Machine Learning, 46, 423–444, 2002

Text Categorization with Support Vector Machines. How to Represent Texts in Input Space?

EDDA LEOPOLD edda.leopold@ais.fraunhofer.de
JÖRG KINDERMANN joerg.kindermann@ais.fraunhofer.de
GMD German National Research Center for Information Technology, Institute for Autonomous intelligent Systems, Schloss Birlinghoven, D-53754 Sankt Augustin, Germany

Editor: Nello Cristianini

Abstract. The choice of the kernel function is crucial to most applications of support vector machines. In this paper, however, we show that in the case of text classification, term-frequency transformations have a larger impact on the performance of SVM than the kernel itself. We discuss the role of importance-weights (e.g. document frequency and redundancy), which is not yet fully understood in the light of model complexity and calculation cost, and we show that time consuming lemmatization or stemming can be avoided even when classifying a highly inflectional language like German.

Keywords: support vector machines, text classification, lemmatization, stemming, kernel functions

1. Introduction

It is known that support vector machines (SVM) are capable of effectively processing feature vectors of some 10 000 dimensions, given that these are sparse. Several authors have shown, that support vector machines provide a fast and effective means for learning text classifyers from examples. Documents of a given topic could be identifyed with high accuracy (Joachims, 1998; Dumais et al., 1998).

Topic identification with SVM implies a kind of semantic space in the sense that the learned hyperplane separates those documents in the input space, which belong to different topics.[1] However the relation between the content of a document, its input vector, and the geometry induced by the feature space it is not fully understood for the time being.

Another question which has to be answered is: What kind of linguistic units should be counted to obtain an input vector from a document? Linguistically spoken: What is the appropriate level of analysis that represents the categories, which are to be learned? An example: If we map each document on a 26-dimensional input vector which consists of the document's letter frequencies we will certainly not be able to separate documents in any feature space because all documents are mapped to the same point, which is invariant to the content of the document. For practical purposes of topic identification the question of the appropriate level of analysis boils down to: Should we use some suffix stripping procedure to reduce words to their stems or should we consider the words-forms as they appear in the running text?

In this paper we study different mappings of frequencies to input space, and combine these mappings with different kernel functions. We also compare different kinds of linguistic preprocessing. Since English is a language with a very limited inventory of grammatical endings, we conduct our experiments not only on English but also on German data. The corpora we used for our experiments are the Reuters data set and two German newspapers.

By "lemmatization" we address a procedure which maps word-forms to a standardized lexicon entry. This procedure is similar to "stemming" namely the reduction of words to their stems. For our German material we used the lemmatizer Morphy. Morphy was designed by Lezius, Rapp, and Wettler (1998) and is freely available on the web.

Throughout this paper the words "term" and "type" are used as synonyms as well as the words "document" and "text". Types are the lexical units under consideration and as we study the effect of lemmatization, types may be either word-forms, lemmas (word stems) or tagged word-forms, i.e. word-forms with a number of tags which indicate their grammatical function. With the word "token" we refer to the types as they occur in the running text. So the number of tokens of a given type in a text is the frequency of that type.

The number of documents in the document collection is denoted by N. The number of types in the whole document collection equals the dimensionality of input space and is therefore designated by n. The number of occurrences of term w_k in document t_i is denoted by $f(w_k, t_i)$ and $f(w_k) = \sum_i f(w_k, t_i)$ is the number occurrences of term w_k in the whole document collection. The term-frequency vector of document t_i is $\mathbf{f_i} = (f(w_1, t_i), \ldots, f(w_n, t_i))$. The number of tokens of a text is also referred to as its length. The asterisk "$*$" is used as a symbol for componentwise multiplication of vectors. So that if $\mathbf{z} = (z_1, \ldots, z_n)$ and $\mathbf{v} = (v_1, \ldots, v_n)$ the asterisk-operation yields $\mathbf{z} * \mathbf{v} = (z_1 v_1, \ldots, z_n v_n)$.

2. Representations of texts in input space

In this paper we study different weighting schemes for the representation of texts in input space. Each of the mappings of text to input space consists of three parts. First the term-frequencies are transformed by a bijective mapping. The resulting vector is multiplyed by a vector of importance weights, and this is finally normalized to unit length. We tested 10 different combinations of frequency transformations, importance weights and normalizations and three different kernel functions.

2.1. Frequency transformations

The well known Zipf-Mandelbrot law (Mandelbrot, 1953) states that the fraction of words with frequency f in a text is given by

$$\alpha(f) = \frac{1}{f^{\gamma-1}} - \frac{1}{(f+1)^{\gamma-1}} \propto \frac{1}{f^{\gamma}}. \tag{1}$$

Here γ is a positive parameter, which usually (i.e. in large homogenious texts) adopts a value of about 2. Several variants of this law can be found in the literature.[2] All of them

have in common that the distribution of frequencies of terms in texts is extremely uneven. Some units occur very often, whereas as a rule of thumb half of the terms—the so called hapax legomena—occur only once. Unfortunately especially the infrequent units contain highly specific information about the content of the text.

A consequence of Zipf's law is that frequencies in large texts scale to different orders of magnitude. For example the two most frequent types in a month of the German Newspaper Frankfurter Rundschau are the articles "die" and "der" occurring 111989 and 108736 times respectively, whereas 94923 types occur only once. In small texts however type frequencies cannot differ to that extend. In small documents of about 100 words nearly all types occur once whereas the maximum type frequency is typically below 10.

As frequencies of lexical units scale to different order of magnitude in larger documents, it is interesting to examine how term frequency information can be mapped to quantities which can efficiently be processed by SVM. For our empirical tests we use the following transformations of type frequencies.

- raw frequencies
- logarithmic frequencies
- inverse frequencies

As the simplest "frequency-transformation" we use the identity i.e. the term-frequencies $f(w_k, t_i)$ themselves. The frequencies are multiplied by one of the importance weights described below and normalized to unit length. Raw frequencies $f(w_k, t_i)$ with importance weight *idf* (see Eq. (7)) normalized with respect to L_2-norm have been used by Joachims (1998) and others. We also tested other combinations, for example raw frequencies with no importance weight normalized with respect to L_1-norm.

The second transformation, which as far as we know is novel to SVM text categorization, is the logarithm. We consider this transform because it is a common approach in quantitative linguistics to consider logarithms of linguistic quantities rather than the quantities themselves. Here we also normalize to unit length with respect to L_1 and L_2. We define the vector of logarithmic type frequencies of document t_i by

$$\mathbf{l}_i = (\log(1 + f(w_1, d_i)), \ldots, \log(1 + f(w_1, d_i))). \tag{2}$$

Logarithmic frequencies are combined with different importance weights. They are normalized with respect to L_1 and L_2.

In order to distribute term frequencies evenly on the interval on the interval [0; 1] we derive a third transformation of term frequencies. To this end we require that transformed frequencies of terms of adjacent ranks differ by the same amount. So the transformation $g(f)$ has to satisfy

$$g(f(r)) - g(f(r + 1)) \equiv \text{const.}, \tag{3}$$

where $f(r)$ is the frequency of the type of rank r. Zipf-Mandelbrot law implies the rank frequency distribution

$$f(r) = \left(\frac{A}{B+r}\right)^{\frac{1}{\gamma-1}} \quad A, B, \gamma > 0. \tag{4}$$

Inserting Eq. (3) into Eq. (4) yields

$$g\left(\frac{A}{B+r}\right) - g\left(\frac{A}{B+r+1}\right) \equiv \text{const.},$$

which is fulfilled for

$$g(f) = a - \frac{\gamma}{f} \tag{5}$$

for some real numbers a and γ. In order to map zero frequencies to a well defined value we add a positive number $\gamma*$ to the denominator and require $a = 1$ and $\gamma = \gamma^*$ so that term frequencies are mapped to the unit interval. We therefore define the inverse frequency of the k-th term $f(w_k)$ by

$$F_{hyp}(w_k, d_i) = 1 - \frac{\gamma}{f(w_k, d_i) + \gamma}, \tag{6}$$

where $\gamma > 0$ is a parameter. We use $\gamma = 1$ in the following.

