International Journal on Artificial Intelligence Tools
Vol. 16, No. 6 (2007) 1047–1067

World Scientific
www.worldscientific.com

WORDS VERSUS CHARACTER N-GRAMS FOR ANTI-SPAM FILTERING

IOANNIS KANARIS, KONSTANTINOS KANARIS, IOANNIS HOUVARDAS and EFSTATHIOS STAMATATOS

Department of Information and Communication Systems Eng., University of the Aegean, Karlovassi, Samos – 83200, Greece
stamatatos@aegean.gr

The increasing number of unsolicited e-mail messages (*spam*) reveals the need for the development of reliable anti-spam filters. The vast majority of content-based techniques rely on word-based representation of messages. Such approaches require reliable tokenizers for detecting the token boundaries. As a consequence, a common practice of spammers is to attempt to confuse tokenizers using unexpected punctuation marks or special characters within the message. In this paper we explore an alternative low-level representation based on character n-grams which avoids the use of tokenizers and other language-dependent tools. Based on experiments on two well-known benchmark corpora and a variety of evaluation measures, we show that character n-grams are more reliable features than word-tokens despite the fact that they increase the dimensionality of the problem. Moreover, we propose a method for extracting variable-length n-grams which produces optimal classifiers among the examined models under cost-sensitive evaluation.

Keywords: Anti-spam filtering; machine learning; n-grams.

1. Introduction

Nowadays, e-mail is one of the cheapest and fastest available means of communication. However, a major problem of any internet user is the increasing number of unsolicited commercial e-mail, or *spam*. Spam messages waste both valuable time of the users and important bandwidth of internet connections. Moreover, they are usually associated with annoying material (e.g., pornographic site advertisements) or the distribution of computer viruses. Hence, there is an increasing need for effective anti-spam filters that either automate the detection and removal of spam messages or inform the user of potential spam messages.

Early spam filters were based on blacklists of known spammers and handcrafted rules for detecting typical spam phrases (e.g., 'free pics'). The development of such filters is a time-consuming procedure. Moreover, they can easily be fooled by using forged e-mail addresses or variations of known phrases that is still readable for a human (e.g., f*r*e*e.). Hence, new rules have to be incorporated continuously to maintain the effectiveness of the filter.

Recent advances in applying machine learning techniques to text categorization[1] inspired researchers to develop content-based spam filters. In more detail, a collection of

both known spam and legitimate (non-spam or *ham*) messages is used by a supervised learning algorithm (e.g., decision trees, support vector machines, etc.) to develop a model for automatically classifying new, unseen messages to one of these two categories. That way, it is easy to develop personalized filters suitable for either a specific user or a mailing list moderator.

Spam detection is not a typical text categorization task since it has some intriguing characteristics. In particular, both spam and legitimate messages can cover a variety of topics and genres. In other words, both classes are not homogeneous. Moreover, the length of e-mail messages varies from a couple of text lines to dozens of text lines. In addition, the message may contain grammatical errors and strange abbreviations (sometimes intentionally used by spammers in order to fool anti-spam filters). Therefore, the learning model should be robust in such conditions. Furthermore, besides the content of the body of the e-mail messages, useful information can be found in e-mail address of the sender, attachments etc. Such additional non-textual information can considerably assist the effectiveness of spam filters.[2] Last, but not least, spam detection is a cost sensitive procedure. In the case of a fully-automated anti-spam filter, the cost of characterizing a legitimate message as spam is much higher than letting a few spam messages pass. This fact of crucial importance should be considered in evaluating spam detection approaches.

All supervised learning algorithms require a suitable representation of the messages, usually in the form of an attribute vector. So far, the vast majority of machine learning approaches to spam detection use the *bag of words* representation, that is, each message is considered as a set of words that occur a certain number of times.[2,3,4,5] Putting it another way, the context information for a word is not taken into account. The word-based text representations require language-dependent tools, such as a tokenizer (to split the message into tokens) and usually a lemmatizer plus a list of stop words (to reduce the dimensionality of the problem). A common practice of spammers is to attempt to confuse tokenizers, using structures such as 'f.r.e.e.', 'f-r-e-e', 'f r e e', etc. Moreover, there is no effective lemmatizers available for any natural language, especially for morphologically rich languages. On the other hand, word n-grams, i.e., contiguous sequences of n words, have also been examined.[6] Such approaches attempt to take advantage of contextual phrasal information (e.g., 'buy now'), that distinguish spam from legitimate messages. However, word n-grams considerably increase the dimensionality of the problem and the results so far are not encouraging.

In this paper, we focus on a different but simple text representation. In particular, each message is considered as a *bag of character n-grams*, that is, strings of length n. For example, the character 4-grams of the beginning of this paragraph would be[a]: |In_t|, |n_th|, |_thi|, |this|, |his_|, |is_p|, |s_pa|, |_pap|, |pape|, |aper|, etc. Character n-grams are able to capture information on various levels: lexical (|the_|, |free|), word-class (|ed_|, |ing_|), punctuation mark usage (|!!!|, |f.r.|), etc. In addition, they are robust to

[a] We use '|' and '_' to denote n-gram boundaries and a single space character, respectively.

grammatical errors (e.g., the word-tokens 'assignment' and 'asignment' share the majority of character *n*-grams) and strange usage of abbreviations, punctuation marks etc. The bag of character *n*-grams representation is language-independent and does not require any text preprocessing (tokenizer, lemmatizer, or other 'deep' NLP tools). It has already been used in several tasks including language identification,[7] authorship attribution,[8] and topic-based text categorization[9] with remarkable results in comparison to word-based representations. A model based on character n-grams is difficult to be semantically interpreted. However, in tasks such as anti-spam filtering, this should be viewed as an advantage since it will be difficult for spammers to find tricks to fool the filter.

An important characteristic of the character-level *n*-grams is that they avoid (at least to a great extent) the problem of sparse data that arises when using word-level *n*-grams. That is, there is much less character combinations than word combinations, therefore, less *n*-grams will have zero frequency. On the other hand, the proposed representation still produces a considerably larger feature set in comparison with traditional bag of words representations. Therefore, learning algorithms able to deal with high dimensional spaces should be used. *Support Vector Machines* (SVM) is a supervised learning algorithm based on the structural risk minimization principle.[10] One of the most remarkable properties of SVMs is that their learning ability is independent of the feature space dimensionality, because they measure the complexity of the hypotheses based on the margin with which they separate the data, instead of the features. The application of SVMs to text categorization tasks[11] has shown the effectiveness of this approach when dealing with high dimensional data and sparse data.

In this paper, we compare character *n*-gram representations with traditional word-based representations in the framework of content-based anti-spam filtering. No extra information coming from, e-mail address of the sender, attachments etc. is taken into account. Experiments on two publicly available corpora using a variety of cost-sensitive evaluation measures provide strong evidence that character *n*-gram representations produce more effective models in comparison with the word-based representations. Moreover, we propose a method for extracting variable-length character *n*-grams and show that this representation produces optimal classifiers among the examined models when considering a cost-sensitive evaluation.

The rest of this paper is organized as follows: Section 2 includes related work on content-based anti-spam filtering. Section 3 describes the character *n*-gram representation while Section 4 comprises the method for variable-length character *n*-gram selection. Section 5 gives an overview of the evaluation measures we used and Section 6 describes the corpora and the performed experiments. Finally, section 7 summarizes the conclusions drawn from this study and indicates future work directions.

