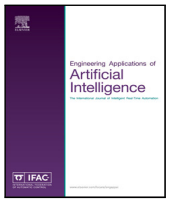
Engineering Applications of Artificial Intelligence 101 (2021) 104215

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Using modified term frequency to improve term weighting for text classification

Long Chen [a], Liangxiao Jiang [a,b,*], Chaoqun Li [c,**]

[a] School of Computer Science, China University of Geosciences, Wuhan 430074, China
[b] Key Laboratory of Artificial Intelligence, Ministry of Education, Shanghai, 200240, China
[c] School of Mathematics and Physics, China University of Geosciences, Wuhan 430074, China

ARTICLE INFO

Keywords:
Text classification
Term weighting
Term frequency factor
Collection frequency factor

ABSTRACT

Text classification (TC) is an essential task of natural language processing (NLP). In order to improve the performance of TC, term weighting is often used to obtain effective text representation by assigning appropriate weights to each term. A term weighting scheme is generally composed of term frequency factor, collection frequency factor and normalization factor. The normalization factor is commonly used as an optional factor to offset the influence of document length. Through the investigation of the existing term weighting schemes, we found that most of them focus on finding a more effective collection frequency factor, but rarely pay attention to finding a new term frequency factor. In this paper, we first proposed a new term frequency factor called modified term frequency (MTF). Different from the normalization factor, MTF directly modifies the raw term frequency based on the length information of all training documents. Then we proposed a new term weighting scheme by combining MTF with an existing collection frequency factor called modified distinguishing feature selector (MDFS). We denoted our scheme by MTF-MDFS (MDFS-based MTF). Extensive experimental results on 19 benchmark text datasets and 6 real-world text datasets show that our proposed MTF and MTF-MDFS are all much better than their state-of-the-art competitors in terms of the classification accuracy and the weighted average of F_1 of widely used base classifiers, such as MNB, SVM and LR.

1. Introduction

With the rapid growth of textual data on the Internet, how to use them effectively has become a challenge. Text classification (TC) is such an essential task of automatically assigning a set of textual documents to appropriate classes from a predefined set. There are various applications for TC, such as sentiment analysis (Hassonah et al., 2020), spam filtering (Gangavarapu et al., 2020), news classification (Silva et al., 2020) and so on. Text representation converts the content of a textual document into a compact format so that a text classification model can be trained in supervised learning and then used to classify documents. The vector space model (VSM) (Salton and McGill, 1984) is widely used due to its simplicity. In the VSM, the content of a textual document is represented as a vector of terms (features) in the term space. Terms can be words, phrases, or other complex units that identify the content of a document. To improve the performance of TC, a straightforward method is to identify the importance of different terms in distinguishing document categories. Term weighting provides such an effective text representation by assigning appropriate weights to each term (Zhang et al., 2016; Jiang et al., 2016a), and its validity in TC has been widely demonstrated.

In recent years, term weighting has become an important means to improve the performance of TC and has attracted more and more attention. A term weighting scheme is generally composed of three factors: term frequency factor, collection frequency factor and normalization factor (Dogan and Uysal, 2019). The formula can be expressed as:

$$TWS = TermFrequencyFactor \cdot CollectionFrequencyFactor \cdot NormalizationFactor, \quad (1)$$

where the term frequency factor reflects the importance of a specific term in a document. The *binary* is the simplest term frequency factor. However, it only considers whether a term occurs in the document, which limits its performance. The most commonly used term frequency factor is the raw term frequency (TF), which represents the number of occurrences of terms in a document. Moreover, some variants of TF can also achieve good performance in TC, e.g., logarithm of TF and square root of TF, which are used to weaken the impact of unrelated high-frequency terms on classification. The term frequency factor alone can be used as the term weight without other factors. However, in this case, all terms have the same importance, which may cause some terms

* Corresponding author at: School of Computer Science, China University of Geosciences, Wuhan 430074, China.
** Corresponding author.
E-mail addresses: lchen@cug.edu.cn (L. Chen), ljiang@cug.edu.cn (L. Jiang), chqli@cug.edu.cn (C. Li).

https://doi.org/10.1016/j.engappai.2021.104215
Received 6 November 2020; Received in revised form 30 January 2021; Accepted 23 February 2021
Available online xxxx
0952-1976/

with strong discriminating power to be underutilized in the process of classifying documents. The collection frequency factor states the distribution information of a specific term in the entire corpus, and its value generally reflects the importance of the term for classification. The normalization factor is used to eliminate the influence of document length, the most common of which is the cosine normalization (Salton and Buckley, 1988). The cosine normalization limit the term weight between 0 and 1. As an optional factor, the normalization factor can be adjusted according to the actual situation.

So far, there are a large number of term weighting schemes in the existing literature. Through the investigation of the existing term weighting schemes, we found that most of them focus on finding a more effective collection frequency factor, but rarely pay attention to finding a new term frequency factor. In fact, according to the previous studies, we found that an appropriate term frequency factor can also bring significant improvements to the performance of TC (Xuan and Quang, 2013; Chen et al., 2016; Dogan and Uysal, 2019).

In this paper, we focus our attention on finding a more effective term frequency factor, so as to further improve the classification performance. Since differences in document length may adversely affect the performance of TC, in this paper, we argue that the term frequency in each document should be adjusted according to the length information of all training documents, which is different from the cosine normalization to normalize the term weight. We consider that information may be lost in the normalization process. For example, the normalized term weights may no longer be applicable to some base classifiers that require frequency information, such as multinomial naive Bayes (MNB). Based on these premises, we first introduce the length information of all training documents into the term frequency factor and propose a new term frequency factor called modified term frequency (MTF). Different from the normalization factor, MTF directly modifies the raw term frequency based on the length information of all training documents. Then we propose a new term weighting scheme by combining MTF with an existing collection frequency factor called modified distinguishing feature selector (MDFS). We denoted our scheme by MTF-MDFS (MDFS-based MTF). Extensive comparison results on 19 benchmark text datasets and 6 real-world text datasets validate the advantages of MTF and MTF-MDFS in terms of the classification accuracy and the weighted average of F_1 of widely used base classifiers, such as MNB, support vector machines (SVM) and logistic regression (LR).