2.2. *Importance weights*

Importance weights are often used in order to reduce dimensionality of a learning task. Common importance weights like inverse document frequency (idf) have originally been designed for identifying index words (Salton & McGill, 1983). And feature extraction in text retrieval is often thought in terms of reduction of dimensionality of the input space.

However, since SVM can manage high dimensional feature spaces effectively, reduction of dimensionality is not necessary, at least not from a computational viewpoint. However importance weights can be used to quantify how specific a given type is to the documents of a text collection. A type which is evenly distributed across the document collection should be given a low importance weight because it is judged to be less specific for the documents it occurs in. A type which is used in only a few documents should be given a high importance weight. The importance weight of a type is multiplied by its transformed frequency of occurrence. So each of the importance weights described below can be combined with each of the frequency transformations described in Section (2.1). We examined the performance of the following settings

- no importance weight
- inverse document frequency (idf)
- redundancy

For SVMs, inverse document frequency is the most commonly used Importance weight. The inverse document frequency of type w in a text collection consisting of N documents is defined by

$$idf_k = \log \frac{N}{df_k} \tag{7}$$

where df_k is the number of those documents in the collection, which contain the term w_k. The vector of inverse document frequencies for all types in the document collection is given by

$$\mathbf{idf} = (idf_1, \ldots, idf_n)$$

Idf has the advantage of being easy to calculate. Its disadvantage is that it disregards the frequencies of the terms within each document. A term which occurs once in all documents except one, but occurs a hundred times in the one remaining document, should be judged as important for the classification of documents, because it is highly specific to one document. In contrast a term which occurs 20 times in all documents with no exception is not specific for any document and is not useful for training and testing a classification procedure. This affects mainly high frequency terms and collections of larger documents. Redundancy quantifies the skewness of a probability distribution, and r_k is a measure of how much the distribution of a term w_k in the various documents deviates from the uniform distribution.

We therefore consider the empirical distribution of a type over the documents in the collection and define the importance weight of type w_k by

$$r_k = \log N + \sum_{i=1}^{N} \frac{f(w_k, d_i)}{f(w_k)} \log \frac{f(w_k, d_i)}{f(w_k)}, \tag{8}$$

where $f(w_k, d_i)$ is the frequency of occurrence of term w_k in document t_i and N is the number of documents in the collection. This yields a vector of importance weights for the whole document collection:

$$\mathbf{r} = (r_1, \ldots, r_n). \tag{9}$$

Equation (8) is the usual definition of redundancy. But as far as we know redundancy has never been used in *this* way. A similar weighting scheme has been proposed by Bookstein and Swanson (1974). They used the probability of the classical occupancy problem, in which exactly $f(w_k)$ balls are distributed at random into N urns.

The advantage of redundancy over inverse document frequency is that it does not simply count the documents a type occurs in but takes into accoount the frequencies of occurrence in each of the document. The difference between redundancy and idf is the larger the larger the documents are.

2.3. *Normalization*

Texts differ in their lengths. Long texts contain several thousands of words whereas short texts consist of some dozen words. Type-frequencies have to be normalized in order to make texts of different length comparable. From the standpoint of quantitative linguistics this is not a trivial task (Orlov, 1982; Margulis, 1993). The best solution is to devide type-frequency by the total number of tokens in the text. This is equivalent to mapping the type-frequency vector to the unit-sphere in the L_1-sense:

$$\mathbf{f} \to \frac{\mathbf{f}}{\|\mathbf{f}\|_{L_1}}$$

In the following we refer to this mapping by L_1-normalization.

From the standpoint of the SVM learning algorithm the best normalization rule is the L_2-normalization because it yields the best error bounds. L_2-normalization has been used by Joachims (1998) and Dumais et al. (1998). It is defined by

$$\mathbf{f} \to \frac{\mathbf{f}}{\|\mathbf{f}\|_{L_2}}.$$

2.4. *Kernel functions and weighting schemes*

It is well known that the choice of the kernel function is crucial to the efficiency of support vector machines. Therefore the data transformations described above were combined with three different kernel functions:

- linear kernel $K(\vec{x}, \vec{x}') = \vec{x} \cdot \vec{x}'$
- 2nd order polynomial kernel $K(\vec{x}, \vec{x}') = (\vec{x} \cdot \vec{x}')^2$
- Gaussian rbf-kernel $K(\vec{x}, \vec{x}') = e^{-\|\vec{x}-\vec{x}'\|/2\sigma^2}$

In our experiments 30 combinations of kernel functions, frequency transformations, importance weights, and text-length normalization were tested. The following list summarizes the combinations of kernels and its numbering is used throughout the rest of the paper. Each of the 10 transformations is combined with each of the three kernels listed above. So (1) addresses relative frequency with linear kernel, (5) stands for logarithmic frequencies normalized with respect to L_1 and combined with polynomial kernel, and (9) means raw frequency with redundancy and L_1-normalization combined with Gaussian rbf-kernel.

(1)–(3) relative frequency (*relFreq*) given by

$$\mathbf{x}_i = \frac{\mathbf{f}_i}{\|\mathbf{f}_i\|_{L_1}}.$$

This transformation is considered because from the standpoint of text statistics it is the most natural way to make frequencies of texts with different lengths comparable. As far as we know this transformation is not used for SVM-text categorization.

(4)–(6) logarithmic frequencies normalized with respect to L_1 (*logRelFreq*)

$$\mathbf{x}_i = \frac{\mathbf{l}_i}{\|\mathbf{l}_i\|_{L_1}},$$

(7)–(9) raw frequency with redundancy normalized with respect to L_1 (*relFreqImp*)

$$\mathbf{x}_i = \frac{\mathbf{f}_i * \mathbf{r}}{\|\mathbf{f}_i * \mathbf{r}\|_{L_1}}.$$

Here "*" denotes the componentwise multiplication and $\mathbf{r}$ is the vector which consists of the redundancies of all types in the document collection as defined in Eq. (9).

(10)–(12) logarithmic frequencies with redundancy normalized with respect to L_1 (*logRelFrqImp*).

$$\mathbf{x}_i = \frac{\mathbf{l}_i * \mathbf{r}}{\|\mathbf{l}_i * \mathbf{r}\|_{L_1}}$$

(13)–(15) tfidf normalized with respect to L_1 refered to as (*tfidf*)

$$\mathbf{x}_i = \frac{\mathbf{f}_i * \mathbf{idf}}{\|\mathbf{f}_i * \mathbf{idf}\|_{L_1}}$$

This is the tfidf wheighting scheme as described by Salton and McGill (1983).

(16)–(18) tfidf normalized with respect to L_2 refered to as (*tfidfL2*).

$$\mathbf{x}_i = \frac{\mathbf{f}_i * \mathbf{idf}}{\|\mathbf{f}_i * \mathbf{idf}\|_{L_2}}$$

This transformation has been used for SVM by Thorsten Joachims (1998) and Dumais et al. (1998).

