Yugoslav Journal of Operations Research
18 (2008), Number 1, 123-138
DOI: 10.2298/YUJOR0801123R

DOCUMENT REPRESENTATIONS FOR CLASSIFICATION OF SHORT WEB-PAGE DESCRIPTIONS

Miloš RADOVANOVIĆ, Mirjana IVANOVIĆ
University of Novi Sad
Faculty of Science, Department of Mathematics and Informatics
Trg D. Obradovića 4, 21000 Novi Sad, Serbia
radacha@im.ns.ac.yu, mira@im.ns.ac.yu

Received: March 2006 / Accepted: March 2008

Abstract: Motivated by applying Text Categorization to classification of Web search results, this paper describes an extensive experimental study of the impact of bag-of-words document representations on the performance of five major classifiers – Naïve Bayes, SVM, Voted Perceptron, kNN and C4.5. The texts, representing short Web-page descriptions sorted into a large hierarchy of topics, are taken from the dmoz Open Directory Web-page ontology, and classifiers are trained to automatically determine the topics which may be relevant to a previously unseen Web-page. Different transformations of input data: stemming, normalization, logtf and idf, together with dimensionality reduction, are found to have a statistically significant improving or degrading effect on classification performance measured by classical metrics – accuracy, precision, recall, F_1 and F_2. The emphasis of the study is not on determining the best document representation which corresponds to each classifier, but rather on describing the effects of every individual transformation on classification, together with their mutual relationships.

Keywords: Text categorization, document representation, machine learning.

1. INTRODUCTION

Text Categorization (TC – also known as *Text Classification* or *Topic Spotting*) is the task of automatically sorting a set of documents into *categories* (or *classes*, or *topics*) from a predefined set [30]. Applications of TC include document indexing for Information Retrieval systems, text filtering (including protection from spam e-mail), word sense disambiguation, and categorization of Web pages. Sebastiani provides thorough overviews of the field in [29] and [30].

The initial motivation for the work presented in this paper lays in the development of a meta-search engine which uses TC to enhance the presentation of search results [25]. Basically, the system forwards the user's keyword-based query to a general-purpose Internet search engine, but instead of showing the results in the form of a list, it automatically categorizes those results looking only at the short snippets of Web pages, and displays a set of topics for the user to choose from. On one hand, this allows for a much more compact presentation of results on the screen, and on the other, it enables the user to refine his search more efficiently, by following topics (s)he is most interested in. The topics themselves are a based on topics from the dmoz Open Directory Web-page ontology, and are organized in a hierarchical manner. Eleven top-level topics were chosen, namely *Arts*, *Business*, *Computers*, *Games*, *Health*, *Home*, *Recreation*, *Science*, *Shopping*, *Society* and *Sports*; and an additional topic *Other* was added for unclassified documents. The second-level topics needed to be "massaged" a bit more, by merging or discarding them, and adding topic *Other*, in order to reach a form suitable for presentation of search results. Since English is a highly predominant language on the Web, at this time the system focuses on English language documents only.

The development of meta-search engines has received some research attention, although not many approaches reach the performance of commercial implementations, like Vivisimo, Dogpile or KartOO, about which there is very little technical information available [6]. For enhancing the presentation of results, clustering techniques are the focus of both academic and commercial interest.

A categorization approach, similar to ours, was described in [4]. Empirical tests performed in a closed environment, with the cooperation of Internet users of many profiles, showed that the topical style of presentation of search results was generally preferred over the list model. In [18] and [10], categorization experiments were performed on the Yahoo! and dmoz Open Directory ontologies, respectively. We plan to build on these experiences by answering the three questions posed in [17], from the context of our system: (1) what representation to use in documents, (2) how to deal with the high number of features, and (3) which learning algorithm to use. This paper focuses on question one, and its interaction with question three, at the same time trying (but not completely succeeding) to avoid question two.

Although the majority of works in TC employ the simple bag-of-words approach to document representation [9], not many comprehensive studies on the impact of its variations on classification performance were reported. Leopold and Kindermann [14] experimented with the Support Vector Machine (SVM) classifier with different kernels, term frequency transformations and lemmatization of German. They found that lemmatization usually degraded classification performance, and had the additional downside of great computational complexity, making SVMs capable of avoiding it altogether. Similar results were reported for neural networks on French [32]. Another study on the impact of document representation on one-class SVM [34] showed that, with a careful choice of representation, classification performance can reach 95% of the performance of SVM trained on both positive and negative examples. Kibriya et al. [12] compared the performance of SVM and a variant of the Naive Bayes classifier [26], emphasizing the importance of term frequency and inverse document frequency transformations (see Section 2.2) for Naive Bayes. A comprehensive experimental study of term weighing schemes for Text Categorization with SVMs was outlined in [13], proposing a new transformation based on relevance frequencies. Debole

and Sebastiani [5] investigated supervised learning of feature weights, and found that their replacement of the idf transformation can in some cases lead to significant improvement of classification performance. Another approach based on statistical confidence intervals is presented in [31], providing empirical evidence of superiority over the classical tfidf representation and the method by Debole and Sebastiani. A bidimensional representation of documents utilizing supervised term weighing was explored in [20]. The impact of word n-grams on Text Categorization was studied in [21] and [35]. Fuzzy approaches to document representation have also been explored, e.g. in [27] and [36].

This paper presents an extensive experimental study of classical bag-of-words document representations, and their impact on the performance on five classifiers commonly used for Text Categorization. An unorthodox evaluation methodology is used to measure and compare the effects of different transformations of input data on each classifier, and to determine their mutual relationships with regards to classification performance. Our primary aim was to use the results as a guideline for the implementation of the meta-search system. However, many of them ought to be applicable to the general case.