2. Related Work

Probably the first study employing machine learning methods for anti-spam filtering was published in late 1990s.[2] A Bayesian classifier was trained on manually categorized

legitimate and spam messages and its performance on unseen cases was remarkable. Since then, several machine learning algorithms have been tested on this task, including boosting decision trees and support vector machines,[5] memory-based algorithms,[4] and ensembles of classifiers based on stacking.[12]

On the other hand, a number of text representations have been proposed dealing mainly with word tokens and inspired from information retrieval research. One common method is to use binary attributes corresponding to word occurrence.[2,4] Alternative methods include word (term) frequencies,[6] *tf-idf*,[5] and word-position-based attributes.[13] The dimensionality of the resulting attribute vectors is usually reduced by removing attributes that correspond to words occurring only a few times. Recent work[13] has showed that the removal of the most frequent words (like 'and', 'to' etc.) considerably improves the classification accuracy. Another common practice is to use a lemmatizer[3] for converting each word-type to its lemma (e.g., 'copies' becomes 'copy'). Naturally, the performance of the lemmatizer affects the accuracy of the filter and makes the method language-dependent. Finally, the dimensionality of the attribute vector can be further reduced by applying a feature selection method[14] that ranks the attributes according to their significance in distinguishing among the two classes. Only a predefined number of top ranked attributes are, then, used in the learning model.

In addition, word n-grams have also been proposed[6,13] but, so far, the results are not encouraging. Although such a representation captures phrasal information, sometimes particularly crucial, the dimensionality of the problem increases significantly. Moreover, the sparse data problem arises since there are many word combinations with low frequency of occurrence. Recently, language modeling techniques have also been tested for anti-spam filtering with promising resuts.[15,16]

In, 2005, the Text REtrieval Conference (TREC) has expanded its tracks with the addition of the spam track aiming at providing a standard evaluation of anti-spam filtering approaches. To this end, a collection of evaluation corpora,[17] both public and private, was compiled and a methodology for filter evaluation was developed.[18] Notably, the TREC spam track focused on the ability of the filter to evolve and improve its performance with use.

Another interesting competition of anti-spam filters was organized in 2006. In more detail, the discovery challenge, in the framework of the ECML-PKDD conference, focused on the development of personalized anti-spam filters for the mailboxes of specific users based on a training set from publicly available sources.[19] The messages of each user are available during the training phase but as unlabeled data. Most approaches, then, focused on combining supervised and unsupervised learning to deal with this combination of labeled and unlabeled training data.[20,21]

A few recent studies attempt to utilize a character-level representation of e-mail messages. In Ref. 22 a suffix-tree approach is described which outperforms a traditional Bayesian classifier that is based on a bag of words representation. On the other hand, a representation based on the combination of character 2-grams and 3-grams is proposed in Ref. 23. However, preliminary results in an e-mail categorization task (where many

message classes are available) show that approaches based on word-based representations perform slightly better. Finally, two participant groups in the TREC 2005 spam track were based on character sequences. The first, extracts profiles of spam and legitimate messages using the most frequent character 4-grams or 5-grams and calculates the dissimilarity between the profile of a new message and these profiles.[24] The other, one of the best performing approaches, used compression models working on the character level.[25] Further experiments using the compression models demonstrated competitive performance on other corpora as well.[26]

Research in spam detection was considerably assisted by publicly available benchmark corpora, so that different approaches to be evaluated on the same testing ground. An important issue is that legitimate messages usually contain personal information of the users which should not become publicly available. One solution is to collect legitimate messages from mailing lists, (e.g., *Ling-Spam*[b]) or directly by users willing to donate them (e.g., *SpamAssasin*[c]). Another solution is to attempt to obscure information about senders and receivers,[15] or encode the words of the body of the messages so that to become unreadable.[6] Recently, the publicly-available Enron corpus has been used as a source of legitimate messages.[27]

3. The Bag-of-Character N-grams Approach

First, for a given n, we extract the L most frequent character n-grams of the training corpus. Let $<g_1, g_2, \ldots, g_L>$ be the ordered list (in decreasing frequency) of the most frequent n-grams of the training corpus. Then, each message is represented as a vector of length L $<x_1, x_2, \ldots, x_L>$, where x_i depends on g_i. In more detail, we examine two representations:

- Binary: The value of x_i may be 1 (if g_i is included at least once in the message) or 0 (if g_i is not included in the message).
- Term Frequency (TF): The value of x_i corresponds to the frequency of occurrence (normalized by the message length) of g_i in the message.

The produced vectors can be arbitrarily long. On one hand, if L is chosen too short, the messages are not represented adequately. On the other hand, if L is chosen too long the dimensionality of the problem increases significantly. In the experiments described in the following sections, all the character n-grams that appear more than 3 times in the training corpus are taken into account. A feature selection method can then be applied to the resulting vectors, so that only the most significant attributes contribute to the classification model. A feature selection method that proved to be quite effective for text categorization tasks is *information gain*.[14] The information gain of a feature x_i is defined as an expected reduction in entropy by taking x_i as given:

$$IG(C, x_i) = H(C) - H(C| x_i) \tag{1}$$

[b] Available at: http://www.aueb.gr/users/ion/data/Ling-Spam_public.tar.gz
[c] Available at: http://spamassassin.apache.org/publiccorpus/

where C denotes the class of the message ($C \in$ {spam, legitimate}) and $H(C)$ is the entropy of C. In other words, $IG(C, x_i)$ is the information gained by knowing x_i. Information gain helps us to sort the features according to their significance in distinguishing between spam and legitimate messages. Only the first m most significant attributes are, then, taken into account.

The produced vectors (of length m) of the training set are used to train a SVM classifier. The Weka[28] implementation of SVM was used (default parameters were set in all reported experiments).

4. Variable-Length N-gram Selection

The approach described above is able to provide fixed-length character n-grams. In this paper, we also propose a method for extracting variable-length character n-gram features based on an existing approach for extracting multiword terms (i.e., word n-grams of variable length) from texts. The original approach aimed at information retrieval applications.[29] In this study, we slightly modified this approach in order to apply it to character n-grams. Having a big initial set of variable-length n-grams (e.g., composed by equal amounts of fixed-length n-grams), the main idea is to compare each n-gram with similar n-grams (either immediately longer or shorter) and keep the dominant n-grams. All the other n-grams are discarded and they don't contribute to the classification model. To this end, we need a function able to express the significance of a n-gram. We view this function as a 'glue' that sticks the characters together within an n-gram. The higher the glue of a n-gram, the more likely for it to be included in the set of dominant n-grams. For example, the 'glue' of the n-gram |the_| will be higher than the 'glue' of the n-gram |theal.

4.1. *Dominant N-gram extraction*

To extract the dominant character n-grams in a corpus we modified the algorithm *LocalMaxs* introduced in Ref. 29. It is an algorithm that computes local maxima comparing each n-gram with similar (shorter or longer) n-grams. Given that:

- $g(C)$ is the glue of n-gram C, that is the power holding its characters together.
- $ant(C)$ is an antecedent of an n-gram C, that is, a shorter string composed by n-1 consecutive characters of C.
- $succ(C)$ is a successsor of C, that is, a longer string of size n+1, i.e., composed by C and one extra character either on the left or right side of C.

