Appl Intell (2016) 45:30–43
DOI 10.1007/s10489-015-0745-z

# Model-induced term-weighting schemes for text classification

Hyun Kyung Kim[1] · Minyoung Kim[1]

Published online: 15 January 2016

**Abstract** The bag-of-words representation of text data is very popular for document classification. In the recent literature, it has been shown that properly weighting the term feature vector can improve the classification performance significantly beyond the original term-frequency based features. In this paper we demystify the success of the recent term-weighting strategies as well as provide possibly more reasonable modifications. We then propose novel term-weighting schemes that can be induced from the well-known document probabilistic models such as the Naive Bayes and the multinomial term model. Interestingly, some of the intuition-based term-weighting schemes coincide exactly with the proposed derivations. Our term-weighting schemes are tested on large-scale text classification problems/datasets where we demonstrate improved prediction performance over existing approaches.

**Keywords** Document/text classification · Feature/term weighting · Feature selection · Supervised learning

## 1 Introduction

Text classification is the task of automatically categorizing unlabeled document or text into one of the pre-defined classes. It has been studied considerably in the natural language processing, machine learning, and related fields. In particular, the applications of text classification include among others: automatically learning user's reading preferences [11, 22], classifying news articles [4, 17, 24], web pages [6, 31, 36], and filtering e-mails [23, 25, 29, 34].

In text classification, it is often very typical and advantageous to have term feature vector representation of a document so as to feed into a classifier for recognition. Broadly there are two types of feature vector representations that are widely used. Popular in Boolean models, the term presence (*tp*) vector is defined to be of size $V$ (i.e., the size of the vocabulary or the number of distinct words considered) where each entry has 1 if the corresponding term appears in the document, and 0 otherwise. Another very popular form, typically used in Bag-of-Words (BoW) models, is the term-frequency (*tf*) representation which maintains the number of occurrences of each term that appears in a document. In this paper, we consider and use both *tp* and *tf* term feature vector representations.

Instead of directly using the feature vectors in learning a classifier, it is known to achieve better prediction often times to weigh the term features according to their relative importance. The most popular term weighting scheme is the so-called *tf-idf* [33], where each term frequency (*tf*) is weighted by the *idf* which is a reciprocal of the document-frequency (*df*) defined as the number of training documents that contain the term. Intuitively, *idf* values are high for those terms that occur exclusively in some documents (most likely, keywords), hence considered to be important. On the other hand, the ubiquitous terms like stop words (e.g., articles) are weighted by near-0 *idf* values, thus effectively ignored in the classification.

There have been considerable research attempts to devise term-weighting schemes that can yield improved prediction

✉ Minyoung Kim
mikim21@gmail.com

Hyun Kyung Kim
khk930608@gmail.com

1 Department of Electronics & IT Media Engineering, Seoul National University of Science & Technology, Seoul 139-743, Korea

performance than *tf-idf*. The approaches can be categorized broadly into unsupervised and supervised schemes depending on whether the class labels in the training data are exploited or not in deriving the weight vector. For instance, *tf-idf* is clearly an unsupervised approach. It is also evident that except for the situations where the training data are unlabeled or weakly labeled, the supervised approaches can potentially outperform unsupervised ones with better discriminative power.

The existing supervised feature weighting strategies typically resort to certain information-theoretic statistics such as $\chi^2$ metric, information gain, and odds ratio, to name a few [7, 9]. Among others, the recent *tf-rf* strategy [20, 21] aims to weigh each term frequency by the so-called *relevance frequency* (*rf*) defined to be the ratio between the numbers of positive and negative training documents that contain the term. Intuitively, terms with high *rf* values can be considered as key terms to decide the positivity of the document, thus the *rf* being a good indicator for classification.

More recently, there have been similar attempts, but with ideas closely related to the *tf-rf*, which include: 1) [16] where the term importance is defined by the imbalance between frequencies of the term in positive and negative documents, 2) [18] which focuses on transferring the vector representation of documents to test samples under multi-class supervised setting by proposing several heuristics to combine class-wise models, 3) in [1, 15] it is aimed to develop ensemble approaches that combine different term weighting schemes, which are applied to sentiment analysis for regular documents and social media data, and 4) [32] focuses on statistical estimation of the importance of a word by the ratio of its confidence interval estimates of the proportions of documents containing the word.

Although the *tf-rf* weighting scheme has achieved superior document prediction performance to existing supervised/unsupervised approaches, it has several critical drawbacks. First, the specific monotonic link, namely the $\log_2(2+\text{ratio})$, adopted in [20, 21] to restrict the weight values no less than 1, is indeed unnecessary (since most classifiers like support vector machines can naturally deal with fractional and negative features), and even worse, can attenuate the importance weight given by the ratio. Secondly, as the *rf* is evaluated as a count, not relative frequency, this method basically assumes that the numbers of positive and negative documents are roughly equal (i.e., classes well balanced). Such assumption can be easily violated, for instance, when the multi-class problem is converted to multiple binary problems. These weak points of the *tf-rf* scheme can potentially result in significant performance degradation in certain scenarios.

In this paper we propose several intuitively appealing modifications or corrections of the *tf-rf* scheme. One of the corrections is the replacement of the relevance counts by the normalized frequencies. Also, we take into account the *absence* of terms in positive/negative documents (not just presence of terms as in the *tf-rf* scheme). We empirically show that these modifications indeed yield considerable improvement in prediction performance over the *tf-rf*.

Another, in fact our major contribution is that we propose novel term-weighting algorithms derived from the popular probabilistic models. The first model we consider is the binary Naive Bayes model that assumes independent class-conditional binary term distributions. Unlike standard classification approach by thresholding the log-odds-ratio (LOR) from the maximum likelihood model estimate, we provide a new interpretation of the LOR as a feature-weight product form, which leads to a novel term-weighting scheme.

In addition, the popular multinomial term model for document representation is similarly investigated, where we derive another term weighting scheme that is based on the log probability ratio. Interestingly, it turns out that some of the intuition-based term-weighting schemes coincide exactly with the proposed model-based term weighting derivations. For several popular text classification benchmark datasets, we demonstrate that the proposed approaches consistently yield higher prediction accuracies than existing supervised/unsupervised term-weighting methods.

The proposed term-weighting schemes are promising in that they utilize the features derived from the sensible probabilistic document models. In addition, by replacing the uniform weighting schemes in the conventional classification based on the odds ratio with the learned weights using the large margin principle, the models can yield higher accuracy in generalization performance.

The paper is organized as follows: In Section 2, we formalize the problem with brief descriptions on the vector representations of documents and the framework of the term weighting in conjunction with classification. In Section 3, we review the existing term-weighting schemes highlighting the *tf-rf* algorithm [20, 21], for which we suggest several modifications based on the intuition. We then propose our new term weighting schemes induced from binary Naive Bayes and multinomial term models together with their variants in Section 4, followed by the experimental results in Section 5.

## 2 Problem setup and notations

We consider the text/document classification problem within the supervised learning setup where we have $n$ training samples of document-class pairs, $\mathcal{D} = \{(d_i, y_i)\}_{i=1}^n$. Throughout the paper we deal with binary classification (i.e., $y \in \{+1, -1\}$) where extension to multi-class problems is straightforward via standard treatments (e.g., one in [5]).

The main goal is to accurately predict the class label $y$ for an unseen document $d$. We assume there is an underlying vocabulary set of terms (or words), $\mathcal{V} = \{t_k\}_{k=1}^{V}$, where there are $V$ terms in total. For instance, $t_{100} = boy$, $t_{203} = car$, and so on. Typically the vocabulary set is formed from the training corpus.

## 2.1 Vector representation of document

A document $d$ is comprised of terms from $\mathcal{V}$, presented in the form of sentences/paragraphs. In machine learning and pattern recognition, it is typical to have a vector representation of $d$, rather than the raw enumerated sentential form. The most popular vector representation is the so-called *term-frequency* (*tf*) vector that counts the frequency of each term that occurs in $d$. It is denoted by $\mathbf{tf}(d) = [tf_1, tf_2, \ldots, tf_V]^\top$, a $V$-dim vector where $tf_k$ = the number of times $t_k$ occurs in $d$. Hence in the $tf$-vector representation, the actual/relative positions of terms in a document do not matter, yet only their frequencies do. In this sense it is often called bag-of-words (BoW) representation.

Another possible representation is to merely count the *presence* of each term, which we denote by the *term-presence* (*tp*) vector. That is, $\mathbf{tp}(d) = [e_1, e_2, \ldots, e_V]^\top$ is a $V$-dim 0/1 vector where $e_k = 1$ if $t_k$ occurs in $d$, and $e_k = 0$ otherwise. Thus, in $tp$-representation, it is only concerned whether a term appears or not, and how many times it occurs does not matter at all provided that it ever occurs. The $tp$-vector is relatively less popular than $tf$ representation, however, our proposed feature weighting strategies are based on the $tp$ representation, which often achieves superior performance to competing methods.