The remainder of this paper is structured as follows. We first review the existing well-known term weighting schemes in Section 2. Then, we propose our new term weighting scheme (MTF-MDFS) in Section 3. The experiments and results are reported in Section 4. Finally, we draw conclusions and summarize some main directions for future work in Section 5.

2. Related work

In this section, we present a comprehensive summary of existing well-known term weighting schemes, as listed in Table 1. In addition, some important notations with their related descriptions are listed in Table 2.

We start with the simplest term weighting scheme called *binary*, which contains only term frequency factor. The *binary* representation ignores the number of occurrences of terms (represented as 0 or 1), so it is not as reasonable and effective as the raw TF. However, the direct use of the raw TF may give a large weight to common terms with weak text discriminating ability, which will have an adverse impact on the classification performance. To compensate for this defect, a collection frequency factor IDF (inverse document frequency) is introduced into the TF scheme. The resulting term weighting scheme is TF–IDF (inverse document frequency-based TF) (Salton and Buckley, 1988). TF–IDF assigns weights to terms based on the number of times they occur in different documents. The less a term occurs in different documents, the more weight it is given.

The success of TF–IDF in TC makes researchers realize that the collection frequency factor, which reflects the distribution information of a specific term in the entire corpus, plays an important role in term weighting. Based on whether class information in the document is used, we can broadly divide term weighting schemes into two categories: namely unsupervised term weighting scheme (if it does not use class information) and supervised term weighting scheme (if it uses class information). Obviously, TF–IDF is an unsupervised term weighting scheme. To introduce the class information of training documents into the collection frequency factor, one of the most straightforward ways is to use term selection scores to assign different weights to terms. Inspired by this idea, the IDF global factor in TF–IDF is replaced by term selection metrics: χ^2 statistic (CHI), information gain (IG) and gain ratio (GR) (Debole and Sebastiani, 2003). The resulting supervised term weighting schemes are TF-CHI (Chi-square statistic-based TF), TF-IG (information gain-based TF), and TF-GR (gain ratio-based TF), respectively. Because of the use of class information, they are more promising and reasonable than TF–IDF.

Since then, researchers have paid more and more attention to the study of supervised term weighting schemes. If a high-frequency term is more concentrated in the positive class than in the negative class, it will make more contributions in selecting the positive samples from the negative samples. Based on this idea, Lan et al. (2009) proposed TF-RF (relevance frequency-based TF). In order to better distinguish the documents in minority classes, Liu et al. (2009) proposed a probability-based term weighting scheme called TF-PB (probability-based TF). Experiments verified the validity of TF-PB in improving the performance of imbalanced TC. Similar to IDF that measures the weight of a term on a single document, inverse class frequency (ICF) measures the weight of a term to a class of documents. When terms occur in only a few classes, they are more specific in corresponding classes, which helps to distinguish different classes, therefore, ICF assigns high weights to them. Based on this premise, Wang and Zhang (2013) proposed TF–ICF (inverse class frequency-based TF) and its variant TF–ICF-Based. By combining ICF and TF–IDF, Ren and Sohrab (2013) proposed TF–IDF–ICF (inverse document frequency and inverse class frequency-based TF). In addition, they found that TF–IDF–ICF provided positive discrimination on rare terms in the vector space, but biased against frequent terms for TC tasks. Then Ren and Sohrab (2013) modified the ICF function and proposed inverse class space density frequency (ICSDF), and generated the TF–IDF-ICSDF (inverse document frequency and inverse class space density frequency-based TF), which provided positive discrimination on infrequent and frequent terms. Class-specific terms help to distinguish different classes and has smaller entropy with respect to these classes. To explore the relationship between the discriminating power of a term and its entropy with respect to the classes of a corpus, Wang et al. (2015b) proposed TF-DC (distributional concentration-based TF) and TF-BDC (balanced distributional concentration-based TF). Chen et al. (2016) found that the more concentrated the inter-class distribution of a term is, the shorter the distance from the center of gravity is to the origin and the less the sum of class-specific gravity moments is, while the higher the maximum class-specific frequency is. Based on their observation, they used a new statistical model called inverse gravity moment (IGM) to accurately measure the class discrimination capability of terms, and then proposed TF-IGM (inverse gravity moment-based TF). Subsequently, Dogan and Uysal (2019) found that TF-IGM could not adequately express the distinguishing power of terms in certain cases, for which they proposed TF-IGM_{imp} (improved inverse gravity moment-based TF). To take full advantage of the distribution information of terms in all training documents, Chen et al. (2021) modified the distinguishing feature selector (DFS) (Uysal and Günal, 2012), which is a term selection method, and then proposed a collection frequency factor called modified distinguishing feature selector (MDFS) and a term weighting scheme called TF-MDFS (MDFS-based TF).

Table 1
Some well-known term weighting schemes in the existing literature.

