A Study on Term Weighting for Text Categorization: A Novel Supervised Variant of tf.idf

Giacomo Domeniconi[1], Gianluca Moro[1], Roberto Pasolini[1] and Claudio Sartori[2]
[1]*DISI, Università Degli Studi Di Bologna, Via Venezia 52, 47523 Cesena, Italy*
[2]*DISI, Università Degli Studi Di Bologna, Viale del Risorgimento 2, 40136 Bologna, Italy*

Keywords: Term Weighting, Supervised Term Weighting Scheme, Text Categorization, tfidf, Text Representation.

Abstract: Within text categorization and other data mining tasks, the use of suitable methods for term weighting can bring a substantial boost in effectiveness. Several term weighting methods have been presented throughout literature, based on assumptions commonly derived from observation of distribution of words in documents. For example, the idf assumption states that words appearing in many documents are usually not as important as less frequent ones. Contrarily to tf.idf and other weighting methods derived from information retrieval, schemes proposed more recently are supervised, i.e. based on knownledge of membership of training documents to categories. We propose here a supervised variant of the tf.idf scheme, based on computing the usual idf factor without considering documents of the category to be recognized, so that importance of terms frequently appearing only within it is not underestimated. A further proposed variant is additionally based on relevance frequency, considering occurrences of words within the category itself. In extensive experiments on two recurring text collections with several unsupervised and supervised weighting schemes, we show that the ones we propose generally perform better than or comparably to other ones in terms of accuracy, using two different learning methods.

1 INTRODUCTION

Text categorization (or *text classification*) is the task of automatically labeling natural language text documents with predefined categories of topics. One of the major problems in working with text is the intrinsic unstructured form of documents: a suitable derived representation must be used in an automated classification process.

The most widely used approach for the structured representation of textual data is the Vector Space Model (VSM), where each document is represented by a vector in a high-dimensional space, also known as *bag of words*. Each dimension of this space represents a word, or *term* equivalently. It is therefore necessary a data preprocessing phase to extract words from documents and to assign them weights according to their importance in each document. These weights are assigned according to a chosen *term weighting scheme*.

Different possible term weighting schemes have been developed throughout the literature. The choice of a suitable scheme can significantly affect the effectiveness of the classification. For example, (Leopold and Kindermann, 2002) thoroughly test different weighting schemes using SVM classifiers, experiencing substantial gaps between accuracies with different schemes. The choice of the term weighting scheme is important not only for text classification, but also for other new types of text mining tasks, such as sentiment analysis (Deng et al., 2014; Paltoglou and Thelwall, 2010), cross-domain classification (Domeniconi et al., 2014) and novelty mining (Tsai and Kwee, 2011). From scientific research, the use of term weighting schemes has moved to practical applications, with its employment in projects of major IT enterprises such as Yahoo! (Carmel et al., 2014) and IBM (Papineni, 2001).

A typical term weighting scheme is the composition of a *local* and a *global* factor: while the former indicates the importance of each term within each document regardless of the other documents, the latter weights the discriminating power of each term throughout the whole collection of documents. Once computed, weights are also often normalized to prevent bias to longer documents.

Many studies in the literature are focused on the global factor. Methods to compute it fall into two

Domeniconi G., Moro G., Pasolini R. and Sartori C..
A Study on Term Weighting for Text Categorization: A Novel Supervised Variant of tf.idf.
DOI: 10.5220/0005511900260037
In *Proceedings of 4th International Conference on Data Management Technologies and Applications* (DATA-2015), pages 26-37
ISBN: 978-989-758-103-8

types: *supervised* methods leverage known information about the membership of training documents to categories, while *unsupervised* methods do not use this information and are only based on distribution of terms across documents. Unsupervised term weighting schemes are generally borrowed from information retrieval (IR); the most widely used is *tf.idf*, proposed by (Sparck Jones, 1988) and justified, in the IR field, by (Robertson, 2004). Supervised weighting schemes, naturally employable in text categorization, have been proposed in more recent times: they are usually based on distribution of terms across categories; one possibility is to use techniques usually employed for feature selection (Debole and Sebastiani, 2003).

We propose here a supervised variant of the *tf.idf* scheme. The intuition behind the basic *idf* factor is that terms appearing frequently throughout the collection have minor discriminating power. This is not always true in text classification, because if a term occurs in a high number of documents belonging to the same category, then we can assert that the term is very effective in discriminating that category from the others. From this intuition, we in practice compute the *idf* factor considering only documents outside of the category under consideration, in order to prevent the frequent appearance of a term in the same category from underestimating its importance. In a second variant that we propose, we combine the intuition above with the relevance frequency factor of *tf.rf* (Lan et al., 2009), in order to also positively consider the appearance of a term in documents of the category under analysis.

The paper is organized as follows. Section 2 presents related works on term weighting methods, focusing on unsupervised and supervised methods for global weighting. In Section 3, we present and motivate the two proposed variants of *idf*. Section 4 presents the general setup of our experimental evaluation, whose results are presented and discussed in Section 5. Conclusive remarks are given in Section 6.

2 RELATED WORK

The problem of text categorization has been extensively investigated in the past years, considering the ever-increasing applications of this discipline, such as news or e-mail filtering and organization, indexing and semantic search of documents, sentiment analysis and opinion mining, prediction of genetic diseases, etc. (Sebastiani, 2002)

In the machine learning approach, a knowledge model to classify documents within a set $\mathcal{C} = \{c_1, c_2, \ldots, c_{|\mathcal{C}|}\}$ of categories is built upon a training set $\mathcal{D}_T = \{d_1, d_2, \ldots, d_{|\mathcal{D}_T|}\}$ of documents with a known labeling $L : \mathcal{D}_T \times \mathcal{C} \rightarrow \{0,1\}$ ($L(d,c) = 1$ if and only if document d is labeled with c). In order to leverage standard machine learning algorithms, documents are generally pre-processed to be represented in a Vector Space Model (VSM).

In the VSM, the content of a document d_j is represented as a vector $\mathbf{w}^j = \{w_1^j, w_2^j, \ldots, w_n^j\}$ in a n-dimensional vector space $\mathbb{R}^n$, where w_i is a weight that indicates the importance of a term t_i in d_j. Terms $t_1, t_2, \ldots, t_n$ constitute a set of features, shared across all documents. In other words, each weight w_i^j indicates how much the term t_i contributes to the semantic content of d_j.

Weights for each term-document couple are assigned according to a predefined term weighting scheme, which must meaningfully estimate the importance of each term within each document.