(19)–(21) raw frequencies normalized with respect to L_2 (*relFreqL2*).

$$\mathbf{x}_i = \frac{\mathbf{f}_i}{\|\mathbf{f}_i\|_{L_2}}.$$

(22)–(24) logarithmic frequencies normalized with respect to L_2 (*logRelFreqL2*).

$$\mathbf{x}_i = \frac{\mathbf{l}_i}{\|\mathbf{l}_i\|_{L_2}}$$

(25)–(27) raw frequencies with redundancy normalized with respect to L_2.(*relFreqImpL2*)

$$\mathbf{x}_i = \frac{\mathbf{f}_i * \mathbf{r}}{\|\mathbf{f}_i * \mathbf{r}\|_{L_2}},$$

(28)–(30) logarithmic frequencies with redundancy normalized with respect to L_2 refered to as (*logRelFreqL2Imp*)

$$\mathbf{x}_i = \frac{\mathbf{l}_i * \mathbf{r}}{\|\mathbf{l}_i * \mathbf{r}\|_{L_2}}$$

We also tested the following two transformations (see Eq. (6)), which are motivated by Zipf's law.
(31) Inverse term Frequency: $f_{hyp}(w_k, t_i) = 1 - \frac{1}{f(w_k,t_i)+1}$
(32) Inverse term frequency with redundancy:

$$f_{hyp,imp}(w_k, d_i) = \left(1 - \frac{1}{f(w_k, d_i) + 1}\right) \cdot r(w_k).$$

These novel transformation could not be combined with polynomial kernel or rbf-kernel, because SVM light did not converge. The results with linear kernel are disapointing and are not reported in the following.

3. The role of linguistic preprocessing

Text classification requires a considerable amount of preprocessing (figure 1). Frequency-lists have to be generated from the textual input, words have to be mapped to their stems, and importance weights have to be calculated. Linguistic preprocessing, in particular, lemmatization) takes the lion's share of computation time when applying SVM to a text classification problem. Furthermore lemmatization is a procedure which strongly depends on the language. For each language a special lemmatizer has to be constructed.

Since Support Vector Machines have the potential to manage high dimensional input spaces effectively, the need for time consuming linguistic preprocessing (i.e. removal of

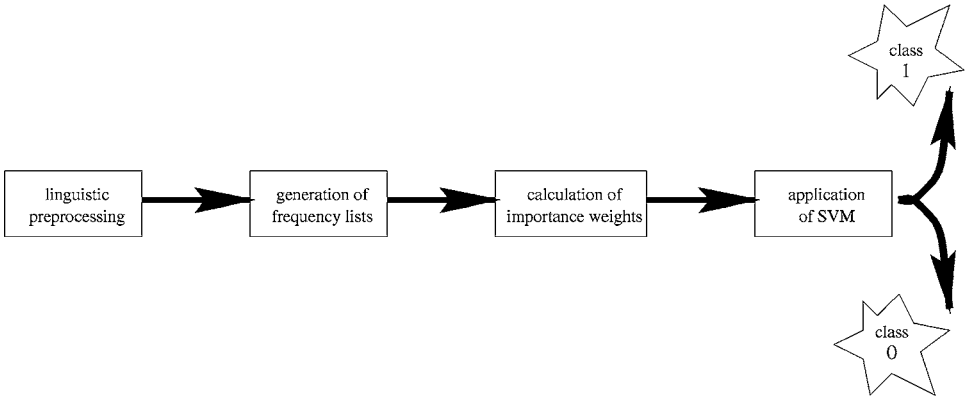

Figure 1. The process of text classification.

Table 1. Example of a list of lemmas word-forms and tagged word-forms.

Lemmas	Word-forms	Tagged word-forms
Der	Der	Der_ART_DEF_NOM_SIN_MAS
Algengürtel	Algengürtel	Algengürtel SUB NOM SIN MAS
Vor	Vor	Vor PRP DAT
Der	Der	Der_ART_DEF_DAT_SIN_FEM
Norwegisch	Norwegischen	Norwegisch_ADJ_DEF_DAT_SIN_FEM
Küste	Küste	Küste_SUB_DAT_SIN_FEM
Haben	Hat	Haben_VER_3_SIN
Sich	Sich	Sich_REF_AKK_SIN_3
Gestern	Gestern	Gestern_ADV
Erneut	Erneut	Erneut_ADV
Um	Um	Um_PRP_AKK
25	25	25_ZAN
Kilometer	Kilometer	Kilometer_SUB_AKK_PLU_MAS
Vergrößern	Vergrößert	Vergrößern_VER_PA2

low frequency words, lemmatization, and application of importance weights) can be put in question.

The lemmatizer, which we used for German, performs a morphological analysis of each word. If there are different readings the most probable is choosen. For about 9% of the running words in our German material the morphological analysis was not successful. For German this is an acceptable rate of successful lemmatization. The unlemmatizable word-forms, which could not be analyzed by the lemmatizer, were treated in two different ways. We generated one data set were the unlemmatizable forms were deleted and another data set where unlemmatizable word-forms were *not* deleted. Furthermore we generated a data set where the grammatical tags, which resulted from the morphological analysis were appended to the word-forms (see Table 1).

Lemmatization and tagging were only performed on the German material. We used the results of Joachims (1998) as a reference for SVM-performance on stemmed English documents. We expected that the effect on precision and recall would be larger in the case of German, because of the morphological richness of German compared to English. The lemmatization procedure resulted in the reduction of the dimensionality of the input of about 60%.

Tagged word-forms were used since we observed that lemmatization can lead to a loss of performance in terms of precision and recall. Our hypothesis was that an improvement of performance could be achieved by providing SVM with as much information as possible. In German many word-forms are grammatically polyfunctional in the sense that e.g. different cases of a noun are represented by the same word-form. The same holds for other parts of speech and for most of the inflecting languages. We therefore resolved grammatical ambiguities of ambiguous word-forms by adding the grammatical tags of the lemmatizer

to the word-forms. For those cases where the lemmatizer could not recognize a word-form the word-form was used without any tagging information. We obtained some 300-thousand tagged word-forms for each newspaper corpus. So tagging increases the number of types by about 150%. Table 1 shows a sentence of the Taz-newspaper. The ambiguity of the article "der" in the first and the fourth line is resolved in the third column (nominative masculine vs. dative feminine). In the left column the word-forms "hat" (Engl. "has") is mapped to its basic form "haben" (Engl. "to have").

In order to measure the effect of lemmatization on the performance of SVM-text categorization it is convenient to remove unlemmatizable words from both the lemmatized material and the unlemmatized data. When unlemmatizable word forms are deleted, the difference between the results on the lemmatized and the unlemmatized input can be directly attributed to lemmatization procedure i.e. the mapping of word-forms to their lemma. However this kind preprocessing is not suitable for practical applications because potentially relevant information is unnecessarily discarded.

We generated two further data sets of lemmas and word-forms where unlemmatizable forms were *not* deleted. Comparing these both data sets shows how much retrieval performance *can benefit* from lemmatization. It does not show the *pure effect* of lemmatization, since many word-forms are not mapped to their lemmas—namely those which are not recognized by the lemmatizer. Keeping unrecognizable wordforms in the type-frequency lists makes the lemmatized input comparable to the results on English reported by Joachims (1998). To examine the effect of lemmatization and tagging we studied five different combinations of linguistic preprocessing.