## 2.2 Term-weighted features and classification

We denote the above-mentioned (nominal) vector representation of $d$ by $\boldsymbol{\phi}(d)$ ($\in \mathbb{R}^V$), that is, either $\boldsymbol{\phi}(d) = \mathbf{tf}(d)$ or $\mathbf{tp}(d)$. In document classification, one may use this nominal vector $\boldsymbol{\phi}(d)$ as a feature vector for $d$, but it is more common to induce a more sensible feature vector from it. For instance, the famous *tf-idf* [33] is defined as:

$$\mathbf{x}(d) = \mathbf{tf}(d) \otimes \mathbf{idf}, \quad (\otimes : \text{ element-wise product}), \tag{1}$$

where $\mathbf{idf} = [idf_1, idf_2, \ldots, idf_V] \in \mathbb{R}^V$ is the *inverse-document-frequency* vector such that $idf_k = \log\left(\frac{n}{df_k}\right)$ where $df_k$ = the number of training documents that contain $t_k$. This essentially reflects the importance of each term into the feature vector $\mathbf{x}(d)$ via term weighting: relatively unique terms (low $df_k$) are highly weighted, while ubiquitous terms (high $df_k$) such as stop words or articles are effectively ignored, which is intuitively quite appealing.

Using the induced feature vector $\mathbf{x}(d)$, one can apply it to any classification algorithm. Among others, in this paper, we mostly deal with the sophisticated linear support vector machine (SVM), $y = \text{sgn}(\mathbf{w}^\top \mathbf{x}(d) + w_0)$, where the model parameters ($\mathbf{w} \in \mathbb{R}^V$, $w_0 \in \mathbb{R}$) are learned from the hinge-loss minimizing SVM optimization [30].

The idea of *tf-idf* can be generally framed as a feature weighting strategy. We basically induce a feature vector $\mathbf{x}(d)$ from the nominal $\boldsymbol{\phi}(d)$ by:

$$\mathbf{x}(d) = \boldsymbol{\phi}(d) \otimes \mathbf{r}, \tag{2}$$

where $\mathbf{r} \in \mathbb{R}^V$ is the feature weighting vector. It is worth noting that $\mathbf{r}$ is not dependent on the document of interest $d$, rather evaluated from the training data, and remains fixed for any document $d$. In *tf-idf*, for instance, $\mathbf{r} = \mathbf{idf}$ is not contingent on $d$, but evidence or information extracted from the training data. There are several variants of *tf-idf* feature weighting, and all of them conform to the form of (2).

In this feature weighting framework, one possible way to find a good $\mathbf{r}$ is by directly formulating an optimization problem. Within the SVM formulation, one can formulate a simultaneous optimization problem:

$$\min_{\{\mathbf{w}, w_0\}, \{\xi_i\}, \mathbf{r}} \quad \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n} \xi_i \tag{3}$$

$$\text{s.t.} \quad y_i\left(\mathbf{w}^\top \mathbf{x}(d_i) + w_0\right) \geq 1 - \xi_i, \quad \xi_i \geq 0, \tag{4}$$

$$\mathbf{x}(d_i) = \boldsymbol{\phi}(d_i) \otimes \mathbf{r}, \quad i = 1, \ldots, n. \tag{5}$$

Although this simultaneous learning of the feature weights $\mathbf{r}$ and the SVM parameters $(\mathbf{w}, w_0)$ appears to be principled, it entails several issues/drawbacks. First the number of parameters to estimate is roughly doubled,[1] which can increase the chance of overfitting. Secondly, unlike the original SVM formulation that is convex (quadratic programming), the optimization problem becomes non-convex, and the consequence is that solving the optimal solution is difficult, and the attained solutions can be merely local optima. Perhaps a more serious issue is that in this framework one tries to learn the feature weights $\mathbf{r}$ in a *blind* fashion, meaning that any prior or domain knowledge (e.g., the intuition of *idf*) cannot be properly exploited, nor any useful structure (e.g., term sparsity patterns) that may possibly reside in the data.

[1]While there are slack variables to be optimized as well, they often tend to be highly sparse with a few non-zero entries for the so-called support vectors.

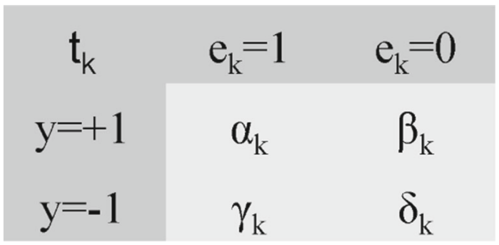
## 3 Prior work: supervised term weighting

The drawbacks mentioned above reinforce term weighting schemes motivated from intuition just as the *tf-idf* weighting. In parallel, the *supervised* feature weighting strategies have recently emerged. Unlike *tf-idf* which is unsupervised, the supervised approaches utilize the class labels in the training set to estimate the weight vector $\mathbf{r}$. This can endow feature weighting hopefully with better discriminative power.

The existing supervised feature weighting strategies typically resort to certain information-theoretic statistics such as $\chi^2$ metric, information gain, and odds ratio, to name a few [7, 9]. Despite their reasonable exploitation of class labels, these methods were unable to exhibit consistently improved performance over the traditional *tf-idf* strategy.

More recently, the *tf-rf* weighting algorithm proposed in [20, 21] is interesting, and has shown superior prediction performance to the other related supervised methods on some benchmark datasets. The approach constructs the feature vector by weighting the nominal *tf* vector by the so-called *rf* weight vector (*rf* stands for relevance frequency). The underlying idea of *tf-rf* is as follows. For each term $t_k$, one computes the contingency table from the training data that may look like Fig. 1. In this table, $\alpha_k$ = the number of positive documents that contain $t_k$, $\beta_k$ = the number of positive documents that do not have $t_k$, and $\gamma_k$ and $\delta_k$ defined similarly for negative documents. Obviously, $\alpha_k + \gamma_k = df_k$, $\alpha_k + \beta_k = n_+$, $\gamma_k + \delta_k = n_-$ ($n_+, n_-$: the numbers of positive and negative documents, respectively), and $\alpha_k + \beta_k + \gamma_k + \delta_k = n$, etc.

The key observation in *tf-rf* is that assuming $n_+ \approx n_-$, having $\alpha_k \approx \gamma_k$ implies that $t_k$ appears in positive documents as frequently as negative samples. Consequently, $t_k$ does not convey any useful discriminative information for the purpose of classification, hence its importance has to be diminished. On the other hand, if $\alpha_k \gg \gamma_k$, it is a crucial clue that presence of $t_k$ decisively leads to positive labeling of the document, and in turn $t_k$ has to be assigned a higher weight.



| $t_k$ | $e_k=1$ | $e_k=0$ |
|---|---|---|
| $y=+1$ | $\alpha_k$ | $\beta_k$ |
| $y=-1$ | $\gamma_k$ | $\delta_k$ |

**Fig. 1** Contingency table for each term $t_k$. The table contains the number of training documents that meet the conditions given by row and column: $e_k$ indicates whether or not $t_k$ appears in the document, and $y$ is the class label of the document. For instance, $\alpha_k$ is the number of positive training documents that contain $t_k$, while $\delta_k$ is the number of negative samples that do not have $t_k$

From the intuition, in [20, 21] the weight for $t_k$ is defined as:

$$r_k = \log_2\left(2 + \frac{\alpha_k}{\gamma_k}\right). \tag{6}$$

This certainly reflects the above-mentioned idea, while they added 2 in the log to make the weight no less than one.[2] Although intuitively appealing, there are several drawbacks and possible modifications that can be made to the *tf-rf* scheme as summarized below:

1. $\alpha_k \ll \gamma_k$ is another decisive clue that $t_k$ tends to appear exclusively in negative samples, hence a good feature for class discrimination. In the *tf-rf* weighting in (6), this cannot be properly reflected mainly due to the addition of 2 in the log (i.e., having $\alpha_k \ll \gamma_k$ makes $r_k \approx 1$, thus just taking the nominal feature). One quick remedy is to remove 2 in (6), which would assign a high-magnitude (either positive or negative) weight to $t_k$ so long as $\alpha_k$ is far away from $\gamma_k$.
2. To make the idea of *tf-rf* sensible, it has been assumed that $n_+ \approx n_-$. If this assumption is violated, that is, if data are highly class imbalanced, the weighting scheme of (6) based on the ratio between $\alpha_k$ and $\gamma_k$ is less meaningful. Often times we have $n_+ \ll n_-$ particularly when the multi-class problem is converted to a series of binary classification problems, e.g., via one-vs-others fashion. To address this issue, it is more appropriate to use the ratio between normalized frequencies, namely
$$r_k = \log\left(\frac{\alpha_k/n_+}{\gamma_k/n_-}\right). \tag{7}$$
3. Whereas *tf-rf* focuses only on the presence of terms (i.e., using $\alpha_k$ and $\gamma_k$ only), one can also exploit the absence of terms. For instance, $\beta_k \ll \delta_k$ exhibits strong evidence of deductive relationship: absence of $t_k$ implies negative labeling. To incorporate this property, one possible way is to reinforce (multiply) the ratio by $\frac{\delta_k}{\beta_k}$ which is along the direction toward positive labeling in accordance with $\frac{\alpha_k}{\gamma_k}$. Considering the normalized

[2]There is no reason to constrain the feature vector to be bounded from below by 1 in the SVM framework. Recall that features with large magnitudes are considered important, and not their signs, since one can simply adapts the signs of SVM weight parameters accordingly. Constraining the weight bound in [20, 21] is solely from their physical interpretation of the weights as *adjusted term frequencies*.

frequencies mentioned previously, the refined weight vector can be expressed as:

$$r_k = \log\left(\frac{\alpha_k/n_+}{\gamma_k/n_-} \cdot \frac{\delta_k/n_-}{\beta_k/n_+}\right) = \log\left(\frac{\alpha_k\delta_k}{\gamma_k\beta_k}\right). \quad (8)$$

4. In the *tf-rf* scheme, the weight vector $\mathbf{r}$ is multiplied by the *tf* vector (i.e., $\boldsymbol{\phi}(d) = \mathbf{tf}(d)$). One natural question is why it should be. An alternative strategy is to employ the term-presence vector instead, $\boldsymbol{\phi}(d) = \mathbf{tp}(d)$, which can be beneficial for suppressing the effect of multiple term occurrences, possibly peripheral in certain situations.

Indeed, in the experiments (Section 5), we show that the above modifications significantly improve the class prediction accuracy over the original *tf-rf* often times. However, these schemes are inherently heuristic and less principled. In the next section we propose novel feature weighting schemes by inducing features/weights from the well-known document probabilistic models such as the Naive Bayes and the multinomial term model. The key idea is to decouple features and weights from the linear predictors based on the odds ratios obtained from the maximum likelihood (ML) estimates of the underlying probabilistic models. This is the main contribution of the paper. Interestingly, it turns out that some of the intuition-based term-weighting schemes coincide exactly with the proposed derivations.

We conclude this section by briefly listing some recent term weighting methods that are highly related to ours. In [35], the tf-rf scheme is extended by the log-link on the tf vector, which empirically exhibits improvement over the tf-rf scheme. In [13] the genetic programming approach is applied to the term weighting framed by the optimization problem of maximizing the classification performance. In [3] the term weighting problem is tackled by syntactic analysis of the corpus and applied to the community question answering domains. For the sentiment analysis, two basic factors of importance of terms in a document and whether terms express sentiment, are considered and exploited in [8]. In [19], the term weights are judged by the measure of statistical independence estimated using the contingency tables.

## 4 Our approaches: model-induced term weighting

We consider two well-known probabilistic models for document data: binary Naive Bayes and multinomial term model. These models are generative models $P(d, y)$ that represent the stochastic process of generating class-document pairs. We induce features and weights from the class predictors derived from the ML-learned models.

### 4.1 Binary naive Bayes model

The Naive Bayes (NB) model is perhaps the simplest model to represent document data in a BoW format. It assumes that given the class $y$ the term generation process is independent across the terms. Despite this strong assumption, it often yields superb prediction accuracy in text classification [27]. Among other possible formulations, in this paper we deal with the binary Naive Bayes model with the term-presence document representation, $\boldsymbol{\phi}(d) = \mathbf{tp}(d) = [e_1, e_2, \ldots, e_V]^\top$ where $e_k = 1$ if $t_k$ occurs in $d$, and $e_k = 0$ otherwise.

As shown in the graphical model representation in Fig. 2, the model is comprised of: i) the Bernoulli class prior distribution with parameter $\pi = P(y = +1)$, and ii) class-conditional term presence distributions which are also Bernoulli with parameters $p_k = P(e_k = 1|y = +1)$ and $\overline{p}_k = P(e_k = 1|y = -1)$ for $k = 1, \ldots, V$. It is straightforward to see that the class-conditional likelihoods of the document $d$ can be written as:

$$P(d|y=+1) = \prod_{k=1}^{V} p_k^{e_k}(1-p_k)^{1-e_k},$$
$$P(d|y=-1) = \prod_{k=1}^{V} \overline{p}_k^{e_k}(1-\overline{p}_k)^{1-e_k}. \quad (9)$$

Together with the class prior, the full data likelihood can be obtained, and it is well known that the maximum likelihood estimates given the training data $\mathcal{D} = \{(d_i, y_i)\}_{i=1}^n$ are:[3]

$$\pi^* = \frac{n_+}{n}, \quad p_k^* = \frac{\sum_{i\in\mathcal{I}_+} e_k^{(i)}}{n_+}, \quad \overline{p}_k^* = \frac{\sum_{i\in\mathcal{I}_-} e_k^{(i)}}{n_-}, \quad (10)$$

where $\mathcal{I}_+ = \{1 \le i \le n : y_i = +1\}$ is the index set of positive documents, thus $|\mathcal{I}_+| = n_+$ ($\mathcal{I}_-$ defined similarly), and $e_k^{(i)}$ is the $e_k$ statistic for $d_i$.

Once the model is learned, the test prediction of an unseen document $d$ with $\boldsymbol{\phi}(d) = [e_1, e_2, \ldots, e_V]^\top$ can be done by the so-called log-odds-ratio (LOR), namely

$$LOR = \log\frac{P(y=+1|d)}{P(y=-1|d)} = \log\frac{\pi}{1-\pi} + \sum_{k=1}^{V} e_k \cdot \log\frac{p_k}{\overline{p}_k}$$
$$+ \sum_{k=1}^{V}(1-e_k)\cdot\log\frac{1-p_k}{1-\overline{p}_k}. \quad (11)$$

[3]In (10), it is typical to add some positive constants to numerators and denominators for $p_k$ and $\overline{p}_k$ (e.g., 1 to numerators and 2 to denominators), often known as the Laplace smoothing. Throughout the paper we skip the Laplace smoothers for simplicity, and incorporating them does not alter the core derivations.

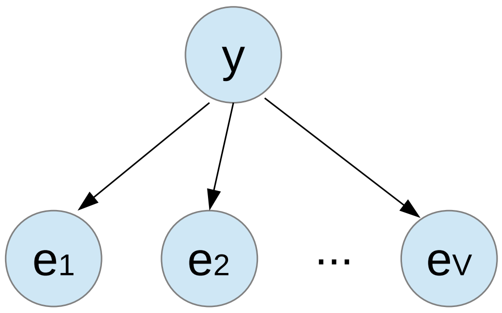


**Fig. 2** Graphical model representation of the binary Naive Bayes model

In the Bayes optimal sense [12], $LOR$ serves as the determinant for class decision: if $LOR > 0$ we predict $y = +1$, while $y = -1$ if $LOR < 0$.

Although prediction based on the log-odds-ratio is reasonable by itself, we try to view it as a *product of features and weights* within a linear classifier. Specifically, rearranging (11) yields:

$$LOR = \sum_{k=1}^{V} e_k \cdot \log \frac{p_k(1-\overline{p}_k)}{\overline{p}_k(1-p_k)} + \left( \log \frac{\pi}{1-\pi} + \sum_{k=1}^{V} \log \frac{1-p_k}{1-\overline{p}_k} \right). \tag{12}$$

If we define the feature vector for $d$ as $\mathbf{x}(d) = \mathbf{tp}(d) \otimes \mathbf{r}$ with the weight vector $\mathbf{r}$ specified as:

$$r_k = \log \frac{p_k(1-\overline{p}_k)}{\overline{p}_k(1-p_k)}, \tag{13}$$

then the first term in the right hand side of (12) is identical to $\mathbf{1}^\top \mathbf{x}(d)$ where $\mathbf{1}$ is the $V$-dim vector with all entries equal to 1.