Three are the considerations discussed in the years regarding the correct assignment of weights in text categorization (Debole and Sebastiani, 2003):

1. the multiple occurrence of a term in a document appears to be related to the content of the document itself (*term frequency* factor);
2. terms uncommon throughout a collection better discriminate the content of the documents (*collection frequency* factor);
3. long documents are not more important than the short ones, normalization is used to equalize the length of documents.

Referring to these considerations, most term weighting schemes can be broken into a *local* (*term frequency*) factor and a *global* (*collection frequency*) factor. Normalization is applied on a per-document basis after computing these factors for all terms, usually by means of *cosine normalization*.

$$\mathbf{w}^j_{\text{normalized}} = \frac{1}{\sqrt{\sum_{i=1}^{n}(w_i^j)^2}} \cdot \mathbf{w}^j \qquad (1)$$

There are several ways to calculate the local term frequency factor, which are summarized in Table 1. The simplest one is binary weighting, which only considers the presence (1) or absence (0) of a term in a document, ignoring its frequency. The perhaps most obvious possibility is the number of occurrencies of the term in the document, which is often the intended meaning of "term frequency" (*tf*). Other variants have been proposed, for example the *logarithmic tf*, computed as $\log(1+tf)$, is now practically the standard local factor used in literature (Debole and Sebastiani, 2003). Another possible scheme is the inverse term

Table 1: Term frequency factors.

Term Frequency Factor	Notation	Description
1/0	BIN	Presence or absence of terms in the document
term frequency	TF	Number of times the term occurs in the document
$\log(1+tf)$	log TF	Logarithm of tf
$1-\frac{r}{r+tf}$	ITF	Inverse term frequency, usually $r=1$
$S(V_a)=$ $(1-d)+d\cdot\sum_{V_b\in Conn(V_A)}\frac{S(V_b)}{\lvert Conn(V_b)\rvert}$	RW	Given a graph $G=(V,E)$, let $Conn(V)$ be the set of vertices connected to V. Typical value for d is 0.85.

frequency, proposed by (Leopold and Kindermann, 2002). Another way to assign the term frequency factor was proposed by (Hassan and Banea, 2006), inspired by the PageRank algorithm: they weight terms using a random walk model applied to a graph encoding words and dependencies between them in a document. Each word of the document is modeled in the graph as a node and an edge (bidirectional, unlike PageRank) connects two words if they co-occur in the document within a certain windows size,. In this work, we have chosen logarithmic term frequency ($\log(1+tf)$) as the local factor for all experiments.

As mentioned earlier, the global collection frequency factor can be *supervised* or *unsupervised*, depending whether it leverages or not the knowledge of membership of documents to categories. In the following, are summarized some of the most used and recent methods proposed in the literature of both types.

2.1 Unsupervised Term Weighting Methods

Generally, unsupervised term weighting schemes, not considering category labels of documents, derive from IR research. The most widely unsupervised method used is *tf.idf*, which (with normalization) perfectly embodies the three assumptions previously seen. The basic idea is that terms appearing in many documents are not good for discrimination, and therefore they will weight less than terms occurring in few documents. Over the years, researchers have proposed several variations in the way they calculate and combine the three basic assumptions (*tf*, *idf* and normalization), the result is the now standard variant "ltc", where $tf(t_i,d_j)$ is the *tf* factor described above denoting the importance of t_i within document d_j. In the following, the form "$t_i \in d_x$" is used to indicate that term t_i appears at least once in document d_x.

$$tf.idf(t_i,d_j) = tf(t_i,d_j)\cdot\log\left(\frac{|\mathcal{D}_T|}{|d_x\in\mathcal{D}_T : t_i\in d_x|}\right) \quad (2)$$

The *idf* factor multiplies the *tf* for a value that is greater when the term is rare in the collection of training documents $\mathcal{D}_T$. The weights obtained by the formula above are then normalized according to the third assumption by means of cosine normalization (Eq. 1).

(Tokunaga and Makoto, 1994) propose an extension of the *idf* called *Weighted Inverse Document Frequency* (*widf*), given by dividing the $tf(t_i,d_j)$ by the sum of all the frequencies of t_i in all the documents of the collection:

$$widf(t_i) = \frac{1}{\sum_{d_x\in\mathcal{D}_T} tf(t_i,d_x)} \quad (3)$$

(Deisy et al., 2010) propose a combination of *idf* and *widf*, called *Modified Inverse Document Frequency* (*midf*) that is defined as follows:

$$midf(t_i) = \frac{|d_x\in\mathcal{D}_T : t_i\in d_x|}{\sum_{d_x\in\mathcal{D}_T} tf(t_i,d_x)} \quad (4)$$

Of course the simplest choice, sometimes used, is to not use a global factor at all, setting it to 1 for all terms and only considering term frequency.

2.2 Supervised Term Weighting Methods

Since text categorization is a supervised learning task, where the knowledge of category labels of training documents is necessarily available, many term weighting methods use this information to supervise the assignment of weights to each term.

A basic example of supervised global factor is *inverse category frequency* (*icf*):

$$icf(t_i) = \log\left(\frac{|\mathcal{C}|}{|c_x\in\mathcal{C} : t_i\in c_x|}\right) \quad (5)$$

where "$t_i \in c_x$" denotes that t_i appears in at least one document labeled with c_x. The idea of the *icf* factor is similar to that of *idf*, but using the categories instead of the documents: the fewer are the categories in which a term occurs, the greater is the discriminating power of the term.

Within text categorization, especially in the multi-label case where each document can be labeled with an arbitrary number of categories, it is common to

train one binary classifier for each one of the possible categories. For each category c_k, the corresponding model must separate its *positive examples*, i.e. documents actually labeled with c_k, from all other documents, the *negative examples*. In this case, it is allowed to compute for each term t_i a distinct collection frequency factor for each category c_k, used to represent documents in the VSM only in the context of that category.

In order to summarize the various methods of supervised term weighting, we show in Table 2 the fundamental elements usually considered by these schemes and used in the following formulas to compute the global importance of a term t_i for a category c_k.

- A denotes the number of documents belonging to category c_k where the term t_i occurs at least once;
- C denotes the number of documents not belonging to category c_k where the term t_i occurs at least once;
- dually, B denotes the number of documents belonging to c_k where t_i does not occur;
- D denotes the number of documents not belonging to c_k where t_i does not occur.