Then, the dominant n-grams are selected according to the following rules:

$$\begin{aligned} &if\,(C.length > 3) \\ &g(C) \geq g(ant(C)) \wedge g(C) > g(succ(C)),\ \forall ant(C), succ(C) \\ &if\,(C.length = 3) \\ &g(C) > g(succ(C)),\ \forall succ(C) \end{aligned} \tag{2}$$

In this study, we only consider 3-grams, 4-grams, and 5-grams as candidate n-grams since they can capture both sub-word and inter-word information and keep the dimensionality of the problem in a reasonable level. Note that, according to the proposed algorithm, 3-grams are only compared with successor n-grams. Moreover, 5-grams are only compared with antecedent n-grams. So, it is expected that the proposed algorithm will favor 3-grams and 5-grams against 4-grams.

4.2. *Representing the glue*

To measure the glue holding the characters of a n-gram together various measures have been proposed, including specific mutual information[30], the φ^2 measure,[31] etc. In this study, we adopt the *Symmetrical Conditional Probability* (SCP) proposed in Ref. 32. The SCP of a bigram |xy| is the product of the conditional probabilities of each given the other:

$$SCP(x,y) = p(x \mid y) \cdot p(y \mid x) = \frac{p(x,y)}{p(x)} \cdot \frac{p(x,y)}{p(y)} = \frac{p(x,y)^2}{p(x) \cdot p(y)} \tag{3}$$

Given a character n-gram $|c_1 \ldots c_n|$, a dispersion point defines two subparts of the n-gram. Hence, a n-gram of length n contains n-1 possible dispersion points (e.g., if * denote a dispersion point, then the 3-gram |the| has two dispersion points: |t*he| and |th*e|). Then, the SCP of the n-gram $|c_1 \ldots c_n|$ given the dispersion point $|c_1 \ldots c_{n-1} * c_n|$ is:

$$SCP((c_1 \ldots c_{n-1}), c_n) = \frac{p(c_1 \ldots c_n)^2}{p(c_1 \ldots c_{n-1}) \cdot p(c_n)} \tag{4}$$

Essentially, this is used to suggest whether the n-gram is more important than the two substrings defined by the dispersion point. The lower the SCP, the less important the initial n-gram. The SCP measure can be easily extended so that to account for any possible dispersion point (since this measure is based on fair dispersion point normalization, will be called *fairSCP*). Hence, the *fairSCP* of the n-gram $|c_1 \ldots c_n|$ is as follows:

$$fairSCP(c_1 \ldots c_n) = \frac{p(c_1 \ldots c_n)^2}{\frac{1}{n-1} \sum_{i=1}^{i=n-1} p(c_1 \ldots c_i) \cdot p(c_{i+1} \ldots c_n)} \tag{5}$$

5. Evaluation Measures

5.1. *Total cost ratio*

Two well known measures from information retrieval community, *recall* and *precision*, can describe in detail the effectiveness of a spam detection approach. In more detail, given that $n_{S \to S}$ is the amount of spam messages correctly recognized, $n_{S \to L}$ is the amount of spam messages incorrectly categorized as legitimate, and $n_{L \to S}$ is the amount of legitimate messages incorrectly classified as spam, then, *spam recall* and *spam precision*

can be defined as follows:

$$\text{Spam Recall} = \frac{n_{S\to S}}{n_{S\to S} + n_{S\to L}} \tag{6}$$

$$\text{Spam Precision} = \frac{n_{S\to S}}{n_{S\to S} + n_{L\to S}} \tag{7}$$

In intuitive terms, spam recall is an indication of filter effectiveness (the higher the recall, the less spam messages pass) while spam precision is an indication of filter safety (the higher the precision, the less legitimate messages blocked).

However, spam detection is a cost sensitive classification task. So, it is much worse to misclassify a legitimate message as spam than vice versa. Therefore, we need an evaluation measure that incorporates an indication of this cost. A cost factor λ is assigned to each legitimate message, that is, each legitimate message is considered as λ messages.[3,4] In other words, if a legitimate message is misclassified, λ errors occur. A cost-sensitive evaluation measure, the *Total Cost Ratio* (TCR) can, then, be defined[3, 4] as follows:

$$\text{TCR} = \frac{n_{S\to S} + n_{S\to L}}{\lambda \cdot n_{L\to S} + n_{S\to L}} \tag{8}$$

The higher the TCR, the better the performance of the approach. In addition, if TCR is lower than 1, then the filter should not be used (the cost of blocking legitimate messages is too high). To be in accordance with previous studies, three cost scenarios were examined:

- Low cost scenario ($\lambda = 1$): This corresponds to an anti-spam filter that lets a message classified as spam to reach the mailbox of the receiver along with a warning that the message is probably spam.
- Medium cost scenario ($\lambda = 9$): This corresponds to an anti-spam filter that blocks (or quarantines) a message classified as spam. The receiver occasionally checks the messages quarantined so far.
- High cost scenario ($\lambda = 999$): This corresponds to a fully-automated filter that deletes a message classified as spam without notifying either the receiver or the sender.

5.2. *ROC graphs*

Receiver Operating Characteristics (ROC) graphs, originated from signal detection theory,[33] are a useful tool for visualizing the performance of classifiers. Since they offer a reliable representation of the classifier properties under imbalanced class distribution and unequal classification error costs, they are especially popular in the machine learning community.[34] An ROC graph depicts relative trade-offs between benefits and costs. In more detail, given that *false positive* (fp) *rate* and *true positive* (tp) *rate* are defined

as follows:

$$\text{fp rate} = \frac{n_{L\to S}}{n_{L\to L} + n_{L\to S}} \tag{9}$$

$$\text{tp rate} = \frac{n_{S\to S}}{n_{S\to S} + n_{S\to L}} \tag{10}$$

the performance of a classifier is depicted in a two-dimensional space in which the false positive rate is plotted on the x axis and the true positive rate is plotted on the y axis. Alternatively, the two axes may correspond to *ham misclassification* (= *fp rate*) and *spam misclassification* (= 1-*tp rate*), respectively.[18] Given that the classifier can assign a probability or score to an instance, a curve is plotted in the ROC space by varying the threshold used to produce binary classification results.

From Eqs. (6) and (10), it is obvious that *recall* and *tp rate* are identical. Notice that they depend only on spam messages. Moreover, as shown in Eq. (9), *fp rate* depends only on legitimate messages. On the other hand, *precision* (see Eq. (7)) depends on both spam and legitimate messages. If the proportion of spam to ham messages changes, the ROC curve of a classifier remains the same while the recall-precession graph is affected drastically. Therefore, an important property of the ROC graphs is that they are insensitive to changes in class distribution.[34] For spam filtering applications, this is a crucial factor since the amount of spam messages that reaches a specific mailbox continuously changes. Another interesting property of ROC graphs is that they are able to produce a single value that represents the expected performance. The most common method is to calculate the area under the ROC curve (AUC). The AUC of a classifier corresponds to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance.[35]

Finally, the operating conditions of a classifier (i.e., different classification error costs) may be translated into iso-performance lines in the ROC space, that is, lines with the same slope.[34] Let λ be the cost of misclassifying a ham message. The slope of the iso-performance lines is defined as:

$$slope = \lambda \frac{n_{L\to L} + n_{L\to S}}{n_{S\to S} + n_{S\to L}} \tag{11}$$

Lines closer to the upper left corner of the ROC space correspond to better classifiers. Given the ROC curves of a set of classifiers, a classifier may be optimal if and only if it lies on the ROC convex hull (ROCCH), that is, a curve connecting the best parts (lying closer to the upper left corner in the ROC space) of the classifiers. If the operating conditions of the classifiers change, the ROCCH remains the same but a different portion of the ROCCH should be examined for identifying the optimal classifier. This portion of ROCCH is indicated by the iso-performance line that corresponds to the new operating conditions. A detailed description of this method for evaluating classifiers is provided by Provost and Fawcett.[34]

6. Experiments

6.1. *Corpora and settings*

In this study we are based on two widely-used corpora to evaluate the usefulness of the character n-grams for spam filtering. The first corpus is Ling-Spam consisting of 2,893 emails, 481 spam messages and 2,412 legitimate messages taken from postings of a mailing list about linguistics. This corpus has a relatively low spam rate (16%) and the legitimate messages are not as heterogeneous as the messages found in the personal inbox of a specific user. However, it has already been widely used in previous studies[3,4,22] and comparison of our results with word-based methods is feasible. Moreover, it provides evidence about the effectiveness of our approach as assistance to mailing list moderators.