- no lemmatization i.e. running words are directly taken as input
- no lemmatization but unlemmatizable words are deleted
- lemmatization where unlemmatizable words are kept
- lemmatization where unlemmatizable words are removed
- tagged word-forms

4. The data

We considered three document collections for empirical tests. The full texts were used as input. Neither stop-words nor low-frequency words were removed from the input. An exception was made in the case of raw frequencies with *idf* normalized with respect to L_2. Here words occurring once or twice were removed in order to make our results comparable to those obtained by Joachims (1998). Documents shorter than 10 words—mostly titles of tables, caricatures, and figures—were not considered as proper texts. They were discarded.

The first document collection under consideration was the Reuters-21578 dataset compiled by David Lewis and originally collected by the Carnegie group from the Reuters newswire in 1987. The "ModApte" split was used leading to a corpus of 8762 training documents, 3009 test documents, and about 1.52 million running words.

The second data collection was obtained from the German daily newspaper "Frankfurter Rundschau" (FR for short) We considered the issues of July 1998 which consist of 11974 documents and about 3.47 million running words. Training set and test sets were obtained randomly. The training set consisted of 7983 documents and the test set consisted of 3991

Table 2. Statistics of the corpora under consideration.

	# Tokens (1000)[a]	# Types (1000)[a]	TTR	Percent hapaxes	Number of texts	Text length
Reuters: word-forms	1520	28.2	53.9	24.4	12902	117.8
FR: word-forms with[a]	3413	228560	14.9	52.9	11974	290.0
FR: lemmas with[a]	3413	178911	19.1	53.8	11974	263.9
FR: word-forms without[a]	3141	137041	22.9	47.4	11974	290.0
FR: lemmas without[a]	3141	87410	35.9	46.1	11974	263.9
FR: plus tags	3472	301.0	11.3	55.6	11974	290.0
Taz: word-forms with[a]	3200	200.3	16.0	53.0	10539	303.6
Taz: plus tags	3200	308.4	10.2	58.9	10539	303.6

[a]With or without unlemmatizable word-forms.

documents. Classes were provided by respective topics in the document collection. We used the five most frequent of the 41 topics in the material. These classes are general news "NAC", economy "WIR", sports "SPO", regional news "LRL", and local affairs "LOK". They covered nearly 75% of the data. We drew 5 data sets from the FR-newspaper: word-forms with unlemmatizable words, lemmas with unlematizable words, word-forms without unlemmatizable words, lemmas without unlemmatizable words, and tagged word-forms (Table 2).

The third document collection was the German daily newspaper "Die Tageszeitung" (Taz for short). The issues June 1988, August 1988, and June 1991 consisted of 10539 texts and about 3.2 million running words. In order to test if SVM are capable of classifying opaque classes we considered different years of publication. So one class of the newspaper consists of the issues of June and August 1988 and the other class consists of the issues of June 1991. Both training set and test set are obtained randomly from these three months of newspaper. The training set contains 7070 articles and the test set comprises 3469 articles. If an article is signed by an author, this signature is removed in order to avoid that different authors in 1988 and 1991 facilitate a keyword-like learning and classifying. The reason why we chose this classification task was the following: if humans classify newspaper articles according to their date of appearance, they refer to implicit world-knowledge which cannot easily be made explicit. Classes in the Taz corpus are not defined by some keywords.

In spite of being a common benchmark for machine learning tasks, the Reuters dataset cannot be considered as a "representative" text-collection. First: text length of Reuters dataset seems to be very unnatural. Most of the news texts consist of less 100 than words, and the distribution in the Reuters dataset displays a sharp cutoff at a document length of 100 running words (see figure 2). The documents in the newspapers are in average more than twice as long as the documents in the Reuters collection (see figure 3). Moreover, the length of articles in the newspapers is more smoothly distributed. It shows a log-normal or Poisson-like pattern, which is often observed in length distributions of linguistic units. For example the distribution of word length often follows a modified Poisson distribution or a log-normal distribution (Grotjahn, 1982; Wimmer et al., 1994; Balasubrahmanyan & Naranan, 1996).

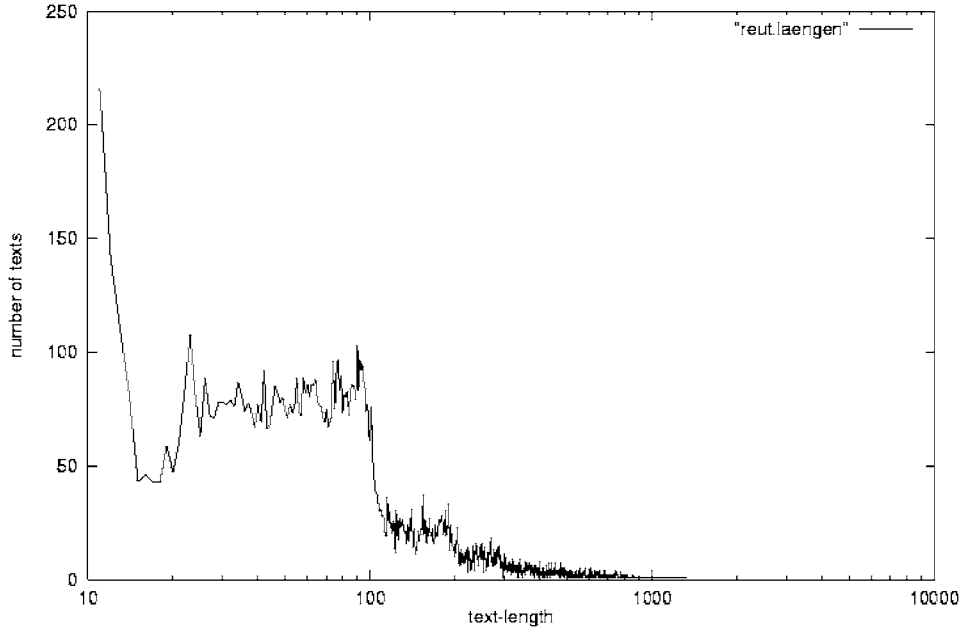

Figure 2. Distribution of document length in Reuters.

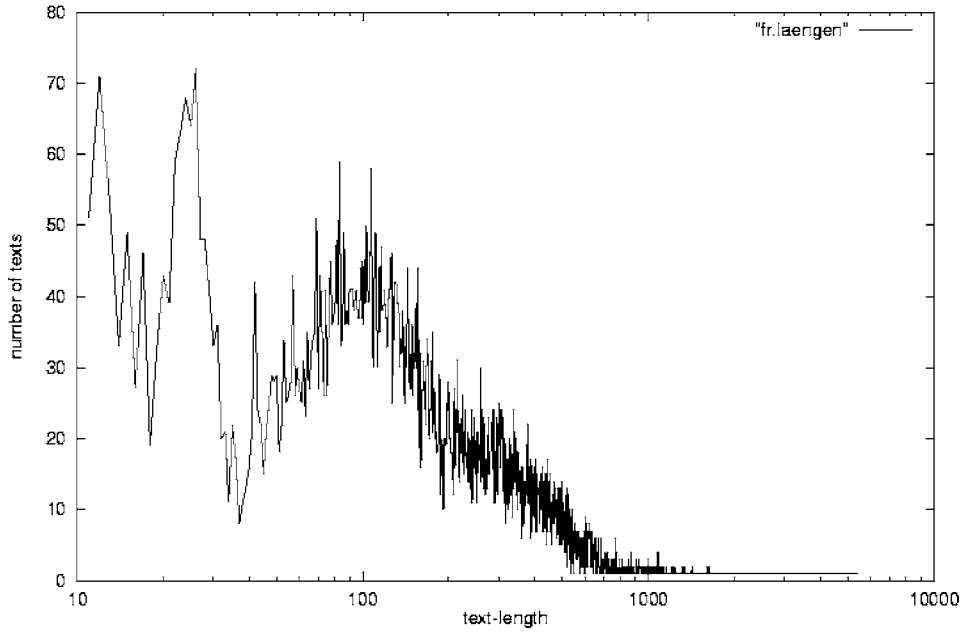

Figure 3. Distribution of document length in Frankfurter Rundschau.