The total number of training documents is denoted with $N = A+B+C+D = |\mathcal{D}_T|$

Table 2: Fundamental elements of supervised term weighting.

	c_k	$\overline{c_k}$
t_i	A	C
$\overline{t_i}$	B	D

In this notation, the *ltc-idf* factor is expressed as:

$$idf = \log\left(\frac{N}{A+C}\right) \tag{6}$$

As suggested by (Debole and Sebastiani, 2003), an intuitive approach to supervised term weighting is to employ common techniques for feature selection, such as χ^2, *information gain*, *odds ratio* and so on.

(Deng et al., 2004) uses the χ^2 factor to weigh terms, replacing the *idf* factor, and the results show that the *tf.*χ^2 scheme is more effective than *tf.idf* using a SVM classifier. Similarly (Debole and Sebastiani, 2003) apply feature selection schemes multiplied by the *tf* factor, by calling them "supervised term weighting". In this work they use the same scheme for feature selection and term weighting, in contrast to (Deng et al., 2004) where different measures are used. The results of the two however are in contradiction: (Debole and Sebastiani, 2003) shows that the *tf.idf* always outperforms χ^2, and in general the supervised methods not give substantial improvements compared to unsupervised *tf.idf*. The widely-used collection frequency factors χ^2, information gain (*ig*), odds ratio (*or*) and mutual information (*mi*) are described as follows:

$$\chi^2 = N \cdot \frac{(A\cdot D - B\cdot C)^2}{(A+C)\cdot(B+D)\cdot(A+B)\cdot(C+D)} \tag{7}$$

$$ig = -\frac{A+B}{N}\cdot\log\frac{A+B}{N} + \frac{A}{N}\cdot\log\frac{A}{A+C} + \frac{B}{N}\cdot\log\frac{B}{B+D} \tag{8}$$

$$or = \log\left(\frac{A\cdot D}{B\cdot C}\right) \tag{9}$$

$$mi = \log\left(\frac{A\cdot N}{(A+B)\cdot(A+C)}\right) \tag{10}$$

Any supervised feature selection scheme can be used for the term weighting. For example, the *gss* extension of the χ^2 proposed by (Galavotti et al., 2000) eliminates N at numerator and the emphasis to rare features and categories at the denominator.

$$gss = \frac{A\cdot D - B\cdot C}{N^2} \tag{11}$$

(Largeron et al., 2011) propose a scheme called *Entropy-based Category Coverage Difference* (*eccd*) based on the distribution of the documents containing the term and its categories, taking into account the entropy of the term.

$$eccd = \frac{A\cdot D - B\cdot C}{(A+B)\cdot(C+D)}\cdot\frac{E_{max} - E(t_i)}{E_{max}} \tag{12}$$

$$E(t_i) = \textit{Shannon Entropy} = -\sum_{c_k\in C} tf_i^k \cdot \log_2 tf_i^k$$

where tf_i^k is the term frequency of the term t_i in the category c_k.

(Liu et al., 2009) propose a *prob-based* scheme, combining the ratios A/C measuring the relevance of a term t_i for the category c_k and A/B, since a term with this ratio high means that it is often present in the documents of c_k and thus highly representative.

$$\textit{prob-based} = \log\left(1 + \frac{A}{B}\cdot\frac{A}{C}\right) \tag{13}$$

Another similar scheme is *tf.rf*, proposed by (Lan et al., 2009): it takes into account the terms distribution in the positive and negative examples, stating that, for a multi-label text categorization task, the higher the concentration of high-frequency terms in the positive examples than in the negative ones, the greater the contribution to categorization.

$$rf = \log\left(2 + \frac{A}{\max(1,C)}\right) \tag{14}$$

Combining this idea with the *icf* factor, (Wang and Zhang, 2013) propose a variant of *tf.icf* called *icf-based*.

$$icf\text{-}based = \log\left(2 + \frac{A}{\max(1,C)} \cdot \frac{|C|}{|c_x \in \mathcal{C} : t_i \in c_x|}\right) \quad (15)$$

(Ren and Sohrab, 2013) implement a category indexing-based *tf.idf.icf* observational term weighting scheme, where the inverse category frequency is incorporated in the standard *tf.idf.icf* to favor the rare terms and is biased against frequent terms. Therefore, they revised the *icf* function implementing a new inverse category space density frequency ($ics_\delta f$), generating the $tf.idf.ics_\delta f$ scheme that provides a positive discrimination on infrequent and frequent terms. The inverse category space density frequency is denoted as:

$$ics_\delta f(t_i) = \log\left(\frac{|C|}{\sum_{c_x \in C} C_\delta(t_i, c_x)}\right) \quad (16)$$

$$C_\delta(t_i, c_x) = \frac{A}{A + C}$$

(Song and Myaeng, 2012) proposes a term weighting scheme that leverages availability of past retrieval results, consisting of queries that contain a particular term, retrieved documents, and their relevance judgments. They assign a term weight depending on the degree to which the mean frequency values for the past distributions of relevant and non-relevant documents are different. More precisely, it takes into account the rankings and similarity values of the relevant and non-relevant documents. (Ropero et al., 2012) introduce a novel fuzzy logic-based term weighting scheme for information extraction.

Another different approach to supervise term weighting is proposed by (Luo et al., 2011): they do not use the statistical information of terms in documents like methods mentioned above, but a term weighting scheme that exploits the semantics of categories and terms. Specifically, a category is represented by the semantic sense, given by the lexical database WordNet, of the terms contained in its own label; while the weight of each term is correlated to its semantic similarity with the category. (Bloehdorn and Hotho, 2006) propose a hybrid approach for document representation based on the common term stem representation enhanced with concepts extracted from ontologies.

3 A SUPERVISED VARIANT OF INVERSE DOCUMENT FREQUENCY

Here we introduce a supervised variant of *tf.idf*. The basic idea of our proposal is to avoid decreasing the weight of terms contained in documents belonging to the same category, so that words that appear in several documents of the same category are not disadvantaged, as instead happens in the standard formulation of *idf*. We refer to this variant with the name *idfec* (Inverse Document Frequency Excluding Category). Therefore, the proposed category frequency factor scheme is formulated as:

$$idfec\ (t_i, c_k) = \log\left(\frac{|\mathcal{D}_T \setminus c_k| + 1}{\max\left(1, |d \in \mathcal{D}_T \setminus c_k : t_i \in d|\right)}\right) \quad (17)$$

where "$\mathcal{D}_T \setminus c_k$" denotes training documents not labeled with c_k. Using the previous notation, the formula becomes:

$$tf.idfec\ (t_i, d_j) = tf(t_i, d_j) \cdot \log\left(\frac{C + D}{\max\left(1, C\right)}\right) \quad (18)$$

Note that with this variant of *idf* we can have particular cases. If the *i*-th word is only contained in *j*-th document, or only in documents belonging to c_k, the denominator becomes 0. To prevent division by 0, the denominator is replaced by 1 in this particular case.