The bare version of this corpus was used (no lemmatizing or stop-word removal was performed) so that to be able to extract accurate character n-gram frequencies. Unfortunately, this corpus was already converted to lower case, so it was not possible to explore the significance of upper case characters.

The second corpus is a part of the publicly-available SpamAssasin corpus. The legitimate messages of this corpus were collected from public forums as well as from direct donation of specific users. In more detail, the corpus we used consists of 2,000 ham and 1,500 spam messages collected in 2005. This corpus has been preprocessed in order to remove attachments, headers, and html-tags. Moreover, all tabs, new line characters, and sequences of white spaces were replaced by a single white space. Since it has been constructed by many individual users, the legitimate messages are expected to be more heterogeneous than the messages found in the personal mailbox of a single user. Moreover, since the messages are in their original form, it is possible to explore whether or not case sensitive character n-grams perform better than case-insensitive character n-grams.

In all the experiments described below, a ten-fold cross-validation procedure was followed. That is, the entire corpus was divided into ten equal parts, in each fold a different part is used as test set and the remaining parts as training set. Final results come from averaging the results of each fold. Finally, in all cases, a SVM classifier was used as learning algorithm with default values (linear kernel, C=1). In general, this setting provides a reliable classifier for text categorization tasks and can reveal the differences of the examined representation approaches. Moreover, the focus of this paper is on the message representation scheme rather than fine tuning the parameters of the classification algorithm.

6.2. *Results on Ling-Spam*

Three sets of experiments were performed based on character 3-gram, 4-gram, and 5-gram representations, respectively. In all three cases, both binary and TF attributes were examined. Moreover, different values of the m attributes left after the feature selection procedure were tested (m starts from 250 and then varies from 500 to 4000 by 500). For evaluating the performance of the classifiers we use the recall, precision, and

TCR measures in order to be able to compare it with reported results of previous word-based studies on the same corpus.

The results of the application of our approach to Ling-Spam are shown in Fig. 1. As can be seen, for binary attributes, 4-grams seems to provide the more reliable representation (for m>2000). On the other hand, for TF attributes there is no clear winner. More significantly, binary attributes seem to provide better spam precision results while TF attributes are better in terms of spam recall. In most cases, spam recall was higher than 97% while, at the same time, spam precision was higher than 98%. Moreover, a few thousands of features are required to get these results. This is in contrast to previous word-based approaches that deal with a limited amount (a few hundreds) of attributes.

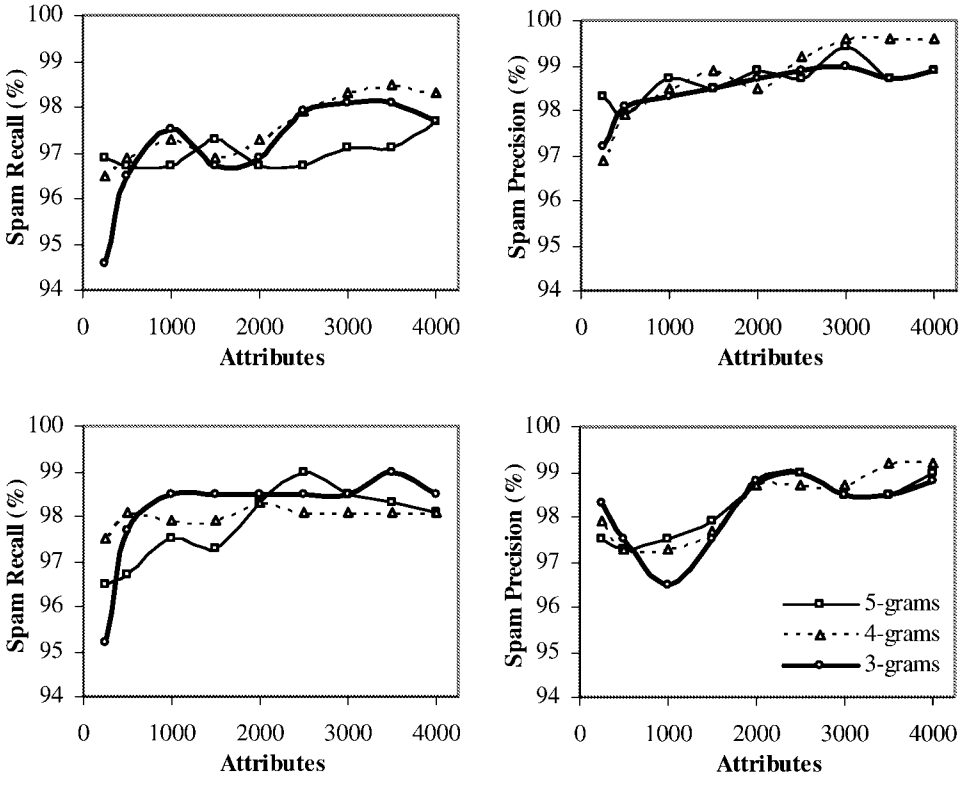

Fig. 1. Spam recall and spam precision of the proposed approach based on character 3-grams, 4-grams, and 5-grams and varying number of attributes on the Ling-Spam corpus. Top: binary attributes. Bottom: TF attributes.

The results of the cost-sensitive evaluation are shown in Fig. 2. In particular, TCR values for 3-grams, 4-grams, and 5-grams are given for varying number of attributes. Results are given for both binary and TF attributes as well as the three evaluation scenarios ($\lambda = 1$, 9, and 999, respectively). As can be seen, in all three scenarios, a representation based on character 4-grams with binary attributes provides the best results.