Schemes	Term frequency factor	Collection frequency factor
binary	1	1
TF	f_{ik}	1
TF–IDF (Salton and Buckley, 1988)	f_{ik}	$\log_2\left(\frac{n}{d(t_i)}\right)$
TF-CHI (Debole and Sebastiani, 2003)	f_{ik}	$\frac{n\cdot\left(d(t_i,c_j)\cdot d(\overline{t_i},\overline{c_j})-d(\overline{t_i},c_j)\cdot d(t_i,\overline{c_j})\right)^2}{d(t_i)\cdot d(\overline{t_i})\cdot d(c_j)\cdot d(\overline{c_j})}$
TF-IG (Debole and Sebastiani, 2003)	f_{ik}	$\sum_{t\in\{t_j,\overline{t_j}\}}\sum_{c\in\{c_j,\overline{c_j}\}}\frac{d(t,c)}{n}\log_2\frac{n\cdot d(t,c)}{d(t)\cdot d(c)}$
TF-GR (Debole and Sebastiani, 2003)	f_{ik}	$\frac{\sum_{t\in\{t_j,\overline{t_j}\}}\sum_{c\in\{c_j,\overline{c_j}\}}\frac{d(t,c)}{n}\log_2\frac{n\cdot d(t,c)}{d(t)\cdot d(c)}}{-\sum_{c\in\{c_j,\overline{c_j}\}}\frac{d(c)}{n}\cdot\log_2\frac{d(c)}{n}}$
TF-RF (Lan et al., 2009)	f_{ik}	$\log_2\left(2+\frac{d(t_i,c_j)}{\max\left(1,d(t_i,\overline{c_j})\right)}\right)$
TF-PB (Liu et al., 2009)	f_{ik}	$\log_2\left(1+\frac{d(t_i,c_j)}{d(\overline{t_i},c_j)}\cdot\frac{d(t_i,c_j)}{d(t_i,\overline{c_j})}\right)$
TF–ICF (Wang and Zhang, 2013)	f_{ik}	$\log_2\left(1+\frac{q}{c(t_i)}\right)$
TF–ICF-Based (Wang and Zhang, 2013)	f_{ik}	$\log_2\left(2+\frac{d(t_i,c_j)}{\max\left(1,d(t_i,\overline{c_j})\right)}\cdot\frac{q}{c(t_i)}\right)$
TF–IDF–ICF (Ren and Sohrab, 2013)	f_{ik}	$\left(1+\log_2\frac{n}{d(t_i)}\right)\cdot\left(1+\log_2\frac{q}{c(t_i)}\right)$
TF–IDF–ICSDF (Ren and Sohrab, 2013)	f_{ik}	$\left(1+\log_2\frac{n}{d(t_i)}\right)\cdot\left(1+\log_2\frac{q}{\sum_{j=1}^{q}\frac{d(t_i,c_j)}{d(c_j)}}\right)$
TF-DC (Wang et al., 2015b)	f_{ik}	$1+\frac{1}{\log_2 q}\cdot\sum_{j=1}^{q}\frac{d(t_i,c_j)}{d(t_i)}\log_2\frac{d(t_i,c_j)}{d(t_i)}$
TF-BDC (Wang et al., 2015b)	f_{ik}	$1+\frac{1}{\log_2 q}\cdot\sum_{j=1}^{q}\frac{\frac{d(t_i,c_j)}{d(c_j)}}{\sum_{j=1}^{q}\frac{d(t_i,c_j)}{d(c_j)}}\log_2\frac{\frac{d(t_i,c_j)}{d(c_j)}}{\sum_{j=1}^{q}\frac{d(t_i,c_j)}{d(c_j)}}$
TF-IGM (Chen et al., 2016)	f_{ik}	$1+\lambda\frac{f_{i1}}{\sum_{r=1}^{q}f_{ir}\cdot r}$
TF-IGM$_{\text{imp}}$ (Dogan and Uysal, 2019)	f_{ik}	$1+\lambda\frac{f_{i1}}{\sum_{r=1}^{q}f_{ir}\cdot r+\log_{10}\left[\frac{D_{i-max}}{f_{i1}}\right]}$
TF-MDFS (Chen et al., 2021)	f_{ik}	$\sum_{j=1}^{q}\log_2\left(1+\frac{d(t_i,c_j)}{\max\left(1,d(t_i,\overline{c_j})\right)}\cdot\frac{d(\overline{t_i},\overline{c_j})}{\max\left(1,d(\overline{t_i},c_j)\right)}\right)\cdot\frac{P(c_j\vert t_i)P(\overline{c_j}\vert\overline{t_i})}{P(\overline{t_i}\vert c_j)+P(t_i\vert\overline{c_j})+1}$
LogTF-RF$_{\max}$ (Xuan and Quang, 2013)	$\log_2(1+f_{ik})$	$\max_{j=1}^{q}\log_2\left(2+\frac{d(t_i,c_j)}{\max\left(1,d(t_i,\overline{c_j})\right)}\right)$
SqrtTF-IGM (Chen et al., 2016)	$\sqrt{f_{ik}}$	$1+\lambda\frac{f_{i1}}{\sum_{r=1}^{q}f_{ir}\cdot r}$
SqrtTF-IGM$_{\text{imp}}$ (Dogan and Uysal, 2019)	$\sqrt{f_{ik}}$	$1+\lambda\frac{f_{i1}}{\sum_{r=1}^{q}f_{ir}\cdot r+\log_{10}\left[\frac{D_{i-max}}{f_{i1}}\right]}$

Table 2
List of important notations and their descriptions.

Notations	Descriptions
f_{ik}	the raw term frequency of term t_i in document d_k.
q	the total number of classes.
n	the total number of training documents.
m	the number of different terms in all documents.
T_i	the total number of raw frequency of term t_i in all training documents.
T_c	the total number of raw frequency in all training documents.
$d(t_i)$	the number of documents containing term t_i.
$d(\overline{t_i})$	the number of documents not containing term t_i.
$d(c_j)$	the number of documents belonging to class c_j .
$d(\overline{c_j})$	the number of documents not belonging to class c_j .
$d(t_i,c_j)$	the number of documents belonging to class c_j containing term t_i.
$d(\overline{t_i},c_j)$	the number of documents belonging to class c_j not containing term t_i.
$d(t_i,\overline{c_j})$	the number of documents not belonging to class c_j containing term t_i.
$d(\overline{t_i},\overline{c_j})$	the number of documents not belonging to class c_j not containing term t_i.
$c(t_i)$	the number of classes containing term t_i.
f_{ir}	the number of documents containing term t_i in the rth class (descending order).
D_{i-max}	the number of total documents in the 1-th class.
λ	the adjustable coefficient ranges from 5.0 to 9.0, and the default value is 7.0.
$P(c_j\vert t_i)$	the conditional probability of class c_j given the presence of term t_i.
$P(\overline{c}_j\vert\overline{t}_i)$	the conditional probability of the absence of class c_j given the absence of term t_i.
$P(\overline{t}_i\vert c_j)$	the conditional probability of the absence of term t_i given the presence of class c_j.
$P(t_i\vert\overline{c}_j)$	the conditional probability of term t_i given the absence of class c_j.