The next section outlines the experimental setup – how datasets were collected, which document representations were considered, and which classifiers. Section 3 presents the results – the representations that were found best, and the effects of and relationships between transformations: stemming, normalization, logtf and idf, together with a discussion on the observed robustness of some classifiers, as well as datasets, with regards to transforming document representations. The final section concludes, and gives guidelines for future work.

2. THE EXPERIMENTAL SETUP

The WEKA Machine Learning environment [33] was used to perform all experiments described in this paper. The classical measures – accuracy, precision, recall, F_1 and F_2 [30] – were chosen to evaluate the performance of classifiers on many variants of the bag-of-words representation of documents (i.e. short Web-page descriptions) taken from the dmoz Open Directory. The F_2 measure, which gives emphasis to recall over precision (F_1 gives them equal importance), is included for reasons similar to those in [16], where false positives are preferred to false negatives. What this means for categorization of search results is that it is preferred to overpopulate topics to a certain extent, over leaving results unclassified in the topic *Other*. Classification time and training time are also important factors, since the system needs to be running on a Web server classifying hundreds, possibly thousands of documents in real-time, and needs to be trained beforehand with as many examples as possible from the huge dmoz taxonomy.

2.1. Datasets

A total of eleven datasets, one for each chosen top-level category, were extracted from the dmoz collection dated July 2, 2005. The examples are either positive – taken from the corresponding category, or negative – distributed over all other categories, making this a binary classification problem. As initial tests showed that many

classifiers implemented in WEKA had difficulties dealing with imbalanced class distributions, (in this case, a much larger number of negative than positive examples) we kept the positive-negative ratio around 50–50. This is also justified by our preference of false positives to false negatives.

Since the dmoz hierarchy is very large (the content occupies over 2Gb of RDF data), we wrote a custom tool dmoz2arff which extracts examples from the dmoz RDF data in one pass, and offers basic facilities for selection of topics and examples, moving examples to higher levels, stopword elimination (with the standard stopword list from [28]), and stemming [23]. Table 1 summarizes the extracted datasets, showing the number of features (including the class feature) before and after stemming, the total number of examples, and the number of positive and negative ones.

Table 1: Extracted datasets

Dataset / Category	**Features**		**Examples**		
	Not stemmed	Stemmed	Total	Positive	Negative
Arts	3811	3142	626	300	326
Business	4248	3444	655	317	338
Computers	4293	3479	700	336	364
Games	4276	3551	764	382	382
Health	4460	3617	766	380	386
Home	4425	3583	765	374	391
Recreation	4389	3564	735	365	370
Science	4695	3792	754	379	375
Shopping	4470	3633	729	361	368
Society	4164	3402	675	344	331
Sports	4094	3391	753	380	373

When constructing the datasets and choosing the number of examples, care was taken to keep the number of features below 5000, for two reasons. The first reason was to give all classifiers an equal chance, because some of them are known not to be able to handle more than a couple of thousand features, and to do this without using some explicit form of feature selection (basically, to avoid question two from the Introduction). The second reason was the feasibility of running the experiments with the C4.5 classifier, due to its long training time. However, results from Section 3.4 (regarding the idf transform) prompted us to utilize the simple dimensionality reduction method based on term frequencies (TFDR), eliminating features representing the least frequent terms, a the same time keeping the number of features at around 1000. Therefore, two bundles of datasets were generated, one with and one without TFDR.

2.2. Document Representations

Let W be the *dictionary* – the set of all terms (words) that occur at least once in the training set of documents D. The bag-of-words representation of document d_j is a vector of weights $\mathbf{w}_j = (w_{1j},\ldots,w_{|W|j})$. For the simplest binary representation where $w_{ij} \in \{0, 1\}$, let the suffix 01 be added to the names of the datasets, so, for instance, Arts-01 denotes the binary representation of the Arts dataset. Similarly, the suffix tf will be used when w_{ij} represent the frequency of the ith term in the jth document. Normalization can be employed to scale down the term frequencies, accounting for

differences in the lengths of documents (norm). The logtf transform can also be applied to term frequencies, replacing the weights with $\log(1 + w_{ij})$. The *inverse document frequency* (idf) transform is defined as $\log(|D| / \text{docfreq}(D, i))$, where $\text{docfreq}(D, i)$ is the number of documents from D the ith term occurs in. It can be used by itself, or be multiplied with term frequency to yield the popular tfidf representation.

All these transformations, along with stemming (m), add up to 20 different variations of document representations, summarized in Table 2. This accounts for a total of $11 \cdot 20 \cdot 2 = 440$ different datasets for the experiments.

Table 2: Document representations

Not stemmed		**Stemmed**	
Not normalized	Normalized	Not normalized	Normalized
01		m-01	
idf		m-idf	
tf	norm-tf	m-tf	m-norm-tf
logtf	norm-logtf	m-logtf	m-norm-logtf
tfidf	norm-tfidf	m-tfidf	m-norm-tfidf
logtfidf	norm-logtfidf	m-logtfidf	m-norm-logtfidf

2.3. Classifiers

Five classifiers implemented in WEKA are used in this study: ComplementNaive-Bayes (CNB), SMO, VotedPerceptron (VP), IBk, and J48.

CNB [26], [12] is an improved version of the NaiveBayesMultinomial classifier [15], optimized for application on text. Initial tests showed that CNB consistently outperforms its predecessor on our datasets, although not at a statistically significant level ($p = 0.05$). SMO [22], [11] is an implementation of Platt's Sequential Minimal Optimization algorithm for training SVMs, offering an efficient solution to the quadratic programming problem posed by a training set. VP was first introduced by Freund and Schapire [8], and shown to be a simple, yet effective classifier for high-dimensional data. IBk is a variation of the classical k-Nearest Neighbor algorithm [2], and J48 is based on revision 8 of the C4.5 decision tree learner [24].

