Searching for Interacting Features for Spam Filtering

Chuanliang Chen[1], Yun-Chao Gong[2], Rongfang Bie[1,†], and X. Z. Gao[3]

[1]Department of Computer Science, Beijing Normal University, Beijing 100875, China
[2]Software Institute, Nanjing University, Nanjing, China
[3]Department of Electrical Engineering, Helsinki University of Technology, Otakaari 5 A, 02150 Espoo, Finland
C.L.Chen86@gmail.com, rfbie@bnu.edu.cn, gao@cc.hut.fi

Abstract. In this paper, we propose a novel feature selection method—INTERACT to select relevant words of emails for spam email filtering, i.e. classifying an email as spam or legitimate. Four traditional feature selection methods in text categorization domain, Information Gain, Gain Ratio, Chi Squared, and ReliefF, are also used for performance comparison. Three classifiers, Support Vector Machine (SVM), Naïve Bayes and a novel classifier—Locally Weighted learning with Naïve Bayes (LWNB) are discussed in this paper. Four popular datasets are employed as the benchmark corpora in our experiments to examine the capabilities of these five feature selection methods and the three classifiers. In our simulations, we discover that the LWNB improves the Naïve Bayes and gain higher prediction results by learning local models, and its performance is sometimes better than that of the SVM. Our study also shows the INTERACT can result in better performances of classifiers than the other four traditional methods for the spam email filtering.

Key words: Interacting Features, Feature Selection, Naïve Bayes, Spam Filtering.

1 Introduction

The increasing popularity of electronic mails has intrigued direct marketers to flood the mailboxes of millions of users with unsolicited messages. These messages are usually referred to as spam or, more formally, Unsolicited Bulk E-mail (UBE), and may advertise anything, from vacations to get-rich schemes [1]. The negative effect of spam has influenced people's daily lives: filling mailboxes, engulfing important personal mails, wasting network bandwidth, consuming users' time and energy to solve it, not to mention all the other problems associated with it (crashed mail-servers, pornography advertisements sent to children, etc.). A study in 1997 indicated that the spam messages constituted approximately 10% of the incoming messages to a corporate network [4]. CAUBE.AU reports that their statistics show the volume of spam is increasing at an alarming rate, and some people claim they are even

[†] Corresponding author.

abandoning their email accounts because of spam [3]. This situation seems to be worsening with time, and without appropriate counter-measures, spam messages could eventually undermine the usability of e-mails. These serious threats from spam make the spam filtering, whose task is to rule out unsolicited emails automatically from the email stream, more important and in need of solving.

In recent years, many studies address the issue of spam filtering based on machine learning, because the attempts to introduce legal measures against spam mailing have limited effect. Several supervised learning algorithms have been successfully applied to spam filtering: Naïve Bayes [5,6,7,8], Support Vector Machine [9,10], Memory Based Learning methods [11,12], and Decision Tree [13]. Among these classification methods, the Naïve Bayes is particularly attractive for spam filtering, as its performance is surprisingly good [12]. The Naïve Bayes classifier has been the filtering engine of many commercial anti-spam software. Therefore, in this paper, we aim at improving the prediction ability of the Naïve Bayes by introducing locally learned model.

In order to train or test classifiers, it is necessary to go through large corpus with spam and legitimate emails. E-mails of corpuses have to be preprocessed to extract their words (features) belonging to the message subjects, the bodies and/or the attachments. As the number of features in a corpus can end up being very high, it is usual to choose those features that better represent each message before carrying out the filter training to prevent the classifiers from over-fitting [14]. The effectiveness of the classifiers relies on the appropriate choice of these features, the preprocessing steps of the e-mail features extraction, and the selection of the most representative features are crucial for the performance of the filters [15].

In this paper, a novel feature selection method—INTERACT and a novel classifier—LWNB are introduced to deal with spam filtering. The remainder of this paper is organized as follows. Section 2 demonstrates the INTERACT algorithm for the spam filtering. We explain the principles of the e-mail representation and preprocessing in Section 3. Classifiers used in this paper are presented in Section 4. We report the performances of the four feature selection methods and three classifiers using F measure and accuracy in Section 5. Section 6 concludes our study with a few remarks and conclusions.

2 INTERACT Algorithm

Interacting features challenge the current feature selection methods for classification. A feature by itself may have little correlation with the target concept. However, when it is combined with some other features, they can be strongly correlated with the target concept [2]. Many traditional feature selection methods usually unintentionally remove these features, and thus result in the poor classification performances. The INTERACT algorithm can efficiently handle the feature interaction with much lower time cost than the traditional methods. A brief description of the INTERACT algorithm is presented below, and more details can be found in [2].