The *tf.idfec* scheme is expected to improve classification effectiveness over *tf.idf* because it discriminates where each term appears. For any category c_k, the importance of a term appearing in many documents outside of it is penalized as in *tf.idf*. On the other side, the importance is not reduced by appearances in the positive examples of c_k, so that any term appearing mostly within the category itself retains an high global weight.

This scheme is similar to *tf.rf* (Lan et al., 2009), as both penalize weights of a term t_i according to the number of negative examples where the t_i appears. The difference is in the numerator of the fraction, which values positive examples with the term in *rf* and negative ones without it in *idfec*.

To illustrate these properties, we use the following numerical example. Considering the notation shown in Table 2, suppose we have a corpus of 100 training documents divided as shown in Table 3, for two terms t_1 and t_2 and a category c_k.

We can easily note how the term t_1 is very representative, and then discriminant, for the category c_k since it is very frequent within it $(A/(A+B)) = 27/30)$ and not in the rest of the documents $(C/(C+D) = 5/70)$. Similarly we can see that t_2 does not

Table 3: Example of document distribution for two terms.

	c_k	$\overline{c_k}$		c_k	$\overline{c_k}$
t_1	27	5	t_2	10	25
$\overline{t_1}$	3	65	$\overline{t_2}$	20	45

seem to be a particularly discriminating term for c_k. In the standard formulation, the *idf* is

$$idf(t_1) = \log(100/(27+5)) = \log(3.125)$$

and for our best competitor *rf* is

$$rf(t_1) = log(2+27/5) = log(7.4)$$

while with the *idfec* we obtain

$$idfec\ (t_1) = \log((65+5)/5) = \log(14)$$

For t_2 we have instead:

$$idf(t_2) = \log(100/(10+25)) = \log(2.857)$$
$$rf(t_2) = log(2+10/25) = log(2.4)$$
$$idfec\ (t_2) = \log((45+25)/25) = \log(2.8)$$

We can see that our supervised version of *idf* can separate the weights of the two terms according to the frequency of terms in documents belonging to c_k or not. In fact, while with the standard *idf* the weights of t_1 and t_2 are very similar, with *idfec* t_1 has a weight much greater than t_2 since t_1 is more frequent and discriminative for the category c_k. This kind of behavior is also exhibited by *rf*, but our method yields an even higher weight for the relevant term t_1.

In its base version, *tf.idfec* takes into account only the negative examples (*C* and *D* in Table 2). Instead it could be helpful, especially for the classification task, also to take into account how many documents belonging to c_k contain the term, i.e. how much the term occurs within the category more than in the rest of the collection. Considering this, in a way similar to (Wang and Zhang, 2013), we propose to mix our idea with that of the *rf* in a new version of our weighting scheme, called *tf.idfec-based* (*tf.idfec-b.* for short) and expressed by the following formula:

$$tf.idfec\text{-}b.(t_i,d_j) = tf(t_i,d_j)\cdot\log\left(2+\frac{A+C+D}{\max(1,C)}\right) \quad (19)$$

Using the example in the Table 3, the new term weighting scheme becomes for t_1 and t_2 respectively:

$$idfec\text{-}b.(t_1) = log(2+((27+5+65)/5)) = log(21.4)$$

$$idfec\text{-}b.(t_2) = log(2+((10+25+45)/25)) = log(5.2)$$

With this term weighting scheme, the difference in weight between a very common term (t_2) and a very discriminative one (t_1) is even more pronounced.

4 EXPERIMENTAL SETUP

We performed extensive experimental evaluation to compare the effectiveness of the proposed term weighting approach with other schemes. In the following, we describe in detail the organization of these experiments.

4.1 Benchmark Datasets

We used two commonly employed text collections as benchmarks in our experiments.

The **Reuters-21578** corpus[1] consists in 21,578 articles collected from Reuters. According to the ModApté split, 9,100 news stories are used: 6,532 for the training set and 2,568 for the test set. One intrinsic problem of the Reuters corpus is the skewed category distribution. In the top 52 categories, the two most common categories (*earn* and *acq*) contain, respectively, 43% and 25% of the documents in the dataset, while the average document frequency of all categories is less than 2%. In literature, this dataset is used considering a various number of categories: we considered two views of this corpus, *Reuters-10* and *Reuters-52* where only the 10 and the 52 most frequent categories are considered, respectively.

The **20 Newsgroups** corpus[2] is a collection of 18,828 Usenet posts partitioned across 20 discussion groups. Some newsgroups are very closely related to each other (e.g. comp.sys.ibm.pc. hardware / comp.sys.mac.hardware), while others are highly unrelated (e.g misc.forsale / soc.religion.christian). Likely to (Lan et al., 2009) we randomly selected 60% of documents as training instances and the remaining 40% make up the test set. Contrarily to Reuters, documents of 20 Newsgroups are distributed rather uniformly across categories.

4.2 Classification Process

For each dataset, all documents were pre-processed by removing punctuation, numbers and stopwords from a predefined list, then by applying the common SnowballStemmer to remaining words. In this way, we obtained a total of 16,145 distinct terms in the Reuters-21578 corpus and 61,483 in 20 Newsgroups.

We performed feature selection on these terms to keep only a useful subset of them. Specifically, we extracted for each category the p terms appearing in most of its documents, where for p all the following values were tested: 25, 50, 75, 150, 300, 600, 900,

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

[2] http://people.csail.mit.edu/jrennie/20Newsgroups/.

1200, 1800, 2400, 4800, 9600. This feature selection method may be considered counterproductive since we selected the most common terms, but it is actually correct considering the use of the VSM as the terms result to be as less scattered as possible. The task of term weighting is therefore crucial to increase the categorization effectiveness, giving a weight to each term according to the category to which the documents belong.

Since we tested both supervised and unsupervised term weighting methods, we used two different procedures. For unsupervised methods we processed the training set in order to calculate the collection frequency factor for each term, which was then multiplied by the logarithmic term frequency factor (referred to as *tf* in the following) for each term in training and test set. Finally, cosine normalization (Eq. 1) was applied to normalize the term weights.

