Naive Bayes Spam Filtering Using Word Position Attributes

Johan Hovold
Department of Computer Science, Lund University
Box 118, 221 00 Lund, Sweden
johan.hovold.363@student.lu.se

Abstract

This paper explores the use of the naive Bayes classifier as the basis for personalized spam filters. Various machine learning algorithms, including variants of naive Bayes, have previously been used for this purpose, but the author's implementation using word position based attribute vectors gives very good results when tested on several publicly available corpora.

The effect of various forms of attribute selection—removal of frequent and infrequent words, respectively, and by using Mutual Information—is investigated. It is also shown how n-grams, with $n > 1$, may be used to boost classification performance. Finally, a weighting scheme for cost-sensitive classification of variable length attribute vectors is introduced.

1 Introduction

The problem of unsolicited bulk e-mail, or *spam*, gets worse for every year. The vast amount of spam being sent wastes resources on the Internet, wastes time for users and may expose children to unsuitable contents (e.g. pornography). This development has stressed the need for automatic spam filters.

Early spam filters were instances of *knowledge engineering*, using hand-crafted rules (e.g. the presence of the string "buy now" indicates spam). The process of creating the rule base requires both knowledge and time, and the rules were thus often supplied by the developers of the filter. Having common and, more or less, publicly available rules made it easy for spammers to construct their e-mails to get through the filters.

Recently, a shift has occurred, as more focus has been put on *machine learning* for the automatic creation of personalized spam filters. A supervised learning algorithm is presented with e-mails from the users mailbox and outputs a filter. The e-mails have previously been classified manually as spam or non-spam. The resulting spam filter has the advantage of being optimized for the e-mail distribution of the individual user. Thus it is able to use also the characteristics of non-spam, or *legitimate*, e-mails (e.g. presence of the string "machine learning") during classification.

Perhaps the first attempt of using machine learning algorithms for the generation of spam filters was reported by Sahami et al. (1998). They trained a *naive Bayes classifier* and reported promising results. Other algorithms have been tested but there seems to be no clear winner (Androutsopoulos et al., 2004). The naive Bayes approach have been picked up by end-user applications such as the Mozilla e-mail client[1] and the free software project SpamAssassin[2], where the latter is using a combination of both rules and machine learning.

Spam filtering differs from other text categorization tasks in at least to ways. First, one might expect a greater class heterogeneity—it is not the contents per se that defines spam, but rather the fact that it is unsolicited. Similarly, the class of legitimate messages may also span a number of diverse subjects. Secondly, misclassifying a legitimate message is generally much worse than misclassifying a spam.

In this paper the results of using a variant of the naive Bayes classifier for spam filtering, will be presented. The effect of various forms of *attribute selection*, will be explored, as will the effect of considering not only single tokens, but rather sequences of tokens, as attributes. A scheme for cost-sensitive classification will also be introduced. All experiments have been conducted on several publicly available corpora, thereby making a comparison with previously published results possible.

The rest of this paper is organized as follows:

[1] http://www.mozilla.org/
[2] http://www.spamassassin.org/

section 2 presents the naive Bayes classifier; section 3 discusses the benchmark corpora used; the experimental results are presented in section 4; section 5 gives a comparison with previously reported results and in the last section some conclusions are drawn.

2 The Naive Bayes Classifier

In the general context, the instances to be classified are described by attribute vectors $A = \langle a_1, a_2 \ldots, a_n \rangle$. Bayes' theorem says that the posterior probability of an instance A being of a certain class c is

$$P(c|A) = \frac{P(A|c)P(c)}{P(A)}. \tag{1}$$

The naive Bayes classifier then assigns to an instance the most probable, or maximum a posteriori, classification from a finite set C of classes

$$c_{MAP} \equiv \operatorname*{argmax}_{c \in C} P(c|A).$$

By noting that the prior probability $P(A)$ in Equation (1) is independent of c, we may rewrite the last equation as

