Oscillating Feature Subset Search Algorithm for Text Categorization

Jana Novovičová[1,2], Petr Somol[1,2], and Pavel Pudil[1,2]

[1] Dept. of Pattern Recognition, Institute of Information Theory and Automation, Academy of Sciences of the Czech Republic
{novovic, somol}@utia.cas.cz
http://ro.utia.cz/

[2] Faculty of Management, Prague University of Economics, Czech Republic
pudil@fm.vse.cz
http://www.fm.vse.cz

Abstract. A major characteristic of text document categorization problems is the extremely high dimensionality of text data. In this paper we explore the usability of the Oscillating Search algorithm for feature/word selection in text categorization. We propose to use the multiclass Bhattacharyya distance for multinomial model as the global feature subset selection criterion for reducing the dimensionality of the bag of words vector document representation. This criterion takes into consideration inter-feature relationships. We experimentally compare three subset selection procedures: the commonly used best individual feature selection based on information gain, the same based on individual Bhattacharyya distance, and the Oscillating Search to maximize Bhattacharyya distance on groups of features. The obtained feature subsets are then tested on the standard Reuters data with two classifiers: the multinomial Bayes and the linear SVM. The presented experimental results illustrate that using a non-trivial feature selection algorithm is not only computationally feasible, but it also brings substantial improvement in classification accuracy over traditional, individual feature evaluation based methods.

1 Introduction

The application of machine learning (ML) and pattern recognition (PR) in the real-world domain data often encounters problems caused by the high dimensionality of the input space. More and more often, data sets in numerous research fields contain a very large number (from hundreds to tens of thousands) of features. The situation becomes worse if the ratio of relevant features to the irrelevant ones is low, as for example in text document categorization problem. By removing these insignificant features, the learning process becomes more effective, and the performance of the classifier improves.

Text categorization (also known as text classification) is the task of automatically sorting a set of documents into predefined classes based on its contents. This task is of great practical importance. Document classification may appear in many applications including e-mail filtering, mail routing, spam filtering, news monitoring, selective dissemination of information to information consumers, automated indexing of scientific articles, and so on.

J.F. Martínez-Trinidad et al. (Eds.): CIARP 2006, LNCS 4225, pp. 578–587, 2006.

An increasing number of statistical classification methods and ML algorithms have been explored to build automatically a classifier by learning from previously labelled documents, e.g., *naive Bayes* ([1], [2]), *k-nearest neighbor* ([3]), *neural networks* ([4]), *decision trees* ([1]), *symbolic rule induction* ([5], [6]), *regression* ([7]), *support vector machines* ([8]), *boosting* ([9]). The overview of Sebastiani [10] discusses the main approaches to text categorization (TC).

In TC, usually a document representation using a *bag of words* approach is employed (each position in the feature vector representation corresponds to a given word). This scheme leads to very high-dimensional feature space, too high for conventional classification methods. Dimensionality reduction (DR) is a very important step in TC because irrelevant and redundant features often degrade the performance of classifiers both in speed and classification accuracy. Two approaches for DR exist, either selecting a subset of the original features (feature selection), or transforming the features into new ones (feature extraction).

In TC problem the dominant approach is feature selection (FS) using various criteria. Traditional methods for feature subset selection in TC use an evaluation function that is applied to single words. All words are independently evaluated and sorted according to the assigned criterion. Then, a predefined number of the best features is taken to form the best feature subset. Scoring of individual words can be performed using some measure like, e.g., *document frequency* (Yang and Pedersen [11]), *mutual information* [11], *information gain* (also known as average mutual information) or χ^2 *statistic* (Caropreso et al. [17], Yang and Pedersen [11], Yang and Liu [18]) and *odds-ratio* (Mladenić [12]). Yang and Pedersen [11] and Mladenić and Grobelnik [13] give experimental comparison of the above mentioned measures in TC. The information gain (IG) and several very simple frequency measures were reported to work well on text data. Forman [14] presents an extensive comparative study of twelve FS criteria for the high-dimensional domain of text categorization.