Furthermore, the second term (in the big parenthesis) in (12) can be considered as a constant for its indenepdency on $d$. Denoting the constant as $w_0$, we have a linear classifier model that is equivalent to (12):

$$y = \text{sgn}(\mathbf{w}^\top \mathbf{x}(d) + w_0) \quad \text{where } \mathbf{w} = \mathbf{1} \text{ and}$$
$$w_0 = \log \frac{\pi}{1-\pi} + \sum_{k=1}^{V} \log \frac{1-p_k}{1-\overline{p}_k}. \tag{14}$$

It essentially means that using the feature weighting scheme of (13) with the nominal feature $\boldsymbol{\phi}(d) = \mathbf{tp}(d)$ in the linear classifier with the fixed specific parameters $(\mathbf{w}, w_0)$ as defined in (14), reduces exactly to the log-odds-ratio based Naive Bayes predictive model.

From this perspective, our main motivation is that one can rather learn the linear classifier, instead of using the fixed $(\mathbf{w}, w_0)$ in (14), to further boost the classification accuracy. That is, we learn the linear SVM classifier $(\mathbf{w}, w_0)$ using the prescribed term-weighted features prescribed by (13). We denote this feature weighting scheme by *tp-bnb*. Interestingly, the weight vector of *tp-bnb* (i.e., (13)) coincides exactly with the weight vector of (8), one that is refined from the *tf-rf* scheme. This can be easily verified by plugging the maximum likelihood estimates (10) in (13) and using the fact that $\alpha_k = \sum_{i \in \mathcal{I}_+} e_k^{(i)}$ and $\gamma_k = \sum_{i \in \mathcal{I}_-} e_k^{(i)}$.

Several variants of *tp-bnb* are immediately available by slight modifications. First, one can adopt the term-frequency vector as a nominal feature, i.e., $\boldsymbol{\phi}(d) = \mathbf{tf}(d)$ instead of $\mathbf{tp}(d)$. This scheme, which we call *tf-bnb*, is less principled than *tp-bnb*, but can be certainly beneficial for domains where the term frequency information is important. Alternatively, one may consider to drop out the third term in the right hand side of the LOR formula in (11), which can be regarded as pseudo-partial log-odds-ratio. The similar view of feature-weight decoupling in a linear classifier model results in $r_k = \log \frac{p_k}{\overline{p}_k}$, which becomes identical to the modified weighting strategy in (7). We refer to these feature weighting schemes as *tf-bnb-mod* and *tp-bnb-mod* contingent on the nominal feature employed.

### 4.2 Multinomial term model

We next consider another popular probabilistic document model that aims to account for the stochastic process of generating frequencies of terms comprising a document, instead of mere presence. The class-conditional observation is modeled by the multinomial distribution, in which the observable document representation is the *tf* vector, $\boldsymbol{\phi}(d) = \mathbf{tf}(d) = [tf_1, tf_2, \ldots, tf_V]^\top$. The model is decomposed into: i) the Bernoulli class prior $\pi = P(y = +1)$ similarly as Naive Bayes, and ii) class-conditional term frequency distributions which are multinomial with parameters $\boldsymbol{\mu} = [\mu_1, \mu_2, \ldots, \mu_V]^\top$ and $\overline{\boldsymbol{\mu}} = [\overline{\mu}_1, \overline{\mu}_2, \ldots, \overline{\mu}_V]^\top$ that lie on the probability simplex (i.e., $\mu_k, \overline{\mu}_k \geq 0$ and $\sum_k \mu_k = \sum_k \overline{\mu}_k = 1$).

The class-conditional likelihoods of the document $d$ can be written as:

$$P(d|y=+1) = \eta(d) \prod_{k=1}^{V} \mu_k^{tf_k}, \quad \text{where } \eta(d) = \frac{\left(\sum_{k=1}^{V} tf_k\right)!}{\prod_{k=1}^{V} (tf_k)!}. \tag{15}$$

The negative-class observation model $P(d|y = -1)$ is defined similarly as (15) with $\mu_k$ replaced by $\overline{\mu}_k$. Note that $\eta(d)$ depends solely on $d$, and is constant with respect to the model parameters.

Given the training data $\mathcal{D} = \{(d_i, y_i)\}_{i=1}^n$, the maximum likelihood estimates can be obtained by simple counting, specifically

$$\pi^* = \frac{n_+}{n}, \quad \mu_k^* = \frac{\sum_{i\in\mathcal{I}_+} tf_k^{(i)}}{\sum_{k=1}^V \sum_{i\in\mathcal{I}_+} tf_k^{(i)}},$$
$$\overline{\mu}_k^* = \frac{\sum_{i\in\mathcal{I}_-} e_k^{(i)}}{\sum_{k=1}^V \sum_{i\in\mathcal{I}_-} tf_k^{(i)}}, \tag{16}$$

where $tf_k^{(i)}$ is the $tf_k$ statistic of the $i$-th document. For the Laplace smoothing in the estimates of $\mu_k$ and $\overline{\mu}_k$, one can add 1 in numerators and $V$ in denominators.

The log-odds-ratio for the test prediction of an unseen document $d$ with $\boldsymbol{\phi}(d) = [tf_1, tf_2, \ldots, tf_V]^\top$ is derived as:

$$LOR = \log \frac{P(y=+1|d)}{P(y=-1|d)} = \log \frac{\pi}{1-\pi} + \sum_{k=1}^V tf_k \cdot \log \frac{\mu_k}{\overline{\mu}_k}. \tag{17}$$

Beyond the log-odds-ratio, we interpret (17) as a linear classifier equipped with term-weighted features similarly as previous section.

We regard the feature vector for $d$ as $\mathbf{x}(d) = \mathbf{tf}(d) \otimes \mathbf{r}$ where the weight vector $\mathbf{r}$ is defined to be:

$$r_k = \log \frac{\mu_k}{\overline{\mu}_k}. \tag{18}$$

Then the class decision based on the log-odds-ratio (17) is equivalent to the linear classifier $y = \text{sgn}(\mathbf{w}^\top \mathbf{x}(d) + w_0)$ with fixed parameters $\mathbf{w} = \mathbf{1}$ and $w_0 = \log \frac{\pi}{1-\pi}$. Our proposal is to learn the linear SVM classifier with the weighted features as in (18) rather than using the fixed parameters. This feature weighting scheme is denoted by *tf-mul*. By plugging the maximum likelihood estimates, the *tf-mul* weight vector (18) is equivalently rephrased as:

$$r_k = \log \frac{TF_-}{TF_+} + \log \frac{\sum_{i\in\mathcal{I}_+} tf_k^{(i)}}{\sum_{i\in\mathcal{I}_-} tf_k^{(i)}}, \tag{19}$$

where $TF_+$ = the sum of all $tf$'s across all positive documents ($TF_-$ defined similarly). In addition, replacing the nominal *tf* vector with the *tp* vector gives birth to an alternative weighting scheme which we denote by *tp-mul*.

## 5 Experiments

In this section we test the performance of the proposed term weighting schemes on several popular benchmark datasets for document classification. We summarize the competing approaches in the next subsection with description of how to use the features from the competing approaches in classification. Some details of the datasets are provided in Section 5.2, in which the experimental setups are also stated. The test performance measured in averaged $F_1$ scores as well as the test statistics to show statistical significance is reported in Section 5.3.

### 5.1 Competing approaches

The competing approaches discussed in the previous sections are summarized as follows:

- ***tf***: The feature representation of a document is simply the term-frequency vector.
- ***tf-idf***: The *tf* vector weighted by the inverse-document-frequency vector. This is a typical example of unsupervised feature weighting.
- ***tp***: Another baseline feature representation of a document based on the term-presence vector.
- ***tp-idf***: The *tp* vector weighted by the inverse-document-frequency vector.
- ***tf-rf***: The relevance-frequency weighted (i.e., (6)) *tf* vector representation proposed in [20, 21]. This is known to be one of the most successful supervised feature weighting schemes by far.
- ***DFIB***: Non-parametric term weighting based on the statistical independence [19].
- ***DFIB-XXX***: Some variants of the DFIB scheme [19]. Specifically, DFIB-CTI multiplies the term weights by the so-called total inertia that measures the deviation from statistical independence) and DFIB-IDF performs re-weighting by the inverse document frequencies.
- ***OR-TP***: The term weighting scheme of [8] based on the importance of terms for expressing sentiments. Specifically it multiplies the term presence by the maximum of the odds ratios between two classes, each estimated for documents in the same class.
- ***IG-TF***: Another scheme in [8] which re-weighs the term frequency by the information gain.
- ***tp-bnb***: Our main proposed weighting scheme based on the binary Naive Bayes model. The *tp* vector is weighted by the weight vector in (13).
- ***tf-mul***: Our main proposed strategy based on the multinomial term model. The *tf* vector is weighted by the weight vector in (18).
- ***tf-bnb***: A variant of *tp-bnb* by taking the *tf* vector (instead of *tp*) as a nominal feature.
- ***tp-mul***: A variant of *tf-mul* by taking the *tp* vector (instead of *tf*) as a nominal feature.
- ***tp-bnb-mod***: A variant of *tp-bnb* with the partial pseudo log-odds-ratio. That is, the intuition-based approach

using (7) with the *tp* vector as a nominal feature vector.

