

Proposing a New Term Weighting Scheme for Text Categorization

Man LAN*

Institute for Infocomm Research
21 Heng Mui Keng Terrace
Singapore 119613
lanman@i2r.a-star.edu.sg

Chew-Lim TAN

School of Computing
National University of Singapore
3 Science Drive 2, Singapore 117543
tancl@comp.nus.edu.sg

Hwee-Boon LOW

Institute for Infocomm Research
21 Heng Mui Keng Terrace
Singapore 119613
hweeboon@i2r.a-star.edu.sg

Abstract

In text categorization, term weighting methods assign appropriate weights to the terms to improve the classification performance. In this study, we propose an effective term weighting scheme, i.e. *tf.rf*, and investigate several widely-used unsupervised and supervised term weighting methods on two popular data collections in combination with SVM and *k*NN algorithms. From our controlled experimental results, not all supervised term weighting methods have a consistent superiority over unsupervised term weighting methods. Specifically, the three supervised methods based on the information theory, i.e. *tf.χ²*, *tf.ig* and *tf.or*, perform rather poorly in all experiments. On the other hand, our proposed *tf.rf* achieves the best performance consistently and outperforms other methods substantially and significantly. The popularly-used *tf.idf* method has not shown a uniformly good performance with respect to different data corpora.

Introduction

Text categorization is the task of automatically assigning unlabelled documents into predefined categories. As a vital step in text categorization, text representation transforms the content of textual documents into compact format so that the documents can be recognized and classified by a computer or a classifier. In the Vector Space model, a document is represented as a vector in the term spaces, $d_j = (w_{1j}, \dots, w_{kj})$, where k is the size of the set of terms (*features*). The value of w_{kj} between $(0, 1)$ represents how much the term t_k contributes to the semantics of document d_j .

In this study, we classify the term weighting methods into two categories according to whether the method makes use of this known information on the membership of training documents, namely, *supervised term weighting method* and *unsupervised term weighting method*. For instance, the traditional methods borrowed from information retrieval field, such as *binary*, *tf*, *tf.idf* (Salton & Buckley 1988) and its variants, belong to the unsupervised term weighting methods. Other methods in our study belong to the supervised

ones since these weights must be calculated based on the information on the membership of training documents in categories. Recently, researchers make use of this prior information to assign weights in two ways. One approach is to weight terms by using feature selection metrics, i.e. χ^2 , *information gain*, *gain ratio*, *odds ratio* and so on, in (Deng *et al.* 2004) and (Debole & Sebastiani 2003). Another approach is to combine the term weighting methods with supervised linear text classifier, for example, in (Mladenic *et al.* 2004).

However, not much work has been done on the comparison between supervised and unsupervised term weighting methods, such as *tf.idf*, *tf* alone, or even the simplest one – *binary*. Though there is one comparison with *tf.idf* (Debole & Sebastiani 2003), the supervised term weighting methods have been shown to give mixed results. On the other hand, another work (Deng *et al.* 2004) has drawn conclusions contrary to our previous findings (Lan *et al.* 2005). Therefore, a fundamental question arises here, i.e. “Are supervised term weighting approaches based on known information able to lead to better performance than unsupervised ones for text categorization?” As a result of this study, we have proposed a new supervised term weighting scheme.

The rest of this paper is structured as follows. In Section 2 we survey the existing term weighting methods and propose a new term weighting method. In Section 3, we describe experimental settings and report results and discussion. We conclude in Section 4.

Term Weighting Methods: Survey and Analysis

Generally, for multi-label classification problem, the benchmark on each corpus is simplified into multiple independent binary classification problems. That is, in each experiment, a chosen category c_i is tagged as the positive category and the other categories in the same corpus are combined together as the negative category \bar{c}_i . Table 1 records the number of documents which contain term t_k and do not contain term t_k under category c_i and \bar{c}_i . Thus the famous *idf* factor is computed as $\log(\frac{N}{a+c})$, where $N = a + b + c + d$. Usually, $d \gg a, b, c$.