For supervised methods we used the multi-label categorization approach, where a binary classifier is created for each category. That is, for each category c_k, training documents labeled with it are tagged as positive examples, while the remaining one constitute negative examples. We computed statistical information related to c_k (as described in Table 2) for each term of training documents. The weight of each term was calculated multiplying its *tf* with the global factor computed on the training set; finally cosine normalization was performed.

4.3 Learning Algorithms

We chose to use support vector machines (SVM), which are usually the best learning approach in text categorization (Lan et al., 2009; Sebastiani, 2002). We used the well known SVM^{Light} implementation[3] (Joachims, 1998), testing both the linear kernel and the radial basis function (RBF) kernel.

Furthermore, to test the effectiveness of classification by varying the term weighting scheme with another algorithm, we used the Weka implementation of *Random Forest* (Breiman, 2001), chosen for both its effectiveness and its speed. As parameters we set the number of trees to $I = 10$ and the number of features to $K = 50$.

4.4 Performance Measures

We measured the effectiveness in terms of precision (π) and recall (ρ), defined in the usual way for text categorization (Lewis, 1995). As a measure of effectiveness that combines π and ρ we used the well-known F_1 measure, defined as:

$$F_1 = \frac{2 \cdot \pi \cdot \rho}{\pi + \rho} \tag{20}$$

For multi-label problems, the F_1 is estimated in two ways: micro-averaged F_1 and macro-averaged F_1 (Sebastiani, 2002). The *micro*-F_1 sums up the individual true positives, false positives and false negatives of the different classifiers and applies them to get the F_1. The *macro*-F_1 is instead the average of the F_1 related to each category.

4.5 Significance Tests

To evaluate the difference of performances between term weighting methods, we employed the McNemar's significance test (Dietterich, 1998; Lan et al., 2005; Lan et al., 2009), used to compare the distribution of counts expected under the null hypotesis to the observed counts.

Let's consider two classifiers f_a and f_b trained from the same documents but with two different term weighting methods and evaluated using the same test set. The outcome of categorization for all test instances is summarized in a contingency table, as shown in Table 4.

Table 4: McNemar's test contingency table.

n_{00}: Number of istances misclassified by both classifiers f_a and f_b	n_{01}: Number of istances misclassified by f_a but not by f_b
n_{10}: Number of istances misclassified by f_b but not by f_a	n_{11}: Number of istances correctly classified by both classifiers f_a and f_b

The null hypotesis for the McNemar's significance test states that on the same test instances, two classifiers f_a and f_b will have the same prediction errors, which means that $n_{01} = n_{10}$. So the χ statistic is defined as:

$$\chi = \frac{(|n_{01} - n_{10}| - 1)^2}{n_{01} + n_{10}} \tag{21}$$

χ is approximately distributed as a χ^2 distribution with 1 degree of freedom, where the significance levels 0.01 and 0.001 correspond respectively to the thresholds $\chi_0 = 6.64$ and $\chi_1 = 10.83$. If the null hypotesis is correct, than the probability that χ is greater than 6.64 is less than 0.01, and similarly 0.001 for a value greater than 10.83. Otherwise we may reject the null hypotesis in favor of the hypotesis that f_a and f_b have different performance and therefore the two term weighting schemes have different impact when used on the particular training set.

[3] http://svmlight.joachims.org/

Table 5: Micro-averaged F_1 (in thousandths) best results obtained with each term weighting scheme ("b." is short for "based") on the three dataset with different learning algorithms. The best result for each dataset and algorithm is marked in bold.

	tf.idfec	*tf.idfec-b.*	*tf.rf*	*tf.icf-b.*	*tf.idf*	*tf.*χ^2	*eccd*	*tf.gss*	*tf.ig*	*midf*	*tf.oddsR*	*rw*	*tf*	*bin*
Reuters-10														
SVM(LIN)	.929	**.933**	**.933**	.930	.930	.847	.839	.863	.852	.920	.918	.914	.932	.916
SVM(RBF)	.932	**.937**	**.937**	.935	.933	.879	.853	.895	.882	.923	.926	.922	.934	.924
RandomForest	**.904**	.902	.903	.899	.903	.901	.885	.901	.903	.898	.903	.899	.902	.897
Reuters-52														
SVM(LIN)	.920	**.925**	.922	.916	.917	.828	.811	.848	.822	.882	.890	.881	.912	.881
SVM(RBF)	.925	**.927**	.926	.924	.922	.848	.828	.873	.848	.886	.895	.887	.915	.887
RandomForest	.855	**.868**	.867	.853	.858	.863	.847	.861	.864	.858	.863	.856	.866	.861
20 Newsgroups														
SVM(LIN)	.754	**.759**	**.759**	.747	.741	.567	.450	.512	.520	.606	.666	.702	.709	.702
SVM(RBF)	.712	**.713**	.712	**.713**	.702	.587	.490	.555	.560	.609	.664	.674	.677	.675
RandomForest	.536	**.570**	.563	.537	.543	.568	.511	**.570**	.565	.536	.562	.556	.566	.555

Table 6: Macro-averaged F_1 (in thousandths) best results obtained with each term weighting scheme ("b." is short for "based") on the three dataset with different learning algorithms. The best result for each dataset and algorithm is marked in bold.

	tf.idfec	*tf.idfec-b.*	*tf.rf*	*tf.icf-b.*	*tf.idf*	*tf.*χ^2	*eccd*	*tf.gss*	*tf.ig*	*midf*	*tf.oddsR*	*rw*	*tf*	*bin*
Reuters-10														
SVM(LIN)	.880	.886	**.892**	.889	.878	.692	.732	.740	.679	.841	.863	.858	.880	.863
SVM(RBF)	.888	.889	**.902**	.899	.883	.816	.755	.835	.801	.846	.876	.872	.883	.874
RandomForest	.843	.850	.853	.849	.847	.855	.825	.847	.846	.843	.849	.838	**.856**	.839
Reuters-52														
SVM(LIN)	.581	**.596**	.583	.593	.574	.316	.293	.320	.282	.244	.520	.348	.465	.348
SVM(RBF)	.611	.623	.595	**.643**	.613	.411	.335	.367	.341	.271	.550	.341	.471	.341
RandomForest	.291	**.363**	.355	.310	.308	.337	.283	.302	.341	.311	.329	.327	.361	.335
20 Newsgroups														
SVM(LIN)	.739	**.744**	**.744**	.733	.724	.530	.405	.467	.476	.578	.650	.681	.689	.681
SVM(RBF)	.691	**.692**	.688	**.692**	.682	.552	.453	.517	.521	.582	.650	.650	.654	.651
RandomForest	.496	**.537**	.532	.497	.504	.535	.470	.536	.531	.498	.527	.521	.532	.519

5 EXPERIMENTAL RESULTS

We tested the effectiveness of classification varying the term weighting scheme on several datasets. For each dataset we tested the classification varying the number of features p selected for each category. For ease of reporting, in tables 5 and 6 we show the best result for each dataset obtained by both SVM^{Light}, in linear and radial basis function and RandomForest classifiers using each term weighting scheme.