All classifiers were run using their default parameters, with the exception of SMO, where the option not to normalize training data was chosen. IBk performed rather erratically during initial testing, with performance varying greatly with different datasets, choices of k and distance weighing, so in the end we kept $k = 1$ as it proved most stable. We were unable to reproduce the state-of-the-art performance achieved elsewhere [30], but report the results anyway, as some of them may still prove valuable. Not until the late phases of experimentation did we realize that the Euclidean distance measure tends to deform with high numbers of features [3], [1]. Therefore, the bad performance of IBk may be treated as an experimental verification of this fact in the context of classification.

Although SVMs are generally considered the best classifier for text (especially when accuracy is concerned), much may depend on the properties of the dataset (as was effectively demonstrated by Gabrilovich and Markovitch [9]), the evaluation measure that is considered important (we are placing an emphasis on the less commonly used F_2), and the final application of the classifier. For these reasons, all mentioned classifiers were included and equally treated in the experiments.

3. RESULTS

A separate WEKA experiment was run for every classifier with the 20 document representation datasets for each of the 11 major categories. Results of evaluation measures were averaged over five runs of 4-fold cross-validation, following [7] and [9]. Measures were compared between *datasets* using the corrected resampled t-test [19] implemented in WEKA, at $p = 0.05$, and the number of statistically significant wins and losses of each representation added up for every classifier over the 11 categories.

For the sake of future experiments and the implementation of the meta-search system, best representations for each classifier were chosen, based on wins–losses values summed-up over all datasets. The declared best representations were not winners for all 11 categories, but showed best performance overall. For VP and J48 the choice was simple: m-logtf appeared among the winners both before and after TFDR. Since TFDR broke the performance of IBk, m-norm-logtf was declared best in that department, for it was the winner before TFDR. As for CNB, m-norm-tf was best before TFDR, while idf was best after, but by a much smaller margin, therefore m-norm-tf was chosen. For SMO, the situation was opposite with regards to the idf transform: m-norm-tfidf was the winner before TFDR, and m-norm-logtf after, by a bigger margin, so m-norm-logtf was considered best. Section 3.4 explains in more detail the reasons we decided to stay away from the idf transform.

Table 3 shows the wins–losses values of the declared best document representations for each classifier, before and after TFDR. Binary representations were practically never among the best, for all datasets, confirming the widespread agreement on the need for tf-based document representations.

Table 3: Wins–losses values of best document representations for each classifier, on datasets without (left columns) and with dimensionality reduction

	CNB m-norm-tf		**SMO** m-norm-logtf		**VP** m-logtf		**IBk** m-norm-logtf	**J48** m-logtf	
Accuracy	41	1	1	37	15	2	119	40	40
Precision	45	1	20	6	29	12	11	-6	-5
Recall	4	1	-4	68	0	0	67	56	57
F_1	28	1	0	47	7	0	120	59	52
F_2	9	0	-3	71	0	0	78	63	57
TOTAL	127	4	14	229	51	14	395	212	201

For illustrating the impact of document representations on classification, Tables 4 and 5 summarize the performance of classifiers on the best representations, and the improvements over the worst ones, on the Home dataset, before and after TFDR, respectively. Note that the emphasis of this paper is not on fine-tuning the performance of classifiers, even using document representations, as much as it is on determining the impacts and relationships between different transforms (stemming, normalization, logtf and idf) and dimensionality reduction, with regards to each classifier. This is the prevailing subject of the remainder of this section.

Table 4: Performance of classification (in %) using the best document representations on the Home dataset *without* dimensionality reduction, together with improvements over the worst representations (statistically significant ones are in **boldface**)

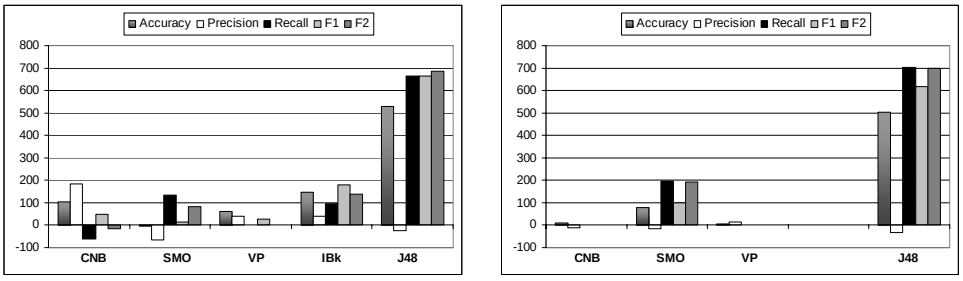
	CNB	SMO	VP	IBk	J48
Accuracy	82.56 (**5.26**)	83.19 (1.67)	78.38 (**5.12**)	74.93 (**21.96**)	71.77 (**3.64**)
Precision	81.24 (**8.66**)	85.67 (**3.86**)	80.45 (**7.85**)	71.32 (**14.32**)	90.24 (1.60)
Recall	83.91 (1.81)	78.93 (3.80)	74.06 (0.96)	81.66 (**45.20**)	47.59 (**10.59**)
F_1	82.48 (**3.64**)	82.07 (2.17)	77.02 (**4.23**)	76.07 (**33.90**)	62.12 (**9.09**)
F_2	83.31 (2.19)	80.14 (3.30)	75.20 (2.16)	79.31 (**39.72**)	52.48 (**10.41**)

Table 5: Performance of classification (in %) using the best document representations on the Home dataset *with* dimensionality reduction, together with improvements over the worst representations (statistically significant ones are in **boldface**)

	CNB	SMO	VP	J48
Accuracy	85.86 (1.12)	82.80 (**4.60**)	79.29 (4.18)	71.49 (**3.15**)
Precision	86.48 (1.78)	83.77 (4.79)	81.10 (**6.40**)	90.21 (1.48)
Recall	84.39 (1.07)	80.64 (**5.72**)	75.45 (2.24)	47.43 (**9.84**)
F_1	85.35 (1.04)	82.07 (**4.88**)	78.08 (3.57)	61.98 (**8.40**)
F_2	84.75 (0.67)	81.19 (**4.96**)	76.46 (2.24)	52.33 (**9.66**)

3.1. Effects of Stemming

The effects of stemming on classification performance were measured by adding-up the wins–losses values for stemmed and non-stemmed datasets, and examining their difference, depicted graphically in Figure 1. It can be seen that stemming improves almost all evaluation measures, both before and after TFDR. After TFDR, the effect of stemming is generally not as strong, which is understandable because its impact as a dimensionality reduction method is reduced. CNB is then practically unaffected, only SMO exhibits an increased tendency towards being improved. Overall, J48 is especially sensitive to stemming.



