2012 2nd International Conference on Industrial Technology and Management (ICITM 2012)
IPCSIT vol. 49 (2012) © (2012) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2012.V49.49

Learning to Filter Unsolicited Commercial E-Mail

KONG Ying[1a+] and ZHAO Jie[2b]

[1]School of Information and Electronic Engineering, Zhejiang University of Science and Technology Hangzhou, Zhejiang Province, China

[2]School of Experimentation and Practice Training, Management Center, Zhejiang Police Vocational Academy, Hangzhou, Zhejiang Province, China

Abstract. We discuss the model and search biases of the learning algorithms, along withworst-case computational complexity figures, and observe how the latter relate to experimental measurements.We study how classification accuracy is affected when using attributes that represent sequences of tokens, as opposed to single tokens, and explore the effect of the size of the attribute and training set, all within a cost-sensitive framework. Furthermore, we describe the architecture of a fully implemented learning-based anti-spam filter, and present an analysis of its behavior in real use over a period of seven months. Information is also provided on other available learning-based anti-spam filters, and alternative filtering approaches.

Keywords: machine learning, spam, text categorization

1. Introduction

In recent years, the increasing popularity and low cost of e-mail have attracted direct marketers. Bulk mailing software and lists of e-mail addresses harvested from Web pages, newsgroup archives, and service provider directories are readily available, allowing messages to be sent blindly to millions of recipients at essentially no cost. Resources of this kind are a temptation for amateur advertisers and opportunists. Consequently, it is increasingly common for users to receive large quantities of unsolicited commercial e-mail (uce), advertising anything, from vacations to get-rich schemes. The term spam, that originally denoted Usenet messages cross-posted to numerous newsgroups, is now also used to refer to unsolicited, massively posted e-mail messages.[1] Spam is actually a broader term than uce, since spam e-mail is not necessarily commercial (e.g., spam messages from religious cults). Non-uce spam messages, however, are rare, to the extent that the two terms can be used interchangeably. Note that messages sent by viruses are not considered spam, although they, too, can be sent blindly to large numbers of users.

Spam messages are extremely annoying to most users, as they clutter their mailboxes and prolong dial-up connections. They also waste the bandwidth and cputime of isps, and often expose minors to unsuitable (e.g., pornographic) content. Following anti-spam campaigns by volunteer organizations1, most respectable companies now seek the consent of potential customers before sending them advertisements by e-mail, usually by inviting them to tick check boxes when visiting their Web pages. Unfortunately, it does not take many irresponsible advertisers to flood the Internet with spam messages. A 1997 study reported that spam messages constituted approximately 10% of the incoming messages to a corporate network Public comments made by aol in 2002 indicated that of an estimated 30 million e-mail messages daily, about 30% on average was spam; and Jupiter Media Matrix predicted that the number of spam messages users receive would more than double from 2002 to 2006, exceeding 1600 messages per user per year in 2006.2 The situation seems to be worsening, and without appropriate counter-measures, spam messages may undermine the usability of e-mail. Legislative

+ Email: [a]Kongying-888@163.com [b]Zhaojie@zjjy.com.cn@xyz.com

counter-measures are gradually being adopted in the United States, Europe, and elsewhere [Gauthronet and Drouard 2001], but they have had a very limited effect so far, as evidenced by the number of spam messages most users receive daily. Of more direct value are anti-spam filters, software tools that attempt to identify incoming spam messages automatically. Most commercially available filters of this type currently appear to rely on white-lists of trusted senders (e.g, as listed in each user's address book), black-lists of known spammers (typically provided and updated by the filters' developers), and hand-crafted rules that block messages containing specific words or phrases (e.g., "be over 21" in the message's body) or other suspicious patterns in the header fields (e.g., empty "To:" field).

2. Benchmark Corpora for Anti-Spam Filtering

Research on text categorization has benefited significantly from the existence of publicly available, manually categorized document collections, like the Reuters corpora, that have been used a sstandard benchmarks.Producing similar corpora foranti-spam filtering ismore complicated.[2] because of privacy issues. Publicizing spam messages does not pose a problem, since spam messages are distributed blindly to very large numbers of recipients, and, hence, they are effectively already publicly available. Legitimate messages, however, in general cannot be released without violating the privacy of their recipients and senders.