Obviously, the main purpose of the above term weighting schemes is to find a more effective collection frequency factor. In fact, an appropriate term frequency factor can also bring significant improvements to the performance of TC. To scale down the effect of noisy terms, Xuan and Quang (2013) used a logarithm operation on TF and proposed a new term frequency factor called logarithm of TF (LogTF). Then they proposed LogTF-RF$_{\max}$ (Maximum of all relevance frequency-based LogTF), which is an improvement to TF-RF. Their experimental results show that the schemes used LogTF factor yield better performance than others used TF factor. Different from LogTF, Chen et al. (2016) adopted an alternative method, i.e., calculating the square root of TF (SqrtTF). They found that the performance of SqrtTF on TC is equivalent to or sometimes better than LogTF. Based on this premise, they proposed SqrtTF-IGM (inverse gravity moment-based SqrtTF). Besides, Dogan and Uysal (2019) also used SqrtTF in their scheme, i.e., SqrtTF-IGM$_{\text{imp}}$ (improved inverse gravity moment-based SqrtTF). Their experimental results show that the SqrtTF-IGM$_{\text{imp}}$ scheme generally outperforms the TF-IGM$_{\text{imp}}$ scheme.

3. Proposed scheme

3.1. A new term frequency factor: MTF

To eliminate the influence of document length, researchers tend to use the normalization factor when assigning weights to terms. Specifically, many term weighting schemes use the cosine normalization to limit the term weight between 0 and 1. This raises the question of whether the information in the term frequency factor and the collection frequency factor may be lost in the normalization process. Due to the change of value range, the normalized weight to each term is even lower than that of the *binary* representation, and therefore no longer applicable to some base classifiers that require frequency information, such as multinomial naive Bayes (MNB). To avoid this problem, we directly modify the term frequency factor in this paper, so that the information in the term frequency factor and the collection frequency factor can be retained as much as possible.

Table 3
The benchmark datasets used in our experiments.

Dataset	#Documents	#Words	#Classes	#min class	#max class	#avg class
fbis	2463	2000	17	38	506	144.9
la1s	3204	31472	6	273	943	534.0
la2s	3075	31472	6	248	905	512.5
new3s	9558	26832	44	104	696	217.2
oh0	1003	3182	10	51	194	100.3
oh10	1050	3238	10	52	165	105.0
oh15	913	3100	10	53	157	91.3
oh5	918	3012	10	59	149	91.8
ohscal	11162	11465	10	709	1621	1116.2
re0	1504	2886	13	11	608	115.7
re1	1657	3758	25	10	371	66.3
tr11	414	6429	9	6	132	46.0
tr12	313	5804	8	9	93	39.1
tr21	336	7902	6	4	231	56.0
tr23	204	5832	6	6	91	34.0
tr31	927	10128	7	2	352	132.4
tr41	878	7454	10	9	243	87.8
tr45	690	8261	10	14	160	69.0
wap	1560	8460	20	5	341	78.0

Our inspiration is derived from a work in information retrieval (Amati and van Rijsbergen, 2002). Since the raw term frequency is closely related to the document length, we can generate the expected term frequencies in a document by comparing the current document length with a given length (usually the average document length). At the same time, we should determine the distribution of term frequencies in documents of different lengths. Based on this premise, a density function $\rho(l)$ with the document length l as an independent variable is introduced to represent the term frequency within a document. For each document d_k $(k = 1, 2, \ldots, n)$, the density function $\rho(l)$ of term t_i is defined as:

$$\rho(l) = s \cdot \frac{f_{ik}}{l}, \tag{2}$$

where s is a constant, and f_{ik} is the raw term frequency of term t_i in d_k. According to Amati and van Rijsbergen (2002), we can also make two initial assumptions on the term frequency density $\rho(l)$:

1. The distribution of a term is uniform in the document. The term frequency density $\rho(l)$ is a constant ρ_c.
2. The term frequency density $\rho(l)$ is a decreasing function of the length l.

To resize the term frequency, we calculate the integral on the same interval $[l(d_k), l(d_k) + avg\_l]$, where $l(d_k)$ is the document length of d_k, which is defined as the sum of all term frequencies, and avg_l is the average document length of all training documents. The resized term frequency (simply denoted as RTF) is defined as:

$$RTF(t_i, d_k) = \int_{l(d_k)}^{l(d_k)+avg\_l} \rho(l)dl. \tag{3}$$

According to the hypothesis 1, the RTF of term t_i in document d_k is:

$$\begin{aligned} RTF_1(t_i, d_k) &= \int_{l(d_k)}^{l(d_k)+avg\_l} \rho_c dl = \rho_c \cdot avg\_l \\ &= s \cdot f_{ik} \cdot \frac{avg\_l}{l(d_k)}, \end{aligned} \tag{4}$$

whereas, according to the hypothesis 2, the RTF of term t_i in document d_k is:

$$\begin{aligned} RTF_2(t_i, d_k) &= \int_{l(d_k)}^{l(d_k)+avg\_l} s \cdot \frac{f_{ik}}{l} dl \\ &= s \cdot f_{ik} \cdot \int_{l(d_k)}^{l(d_k)+avg\_l} \frac{dl}{l} \\ &= s \cdot f_{ik} \cdot \log_e(1 + \frac{avg\_l}{l(d_k)}). \end{aligned} \tag{5}$$