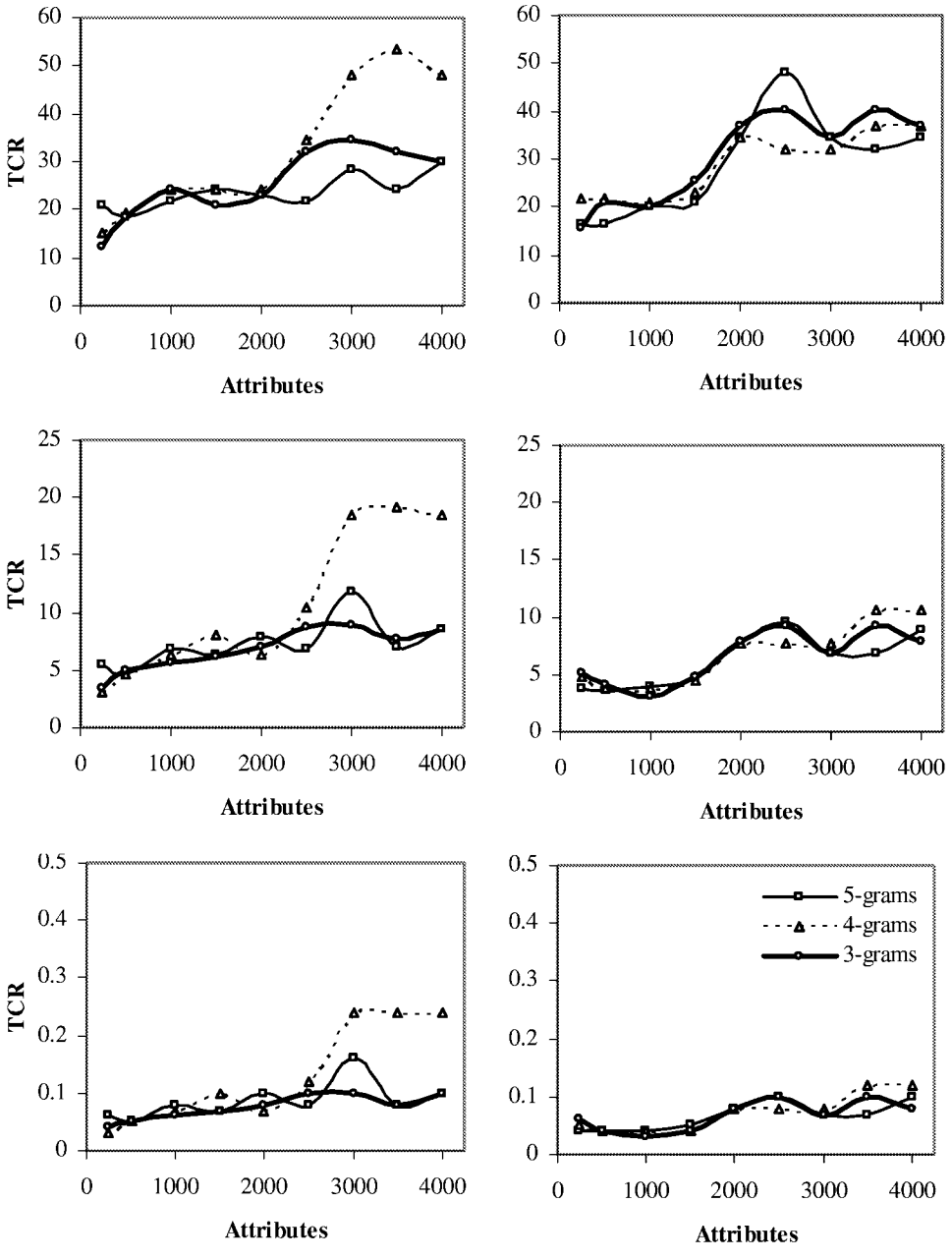

Fig. 2. Results of cost-sensitive evaluation on the Ling-Spam corpus. TCR values for $\lambda = 1$(top), $\lambda = 9$ (middle), and $\lambda = 999$ (bottom) and varying number of attributes and n-gram length on the Ling-Spam corpus. Left column: binary attributes. Right column: TF attributes.

This stands for a relatively high number of attributes (m>2500). For $\lambda = 1$, and $\lambda = 9$ the TCR results are well above 1 indicating the effectiveness of the filter. On the other hand, for $\lambda = 999$, the TCR results are less than 1 indicating that the filter should not be used at all. However, it is difficult for this scenario to be used in practice.

Table 1 shows a comparison of the proposed approach with previously published results on the same corpus in terms of spam recall, spam precision, and TCR values. In more detail, best results achieved by three methods are reported: a Naïve Bayes (NB) classifier,[3] a Memory-Based Learner (MBL),[4] and a Stacked Generalization approach (SG)[12] using word-based features and a Suffix Tree (ST)[22] approach based on character-level information. The number of attributes that correspond to the best results of each method per evaluation scenario is also given. It should be noted that the results for the ST approach are referred to a sub-corpus of Ling-Spam with a proportion of spam to legitimate messages approximately equal to the entire Ling-Spam corpus (200 spam and 1,000 legitimate messages). Moreover, no results were reported for the SG approach based on the high cost scenario.

Table 1. Comparison of cost-sensitive evaluation (λ=1, 9, and 999) of the proposed approach with previously published results on Ling-Spam. Best reported results for spam recall, spam precision, and TCR are given. ST results refer to a sub-corpus of Ling-Spam.

Approach	λ	Attributes	Recall	Precision	TCR
NB	1	100	82.35%	99.02%	5.41
MBL	1	600	88.60%	97.39%	7.81
SG	1	300	89.60%	98.70%	8.60
ST	1	-	97.22%	100%	35.97
Proposed	1	3,500	98.50%	99.60%	52.75
NB	9	100	77.57%	99.45%	3.82
MBL	9	700	81.93%	98.79%	3.64
SG	9	100	84.80%	98.80%	4.08
ST	9	-	98.89%	98.89%	9.01
Proposed	9	3,500	98.50%	99.60%	19.76
NB	999	300	63.67%	100%	2.86
MBL	999	600	59.91%	100%	2.49
ST	999	-	97.78%	100%	45.04
Proposed	999	3,500	98.50%	99.60%	0.25

As concerns the TCR, the proposed approach is by far more effective than word-based approaches for the low and medium cost scenarios. This is due to the fact that it manages to achieve high spam recall while maintaining spam precision on equally-high level. ST is also quite competitive. This provides extra evidence that character-based representations are better able to capture the characteristics of spam messages. On the other hand, the proposed approach failed to produce a TCR value greater than 1 for the high cost scenario. That is because the precision failed to be 100%. It must be underlined that previous studies[3, 4] show that TCR is not stable for the high cost scenario and it is common for TCR to exceed 1 only for very specific settings of the filter.

6.3. *Results on SpamAssasin*

Using the SpamAssasin corpus we are able to evaluate the character n-grams approach based on case sensitive and case insensitive datasets. Three sets of experiments were performed based on character 3-gram, 4-gram, and 5-gram binary representations, respectively. All character n-grams appearing more than 3 times in the corpus constitute the initial feature set. Then, information gain is applied to this initial feature set to reduce the dimensionality. Different values of the m attributes left after the feature selection procedure were tested (m varies from 500 to 4,000 by 500) for each character n-gram category. Additionally, traditional word-based models were also constructed for comparative purposes. First, a bare bag of words approach and, second, a bag of words approach in combination with a lemmatizer and stop words removal were tested using the TMG toolbox.[36] For evaluating the performance of the produced classifiers we use the ROC graphs since they offer an insight view into the properties of the produced filters.

Figure 3 shows the performance of the case sensitive and case insensitive character n-gram-based classifiers in terms of the AUC. As can be seen, case sensitive character n-grams outperform case insensitive character n-grams. Moreover, the 3-gram model is the most effective for this corpus followed by 4-grams and 5-grams. This contrasts the case of Ling-Spam where 4-grams were found to perform better. Recall that SpamAssasin legitimate messages are not homogeneous in topic as Ling-Spam messages. Some long character n-grams taken from very common words of the Ling-Spam corpus (e.g., 'language', 'linguistics', etc) tend to be the most important for identifying ham messages. There is no such keywords in SpamAssasin corpus. Therefore, shorter character n-grams prevail in this corpus.

Figure 4 depicts the performance of the case sensitive models in comparison to the word-based models. Again, 3-grams are the most affective features. On the other hand, word-based models outperform 4-grams and 5-grams. Finally, the use of lemmatizer and stop word removal improves the results for low dimensional datasets ($m < 2{,}500$) while the bare bag of words approach is better for higher dimensionality.