One way to bypass privacy problems is to experiment with legitimate messages collected from freely accessible newsgroups, or mailing lists with public archives. Ling-Spam, the earliest of our benchmark corpora, follows this approach. It consists of 481 spam messages received by the first author, and 2412 legitimate messages retrieved from the archives of the Linguist list, a moderated – and, hence, spam-free – list about linguistics.[2] Ling-Spam has the disadvantage that its legitimate messages are more topic-specific than the legitimate messages most users receive. Hence, the performance of a learning-based anti-spam filter on Ling-Spam may be an over-optimistic estimate of the performance that can be achieved on the incoming messages of a real user, where topic-specific terminology may be less dominant among legitimate messages. In that sense, Ling-Spam is more appropriate to experiments that explore the effectiveness of filters that guard against spam messages sent to topic-specific mailing lists

The SpamAssassin public mail corpus is similar to Ling-Spam, in that its legitimate messages are publicly available.[3] It contains 1897 spam and 4150 legitimate messages, the latter collected from public fora or donated by users with the understanding that they may be made public. The corpus is less topic-specific than Ling-Spam, but its legitimate messages are again not indicative of the legitimate messages that would arrive at the mailbox of a single user, this time for the opposite reason. Many of the legitimate messages that a user receives contain terminology reflecting his/her profession, interests, etc. that is rare in spam messages and part of the success of personal learning-based filters is due to the fact that they learn to identify this user-specific terminology. In a concatenation of legitimate messages from different users this user-specific terminology becomes harder to identify. Hence, the performance of a learning-based filter on the SpamAssassin corpus may be an under-estimate of the performance that a personal filter can achieve.

3. Learning Algorithms

This section describes the four learning algorithms that we examined in the experiments of this paper. Each algorithm can be viewed as searching for the most appropriate classifier in a search space that contains all the classifiers it can learn. We focus on the following characteristics of the algorithms, which bias their search:Instance representation. In our case, the representation of the messages that is used during the training and classification stages. Model representation. The model the algorithm uses to represent the classifiers it can learn. Classifiers that cannot be represented are not considered, and, hence, cannot be learnt. This introduces a bias towards the classifiers that the model can represent, the model bias. The more restrictive the model representation is, the strongest the model bias.Search method. The search method, including classifier selection criteria, that the algorithm adopts to find a good classifier among the ones it can learn. The search method introduces its own bias, the search bias; for example, it may disregard whole areas of the search space, thus favoring some classifiers against other possibly more appropriate ones.

The discussion aims to provide a clear picture of the algorithms that we used, for the benefit of readers not familiar with them, but also to clarify their differences and the limitations that the choices of representations and search methods introduce as implicit biases. Furthermore, we examine the computational complexity of each algorithm, both at the learning and classification stages, speed being an important factor in the design of on-line anti-spam filters.The rest of this section is structured as follows. Subsection 3.1 describes the representation of instances, which is the same in all of the learning algorithms we considered. Subsections 3.2 to 3.5 then describe the four learning algorithms, each in terms of its model and search bias. Subsection 3.6 summarizes the most important differences among the algorithms.

3.1. Instance representation

The experiments of this paper explored richer vector representations. First, instead of Boolean attributes, frequency attributes were used. With frequency attributes, the value of Xi in each message d is xi = ti(d)/l(d), where ti(d) is the number of occurrences in d of the token represented by Xi, and l(d) is the length of dmeasured in token occurrences.12 Frequency attributes are more informative than Boolean ones. For example, reporting simply that the token "$" is present in a message, as in the case of Boolean attributes, may not be enough to raise suspicions that the message is spam, especially if the message is very long; but if a very high proportion of "$" tokens is present in the message, a fact that can be represented with frequency attributes, this may be a good indicator that the message is spam. The additional information of frequency attributes usually comes at the cost of extra complexity in the learning algorithms, which is needed to allow them to cope with attributes whose values are real numbers.