$$c_{MAP} = \operatorname*{argmax}_{c \in C} P(A|c)P(c). \tag{2}$$

The posterior probabilities $P(A|c) = P(a_1, a_2 \ldots, a_n|c)$ could be estimated directly from the training data, but are generally infeasible to estimate unless the available data is vast. Thus the *naive Bayes assumption*—that the individual attributes are conditionally independent of each other, given the classification—is introduced:

$$P(a_1, a_2, \ldots, a_n|c) = \prod_i P(a_i|c).$$

With this strong assumption, Equation (2) becomes the naive Bayes classifier:

$$c_{NB} = \operatorname*{argmax}_{c \in C} P(c) \prod_i P(a_i|c) \tag{3}$$

(Mitchell, 1997).

In text classification applications, one may choose to define one attribute for each word position in a document. This means that we need to estimate the probability of a certain word w_k occurring at position i, given the target classification c_j: $P(a_i = w_k|c_j)$. Due to training data sparseness, we introduce the additional assumption that the probability of a specific word w_k occurring at position i is identical to the probability of that same word occurring at position m: $P(a_i = w_k|c_j) = P(a_m = w_k|c_j)$ for all i, j, k, m. Thus we estimate $P(a_i = w_k|c_j)$ with $P(w_k|c_j)$. The probabilities $P(w_k|c_j)$ may be estimated with *maximum likelihood estimates*, using Laplace smoothing to avoid zero probabilities:

$$P(w_k|c_j) = \frac{C_j(w_k) + 1}{n_j + |Vocabulary|},$$

where $C_j(w_k)$ is the number of occurrences of the word w_k in all documents of class c_j, n_j is the total number of word positions in documents of class c_j and $|Vocabulary|$ is the number of distinct words in all documents (Mitchell, 1997).

Note that during classification the index i in Equation (3) ranges over all word positions containing words also in the vocabulary, thus ignoring so called *out-of-vocabulary* words. For a more elaborate discussion of the text model used see Joachims (1997).

3 Benchmark Corpora

The experiments were be conducted on the PU corpora[3] and the SpamAssassin corpus[4]. The four PU corpora, dubbed PU1, PU2, PU3 and PUA respectively, have been made publicly available by Androutsopoulos et al. (2004) in order to promote standard benchmarks. The four corpora contain private mailboxes of four different users in encrypted form. The messages have been preprocessed and stripped from attachments, HTML-tags and mail headers (except `Subject`). This may lead to overly pessimistic results since attachments, HTML-tags and mail headers may add useful information to the classification process. For more information on the compositions and characteristics of the PU corpora see Androutsopoulos et al. (2004).

The SpamAssassin corpus (SA) consists of private mail, donated by different users, in unencrypted form with headers and attachments retained[5]. The fact that the e-mails are collected from different distributions may lead to overly optimistic results, e.g. if (some of) the

[3]The PU corpora may be downloaded from http://www.iit.demokritos.gr/skel/i-config/

[4]The SpamAssassin corpus is available at http://spamassassin.org/publiccorpus/

[5]Due to a primitive mbox parser, e-mails containing non-textual or encoded parts, i.e. most e-mails with attachments, are ignored completely in the experiments.

spam messages have been sent to a particular address, but none of the legitimate messages have. On the other hand, the fact that the legitimate messages have been donated by different users may lead to underestimates since this should imply greater diversity of the topics of legitimate e-mails.

The sizes and compositions of the five corpora are shown in Table 1.

corpus	messages	spam freq
PU1	1099	44%
PU2	721	20%
PU3	4139	44%
PUA	1142	50%
SA	6047	31%

Table 1: Sizes and spam frequencies of the five corpora.