(a) (b)

Figure 1: The effects of *stemming* before (a) and after dimensionality reduction (b)

To investigate the relationships between stemming and other transformations, a chart was generated for each transformation, measuring the effect of stemming on representations with and without the transformation applied. Figure 2 shows the effect of stemming on non-normalized and normalized data, without TFDR. It can be noted that non-normalized representations are affected by stemming more strongly (for the better). The same holds with TFDR applied (charts not shown).

The logtf transform exhibited no influence on the impact of stemming, regardless of TFDR. The corresponding charts are only scaled-down versions of Figure 1.

Applying the idf transform to tf, without TFDR, made no difference in stemming performance, except for greater improvements on IBk. After TFDR the situation was a little different: application of idf led to a drop in the effect on accuracy and precision of CNB, and to a rise of the accuracy of SMO (Figure 3).

The above analysis confirms the common view of stemming as a method for improving classification performance for English. However, this may not be the case for other languages, for instance German [14] and French [32].

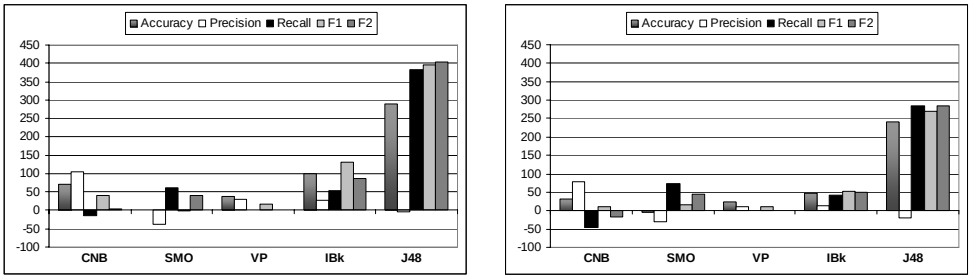


Figure 2: The effects of *stemming* on non-normalized (a) and normalized data (b), *without* dimensionality reduction

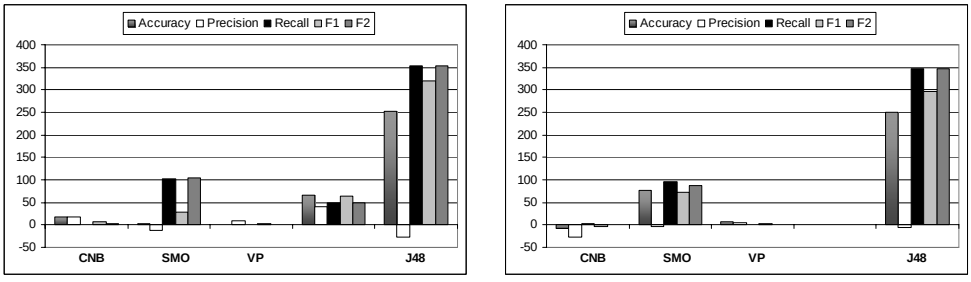
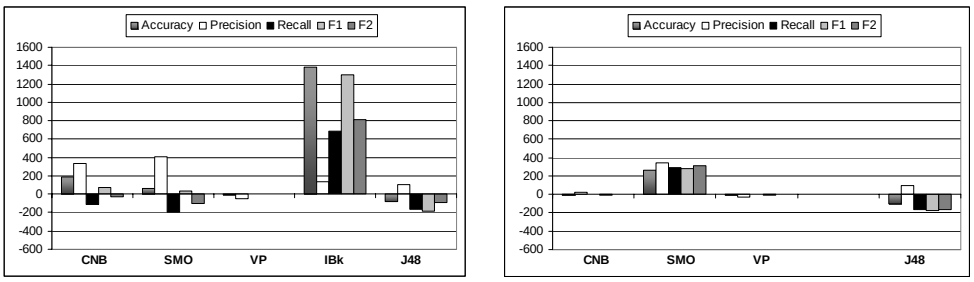
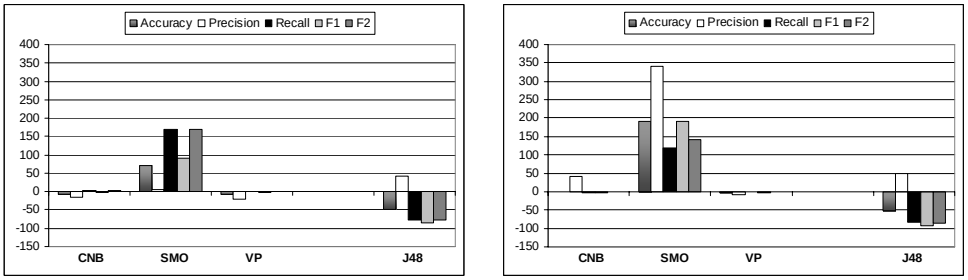


Figure 3: The effects of *stemming* on data without (a) and with the idf transform applied to tf (b), *with* dimensionality reduction



(a) (b)

Figure 4: The effects of *normalization* before (a) and after dimensionality reduction (b)



(a) (b)

Figure 5: The effects of *normalization* on data without (a) and with the idf transform applied to tf (b), *with* dimensionality reduction

3.2. Effects of Normalization

The chart in Figure 4 shows that normalization tends to improve classification performance in a majority of cases. Without TFDR, VP was virtually unaffected, CNB and SMO were improved on all counts but recall (and consequently F_2), while the biggest improvement was on IBk, which was anticipated since normalization assisted the comparison of document vectors. J48 was the only classifier whose performance worsened with normalization. Apparently, J48 found it tougher to find appropriate numeric intervals within the normalized weights for branching the decision tree. After TFDR, CNB joined VP in its insensitivity, while SMO witnessed a big boost in performance when data was normalized.