The INTERACT algorithm searches for the interacting features by solving two key problems: *how to update c-contribution effectively*, and *how to deal with the feature* *order problem*? C-contribution of a feature is an indicator about how significantly the elimination of that feature will affect consistency. Especially, the C-contribution of an irrelevant feature is zero.

To solve the first problem, the INTERACT algorithm calculates the C-contribution efficiently with a hashing mechanism [2]: each instance is inserted into the hash table, and its values of those features in S_{list} are used as the hash keys, where S_{list} is the set of the ranked features not yet eliminated (S_{list} is initialized with the full set of features). Instances with the same hash keys will be inserted into the same entry in the hash table and cover the old information of the labels.

For the second problem, we assume that the set of features can be divided into subset S_1 including relevant features, and subset S_2 containing irrelevant ones. The INTERACT algorithm intends to remove the features in S_2 first, and preserve features in S_1, which more probably remain in the final set of selected features. The INTERACT algorithm achieves this target by applying a heuristic to rank the individual features using *symmetrical uncertainty* (SU) in an descending order so that the (heuristically) most relevant feature is positioned at the beginning of the list. SU has been described in the information theory books and numerical recipes. It is often used as a fast correlation measure to evaluate the relevance of individual features [12,17].

The INTERACT is a filtering algorithm that employs backward elimination to remove the features with no or low C-contribution. Given a full set with N features and a class attribute C, the INTERACT finds a feature subset S_{best} for the class concept [2]. The algorithm consists of two major parts: firstly, the features are ranked in the descending order based on their *Symmetrical Uncertainty* values; secondly, the features are evaluated one by one starting from the end of the ranked feature list. The process is shown as follows.

Algorithm 1. INTERACT Algorithm.

```
Input:    $F$ is the full features set with $N$ features $\{F_1,F_2, \ldots, F_N\}$;
          $C$ is the class label;
          $\delta$ is a predefined threshold.
Output:   $S_{best}$ is subset of selected features.
Process:
          $S_{best} = \varnothing$
          for $i = 1$ to $N$ then
              calculate $SU_{Fi,c}$ for $F_i$
              append $F_i$ to $S_{best}$
          end
          sort $S_{best}$ in descending  order according to $SU_{i,c}$
          $F \leftarrow$ last element of $S_{best}$
          repeat
              if $F \neq NULL$ then
                  $p \leftarrow$ c-contribution of $F$
                  if $p \leq \delta$ then
                      remove $F$ from $S_{best}$
                  end
              end
          until $F = NULL$
          return $S_{best}$
```

3 Preprocessing of Corpus and Message Representation

3.1 Feature Selection Methods for Comparison

Other four feature selection methods are used in this paper to test the capability of the INTERACT algorithm. They are Chi Squared (i.e. χ^2) Statistic, Information Gain, Gain Ratio, and ReliefF. Their definitions are given as follows.

In the following formulas, *m* is the number of classes (in spam filtering domain, *m* is 2), and C_i denotes the *i*th class. *V* represents the number of partitions a feature can split the training set into. Let *N* is the total number of samples, and N_{C_i} is that of class *i*. In the *v*th partition, $N_{C_i}^{(v)}$ denotes the number of samples belonging to class *i*.

Chi Squared: The Chi Squared Statistic is calculated by comparing the obtained frequency with the priori frequency of the same class. The definition is:

$$\chi^2 = \sum_{i=1}^{m} \sum_{v=1}^{V} \frac{(N_{C_i}^{(v)} - \widetilde{N}_{C_i}^{(v)})^2}{\widetilde{N}_{C_i}^{(v)}} \ . \tag{1}$$

where $\widetilde{N}_{C_i}^{(v)} = (N^{(v)} / N) N_{C_i}$ denotes the prior frequency.