In Section 2 and 3 the standard methodology of feature selection for text categorization is discussed. Section 4 introduces our novel approach. Section 5 shows experimental results. Conclusions are given in Section 6.

2 Text Categorization Task

The *text categorization* task is the task of assigning free documents expressed in natural language into one or more thematic *classes* (*categories*) belonging to the predefined set $\mathcal{C} = \{c_1, \ldots, c_{|\mathcal{C}|}\}$ of $|\mathcal{C}|$ classes. The construction of a *text classifier* relies on an *initial collection* of documents pre-classified under $\mathcal{C}$. Let $\mathcal{D} = \{d_1, \ldots, d_{|\mathcal{D}|}\}$ be the finite training document set and $\mathcal{V} = \{w_1, \ldots, w_{|\mathcal{V}|}\}$ be the vocabulary set containing $|\mathcal{V}|$ distinct words occurring in training documents. Given a set of document vectors $\{d_1, \ldots, d_{|\mathcal{D}|}\}$ and their associated class labels, the problem is to estimate the true class label of a free document. The construction of a text classifier consists of essentially three phases: *document indexing* and *dimensionality reduction*, *classifier learning*, and *classifier evaluation*.

In TC a document d_i is usually transformed into a vector in the *term space*. The indexing process is characterized by a definition of what a term is. In TC applications of the thematic kind, the set of terms for a document is usually made to coincide with the set of words occurring in the document. The number of features can be dramatically reduced by the domain dependent methods, which include the elimination of stop words, stripping of special characters as well as stemming algorithms or morphological analysis. For further DR the domain independent methods can be used.

A wide variety of supervised learning approaches have been explored to construct automatically a classifier by learning from previously labelled documents. All of the classification methods mentioned in Section 1 still retain their popularity, but in recent years Support Vector Machines (SVM) and boosting have become the two dominant learning techniques in TC. The reasons for this are twofold. First, these are the two methods that have unquestionably shown the best performance in comparative TC experiments performed so far. Second, these are the newest methods in the classifier learning area, and the ones with the strongest justifications from computational learning theory. A further factor that has determined their success is the free availability, at least for research purposes, of respective software packages, such as SVMlight and BoosTexter.

3 Feature Subset Selection

Because of computational complexity the filter approach is preferable over the wrapper approach to feature subset selection in TC. Given a predefined integer $|\mathcal{V}'|$, the methods for word selection attempt to select from the original set $\mathcal{V}$, the set $\mathcal{V}'$ of words with $|\mathcal{V}'| \ll |\mathcal{V}|$ that, when used for document representation, yields the highest effectiveness. Different methods for feature subset selection have been developed in ML and PR using different evaluation functions and search procedures. In the following we consider the global (a feature subset is chosen for the classification under all classes) filter FS approach.

3.1 Traditional Feature Selection Evaluation Functions

A simple and effective global word selection function is the *document frequency* of a word w, that is, only the terms that occur in the highest number of documents are retained.

Information-theoretic functions for FS have been used in the literature, among them *mutual information*, *information gain*, χ^2 *statistic* and *odds-ratio*. The mathematical definitions of the main functions used for DR in text classification task are summarized in the paper of Sebastiani [10]. All functions are specified locally to a specific class $c_j \in \mathcal{C}$. In order to assess the value of word w in a *global* class-independent sense, either the sum or the weighted sum, or the maximum of their class-specific values are usually computed.

Information gain, used in our experiments for comparison as a ranking measure for selection of the best individual features, is defined as

$$IG(w_v) = \sum_{j=1}^{|\mathcal{C}|} P(c_j, w_v) \log \frac{P(c_j, w_v)}{P(c_j)P(w_v)} \ . \tag{1}$$

Here $P(w_v)$ is the probability, that the word w_v occurred in document, $P(c_j)$ is the probability of the class c_j. $P(c_j, w_v)$ is the joint probability of the class c_j and the occurrence of the word w_v.