No significant interaction between normalization and stemming was revealed, only that stemmed J48 was more strongly worsened by normalization. It seems that normalization misleads J48 from the discriminative features introduced by stemming.

Normalization and the logtf transform exhibited no notable relationship, while with idf transformed data, normalization had stronger influence on classification. After TFDR, this tendency was especially noticeable with the improvement of the precision of SMO (Figure 5). This can be explained by the fact that idf severely worsens the performance of SMO after TFDR (see Section 3.4), and normalization compensated

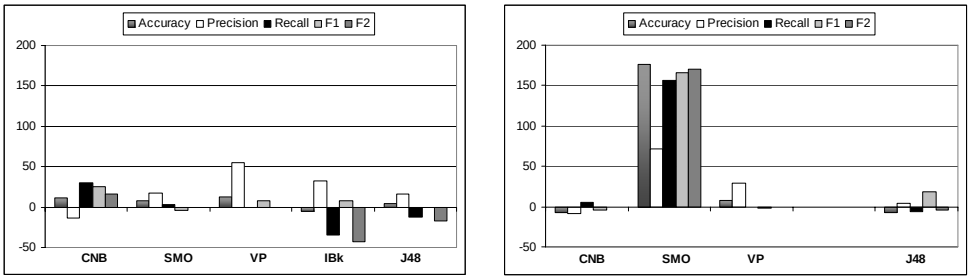
somewhat for this. The compensating effect of one transform on the performance degrading influences of another was found to be quite common in the experiments.

It is important to emphasize that the datasets used in these experiments consist of short documents, and therefore normalization does not have as strong an impact as it would have if the differences in document lengths were more drastic. Therefore, the conclusions above may not hold for the general case, for which a further, more comprehensive study is needed.

3.3. Effects of the logtf Transform

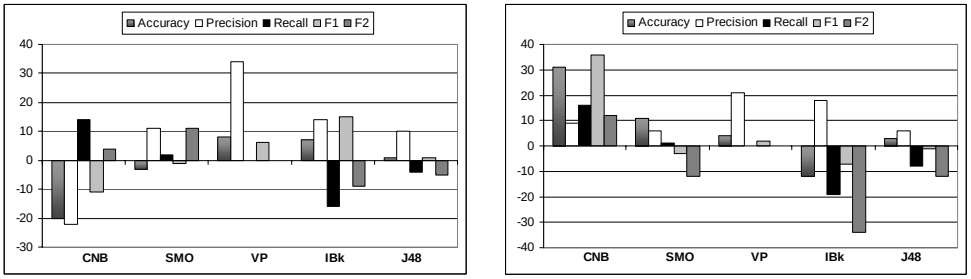
As can be seen in Figure 6, the logtf transform causes mostly mild improvements of classification. After TFDR, improvements are greater on SMO, while the impact on other classifiers is weaker.

Figure 7 shows that logtf has a much better impact on CNB when idf is also applied, without TFDR. This is similar to the compensating effect of normalization on idf with the SMO classifier from the previous section. Relations change quite dramatically when TFDR is applied (Figure 8), but the effect on SMO is again analogous to the previous section. The improvements on CNB are especially significant, meaning that logtf and idf work together on improving classification performance.



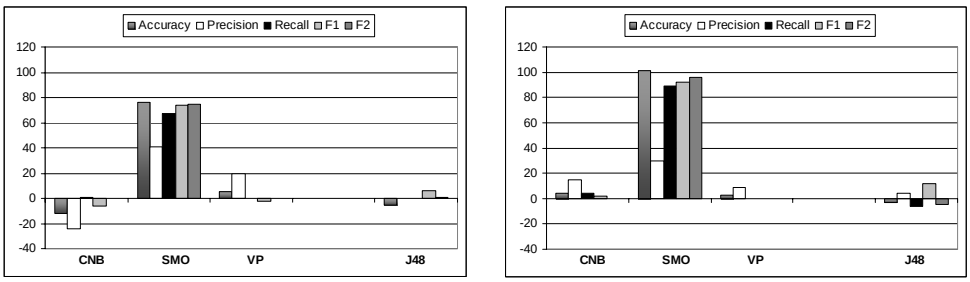
(a) (b)

Figure 6: The effects of the logtf transform before (a) and after dimensionality reduction (b)



(a) (b)

Figure 7: The effects of the logtf transform on data without (a) and with the idf transform applied to tf (b), *without* dimensionality reduction



(a) (b)

Figure 8: The effects of the logtf transform on data without (a) and with the idf transform applied to tf (b), *with* dimensionality reduction

Before TFDR, the interaction of logtf and normalization varied across classifiers: logtf improved CNB and IBk on normalized data, while the others were improved without normalization. After TFDR, the chart looks very much like the left-right reverse of Figure 8 – with logtf having a weaker positive effect on normalized data, especially for CNB and SMO, which were already improved by normalization.