*Also a PhD candidate in School of Computing, National University of Singapore.
Copyright © 2006, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

Table 1: The Contingency Table for Category c_i and Term t_k

	t_k	\bar{t}_k
Positive Category: c_i	a	b
Negative Category: \bar{c}_i	c	d

Traditional Term Weighting Methods

The traditional term weighting methods for text categorization are usually borrowed from information retrieval field and belong to the unsupervised term weighting methods. For example, the simplest *binary* representation is adopted by text classifiers which only involve binary operation rather than floating number operation, such as Decision Tree. The term occurrences alone could be used as term weighting methods without other factors, such as raw term frequency (tf), $\log(1 + tf)$ and ITF (Leo 2002). The previous study (Lan *et al.* 2005) showed that there is no significant difference among them in a SVM-based text classifier because they actually are derived from the same base.

The most popular term weighting approach is $tf.idf$ (Salton & Buckley 1988), which has been widely used in information retrieval and has consequently been adopted by researchers in text categorization. There are several variants of $tf.idf$, such as $\log(tf).idf$, $tf.idf-prob$ (also *term relevance*, see (Wu & Salton 1981)). The previous study (Lan *et al.* 2005) also showed there is no significant difference among them. Consequently, we adopt the most popular $tf.idf$ method as a baseline in this study.

Supervised Term Weighting Methods

Usually, “text categorization” is a form of *supervised* learning as it makes use of prior information on the membership of training documents in predefined categories. This known label of training data set is effective and has been widely used in the step of feature selection and the construction of text classifier to improve the performance (Yang & Pedersen 1997). Generally, the supervised term weighting methods adopt this known information in the following two ways.

Combined with information-theory functions or statistic metrics One approach is to weight terms by adopting feature selection metrics, such as χ^2 , *information gain*, *gain ratio*, *odds ratio* and so on. The main purpose of feature selection is to select the most relevant and discriminating features for the classification task. The terms with higher feature selection scores are deemed to have contributed more to the text categorization than those with lower scores.

In (Deng *et al.* 2004), Deng *et al* replaced the idf factor with χ^2 factor to weight terms and asserted that $tf.\chi^2$ is most effective than $tf.idf$ in their experiments with a SVM-based text categorization. Similarly, Debole (Debole & Sebastiani 2003) assigned weight to terms by replacing the idf factor with the metrics that have been used for feature selection process, namely, *information gain*, χ^2 and *gain ratio*. However, these supervised term weighting methods have not been shown to have a consistent superiority over the standard $tf.idf$ -based term weighting.

Beside text categorization task, Mori (Mori 2002) adopted this idea in document summarization and their result showed that this *gain ratio (gr)* *gr*-based term weighting method is very effective in summarization.

Interaction with Linear Text Classifier Another approach is to combine the term weighting approaches with supervised linear learning algorithms. Since the text classifier selects the positive test documents from negative test documents by assigning different scores to the test samples, these scores are believed to be effective in assigning more appropriate weights to the terms. Mladenic *et al* (Mladenic *et al.* 2004) compared three term weighting methods, i.e. *Odds Ratio*, *Information Gain* and weights from linear models (linear SVM and Perceptron). Their results showed that weighting methods from linear SVMs yields better classification performance than other methods when combined with three algorithms, namely, Naive Bayes, Perception and SVM.

Our Newly Proposed rf Scheme The traditional idf factor and its variants have been introduced to improve the discriminating power of terms in the traditional information retrieval field. However, in text categorization, it may not be the case.

Let us consider the examples in Table 2. Given one chosen positive category c_i , the three terms, i.e. t_1 , t_2 and t_3 , share the same idf but have different ratio of a and c . We assume they have the same term frequency (tf) and $N = 1000$.