Information Gain: Information Gain is based on the feature's impact on the decreasing entropy, and is defined as follows:

$$InfoGain = [\sum_{i=1}^{m} -(\frac{N_{C_i}}{N}) \log(\frac{N_{C_i}}{N})] - [\sum_{v=1}^{V} (\frac{N^{(v)}}{N}) \sum_{i=1}^{m} -(\frac{N_{C_i}^{(v)}}{N^{(v)}}) \log(\frac{N_{C_i}^{(v)}}{N^{(v)}})] \ . \tag{2}$$

Gain Ratio: Gain Ratio is firstly used in C4.5, which is defined as (3):

$$GainRatio = InfoGain / [\sum_{i=1}^{m} -(\frac{N_{C_i}}{N}) \log(\frac{N_{C_i}}{N})] \ . \tag{3}$$

ReliefF: The key idea of Relief is to estimate the features according to how well their values distinguish among the instances that are near to each other. The ReliefF is an extension of the Relief, improving the Relief algorithm by estimating the probabilities more reliably and extending to deal with the incomplete and multiclass data sets. More details can be found in [17].

3.2 Corpus Preprocessing and Message Representation

Each e-mail in the corpora is represented as a set of words. After analyzing all the e-mails of a corpus, a dictionary with *N* words/features is formed. Every e-mail is represented as a feature vector including *N* elements, and the *i*th word of the vector is a binary variable representing whether this word is in this e-mail. During preprocessing, we perform the word stemming, stop-word removable and Document

Frequency Threshold (DFT), in order to reduce the dimension of feature space. The HTML tags of the e-mails are also removed during preprocessing. Finally, we extract the first 5,000 tokens of the dictionary according to their mutual information to form the corpora used in this paper.

4 Classifiers for Spam Filtering

In this paper, we use three classifiers to test the capabilities of the aforementioned feature selection methods. The three classifiers are Support Vector Machine (SVM), Naïve Bayes, and Locally Weighted learning with Naïve Bayes (LWNB) that is an improvement of Naïve Bayes firstly introduced into spam filtering domain by us. We here only briefly introduce the LWNB, and more details can be found in [1].

In the LWNB, the Naïve Bayes is learned locally in the same way as the linear regression is used in locally weighted linear regression. A local Naïve Bayes model is fit to a subset of the data that is in the neighborhood of the instance, whose class value is to be predicted [1]. The training samples in this neighborhood are weighted, and further ones are assigned with less weight. The classification is then obtained from these Naïve Bayes models.

The subset of the data used to train each locally weighted Naïve Bayes model is determined by a nearest neighbors algorithm. In the LWNB, the first *k* nearest neighbors are selected to form this subset, where *k* is a user-specified parameter. How to determine the weight of each instance of the subset? As in [1], we use a linear weighting function in our experiments, which is defined as:

$$f_{linear} = 1 - d_i / d_k \ , \tag{4}$$

where d_i is the Euclidean distance to the *i*th nearest neighbor x_i. Obviously, by using f_{linear}, the weight decreases linearly with the distance. Empirical study shows the LWNB is not particularly sensitive to the choice of *k* as long as *k* is not too small [1]. Too small *k* may cause the local Naïve Bayes model to fit the noise in the data.

The Naïve Bayes calculates the posterior probability of class c_l for a test instance with *m* attribute values a_1, a_2, …, a_m as follows:

$$p(c_l \mid a_1, a_2, ..., a_m) = \frac{p(c_l)\prod_{j=1}^{m} p(a_j \mid c_l)}{\sum_{i=1}^{C}[p(c_i)\prod_{j=1}^{m} p(a_j \mid c_i)]} \ , \tag{5}$$

where *C* is the total number of classes. In the LWNB, the individual probabilities on the right-hand side of (5) are estimated based on the weighted data. The prior probability for class c_l becomes:

$$p(c_l) = \frac{1 + \sum_{i=0}^{n} I(c_i = c_l) w_i}{C + \sum_{i=0}^{n} w_i} \ , \tag{6}$$

where c_i is the class value of the *i*th training instance, and the indicator function $I(x{=}y)$ is 1 iff $x = y$.

The attribute of data is assumed nominal, and as for the numeric attributes, they are discretized. The conditional probability of a_j is given by:

$$p(a_j \mid c_l) = \frac{1+\sum_{i=0}^{n} I(a_j = a_{ij}) I(c_i = c_l) w_i}{n_j + \sum_{i=0}^{n} I(a_j = a_{ij}) w_i} \ , \tag{7}$$

n_j is the number of values of attribute j, and a_{ij} is the value of attribute j of ith instance.

5 Experiments and Analysis

5.1 Corpus in Simulations

The experiments are based on four popular benchmark corpora, PU1, PU2, PUA, and Ling Spam, which are all available on [16]. In all PU corpora and Ling Spam corpus, attachments, html tags, and header fields other than the subjects are removed, leaving only subject lines and mail body texts. In order to address privacy, each token of a corpus is encoded to a unique integer. The details about each corpus are given below.

