Evaluating some Feature Selection Methods for an Improved SVM Classifier

Daniel Morariu, Lucian N. Vintan, and Volker Tresp

***Abstract*—**Text categorization is the problem of classifying text documents into a set of predefined classes. After a preprocessing step the documents are typically represented as large sparse vectors. When training classifiers on large collections of documents, both the time and memory restrictions can be quite prohibitive. This justifies the application of features selection methods to reduce the dimensionality of the document-representation vector. Four feature selection methods are evaluated: Random Selection, Information Gain (IG), Support Vector Machine (called SVM_FS) and Genetic Algorithm with SVM (GA_FS). We showed that the best results were obtained with SVM_FS and GA_FS methods for a relatively small dimension of the features vector comparative with the IG method that involves longer vectors, for quite similar classification accuracies. Also we present a novel method to better correlate SVM kernel's parameters (Polynomial or Gaussian kernel).

***Keywords*—**Features Selection, Learning with Kernels, Support Vector Machine, Genetic Algorithms and Classification.

I. Introduction

WHILE more and more textual information is available online, effective retrieval is difficult without good indexing and summarization of document content. Document categorization is one solution to this problem. In recent years a growing number of categorization methods and machine learning techniques have been developed and applied in different contexts.

Documents are typically represented as vectors of word frequencies in a features space. Each word in the vocabulary is represented as a separate dimension. The number of occurrences of a word in a document represents the value of the corresponding component in the document's vector. This document representation results in a huge dimensionality of the feature space, which poses a major problem to text categorization. The native feature space consists of the unique terms that occur into the documents, which can be tens or hundreds of thousands of terms for even a moderate-sized text collection and can lead to obtaining poorer results. Due to the large dimensionality, much time and memory are needed for training a classifier on a large collection of documents. This is why some techniques for relevant feature selection are used to improve the classification results [1]. We explore various methods to reduce the feature space and the response time. As we'll show the categorization results are better when we work with a smaller optimized dimension of the feature space. As the feature space grows, the accuracy of the classifier doesn't grow significantly; actually it even can decreases due to noisy vector elements.

This paper present a comparative evaluation of four feature selection methods used prior to documents classifications (Random Selection, Information Gain [9], SVM_FS [12, 16] and GA_FS [17]). Each of these developed techniques computes the relevance of the feature from a different point of view and thus they have more or less influence on the classification results. Also we investigated the influence of the input data representation on classification accuracy. We have used three type of representation, Binary, Nominal and Cornell Smart. For the classification process we used the Support Vector Machine technique, which has proven to be efficient for nonlinearly separable input data [23], [19].

The general process of classifying text data can be considered as having four steps. The first step consists of feature extraction from the text file, eliminating the stop-words, extracting the root of the words and creating the feature vectors. In the second step features are selected. The third step is the learning step in which the preprocessed data are used as inputs to the learning algorithm. In the last step the classification process is evaluated.

The Support Vector Machine (SVM) is actually based on learning with kernels some of which form the support vectors. A great advantage of this technique is that it can use large input data and feature sets. Thus, it is easily to test the influence of the number of features on classification accuracy. We implemented SVM classification for two types of kernels: *polynomial kernel* and *Gaussian kernel (Radial Basis Function - RBF)*. We tried to find a simplified form of the kernels using correlating the parameters [15]. We have also modified this SVM representation that it can be used as a method of features selection in the text-mining step [16].

The SVM algorithm is designed for working with two classes; for multi-class categorization we chose the well-known method "*one class versus the rest*" [21]. Thus we repeated two class classification for each topic (the category where the document is classified) obtaining M decision functions if there are M classes.

Next two sections contain prerequisites for the work that we

Manuscript received October 13, 2006.

D. Morariu is with the Faculty of Engineering, "Lucian Blaga" University of Sibiu, Computer Science Department, E. Cioran Street, No. 4, 550025 Sibiu, Romania (phone: 40/0740/092202; e-mail: daniel.morariu@ulbsibiu.ro).

L. Vintan is with the Faculty of Engineering, "Lucian Blaga" University of Sibiu, Computer Science Department, E. Cioran Street, No. 4, 550025 Sibiu, Romania (e-mail: lucian.vintan@ulbsibiu.ro).

V. Tresp is with the Siemens AG, Information and Communications, 81739 Munchen, Germany (e-mail: volker.tresp@siemens.com).

present in this paper. After that we present the framework and the methodology used for our experiments, followed by presenting the main results of our experiments. The last section debates and concludes about the most important obtained results and proposes some further work.

II. Support Vector Machine

The Support Vector Machine (SVM) is a classification technique based on statistical learning theory [21], [18] that was applied with great success in many challenging non-linear classification problems and was successfully applied to large data sets.

The SVM algorithm finds a hyperplane that optimally splits the training set. The optimal hyperplane can be distinguished by the maximum margin of separation between all training points and the hyperplane (a practical idea about implementation of this algorithm can be found in [3]). Looking at a two-dimensional problem we actually want to find a line that "best" separates points in the positive class from points in the negative class. The hyperplane is characterized by a decision function like:

$$f(x) = \operatorname{sgn}\left(\left\langle \mathbf{w}, \mathbf{\Phi}(x) \right\rangle + b\right), \tag{1}$$

where **w** is the weight vector, orthogonal to the hyperplane, "*b*" is a scalar that represents the margin of the hyperplane, "*x*" is the current sample tested, "*Φ(x)*" is a function that transforms the input data into a higher dimensional feature space and $\langle \cdot, \cdot \rangle$ representing the dot product. *Sgn* is the signum function that returns 1 if the value is greater or equal to 0 and -1 otherwise. If **w** has unit length, then <**w**, Φ*(x)*> is the length of Φ*(x)* along the direction of **w**. Generally **w** will be scaled by ||**w**||. The training part the algorithm needs to find the normal vector "**w**" that leads to the largest "*b*" of the hyperplane.