Table 2: Examples of three terms which share the same idf but have different ratio of a and c

Term	Sum(a,c)	a : c	idf
t_1	100	10 : 1	$\log(N/100) = 3.322$
t_2	100	1 : 1	$\log(N/100) = 3.322$
t_3	100	1 : 10	$\log(N/100) = 3.322$

Clearly, the traditional idf factor gives equal weights to the three terms. But we can easily find that these three terms show different discriminating power to text categorization. t_1 and t_3 contribute more power to discriminate the documents in the positive and negative categories respectively but t_2 gives little contribution to this categorization. Therefore, the traditional $tf.idf$ representation scheme may lose its ability to discriminate these positive documents from the negative ones with respect to the three terms.

Based on this analysis, we propose a new factor rf (*relevance frequency*) to improve the term’s discriminating power. It is defined as in Equation 1.

$$rf = \log\left(2 + \frac{a}{c}\right) \quad (1)$$

We assign the constant value 2 in the rf formula because the base of this logarithmic operation is 2. Compared with the other known collection frequency factors, such as idf , χ^2 , *information gain* and *Odds Ratio*, the rf factor does not involve the d value. This is based on the following observation. Since the d value is much larger than a, b and c , and

also d dominates the results of the other formulae, it lessens the significance of a and c in expressing the term’s discriminating power for text categorization.

Therefore, in the above case, we weight $t1$ more than $t2$ and $t3$ since $t1$ contributes more to the positive category by using rf factor. The reason why we give more weight to the terms which are assigned more in the positive documents than in the negative ones is that the positive documents belong to one category while all negative documents spread over the other remaining categories.

To elaborate the difference between idf and rf , let us look at the four examples selected from the Reuters News Corpus in combination with two categories. Table 3 lists the idf and rf value of four terms in combination with two categories, namely, 00_acq and 03_earn respectively. Based on

Table 3: Comparison of the weighting value of four features in Category 00_acq and 03_earn

Feature	Category: 00_acq		Category: 03_earn	
	idf	rf	idf	rf
<i>acquir</i>	3.553	4.368	3.553	1.074
<i>stake</i>	4.201	2.975	4.201	1.082
<i>payout</i>	4.999	1	4.999	7.820
<i>dividend</i>	3.567	1.033	3.567	4.408

the literal meaning, the first two terms, *acquir* and *stake*, are closely related to the content of category 00_acq while the last two terms, *payout* and *dividend*, are closely related to the content of category 03_earn . However, as the idf factor neglects the category information of the training set, each of these four terms is weighted equally by the idf factor in terms of the two categories. On the contrary, by adopting the rf scheme which takes the category information into account, each term is assigned more appropriate weights in terms of different categories.

Combined Term Weighting Methods

In the present study, we pick out eight different supervised and unsupervised term weighting methods listed in Table 4. These eight weighting methods are chosen due to their re-

Table 4: Summary of Eight Supervised and Unsupervised Term Weighting Methods

Methods	Denotation	Description
Unsupervised Term Weighting	$binary$ tf $tf.idf$	0 for absence, 1 for presence term frequency alone classic $tf.idf$
Supervised Term Weighting	$tf.rf$ rf $tf.\chi^2$ $tf.ig$ $tf.or$	our newly proposed scheme rf factor alone $tf.chi^2$ $tf.information\ gain$ $tf.Odds\ Ratio$

ported superior classification results or their typical representation in text categorization.

Experiments

Inductive Learning Algorithms

We choose two promising learning algorithms in this study, namely, kNN and SVM, for two reasons. First, the two algorithms have shown better performance than other algorithms in previous studies (Dumais *et al.* 1998), (Joachims 1998) and (Yang & Liu 1999). Second, besides the *binary* text representation, the floating number format of term weights can be used by these two algorithms.