– ***tf-bnb-mod***: A variant of *tf-bnb* with the partial pseudo log-odds-ratio. That is, the weight vector defined by (7) with the *tf* vector as a nominal feature vector.

Hence, the last six methods (*tp-bnb*, *tf-mul*, *tf-bnb*, *tp-mul*, *tp-bnb-mod*, and *tf-bnb-mod*) are what we propose, and the others are existing ones.

For each of the feature weighting schemes, once the feature vectors are computed, we use the linear SVM training for classification. Because our goal is to compare term weighting methods, the linear SVM can be a right tool for its fast learning and simplicity in use with only a few hyper-parameters to be tuned. Specifically, we use the publicly available LIBLINEAR[4] software package [14] that is specially tailored for large-scale problems implemented based on the popular LIBSVM toolbox. The hyperparameter $C$ in the SVM, namely the balancing constant that trades off the margin constraint violation against regularization, is estimated from the candidate set (from $C = 10^{-6}$ to $10^{6}$ with uniform spacing in log-space) by 5-fold cross-validation on the training set (i.e., selecting $C$ that attains the minimum classification error on the held-out portion of the training set). Note that we use the same training datasets to learn the term weights as well as the SVM classifiers.

## 5.2 Datasets

We consider four well-known benchmark datasets for text classification to test the performance of the proposed term-weighting approaches: 20 newsgroup, Reuters-21578, WebKB, and IMDB. The datasets are publicly available from the websites.[5,6] For all datasets, we perform fairly standard pre-processing of term stemming, which essentially removes derived words from a stem word. For instance, *fishing*, *fisher*, and *fished* are all regarded as the same root word *fish*. For this purpose, we apply the famous Porter's Stemmer algorithm [28]. In the below we provide brief summary of the datasets.

### *5.2.1 20 newsgroup*

The 20 newsgroup dataset is composed of about 20,000 newsgroup documents annotated by 20 different subject categories (classes). These class categories are often hierarchically structured, where some sibling categories (e.g., *crypt* and *electronics* under the parent concept *science*) appear to be quite similar to each other. Yet some categories are quite different from others (e.g., *guns* under *politics*). We follow the standard training/test set split as provided in the dataset: among the 18,821 documents, about 11,300 samples ($\sim$ 60 %) are designated as a training set, while the rest a test set. Compared to other datasets, the class proportions are well balanced. We convert this 20-way classification problem into 20 binary problems by the standard one-vs-others transformation. The number of terms (i.e., the vocabulary size $V$) is 54,487 in total, which are collected from the training corpus only.

### *5.2.2 Reuters-21578 (R8 and R52)*

The Reuters-21578 dataset consists of collections of documents on the Reuters newswire in 1987. Among several variants of the dataset, we use two popular datasets: the $R8$ dataset that is formed by documents from 8 classes (e.g., *crude*, *earn*, *ship*, and so forth) having the largest numbers of samples, and the $R52$ dataset that comprises of 52 categories. These datasets have only the documents with single topics. We also conform to the training test split provided in the datasets (the ModApte split), which guarantees that all classes have at least one training and one test example. In summary, there are 5,485 training documents and 2,189 test samples in the $R8$, while 6532/2568 training/test documents in the $R52$. The number of terms collected from the training documents is 14,577 in the $R8$ dataset. We form $K$ different binary classification problems by the one-vs-others treatment where $K = 8$ or 52.

### *5.2.3 WebKB*

The WebKB dataset, from the world-wide knowledge base project of the CMU text learning group, is a collection of approximately 4,000 web pages gather from various universities in US, where the web pages are manually labeled as one of six or seven classes including *student*, *faculty*, *staff*, and more. After discarding classes with small numbers of samples, there are overall four classes: *student*, *faculty*, *course*, and *project*. Similar to the setup of [2], the documents are partitioned randomly into training/test sets with proportions about two thirds and one third, respectively. The vocabulary size is $V = 7,290$.

### *5.2.4 IMDB*

This large-scale dataset contains about 50,000 movie reviews collected in [26] for the purpose of sentiment analysis. Each movie has no more than 30 reviews where the review scores range from 1 (worst) to 10 (best). The related

[4] http://www.csie.ntu.edu.tw/~cjlin/liblinear/.

[5] http://web.ist.utl.pt/acardoso/datasets/. (Also, [2])

[6] http://ai.stanford.edu/~amaas/data/sentiment/.

classification task is to predict the sentiment (either positive or negative) of each review, thus considered as a binary text classification problem. The review scores are converted into binary sentiment classes by thresholding: score $\leq 4$ is regarded as a negative sentiment, while score $\geq 7$ a positive class. Neutral reviews (score 5 or 6) are simply discarded from the dataset. They divided the reviews into training/test sets with equal proportions while preserving the positive/negative proportions about the same for both sets. Overall there are approximately 90,000 terms collected from the training set.

## 5.3 Results and analysis

We first describe the performance measures used to contrast the competing approaches with the McNemar's significance test [10] to indicate statistical significance. In what follows, the test classification results are reported with some qualitative discussion.

### *5.3.1 Performance measures*

Motivated from the information retrieval literature [27], we use the $F_1$ measure that incorporates both the precision and recall scores of the classifiers equipped with competing feature weighting schemes. In the binary classification, the precision ($p$) of a classifier is defined as the proportion of the true positives ($TP$) out of the predicted positives ($PP$), i.e., $p = \frac{TP}{PP}$, while the recall ($r$) is defined as $r = \frac{TP}{WP}$ where $WP$ is the number of whole positive documents in the test set. Then one defines $F_1 = \frac{2pr}{p+r}$.

Note that four of the datasets we consider exhibit multi-class problems. Since we deal with a $K$-way classification problem by $K$ different binary classification problems via one-vs-others transformation (i.e., for each $c = 1, \ldots, K$, documents with class $c$ are regarded as $+$ class, while the rest as $-$ class), we consider two (micro and macro) averaged $F_1$ scores following [20, 21]. Specifically, the micro-averaged $F_1$ is defined as: $micro - F_1 = \frac{2PR}{P+R}$ where $P = \frac{\sum_c TP(c)}{\sum_c PP(c)}$ and $R = \frac{\sum_c TP(c)}{\sum_c WP(c)}$ with $TP(c)$, PP(c), and $RP(c)$ defined for the $c$-th binary classification problem. On the other hand, the macro-averaged $F_1$ is defined as the $F_1$ score with averaged precision and recall over $K$ problems, that is, $macro - F_1 = \frac{2P'R'}{P'+R'}$ $P' = (1/K)\sum_c p(c)$ and $R' = (1/K)\sum_c r(c)$ where $p(c)$ and $r(c)$ are the precision and the recall confined to the $c$-th problem.

However, for the IMDB dataset which is binary classification, we stick to the standard 0/1 accuracy $\left(\text{i.e., } \frac{the-number-of-docs-correctly-classified}{the-number-of-test-docs}\right)$ as is the most popular performance measure used for the dataset [26]. For both $F_1$ (micro and macro) and the accuracy, a higher score indicates a better classifier.

In addition to merely comparing the performance measures, we also report the statistical significance by the McNemar's test similar to that used in [20, 21]. In a nutshell, we compare two binary classifiers $C_A$ and $C_B$ by evaluating the statistics, $n_{10} = the - number - of - docs - C_A - correct - C_B - wrong$ and $n01 = the - number - of - docs - C_A - wrong - C_B - correct$. Then we utilize the fact that the test statistics $x = \frac{(|n01-n10|-1)^2}{n01+n10}$ conforms to the $\chi^2$ distribution with d.o.f. 1 under the null hypothesis that two classifiers are statistically indistinguishable. In particular, high value of $x$ indicates strong evidence to reject the null hypothesis, that is, hard to believe that $C_A$ and $C_B$ are indistinguishable (i.e., one is likely to be better than the other). The significance is quantified by the $p$-value, the probability of observing $x$ under the null hypothesis (say, $p < 0.05$ indicates strong statistical significance).