Tables 5 and 6 show the performance of 14 different term weighting methods: *tf.idfec*, *tf.idfec-based*, *tf.rf*, *tf.icf-based*, *tf.idf*, *tf.*χ^2, *eccd*, *tf.gss*, *tf.ig*, *midf*, *tf.oddsR*, *rw*, *tf* and *bin*, in terms of *micro-*F_1 and *macro-*F_1 on the three datasets previously described. In general, our second proposed scheme *tf.idfec-based* achieved top results in all datasets and with each classifier.

In particular, on *Reuters-52*, our *tf.idfec-based* outperforms every other scheme with all classifiers in terms of *micro-*F_1: using a SVM with linear kernel, compared with the standard *tf.idf* we have an improvement of 0.8% of *micro-*F_1 and 2.2% of *macro-*F_1. Compared with standard supervised schemes such as *tf.ig* and *tf.*χ^2 we have an improvement respectively of 10.3% and 9.7% of *micro-*F_1 and 31.4% and 28% of *macro-*F_1. Furthermore, *tf.idfec-based* outperforms albeit slightly the *tf.rf* and also the *tf.icf-based* in terms of *micro-*F_1. It is possible to note a marked difference between the *micro* and *macro-*F_1 values, this is due to the strong unbalancing of the classes in this dataset; the *macro-*F_1 is strongly negatively biased by classes with few documents, for which the classification has low effectiveness, thus in this dataset the *micro-*F_1 is much more meaningful.

On *Reuters-10* we note that the results obtained with our proposed methods are very close to *tf.rf*, *tf.icf-based* and also to the standard *tf.idf* and *tf*

Table 7: McNemar's significance test results. Each column is related to a dataset and a supervised classifier, the term weighting schemes are shown in decreasing order of effectiveness, separating groups of schemes by significance of differences in their perfomances. Results for RandomForest on *Reuters-10* are omitted as no significant difference is observed between them.

Reuters-10		*Reuters-52*			*20 Newsgroups*		
SVM(LIN)	SVM(RBF)	SVM(LIN)	SVM(RBF)	RandomFor.	SVM(LIN)	SVM(RBF)	RandomFor.
tf.idfec-b.	***tf.idfec-b.***	***tf.idfec-b.***	***tf.idfec-b.***	***tf.idfec-b.***	***tf.idfec-b.***	***tf.idfec-b.***	***tf.idfec-b.***
tf.rf	*tf.rf*	*tf.rf*	*tf.rf*	*tf.rf*	*tf.rf*	*tf.icf-b.*	*tf.gss*
tf	*tf.icf-b.*	***tf.idfec***	***tf.idfec***	*tf*	***tf.idfec***	***tf.idfec***	*tf.χ^2*
tf.icf-b.	*tf*	*tf.idf*	*tf.icf-b.*	*tf.ig*	*tf.icf-based*	*tf.rf*	*tf*
tf.idf	*tf.idf*	*tf.icf-b.*	*tf.idf*	*tf.oddsR*	*tf.idf*	*tf.idf*	*tf.ig*
tf.idfec	***tf.idfec***	*tf*	*tf*	*tf.χ^2*	*tf*	*tf*	*tf.rf*
midf	*tf.oddsR*	*tf.oddsR*	*tf.oddsR*	*tf.gss*	*bin*	*bin*	*tf.oddsR*
tf.oddsR	*bin*	*bin*	*bin*	*bin*	*rw*	*rw*	*rw*
bin	*midf*	*rw*	*rw*	*tf.idf*	*tf.oddsR*	*tf.oddsR*	*bin*
rw	*rw*	*midf*	*midf*	*midf*	*midf*	*midf*	*tf.idf*
tf.gss	*tf.gss*	*tf.gss*	*tf.gss*	*rw*	*tf.χ^2*	*tf.χ^2*	*tf.icf-b.*
tf.ig	*tf.ig*	*tf.ig*	*tf.χ^2*	***tf.idfec***	*tf.ig*	*tf.ig*	***tf.idfec***
tf.χ^2	*tf.χ^2*)	*tf.χ^2*	*tf.ig*	*tf.icf-b.*	*tf.gss*	*tf.gss*	*midf*
eccd	*eccd*	*eccd*	*eccd*	*eccd*	*eccd*	*eccd*	*eccd*

schemes. These results show how in this dataset, consisting of only 10 categories, using a supervised term weighting method is not relevant to the effectiveness of classification: this can be deduced from the difference between the effectiveness of standard *tf.idf* and our supervised versions on *Reuters-52*, which contains the same documents of *Reuters-10* but labeled with more categories. However, our schemes outperform standard supervised term weighting by more than 10%.

The results on *20 Newsgroups* show that *tf.idfec-based* obtains the best *micro-*F_1, in parity with *tf.rf* using linear SVM, with *tf.icf-based* using radial kernel SVM and with *tf.gss* using RandomForest. Using linear SVM, the best *micro-*F_1 of *tf.idfec-based* is higher by 1.8% compared to that obtained from *tf.idf* and of 23.9% compared with a standard supervised method like *tf.ig*.

Observing all the results, we can see that our first proposed scheme *tf.idfec* obtains results always in line but slightly lower than the *tf.idfec-based* variant. This evidently means that considering only the information about the negative categories of the terms is not enough to achieve an optimal accuracy. Conversely, adding information about the ratio between A and C (from notation of Table 2), it is obtained an optimal mixture that leads to better classification results, using either SVM or RandomForest classifiers.

We employ the McNemar's significance test to verify the statistical difference between performances of the term weighting schemes. We report these results in Table 7, where each column is related to a dataset and a classifier and the term weighting schemes are shown in decreasing order of effectiveness, separated according to the significance of difference between them. Schemes not separated with lines do not give significantly different results, a single line denotes that the schemes above perform better than the schemes below with a significance level between 0.01 and 0.001 (commonly indicated as "$A > B$"), while a double line denotes a significance level better than 0.001 ("$A >> B$"). To save space, we do not report the results on the *Reuters-10* corpus with RandomForest, because there are no significant statistical differences between them. From Table 7 we can note that our proposed *tf.idfec-based* scheme always provides top effectiveness. In addition, this table shows that with SVM classifiers, either linear or RBF kernel, some term weighting methods are more efficient than others. The best methods in general seem to be the latest supervised methods, such as *tf.idfec-based*, *tf.rf*, *tf.icf-based* and *tf.idfec*. Instead, using RandomForest, the classic supervised methods seem to work better, with results comparable or slightly below with respect to *tf.idfec-based*.