Specifically, we adopt the linear SVM rather than non-linear SVM for the following reasons. First, linear SVM is simple and fast (Dumais *et al.* 1998). Second, our preliminary experimental results have shown that the linear SVM performs better than the non-linear models and this result is also consistent with the findings in (Yang & Liu 1999) and (Dumais *et al.* 1998). Third, our current focus is on the comparison of term weighting methods rather than how to tune the parameters of kernel functions. The SVM software we used is LIBSVM-2.8 (Chang & Lin 2001).

kNN algorithm is very simple and effective, but the most important drawback is its inefficiency in the case of high dimensional and large-scale data sets as it actually does not have a true learning phase and thus incur a high computational cost at the classification time. Unlike the previous work in which k was set as 30-45 (Yang & Liu 1999) and (Joachims 1998), our preliminary results have indicated that when k is 1 the text classifier showed a better classification performance than when k is 30.

Data Corpora

In this study, two widely-used data corpora are used as the benchmark data sets, i.e. Reuters corpus and 20 Newsgroups corpus.

Reuters News Corpus According to the ModApte split, the 9980 news stories from the top 10 largest categories of the Reuters-21578 corpus have been partitioned into a training set of 7193 documents and a test set of 2787 documents. Stop words, punctuation and numbers were removed and the Porter’s stemming was done (Porter 1980). The resulting vocabulary has 15937 words (features). One noticeable issue of the Reuters corpus is the skewed category distribution problem. Among the 7193 training documents, the most common category (*earn*) has a training set frequency of 2877 (40%), but 80% of the categories have less than 7.5% instances.

20 Newsgroups Corpus The 20 Newsgroups corpus is a collection of approximate 20,000 newsgroup documents nearly evenly divided among 20 discussion groups. Some newsgroups are closely related to each other while others are highly unrelated. After removing duplicates and headers, the remaining 18846 documents are sorted by date and partitioned into 11314 training documents (60%) and 7532 test documents (40%). Compared with the skewed category distribution in the Reuters corpus, the 20 categories in the 20 Newsgroups corpus are approximate uniform distribution.

Table 5: McNemar’s test contingency table

n_{00} : Number of examples misclassified by both classifiers f_A and f_B	n_{01} : Number of examples misclassified by f_A but not by f_B
n_{10} : Number of examples misclassified by classifiers f_B but not by f_A	n_{11} : Number of examples misclassified by neither f_A nor f_B

Performance Evaluation

Classification effectiveness is usually measured by using *precision*(p) and *recall*(r). *Precision* is the proportion of truly positive examples labelled positive by the system that were truly positive and *recall* is the proportion of truly positive examples that were labelled positive by the system. The F_1 function which combines *precision* and *recall* is computed as:

$$F_1 = \frac{2.p.r}{p+r} \quad (2)$$

Usually, F_1 function is estimated from two ways, i.e. micro-averaged and macro-averaged. The two measures may give quite different results, that is, the ability of a classifier to behave well also on categories with low generality (i.e., categories with few positive training instances) will be emphasized by macro-averaged and much less so by micro-averaged.

Significance Tests

We employ the McNemar’s significance tests (Dietterich 1998) in terms of micro-averaged F_1 measures to compare the performance between two term weighting methods. Two classifiers f_A and f_B based on two different term weighting methods are performed on the test set. For each example in the test set, we record how it is classified and construct the contingency table as shown in Table 5. Then the statistic χ is defined as

$$\chi = \frac{(|n_{01} - n_{10}| - 1)^2}{n_{01} + n_{10}} \quad (3)$$

where n_{01} and n_{10} are defined in Table 5. Under the null hypothesis, χ is approximately distributed as χ^2 distribution with 1 degree of freedom, where the significance levels 0.01 and 0.001 correspond to the two thresholds $\chi_0 = 6.64$ and $\chi_1 = 10.83$ respectively.

Experimental Results

The experimental results of these eight supervised and unsupervised term weighting methods with respect to micro-averaged F_1 measure on two data collections in combination with two learning algorithms are reported from Figure 1 to Figure 4.