Another reason why the Reuters dataset is not representative for common natural language is its very restricted use of vocabulary. The number of hapaxes (i.e. words occurring only once) is only 24.4% of the types and thus much smaller than usual. Usually about half of the types in a large homogenous text occur only once. Correspondingly the slope of spectrum ($\gamma = 1.476$) is much smaller than 2 which is usually observed in natural language texts (Zipf, 1935; Krylov, 1995; Balasubrahmanyan & Naranan, 1996). The type-token ratio (TTR) in the Reuters dataset is very different from what is normally found in natural language texts. The same holds for the percentage of hapaxes, which usually amounts to about 50%. This reveals that the difference in the corpora consists in the different use of infrequent words. In the newspapers words are used as creatively as in large homogeneous texts, whereas Reuters shows a standardized use of language.

Therefore it is not surprising that Joachims (1998) as well as Nigam et al. (1999) observed a strong correspondence between a small set of words (like their title words) and the document's topic in Reuters. Texts in the Reuters corpus are constructed in a way that they can be quickly classified by humans according to their title and the first sentence in the text body.

We use German material because German is a highly inflectional language. Having observed that SVM-classifiers do not need preprocessing (stemming and elimination of stop-words) for English texts, we want to verify this also in the case of a morphologically richer language. This is especially interesting because stemming is even more difficult and costly if the a language has more grammatical endings and greater morpho-syntactical complexity. We hope that the results obtained for German can be transfered to other inflecting languages as well.

5. Empirical results

As described above the five most frequent classes of Reuters and FR were used in the experiments. On the Taz newspaper only one classification task was performed: the discrimination of volume 1988 against volume 1991. For all our experiments we used the computer program SVM-light written by Thorsten Joachims, which is available at `http://ais.gmd.de/~thorsten/svm-light/`. In all experiments the precision/recall break-even points as well as the number of support vectors were calculated.

5.1. The effect of lemmatization

We used the lemmatizer "Morphy" for lemmatization of the German material. It was written by Lezius, Rapp, and Wettler at the university of Paderborn (Lezius, Rapp, & Wettler, 1998). Morphy is available on the Internet at `http://www-psycho.uni-paderborn.de/lezius/`. The lemmatization of both newspapers was 91% correct.

In Table 3 the precision/recall break-even for transformations (17), and (28) is displayed. Unlemmatizable word-forms are not deleted neither from the word-forms nor from the lemmas or tagged word-forms. The values in the first column of Table 3 are obtained by using the state of the art procedure namely (*tfidfL*2) of lemmas, where non-analyzable forms are not removed. The results for the Reuters data set, which are displayed in the first column of Table 3 have been obtained by Joachims (1998) One can see that the lemmatized

Table 3. The effect of lemmatization on the performance of transformations (17) and (28).

	(*tfidf L2*) + poly			*logRelFreqL2Imp* + linear		
Category	Lemmas	Plain	Tagged	Lemmas	Plain	Tagged
Earn	98.4	98.37	–	–	98.75	–
Acq	94.6	94.28	–	–	94.05	–
Money	72.5	70.50	–	–	66.20	–
Grain	93.1	89.55	–	–	91.11	–
Crude	87.3	86.25	–	–	86.25	–
LOK	82.89	84.70	83.71	85.12	86.51	87.68
LRL	91.02	90.82	90.48	91.49	91.94	91.53
NAC	84.19	84.84	82.95	85.33	85.25	85.24
SPO	89.24	89.20	88.26	88.24	90.62	88.11
WIR	86.89	87.22	82.42	87.43	86.96	83.73
Taz	83.05	82.92	76.25	84.49	84.64	78.83

data yields slightly better results when the tfidf weighting scheme is used. The picture changes if another frequency transformation is used. The collumns 4 to 6 show the results for logarithmic frequencies with redundancy (transformation (28)), which yields the best microaveraged performance over all German categories as can be seen in figures 4 and 5. When transformation (28) is used the word-forms perform slightly better. One can see that the difference between the the results for lemmas and word-forms varies depending on the classes under consideration. This suggests that one hardly can decide wether word-forms or lemmas yield better precision/recall results.

The results obtained from the tagged data (presented in the third and sixth column) are worse than the results based on word-forms or lemmas. This is not only the case for the transformations and kernels shown in Table 3 but for all combinations tested. Since tagging (as we performed it) requires even more effort than lemmatization, the resolution of grammatically ambiguous word-forms seems to be the wrong way for text categorization with SVM.

If one wants to measure the pure effect of lemmatization on the performance of the learning algorithm, one has to remove those words-forms which refused to the morphological analysis from both the word-form data and the lemmatized data. A comparison of these both datas-sets is displayed in figure 6. For each data transformation the micro-average difference between the German lemmatized and un-lemmatized material is displayed. Most transformations show an advantage between 1% and 3% of micro-averaged precision/recall break-even in favor of the unlemmatized text. However some transformations with rbf-kernel and L_2-normalization perform better on lemmas.

When unlemmatizable words are not removed the differences between the results for word-forms and lemmas become smaller and as a general tendency the picture changes in favour of the lemmatized data (see figure 7). The reason for this is that the data-sets of