Assume that the average length of all training documents is equal to the effective length of the document. When $l(d_k)$ is equal to avg_l, $RTF(t_i, d_k)$ is equal to f_{ik}. So in this case, we can figure out that the constant s is 1 under the hypothesis 1 and $s = 1/\log_e 2 = \log_2 e$ under the hypothesis 2. Under the hypothesis 1 and the hypothesis 2, the RTF of term t_i in document d_k is defined by Eqs. (6) and (7), respectively.

$$RTF_1(t_i, d_k) = f_{ik} \cdot \frac{avg\_l}{l(d_k)}. \tag{6}$$

$$\begin{aligned} RTF_2(t_i, d_k) &= \log_2 e \cdot f_{ik} \cdot \log_e(1 + \frac{avg\_l}{l(d_k)}) \\ &= f_{ik} \cdot \log_2(1 + \frac{avg\_l}{l(d_k)}). \end{aligned} \tag{7}$$

In the experiments, Amati and van Rijsbergen (2002) found that in most cases, the RTF under the hypothesis 2 is superior to that under the hypothesis 1. Therefore, we only adopt the RTF under the hypothesis 2 for our study. Moreover, we consider that when the document length differs greatly, it is easy to cause the RTF value to be too large or too small. So we also use a square root function to suppress this effect. Based on these premises, we propose a new term frequency factor called modified term frequency (MTF), which is defined as:

$$MTF(t_i, d_k) = \sqrt{f_{ik} \cdot \log_2(1 + \frac{avg\_l}{l(d_k)})}. \tag{8}$$

3.2. A new term weighting scheme: MTF-MDFS

In our previous work, we proposed a new collection frequency factor called modified distinguishing feature selector (MDFS) (Chen et al., 2021). MDFS makes full use of the distribution information of terms in all training documents, and it is more effective than other collection frequency factors for term weighting in most cases. However, we did not consider the influence of term frequency factor on term weighting schemes. In this subsection, we use our proposed MTF above to improve our previous work.

We start with an introduction to MDFS. MDFS is derived from a widely accepted term selection method called distinguishing feature selector (DFS) (Uysal and Günal, 2012). The DFS argues that an ideal term (feature) selection method should assign high scores to distinctive terms and assign low scores to irrelevant terms. Specifically, DFS must meet four requirements:

1. If a term occurs frequently in a single class and does not occur in other classes, it is distinctive and must be assigned a high score.
2. If a term occurs frequently in all classes, it is irrelevant and must be assigned a low score.
3. If a term occurs rarely in a single class and does not occur in other classes, it is irrelevant and must be assigned a low score.
4. If a term occurs in some of the classes, it is relatively distinctive and must be assigned a medium score.

DFS defines the term selection score of term t_i as follows:

$$DFS(t_i) = \sum_{j=1}^{q} \frac{P(c_j|t_i)}{P(\bar{t}_i|c_j) + P(t_i|\bar{c}_j) + 1}. \tag{9}$$

We used DFS as a collection frequency factor for term weighting and found that its performance was very limited. By analyzing the formula of DFS, we found that the specificity of a term in a class has not been adequately demonstrated. To address this issue, we argue that an ideal term selection score should satisfy the fifth requirement: "For class c_j, $d(t_i, c_j)$ must be large and $d(\bar{t}_i, c_j)$ must be small; For class $\bar{c}_j$ (absence of class c_j), $d(\bar{t}_i, \bar{c}_j)$ must be large and $d(t_i, \bar{c}_j)$ must be small". Then we modified the DFS and proposed a new collection frequency factor simply denoted by MDFS. In MDFS, each term t_i has a class-specific score for each class c_j. In addition, a weighting factor is used for class-specific scores to further reflect the contributions of terms to a single

Table 4
Classification accuracy comparisons for MTF versus its competitors based on MNB.

Dataset	TF	LogTF	SqrtTF	MTF
fbis	77.11 ± 2.49	77.99 ± 2.11	77.97 ± 2.10	79.66 ± 2.05
la1s	88.41 ± 1.62	88.62 ± 1.54	88.64 ± 1.50	89.20 ± 1.50
la2s	89.88 ± 1.55	89.99 ± 1.58	89.80 ± 1.60	90.71 ± 1.65
new3s	79.28 ± 1.09	79.14 ± 1.12	78.53 ± 1.16	80.85 ± 1.17
oh0	89.55 ± 2.82	89.87 ± 2.87	89.81 ± 2.90	90.18 ± 2.77
oh10	80.60 ± 3.13	81.99 ± 3.01	81.88 ± 3.07	82.19 ± 3.03
oh15	83.60 ± 3.13	83.91 ± 3.50	83.78 ± 3.60	83.68 ± 3.52
oh5	86.63 ± 3.07	86.05 ± 3.41	85.96 ± 3.51	86.01 ± 3.42
ohscal	74.70 ± 1.18	74.75 ± 1.28	74.62 ± 1.29	74.70 ± 1.31
re0	80.02 ± 2.95	79.80 ± 2.81	79.67 ± 2.66	80.29 ± 2.85
re1	83.31 ± 2.75	81.29 ± 2.52	80.51 ± 2.49	81.03 ± 2.48
tr11	85.21 ± 4.90	85.35 ± 4.81	85.04 ± 4.58	86.61 ± 4.44
tr12	80.99 ± 6.08	85.17 ± 5.63	84.41 ± 5.52	85.97 ± 5.08
tr21	61.90 ± 8.78	80.22 ± 6.56	79.66 ± 6.77	86.18 ± 5.34
tr23	71.15 ± 9.68	75.33 ± 9.34	75.28 ± 9.12	90.85 ± 5.68
tr31	94.60 ± 2.41	95.81 ± 2.41	95.85 ± 2.39	96.04 ± 2.34
tr41	94.65 ± 2.21	94.35 ± 2.06	94.31 ± 2.05	94.42 ± 1.92
tr45	83.64 ± 4.33	88.57 ± 3.23	88.19 ± 3.38	91.26 ± 2.92
wap	81.22 ± 2.59	79.65 ± 2.54	79.26 ± 2.56	79.75 ± 2.46
Average	82.44	84.10	83.85	85.77