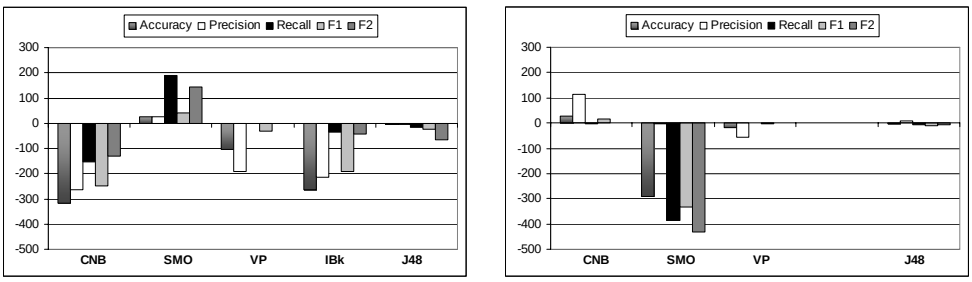
Understandably, the logtf transform has a stronger positive impact on non-stemmed data, regardless of dimensionality reduction, with the exception of VP which exhibited no variations. This is in line with the witnessed improvements that stemming introduces on its own, and the already noted compensation phenomenon.

3.4. Effects of the idf Transform

Applying the idf transform on data turned out to have the richest repertoire of effects, from significant improvement, to severe degradation of classification performance. Figure 9(a) illustrates how idf drags down the performance of all classifiers except SMO, without TFDR. It was for this reason we introduced TFDR in the first place, being aware that our data had many features which were present in only a few documents. We expected idf to improve, or at least degrade to a lesser extent the performance of classification. That did happen, as Figure 9(b) shows, for all classifiers *except* SMO, whose performance drastically degraded! The simple idf document representation rose from being one of the worst, to one of the best representations of documents, for all classifiers but SMO.

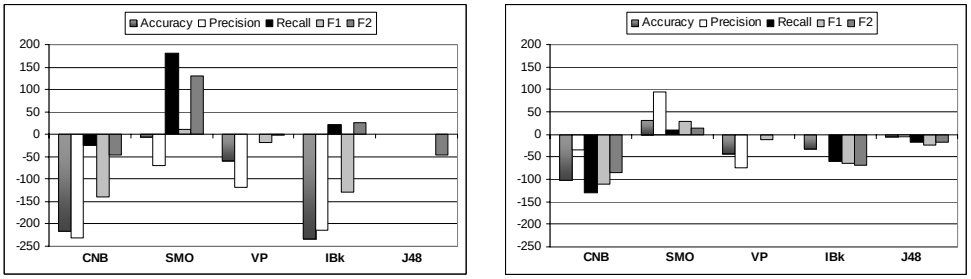
No significant correlation was detected by applying idf on stemmed and non-stemmed data. However, plenty of different effects were noticeable with regards to normalization. Without TFDR (Figure 10), a stronger worsening effect on non-normalized data was exhibited with CNB, VP and IBk, while for SMO normalization dampened idf's improvement of recall, but overturned the degradation of accuracy and precision. With TFDR, the picture is quite different (Figure 11): normalization improved the effects on CNB and VP, with SMO witnessing a partial improvement on precision, while J48 remained virtually intact.

The impact of idf on (non-)logtfed datasets showed no big differences – trends remained the same as in Figure 9, perhaps a little stronger with logtf applied.



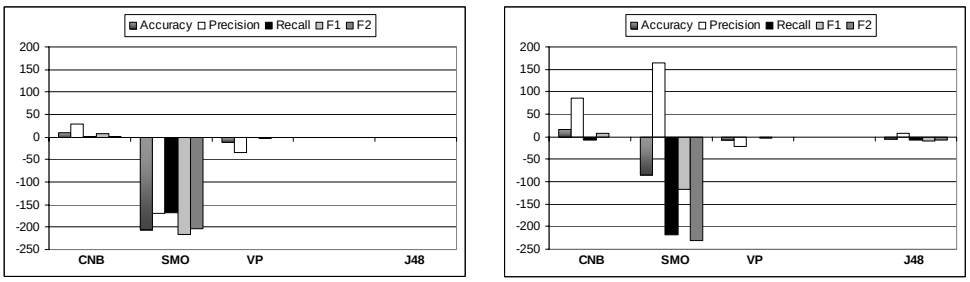
(a) (b)

Figure 9: The effects of idf applied to tf before (a) and after dimensionality reduction (b)



(a) (b)

Figure 10: The effects of idf applied to tf on non-normalized (a) and normalized data (b), *without* dimensionality reduction



(a) (b)

Figure 11: The effects of idf applied to tf on non-normalized (a) and normalized data (b), *with* dimensionality reduction

The above analysis shows that one needs to be careful when including the idf transform in the representation of documents. Removing infrequent features is an important prerequisite to its application, since idf assigns them often unrealistic importance, but that may not be enough, as was proved by the severe degradation of SMO's performance.

3.5. Robustness

An interesting phenomenon observed in the experiments is the apparent insensitivity of some classifiers to document representations, which we will refer to as *robustness*. The Total row of Table 3 provides a hint, with the winning representation for VP showing the lowest number of wins–losses overall, especially with regards to recall and F_2, as did CNB with, and SMO without TFDR (although the representations were not exactly optimal for those cases). A better indicator is the summed-up number of wins (the number of losses is the same) for each classifier, over all datasets, document representations and evaluation measures, shown in the Total *row* of Table 6, which confirms the above observation. When examining the partial sums for each evaluation measure (the Total *column*), precision shows the lowest variation with regards to document representation. However, every classifier exhibits its lowest sensitivity at different measures: CNB and VP at recall and F_2, SMO at accuracy and F_1, IBk and J48 at precision. This correlates with the lowest improvement rates exemplified on the Home dataset in Tables 4 and 5.