3.2. Naive Bayes

The Naive Bayes learner is the simplest and most widely used algorithm that derives from Bayesian Decision Theory.From Bayes' theorem and the theorem of total probability, the probability that a message with vector ~x = hx1, . . . , xmi belongs in category c is:

$$p(C=c|\vec{X}=\vec{x})=\frac{p(C=c)\cdot p(\vec{X}=\vec{x}|C=c)}{\sum_{c'\in(c_l,c_s)} p(C=c')\cdot p(\vec{X}=\vec{x}|C=c)} \quad (1)$$

The optimal choice for ~x is the category c that maximizes equation (1). The denominator does not affect this choice, as it remains the same for all categories.What matters are the a priori probabilities of the categories and the a posteriori probabilities of the vectors given the category, which need to be estimated from the training data. The number of possible vectors (combinations of attribute values),however, is very large, and many of the vectors will be rare or may not even occur in the training data, making it difficult to obtain reasonable estimates of the required a posteriori probabilities. Hence, a simplifying assumption is made, leading to the Naive Bayes classifier: attributes X1, . . . ,Xm are considered conditionally independent given the category C. This allows (1) to be rewritten as (2), where now only the much fewer a posteriori probabilities P(Xi = xi | C = c) have to be estimated. Along with the a priori probabilities P(C = c), they constitute the model that Naive Bayes uses to represent a classifier.

$$p(C=c\mid \vec{X}=\vec{x})=\frac{p(C=c)\cdot\prod_{i=1}^{m} p(X_i=x_i\mid C=c)}{\sum_{c'\in(c_l,c_s)}\cdot\prod_{i=1}^{m} p(X_i=x_i\mid C=c')} \quad (2)$$

c ∈{cL,cS} i=1Although the independence assumption is in most cases over-simplistic, studies in several domains, including anti-spam filtering, have found Naive Bayes to be surprisingly effective.

linear discriminant, a function of the following form, where the weights wi and w0 can be expressed in terms of the probabilities P(C) and P(Xi | C).

$$p(C=c|\vec{X}=\vec{x})=\frac{p(C=c)\cdot\prod_{i=1}^{m} p(X_i=x_i|C=c)}{\sum_{c'\in(c_l,c_s)} p(C=c')\cdot\prod_{i=1}^{m} p(X_i=x_i|C=c')} \quad (3)$$

i=1The equality f(~x) = 0 corresponds to a hyperplane in the vector space, which in our case aims to separate the legitimate from the spam messages of the training corpus. Hence, when using discrete attributes, an alternative view of Naive Bayes' model is that it is a hyperplane in the vector space. In the case of continuous attributes, which is more relevant to this paper, each a posteriori probability P(Xi = xi | C = c) can

be modeled by a normal (Gaussian) probability density function g(xi; μi,c, σi,c).14 Thus, equation (4) becomes:

$$p(C=c \mid \vec{X}=\vec{x}) = \frac{p(C=c)\cdot\prod_{i=1}^{m} g(x_i;u_{i,c},c,\sigma_{i,c})}{\sum_{c'\in(c_l,c_s)} p(C=c')\cdot\prod_{i=1}^{m} g(x_i;u_{i,c},c,\sigma_{i,c})} \quad (4)$$

The mean and typical deviation of each density function are estimated from the training corpus. The resulting classifier is a quadratic function of the attributes, which is reduced to a linear discriminant under the assumption of homoscedacity, i.e., same variance for all categories [Duda and Hart 1973]. The latter assumption, however, is uncommon in the Naive Bayes literature, and is not adopted in the implementation of Naive Bayes that we used. Hence, for our purposes, an quadratic surface in the vector space, which attempts to separate the legitimate from the spam messages of the training corpus.

4. Cost sensitive Classification

Mistakenly treating a legitimate message as spam can be a more severe error than treating a spam message as legitimate, i.e., letting it pass the filter. This section explains how we took this unbalanced misclassification cost into account in our experiments. We first discuss different usage scenarios for anti-spam filtering, and how each can be captured in a cost matrix. We then turn to the issue of making learning algorithms sensitive to a cost matrix. Finally, we discuss why evaluation measures that are in accordance to the cost matrix are needed, and present the evaluation measures that we used.