4 Experimental Results

As mentioned above, misclassifying a legitimate mail as spam ($L \rightarrow S$) is in general worse than misclassifying a spam message as legitimate ($S \rightarrow L$). In order to capture such asymmetries when measuring classification performance, two measures from the field of information retrieval, called precision and recall, are often used. Denote with $|S \rightarrow L|$ and $|S \rightarrow S|$ the number of spam messages classified as legitimate and spam, respectively, and similarly for $|L \rightarrow L|$ and $|L \rightarrow S|$. Let N_S and N_L be the total number of spam and legitimate messages, respectively. Then *spam recall*(R) and *spam precision*(P) are defined as

$$R = \frac{|S \rightarrow S|}{N_S} \quad \text{and} \quad P = \frac{|S \rightarrow S|}{|S \rightarrow S| + |L \rightarrow S|}.$$

In the rest of this paper spam recall and spam precision will be referred to simply as recall and precision. Intuitively, recall measures effectiveness and precision gives a measure of safety. One is often willing to accept lower recall (more spam messages slipping through) in order to gain precision (fewer misclassified legitimate messages).

Sometimes *accuracy* (*Acc*) is used as a combined measure

$$Acc = \frac{|L \rightarrow L| + |S \rightarrow S|}{N_L + N_S}.$$

All experiments have been conducted using 10-fold cross validation, i.e. the messages have been divided into ten partitions[6] and at each iteration nine partitions have been used for training and the remaining tenth for testing. The reported figures are the means of the values from the ten iterations.

4.1 Attribute Selection

It is common to apply some form of attribute selection process, retaining only a subset of the words—or rather tokens, since punctuation signs and other symbols are often included—found in the training messages. This way the learning and classification process may be sped up and memory requirements are lowered. Attribute selection may also lead to increased classification performance, e.g. since the risk of overfitting the training data is reduced.

Removing infrequent and frequent words, respectively, are two possible approaches. The rationale behind removing infrequent words is that this is likely to have a significant effect on the size of the attribute set and that predictions should not be based on such rare observations anyway. Removing the most frequent words is motivated by the fact that common words, such as the English words “the” and “to”, are as likely to occur in spam as in legitimate messages. Furthermore, this has the effect of making sure that very frequent tokens do not dominate Equation (3) completely.

Another possibility—used by Sahami et al. (1998), Androutsopoulos et al. (2000) and Androutsopoulos et al. (2004)—is to rank the attributes using *Mutual Information*(MI), and to keep only the highest scoring ones. $MI(X;C)$ gives a measure of how well an attribute X discriminates between the various classes in C, and is defined as

$$\sum_{x \in \{0,1\}} \sum_{c \in C} P(x,c) \log \frac{P(x,c)}{P(x)P(c)}$$

(Cover and Thomas, 1991). The probability distributions are estimated using maximum likelihood estimates with Laplace smoothing.

In the experiment tokens occurring less than $n = 1, \ldots, 15$ times were removed. The results indicated unaffected or slightly increased precision at the expense of slightly reduced recall, as n grew. The exception was the PU2 corpus, where precision dropped significantly. The rea-

[6] The PU corpora come prepartitioned and the SA corpus has been partitioned according to the last digit of the messages decimal id.

son for this may be that PU2 is the smallest corpus and contains many infrequent tokens. On the other hand, removing infrequent words had a dramatic impact on the vocabulary size (see Figure 1). Removing tokens occurring less than three times seems to be a good trade-off between memory usage and classification performance, reducing the vocabulary size with 56–69%. This selection scheme was used throughout the remaining experiments.

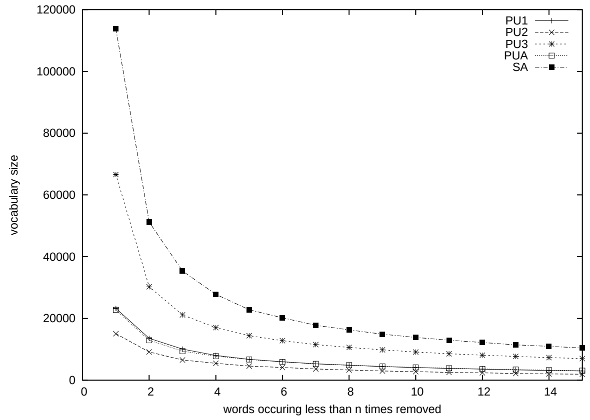

Figure 1: Impact on vocabulary size when removing infrequent words.