Algorithm 1 MTF-MDFS learning in the training phase (D)

Input: D-a training document set
Output: All term weights $W_{\text{MTF-MDFS}}(t_i, d_k)$, avg_l, $MDFS(t_i)$ $(i = 1, 2, \cdots, m)$
for each document d_k $(k = 1, 2, \cdots, n)$ **do**
  Calculate $l(d_k)$
end for
Calculate avg_l
for each term t_i $(i = 1, 2, \cdots, m)$ and each class c_j $(j = 1, 2, \cdots, q)$ **do**
  Calculate w_{ij} by Eq. (11)
  Calculate $MDFS_{cs}(t_i, c_j)$ by Eq. (12)
end for
for each term t_i $(i = 1, 2, \cdots, m)$ **do**
  Calculate $MDFS(t_i)$ by Eq. (10)
  for each document d_k $(k = 1, 2, \cdots, n)$ **do**
    Calculate $MTF(t_i, d_k)$ by Eq. (8)
    Calculate $W_{\text{MTF-MDFS}}(t_i, d_k)$ by Eq. (13)
  end for
end for
return All term weights $W_{\text{MTF-MDFS}}(t_i, d_k)$, avg_l, $MDFS(t_i)$ $(i = 1, 2, \cdots, m)$

Algorithm 2 MTF-MDFS learning in the testing phase (d)

Input: d-a test document, avg_l, $MDFS(t_i)$ $(i = 1, 2, \cdots, m)$
Output: All term weights $W_{\text{MTF-MDFS}}(t_i, d)$
Calculate $l(d)$
for each term t_i $(i = 1, 2, \cdots, m)$ **do**
  Calculate $MTF(t_i, d)$ by Eq. (8)
  Calculate $W_{\text{MTF-MDFS}}(t_i, d)$ by Eq. (13)
end for
return All term weights $W_{\text{MTF-MDFS}}(t_i, d)$

class (Jiang et al., 2019b; Zhang et al., 2020). MDFS is defined as a global weighting factor, calculated by the weighted sum across all class-specific scores. The detailed formula is:

$$MDFS(t_i) = \sum_{j=1}^{q} w_{ij} \cdot MDFS_{cs}(t_i, c_j), \tag{10}$$

where w_{ij} and $MDFS_{cs}(t_i, c_j)$ represent the weighting factor and class-specific score of term t_i for class c_j, respectively. They are defined respectively as follows:

$$w_{ij} = \log_2\left(1 + \frac{d(t_i, c_j)}{\max\left(1, d(t_i, \overline{c_j})\right)} \cdot \frac{d(\overline{t_i}, \overline{c_j})}{\max\left(1, d(\overline{t_i}, c_j)\right)}\right), \tag{11}$$

$$MDFS_{cs}(t_i, c_j) = \frac{P(c_j|t_i)P(\overline{c}_j|\overline{t}_i)}{P(\overline{t}_i|c_j) + P(t_i|\overline{c}_j) + 1}. \tag{12}$$

Finally, by combining MTF with MDFS, we propose a new term weighting scheme called MTF-MDFS (MDFS-based MTF). The detailed formula is:

$$W_{\text{MTF-MDFS}}(t_i, d_k) = MTF(t_i, d_k) \cdot MDFS(t_i). \tag{13}$$

Now, the detailed learning processes for MTF-MDFS in the training and testing phases can be briefly depicted as **Algorithm 1** and **Algorithm 2**, respectively.

4. Experiments and results

4.1. Experiments on benchmark text datasets

The purpose of this section is to validate the effectiveness of our proposed term frequency factor MTF and term weighting scheme MTF-MDFS in terms of classification performance. Therefore, we at first compare MTF with TF and two existing state-of-the-art term frequency factors, including LogTF and SqrtTF. Then, we compare MTF-MDFS with TF-MDFS and three existing state-of-the-art term weighting schemes, including LogTF-RF_{max}, SqrtTF-IGM and SqrtTF-IGM_{imp}. Now, we introduce these competitors and their abbreviations as follows.

- LogTF-RF_{max}: Maximum of all relevance frequency-based LogTF (Xuan and Quang, 2013).
- SqrtTF-IGM: inverse gravity moment-based SqrtTF (Chen et al., 2016).
- SqrtTF-IGM_{imp}: improved inverse gravity moment-based SqrtTF (Dogan and Uysal, 2019).
- TF-MDFS: modified distinguishing feature selector-based TF (Chen et al., 2021).

We designed three groups of experiments to compare our proposed MTF and MTF-MDFS with their competitors, using the well-accepted base classifiers, including multinomial naive Bayes (MNB) (McCallum and Nigam, 1998; Jiang et al., 2013; Wang et al., 2015a), support vector machines (SVM) (Cortes and Vapnik, 1995) and logistic regression (LR) (Zhang and Oles, 2001; Aseervatham et al., 2012). Naive Bayes is widely used in text classification because of its simple model and high efficiency of calculation, among which MNB is a dominant modeling approach. SVM is one of the most effective classifiers for text classification. Compared to MNB, SVM is more suitable for high feature dimension learning. SVM separates samples of different classes by establishing a linear or nonlinear hyperplane. In this paper, we used the linear SVM classifier: LibLINEAR (Fan et al., 2008). LR is also an effective classifier for text classification. It directly models the possibility of classification without assuming the data distribution in advance, thus avoiding the problem caused by inaccurate hypothesis

Table 5
Classification accuracy comparisons of Wilcoxon tests with regard to MTF based on MNB.