The problem seems very easy to be solved but we have to keep in mind that the optimal classification line should classify correctly all the elements generated by the same given distribution. There are a lot of hyperplanes that meet the classification requirements but the algorithm tries to determine the optimum one. This learning algorithm can be performed in a dot product space and for data which is linear separable, by constructing *f* from empirical data. It is based on two facts. First, among all the hyperplanes separating the data, there is a unique optimal hyperplane, distinguished by the maximum margin of separation between any training point and the hyperplane. Second, the capacity of the hyperplane to separate the classes decreases with the increasing of the margin.

For training data which is not separable by a hyperplane in the input space the idea of SVM is to map the training data into a *higher-dimensional feature space* via Φ, and construct a separating hyperplane with the maximum margin there. This yields a non-linear decision boundary into the input space. By the use of a kernel function $\langle \mathbf{w}, \phi(x) \rangle$ it is possible to compute the separating hyperplane without explicitly carrying out the map into the feature space [21].

In order to find the optimal hyperplane, we need to solve the following objective function:

$$\underset{\mathbf{w}\in H, b\in\Re}{\text{minimize}}\ \tau(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 \tag{2}$$

subject to $y_i\left(\left\langle \mathbf{w}, \mathbf{x_i} \right\rangle + b\right) \ge 1$ for all *i*=1,...,m

The constraints ensure that *f(x$_i$)* will be +1 for y_i=+1 and -1 for y_i=-1. This problem is computationally attractive because it can be constructed by solving a quadratic programming problem for which efficient algorithms already exist. Function τ is called objective function with inequality constrain. Together, they form a so-called *primal optimization problem*. To solve this type of problems it is more convenient to deal with the dual problem by introducing the Lagrange multipliers $\alpha_i \ge 0$ and the Lagrangian [21] which lead to the so-called *dual optimization problem*.

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{m} \alpha_i (y_i (\langle \mathbf{x_i}, w \rangle + b) - 1) \tag{3}$$

,with Lagrange multipliers $\alpha_i \ge 0$. The Lagrangian L must be maximized with respect to the dual variables α_i, and minimized with respect to the primal variables **w** and *b*. This leads to:

$$\mathbf{w} = \sum_{i=0}^{m} \alpha_i y_i x_i \text{ , and } \sum_{i=0}^{m} \alpha_i y_i = 0 \tag{4}$$

The solution vector thus has an expansion in terms of training examples. Note that although the solution **w** is unique (due to the strict convexity of primal optimization problem), the coefficients α_i, need not be. According to the *Karush-Kuhn-Tucker (KKT) theorem*, only the Lagrange multipliers α_i that are non-zero at the saddle point, correspond to constraints that are precisely met. Formally, for all *i*=1,…,m, it can be written:

$$\alpha_i\left[y_i\left(\left\langle \mathbf{x_i}, \mathbf{w} \right\rangle + b\right) - 1\right] = 0 \text{ for all i=1,...,m} \tag{5}$$

The samples x_i for which $\alpha_i > 0$ are called *Support Vectors*. According to the KKT condition they lie exactly on the margin. All remaining training samples are irrelevant. By eliminating the primal variables **w** and *b* in the Lagrangian we arrive to the so-called dual optimization problem, which is the problem that one usually solves in practice.

$$\underset{\alpha\in\Re^m}{\text{maximize}} W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \left\langle x_i, x_j \right\rangle \tag{6}$$

Thus the hyperplane can be written in the dual optimization problem as:

$$f(x) = \operatorname{sgn}\left(\sum_{i\in\Re} y_i \alpha_i \left\langle x_i, x \right\rangle + b\right) \tag{7}$$

Everything was formulated in a dot product space. On the practical level, changes have to be made to perform the algorithm in a higher-dimensional feature space. Thus the new patterns $\Phi(x_i)$ can equally well be the result of mapping the original input patterns x_i into a higher dimensional feature space using function Φ. Maximizing the target function and evaluating the decision function involve the computation of dot products $\langle \phi(x), \phi(x) \rangle$ in a higher dimensional space. These expensive calculations are reduced significantly by using a positive definite kernel *k*, such that $k(x, x') = \langle x, x' \rangle$. This substitution, which is referred sometimes as the *kernel trick* is used to extend hyperplane classification to nonlinear Support Vector Machines. The kernel trick can be applied since all feature vectors only occur in dot products. The weight vectors than becomes an expression in the feature space, and therefore Φ will be the function through which we represent the input vector in the new space. Thus we obtain the *decision function* as the following form:

$$f(x) = \mathrm{sgn}\left(\sum_{i \in \Re} y_i \alpha_i k(x, x_i) + b \right) \tag{8}$$

III. Features Selection Methods

A substantial fraction of the available information is stored in text or document databases which consist of a large collection of documents from various sources such as news articles, research papers, books, web pages, etc. Data stored in text format is considered semi-structured data that means neither completely unstructured nor completely structured.

In text categorization, feature selection is typically performed by assigning a score or a weight to each term and keeping some number of terms with the highest scores while discarding the rest. After this, experiments evaluate the effects that features selection has on both the classification performance and the response time.

Numerous feature scoring measures have been proposed and evaluated: Odds Ratio [11], Information Gain, Mutual Information [10], Document Frequency, χ^2-test, Term Strength [24], or Support Vector Machine [12], a. o.

As follows we'll present our four methods of features selection that we will further use in our work. All feature selection methods use as a starting point the same vectors obtained after the extraction step.

A. Random Selection (RAN)

In this feature selection method random weights between 0 and 1 are assigned to each feature. We chose this simple method just to have a base (lower limit) in evaluating the performance gains introduced by the other three methods. Then training and testing sets of various sizes are chosen by selecting the features according to their descending weights. These sets (with various sizes) are generated so that the larger sets are containing the smaller sets. We repeat this process for three times. After doing this we classify all of the sets and then we compute the average classification accuracy. This value will be considered the classification accuracy for random selection.

B. Information Gain (IG)

Information Gain and Entropy [6], [9] are functions of the probability distribution that underlie the process of communications. The entropy is a measure of uncertainty of a random variable. Given a collection *S* of *n* samples grouped in *c* target concepts (classes), the entropy of S relative to the classification is:

$$Ent(S) = -\sum_{i=1}^{c} p_i \log_2(p_i) \tag{9}$$

, where p_i is the proportion of *S* belonging to class *i*.