6.4. *Results with variable-length N-grams*

So far, all the experiments were based on fixed-length character n-grams. In order to test the approach proposed in Section 4 for extracting variable-length n-grams we performed the following experiment.

- An initial large feature set consisting of case sensitive character n-grams of variable length is extracted from the training corpus. This feature set includes the L most frequent n-grams for certain values of n. That is, for $L = 1{,}000$, the 1,000 most frequent 3-grams, the 1,000 most frequent 4-grams, and the 1,000 most frequent 5-grams compose the initial feature set.
- The proposed variable-length feature selection method (section 4) is applied to this initial feature set to produce the dominant n-grams.
- The resulting feature set is used to train a SVM classifier.

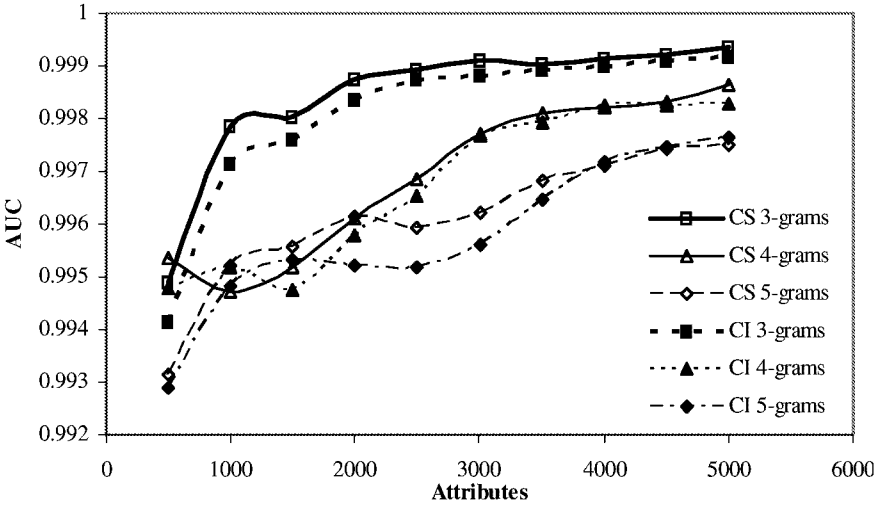

Fig. 3. The performance of the case sensitive (CS) and case insensitive (CI) character n-gram models on the SpamAssasin corpus

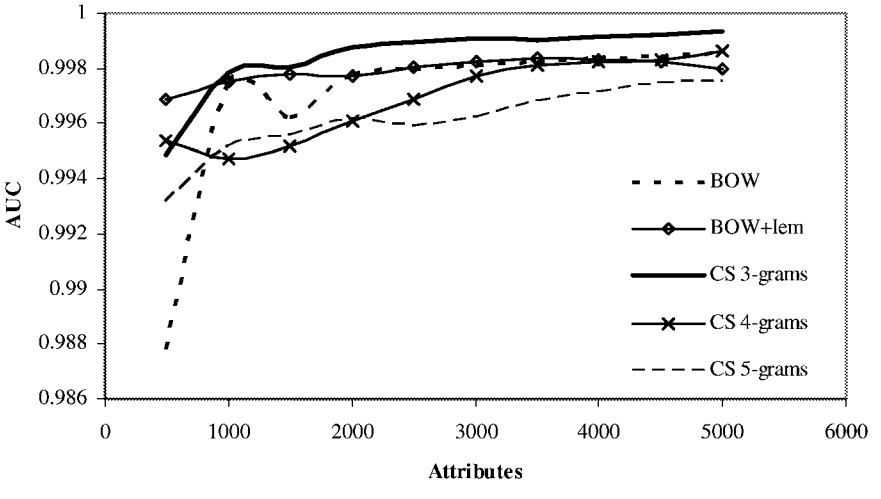

Fig. 4. The performance of the word-based models (simple bag-of-words and bag-of-words with lemmatizer and stop word removal) and the case sensitive (CS) character n-grams models on the SpamAssasin corpus.

This procedure was followed for L ranging from 1,000 to 5,000 with a step of 500. Fig. 5 shows the performance (in terms of AUC) of the variable-length character n-grams in comparison with the case sensitive 3-grams and the word-based approaches. Note that the variable-length model is not able to produce a predefined number of attributes. However, by varying the value of L, it is possible to get roughly as many attributes as the

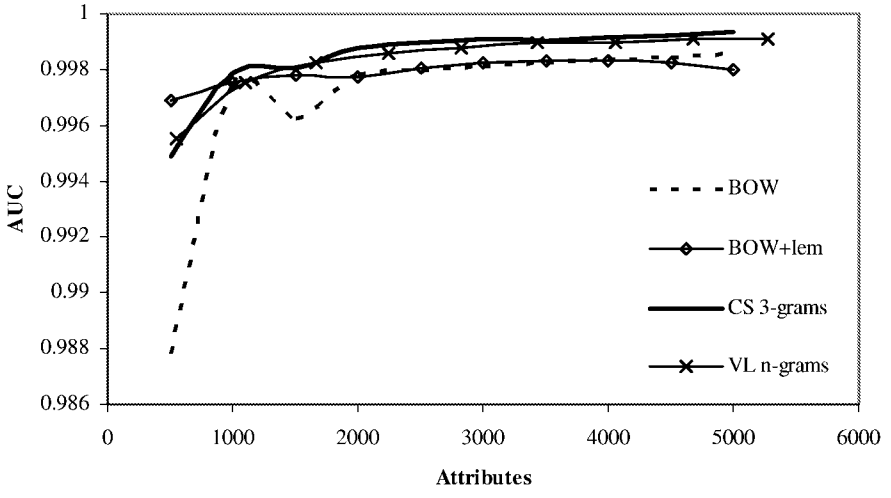

Fig. 5. Performance of the word-based approaches (simple bag-of-words and bag-of-words with lemmatizer and stop word removal), the case sensitive (CS) 3-gram model and the variable-length (VL) n-gram model on the SpamAssasin corpus.

other models produce. The variable-length n-gram approach outperforms the word-based approaches and it is quite competitive with the case sensitive 3-gram model. However, the latter is still the best performing model in most of the cases.

Note that the AUC measure is an overall indication of the effectiveness of the classifier. A closer look at the ROC graphs reveals that the variable-length n-gram model is optimal when considering a cost-sensitive evaluation. In particular, Fig. 6 shows the ROC graphs for the case sensitive 3-gram and the word-based models (m = 5,000) as well as the variable-length n-gram model (L = 4,500 producing 5,277 selected attributes). The ROCCH (connecting the best parts of the examined classifiers) is also depicted. As can be seen, the 3-gram model and the variable-length n-gram model dominate the ROCCH. The former is better especially for both very low and very high (not shown in the figure) levels of fp rate while the latter is better for the middle fp rate level. On the other hand, word-based models are outperformed. Recall from section 5.2 that according to the operating conditions of the classifier, a specific part of the ROCCH corresponds to the optimal classifier. In the framework of a cost-sensitive evaluation, the optimal classifier is identified based on the iso-performance lines (see Eq. (11)). Figure 7 shows the best iso-performance lines for different costs of misclassifying a legitimate message as spam: λ = 1, 9, and 999. As can be seen, different misclassification costs correspond to different slopes of the iso-perofmance lines. The point of the ROCCH intersected by the iso-performance line indicates the optimal classifier for each cost-sensitive evaluation scenario. In all three cases, the part of ROCCH taken up by the variable-length n-gram model corresponds to the optimal classifier. Therefore, despite the fact that the 3-gram model is more effective in general, as indicated by the AUC measure, the variable-length n-gram model is optimal in a cost sensitive evaluation.