Algorithm	TF	LogTF	SqrtTF	MTF
TF	–			∘
LogTF		–	•	∘
SqrtTF		∘	–	∘
MTF	•	•	•	–

distribution. Linear regression can also perform well, but it cannot always get the membership values between 0 and 1. LR uses a sigmoid function to approximate the logarithmic probability of the true marker with the predicted results of the linear regression model. We also used LIBLINEAR to find the optimal solution in LR. We used the existing implementations of MNB, SVM (LibLINEAR with L2-regularized L2-loss support vector classification (dual)) and LR (LibLINEAR with L2-regularized logistic regression (dual)) in the WEKA platform (Witten et al., 2011) and implemented all term frequency factors and term weighting schemes in the WEKA platform.

We ran our experiments on 19 widely used multi-class text datasets[1] published on the main web site of the WEKA platform, which represent a wide range of domains and data characteristics. Table 3 describes the detailed information of these 19 benchmark text datasets. All these datasets are derived from TREC, OHSUMED-233445, Reuters-21578, and WebACE. They were originally converted to term counts by Han and Karypis (2000). Dataset "fbis" comes from the Foreign Broadcast Information Service data of TREC-5. Datasets "la1s" and "la2s" come from the Los Angeles Times data of TREC-5. The classes of "la1s" and "la2s" were generated from the name of the news-paper sections that these articles appeared, such as "Entertainment", "Financial", "Foreign", "Metro", "National", and "Sports". Datasets "new3s", "tr11", "tr12", "tr21", "tr23", "tr31", "tr41", and "tr45" are derived from TREC-5, TREC-6, and TREC-7 collections. The classes of the various "trXX", "new3s", and "fbis" datasets were generated according to the relevance judgment provided in these collections. Datasets "oh0", "oh5", "oh10", "oh15", and "ohscal" are from OHSUMED collection subset of MEDLINE database, and their classes were divided into different subsets to construct these datasets. Datasets "re0" and "re1" evolves from Reuters-21578 text categorization test collection Distribution 1.0. The labels of this corpus were divided into 2 sets and the documents with a single label were selected to construct datasets accordingly. Dataset "wap" derives from the WebACE project (WAP), where each document corresponds to a web page.

Tables 4, 6 and 8 show the detailed classification accuracy comparisons for MTF versus its competitors based on MNB, SVM and LR, respectively. In our experiments, the classification accuracy is defined as the percentage of correctly classified test instances in the total test instances. All classification accuracy estimates were obtained by averaging the results from 10 separate runs of stratified 10-fold cross-validation. The average of each classifier across all datasets provides a gross indicator of the relative performance in addition to other statistics.

Then, we took advantage of the well-known KEEL[2] Data-Mining Software Tool (Alcalá-Fdez et al., 2011) to complete the Wilcoxon signed-rank tests (Demsar, 2006; Garcia and Herrera, 2008; Jiang et al., 2016b, 2019a) to thoroughly compare each pair of term weighting schemes. Tables 5, 7 and 9 summarize the detailed comparison results of the Wilcoxon tests with regard to MTF based on MNB, SVM and LR, respectively. In these Tables, ∘ indicates that the algorithm in that column improves the algorithm in the corresponding row, and • indicates that the algorithm in that row improves the algorithm in the corresponding column. The lower diagonal level of significance is $\alpha = 0.05$, and the upper diagonal level of significance is $\alpha = 0.1$. From these comparison results, we summarize the highlights as follows:

[1] https://waikato.github.io/weka-wiki/datasets/
[2] https://sci2s.ugr.es/keel/download.php

Table 6
Classification accuracy comparisons for MTF versus its competitors based on SVM.

Dataset	TF	LogTF	SqrtTF	MTF
fbis	82.98 ± 1.99	86.70 ± 2.09	86.68 ± 2.02	86.57 ± 1.87
la1s	88.30 ± 1.85	89.35 ± 1.76	89.55 ± 1.82	90.40 ± 1.60
la2s	90.15 ± 1.43	91.10 ± 1.55	91.22 ± 1.48	92.01 ± 1.49
new3s	87.33 ± 1.03	90.26 ± 0.98	90.34 ± 0.95	90.42 ± 0.97
oh0	89.49 ± 2.56	91.35 ± 2.53	91.37 ± 2.55	91.69 ± 2.54
oh10	80.36 ± 3.69	83.94 ± 2.97	84.06 ± 3.10	84.18 ± 3.01
oh15	84.26 ± 3.65	85.90 ± 3.48	85.78 ± 3.31	86.45 ± 3.31
oh5	89.77 ± 2.83	91.50 ± 2.74	91.38 ± 2.79	91.95 ± 2.52
ohscal	75.68 ± 1.29	78.09 ± 1.26	78.12 ± 1.25	78.17 ± 1.24
re0	84.91 ± 2.38	86.49 ± 2.60	86.41 ± 2.49	86.49 ± 2.49
re1	85.88 ± 2.62	87.56 ± 2.39	87.64 ± 2.32	87.39 ± 2.37
tr11	89.36 ± 4.14	91.36 ± 3.74	91.24 ± 3.87	90.73 ± 3.91
tr12	88.85 ± 5.30	91.08 ± 5.36	91.78 ± 5.07	91.72 ± 4.89
tr21	91.39 ± 4.30	95.18 ± 3.22	94.73 ± 3.40	95.80 ± 3.06
tr23	90.50 ± 6.41	91.05 ± 6.16	91.04 ± 6.03	92.31 ± 6.00
tr31	98.39 ± 1.36	98.99 ± 1.16	98.90 ± 1.12	98.78 ± 1.19
tr41	96.59 ± 2.00	98.11 ± 1.30	98.17 ± 1.34	98.30 ± 1.26
tr45	94.45 ± 2.77	96.80 ± 2.16	96.59 ± 2.20	96.70 ± 2.07
wap	85.15 ± 2.39	85.58 ± 2.29	85.66 ± 2.28	86.43 ± 2.31
Average	88.09	90.02	90.03	90.34

Table 7
Classification accuracy comparisons of Wilcoxon tests with regard to MTF based on SVM.