3.2 Traditional Feature Selection Search Methods

Most methods for feature subset selection that are used for text processing tasks are very simple in comparison to the other methods known in ML and PR.

Best individual features (BIF) method (see e.g. [15]) evaluates all the $|\mathcal{V}|$ words individually according to a given criterion, sorts them and selects the best $|\mathcal{V}'|$ words. Since the vocabulary usually consists of several thousands or tens of thousands of words, the BIF methods are popular in TC because they are rather fast, efficient and simple. However, they evaluate each word separately and completely ignore the existence of other words and the manner how the words work together.

Forward selection algorithms start with an empty set of features and add one feature at a time until the final feature set is reached. *Backward elimination* algorithms start with a feature set containing all features and remove features one at a time. As opposed to the BIF methods these feature selection procedures reflect to a certain extent the dependencies between words (see e.g. [15]). However, despite their relative simplicity even these basic sequential methods can show to be too slow to yield results in reasonable time.

4 Proposed Feature Selection Algorithm

For the exceedingly high-dimensional problem of text categorization we have found the Oscillating Search (OS) algorithm [16] to be a rare case of a more advanced algorithm being capable of yielding promising results in reasonable time. In the following we show the simplified (thus the fastest) modification of the algorithm to consistently improve the selected subsets of terms and, consequently, improve the classification performance. This is made possible by using the following form of Bhattacharyya distance as the term subset evaluation criterion.

4.1 Bhattacharyya Distance for Multinomial Model

We proposed to use the *multiclass Bhattacharyya distance* developed for multinomial model for class conditional densities in Novovicova [20] as the FS criterion in text categorization.

In the *multinomial* model, the document d_i is represented by a feature vector, each feature variable N_{iv} is the number of times certain word w_v occurs in that document. In this model each document d_i is drawn from a multinomial distribution over the set of all words in the vocabulary $\mathcal{V}$ with as many independent trials as the length $|d_i|$ (the number of words from $\mathcal{V}$ occurring in the document). The order of the words is lost, however the number of occurrences is captured. The *multiclass Bhattacharyya distance* of document d_i is given as

$$B(d_i) = \sum_{j=1}^{|\mathcal{C}|-1} \sum_{k=j+1}^{|C|} P(c_j)P(c_k)B_{jk}(d_i), \tag{2}$$

where $P(c_j)$ and $P(c_k)$ are the prior probabilities of the class c_j and c_k, respectively. $B_{jk}(d_i)$ is the *pairwise Bhattacharyya distance* of d_i defined as

$$B_{jk}(d_i) = -|d_i| \ \log \sum_{v=1}^{|\mathcal{V}|} \sqrt{P(w_v|c_j)P(w_v|c_k)}. \tag{3}$$

Here $P(w_v|c_j)$ and $P(w_v|c_k)$ are the probabilities of w_v in c_j and c_k, respectively. The Bhattacharyya distance for one feature corresponding to the word w_v between classes c_j and c_k, to be denoted *individual Bhattacharyya distance* is then

$$B_{jk}(w_v) = -|d_i| \log \left(\sqrt{P(w_v|c_j)P(w_v|c_k)} + \sqrt{(1 - P(w_v|c_j))(1 - P(w_v|c_k))} \right). \tag{4}$$

4.2 Oscillating Search

As opposed to other sequential subset selection methods the *Oscillating Search* (OS) [16] is not dependent on pre-specified direction of search (forward or backward). It is based on repeated modification of the current subset $\mathcal{V}_r$ of r features. This is achieved by alternating the so-called *down-* and *up-swings*. The *down-swing* removes successively o worst features from the current set $\mathcal{V}_r$ to obtain a new set $\mathcal{V}_{r-o}$ at first, then adds o best ones to $\mathcal{V}_{r-o}$ to obtain a new current set $\mathcal{V}_r$. The *up-swing* adds o "good" features to the current set $\mathcal{V}_r$ to obtain a new set $\mathcal{V}_{r+o}$ at first, then removes o "bad" ones from $\mathcal{V}_{r+o}$ to obtain a new current set $\mathcal{V}_r$ again. The up- and down-swings are repeated as long as the set $\mathcal{V}_r$ gets improved. The value of o shall be set to 1 initially and may be later increased to allow more thorough search at a cost of more computational time. The algorithm then terminates when o exceeds a user-specified *limit* Δ.