4.1. Cost scenarios

Let L → S and S → L denote the two error types, whereby a legitimate message is classified as spam, and vice versa, respectively. Invoking a decision-theoretic notion of cost, let us assume that L → S is λ times more costly than S → L, where λ depends on the usage scenario; for example, whether the filter deletes messages it classifies as spam, or simply flags them as low-priority. More precisely, we use thecost matrix of Table IV, where L → L and S → S denote the cases where the filter classifies correctly a legitimate or spam message, respectively. Here, cost is intended to reflect the effort that users waste to recover from failures of the filter.[4,5] Correctly classifying a message is assigned zero cost, since no user effort is wasted. Misclassifying a spam message (S → L) is assigned unary cost, and misclassifying a legitimate message (L → S) is taken to be λ times more costly.[5] This cost model assumes that the effort to recover from the misclassification of a legitimate message is always the same, regardless of its sender or subject. This is a simplification that will allow us to reach some useful initial conclusions.

Table 1 The cost matrix used in this paper

	Classified as legitimate	***Classified as spam***
Legitimate message	C(L->L)=0	C(L->S)= λ
Spam message	C(S->L)=1	C(S->S)=0

In the experiments of this paper, we used two different usage scenarios. In the first one, the anti-spam filter simply flags messages it considers to be spam, without removing them from the user's mailbox, to help the user prioritize the reading of incoming messages. In this case, λ = 1 seems reasonable, since none of the two error types (L → S or S → L) is graver than the other. The second scenario assumes that messages classified as spam are returned to the sender. An extra paragraph in the returned message explains that the message was blocked by a filter, and asks the original sender to repost the message, this time including a special string in the subject that instructs the filter to let the message pass. The string can be the answer to a frequently changing user-specific riddle,which spamming software cannot guess. In this scenario, λ is set to 9, reflecting the assumption that, if the sender's extra work is counted, recovering from a blocked legitimate message, i.e., reposting the message as instructed, requires as much extra effort as deleting manually approximately 9 spam messages that passed the filter. In previous work, we had also considered a scenario where messages classified as spam were deleted without further action. In that scenario, the cost of recovering from an accidentally misclassified legitimate message was very high; we had used λ = 999. However, our previous results indicated that the third scenario was too difficult for both learning-based filters and filters with hand-crafted rules, and we decided not to study it further.

4.2. Cost-sensitive learning

We now examine how learning algorithms can be made sensitive to a cost matrix.The optimal strategy is to classify an incoming message represented by ~x as spam ifthe expected cost of classifying it as legitimate exceeds that of classifying it as spam, i.e., if inequality (5) holds.

$$p(C = c_s \mid \vec{x}) \cdot c(S \to L) + P(C = c_L \mid \vec{x}) \cdot c(L \to L) > p(C = c_L \mid \vec{x}) \cdot c(L \to S) + P(C = c_S \mid \vec{x}) \cdot c(S \to S) \quad (5)$$

4.3. Cost-sensitive evaluation

We have already considered ways to make the classifiers sensitive to the cost difference between the two types of misclassification errors.[4,6] In a similar manner, this cost difference must be taken into account when evaluating the performance of anti-spam filters. In classification tasks, performance is usually measured in terms of accuracy (Acc) or error rate (Err = 1 − Acc). Let RL and RS be the numbers of legitimate and spam messages to be classified at run time, |L → S| and |S → L| be counters of the L → S and S → L errors at run time, and |S → S| and |L → L|be counters of the correctly classified spam and legitimate messages, respectively. Accuracy and error rate are defined as follows:

$$Acc = \frac{|L \to L| + |S \to S|}{R_L + R_S} \qquad Err = \frac{|L \to S| + |S \to L|}{R_L + R_S} \quad (6)$$

5. Corpus experments

We now move on to the presentation of our corpus experiments, which were performed using the pu1, pu2, pu3, and pua collections. All the experiments wereperformed using stratified 10-fold cross-validation (Section 3.1).The corpus experiments served several goals. First, to explore the degree to which a learning-based anti-spam filter improves, if at all, upon the baseline (no filter) and the hand-crafted rules of Section 4.3; both cost scenarios of Section4.1 were considered. A second goal was to investigate if some of the learning algorithms lead to consistently better performance, identifying the conditions under which particular algorithms excel. A third, intertwined goal was to investigate the effect of the following parameters: the size of the attribute set, the nature of the attributes (whether they represent single tokens or sequences of tokens), and the size of the learning corpus.[7]