Removing the most frequent words turned out to have a major effect on both precision and recall (see Figure 2). This was most significant on the largest and non-preprocessed SA corpus where recall increased from 77% to over 95% by just removing the hundred most common tokens, but classification gained from removing the 100–200 most frequent tokens on all corpora. Removing too many tokens reduced classification performance—again most notably on the smaller PU2 corpus.

In the last attribute selection experiment MI-ranking was used instead of removing the most frequent tokens. Although the gain in terms of reduced memory usage was high—the vocabulary size dropped from 7000–35000 to the number of attributes chosen to be kept, e.g. 500–3000—classification performance was significantly reduced (see Figure 3). Since learning and classification time is mostly unaffected—MI still has be calculated for all attributes—I see no reason for using MI-ranking, if memory usage is not crucial[7].

[7] Androutsopoulos et al. (2004) reaches the opposite conclusion.

4.2 n-grams

Up to now each attribute has corresponded to a single word position, or unigram. Is it possible to obtain better results by considering also token sequences of length two and three, i.e. n-grams for $n = 2, 3$? The questioned was raised and answered partially in Androutsopoulos et al. (2004). Although many bi- and trigrams were shown to have very high information contents, as measured by MI, no improvement was found.

There are many possible ways of extending the attribute set with general n-grams, e.g. by using all available n-grams, by just using some of them or by using some kind of back-off approach. The attribute probabilities, $P(w_i, w_{i+1}, \ldots w_{i+n}|c_j)$, are still estimated using maximum likelihood estimates with Laplace smoothing

$$\frac{C_j(w_i, w_{i+1}, \ldots, w_{i+n}) + 1}{n_j + |Vocabulary|}$$

(see Section 2). Note that extending the attribute set in this way will result in a total probability mass greater than one. Fortunately, this need not be a problem since we are not estimating the classification probabilities explicitly (see Equation (3)).

It turned out that adding bi- and trigrams to the attribute set increased classification performance on all the PU corpora, but not on the SA corpus. The various methods for extending the attribute set all gave similar results and I settled on the simple version which just considers each n-gram as an independent attribute[8]. The results are shown in Table 2.

The precision gain was highest for the corpus with lowest initial precision, namely PU2. For the other PU corpora the precision gain was relatively small or even non-existing. At first the significantly decreased classification performance on the SA corpus came as a bit of a surprise. The reason turned out to be that when considering all bi- and trigrams in the non-preprocessed SA corpus, a lot of very frequent attributes, originating from mail headers and HTML, are added to the attribute set. This had the effect of giving badly discriminating attributes (e.g. some mail headers) and HTML, a too dominant role in Equation (3). By removing

[8] This is clearly not true. The three n-grams in the phrase “buy now”—“buy”, “now” and “buy now”—are obviously not independent.