Based on entropy an attribute effectiveness a measure is defined in features selection. The measure is called *Information Gain*, and is the expected reduction in entropy caused by partitioning the samples according to this attribute. More precisely, the information gain of an attribute relative to a collection of samples *S*, is defined as:

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v) \tag{10}$$

, where *Values(A)* is the set of all possible values for attribute A, and S_v is the subset of S for which attribute A has the value v.

Forman in [5] reported that Information Gain failed to produce good results on an industrial text classification problem, as Reuter's database. The author attributed this to the property of many feature scoring methods to ignore or to remove features needed to discriminate difficult classes.

C. SVM Feature Selection (SVM_FS)

Mladenic et al. [12], present a method for selecting features based on a linear support vector machine. The authors compare more traditional features selection methods, such as Odds Ratio and Information Gain, in achieving the desired tradeoff between the vector sparseness and the classification performance. The results indicate that at the same level of sparseness, features selection based on normal SVM yields better classification performances. First the authors train the linear SVM on a subset of training data and retain only those features that correspond to highly weighted components (in the absolute value, without any normalization) of the resulting hyperplane that separates positive and negative samples. The reduced feature space is then used to train a classifier over a large training set because more documents now fit into the same amount of memory. This idea was also presented in [16]. In [6] and [7] the advantages of using the same methods in the features selection step and in the learning step are explained.

Following this idea we have used the SVM algorithm, with linear kernel, for feature selection. Thus the feature selection step becomes a learning step that trains using all features calculate the (optimal) hyperplane that splits best the positive and negative samples. We obtain for each topic from the initial set the specified weight vector (the weight vector have the input space dimension) using linear kernels (multi-class classification). In contrast with Mladenic et al., we normalized all weight vectors obtained for each topic. We make an average over all weight vectors and obtain the weight vector used in the subsequent step. Using this weight vector we select only the features with a weight with an absolute value greater then a specified threshold.

D. Genetic Algorithm for Feature Selection (GA-FS)

Genetic algorithms encode a potential solution to a specific problem on a simple chromosome-like data structure and apply genetic operators to these structures so as to preserve critical information [22, 25]. In our feature selection problem the chromosome is considered to be of the following form:

$$c = \left(w_1, w_2, ..., w_n, b\right) \qquad (11)$$

where w_i, $i = \overline{1, n}$ represent the weight for each feature, and b represent the bias of the hyperplane of SVM. We consider that the training set has the form $\left\{\left\langle \vec{x}_i, y_i \right\rangle, i = 1, ..., m\right\}$, where y_i represents the output for the input sample $\vec{x}_i$, and it can only take -1 and +1. We chose this form of chromosome to facilitate using of SVM for fitness function, keeping into the chromosome the parameters that are modified into SVM decision function (equation 8). Thus potential solutions to the problem encode the parameters of the separating hyperplane, w and b. In the end of the algorithm, the best candidate from all generations gives the optimal values for separating hyperplane orientation w and location b. Following the idea proposed for multi-class classification ("one versus the rest"), we try to find the best chromosome for each of the 24 considered Reuters topics. For each topic we start with a generation of 100 chromosomes, each of them having values randomly generated between -1 and 1.

Using the SVM algorithm with linear kernel $\left\langle w, x \right\rangle + b$ we can compute the fitness function for each chromosome. The evaluation through the fitness function is defined as:

$$f(c) = f((w_1, w_2, ..., w_n, b)) = \left\langle \mathbf{w}, \mathbf{x} \right\rangle + b, \qquad (12)$$

where **x** represents the current sample and *n* represents the number of features. In the next step we generate the next population using selection, crossover or mutation [22].

The evolutionary process stops after a predefined number of steps are taken or when in the last 20 steps no change occurs.

At the end of the algorithm, we obtain for each topic the best chromosome that represents the decision function. We then normalize each weight vector in order to obtain all weights between 0 and 1. For selecting the best features we make an average over all those 24 obtained weight vectors and select the features according to their descending weights. The developed method is detailed in [14]. As far as we know, we are the first authors proposing a feature selection method using Genetic Algorithms with SVM for calculating fitness function and a simplified chromosome structure.

IV. Experimental Framework

A. The Dataset

Our experiments are performed on the Reuters-2000 collections [20], which have 984Mb of newspapers articles in a compressed format. Collection includes a total of 806,791 documents, with news stories published by Reuters Press covering the period from 20.07.1996 through 19.07.1997. The articles have 9822391 paragraphs and contain 11522874 sentences and 310033 distinct root words. Documents are pre-classified according to 3 categories: by the *Region* (366 regions) the article refers to, by *Industry Codes* (870 industry codes) and by *Topics* proposed by Reuters (126 topics, 23 of them contain no articles). Due to the huge dimensionality of the database we will present here results obtained using a subset of data. From all documents we selected the documents for which the industry code value is equal to "System software". We obtained 7083 files that are represented using 19038 features and 68 topics. We represent documents as vectors of words, applying a stop-word filter (from a standard set of 510 stop-words) and extracting the word steam. From these 68 topics we have eliminated those topics that are poorly or excessively represented. Thus we eliminated those topics that contain less than 1% documents from all 7083 documents in the entire set. We also eliminated topics that contain more than 99% samples from the entire set, as being excessively represented. The elimination was necessary because with these topics we have the risk to use only a single decision function for classifying all documents ignoring the rest of the decision functions. After doing so we obtained 24 different topics and 7053 documents that were split randomly in training set (4702 samples) and evaluation set (2531 samples). In the feature extraction part we take into consideration both the article and the title of the article in order to create the characteristic vector.

B. Kernel Types

The idea of the kernel is to compute the norm of the difference between two vectors in a higher dimensional space without representing those vectors in the new space. In practice we can see that by adding a constant bias to the kernel involves better classifying results. In this work we present result using a new idea to correlate this bias with the dimension of the space where the data will be represented. More information about this idea can be found in our previous work [15]. We consider that those two parameters (the degree and the bias) need to be correlated in order to improve the classification accuracy.