As an indication of the computational effciency of the particular implementations that we used, Figure 5 shows the average learning and classification times per training and test message, respectively, as recorded in our experiments on pua forλ = 1.21 In almost all cases, there is a significant increase in both learning and classification time as more attributes are retained (notice the logarithmic vertical scale). Hence, in a real-life application one should be skeptical about using large attribute sets, especially since, as shown in Figure 1, retaining large numbers of attributes does not always lead to significant improvements in accuracy.

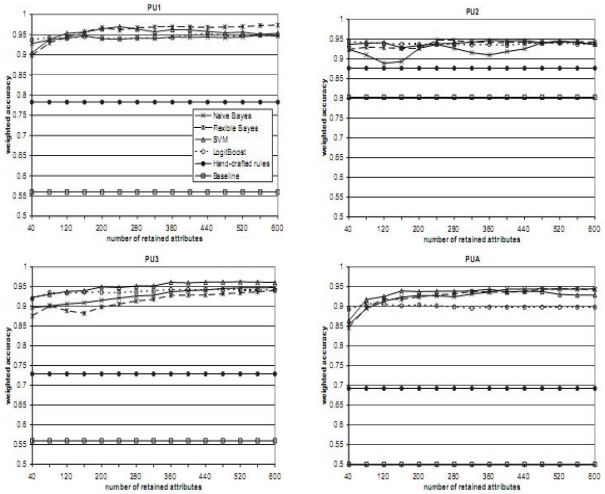

Fig. 1 WAcc results with varying number of 1-gram attributes, forλ=1.

The results of Section 3 guided the design and implementation of a fully functional learning-based anti-spam filter, named Filtron, whose source code is publicly available. Filtron uses the weka machine learning platform.we discuss Filtron's architecture, as an example of how the techniques of the previous sections can be embedded in operational filters.

6. Conclusion

Overall, the results performance was satisfactory for the chosen scenario, though there is scope for improvement. The e-mail client rule that moved messages tagged as spam to the special folder left on average fewer spam messages per week (5.71) in the second author's inbox than he received in a single day (7.66). The downside was that approximately two (1.72) legitimate messages were incorrectly moved to the special folder each week. The second author developed the habit of checking that folder at the end of each week, which made the misclassifications easier to accept. He also felt that in many cases the misclassified legitimate messages were rather indifferent to him, an observation that the analysis of the misclassified messages below confirms.

7. Reference

[1] Burges, C. 1998. A tutorial on Support Vector machines for pattern recognition. Journal of data Mining and Knowledge Discovery 2, 2, 121–167.

[2] Carreras, X. and Marquez, L. 2001. Boosting trees for anti-spam email filtering. In 4th International Conference on Recent Advances in Natural Language Processing. Tzigov Chark, Bulgaria, 58–64.

[3] Chandrinos, K., Androutsopoulos, I., Paliouras, G., and Spyropoulos, C. 2000. Automatic web rating: Filtering obscene content on the web. In Proceedings of the 4th European Conference on Research and Advanced Technology for Digital Libraries. Lisbon, Portugal, 403–406.

[4] Domingos, P. 1999. Metacost: A general method for making classifiers cost-sensitive. In Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining. ACMPress, San Diego, California, 155–164.

[5] Domingos, P. and Pazzani, M. 1996. Beyond independence: Conditions for the optimality of the simple Bayesian classifier. In Proceedings of the 13th International Conference on Machine Learning. Bari, Italy, 105–112.

[6] B. Wellner, A. McCallum, F. Peng, and M. Hay. An integrated, conditional model of information extraction and coreference with application to citation matching. In the proceedings of Uncertainty in Artificial Intelligence (UAI), Banff, Canada, July 2004.

[7] W.R.Gilks, S. Richardson, and D.J.Spiegelhalter, editors. Markov Chain Monte Carlo in Practice. Chapman and Hall, 1996.