### *5.3.2 Test results*

We report the micro- and macro-averaged $F_1$ scores (accuracy for the IMDB dataset) of the competing term weighting strategies on test sets in Tables 1, 2, 3, 4, and 5. For each performance measure, the best methods are boldfaced. The last columns titled "Significance" contain the $p$-values in the McNemar's tests against the best method in the $micro - F_1$ measure (accuracy for the IMDB dataset). Based on the $p$-values, the markers indicating inferiority against the best method are depicted as: "=" assigned for

**Table 1** (20 Newsgroup) Test $micro - F_1$ and $macro - F_1$ scores (%) are shown

| Methods | $micro - F_1$ | $macro - F_1$ | Significance |
|---|---|---|---|
| *tf* | 74.58 | 73.75 | $p < 10^{-4}$ (−−) |
| *tf-idf* | 73.02 | 72.25 | $p < 10^{-4}$ (−−) |
| *tp* | 75.11 | 74.33 | $p < 10^{-4}$ (−−) |
| *tp-idf* | 73.89 | 73.15 | $p < 10^{-4}$ (−−) |
| *tf-rf* | 76.37 | 75.54 | $p = 0.0036$ (−) |
| *DFIB* | 75.50 | 74.81 | $p < 10^{-4}$ (−−) |
| *DFIB-CTI* | 68.84 | 68.21 | $p < 10^{-4}$ (−−) |
| *DFIB-IDF* | 72.04 | 71.29 | $p < 10^{-4}$ (−−) |
| *OR-TP* | 74.23 | 73.48 | $p < 10^{-4}$ (−−) |
| *IG-TF* | 71.88 | 71.06 | $p < 10^{-4}$ (−−) |
| *tp-bnb* | 76.48 | 75.89 | $p = 0.0026$ (−) |
| *tf-mul* | 76.54 | 75.81 | $p = 0.0035$ (−) |
| *tf-bnb* | 76.03 | 75.42 | $p < 10^{-4}$ (−−) |
| *tp-mul* | **77.20** | **76.55** | Best |
| *tp-bnb-mod* | 76.45 | 75.86 | $p = 0.0023$ (−) |
| *tf-bnb-mod* | 76.44 | 75.78 | $p = 0.0036$ (−) |

The last column ("Significance") contains the $p$-values in the McNemar's test against the best method in the $micro - F_1$ measure. In addition we place a marker indicating inferiority against the best method: "=" assigned for methods with $p$-values no less than 0.05 against the best method, "−" for $10^{-4} \leq p < 0.05$, and "−−" for $p < 10^{-4}$

**Table 2** (Reuters R8) Test $micro - F_1$ and $macro - F_1$ scores (%) are shown

| Methods | $micro - F_1$ | $macro - F_1$ | Significance |
|---|---|---|---|
| *tf* | 95.96 | 86.99 | $p = 0.0321$ (−) |
| *tf-idf* | 96.28 | 87.99 | $p = 0.4633$ (=) |
| *tp* | 95.02 | 85.29 | $p < 10^{-4}$ (−−) |
| *tp-idf* | 95.33 | 85.87 | $p = 0.0004$ (−) |
| *tf-rf* | 96.09 | 88.44 | $p = 0.1138$ (=) |
| *DFIB* | 95.47 | 86.99 | $p = 0.0007$ (−) |
| *DFIB-CTI* | 95.06 | 87.14 | $p < 10^{-4}$ (−−) |
| *DFIB-IDF* | 94.70 | 85.82 | $p < 10^{-4}$ (−−) |
| *OR-TP* | 94.79 | 87.39 | $p < 10^{-4}$ (−−) |
| *IG-TF* | 95.94 | 86.71 | $p = 0.0286$ (−) |
| *tp-bnb* | 95.82 | 88.20 | $p = 0.0263$ (−) |
| *tf-mul* | **96.45** | **88.56** | Best |
| *tf-bnb* | 96.01 | 87.05 | $p = 0.0302$ (−) |
| *tp-mul* | 95.94 | 88.54 | $p = 0.0573$ (=) |
| *tp-bnb-mod* | 95.82 | 88.28 | $p = 0.0242$ (−) |
| *tf-bnb-mod* | 96.26 | 88.15 | $p = 0.3556$ (=) |

The last column ("Significance") contains the $p$-values in the McNemar's test against the best method in the $micro - F_1$ measure. In addition we place a marker indicating inferiority against the best method: "=" assigned for methods with $p$-values no less than 0.05 against the best method, "−" for $10^{-4} \le p < 0.05$, and "−−" for $p < 10^{-4}$

methods with $p$-values no less than 0.05 against the best method (thus weak significance), "−" for $10^{-4} \le p < 0.05$, and "−−" for $p < 10^{-4}$ (corresponding to very strong significance).

**Table 3** (Reuters R52) Test $micro - F_1$ and $macro - F_1$ scores (%) are shown

| Methods | $micro - F_1$ | $macro - F_1$ | Significance |
|---|---|---|---|
| *tf* | 92.35 | 61.79 | $p = 0.0505$ (=) |
| *tf-idf* | 92.12 | 59.95 | $p = 0.0207$ (−) |
| *tp* | 91.01 | 54.04 | $p < 10^{-4}$ (−−) |
| *tp-idf* | 91.13 | 57.67 | $p < 10^{-4}$ (−−) |
| *tf-rf* | 92.42 | 61.97 | $p = 0.1051$ (=) |
| *DFIB* | 91.29 | 53.62 | $p < 10^{-4}$ (−−) |
| *DFIB-CTI* | 90.08 | 54.78 | $p < 10^{-4}$ (−−) |
| *DFIB-IDF* | 90.21 | 54.69 | $p < 10^{-4}$ (−−) |
| *OR-TP* | 90.31 | 44.71 | $p < 10^{-4}$ (−−) |
| *IG-TF* | 90.42 | 44.98 | $p < 10^{-4}$ (−−) |
| *tp-bnb* | 91.54 | 59.97 | $p = 0.0002$ (−) |
| *tf-mul* | **92.90** | 61.91 | Best |
| *tf-bnb* | 92.61 | **68.47** | $p = 0.1809$ (=) |
| *tp-mul* | 91.77 | 57.91 | $p = 0.0016$ (−) |
| *tp-bnb-mod* | 91.35 | 58.94 | $p < 10^{-4}$ (−−) |
| *tf-bnb-mod* | 92.19 | 67.27 | $p = 0.0038$ (−) |

The last column ("Significance") contains the $p$-values in the McNemar's test against the best method in the $micro - F_1$ measure. In addition we place a marker indicating inferiority against the best method: "=" assigned for methods with $p$-values no less than 0.05 against the best method, "−" for $10^{-4} \le p < 0.05$, and "−−" for $p < 10^{-4}$

**Table 4** (WebKB) Test $micro - F_1$ and $macro - F_1$ scores (%) are shown with the best method boldfaced

| Methods | $micro - F_1$ | $macro - F_1$ | Significance |
|---|---|---|---|
| *tf* | 88.75 | 87.01 | $p < 10^{-4}$ (−−) |
| *tf-idf* | 87.39 | 85.70 | $p < 10^{-4}$ (−−) |
| *tp* | 90.50 | 89.07 | $p = 0.1449$ (=) |
| *tp-idf* | 88.38 | 87.06 | $p < 10^{-4}$ (−−) |
| *tf-rf* | 88.38 | 86.58 | $p < 10^{-4}$ (−−) |
| *DFIB* | 90.69 | 89.25 | $p = 0.4879$ (=) |
| *DFIB-CTI* | 87.16 | 85.52 | $p < 10^{-4}$ (−−) |
| *DFIB-IDF* | 87.44 | 85.38 | $p < 10^{-4}$ (−−) |
| *OR-TP* | 90.17 | 88.56 | $p = 0.0023$ (−) |
| *IG-TF* | 88.47 | 86.27 | $p < 10^{-4}$ (−−) |
| *tp-bnb* | 90.73 | 89.78 | $p = 0.1360$ (=) |
| *tf-mul* | 89.06 | 86.96 | $p < 10^{-4}$ (−−) |
| *tf-bnb* | 89.58 | 88.21 | $p = 0.0006$ (−) |
| *tp-mul* | 91.06 | 89.61 | $p = 0.8283$ (=) |
| *tp-bnb-mod* | **91.12** | **90.04** | Best |
| *tf-bnb-mod* | 89.54 | 88.16 | $p = 0.0005$ (−) |

The last column ("Significance") contains the $p$-values in the McNemar's test against the best method in the $micro - F_1$ measure. In addition we place a marker indicating inferiority against the best method: "=" assigned for methods with $p$-values no less than 0.05 against the best method, "−" for $10^{-4} \le p < 0.05$, and "−−" for $p < 10^{-4}$

In the 20 newsgroup dataset (Table 1), most of the proposed approaches (*tf-bnb* exceptional) outperform all the existing methods for both $F_1$ measures, clearly supporting the effectiveness of the proposed term weighting strategies