We'll present the results for different kernels and for different parameters for each kernel. For the polynomial kernel we vary the degree and for the Gaussian kernel we change the parameter C according to the following formulas (*x* and *x'* being the input vectors):

- Polynomial

$$k(x, x') = \left(2 \cdot d + \left\langle x \cdot x' \right\rangle\right)^d \qquad (13)$$

d being the only parameter to be modified

- Gaussian (radial basis function RBF)

$$k(x,x') = \exp\left(-\frac{\|x-x'\|^2}{n \cdot C}\right) \tag{14}$$

C being the classical parameter and *n* being the new parameter, introduced by us, representing the number of elements from the input vectors that are greater than 0.

As linear kernel we used the polynomial kernel with degree 1. For feature selection with SVM method we used only the linear kernel.

C. Correlating Kernel's Parameters

Usually when learning with a polynomial kernel researchers use a kernel that can be expressed as like $\left(\langle \mathbf{x} \cdot \mathbf{x}' \rangle + b\right)^d$ where *d* and *b* are independent parameters. Parameter "*d"* is the kernel degree and it is used as a parameter that helps mapping the input data into a higher dimensional space. Thus, this parameter is intuitive. The second parameter "*b"* (the bias), is not so easy to infer. In all studied articles, the researchers used a nonzero *b*, but they didn't present a method for selection it. We notice that if this parameter was eliminated (i.e., chosen to be zero) the quality of the results can be poor. It is logically that there is a need to correlate the parameters *d* and *b* because the offset *b* needs to be modified as the dimension of the space modifies. Due to this, based on running laborious classification simulations presented in [13], we suggest the best correlation is "*b=2*d"*.

Also for the Gaussian kernel we modified the standard kernel used in the research community given by formula $k(x,x') = \exp\left(-\|x-x'\|^2 / C\right)$, where the parameter C is a number witch usually takes values between 1 and total numbers of features. We introduce the parameter *n* that multiply the usually parameter *C* with a value that represents the number of distinct features having weights greater than 0 that occurs into the current two input vectors, decreasing substantially value of *C* (see equation 14). As far as we know, we are the first authors proposing a correlation between these two parameters for both polynomial and Gaussian kernels.

D. Representing the Data

Because there are many ways to define the feature-weight, we represent the input data in different formats, and we try to analyze their influence on the classification accuracy. We take in consideration three formats for representing data [2]. In the following formulas *n(d, t)* is the number of times that term *t* occurs in document *d, n(d,τ)* is the maximum frequency occurring in document *d*.

- **Binary representation** – in the input vector we store "0" if the word doesn't occur in the document and "1" if it occurs.
- **Nominal representation** – we compute the value of the weight using the formula:

$$TF(d,t) = \frac{n(d,t)}{\max_\tau n(d,\tau)} \tag{15}$$

- **Cornell SMART representation** –we compute the value of the weight using the formula:

$$TF(d,t) = \begin{cases} 0 & if\ n(d,t) = 0 \\ 1 + \log(1 + \log(n(d,t)) & otherwise \end{cases} \tag{16}$$

V. Experimental Results

A. Feature Selection for Multi-Class Classification

In text classification problems, usually we have a great number of features that are extracted from the dataset in order to create the input vector. Usually many of those features are rather irrelevant in classification. These features don't generally improve the accuracy of the classification and only increase the training and testing time and the memory requirement. These features are generally considered noise and some methods for reducing the number of these features are used.

For a fair comparison between the four feature selections methods used, we need to use the same number of features. For the Information Gain method the threshold for selecting the features represents a value between 0 and 1. For the other three methods the threshold represents the number of features that we want to obtain. This number must be equal with the number of features obtained through Information Gain method.

In what follows we present the influence of the number of features regarding to the classification accuracy for each input data representation and for each feature selection method, considering 24 distinct classes. We present results only for a numbers of features smaller or equals to 8000. In [13] we show that for a numbers of features greater than 8000 the classification accuracy doesn't increase, sometimes even decreases. Also in [13] we present results for different value of kernel degrees. A similar comparison was presented by Mladenic in [11] and [12].

The classification performance, as it can be observed from Fig. 1 and Fig. 2, is not improved when the number of features increases (especially for SVM_FS). We notice that there is a slight increase in the accuracy when we raise the percentage of features from the initial set from 2.5% (475 features) to 7% (1309 features) for polynomial kernel. The accuracy doesn't increase for a larger percentage of selected features. More than this, if more than 42% of the features were selected, the accuracy slightly decreases [13]. This can occur because the additional features can be noisy. As we expected, for Random features selection the value of the accuracy is very poor in comparison with the other methods. The other methods, Information Gain, SVM_FS and GA_FS obtained comparable results. SVM_FS has slightly better results in comparison with IG (Fig. 1, Fig. 2) for polynomial kernels and obtains best results using a small number of features.

The SVM algorithm depends on the order of selecting input vectors, finding different optimal hyperplanes when the input data are selected in different order. Genetic algorithm with SVM fitness function stipulates this in feature selection step. The SVM_FS and GA_FS obtained comparable results but

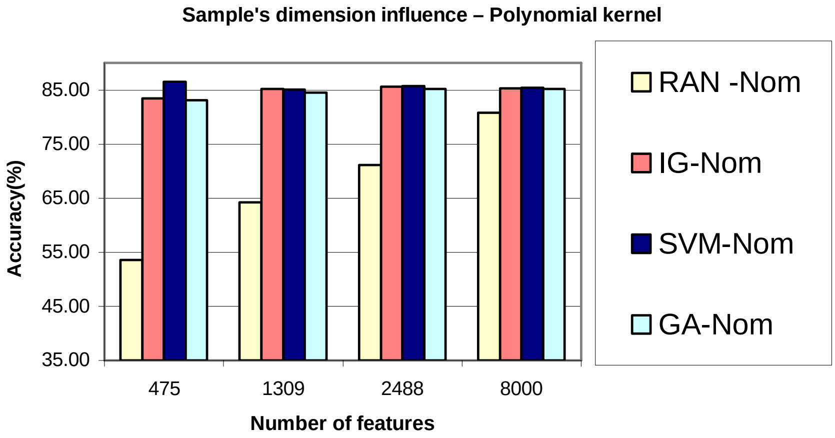

Fig. 1 Influence of the number of features on the classification accuracy using Polynomial kernel with degree equal to 2 (Nom– means nominal representation)