The OS algorithm starts the search from some initial set $\mathcal{V}_r$ of r features and brings improvement after each pair of swings (terminates otherwise). This is very advantageous for the high-dimensional problem we deal with here, as no time is lost in evaluating term subsets of sizes far from the desired one. Note that while the sequential forward search (see Section 3.2) needs 400 complex steps to find a subset of 400 features, the OS may achieve the same or better result in a couple of steps, provided it is initialized properly. The perfect way of initializing OS for TC is taking use of some of the standard fast BIF methods.

Let us describe the essential form of the algorithm formally:

Step 1: (*Initialization*) Find the initial set $\mathcal{V}_r$. (For TC do it by means of BIF). Let $c = 0$. Let $o = 1$.

Step 2: (*Down-swing*) Remove such o-tuple from $\mathcal{V}_r$, so that the new set $\mathcal{V}_{r-o}$ retains the highest criterion value. Add such o-tuple from $\mathcal{V} \setminus \mathcal{V}_{r-o}$ to $\mathcal{V}_{r-o}$, so that the new subset $\mathcal{V}_r^{new}$ yields the highest criterion value. If $\mathcal{V}_r^{new}$ is better than $\mathcal{V}_r$, let $\mathcal{V}_r = \mathcal{V}_r^{new}$, $c = 0$, $o = 1$ and go to Step 4.

Step 3: (*Last swing did not find better solution*) Let $c = c + 1$. If $c = 2$, then none of previous two swings has found better solution; extend the search by letting $o = o + 1$. If $o > \Delta$, stop the algorithm, otherwise let $c = 0$.

Step 4: (*Up-swing*) Add such o-tuple from $\mathcal{V} \setminus \mathcal{V}_r$ to $\mathcal{V}_r$, so that the new set $\mathcal{V}_{r+o}$ has the highest criterion value. Remove such o-tuple from $\mathcal{V}_{r+o}$, so that the new set $\mathcal{V}_r^{new}$ yields the highest criterion value. If $\mathcal{V}_r^{new}$ is better than $\mathcal{V}_r$, let $\mathcal{V}_r = \mathcal{V}_r^{new}$, $c = 0$, $o = 1$ and go to Step 2.

Step 5: (*Last swing did not find better solution*) Let $c = c + 1$. If $c = 2$, then none of previous two swings has found better solution; extend the search by letting $o = o + 1$. If $o > \Delta$, stop the algorithm, otherwise let $c = 0$ and go to Step 2.

The OS algorithm can be considered a "higher level" procedure, as it may take use of other feature selection methods as sub-procedures in place of up- and down-swings. One notable property of the OS algorithm is the fact that the fastest improvement of the target subset may be expected in initial phases of the search. This behavior is advantageous, because it gives the option of stopping the search prematurely without too serious result-degrading consequences.

In the context of TC it is desirable to keep the procedure as simple and fast as possible, thus we consider only the set-up with $\Delta = 1$ and the simplest form of up- and down-swings as described here. For more details see [16].

5 Experimental Study

Our experiments are conducted to illustrate that using a non-trivial Oscillating Search feature selection algorithm is not only computationally feasible, but it also brings substantial improvement in classification accuracy.

We adopted the bag of words approach. After text preprocessing the global filter feature selection is performed. For evaluating the classification performance assuming single-label TC we use the standard classification accuracy.