Table 6: Total number of wins (=losses) of all document representations, for each classifier and evaluation measure, on datasets without (left columns) and with TFDR. The right Total column includes the left IBk column in the sum, to enable comparison of the number of wins without and with TFDR

	CNB		**SMO**		**VP**		**IBk**	**J48**		**TOTAL**	
Accuracy	265	23	46	277	67	9	1155	336	321	1869	1785
Precision	355	79	262	465	124	34	494	64	56	1299	1128
Recall	121	5	234	626	0	0	929	468	486	1752	2046
F_1	178	14	35	297	21	2	1140	477	437	1851	1890
F_2	83	2	152	535	2	0	837	568	489	1642	1863
TOTAL	1002	123	729	2200	214	45	4555	1913	1789	8413	8712

Robustness regarding document representations was also observed on *datasets*. For example, the total number of wins for all document representations, over all classifiers and measures, for the Computers dataset is 167, while for Games the number is 1101 (before TFDR; after TFDR the numbers are almost the same). This may be due to a presence of more discriminating features in the Computers dataset, or a combination of that and other factors, and calls for a further investigation towards developing a simple, theoretical criteria for determining the robustness and, going further, a best document representation for a particular dataset.

4. CONCLUSIONS AND FURTHER WORK

By using transformations in bag-of-words document representations there is, essentially, no new information added to a dataset which is not already there (except for the transition from `01` to `tf` representations). The general-purpose classification algorithms, however, are unable to derive such information on their own, which is understandable because they are not aware of the exact nature of the data being processed. It is therefore expected of transforms to have a significant effect on

classification performance, which was experimentally demonstrated at the beginning of Section 3. The fact that this issue is ignored by many studies and applications of text classification is somewhat striking.

Besides helping to determine a best representation for each classifier, the experiments revealed the individual effects of transforms on different evaluation measures of classification performance, and some of their relationships. Stemming generally improved classification, partly because of its role as a dimensionality reduction method, and had an exceptionally strong improving impact on J48, which can be explained by its merging of words into more discriminative features, suiting the algorithm's feature selection method when constructing the decision tree. Normalization enhanced CNB, SMO and especially IBk, leaving VP practically unaffected and worsening the performance of J48. Although dmoz data contains only short documents, normalization did have a significant impact, but no definite conclusions can be drawn for the general case. The logtf transform had mostly a mild improving impact, except on SMO after TFDR, which exhibited stronger improvement. The situation with idf was trickier, with the effects depending strongly on dimensionality reduction for CNB and SMO, but in opposite directions: CNB was degraded by idf before, and improved after TFDR; for SMO it was vice versa.

The most common form of relationship between transforms that was noticed were the compensating effects of one transform on the performance degrading impact of another (e.g. normalization with idf on SMO, logtf with idf on CNB and SMO). The logtf and idf transforms seemed to work together on improving CNB after TFDR. The impact of idf on normalization was most complex, with great variation in the effects on different evaluation measures. Note that the method for determining relations between transforms appeared not to be commutative – for instance, the effects of normalization on idf transformed data and of idf on normalized data are not the same. Some relationships can be missed when looking only one way.

The comments above refer to the general case of performance measuring. Some transforms (especially idf) may improve one measure, at the same time degrading another. Often, the preferred evaluation measure, chosen with the application of the classifier in mind, will need to be monitored when applying the results presented in this paper. In our case, for applying classification to search results, the F_2 measure was considered most important.

Robustness regarding document representations, which applies both to classifiers and datasets, is an interesting area for further theoretical investigations – exploring possibilities for developing simple tests for determining the robustness, interactions with particular transformations and, ultimately, a best document representation for a particular classifier and/or dataset, without extensive experimentation. Such tests may be useful in situations where detailed fine-tuning of document representations is not feasible.

The main difficulty with comprehensive experiments like the ones described in this paper is sheer size. Roughly speaking, factors such as datasets, document representations, dimensionality reduction methods, reduction rates, classifiers, and evaluation measures, all have their counts multiplied, leading to a combinatorial explosion which is hard to handle. We tackled this problem by excluding detailed experimentation with dimensionality reduction, and using dmoz as the only source of data. Therefore, no definite truths, but only pointers can be derived from the described

experience. A more comprehensive experiment, featuring other common corpora (Reuters, OHSUMED, 20Newsgorups etc.), longer documents, and dimensionality reduction methods is called for to shed more light on the impacts and relationships of all the above mentioned factors.

Acknowledgments: This work was supported by project "Abstract Methods and Applications in Computer Science" (no. 144017A), of the Serbian Ministry of Science and Environmental Protection.

REFERENCES

[1] Aggarwal, C.C., Hinneburg, A., and Keim, D.A., "On the surprising behavior of distance metrics in high dimensional spaces", *Proc. ICDT'01, 8th International Conference on Database Theory*, LNCS 1973, London, UK, Springer-Verlag, 2001, 420–434.

[2] Aha, D., Kibler, D., and Albert, M.K., "Instance-based learning algorithms", *Machine Learning*, 6(1) (1991) 37–66.

[3] Beyer, K.S., Goldstein, J., Ramakrishnan, R., and Shaft, U., "When is "nearest neighbor" meaningful?", *Proc. ICDT'99, 7th International Conference on Database Theory*, LNCS 1540, Jerusalem, Israel, Springer-Verlag, 1999, 217–235.

[4] Chen, H., and Dumais, S.T., "Bringing order to the Web: Automatically categorizing search results", *Proc. CHI'00, Human Factors in Computing Systems*, 2000, 145–152.

[5] Debole, F., and Sebastiani, F., "Supervised term weighting for automated text categorization", in: S. Sirmakessis (ed.), *Text Mining and its Applications*, Studies in Fuzziness and Soft Computing 138, Physica-Verlag, Heidelberg, Germany, 2004, 81–98.

[6] Ferragina, P., and Gulli, A., "A personalized search engine based on Web-snippet hierarchical clustering", *Proc. WWW'05, 14th International World Wide Web Conference*, Chiba, Japan, 2005, 801–810.

[7] Forman, G., "An extensive empirical study of feature selection metrics for text classification", *Journal of Machine Learning Research*, 3 (2003) 1289–1305.

[8] Freund, Y, and Schapire, R.E., "Large margin classification using the perceptron algorithm", *Machine Learning*, 37(3) (1999) 277–296.