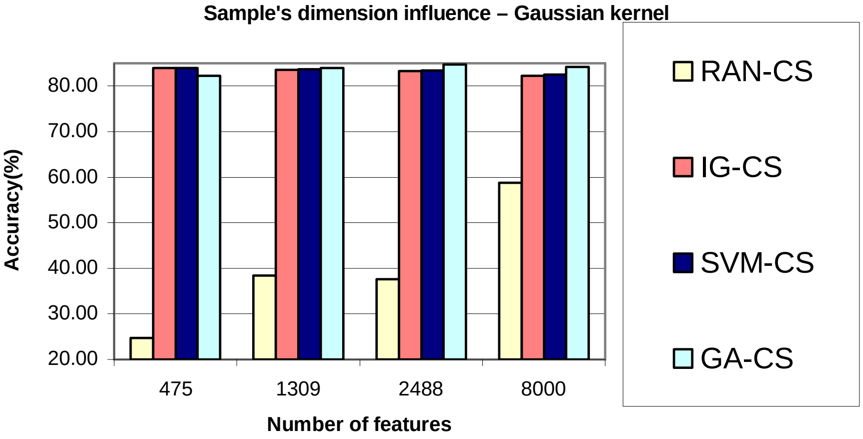

Fig. 2 Influence of the number of features on the classification accuracy using Gaussian kernel with parameter C equal to 1.3 (CS – means Cornell Smart representation

there are better comparatively with Information Gain.

In Fig. 1 the influence of number of features in classification accuracy obtained for all feature selection methods are presented here only for Nominal data representation. In the classification step we use the SVM algorithm with polynomial kernel degree 2 and Nominal data representation. In Fig. 2 are presented results obtained using Gaussian kernel with parameter *C*=1.3 and Cornell Smart data representation. As can be observed GA_FS obtain better results with Gaussian kernel comparatively with others three methods. So, GA_FS is better in average with 1% comparatively with SVM_FS (84.27% for GA_FS comparatively with 83.19% for SVM_FS) and with 1.7% comparatively with IG (84.27% for GA_FS and 82.58% for IG). For polynomial kernels SVM_FS obtain in average better results with 0.9% comparatively with IG (from 86.24% for SVM_FS to 85.31% for IG) and with 0.8% comparatively with GA_SVM (from 86.24% to 85.40%). In almost all cases the best results are obtained for a small numbers of features (in average for 1309).

In the Fig. 3 we present the training classification time as a function of selected numbers of features for each feature selection method and polynomial kernel with degree 2. As it can be observed, the timing increases with about 3 minutes when the features increases from 475 to 1309 and with about 32 minutes when the features increases from 1309 to 2488 for

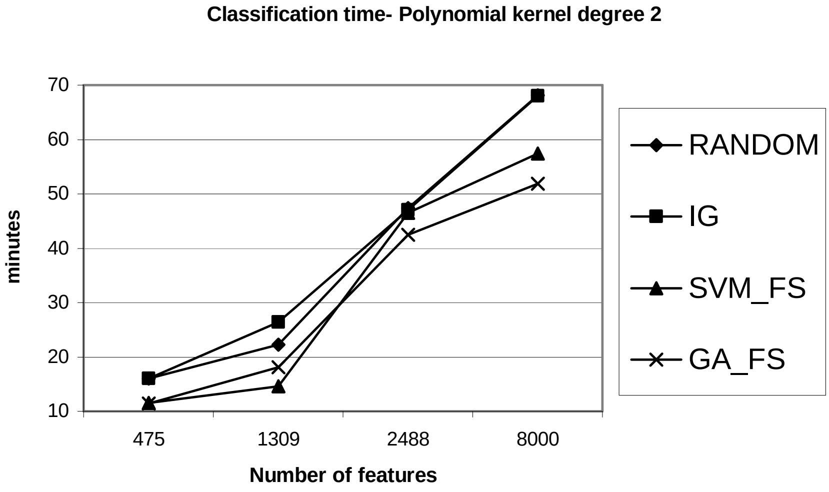

Fig. 3 Learning classification time as a function of selected numbers of features

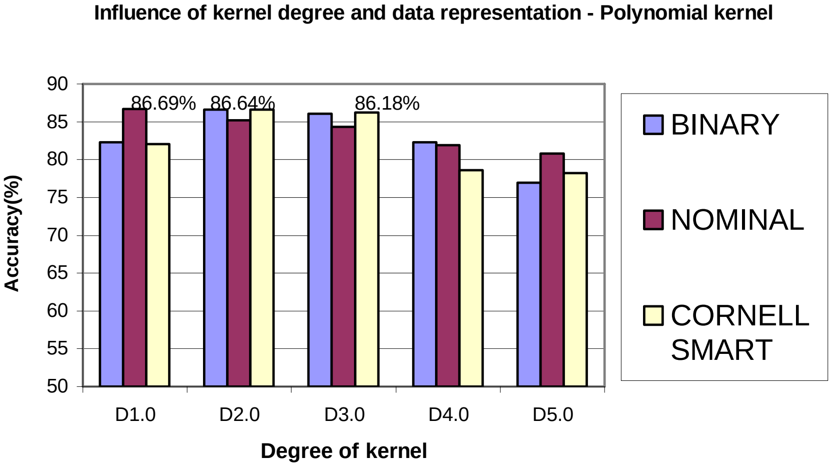

Fig. 4 Influence of data representation and degree of the kernel for polynomial kernel

SVM_FS method. Also the time needed for training with features selected using IG is usually greater than the time needed for training with features selected with SVM_FS. When number of selected features increases the best classification time was obtained of GA_FS method.

For Gaussian kernel the time is in average (for all made testes) with 20 minutes greater then time needed for training polynomial kernel for all features selected with IG, SVM_FS or GA_FS. The numbers are given for a Pentium IV at 3.4 GHz, with 1GB DRAM and 512KB cache, and WindowsXP.