5.1 Data Set

In our experiments we examined the commonly used Reuters-21578 data set.[1] Our text preprocessing included removing all non-alphabetic characters, ignoring

[1] http://www.daviddlewis.com/resources/testcollections/reuters21578

all the words that contained digits or non alpha-numeric characters, removing words from a stop-word list. We replaced each word by its morphological root, removed all the words which had less than three occurrences. After pre-processing the data set contained 33 classes of document-representing vectors of dimensionality 10105. The largest class contained 3924, the smallest only 19 non-zero documents. All tests have been done by means of 10-fold cross-validation over the whole data set.

5.2 Examined Criteria and Search Methods

In our experiments we used the Bhattacharyya distance (4) and information gain (1) as ranking measures for the selection of the best individual features. The proposed oscillating algorithm then uses the Bhattacharyya distance (2) to evaluate groups instead of individual features only.

For selecting r salient words from the complete vocabulary set $\mathcal{V}$ we used not only the proposed oscillating procedure (Section 4), but for comparison also the best individual features method (BIF) based on individual Bhattacharyya distance (BIF IB) and information gain (BIF IG) presented in Section 3.1. We initialize the OS algorithm by feature subsets found by means of BIF IB. In our experiments, we set $r = \{6, 12, 25, 50, 100, 200, 400, 600, 1000, 1500, 2000, 3000, 4500, 6000\}$, respectively.

5.3 Classifiers

All feature selection methods were examined on two classifiers: the Bayes classifier for multinomial probabilistic model and the linear SVM.

Bayes classifier. We use the multinomial model as described by McCallum and Nigam in [2]. The predicted class for document d_i is the one that maximizes the posterior probability of each class given the test document $P(c_j|d_i)$,

$$P(c_j|d_i) \propto P(c_j) \prod_{v}^{|\mathcal{V}|} P(w_v|c_j)^{N_{iv}}.$$

Here $P(c_j)$ is the prior probability of the class c_j, $P(w_v|c_j)$ is the probability that a word chosen randomly in a document from class c_j equals w_v, and N_{iv} is the number of occurrences of word w_v in a document d_i. We smoothed the word and class probabilities using Bayesian estimate with word priors and a Laplace estimate, respectively.

Linear Support Vector Machine. The SVM method has been found recently one of the best performing tools in TC. Especially its excellent generalization ability in high-dimensional sparse spaces is very suitable for text related problems. Linear kernel has been found sufficient in this context. For our experiments we used the LibSVM implementation [21]. We used the standard C-SVC form of the classifier with default value of $C = 1$. No data scaling has been done.

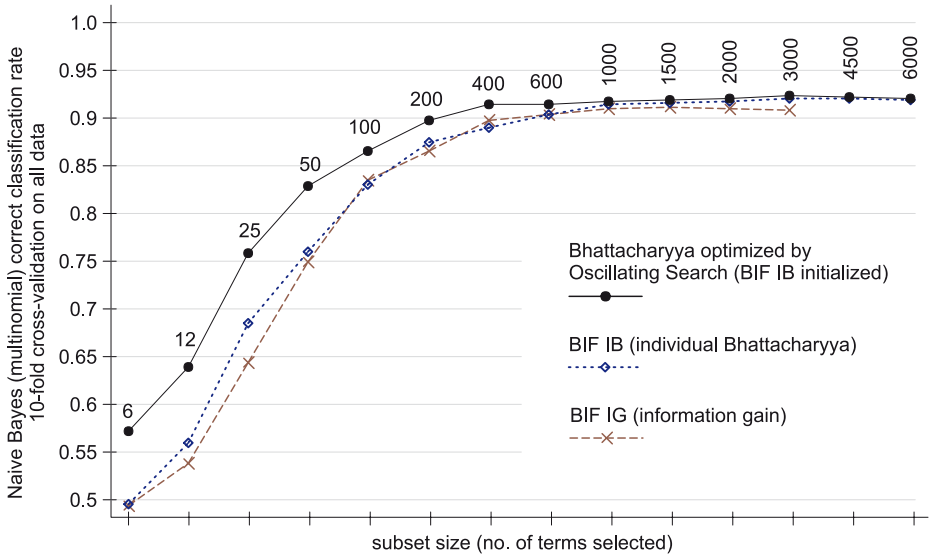

Fig. 1. Multinomial Bayes classifier 10-fold cross-validated classification rate.