PU1 Corpus: The PU1 corpus consists of 1,099 messages, which has 481 spam messages and 618 legitimated ones. The spam rate is 43.77%.

PU2 Corpus: The PU2 corpus contains less messages than PU1, which has 721 messages. Among them, there are 579 messages labeled legitimate and 142 spam.

PUA Corpus: The PUA corpus has 1,142 messages, half of which, i.e., 571 messages, are marked as spam and the other half legitimate.

Ling Spam Corpus: The Ling spam corpus includes 2,412 legitimate messages from a linguistic mailing list and 481 spam ones collected by the author. The spam rate is 16.63%. Different from PU corpora, the messages of Ling spam corpus come from different sources: the legitimate messages are collected from a spam-free, topic-specific mailing list and the spam ones from a personal mailbox. Therefore, the distribution of mails is less similar from the normal user's mail stream, which makes the messages of Ling spam corpus easily separated.

5.2 Performance Measures

We use two popular evaluation metrics of the text categorization domain to measure the performance of the classifiers: accuracy and F measure.

Accuracy: Accuracy is the percentage of the correct predictions in the total predictions. It is defined as follows:

$$Accuracy = \frac{P_c}{P_t} \times 100\% \ . \tag{8}$$

where P_c is the number of the correct predictions, and P_t is the number of the total predictions. The higher of the accuracy, the better.

***F* measure**: The definition of *F* measure is as follows:

$$F = \frac{2R \times P}{R + P} \quad , \tag{9}$$

where *R* represents Recall, which is the percentage of the messages for a given category that are classified correctly; *P* is the Precision, the percentage of the predicted messages for a given class that are classified correctly. *F* measure ranges from 0 to 1, and the higher, the better.

5.3 Results and Analysis

The following classification performance is measured through a 10-fold cross-validation. We select all of the interacting features, i.e., features with non-negative C-contribution. Table 1 summarizes the results of dimension reduction after the INTERACT selects the features.

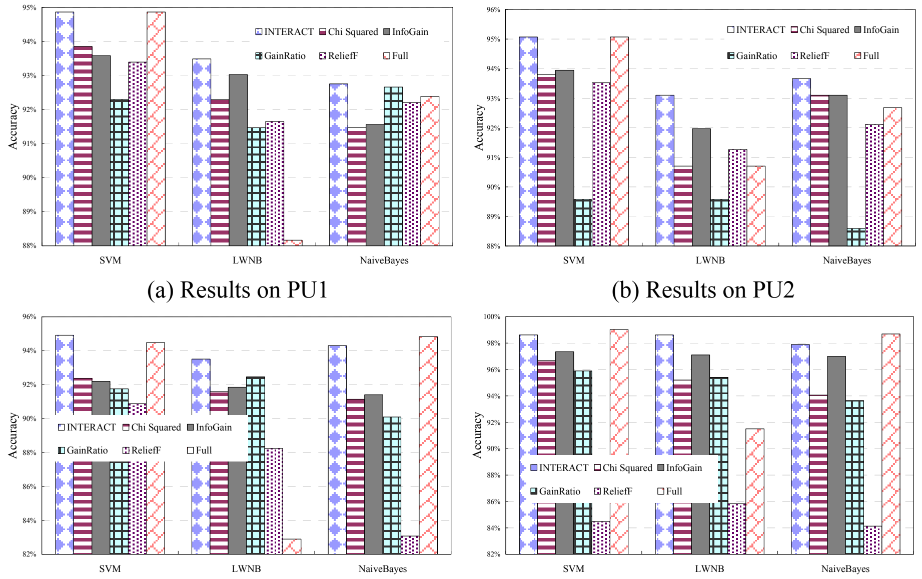
Table 1. Summary of results of INTERACT selected features on the four benchmark corpora.

	PU1	PU2	PUA	Ling Spam
Num. of features with non-negative c-contribution	43	43	42	64

From Table 1, we can find that the dimensions of data have been reduced sharply after removing irrelevant features by the INTERACT. Therefore, we just run the classifiers on these data rather than reducing them further by adjusting the parameter *δ*. From Table 1, we also can conclude that there are many irrelevant words/features existing in corpus for the spam filtering, and more than 99% of the features are removed by the INTERACT.

The following histograms show the performances of the three classifiers, SVM (using linear kernel), Naïve Bayes, and LWNB, on the four corpora. As for other four feature selection methods for comparison, we select the first *M* features according to the features' scores, where *M* is the number of the interacting features found by the INTERACT algorithm.