B. Influence of Kernel Degree and Data Representation

For extending the SVM algorithm from two-class classification to multi-class classification typically one of two methods is used: "*One versus the rest*" that was presented above and "*One versus the one*", where a separate classifier is trained for each pair of topics. The Reuter's database contains strongly overlapping classes and assigns almost all samples in more than one class. Therefore we chose the first method for multi-class classification. Also we tested the method "*one versus the one*", but the obtained results are not as good. Also the training time doesn't decrease so much because there are more decision functions to be learned even for small datasets.

In the training phase for each topic a decision function is learned. In the evaluating phase each sample is tested with each decision function and is classified in the class having the greatest absolute value. The obtained results are compared

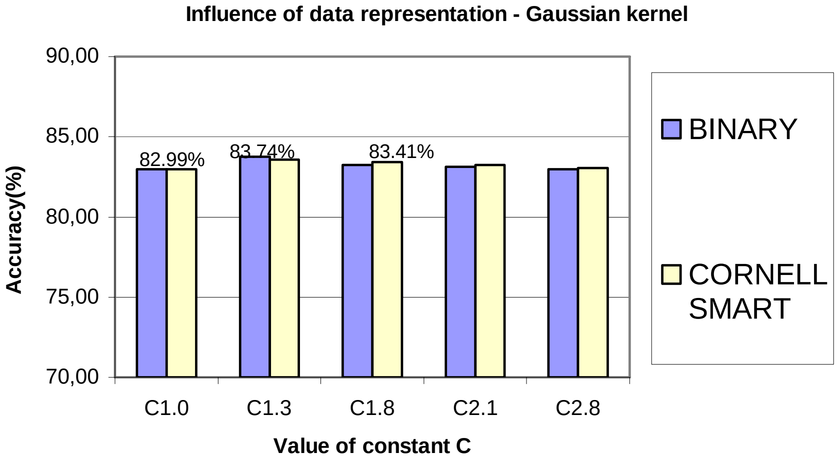

Fig. 5 Influence of data representation and constant C for Gaussian kernel

with the known Reuter's classification.

In order to find a good combination of kernel type, kernel degree and data representation we run ten tests, five tests for a polynomial kernel with a kernel degree between 1 and 5, and respectively five tests for Gaussian kernel with different values for the parameter C (1.0, 1.3, 1.8, 2.1 and 2.8). In [13] we report additional results. In Fig. 4 we present results and obtained for polynomial kernel and SVM_FS method with a data set with 1309 features, which was proven to be the best number (see Fig. 1, 2).

Fig. 4 shows that text files are generally linearly separable in the input space (if the input space has the right dimensionality) and the best results were obtained for a linear kernel and for a small kernel degree using a nominal representation of the input data.

TABLE I
AVERAGE ACHIEVED OVER ALL DATA SETS TESTED FROM POLYNOMIAL KERNEL AND NOMINAL REPRESENTATION

Method / Nr. features	Random	IG	SVM_FS	GA_FS
475	44.56	82.91	**85.03**	81.94
1309	51.02	84.62	**84.90**	83.91
2488	60.57	84.54	**85.02**	84.90
8000	78.63	84.72	82.42	**85.01**

TABLE II
AVERAGE ACHIEVED OVER ALL DATA SET TESTED FOR GAUSSIAN KERNEL

Method / Nr. features	Random	IG	SVM_FS	GA_FS
475	25.26	83.27	83.31	81.93
1309	38.39	83.33	**83.39**	**83.41**
2488	39.49	83.07	83.02	**84.02**
8000	56.61	82.20	82.42	**83.32**

For the same dimension of the feature space (1309 features), the Information Gain method achieved an average accuracy (computed for all three types of data representation) of 84.62% in comparison with the SVM_FS method that achieved an average accuracy of 84.90% and GA_FS that obtain only 83.91% (for Random selection method an average accuracy of 51.02% was achieved). In Table I we present all averages accuracies obtained and we can observe that the SVM_FS method obtains better results for each dimension of the data set. Also we can observe that the average accuracy doesn't increase when the dimension of the set increases (especially for SVM_FS). The GA_FS obtain the best results only for 8000 selected features. The SVM_FS method obtains best results with a small dimension of the features space (85.03% for 475 features) in comparison with IG that needs more features (8000 features for 84.72%) for obtain the best results. GA_FS needs also 8000 features for obtain best results 85.01%.

In Table II we compute the average over all tested values for the Gaussian kernel. For Gaussian kernel GA_FS method the results are better comparatively with SVM_FS, and in both case the results are greater than results obtained with IG or Random selection (see Table I and Table II, IG and Random columns). In comparison with results obtained using the polynomial kernel the results obtained using Gaussian kernel are smaller, whatever of feature selection method used.

In Fig. 5 we present results obtained for Gaussian kernel for two types of data representation and for five distinct value of parameter C, using a data set with 1309 features obtained with SVM_FS method. Into Gaussian kernel Fig. 5 we add a parameter that represents the number of elements greater then zero (parameter "n" from equation 12). Nominal representation (equation 14) represents all weight values between 0 and 1. When parameter "*n*" is used, all the weights become very close to zero involving very poor classification accuracies (for example, due to its almost zero weight, a certain word really belonging to the document, might be considered to not belong to that document). So we don't present here the results obtained using the nominal representation.

In all presented results when we used SVM technique (in feature selection step with SVM and classification step) we used kernels presented above with correlating parameters (*d* and *b* for polynomial kernel and *n* with input vectors for Gaussian kernel). Result about kernels correlations are presented in the next section.

C. *Kernel's Influence*

In this section we present the influence of correlating kernel's parameters on classification accuracy. In order to do this we make a short comparison between the results that we obtained with usually used implementation of SVM, called LibSvm [8], and our implemented application called UseSvm [13]. LibSvm uses "one versus the one" method for multi-class classification. Our developed UseSvm program uses "one versus the rest" method, as we already mentioned. Reuter's database, used in our tests, contains strongly overlapped data and in this case the first method usually obtains poor results.