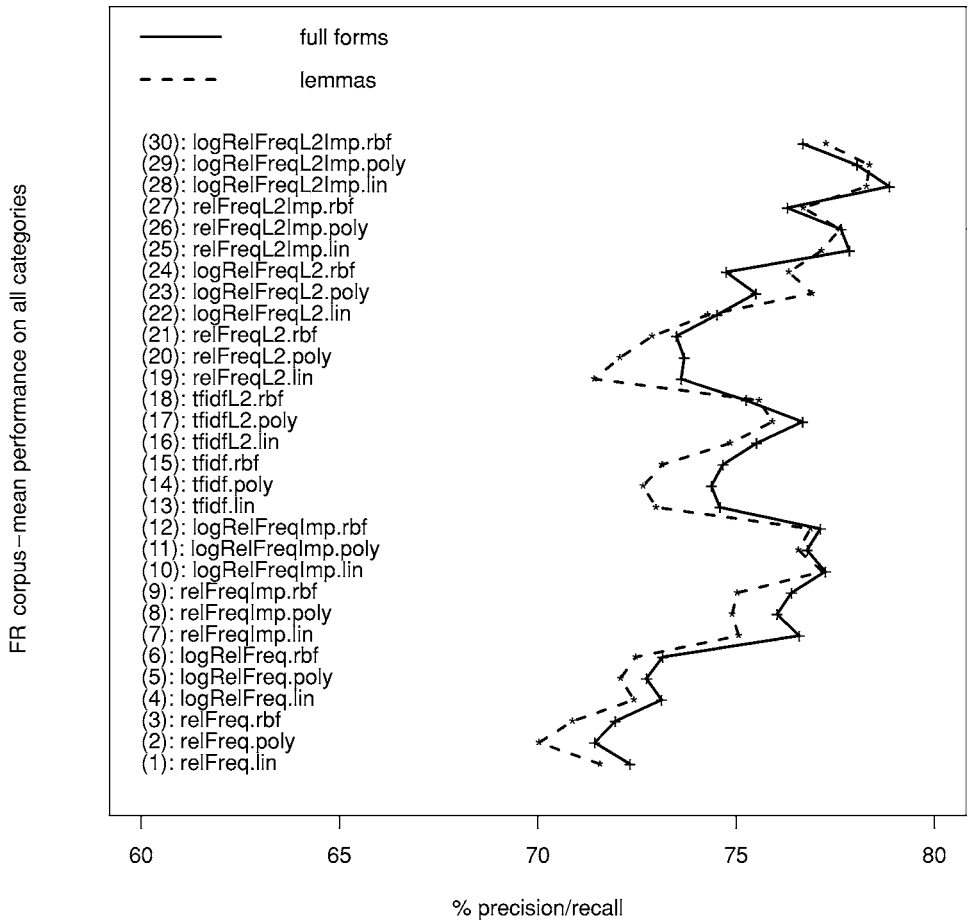

Figure 4. The micro-averaged performance of kernels and transformations on word-forms and lemmas, where un-lemmatizable word-forms are not deleted.

lemmas and word-forms are more similar than in the case where unlemmatizable forms were deleted.

Lemmatization does not lead to an improvement of performance irrespective of removal of unlemmatizable word-forms. This can be explained by the fact that it is not allways easy to decide wether morphemes fulfill a semantic or grammatical function in the language. Even the plural-morpheme can produce a semantic difference. Stricker et al. (2000) give an example for French: The word "action" in the sentence "Le judgement est plus nuancé selon le domaine d'*action* du governement." the word "action" can be translated with action. However in the sentence "Den Danske Bank a acquis en décembre dernier 90% des *actions* de Fokus Bank." the word "actions" means shares.

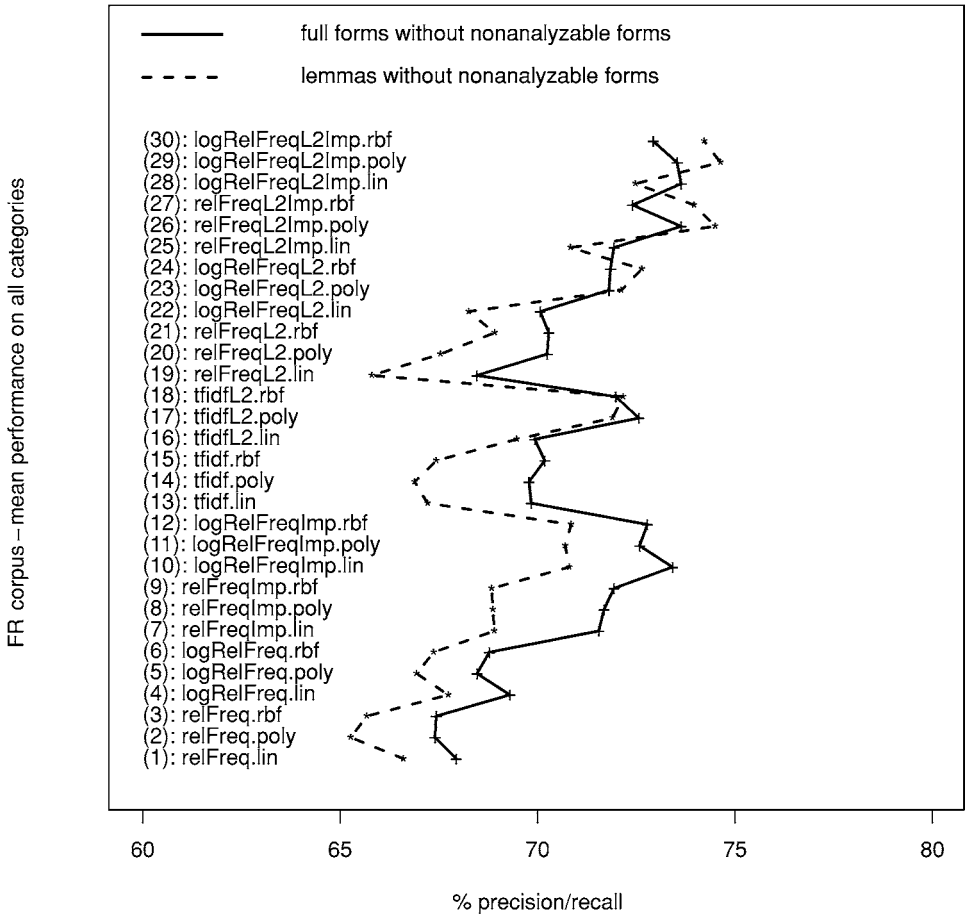

Figure 5. The micro-averaged performance of on lemmas and word-forms, where unlemmatizable forms are deleted.

The situation can become even more complicated. For example in the German sentence "Mir ist kalt." (I am freezing) and "Ich bin kalt." (I am heartless) the meaning of "kalt" (cold) changes depending on the case of the personal pronoun (mir = I + dative vs. ich = I + nominative). This example also suggests that the usual removal of stop words (pronouns are typically judged as stop words) can influence the the content of a document and can therefore have an impact on retrieval performance. Sometimes it is difficult to say wether a word is a content-word that caries meaning or a functional-word that solely fulfills a grammatic function. Since lemmatization is the slowest process when applying SVM to text-classification problems, we conclude that considering word-forms instead of lemmas leads to a substantial reduction of processing time without any loss of acuracy.

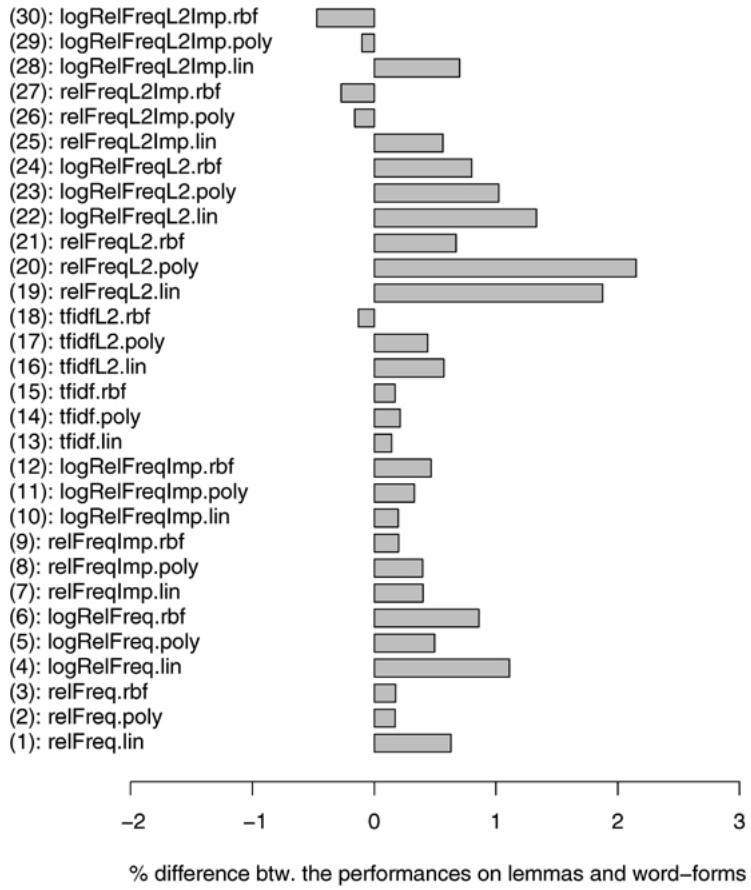

Figure 6. The micro-averaged difference of performance on lemmas and word-forms, where unlemmatizable word-forms are not removed.