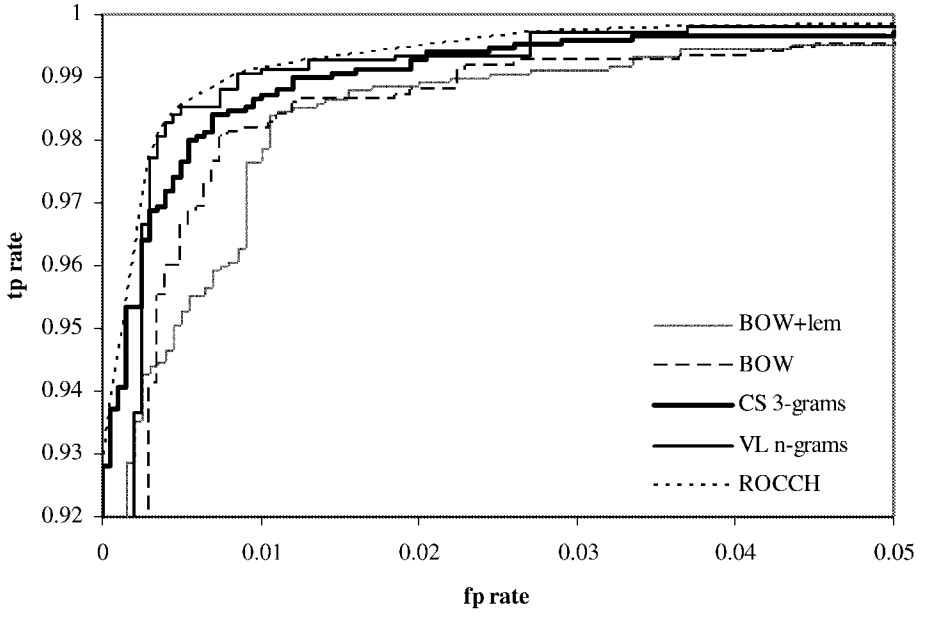

Fig. 6. ROC graphs for the word-based models, the case sensitive 3-gram model, and the variable-length n-gram model. The ROC convex hull indicating optimal performance among the examined models is also depicted.

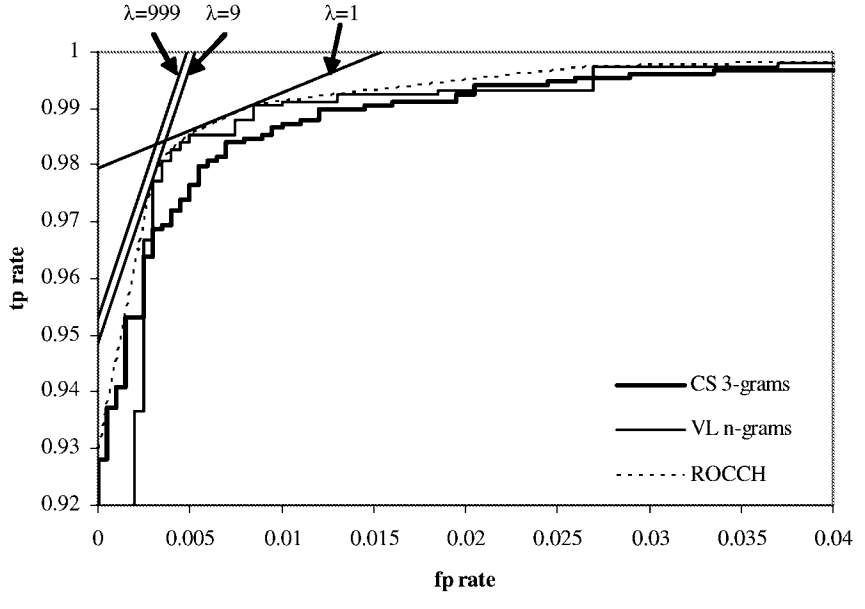

Fig. 7. ROC graphs for the case sensitive 3-gram model and the variable-length n-gram model as well as the ROC convex hull. Iso-performance lines indicating the optimal classifier for different cost values ($\lambda = 1$, 9, and 999) of misclassifying a legitimate message are also depicted.

Note that the attributes of the case sensitive 3-gram model of the previous experiment were selected using a 22,673 initial feature set (i.e., all character 3-grams that appear more than 3 times in the corpus). On the other hand, the variable-length n-gram attributes were selected using a 13,500 initial feature set (since $L = 4{,}500$). Figure 8 shows the composition of this variable-length n-gram model produced for $L = 4{,}500$ (resulting 5,277 attributes). The prevailing character n-gram category is 3-grams, followed by 5-grams. However, the contribution of the longer n-grams seems to be of crucial importance for constructing more reliable cost sensitive filters. Interestingly, the common 3-grams of the variable-length n-gram set and the case sensitive 3-gram set are only 926. That is, only 18% of the case sensitive 3-grams are included in the variable-length n-gram set. This fact means that many 3-grams not selected by information gain are now included in the feature set and are helpful for producing more effective classifiers.

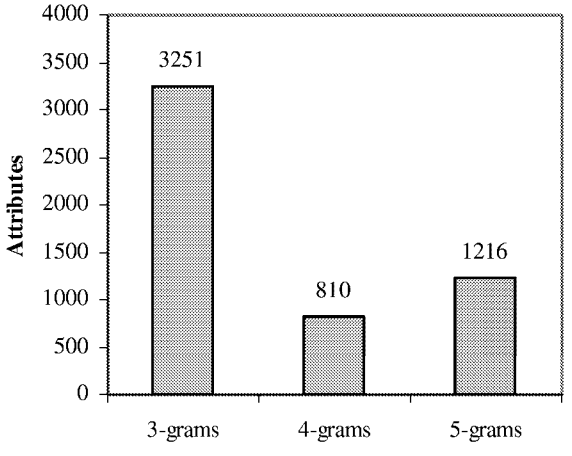

Fig. 8. The composition of the variable-length n-gram model for L=4,500 (5,277 extracted attributes).

7. Conclusions

In this paper we presented a comparison of words and character n-grams in the framework of content-based anti-spam filtering. A series of experiments using two benchmark corpora and a variety of cost-sensitive evaluation measures provides strong evidence that character-level information is better able to discriminate between spam and legitimate messages. The most important property of the character n-gram approach is that it avoids the use of tokenizers, lemmatizers and other language-dependent tools. Those tools are quite vulnerable for the spammers. On the other hand, a model based on character-level information can capture nuances of the *spamminess* of a message and, more importantly, it is not easy for the spammers to fool it by incorporating punctuation marks or other special symbols within a word.

One corpus used in this study comprised homogeneous legitimate messages, since they were taken from a mailing list about linguistics. The legitimate messages of the other corpus are quite heterogeneous, more than the messages found in the mailbox of a single user, since they consist of messages donated by different users. This difference was reflected in the results of the character n-gram models on these two corpora. When considering heterogeneous legitimate messages, short n-grams (3-grams) produced the best models while longer n-grams (4-grams) were optimal when considering homogeneous legitimate messages. This indicates the ability of the character-level approach to be adapted to the properties of a specific corpus or the mailbox of a specific user.