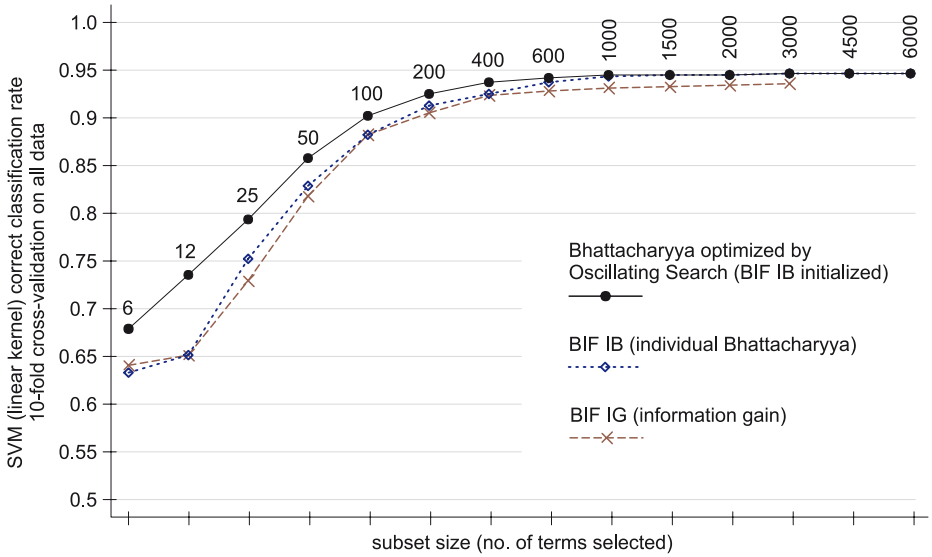

Fig. 2. SVM classifier 10-fold cross-validated classification rate

5.4 Results

It can be clearly seen in Figs. 1 and 2 that feature selection based on Oscillating Search that optimizes Bhattacharyya distance on groups of terms is constantly

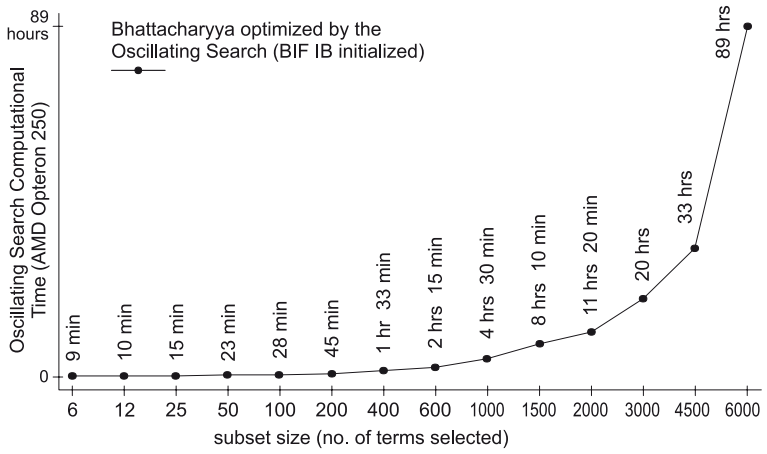

Fig. 3. Oscillating Search computational time

superior to BIF approaches for subset sizes roughly ≤ 1000. For larger subsets the improvement is hardly observable or not present at all; the search time then becomes inadequate as well (see Fig. 3). Note, that the improvement is equally notable for both of the tested classifiers. This also adds to the discussion about the impact of feature selection on SVM performance.

A second notable observation is the slight superiority of individual Bhattacharyya over information gain in BIF based search.