We have used only one set for these tests, set that obtains the best results in previous section (having 1309 features, obtained using SVM_FS method). In order to fairly compare LibSvm with our UseSvm, we eliminated, when possible, Reuters overlapped data (for working only on non-overlapped classes, formally: $\forall i, j = \overline{1, 13}, \mathbf{C}_i \cap \mathbf{C}_j = \varnothing \; \textit{for each} \; i \neq j$). We choose for each sample only first class that was proposed by Reuters. We also eliminated classes that are poorly or excessively represented. We obtained only 13 classes randomly split in two sets and used for training and testing for both LibSvm and UseSvm. Results obtained by LibSvm are poor in comparison with the results of our application, because, despite our efforts, the data are however slightly overlapped. In the next figures we present results obtained for the polynomial kernel and the Gaussian kernel. We are using equivalent parameters for both applications. As LibSvm has more parameters than UseSvm, we have left on default value the parameters that appear only in LibSvm.

As we already specified, for polynomial kernel our suggestion was to make "*b=2*d*" (see Section IV.C). We present results using LibSvm with *b=0* (default value) respectively with *b=2*d* (specified explicitly by command line) comparing with our UseSvm program.

As it can be observed from Fig. 6, our UseSvm program obtains far better results than the well-known LibSvm (with an average gain of 18.82% better). By comparing LibSvm with the default bias with LibSvm with modified bias (according to our formula), we noticed that the modified bias leads to better results (with an average gain of 24.26% better). The average gain is computed as average obtained by LibSvm with the default bias divided by the average obtained by LibSvm with modified bias. For degree 1 were obtained similar results because values of default bias and value computed using our formula are quite equal.

For the Gaussian kernel simulations, presented in Fig. 7, our suggestion was to multiply the constant C with a parameter *n* (like we already explained in Section IV.C). It is difficult to give this parameter from the LibSvm's command line because *n* is computed dynamically and LibSvm have only one parameter that can be modified, called *gamma*. More precisely, LibSvm uses $gamma = 1/n$ only when gamma is default (*n* means the average of number of attributes in the input data). For LibSvm we have used *gamma* as $1/C$. For *LibSvm+"gamma"* we considered "*gamma*" to be equal to $2/nC$, where *n* is the number of features. The single case of equivalence between these two programs (LibSvm and UseSvm) is obtained for the default value of *gamma* in LibSvm and respectively for *C*=1 in UseSvm. This case is presented separately as "def" in Fig. 7.

As it can be observed, using our idea to modify the Gaussian kernel the results obtained using LibSvm are better in comparison with results obtained using LibSvm with standard kernel (with an average gain of 28.44%). Our UseSvm program obtains far better results than the well-known LibSvm (with an average gain of 25.57% better). For the default parameter of LibSvm our application also has obtained better results (76.88% in comparison with 69.97% for LibSvm).

VI. Conclusions and Further Work

In this paper, we investigated whether feature selection methods can improve the accuracy of document classification. Four types of feature selection methods were tested and three types of input data representations were used. Simulations were developed using a powerful classification technique based on kernels, i.e. the Support Vector Machine. In the case of multi-class classification, the best results were obtained when we chose a small (but relevant) dimension of the data set. After selecting relevant features, we showed that using between 2.5% to 7% from the total number of features, the classification accuracies are significantly better (with a maxim of 86.69% for SVM_FS method, polynomial kernel and Nominal data representation). If we further increase the number of features to more than 10%, the accuracy does not improve or even decreases. When we used SVM_FS, better classification accuracy is obtained using a small number of features (85.28%, for 475 features representing about 3% from the total number of features)-needing small training time. Generally speaking, the SVM_FS and GA_FS methods were better than IG and Random methods and both obtain comparable results. We have also observed that the polynomial kernel obtains better results when we used a nominal data representation and the Gaussian kernel obtains better results when we used Cornell Smart data representation. The best accuracy was obtained by the polynomial kernel with

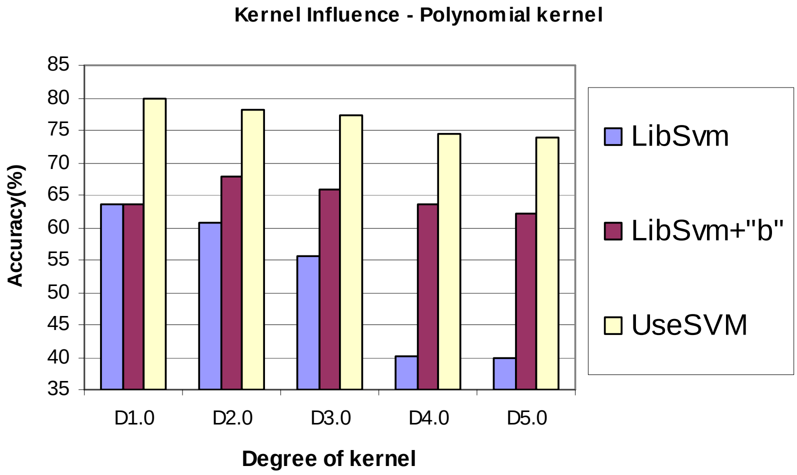

Fig. 6 Influence of correlation between parameters from polynomial kernel

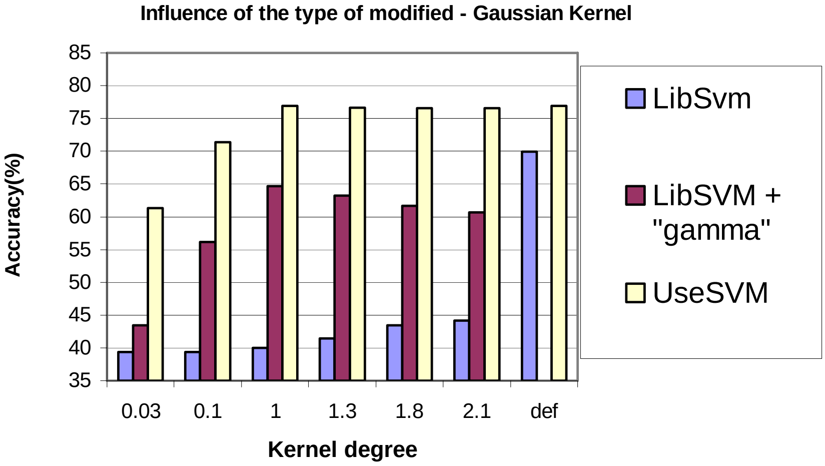

Fig. 7 Influence of modified about Gaussian Kernel

a degree of one (86.69% for nominal representation) in comparison with Gaussian kernel that obtained only 84.85% accuracy for C=1.3 and Cornell Smart representation. The GA_FS method obtains best results for a greater numbers of features (8000). Also we showed that the training classification time increases only by 3 minutes, as the number of features increases from 485 to 1309 and increases by 32 minutes when number of features increases from 1309 to 2488. As far as we know, we are the first authors proposing a feature selection method using Genetic Algorithms with SVM for calculating fitness function and a simplified chromosome structure.