From Fig. 1 and Fig. 2, we discover that the INTERACT algorithm can improve the performances of all the three classifiers. Their performances on the reduced corpus are equal to or better than those on the full corpus, evaluated by the accuracy and *F* measure. For example, the performances of the SVM on PU1 and PU2 corpora reduced by the INTERACT is equal to those on the full corpora, and its performance on PUA corpus reduced by the INTERACT is better than that on the full corpus. However, the performance of the SVM on Ling Spam corpus reduced by the INTERACT is slightly worse than that on the full corpus. The feature selection capability of the INTERACT is obviously better than the other popular feature selection methods. The competitive performances of the classifiers on the data handled by the INTERACT show that only a few relevant words can still distinguish between the spam and legitimate emails. This is true in practice, for example, it is well known that the words "buy, purchase, jobs, …" usually appear in the spam e-mails, and they thus are useful email category distinguishers.



(a) Results on PU1

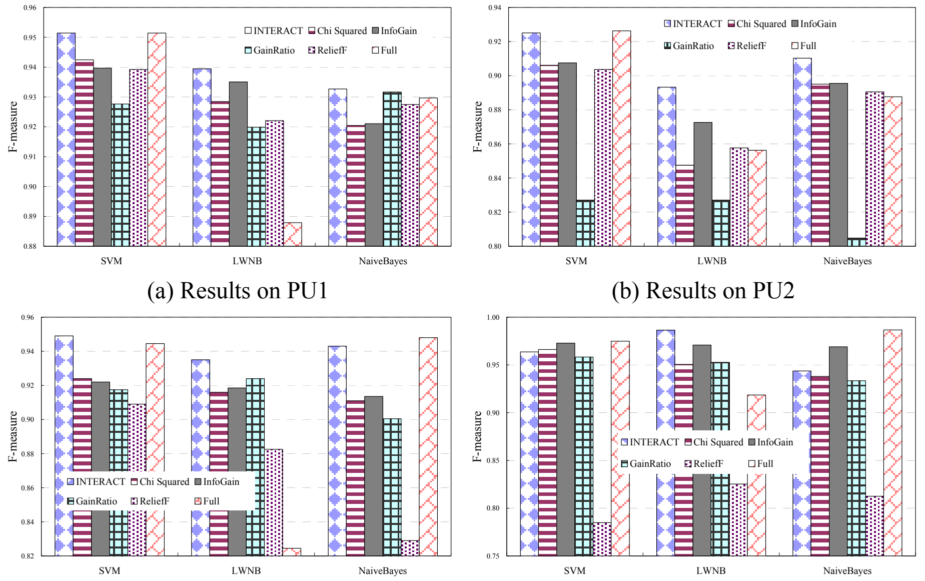
(b) Results on PU2

(c) Results on PUA

(d) Results on Ling Spam

Fig. 1. Performances of aforementioned three classifiers and four feature selection methods on PU1, PU2, PUA, and Ling Spam benchmark corpora with accuracy evaluation measure.

The performance of the LWNB is also promising. On Ling Spam corpus, its performance is even better than that of the SVM, which is a well-known powerful classifier. On PU1 and Ling Spam corpora, the LWNB successfully improves the performance of the Naïve Bayes by using locally weighted model. However, its performance is worse than that of the Naïve Bayes on PU2 and PUA corpora. The reason may be that the task of the spam filtering suits the hypothesis of the class conditional independence of the Naïve Bayes, that is, given the class label of the others, the frequencies of the words in one email are conditionally independent of one another. Based on a careful observation, we have another question "*why the LWNB performs poorly on full corpus*"? The reason is: there are many irrelevant features existing on full corpus, which can be also concluded from the feature selection results by performing the INTERACT. When determining the neighbors, all the features take part in calculating *distance*, and too many irrelevant features conceal the truly useful effects of the relevant features, and therefore result in that the LWNB finds the wrong or irrelevant neighbors to generate locally weighted Naïve Bayes models. However, the LWNB is still a promising classifier for the spam filtering, when combined with some excellent feature selection methods, such as the INTERACT.



(a) Results on PU1

(b) Results on PU2

(c) Results on PUA

(d) Results on Ling Spam

Fig. 2. Performances of aforementioned classifiers and four feature selection methods on PU1, PU2, PUA and Ling Spam benchmark corpora with *F* measure evaluation measure.