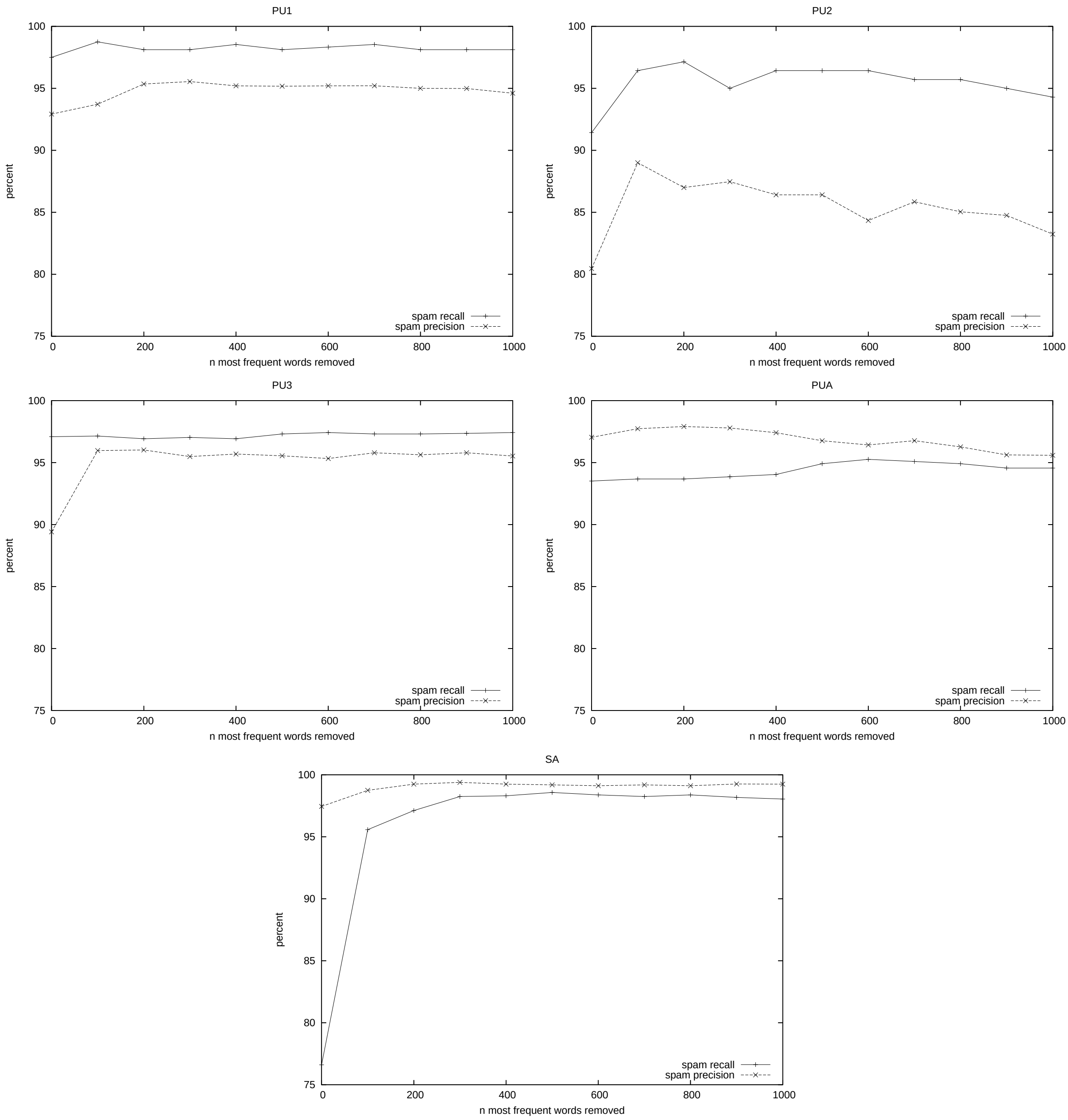

Figure 2: Impact on spam precision and recall when removing the most frequent words.

more of the most frequent words, classification performance was increased also for the SA corpus (see Table 3). The conclusion to be drawn is that mail headers and HTML, although containing useful information, shouldn't be included by brute force. Perhaps some kind of weighting scheme or selective inclusion process would be appropriate.

Finally, considering that extending the attribute set with bi- and trigrams has a dramatic effect on the vocabulary size, the gained classification performance is unlikely to compensate for the increased memory requirements.

4.3 Cost-Sensitive Classification

Generally it is much worse to misclassify legitimate mails than letting spam slip through the filter. Hence, it would be desirable to be able to bias the filter towards classifying messages as legitimate, yielding higher precision at the expense of recall.

One way of biasing the filter is to multiply the prior probability of legitimate messages by some factor $\lambda > 1$ (Androutsopoulos et al., 2000;