We have proposed also an interesting method to better correlate kernel's parameters. The method correlates the degree of the Polynomial kernel with the bias respectively correlates the constant from the Gaussian kernel with a value that represents the number of distinct features that occurs into the currently used vectors and having weights greater than 0. Through this method we obtained an average accuracy classification gain of 24.26% for polynomial kernel, respectively 28.44% for Gaussian kernel. As far as we know, we are the first authors proposing a correlation between these two parameters for both polynomial and Gaussian kernels.

Work is ongoing to classify larger text data sets (the complete Reuters database). In this work we want to develop a pre-classification of all documents, obtaining fewer samples (using simple algorithms like Linear Vector Quantization or Self Organizing Maps). After that we'll use the obtained samples as entry vectors for the already developed features selection and classification methods.

Because almost all available data from a real world are in fact unlabeled data, we will try to combine classifying method with a clustering method, also based on SVM, in order to use labeled and unlabeled data into a hybrid classification algorithm. An interesting natural extension of our algorithm is to be used into a Web mining application to extract and categorized online news.

ACKNOWLEDGEMENTS

We would like to thank to SIEMENS AG, CT IC MUNCHEN, Germany, especially to Mr. Vice-President Dr. H. C. Hartmut RAFFLER, for his generous and various supports, both professional and material, that he has provided in developing this work.

REFERENCES

[1] Bhatia S., Selection of Search Terms Based on User Profile, ACM Explorations, pages 224-233, 1998.

[2] Chakrabarti S.: Mining the Web- Discovering Knowledge from hypertext data, Morgan Kaufmann Press, 2003.

[3] Chih-Wei Hsu, Chih-Chang Chang and Chih-Jen Lin, A Practical Guide to Support Vector Classification, Department of Computer Science and Information Engineering National Taiwan University, 2003 (Available http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide).

[4] Douglas H., Ioannis T., Constantin A.: A Theoretical Characterization of Linear SVM-Based Feature Selection, Proceedings of the 21st International Conference on Machine Learning, Banff, Canada, 2004.

[5] Forman G.: A Pitfall and Solution in Multi-Class Feature Selection for Text Classification, Proceedings of the 21st International Conference on Machine Learning, Banff, Canada, 2004.

[6] Jebara T. and Jaakkola T.: Feature selection and dualities in maximum entropy discrimination, Uncertainty in Artificial Intelligence 16, 2000.

[7] Jebara T.: Multi Task Feature and Kernel Selection for SVMs, Proceedings of the 21st International Conference on Machine Learning, Banff, Canada, 2004.

[8] http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[9] Mitchell T.: Machine Learning, McGraw Hill Publishers, 1997.

[10] Mladenic D., Feature Subset Selection in Text Learning, Proceedings of the 10th European Conference on Machine Leraning (ECML-98), pages 95-100, 1998.

[11] Mladenic D., Grobelnik M.: Feature selection for unbalanced class distribution and naïve bayes, In Proceedings of the 16th International Conference on Machine Learning ICML, p.258-267, 1999.

[12] Mladenic D., Brank J., Grobelnik M., Milic-Frayling N.: Feature Selection Using Support Vector Machines The 27th Annual International ACM SIGIR Conference (SIGIR2004), pp 234-241, 2004.

[13] Morariu D., Classification and Clustering using Support Vector Machine, 2nd PhD Report, University „Lucian Blaga“ of Sibiu, September, 2005, http://webspace.ulbsibiu.ro/ daniel.morariu/html/Docs/ Report2.pdf.

[14] Morariu D., Relevant characteristics extraction from semantically unstructured data, 3rd PhD Report, University “Lucian Blaga” of Sibiu, September, 2006, http://webspace.ulbsibiu.ro/ daniel.morariu/html/ Docs/Report3.pdf.

[15] Morariu D., Vintan L.: A Better Correlation of the SVM kernel’s Parameters, Proceeding of the 5th RoEduNet International Conference, Sibiu, June 2006.

[16] Morariu D., Vintan L. Tresp V.: Feature selection methods for an Improved SVM Classifier, Proceeding of the 3rd International Conference on Intelligent Systems, ICIS06, Prague, August, 2006.

[17] Morariu D., Vintan L. Tresp V.: Evolutionary Feature Selection for Text Documents using the SVM, Proceeding of the 3rd International Conference on Neural Networks and Pattern Recognition, NNPR06, Barcelona, October, 2006.

[18] Nello C., John Swawe-Taylor: An introduction to Support Vector Machines, Cambridge University Press, 2000

[19] Platt J.: Fast training of support vector machines using sequential minimal optimization. In B. Scholkopf, C. J. C. Burges, and A. J. Smola, editors, Advances in Kernel Methods – Support Vector Learning, pages 185-208, Cambridge, MA, 1999, MIT Press.

[20] Reuters Corpus, Volume 1, English Language, 1996-08-20 to 1997-08-19. Available through http://about.reuters.com/researchandstandards /corpus/. Released in November 2000.

[21] Schoslkopf B., Smola A.: Learning with Kernels, Support Vector Machines, MIT Press, London, 2002.

[22] Smith, J.E., Eiben, A.E., Introduction to evolutionary computing, Springer-Verlag, 2003

[23] Vapnik V.: The nature of Statistical learning Theory. Springer, New York, 1995.

[24] Yang Y., J.O. Pedersan,: A Comparative Study on Feature Selection in Text Categorization, Proceedings of ICML, 14th International Conference of Machine Learning, pages 412-420, 1997.

[25] Whitely, D., A genetic Algorithm Tutorial, Foundation of Genetic Algorithms, ed. Morgan Kaufmann