Figure 1 depicts the results on the Reuters corpus using SVM. The trends are distinctive in that the micro-averaged F_1 points of different term weighting methods generally increase as the number of features grows. All term weighting methods reach a peak at the full vocabulary and the best three micro-averaged F_1 points 0.9272, 0.9232 and 0.9191 are reached using $tf.rf$, tf and rf , respectively.

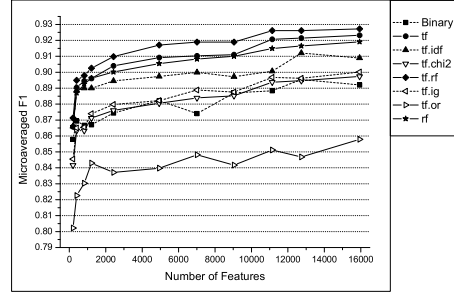


Figure 1: Results of different methods on the Reuters-21578 using linear SVM algorithm

Figure 2 depicts the results on the Reuters corpus using k NN. Unlike the performance using SVM (Figure 1), each term weighting method reaches a peak at a small feature set size of 405 in micro-averaged F_1 . As the number of features grows, the performance of all methods declines except for $tf.ig$ and $tf.\chi^2$. The best two micro-averaged F_1 points 0.8404 and 0.8399 are achieved by using *binary* and *tf.rf* methods at the feature set size of 405. When the number of features is more than 1000, the built-in computation inefficiency of k NN algorithm prevented us from conducting further experiments.

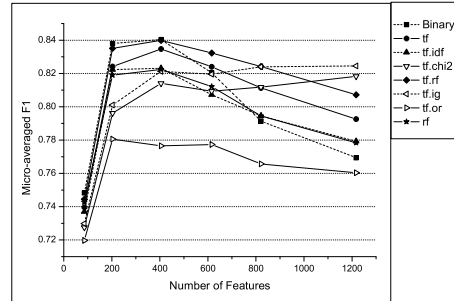


Figure 2: Results of different methods on the Reuters-21578 using k NN algorithm

Figure 3 depicts the results on the 20 Newsgroups corpus using SVM. The trends of the curves are similar to those in Figure 1. However, these curves almost approach a plateau when the number of features exceeds 5000. The best three micro-averaged F_1 points 0.8081, 0.8038 and 0.8012 are reached at the full vocabulary, using rf , $tf.rf$ and $tf.idf$, respectively.

Figure 4 depicts the results on the 20 Newsgroups corpus using k NN. The trends of curves are generally similar to those in Figure 2 (Reuters and k NN). Almost all the curves reach a peak at a small features size around 500 except for $tf.rf$ and rf which show a tendency to increase slowly as the number of features grows. The best two micro-averaged F_1 points 0.6913 and 0.6879 are achieved by using $tf.rf$

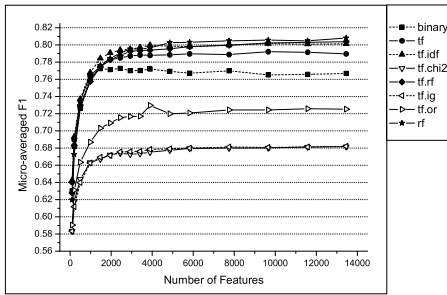


Figure 3: Results of different methods on the 20 News-groups using linear SVM algorithm

and rf at the feature set size of 2000.

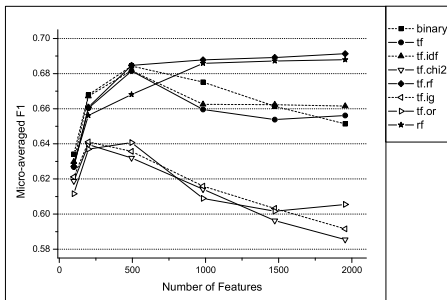


Figure 4: Results of different methods on the 20 News-groups using kNN algorithm

Discussion

We stated that the performance of different term weighting methods is closely related to the learning algorithm (SVM or kNN) and the property of the data corpus (skewed or uniform category distribution). That is, the comparison of supervised and unsupervised term weighting methods should be studied in conjunction with the text classifier and the property of the data corpus.