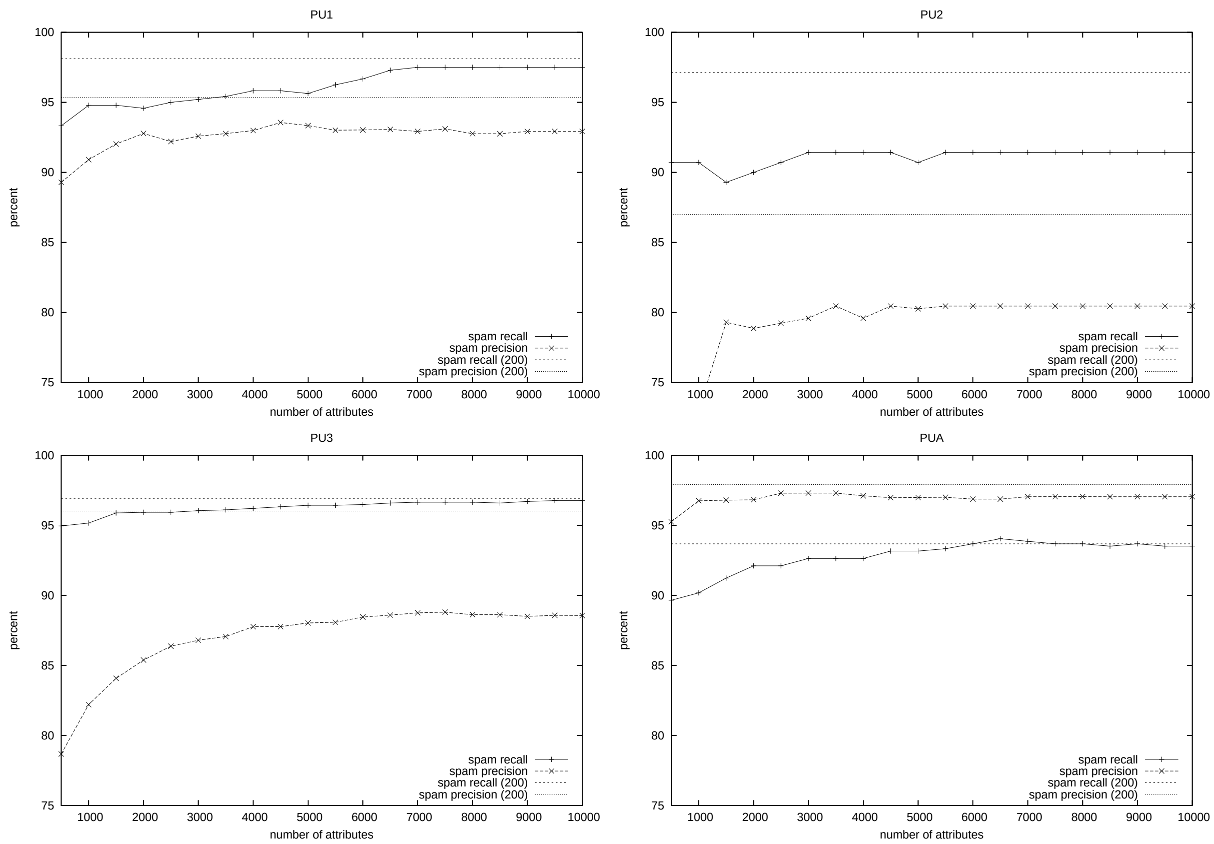

Figure 3: Attribute selection using Mutual Information on the PU corpora—spam recall and precision versus the number of retained attributes. Included is also the precision and recall figures when only the 200 most frequent words have been removed.

Androutsopoulos et al., 2004). This turns out to have a very limited effect, since the expression in Equation (3) is dominated by the posterior probabilities. Another problem is that this weighting scheme is inappropriate to use with word position based attribute vectors, as the impact of the cost factor λ will vary with the length of the document being considered.

To overcome these problems the following simple weighting scheme was used; each posterior probability $P(w_i|c_{legit})$ in Equation (3) was multiplied with a weight $w > 1$. The result of using this "tuning knob" can be seen in Figure 4.

5 Evaluation

Many different machine Machine Learning algorithms besides naive Bayes, such as C4.5, k-Nearest Neighbor and Support Vector Machines, have previously been used in spam filtering experiments. There seems to have been no clear winner, but there is a difficulty in comparing the results of different experiments, since the used corpora have rarely been made publicly available (Androutsopoulos et al., 2004). This section gives a comparison with the implementation and results of the authors of the PU corpora.