[9] Gabrilovich, E., and Markovitch, S., "Text categorization with many redundant features: Using aggressive feature selection to make SVMs competitive with C4.5", *Proc. ICML'04, 21st International Conference on Machine Learning*, Banff, Canada, 2004, 41–48.

[10] Grobelnik, M., and Mladenić, D., "Simple classification into large topic ontology of Web documents", *Proc. ITI'05, 27th International Conference on Information Technology Interfaces*, Cavtat, Croatia, 2005, 188–193.

[11] Keerthi, S.S., Shevade, S.K., Bhattacharyya, C., and Murthy, K.R.K., "Improvements to Platt's SMO algorithm for SVM classifier design", *Neural Computation*, 13(3) (2001) 637–649.

[12] Kibriya, A.M., Frank, E., Pfahringer, B., and Holmes, G., "Multinomial naive Bayes for text categorization revisited", *Proc. AI'04, 17th Australian Joint Conference on Artificial Intelligence*, LNAI 3339, Cairns, Australia, Springer-Verlag, 2004, 488–499.

[13] Lan, M., Tan, C.-L., Low, H.-B., and Sung, S.-Y., "A comprehensive comparative study on term weighting schemes for text categorization with support vector machines", *Proc. WWW'05, 14th International World Wide Web Conference*, Chiba, Japan, 2005, 1032–1033.

[14] Leopold, E., and Kindermann, J., "Text categorization with support vector machines. How to represent texts in input space?", *Machine Learning*, 46(1–3) (2002) 423–444.

[15] McCallum, A., and Nigam, K., "A comparison of event models for naive Bayes text classification", *Proc. AAAI'98 Workshop on Learning for Text Categorization*, 1998, 41–48.

[16] Mladenić, D., "Machine Learning on non-homogenous, distributed text data", PhD thesis, University of Ljubljana, Slovenia, 1998.
[17] Mladenić, D., "Text-learning and related intelligent agents", *IEEE Intelligent Systems*, *Special Issue on Applications of Intelligent Information Retrieval*, 14(4) (1999) 44–54.
[18] Mladenić, D., and Grobelnik, M., "Mapping documents onto Web page ontology", in: B. Berendt et al. (eds.), *Web Mining: From Web to Semantic Web*, LNAI 3209, Springer-Verlag, 2004, 77–96.
[19] Nadeau, C., and Bengio, Y., "Inference for the generalization error", *Machine Learning*, 52(3) (2003) 239–281.
[20] Nunzio, D.M., "A bidimensional view of documents for text categorization", *Proc. ECIR'04, 26th European Conference on IR Research*, LNCS 2997, Sunderland, UK, Springer-Verlag, 2004, 112–126.
[21] Peng, F., and Schuurmans, D., "Combining naive Bayes and n-gram language models for text classification", *Proc. ECIR'03, 25th European Conference on IR Research*, LNCS 2663, Pisa, Italy, Springer-Verlag, 2003, 335–350.
[22] Platt, J., "Fast training of support vector machines using sequential minimal optimization.", in: B. Schoelkopf, C. Burges, A. Smola (eds.), A*dvances in Kernel Methods—Support Vector Learning*, MIT Press, 1998, 185–208.
[23] Porter, M.F., "An algorithm for suffix stripping", *Program*, 14(3) (1980) 130–137.
[24] Quinlan, R., *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, San Mateo, CA, 1993.
[25] Radovanović, M., and Ivanović, M., "CatS: A classification-powered meta-search engine", in: M. Last, P. S. Szczepaniak, Z. Volkovich, A. Kandel (eds.)*, Advances in Web Intelligence and Data Mining*, Studies in Computational Intelligence 23, Springer-Verlag, 2006, 191–200.
[26] Rennie, J.D.M., Shih, L., Teevan, J., and Karger, D.R., "Tackling the poor assumptions of naive Bayes text classifiers", *Proc. ICML'03, 20th International Conference on Machine Learning*, 2003.
[27] Ribeiro, A., Fresno, V., Garcia-Alegre, M.C., and Guinea, D., "Web page classification: A soft computing approach", *Proc. AWIC'03, 1st Atlantic Web Intelligence Conference*, LNAI 2663, Madrid, Spain, Springer-Verlag, 2003, 103–112.
[28] Salton, G. (ed.), *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice-Hall, 1971.
[29] Sebastiani, F., "Machine learning in automated text categorization", *ACM Computing Surveys*, 34(1) (2002) 1–47.
[30] Sebastiani, F., "Text categorization", in: Alessandro Zanasi (ed.), *Text Mining and its Applications*, WIT Press, Southampton, UK, 2005, 109–129.
[31] Soucy, P., and Mineau, G.W., "Beyond TFIDF weighting for text categorization in the vector space model", *Proc. IJCAI'05, 19th International Joint Conference on Artificial Intelligence*, Edinburgh, UK, 2005, 1130–1135.
[32] Stricker, M., Vichot, F., Dreyfus, D., and Wolinski, F., Vers la conception automatique de filtres d'informations efficaces. *Reconnaissance des Formes et Intelligence Artificielle RFIA2000*, 2000, 129–137.
[33] Witten, I.H., and Frank, E., *Data Mining: Practical Machine Learning Tools and Techniques*. Second Edition, Morgan Kaufmann Publishers, San Francisco, 2005.
[34] Wu, X., Srihari, R., and Zheng, Z., "Document representation for one-class SVM", *Proc. ECML'04, 15th European Conference on Machine Learning*, LNAI 3201, Pisa, Italy, Springer-Verlag, 2004, 489–500.
[35] Yetisgen-Yildiz, M., and Pratt, W., "The effect of feature representation on MEDLINE document classification", *Proc. AMIA'05, American Medical Informatics Association Fall Symposium*, Washington D.C., 2005.
[36] Zadrozny, S., Kacprzyk, J., "Computing with words for text processing: An approach to the text categorization", *Information Sciences*, 176 (2006) 415–437.