McNemar’s Significance Tests We use the McNemar’s tests to validate if there is significant difference between two term weighting methods in terms of the micro-averaged F_1 measure. Table 6 summarizes the McNemar’s statistical significance tests results of different methods at a certain feature set size where most of the methods reach their best performance. Although these methods achieve their best performance at different numbers of features, the results in Table 6 show approximate ranks of these methods since most of them show consistent performance with respect to each other as the number of features grows given specific data corpus and learning algorithm.

Effects of Feature Set Size on Algorithms The size of the feature set at which each term weighting method reaches

the peak is closely dependent upon the learning algorithm in use rather than the term weighting method itself and the benchmark data collection.

Specifically, for the SVM-based text classifier, almost all term weighting methods achieve their best performance when inputting the full vocabulary (Figure 1 and Figure 3). These findings are entirely consistent with those in (Debole & Sebastiani 2003) and (Joachims 1998). However, for the kNN -based text classifier, all term weighting methods achieve their best performance at a small feature set size. For example, most methods reach a peak at the feature set size of 400 in Figure 2 and 500 in Figure 4.

The possible explanation may be the increasing noise with the larger feature set. SVM is resilient to noise because only the *support vectors* are effective for the classification performance. However, kNN is an example-based learning algorithm, thus all the inputting features’ noise has impact on the classification performance.

Summary of Different Term Weighting Methods Generally, these supervised and unsupervised term weighting methods have not shown a universal consistent performance on the two corpora and the two different algorithms. However, there is one exception, that is, our $tf.rf$ method consistently shows the best performance with respect to the two different algorithms and the two text corpora. Moreover, the rf alone method also shows a comparable performance to $tf.rf$ in most experiments except for on Reuters data set using the kNN algorithm (Figure 2).

Specifically, the three typical supervised methods based on the information theory, i.e. $tf.\chi^2$, $tf.ig$ and $tf.or$, are the worst methods among these eight methods. These findings indicate that these sophisticated methods based on information theoretic functions have no superiority over the simpler unsupervised ones. This is contrary to our original expectation that the supervised term weighting methods which consider the document distribution in the training documents should always be better than the unsupervised ones.

Moreover, the three unsupervised term weighting methods, i.e. tf , $tf.idf$ and $binary$, show mixed results with respect to each other. For example, tf is better than $tf.idf$ in Figure 1 and but it is the other way round in Figure 3. However, in Figure 4 the performance of the three methods are comparable to each other. These findings indicate that the performance of the unsupervised term weighting methods is dependent on the special data corpus and the learning algorithm in use. Three general observations can be made here. First, kNN favors $binary$ while SVM does not. Second, the good performance of $tf.idf$ in Figure 3 and Figure 4 may be attributed to the uniform category distribution. Third, although tf does not have a comparable performance to $tf.rf$ in all experiments, it outperforms many other methods consistently and significantly.

Concluding Remarks

The following conclusions are drawn from our controlled experimental results.

Generally, not all supervised term weighting methods have a consistent superiority over unsupervised term weight-

Table 6: Statistical significance tests results of different methods. The term weighting methods with insignificant performance differences are grouped into one set and ">" and ">>" denote better than at significance level 0.01 and 0.001 respectively.

Algorithm	Data Corpus	#_Features	McNemar's Test Results
SVM	Reuters	15937	$(tf.rf, tf, rf) > (tf.idf) > (tf.ig, tf.\chi^2, binary) >> tf.or$
SVM	20 Newsgroups	13456	$(rf, tf.rf, tf.idf) > tf >> binary >> tf.or >> (tf.ig, tf.\chi^2)$
kNN	Reuters	405	$(binary, tf.rf) > tf >> (tf.idf, rf, tf.ig) > tf.\chi^2 >> tf.or$
kNN	20 Newsgroups	494	$(tf.rf, binary, tf.idf, tf) >> rf >> (tf.or, tf.ig, tf.\chi^2)$

ing methods. Specifically, the three supervised methods based on the information theory, i.e. $tf.\chi^2$, $tf.ig$ and $tf.or$, have shown to perform rather poorly in all experiments. On the other hand, newly proposed supervised method, $tf.rf$, achieves the best performance consistently and outperforms other methods substantially and significantly.