In Androutsopoulos et al. (2004), a variant of naive Bayes was compared with three other learning algorithms; Flexible Bayes, LogitBoost and Support Vector Machines (SVM). All of the algorithms used real valued word frequency attributes. The attributes were selected by removing words occurring less than five times and then keeping the 600 words with highest Mutual Information (see Section 4.1). As can be seen in Table 4, the word position based naive Bayes implementation of this paper achieved significantly higher precision and better or comparable recall on all four PU corpora. The results were also better or comparable to the results of the best-performing algorithm on each corpus.

In Androutsopoulos et al. (2000), the authors used a naive Bayes implementation based on boolean attributes, representing the pres-

n-grams	R	P	Acc
PU1			
$n = 1$	98.12	95.35	97.06
$n = 1, 2, 3$	99.17	96.19	97.89
PU2			
$n = 1$	97.14	87.00	96.20
$n = 1, 2, 3$	95.00	93.12	96.90
PU3			
$n = 1$	96.92	96.02	96.83
$n = 1, 2, 3$	96.59	97.83	97.53
PUA			
$n = 1$	93.68	97.91	95.79
$n = 1, 2, 3$	94.56	97.90	96.23
SA			
$n = 1$	97.12	99.25	98.95
$n = 1, 2, 3$	92.26	98.70	97.42

Table 2: Comparison of classification results when using only unigram attributes and uni-, bi- and trigram attributes, respectively. In the experiment words occurring less than three times and the 200 most frequent words have been removed.

n-grams	f	R	P	Acc
$n = 1$	200	97.12	99.25	98.95
$n = 1, 2, 3$	200	92.26	98.70	97.42
$n = 1, 2, 3$	5000	98.46	99.66	99.46

Table 3: Comparison of classification results on the SA corpus when using only unigram attributes and uni-, bi- and trigram attributes, respectively. In the experiment words occurring less than three times and the f most frequent words have been removed.

ence or absence of a fixed number of words. The attributes were selected using Mutual Information. In their experiments three different cost scenarios were explored. Table 5 compares the best results achieved on the PU1 corpus[9] for each scenario, with the results achieved by the naive Bayes implementation of this paper. Due to the difficulty of relating the two different weights, λ and w, the weight w has been selected in steps of 0.05 in order to get equal or higher precision. The authors deemed out the $\lambda = 999$ scenario because of the low recall figures.

[9]The results are for the *bare* PU1 corpus, i.e. the stop-list and lemmatizer have not been applied. The number of attributes have been optimized for each cost scenario.

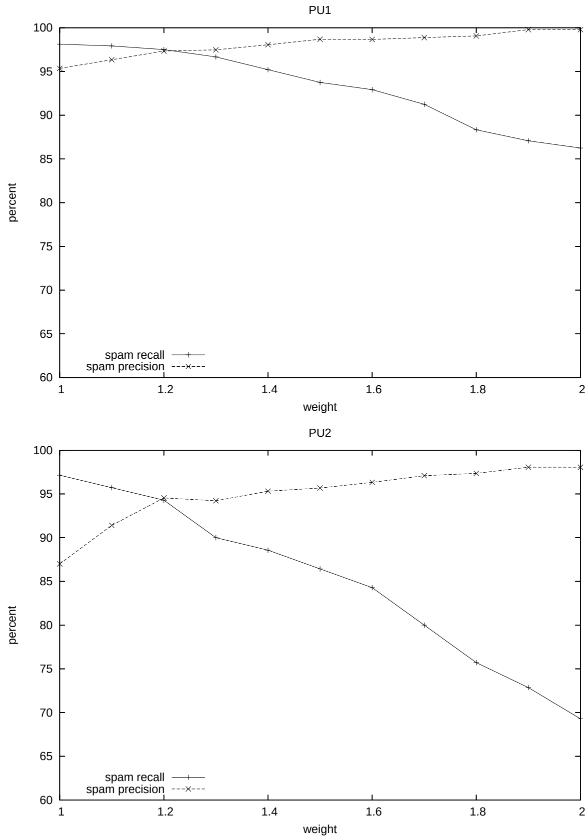

Figure 4: Cost-sensitive classification on the PU1 and PU2 corpora—spam recall and precision versus classification weight.

