle, A Case-Based Technique for Tracking Concept Drift in Spam Filtering, In Proc. of the 24th SGAI Int. Conf. on Innovative Techniques and Applications of Artificial Intelligence (Cambridge, UK, 2004).
[35]D. McSherry, Explaining the Pros and Cons of Conclusions in CBR, In Proc. of the 7th European Conference on Case-Based Reasoning (Madrid, Spain, 2004).
[36]R. Kohavi, A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection, In Proc. of the 14th International Joint Conference on Artificial Intelligence (Montreal, Canada, 1995).
[37]J.R. Méndez, E.L. Iglesias, F. Fdez-Riverola, F. Díaz and J.M. Corchado, Analyzing the Impact of Corpus Preprocessing on Anti-Spam Filtering Software, Research on Computing Science 17 129–138 (2005).
[38]Y. Yang and J.O. Pedersen, A Comparative Study on Feature Selection in Text Categorization, In Proc. of the Fourteenth International Conference on Machine Learning (San Francisco, CA, 1997).
[39]I.H. Witten and E. Frank, Data Mining: Practical machine learning tools and techniques, 2nd Edition (Morgan Kaufmann, San Francisco, 2005).

Table 1. Representation of header features stored in the case of SpamHunting

Variable	Type	Description
From	String	Source mailbox
Return Path	String	Indicates the address that the message will be re-turned to if one chose to reply
Date	Date	Date in which the message was sent
Language	String	Particular tongue of the message
Attached Files	Integer	Indicates the number of attached files
Content Type	String	MIME type

Table 2. Information features belonging to the subject, body and attachments of an e-mail

Variable	Type	Description
Relevant Terms	Integer	Number of selected features to cluster the message
Total Terms	Integer	Number of features contained in the message
Frequency-Term Descriptor	Array of feature-frequency pairs	Storing for each feature a measure of their frequency in the message

Table 3. Description of the SpamAssassin corpora of e-mails

Year	Legitimate Messages	Spam Messages	L:S Ratio	Total Messages
2002	2801 (84.9%)	498 (15.1%)	5.62	3299
2003	4150 (68.8%)	1883 (31.2%)	2.20	6033

Table 4. Mean value of false positives and false negatives with 10 fold-cross validation

	Cunn Odds Rate [60]	Spam Hunt-ing	SVM [2000]	ECUE [700]	Adaboost [700]	NaïveBayes [1000]
False Positives	0.6	1.6	5.7	7.9	12.3	60.9
False Negatives	172.3	32.9	6.3	27.9	35.6	29.5

Table 5. Time spent (in minutes) by the models analyzed in feature selection and training

	NaïveBayes [1000]	Adaboost [700]	SVM [2000]	Cunn Odds Rate [60]	ECUE [700]	Spam Hunting
Feature selection	4.86	4.30	6.71	2.64	3.56	-
Model training	4.77	243.48	51.62	0.22	2.49	2.71
Total	9.63	247.78	58.33	2.86	6.05	2.71

Table 6. Mean value of message and model terms for several configurations of the EIRN network

	EIRN 5%	EIRN 10%	EIRN 15%	EIRN 20%	EIRN 25%	EIRN 30%	EIRN 35%	EIRN 40%
Message terms	1.59	3.18	5.93	8.25	11.82	16.04	21.00	26.63
Model terms	2217	3974	6060	8268	10551	13021	15518	17787