**Table 5** (IMDB) Test accuracies (%) are shown with the best method boldfaced

| Methods | Accuracy | Significance |
|---|---|---|
| *tf* | 87.78 | $p < 10^{-4}$ (−−) |
| *tf-idf* | 87.41 | $p < 10^{-4}$ (−−) |
| *tp* | 87.83 | $p < 10^{-4}$ (−−) |
| *tp-idf* | 86.06 | $p < 10^{-4}$ (−−) |
| *tf-rf* | 87.87 | $p < 10^{-4}$ (−−) |
| *DFIB* | 88.41 | $p = 0.0443$ (−) |
| *DFIB-CTI* | 84.92 | $p < 10^{-4}$ (−−) |
| *DFIB-IDF* | 85.84 | $p < 10^{-4}$ (−−) |
| *OR-TP* | 88.57 | $p = 0.0562$ (=) |
| *IG-TF* | 88.18 | $p = 0.0021$ (−) |
| *tp-bnb* | **88.72** | Best |
| *tf-mul* | 88.63 | $p = 0.5727$ (=) |
| *tf-bnb* | 88.44 | $p = 0.0581$ (=) |
| *tp-mul* | 88.58 | $p = 0.0607$ (=) |
| *tp-bnb-mod* | 88.55 | $p = 0.0465$ (−) |
| *tf-bnb-mod* | 88.25 | $p = 0.0012$ (−) |

The last column ("Significance") contains the $p$-values in the McNemar's test against the best method. In addition we place a marker indicating inferiority against the best method: "=" assigned for methods with $p$-values no less than 0.05 against the best method, "−" for $10^{-4} \le p < 0.05$, and "−−" for $p < 10^{-4}$

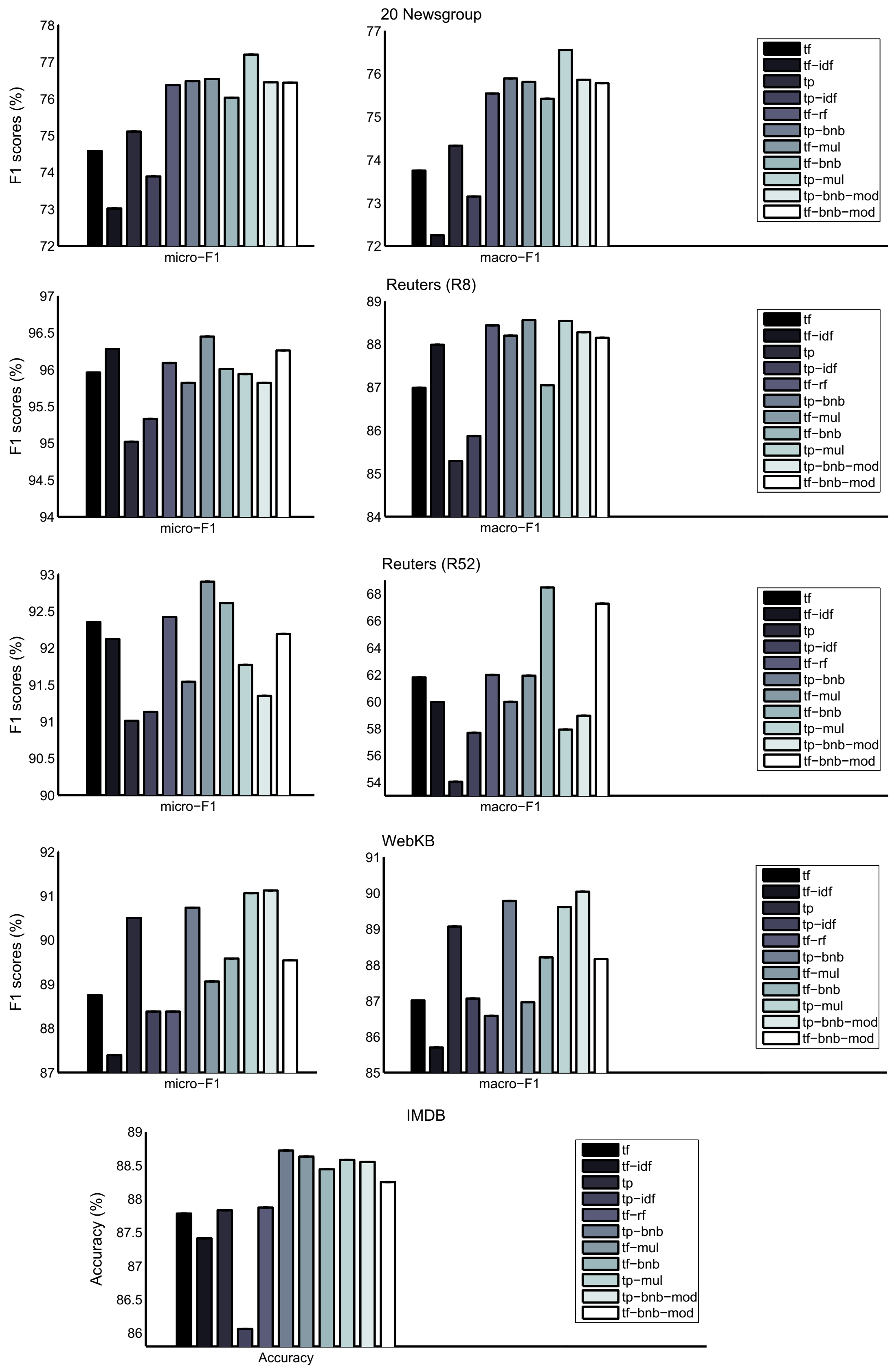


**Fig. 3** The bar plots depicting the test performance scores of the competing methods (please refer to the legends) on five datasets. Among the nine methods in each measure group, the left five correspond to existing methods while the rest six are the proposed ones

based on both intuition and theoretical underpinnings. The best performing one is *tp-mul*, the variant of the *tf-mul* induced from the multinomial term model, with the nominal features replaced by the term presence vectors. By comparing the same strategies with different nominal features, it turns out that the dataset overall exhibits such a trend that the presence/absence of terms is considered to be more discriminative than their frequencies. The superiority of the *tp-mul* to the other ones is also statistically significant as the $p$-values indicate.

In the Reuters $R8$ dataset (Table 2), the best method is our *tf-mul*, the principled term-weighting method derived from a multinomial term model, outperforming others with some marginal significances. Compared to the 20 newsgroup dataset, the $F_1$ scores are overall much higher, implying that the classification task on this dataset is easier than the previous dataset. Noticeably, the differences between competing approaches are rather small, and indeed the statistical significance is not so strong. In other words, most approaches perform equally well. For the $R52$ dataset (Table 3), we have similar observations where *tf-mul* and *tf-bnb* perform the best in $micro - F_1$ and $macro - F_1$ measures, respectively.

In the WebKB dataset (Table 4), all the proposed approaches based on the term presence nominal features (*tp-bnb*, *tp-mul*, and *tp-bnb-mod*) outperform the existing five methods with strong statistical significance. The best performing method is the *tp-bnb-mod* (the variant of the *tp-bnb* with partial pseudo log-odds-ratio) for both measures, and the *tp-mul* (the variant of the *tf-mul*) achieves comparable performance. Within the margin of significance, the principled weighting strategy *tp-bnb* induced from a binary Naive Bayes model attains near top scores for both performance measures.

Finally, in the IMDB dataset (Table 5) where we report the classification accuracies only, the two principled methods (*tp-bnb* and *tf-mul*) are clearly superior to other strategies, while two are statistically indistinguishable. Promisingly, their performances are comparable to the state-of-the-art approaches in the sentiment analysis literature [26]: the best reported model in [26] attained 88.89 % accuracy while our best approach *tp-bnb* yields 88.72 %, nearly identical. Considering that the former approach is the word vector model specially tailored for the sentiment recognition task while exploiting more sophisticated semantic term-document features, the results achieved by the simple term presence features with the discriminative term weighting in our *tp-bnb* are quite promising. The difference in test accuracy is merely 0.17 %, which roughly amounts to 40 reviews mis-classified out of $25K$ documents.

In summary, except for few methods depending on the datasets, the proposed term weighting strategies consistently outperform the existing supervised (*tf-rf*) and unsupervised (*tf*, *tf-idf*, *tp*, and *tp-idf*) term weighting methods. To illustrate this visually more clear, we depict in Fig. 3 the bar plots of the test performance scores of the competing methods. Especially for the IMDB dataset, the performance improvement of our approaches against existing five methods are visually outstanding.