5.2. *The effect of the different data transformations*

Having observed that most weighting schemes yield slightly better results when no lemmatization is performed (see figure 6), it is interesting to compare the results of different transformations. Figures 4 and 5 show the results obtained for all 30 transformations. The micro-average of precision/recall break-even over all classes in the Frankfurter Rundschau is displayed. On can see that the L_1-transformations perform worse than the transformations (1) to (15) with L_2-normalization.

The best data-transformation in terms of precision/recall is logarithmic frequency with redundancy and L_2-normalization (transformations (29) and (30)). The usual transformation—tfidf with L_2-normalization yields medium results. As a tendency we observed that Euclidian normalization is the better (compared to L_1) the longer the texts and the larger the exponent γ of the spectrum (see Eq. (1)).

5.3. *The effect of different kernels*

In all of our experiments the application of different kernel-functions did not affect precision/recall performance very much. However as a general tendency we observed that rbf-kernels generate more support vectors than polynomial kernels and that the polynomial kernels generate more support vectors than linear kernels.

Figures 6 and 7 show an interesting pattern concerning the relation between linguistic preprocessing and kernel functions. One can see that for the weighting schemes with L_2-normalization (transformation (16) to (30)) rbf-kernels tend to perform better on the lemmatized data than polynomial or linear kernels.

This corresponds to the number of dimensions of the induced feature space. Since the number of different word-forms is much larger than the number of different lemmas, the

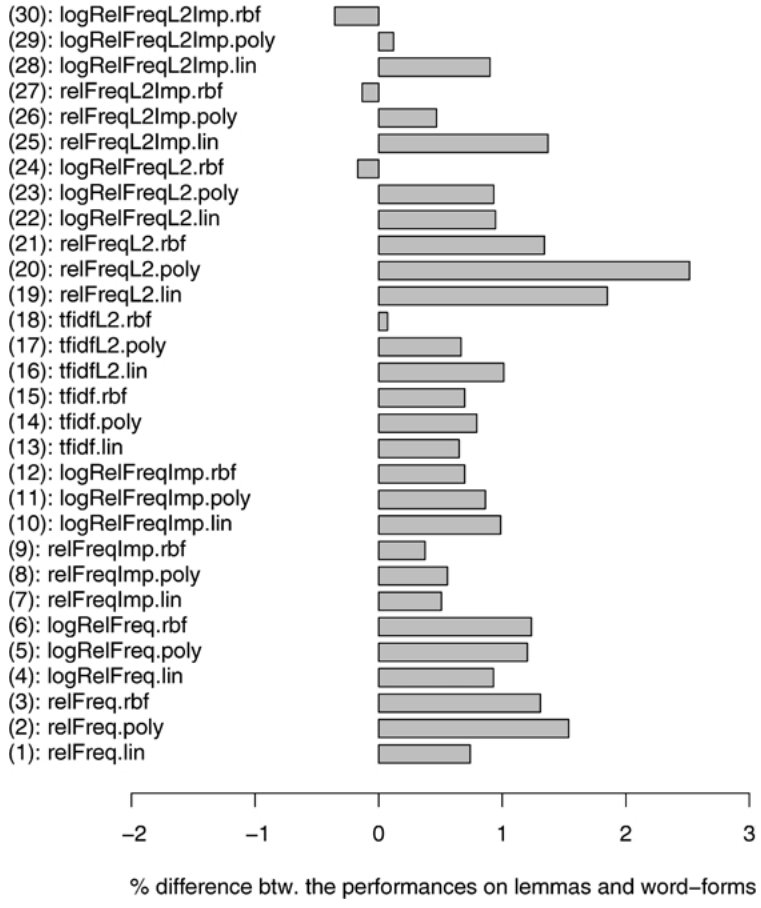

Figure 7. The micro-averaged difference of performance on lemmas and lemmatizable word-forms.

higher dimensionality of the feature space is compensated by the lower dimension of input vectors.

5.4. *The effect of importance weights*

From the results of micro-averaged precision/recall break-even, which are displayed in figures 4 and 5 one can see that for most frequency transformations acuracy improved when an importance weight was used. Those frequency transformations without importance weight ((1) to (6) and (19) to (24)) showed poor precision/recall performance.

From figures 4 and 5 one can also see that redundancy (transformations (10) to (12)) yields better results than inverse document frequency (transformations (13) to (15)) when L_1-normalization was used. Inverse documents transformations still perform better than the transformations without importance weight (transformations (1) to (3)). When L_2-normalization is applied the results between no importance, idf, and redundancy do not differ to that extend.

Table 4 allows a more detailed view on the effect of importance weights. Precision/recall break-even for different importance weights combined and raw frequency with L_2-normalization (transformations (19) to (21), (16) to (18), and (25) to (27)) are displayed. One can see that the results for redundancy and idf do not differ significantly for most classes. There are however some classes where redundancy yields better results than idf, namely the money category in Reuters, and the 1988-issues of Taz. The average length of Taz-documents is comparatively long (303.6 words). The documents in the Reuters money-category are longer (113.6 words in average) than in the categories earn (35.32 words) and acq (74.8 words). These results suggests that there is an advantage of redundancy over idf

Table 4. Comparison of different importance weights.

	No imp.			idf			Redundancy		
Category	lin	poly	rbf	lin	poly	rbf	lin	poly	rbf
Earn	98.47	98.66	97.62	98.27	98.37	98.65	98.47	98.37	98.37
Acq	94.25	93.93	92.85	94.87	94.28	93.62	93.78	93.46	93.00
Money	74.47	72.60	68.09	73.47	70.50	63.12	73.38	75.54	73.24
Grain	88.06	88.06	86.57	89.55	89.55	89.34	88.81	88.81	88.06
Crude	85.09	84.47	80.75	86.42	86.25	86.25	85.00	85.09	85.28
LOK	82.48	82.45	82.61	84.26	84.70	82.48	86.65	87.36	84.56
LRL	90.02	90.39	89.79	90.38	90.82	91.06	91.61	91.11	91.86
NAC	83.56	84.00	83.33	83.78	84.84	86.00	83.04	83.59	85.11
SPO	87.76	88.24	88.89	88.50	89.20	89.58	88.19	87.97	89.62
WIR	84.70	85.87	86.96	83.70	87.22	88.52	83.06	84.78	87.50
Taz	80.65	81.42	80.91	82.68	82.92	82.56	84.15	84.64	84.38

when documents become larger. This can be explained by the fact that *idf* just counts the number of documents a type occurs in. It does not take into account how frequent the type is. However larger documents show a greater variation in type-frequencies than do short documents. Therefore type-frequencies carry more information in long texts than in short documents.