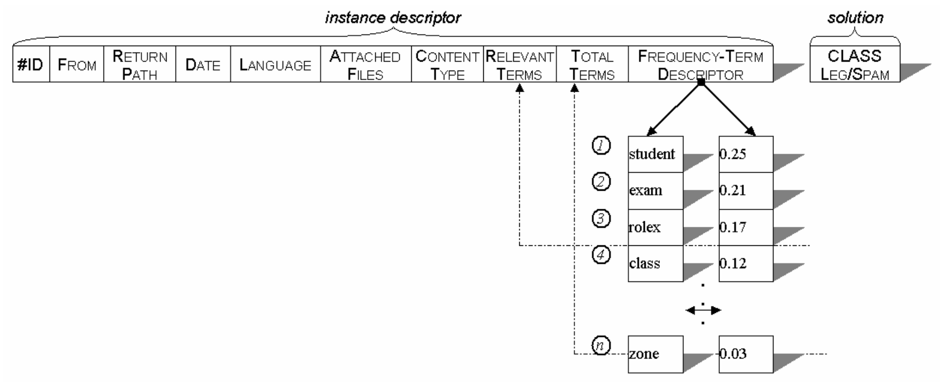


Fig. 1. Instance-message representation for each e-mail in SpamHunting

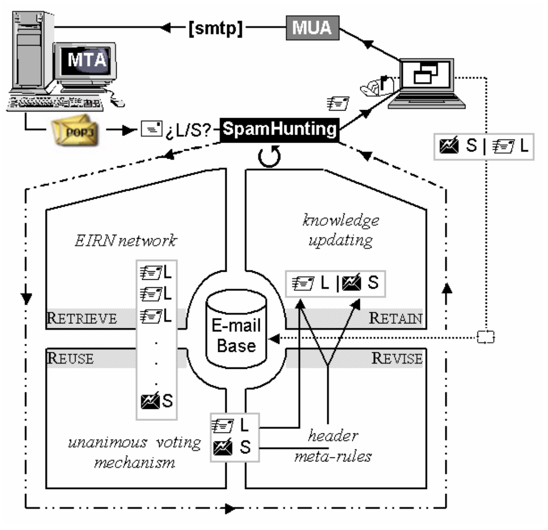


Fig. 2. SpamHunting architecture and model integration

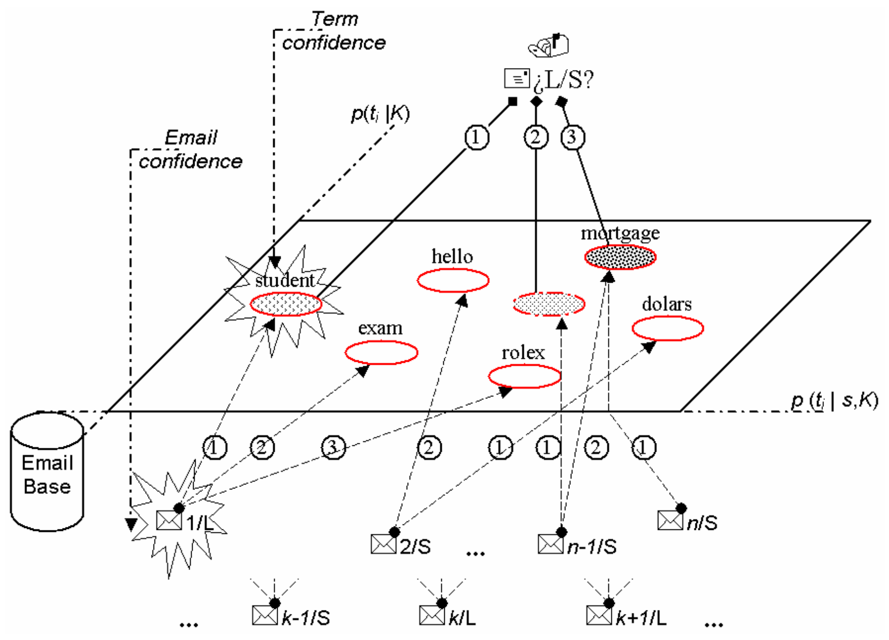


Fig. 3. Enhanced Instance Retrieval Network for e-mail indexing

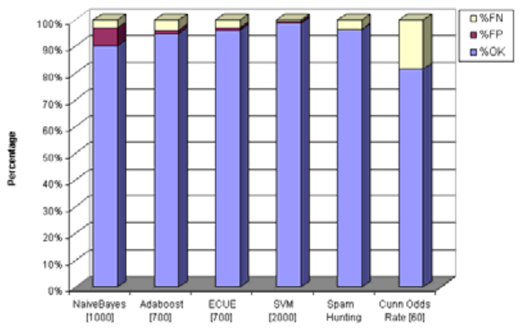
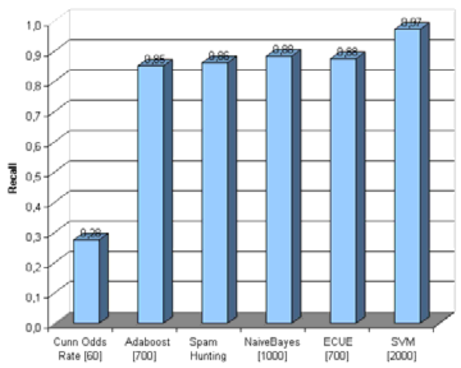
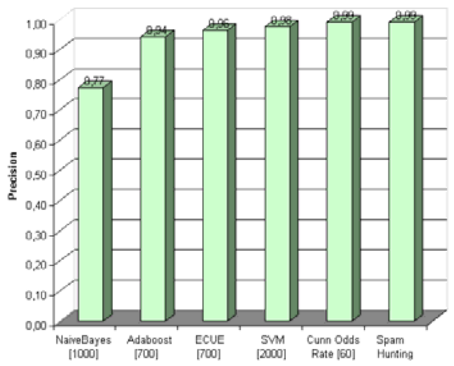


Fig. 4. Percentage of correct classifications, false positives and false negatives from validation over the SpamAssassin corpus