6 Conclusions and Future Work

We have shown that in text categorization tasks it is possible to achieve considerable classification accuracy improvement by employing a feature search procedure, that, unlike traditional approaches, evaluates feature groups instead of individuals. The most notable improvement is to be expected with subsets of lower sizes, where the time requirements of the discussed Oscillating Search procedure stay in reasonable limits. We have also shown that the multinomial Bhattacharyya distance is a good measure for both group-wise and individual feature selection, capable of identifying features that are good for fundamentally different classifiers.

In the future we intend to investigate in more detail the applicability of alternative Oscillating Search versions and other even more complex search procedures to text categorization problems. Another option for further research is including SVM parameter optimization in the feature selection process.

Acknowledgement. The work has been supported by EC MUSCLE project FP6-507752, the Grant Agency of the Academy of Sciences of the Czech Republic (CR) project A2075302, CR MŠMT grant 2C06019 and 1M0572 DAR.

References

1. Lewis, D.D., Ringuette, M.: A comparison of two learning algorithms for text categorization. In: Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval, Las Vegas, US (1994) 81–93
2. McCallum, A., Nigam, K.: A comparison of event models for naive Bayes text classification. In: Proc. Workshop Learning for Text Categorization AAAI (1998)
3. Yang, Y., Liu, X.: A re-examination of text categorization methods. In: Proc. 22nd ACM SIGIR-99 Int. Conf. on R. & D. in Information Retrieval (1999) 42–49
4. Ng, H.T., Goh, W.B., Low, K.L.: Feature selection, perceptron learning, and a usability case study for text categorization. In: Proc. 20th ACM Int. Conf. on R. & D. in Information Retrieval SIGIR-97 (1997) 67–73
5. Apté, C., Damerau, F.J., Weiss, S.M.: Automated learning of decision rules for text categorization. ACM Transactions on Information Systems **12**(3) (1994) 233–251
6. Cohen, W.W., Singer, Y.: Context-sensitive learning methods for text categorization. ACM Transactions on Information Systems **17**(2) (1999) 141–173
7. Yang, Y., Chute, C.G.: An example-based mapping method for text categorization and retrieval. ACM Transactions on Information Systems **12**(3) (1994) 252–277
8. Joachims, T.: Transductive inference for text classification using support vector machines. In: Proc. 16th Int. Conf. on Machine Learning ICML-99 (1999) 200–209
9. Schapire, R.E., Singer, Y.: BoosTexter: a boosting-based system for text categorization. Machine Learning **39**(2/3) (2000) 135–168
10. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys **34**(1) (2002) 1–47
11. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: Proc. 14th Int. Conf. on Machine Learning ICML-97. (1997) 412–420
12. Mladenić, D.: Feature subset selection using in text learning. In: 10th European Conference on Machine Learning. (1998) 95–100
13. Mladenić, D., Grobelnik, M.: Feature selection for unbalanced class distribution and naive Bayes. In: Proceedings of the Sixteenth International Conference on Machine Learning. (1999) 258–267
14. Forman, G.: An experimental study of feature selection metrics for text categorization. Journal of Machine Learning Research **3** (2003) 1289–1305
15. Mladenić, D.: Machine Learning on non-homogeneous, distributed text data. PhD thesis, J. Stefan Institute, University of Ljubljana, Ljubljana, SL (1998)
16. Somol, P., Pudil, P.: Oscillating search algorithms for featute selection. In: Proc. of the 15th IAPR Int. Conference on Pattern Recognition. (2000) 406–409
17. Caropreso, M., Matwin, S., Sebastiani, F.: A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. In: Text Databases and Document Management: Theory and Practice. Idea Group Publishing, Hershey, PA (2001) 78–102
18. Yang, Y., Liu, X.: A re-examination of text categorization methods. In: Proc. 22nd Int. ACM SIGIR Conf. on R. & D. in Information Retrieval. (1999) 42–49
19. Jain, A.K., Duin, R.P.W., Mao, J.: Statistical pattern recognition: A review. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 4–37
20. Novovičová, J., Malík, A.: Text document classification using finite mixtures. Research Report 2063, ÚTIA AVČR, Prague, Czech Republic (2002)
21. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. (2001) Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.