6 Conclusions

In this paper, we present our work on the spam filtering. Firstly, we introduce the INTERACT algorithm to select interacting words/features for the spam filtering. Other four traditional feature selection methods are also performed in the experiments for performance comparison. Secondly, we propose a novel classifier LWNB to improve the performance of the Naïve Bayes, a most popular classifier in the spam filtering area, to deal with the spam filtering. Totally, three classifiers, SVM, Naïve Bayes and LWNB, are run on four corpora preprocessed by the five feature selection methods and corresponding full corpora in our simulations. Two popular evaluation metrics, *accuracy* and *F* measure, are used to measure the performances of these three classifiers. Our empirical study shows that the INTERACT feature selection can improve all of the three classifiers' performances, and its feature selection ability is better than that of the four traditional feature selection methods. We briefly analyze the reason why the INTERACT and other four methods can work together to perform well. We also find out that the LWNB can improve the performance of the Naïve Bayes, which is sometimes superior to the SVM.

Acknowledgements. The research work presented in this paper was supported by grants from the National Natural Science Foundation of China (Project No.

10601064). X. Z. Gao's research work was funded by the Academy of Finland under Grant 214144.

References

1. Frank, E., Hall, M., Pfahringer, B.: Locally Weighted Naive Bayes. Proceedings of the Conference on Uncertainty in Artificial Intelligence. Morgan Kaufmann, pp.249-256, 2003.
2. Zhao, Z., Liu, H.: Searching for Interacting Features. In: Proc. of International Joint Conference on Artificial Intelligence (IJCAI), Hyderabad, India, 2007.
3. CAUBE.AU: http://www.caube.org.au/spamstats.html 2006.
4. Cranor, L.F., LaMacchia, B.A.: Spam! Communications of ACM, 41(8):74-83, 1998.
5. Sahami, M, Dumais, S., Heckerman, D., et al.: A Bayesian approach to filtering junk e-mail. In Learning for Text Categorization, Madison, Wisconsin, 1998.
6. Schneider, K.M.: A Comparison of Event Models for Naïve Bayes Anti-Spam E-Mail Filtering. In: Proc. of the 10th Conference of the European Chapter of the Association for Computational Linguistics, Budapest, Hungary, pp. 307-314, 2003.
7. Androutsopoulos, I, Paliouras, G., Karkaletsis, V., et al.: Learning to Filter Spam E-mail: A Comparison of a Naïve Bayesian and a Memory-based Approach. In: Proc. of the Workshop on Machine Learning and Textual Information Access, pp. 1-13, 2000.
8. Zhang, L., Zhu, J., Yao, T.: An Evaluation of Statistical Spam Filtering Techniques. ACM Trans. Asian Lang. Inf. Process, 3(4):243-269, 2004.
9. Drucker, H., Wu, D., Vapnik, V.N.: Support vector machines for spam categorization. IEEE Trans. on Neural Networks, 10(5):1048-1054, 1999.
10. Kolcz, A., Alspector, J.: SVM-based filtering of e-mail spam with content-specific misclassification costs. In: Proc. of the TextDM'01 Workshop on Text Mining - held at the 2001 IEEE International Conference on Data Mining, 2001.
11. Sakkis, G., Androutsopoulos, I., Paliouras, G., et al.: A Memory-Based Approach to Anti-Spam Filtering for Mailing Lists. Information Retrieval, 6(1):49-73, 2003.
12. Yu, L., Liu, H.: Feature selection for high-dimensional data: a fast correlation-based filter solution. In: Proc. of ICML 2003, 2003.
13. Carreras, X., Marquez, L.: Boosting trees for anti-spam email filtering. In: Proc. of RANLP01, 4th Int. Conference on Recent Advances in Natural Language Processing, 2001.
14. Méndez, J.R., Iglesias, E.L., Fdez-Riverola, F., et al.: Analyzing the Impact of Corpus Preprocessing on Anti-Spam Filtering Software. Research on Computing Science, 17:129-138, 2005.
15. Méndez, J.R., Fdez-Riverola, F., Díaz, F., et al.: A Comparative Performance Study of Feature Selection Methods for the Anti-spam Filtering Domain. In: Proc. of Advances in Data Mining, Applications in Medicine, Web Mining, Marketing, Image and Signal Mining (ICDM 2006), pp. 106-120, 2006.
16. Email Benchmark Corpus, http://www.aueb.gr/users/ion/publications.html, 2006.
17. Kononenko, I.: Estimating Attributes: Analysis and Extensions of Relief. In: Proc. of European Conference on Machine Learning, pp. 171-182, 1994.