Let's now observe how classification effectiveness varies by changing the number of features considered to create the dictionary of the VSM. For reasons of readability of the graph, we do not show all the 14 term weighting methods investigated, but only the five most recent and with better results, i.e. *tf.idfec*, *tf.idfec-based*, *tf.rf*, *tf.icf-based*, *tf.idf*. For each dataset, we show the results obtained using an SVM classifier with a linear kernel.

Figures 1 and 2 show that on both views of Reuters corpus, when using *tf.idfec-based* we obtain the best

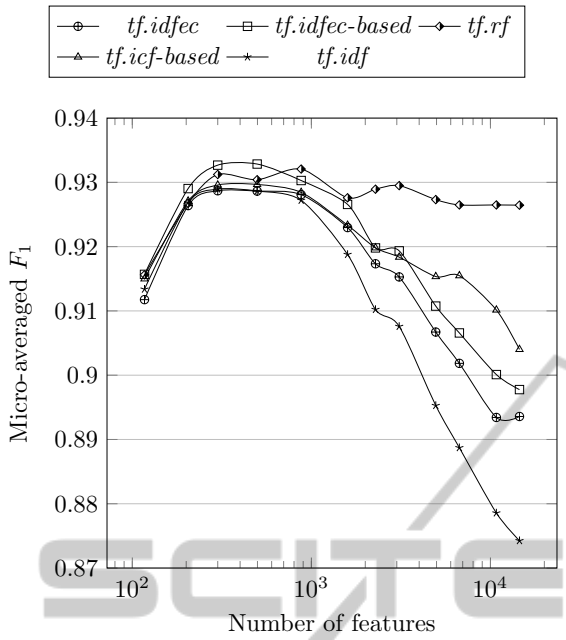

Figure 1: Results obtained on *Reuters-10* varying the number of top *p* features selected per category using linear SVM classifier. The X axis (in logarithmic scale) indicates the resulting total number of features.

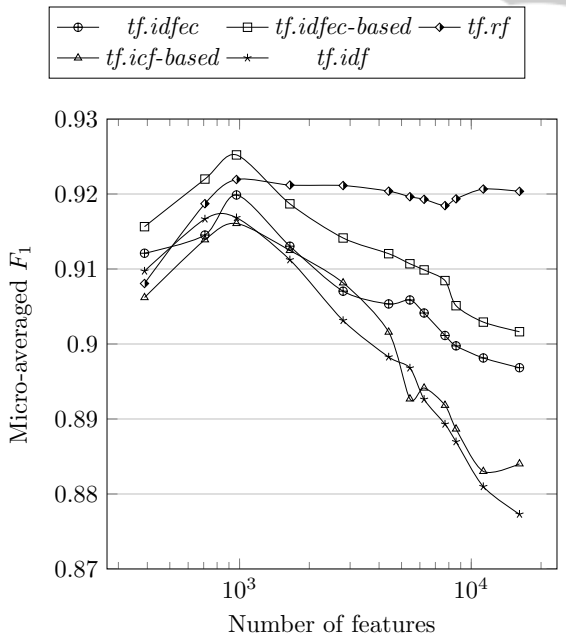

Figure 2: Results obtained on *Reuters-52* varying the number of top *p* features selected per category using linear SVM classifier. The X axis (in logarithmic scale) indicates the resulting total number of features.

results by using few features per category, considering the variations of *p* described in Section 4.2. We note that the best results on *Reuters-10* are obtained with 150 features per category and on *Reuters-52* with 75

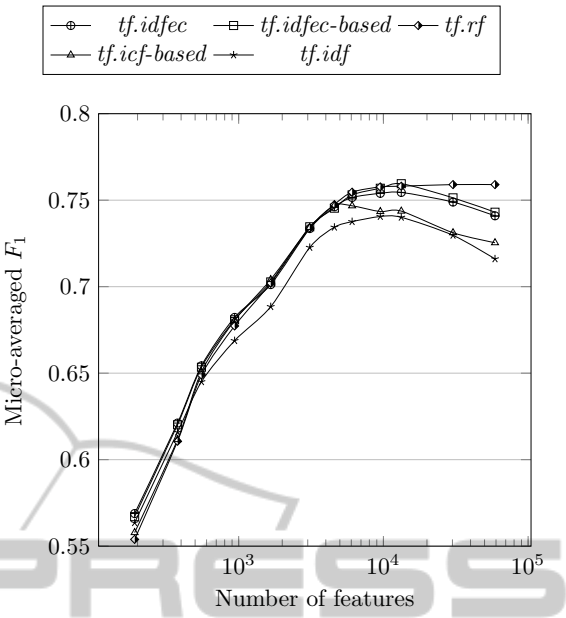

Figure 3: Results obtained on *20 Newsgroups* varying the number of top *p* features selected per category using linear SVM classifier. The X axis (in logarithmic scale) indicates the resulting total number of features.

features, corresponding respectively to overall dictionary sizes of 498 and 970. From these plots, however, we can see as the effectiveness of the classification using *tf.idfec-based* deteriorates by increasing the number of features considered, therefore introducing terms less frequent and discriminative. Analysing the behavior of the schemes from which *idfec-based* takes its cue, i.e. standard *tf.idf* and *tf.rf*, we note that this performance degradation is probably due to the *idf* factor of the weight, as even the *tf.idf* has the same type of trend results, while *tf.rf* seems to remain stable at values comparable to the best results also by increasing the dictionary size.

Figure 3 shows how to obtain the best results with the *20 Newsgroups* corpus is necessary a greater number of features. Using *tf.idfec-based* we obtain the best result with a dictionary of about 10000 features; after that the efficiency shows a slight decrease, but more moderate than that shown in the Reuters dataset.

6 CONCLUSIONS

We proposed two novel supervised variations of the *tf.idf* term weighting approach. In the first one, we employ the basic idea of *tf.idf*, but considering only documents not belonging to the modeled category: this prevents having a low *idf* factor if a term is largely present in the documents of the category un-

der consideration. The second variant uses a mixture of the previous idea and the relevance frequency *rf* in order to consider also the amount of documents belonging to the category in which the term occurs.