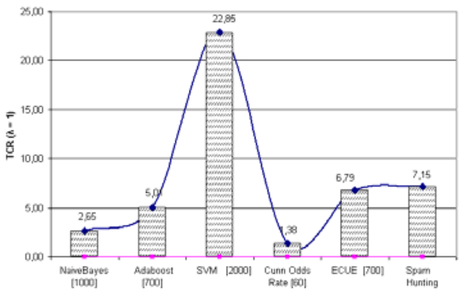


(a) Recall (*filter effectiveness*)

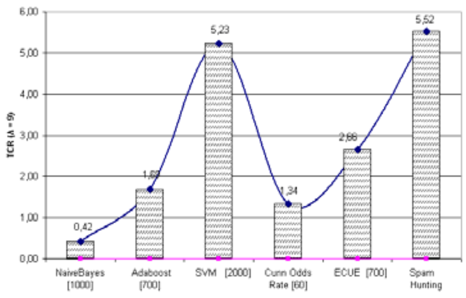


(b) Precision (*filter safety*)

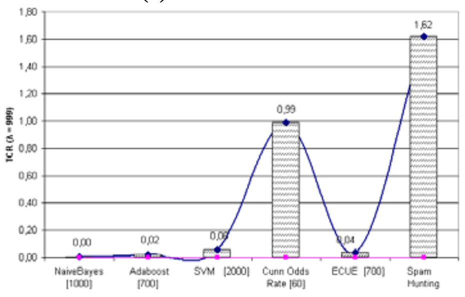
Fig. 5. Recall and precision values for the analyzed models



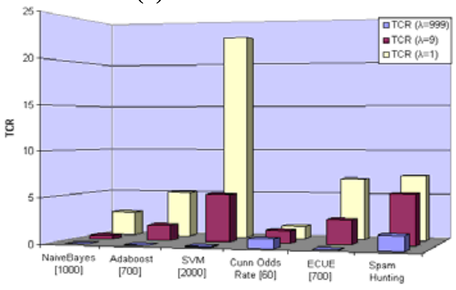
(a) TCR with λ = 1



(b) TCR with λ = 9



(c) TCR with λ = 999



(d) Comparative 3D view of different λ values

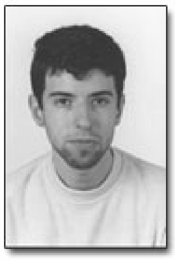
Fig. 6. TCR values for the analyzed models varying the λ parameter over the SpamAssassin corpus