6 Conclusions

In this paper it has been shown that it is possible to achieve very good classification performance using a word position based variant of naive Bayes. The simplicity and low time complexity of the algorithm, thus makes naive Bayes a good choice for end-user applications.

The importance of attribute selection has been stressed—memory requirements may be lowered and classification performance increased.

By extending the attribute set with n-grams ($n = 1, 2, 3$), better classification performance may be achieved, although at the cost of significantly increased memory requirements.

With the use of a simple weighting scheme, precision may be boosted further, while still retaining a high enough recall level—a feature very important in real life applications.

7 Acknowledgments

The author would like to thank Pierre Nugues for inspiring comments during this work. Many

learner	R	P	Acc
PU1			
Androutsopoulos	99.38	89.58	94.59
Hovold	98.12	95.35	97.06
Flexible Bayes	97.08	96.92	97.34
PU2			
Androutsopoulos	90.00	80.77	93.66
Hovold	97.14	87.00	96.20
Flexible Bayes	79.29	90.57	94.22
PU3			
Androutsopoulos	94.84	93.59	94.79
Hovold	96.92	96.02	96.83
SVM	94.67	96.48	96.08
PUA			
Androutsopoulos	94.04	95.11	94.47
Hovold	93.68	97.91	95.79
Flexible Bayes	91.58	96.75	94.04

Table 4: Comparison of the results achieved by naive Bayes in Androutsopoulos et al. (2004) and by the author's implementation. In the latter, attributes were selected by removing the 200 most frequent words as well as words occurring less than three times. Included is also the results of the best-performing algorithm for each corpus, as found in Androutsopoulos et al. (2004).

thanks also to Peter Alriksson and Pontus Melke for useful suggestions regarding this paper.

References

Ion Androutsopoulos, John Koutsias, Konstantinos Chandrinos, and Constantine D. Spyropoulos. 2000. An experimental comparison of naive bayesian and keyword-based anti-spam filtering with personal e-mail messages. *CoRR*, cs.CL/0008019.

Ion Androutsopoulos, Georgios Paliouras, and Eirinaios Michelakis. 2004. Learning to filter unsolicited commercial e-mail. Technical Report 2004/2, NCSR "Demokritos". Revised version.

Thomas M. Cover and Joy A. Thomas. 1991. *Elements of Information Theory*. Wiley.

Thorsten Joachims. 1997. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In Douglas H. Fisher, editor, *Proceedings of ICML-97, 14th International Conference on Machine Learning*, pages 143–151, Nashville, US. Morgan Kaufmann Publishers, San Francisco, US.

Tom M. Mitchell. 1997. *Machine Learning*. McGraw-Hill.

learner	R	P
Androutsopoulos ($\lambda = 1$)	83.98	95.11
Hovold (unigram, w=1)	98.12	95.35
Hovold (n-gram, w=1)	99.17	96.19
Androutsopoulos ($\lambda = 9$)	78.77	96.65
Hovold (unigram, w=1.20)	97.50	97.34
Hovold (n-gram, w=1.05)	99.17	97.15
Androutsopoulos ($\lambda = 999$)	46.96	98.80
Hovold (unigram, w=1.65)	91.67	98.88
Hovold (n-gram, w=1.30)	96.04	98.92

Table 5: Comparison of the results achieved by naive Bayes on the PU1 corpus in Androutsopoulos et al. (2000) and by the author's implementation. Results for the latter are shown for both unigram and n-gram ($n = 1, 2, 3$) attributes. In both cases, attributes were selected by removing the 200 most frequent n-grams as well as n-grams occurring less than three times. For each cost scenario, the weight w has been selected in steps of 0.05 in order to get equal or higher precision.

Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. 1998. A bayesian approach to filtering junk E-mail. In *Learning for Text Categorization: Papers from the 1998 Workshop*, Madison, Wisconsin. AAAI Technical Report WS-98-05.