We performed extensive experimental studies on two datasets, i.e. *Reuters* corpus with either 10 or 52 categories and *20 Newsgroups*, and three different classification methods, i.e. SVM classifier with linear and RBF kernel functions and RandomForest. The obtained results show that the *tf.idfec-based* method that combines *idfec* and *rf* generally gets top results on all datasets and with all classifiers. Through statistical significance tests, we showed that the proposed scheme always achieves top effectiveness and is never worse than other methods. The results put in evidence a close competition between our *tf.idfec-based* and the *tf.rf* schemes; in particular, the best results obtained with the different datasets and algorithms, varying the amount of feature selection, are very similar, but with some differences: *tf-rf* seems to be more stable when the number of features is high, while our *tf.idfec-based* gives excellent results with few features and shows some decay (less than 4%) when the number of features increases.

As future works, we plan to apply this idea to larger datasets and to hierarchical text corpora, using a variation of *idfec* able to take into account the taxonomy of categories. We also plan to test the use of this scheme for feature selection, other than for term weighting. Moreover, we are going to investigate the effectiveness of this variant of *tf.idf* in other fields where weighting schemes can be employed with possible efficacy improvements, such as sentiment analysis and opinion mining.

REFERENCES

Bloehdorn, S. and Hotho, A. (2006). Boosting for text classification with semantic features. In Mobasher, B., Nasraoui, O., Liu, B., and Masand, B., editors, *Advances in Web Mining and Web Usage Analysis*, volume 3932 of *Lecture Notes in Computer Science*, pages 149–166. Springer Berlin Heidelberg.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Carmel, D., Mejer, A., Pinter, Y., and Szpektor, I. (2014). Improving term weighting for community question answering search using syntactic analysis. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '14, pages 351–360, New York, NY, USA. ACM.

Debole, F. and Sebastiani, F. (2003). Supervised term weighting for automated text categorization. In *In Proceedings of SAC-03, 18th ACM Symposium on Applied Computing*, pages 784–788. ACM Press.

Deisy, C., Gowri, M., Baskar, S., Kalaiarasi, S., and Ramraj, N. (2010). A novel term weighting scheme midf for text categorization. *Journal of Engineering Science and Technology*, 5(1):94–107.

Deng, Z.-H., Luo, K.-H., and Yu, H.-L. (2014). A study of supervised term weighting scheme for sentiment analysis. *Expert Systems with Applications*, 41(7):3506–3513.

Deng, Z.-H., Tang, S.-W., Yang, D.-Q., Li, M. Z. L.-Y., and Xie, K.-Q. (2004). A comparative study on feature weight in text categorization. In *Advanced Web Technologies and Applications*, pages 588–597. Springer.

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Comput.*, 10(7):1895–1923.

Domeniconi, G., Moro, G., Pasolini, R., and Sartori, C. (2014). Cross-domain text classification through iterative refining of target categories representations. In *Proceedings of the 6th International Conference on Knowledge Discovery and Information Retrieval*.

Galavotti, L., Sebastiani, F., and Simi, M. (2000). Experiments on the use of feature selection and negative evidence in automated text categorization. In *Research and Advanced Technology for Digital Libraries*, pages 59–68. Springer.

Hassan, S. and Banea, C. (2006). Random-walk term weighting for improved text classification. In *In Proceedings of TextGraphs: 2nd Workshop on Graph Based Methods for Natural Language Processing. ACL*, pages 53–60.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features.

Lan, M., Sung, S.-Y., Low, H.-B., and Tan, C.-L. (2005). A comparative study on term weighting schemes for text categorization. In *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, volume 1, pages 546–551. IEEE.

Lan, M., Tan, C. L., Su, J., and Lu, Y. (2009). Supervised and traditional term weighting methods for automatic text categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(4):721–735.

Largeron, C., Moulin, C., and Géry, M. (2011). Entropy based feature selection for text categorization. In *Proceedings of the 2011 ACM Symposium on Applied Computing*, SAC '11, pages 924–928, New York, NY, USA. ACM.

Leopold, E. and Kindermann, J. (2002). Text categorization with support vector machines. how to represent texts in input space? *Mach. Learn.*, 46(1-3):423–444.

Lewis, D. D. (1995). Evaluating and optimizing autonomous text classification systems. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '95, pages 246–254, New York, NY, USA. ACM.

Liu, Y., Loh, H. T., and Sun, A. (2009). Imbalanced text classification: A term weighting approach. *Expert Syst. Appl.*, 36(1):690–701.

Luo, Q., Chen, E., and Xiong, H. (2011). A semantic term weighting scheme for text categorization. *Expert Syst. Appl.*, 38(10):12708–12716.

Paltoglou, G. and Thelwall, M. (2010). A study of information retrieval weighting schemes for sentiment analysis. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1386–1395, Stroudsburg, PA, USA. Association for Computational Linguistics.

Papineni, K. (2001). Why inverse document frequency? In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, pages 1–8. Association for Computational Linguistics.

Ren, F. and Sohrab, M. G. (2013). Class-indexing-based term weighting for automatic text classification. *Inf. Sci.*, 236:109–125.

Robertson, S. (2004). Understanding inverse document frequency: on theoretical arguments for idf. *Journal of documentation*, 60(5):503–520.

Ropero, J., Gómez, A., Carrasco, A., and León, C. (2012). A fuzzy logic intelligent agent for information extraction: Introducing a new fuzzy logic-based term weighting scheme. *Expert Systems with Applications*, 39(4):4567–4581.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47.

Song, S.-K. and Myaeng, S. H. (2012). A novel term weighting scheme based on discrimination power obtained from past retrieval results. *Inf. Process. Manage.*, 48(5):919–930.

Sparck Jones, K. (1988). Document retrieval systems. chapter A statistical interpretation of term specificity and its application in retrieval, pages 132–142. Taylor Graham Publishing, London, UK, UK.

Tokunaga, T. and Makoto, I. (1994). Text categorization based on weighted inverse document frequency. In *Special Interest Groups and Information Process Society of Japan (SIG-IPSJ*. Citeseer.

Tsai, F. S. and Kwee, A. T. (2011). Experiments in term weighting for novelty mining. *Expert Systems with Applications*, 38(11):14094–14101.

Wang, D. and Zhang, H. (2013). Inverse-category-frequency based supervised term weighting schemes for text categorization. *Journal of Information Science and Engineering*, 29(2):209–225.