Neither $tf.idf$ nor $binary$ shows consistent performance. Specifically, $tf.idf$ has comparable good performance as $tf.rf$ on the uniform category distribution corpus while $binary$ is comparable to $tf.rf$ on the kNN-based text classifier. On the other hand, although tf does not perform as well as $tf.rf$, it performs consistently well and outperforms other methods consistently and significantly.

We suggest that $tf.rf$ be used as term weighting method for text categorization task as it performed consistently well on the two benchmark data collections with either skewed or uniform category distribution in combination with either SVM-based or kNN-based text classifiers.

We should point out that the observations above are made based on the controlled experiments. It will be interesting to see in our future work if we can observe the similar results on more general experimental settings, such as different learning algorithms, different performance measures and other benchmark collections. In addition, since the term weighting is the most basic component of text preprocessing methods, we expect that our weighting method can be integrated into various text mining tasks, such as information retrieval, text summarization and so on.

References

- Chang, C.-C., and Lin, C.-J. 2001. *LIBSVM: a library for support vector machines*. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- Debole, F., and Sebastiani, F. 2003. Supervised term weighting for automated text categorization. In *Proceedings of the 2003 ACM symposium on Applied computing*, 784–788. ACM Press.
- Deng, Z.-H.; Tang, S.-W.; Yang, D.-Q.; Zhang, M.; Li, L.-Y.; and Xie, K. Q. 2004. A comparative study on feature weight in text categorization.
- Dietterich, T. G. 1998. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Comput.* 10(7):1895–1923.
- Dumais, S.; Platt, J.; Heckerman, D.; and Sahami, M. 1998. Inductive learning algorithms and representations for text categorization. In *Proceedings of the seventh international conference on Information and knowledge management*, 148–155. ACM Press.
- Joachims, T. 1998. Text categorization with support vector machines: learning with many relevant features. In Nédellec, C., and Rouveirol, C., eds., *Proceedings of ECML-98, 10th European Conference on Machine Learning*, number 1398, 137–142. Chemnitz, DE: Springer Verlag, Heidelberg, DE.
- Lan, M.; Sung, S.-Y.; Low, H.-B.; and Tan, C.-L. 2005. A comparative study on term weighting schemes for text categorization. In *Proceedings of the International Joint Conference on Neural Networks 2005*.
2002. Text categorization with support vector machines. how to represent texts in input space? *Machine Learning* 46(1-3):423 – 444.
- Mladenic, D.; Brank, J.; Grobelnik, M.; and Milic-Frayling, N. 2004. Feature selection using linear classifier weights: interaction with classification models. In *Proceedings of the 27th annual international conference on Research and development in information retrieval*, 234–241. ACM Press.
- Mori, T. 2002. Information gain ratio as term weight: the case of summarization of ir results. In *Proceedings of the 19th International Conference on Computational Linguistics*, 688–694. Association for Computational Linguistics Publisher.
- Porter, M. 1980. An algorithm for suffix stripping. *Program* 130–137.
- Salton, G., and Buckley, C. 1988. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.* 24(5):513–523.
- Wu, H., and Salton, G. 1981. A comparison of search term weighting: term relevance vs. inverse document frequency. In *SIGIR '81: Proceedings of the 4th annual international ACM SIGIR conference on Information storage and retrieval*, 30–39. New York, NY, USA: ACM Press.
- Yang, Y., and Liu, X. 1999. A re-examination of text categorization methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 42 – 49. ACM Press.
- Yang, Y., and Pedersen, J. O. 1997. A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, 412–420. Morgan Kaufmann Publishers Inc.