Biographical notes:

F. Fdez-Riverola Ph.D. from the University of Vigo, Spain. He was born in Langen-Hessen, Germany, in 1973. He works as an Associate Professor for the Computer Science Department of the University of Vigo, collaborating as investigator with the Group of Intelligent Computer Systems of the University of Salamanca and the research group SING (Computer Systems of New Generation) belonging to the University of Vigo. Talking about his investigation field, he is cantered in the study of hybrid methods of Artificial Intelligence and their application to real problems, although he has also worked in topics related with the development of communication protocols for wireless networks. He is joint author of several books and book's chapters, as well as author of different articles published by well-known editorials like Springer-Verlag, Ios Press, Kluwer, etc. (http://sing.ei.uvigo.es/).

Eva L. Iglesias received a Ph.D. in Computer Science from the University of Coruña (Spain) in 2003. She works as an Associate Professor for the Department of Computer Science of the University of Vigo from 1998 collaborating as investigator with the SING (Computer Systems of New Generation) research group belonging to this University. She has participated in several investigation projects related with the field of information systems and automatic generation of interfaces for the access to documental databases via Web. It has also directed numerous I+D projects with public and private organizations for the development of cataloguing systems and resource control systems. The investigation work carried out in the environment of these projects has given place to an important number of national and international publications.

F. Díaz Ph. D. from the University of Vigo, Spain. He is currently assistant professor at the Computer Science Department of the University of Valladolid and he participates as researcher in the Group of Intelligent Computer Systems of the University of Salamanca and the Computer Systems of New Generation Group of the University of Vigo. His research is focussed on the Machine Learning field, especially in knowledge acquisition, decision and reasoning under uncertainty, and inductive algorithms. He is author or coauthor of several books, book's chapters and scientific articles, which have been published by well-known editors.



José R. Méndez Ph.D. student in the Computer Science Department at the University of Vigo, Spain. His research work is focused in the study of hybrid methods of Artificial Intelligence such as CBR (Case Based Reasoning) and its application to real problems. During the last years, he has been working as network administrator and developing software in the business world. He collaborates with the research group SING and with the Computer Security group at the Computer Engineering School at University of Vigo. He has instructed many people about computer security over a lot of conferences and courses. He is joint author of several books and book's chapters about computer security and he has been collaborating in the development of JaBaCATs (Java Basic Certificate Authority Tools), an Open Source Tool designed for working with digital certificates.

Juan M. Corchado received a Ph.D. in Computer Science from the University of Salamanca in 1998 and a Ph.D. in Artificial Intelligence from the University of Paisley (UK) in 2000. At present he is Profesor Titular de Universidad at the University of Salamanca (Spain) and Director of the Postgraduate Programs at the Computing and Control Department, previously he was Sub-director of the Escuela Superior de Ingeniería Informática of the University of Vigo (Spain, 1998–00) and Researcher at the University of Paisley (UK, 1995–98). He has been a research collaborator with the Plymouth Marine Laboratory (UK) since 1993, working in the application of artificial intelligence models to oceanographic problems. He has worked on several Artificial Intelligence (AI) Research projects sponsored by Spanish and European public and private Institutions. He is the coauthor of over 100 books, book chapters, journal papers, technical reports, etc. published by organisations such us IEEE, IEE, ACM, AAAI, Kluwer Academic Publishers, SpringerVerlag, Elsevier, Morgan Kaufmann, etc., most of these present practical and theoretical achievements of Hybrid AI and Distributed Systems. (http://gsii.usal.es/).