6. Conclusion

Our experiments showed that support vector machines are capable of efficiently processing very high dimensional feature vectors, given that these vectors are sparse. We have used more than 300 000 types in the tagged Taz and FR corpus. Deletion of rare words and stop-words is not necessary when applying SVM to text classification both from the standpoint of efficiency and from the point of view of accuracy.

Lemmatization slightly improves precision/recall when those word-forms for which lemmatization fails are not removed, and when frequency transformatons (30), (27), and (26) are used. All other frequency transformations yield better results on the unlemmatized documents. The unlemmatized material with transformations (10) to (12) or (26) to (30) yields better results than the state of the art proceedure, namely lemmatized data and tfidf with L_2-normalization (transformations (16) to (18)).

Since lemmatization is the slowest process in the text-classification proceedure, considering word-forms instead of lemmas leads to a substantial reduction of processing time without leading to a substancial loss of precision/recall.

The resolution of grammatical ambiguity through lemmatization leads to a loss of performance. Importance weights improve the performance significantly. Redundancy defined in Eq. (8) is a better importance weight than inverse document frequency, which is commonly used. The advantage of redundancy over inverse document frequency seems to be greater for larger documents.

We do not claim that SVM are the only algorithm which allows to skip the lemmatization procedure. Nor do we know if our results can be generalized to other languages. However our results aggree with results for neural nets on French (see Stricker et al., 2000). The great advantage of SVM is that they can manage such a large number of dimensions (up to 300 000). Therefore from the stantpoint of computational efficiency no feature selection is needed for SVM. Other machine learning algorithms like C4.5, decision trees, neural nets, and Bayesian nets cannot handle more than some thousand dimensions. A comparison with these algorithms is not possible without introducing an aditional variable namely the procedure of reduction of dimensionality of input vectors. One algorithm, which could operate on the type frequency vectors obtained directly from the running text is k-nearest neighbor. This algorithm, however, scales to $\mathcal{O}(\ell \cdot m)$ in classification time, where ℓ is the number of training examples and m is the number of test examples. Because of the high computational expense in classification time k-nearest neighbor cannot be seen as an alternative to SVM in practical applications.

In future work we want to see if the results can be generalzed to other languages i.e. slavic, romance, and non-indoeuropean. If the results were positive, a generic algorithm would be found that worked well on nearly any language.

Acknowledgments

We thank Reinhard Köhler (Dept. of Computational Linguistics, University of Trier) who contributed the frequency data of "Frankfurter Rundschaug" and "Die Tageszeitung." We also thank Gerhard Paaß (GMD) and Thorsten Joachims (University of Dortmund) for fruitful discussions on the topic. We also thank the referees for valuable comments which helped to improove the paper.

Notes

1. Other constructions of semantic spaces with good theoretical foundation can be found in the literature (Rieger, 1999; Manning & Schütze, 1999).
2. See e.g. Chitashvili and Baayen, 1993 for a summary.

References

Altmann, G. (1988). *Wiederholungen in texten* [Repetitions in texts]. Bochum, Germany: Brockmeyer.

Balasubrahmanyan, V. K. & Naranan, S. (1996). Quantitative linguistics and complex system studies. *Journal of Quantitative Linguistics, 3:3*, 177–228.

Bookstein, A. & Swanson, Don R. (1974). Probabilistic models for automatic indexing. *Journal of the American Society of Information Science, 25*, 312–318.

Chitashvili, R. J. & Baayen, R. H. (1993). Word frequency distributions. In G. Altmann & L. Hřebíček (Eds.). *Quantitative Text Analysis* (pp. 46–135). Trier, Germany: wvt.

Dumais, S., Platt, J., Heckerman, D., & Sahami, M. (1998). Inductive learning algorithms and representations for text categorization. In *Proceedings of the Seventh International Conference on Information Retrieval and Knowledge Management (ACM-CIKM-98)* (pp. 148–155).

Grotjahn, R. (1982). Ein statistisches Modell für die Verteilung der Wortlänge [A statistical model for the distribution of word length]. *Zeitschrift für Sprachwissenschaft, 1*, 44–75.

Harter, S. P. (1975). A probabilistic approach to automatic keyword indexing, Part I. *Journal of the American Society for Information Science, 26*, 197–206.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the Tenth European Conference on Machine Learning (ECML '98)*, Lecture Notes in Computer Science, Number 1398 (pp. 137–142).

Kralík, J. (1977). An application of exponential distribution law in quantitative linguistics. *Prague Studies in Mathematical Linguistics, 5*, 223–235.

Krylov, Ju. K. (1995). A stationary model of coherent text generation. *Journal of Quantitative Linguistics, 2:2*, 157–167.

Lezius, W., Rapp, R., & Wettler, M. (1998). A freely available morphological analyzer, disambiguator and context sensitive lemmatizer for German. In *Proceedings of the COLING-ACL 1998* (pp. 743–747).

Mandelbrot, B. (1953). On the theory of word frequencies and on related Markovian models of discourse. In R. Jakobson (Ed.), *Structure of Language and its Mathematical Aspects, Proceedings of Symposia in Applied Mathematics* (Vol. XII, pp. 190–210). Providence, RI: American Mathematical Society.

Manning, C. D. & Schütze, H. (1999). *Foundations of statistical natural language processing*, Cambridge, MA: MIT-Press.

Margulis, E. L. (1993). Modelling documents with multiple poisson distributions. *Information Processing and Management, 29*, 215–228.

Nigam, K., McCallum, A. K., Thrun, S., & Mitchell, T. (2000). Text classification from labeled and unlabeled documents using EM. *Machine Learning, 39:1/2*, 103–134.

Orlov, Ju. K. (1982). Linguostatistik: Aufstellung von Sprachnormen oder Analyse des Redeprozesses? (Die Antinomie 'Sprache-Rede' in der statistischen Linquistik)[Linguostatistics: Establishing language norms of

analysis of the speech process (The antinomy 'language-speech' in statistical linguistics.).] In Ju. K. Orlov, M. G. Boroda, & I. S. NadarejČvili (Eds.). *Sprache, Text, Kunst. Quantitative Analysen* (pp. 1–55). Bochum, Germany: Brockmeyer.

Porter, M. F. (1980) An algorithm for suffix stripping. *Program (Automated Library and Information Systems), 14:3*, 130–137.

Rieger, B. B. (1999). Semiotics and computational linguistics. On semiotic cognitive information processing. In Zadeh, L. A. & J. Kacprzyk (Eds.). *Computing with words in information/intelligent systems I. foundations* (pp. 93–118). Heidelberg, Germany: Physica.

Salton, G. & McGill, M. J. (1983). *Introduction to modern information retrieval*. New York: McGraw Hill.

Stricker, M., Vichot, F., Dreyfus, G., & Wolinski F. (2000). Vers la conception automatique de filtres d'informations efficaces [Towards the automatic design of efficient custom filters]. In *Reconnaissance des Formes et Intelligence Artificielle (RFIA '2000)* (pp. 129–137).

Vapnik, Vladimir N. (1998). *Statistical learning theory*. New York: Wiley.

Wimmer, G., Köhler, R., Grotjahn, R., & Altmann, G. (1994). Towards a theory of word length distribution. *Journal of Quantitative Linguistics, 1*, 98–106.

Zipf, G. K. (1949). Human behavior and the principle of least effort. An introduction to human ecology. Cambridge, MA: Addison-Wesley.

Received March 14, 2000
Revised February 23, 2001
Accepted February 23, 2001
Final manuscript July 20, 2001