Next, we aim to demonstrate that our model-induced term-weighted features (*tp-bnb* and *tf-mul*) in conjunction with the SVM weight learning are indeed effective compared to the log-odds-ratio based classifier (i.e., no classifier weight learning, but fixed $\mathbf{w} = \mathbf{1}$). That is, we contrast the performance of *tp-bnb* and *tf-mul* with the LOR classifiers $y(\mathbf{x}) = \text{sign}\left(\log \frac{P(y=+1|\mathbf{x})}{P(y=-1|\mathbf{x})}\right)$ with the underlying probabilistic models, binary Naive Bayes and multinomial term models. The results are summarized in Table 6, where we denote them by *SVM-Ind-Feats* and *LOR-Ind-Feats*, respectively. Clearly, learning SVM classifier weights with the model-induced features results in significantly more accurate prediction than simply using the LOR classifier (i.e., classifiers weight fixed as $\mathbf{1}$), consistently for all datasets. This signifies the effectiveness of discriminative feature

**Table 6** Test performance in $micro - F_1$ and $macro - F_1$ measures (accuracy for IMDB) (%) comparing the LOR classifier and the SVM classifier with the proposed model-induced features for (Left) binary NB model and (Right) multinomial term model

| | Binary NB model | | Multinomial model | |
|---|---|---|---|---|
| Classifiers | $micro - F_1$ | $macro - F_1$ | $micro - F_1$ | $macro - F_1$ |
| (a) 20 Newsgroup | | | | |
| LOR-Ind-Feats | 22.09 | 24.88 | 74.39 | 74.43 |
| SVM-Ind-Feats | 76.48 | 75.89 | 76.54 | 75.81 |
| (b) Reuters (R8) | | | | |
| LOR-Ind-Feats | 62.52 | 49.02 | 93.91 | 83.82 |
| SVM-Ind-Feats | 95.82 | 88.20 | 96.45 | 88.56 |
| (c) Reuters (R52) | | | | |
| LOR-Ind-Feats | 6.24 | 9.93 | 87.19 | 28.73 |
| SVM-Ind-Feats | 91.54 | 59.97 | 92.90 | 61.91 |
| (d) WebKB | | | | |
| LOR-Ind-Feats | 74.25 | 73.74 | 81.77 | 80.68 |
| SVM-Ind-Feats | 90.73 | 89.79 | 89.06 | 86.96 |
| (e) IMDB | | | | |
| Classifiers | Binary NB Model | | Multinomial Model | |
| LOR-Ind-Feats | 82.74 | | 81.36 | |
| SVM-Ind-Feats | 88.72 | | 88.63 | |

weighting (like our approaches) in conjunction with the sophisticated classifier learning.

## 6 Conclusion

In this paper we have proposed novel term weighting schemes that are induced from popular probabilistic models for document data. From the extensive experiments on several large-scale benchmark datasets, we have verified that the proposed approaches based on both intuition and theoretical underpinnings can yield substantially higher test accuracy than existing term weighting methods. However, occasionally some (not all) of our approaches tend to lead to slight performance degradation for datasets that exhibit large class imbalances, which is to be investigated and addressed in our future work.

**Compliance with Ethical Standards** This work is supported by National Research Foundation of Korea (NRF-2013R1A1A1076101). The authors have no conflict of interest. This research does not involve human participants nor animals. Consent to submit this manuscript has been received tacitly from the authors' institution, Seoul National University of Science & Technology.

## References

1. Akhmedova S, Semenkin E, Sergienko R (2014) Automatically generated classifiers for opinion mining with different term weighting schemes. In: International Conference on Informatics in Control, Automation and Robotics
2. Cardoso-Cachopo A (2007) Improving Methods for Single-label Text Categorization. PhD Thesis, Instituto Superior Tecnico, Universidade Tecnica de Lisboa
3. Carmel D, Mejer A, Pinter Y, 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
4. Chy A, Seddiqui M (2014) Das, S. In: International Conference on Computer and Information Technology, Bangla news classification using naive Bayes classifier
5. Crammer K, Singer Y, Cristianini N, Shawe-taylor J, Williamson, B (2001) On the algorithmic implementation of multiclass kernel-based vector machines. J Mach Learn Res 2
6. Craven M, DiPasquo D, Freitag D, McCallum A, Mitchell T, Nigam K, Slattery S (1998) Learning to extract symbolic knowledge from the world wide web. In: Proceedings of the 21st national conference on Artificial intelligence (AAAI)
7. Debole F, Sebastiani F. (2003) Supervised term weighting for automated text categorization. In: Proceedings of the ACM symposium on Applied computing
8. Deng ZH, Luo KH, Yu HL (2014) A study of supervised term weighting scheme for sentiment analysis. Expert Syst Appl 41(7):3506–3513
9. Deng ZH, Tang SW, Yang DQ, Li M. Z. L. Y., Xie KQ (2004) A comparative study on feature weight in text categorization. Advanced Web Technologies and Applications. Lect Notes Comput Sci 3007:588–597
10. Dietterich TG (1998) Approximate statistical tests for comparing supervised classification learning algorithms. Neural Comput 10(7):1895–1923
11. Domingos P, Pazzani M (1996) Beyond independence: conditions for the optimality of the simple Bayesian classifier. In: International Conference on Machine Learning
12. Duda RO, Hart PE, Stork DG (2000) Pattern classification. Wiley
13. Escalante HJ, Garcia-Limon MA, Morales-Reyes A, Graff M, Gomez MM, Morales EF, Martinez-Carranza J (2015) Term-weighting learning via genetic programming for text classification. Knowl-Based Syst 83:176–189
14. Fan RE, Chang KW, Hsieh CJ, Wang XR, Lin CJ (2008) LIBLINEAR: A library for large linear classification. J Mach Learn Res 9:1871–1874
15. Fattah MA (2015) New term weighting schemes with combination of multiple classifiers for sentiment analysis. Neurocomputing 167:434–442
16. Jiang H, Li P, Hu X, Wang S. (2009) An improved method of term weighting for text classification. In: IEEE International Conference on Intelligent Computing and Intelligent Systems
17. Joachims T (1998) Text categorization with suport vector machines: learning with many relevant features. In: European Conference on Machine Learning
18. Ko Y (2012) A study of term weighting schemes using class information for text classification. In: ACM SIGIR conference on Research and development in information retrieval
19. Kocabas I, Dincer BT, Karaoglan B (2014) A nonparametric term weighting method for information retrieval based on measuring the divergence from independence. Inf Retr 17 (2):153–176
20. Lan M, Tan C, Su J, Lu Y (2009) Supervised and traditional term weighting methods for automatic text categorization. IEEE Trans Pattern Anal Mach Intell 31(4):721–735
21. Lan M, Tan CL, Low HB (2006) Proposing a new term weighting scheme for text categorization. In: Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)
22. Lang K (1995) Newsweeder: learning to filter netnews. In: International Conference on Machine Learning
23. Lewis D, Knowles K (1997) Threading electronic mail: a preliminary study. Int J Inf Process Manag 33(2):209–217
24. Li JR, Yang K. (2010) News clustering system based on text mining. In: International Conference on Advanced Management Science
25. Liu WY, Wang L, Wang T. (2010) Online supervised learning from multi-field documents for email spam filtering. In: International Conference on Machine Learning and Cybernetics
26. Maas AL, Daly RE, Pham PT, Huang D, Ng AY, Potts C (2011) Learning word vectors for sentiment analysis. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies
27. Manning CD, Raghavan P, Schütze H (2008) Introduction to information retrieval. Cambridge University Press
28. Porter MF (1980) An algorithm for suffix stripping. Program 14(3):130–137

29. Sahami M, Dumais S, Heckerman D, Horvitz E (1998) A bayesian approach to filtering junk e-mail. In: Proceedings of the 21st national conference on Artificial intelligence (AAAI)
30. Schölkopf B, Smola A (2002) Learning with Kernels. MIT Press, Cambridge
31. Shavlik J, Eliassi-Rad T (1998) Intelligent agents for web-based tasks: An advice-taking approach. In: Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)
32. Soucy P, Mineau GW (2005) Beyond tfidf weighting for text categorization in the vector space model. In: International Joint Conference on Artificial Intelligence
33. Spärck Jones K (1972) A statistical interpretation of term specificity and its application in retrieval. J Doc 28:11–21
34. Upasana CS (2010) A survey on text classification techniques for e-mail filtering. In: International Conference on Machine Learning and Computing
35. Xuan NP, Quang HL (2014) A new improved term weighting scheme for text categorization. Advances in Intelligent Systems and Computing 244:261–270
36. Youquan H, Jianfang X, Cheng X (2011) An improved naive Bayesian algorithm for web page text classification. In: International Conference on Fuzzy Systems and Knowledge Discovery