In addition to character n-grams of fixed length, we also proposed a method for extracting variable-length character n-gram based on an existing approach, originally used for extracting multi-word terms for information retrieval applications. Results of cost-sensitive evaluation indicate that the variable-length n-gram model is more effective in any of the three examined cost scenarios (i.e., low, medium, or high cost). Although the majority of the variable-length n-grams consists of 3-grams, there are only a few common members with the fixed-length 3-gram set. Hence, the information included in the variable length n-grams is quite different in comparison to the information represented by case sensitive 3-grams.

An interesting future work direction is to compare the character n-gram and word-based representations in the framework of on-line evaluation of anti-spam filtering. This implies that the set of significant character n-grams should be evolved with use as new messages arrive and classified by the filter so that to better capture the properties of spam and legitimate messages. Another direction is the comparison of character n-grams and bag-of-words representations in cases where both labeled and unlabeled data are available for training.

References

1. F. Sebastiani, Machine learning in automated text categorization, *ACM Computing Surveys*, **34**(1) (2002) pp. 1–47.
2. M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz, A Bayesian approach to filtering junk e-mail, In *Proc. of AAAI Workshop on Learning for Text Categorization* (1998).
3. I. Androutsopoulos, J. Koutsias, K.V. Chandrinos, G. Paliouras, and C.D. Spyropoulos, An evaluation of naive Bayesian anti-spam filtering, In eds. G. Potamias, V. Moustakis, and M. van Someren, *Proc. of the Workshop on Machine Learning in the New Information Age, 11th European Conference on Machine Learning* (2000) pp. 9-17.
4. G. Sakkis, I. Androutsopoulos, G. Paliouras, V. Karkaletsis, C.D. Spyropoulos, and P. Stamatopoulos, A memory-based approach to anti-spam filtering for mailing lists, *Information Retrieval*, **6**(1) (2003) pp. 49-73.
5. H. Drucker, D. Wu, and V. Vapnik, Support vector machines for spam categorization, *IEEE Trans. Neural Networks*, **10** (1999) pp. 1048-1054.
6. I. Androutsopoulos, G. Paliouras, and E. Michelakis, *Learning to filter unsolicited commercial e-mail*, Technical report 2004/2, NCSR "Demokritos" (2004).

7. W. Cavnar and J. Trenkle, N-gram-based text categorization. In *Proc. 3rd Int'l Symposium on Document Analysis and Information Retrieval* (1994) pp. 161-169.
8. V. Keselj, F. Peng, N. Cercone, and C. Thomas, N-gram-based author profiles for authorship attribution. In *Proc. of the Conference Pacific Assoc. Comp. Linguistics* (2003).
9. H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, Text classification using string kernels, *The Journal of Machine Learning Research*, **2** (2002) pp. 419 – 444.
10. V. Vapnik, *The Nature of Statistical Learning Theory*, (Springer, New York 1995).
11. T. Joachims, Text categorization with support vector machines: learning with many relevant features, In *Proc. of the European Conference on Machine Learning* (1998).
12. G. Sakkis, I. Androutsopoulos, G. Paliouras, V. Karkaletsis, C. D. Spyropoulos, and P. Stamatopoulos, Stacking classifiers for anti-spam filtering of e-mail, In *Proc. of 6th Conf. Empirical Methods in Natural Language Processing* (2001) pp. 44-50.
13. J. Hovold, Naive Bayes spam filtering using word-position-based attributes, In *Proc. of the Second Conference on Email and Anti-Spam* (2005).
14. Y. Yang and J.O. Petersen, A comparative study on feature selection in text categorization, In *Proc. of the 14th Int. Conference on Machine Learning* (1997) pp. 412-420.
15. B. Medlock, An adaptive, semi-structured language model approach to spam filtering on a new corpus. In *Proc. of the 3rd Conference on Email and Anti-Spam* (2006).
16. E. Terra, Simple language models for spam detection, In *Proc. of the 14th Text Retrieval Conference* (2005).
17. G. Cormack and T. Lynam, Spam corpus creation for TREC, In *Proc. of 2nd Conference on Email and Anti-Spam* (2005).
18. G. Cormack and T. Lynam, TREC 2005 spam track overview, In *Proc. of the 14th Text Retrieval Conference* (2005).
19. S. Bickel, ECML-PKDD 2006 discovery challenge overview, *In Proc. of the Discovery Chalange Workshop, 17th ECML and 10th PKDD* (2006) pp. 1-9.
20. B. Pfahringer, A semi-supervised spam email detector, *In Proc. of the Discovery Chalange Workshop, 17th ECML and 10th PKDD* (2006) pp. 48-52.
21. N. Tsogkanis and G. Paliouras, TPN2: Using positive-only learning to deal with the heterogeneity of labeled and unlabeled data, *In Proc. of the Discovery Chalange Workshop, 17th ECML and 10th PKDD* (2006) pp. 63-74.
22. R. Pampapathi, B. Mirkin, and M. Levene, A suffix tree approach to text categorisation applied to spam filtering, http://arxiv.org/abs/cs.AI/0503030.
23. H. Berger, M. Koehle, D. Merkl, On the impact of document representation on classifier performance in e-mail categorization, In *Proc. of the 4th International Conference on Information Systems Technology and its Applications* (2005) pp. 19-30.
24. V. Keselj, E. Milios, A. Tuttle, S. Wang, and R. Zhang, DalTREC 2005 spam track: spam filtering using n-gram based techniques, In *Proc. of the 14th Text Retrieval Conference* (2005).
25. A. Bratko and B. Filipic, Spam filtering using character-level Markov models: experiments for the TREC 2005 spam track, In *Proc. of the 14th Text Retrieval Conference* (2005).
26. G. Cormack and A. Bratko, Batch and on-line spam filter comparison, In *Proc. of the Third Conference on Email and Anti-Spam* (2006).
27. V. Metsis, I. Androutsopoulos, and G. Paliouras, Spam filtering with naive Bayes – which naive Bayes?, In *Proc. of the 3rd Conference on Email and Anti-Spam* (2006).
28. I.H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools with Java Implementations*, (Morgan Kaufmann, San Francisco 2000).
29. J. Silva and G. Lopes, A local maxima method and a fair dispersion normalization for extracting multiword units, In *Proc. of the 6th Meeting on the Mathematics of Language* (1999) pp. 369-381.

30. K. Church and K. Hanks, Word association norms, mutual information and lexicography, *Computational Linguistics*, **16**(1) (1990) pp. 22-29.
31. W. Gale and K. Church, Concordance for parallel texts, In *Proc. of the 7th Annual Conference for the new OED and Text Research*, (Oxford, 1991) pp. 40-62.
32. J. Silva, G. Dias, S. Guilloré, and G. Lopes, Using LocalMaxs algorithm for the extraction of contiguous and non-contiguous multiword lexical units, *Lecture Notes on Artificial Intelligence*, 1695 (Springer, 1999) pp. 113-132.
33. J. Egan, *Signal Detection Theory and ROC Analysis*, Series in Cognition and Perception, (New York: Academic Press, 1975).
34. F. Provost and T. Fawcett, Robust classification for imprecise environments, *Machine Learning,* **42**(3) (2001) pp. 203-231.
35. D. Hand and R. Till, A simple generalization of the area under the ROC curve for multiple class classification problems, *Machine Learning*, **45**(2) (2001) pp. 171-186.
36. D. Zeimpekis and E. Gallopoulos, Design of a MATLAB toolbox for term-document matrix generation, In *Proc. Workshop on Clustering High Dimensional Data and its Applications*, (2005) pp. 38-48.