Algorithm	TF	LogTF	SqrtTF	MTF
TF	–	∘	∘	∘
LogTF	•	–		∘
SqrtTF	•		–	∘
MTF	•	•	•	–

Table 8
Classification accuracy comparisons for MTF versus its competitors based on LR.

Dataset	TF	LogTF	SqrtTF	MTF
fbis	85.42 ± 1.89	87.93 ± 1.81	88.04 ± 1.87	88.02 ± 1.81
la1s	89.83 ± 1.71	90.99 ± 1.53	91.03 ± 1.53	91.41 ± 1.48
la2s	91.41 ± 1.47	92.15 ± 1.37	92.08 ± 1.43	92.54 ± 1.42
new3s	88.75 ± 0.99	91.15 ± 0.87	91.13 ± 0.87	91.02 ± 0.85
oh0	90.12 ± 2.43	91.70 ± 2.47	91.60 ± 2.51	91.75 ± 2.64
oh10	82.05 ± 3.39	84.70 ± 3.04	84.79 ± 3.20	84.94 ± 3.16
oh15	85.22 ± 3.48	86.44 ± 3.26	86.35 ± 3.31	86.30 ± 3.16
oh5	90.81 ± 2.88	92.16 ± 2.47	92.10 ± 2.38	92.19 ± 2.39
ohscal	78.28 ± 1.20	80.46 ± 1.25	80.52 ± 1.21	80.53 ± 1.20
re0	86.87 ± 2.24	87.53 ± 2.10	87.55 ± 2.11	88.08 ± 2.14
re1	88.04 ± 2.41	88.56 ± 2.24	88.57 ± 2.26	88.48 ± 2.23
tr11	89.51 ± 3.89	90.37 ± 3.66	90.32 ± 3.62	90.42 ± 3.67
tr12	89.64 ± 5.05	91.34 ± 4.92	91.85 ± 4.58	91.37 ± 4.57
tr21	91.99 ± 4.05	93.57 ± 3.42	93.33 ± 3.58	95.00 ± 2.99
tr23	90.50 ± 6.73	91.92 ± 5.84	91.29 ± 6.13	92.02 ± 5.89
tr31	98.17 ± 1.39	98.96 ± 1.18	98.90 ± 1.12	98.81 ± 1.18
tr41	96.61 ± 1.82	98.09 ± 1.41	98.20 ± 1.26	98.21 ± 1.37
tr45	94.48 ± 2.59	95.88 ± 2.07	95.83 ± 2.02	96.04 ± 2.01
wap	85.66 ± 2.37	86.22 ± 2.38	86.29 ± 2.28	86.79 ± 2.35
Average	89.12	90.53	90.51	90.73

Table 9
Classification accuracy comparisons of Wilcoxon tests with regard to MTF based on LR.

Algorithm	TF	LogTF	SqrtTF	MTF
TF	–	∘	∘	∘
LogTF	•	–		∘
SqrtTF	•		–	∘
MTF	•	•	•	–

1. In terms of MNB, the averaged classification accuracy of MTF on 19 datasets is 85.77%, which is considerably higher than those of TF (82.44%), LogTF (84.10%) and SqrtTF (83.85%). In terms of SVM, the averaged classification accuracy of MTF on 19 datasets is 90.34%, which is relatively higher than those of

Table 10
Classification accuracy comparisons for MTF-MDFS versus its competitors based on MNB.

Dataset	LogTF-RF$_{max}$	SqrtTF-IGM	SqrtTF-IGM$_{imp}$	TF-MDFS	MTF-MDFS
fbis	79.46 ± 2.30	79.47 ± 2.35	79.47 ± 2.35	79.97 ± 2.57	82.03 ± 2.16
la1s	89.71 ± 1.64	89.92 ± 1.72	89.91 ± 1.62	88.80 ± 1.66	89.58 ± 1.63
la2s	90.82 ± 1.43	90.69 ± 1.55	90.87 ± 1.49	90.21 ± 1.52	91.04 ± 1.64
new3s	81.68 ± 1.22	82.39 ± 1.22	82.31 ± 1.25	82.55 ± 1.24	84.71 ± 1.16
oh0	93.88 ± 2.30	93.46 ± 2.47	93.75 ± 2.33	93.55 ± 2.20	94.49 ± 2.11
oh10	85.02 ± 2.86	84.34 ± 3.06	84.69 ± 2.84	84.60 ± 2.90	86.21 ± 2.89
oh15	87.26 ± 3.11	86.87 ± 3.15	87.20 ± 3.04	88.35 ± 2.68	88.40 ± 2.83
oh5	91.50 ± 2.68	91.21 ± 2.69	91.48 ± 2.77	93.06 ± 2.77	93.53 ± 2.50
ohscal	76.93 ± 1.25	76.52 ± 1.20	77.14 ± 1.22	79.20 ± 1.20	80.04 ± 1.27
re0	79.67 ± 2.96	78.10 ± 3.22	78.73 ± 3.12	82.71 ± 2.82	82.75 ± 2.64
re1	85.76 ± 2.02	84.57 ± 2.44	85.37 ± 2.31	88.18 ± 2.21	87.93 ± 2.04
tr11	87.33 ± 4.26	87.57 ± 4.16	87.34 ± 4.12	88.66 ± 4.24	90.11 ± 4.38
tr12	89.04 ± 5.18	87.38 ± 5.67	88.21 ± 5.27	86.67 ± 6.14	90.51 ± 4.02
tr21	82.71 ± 5.65	84.93 ± 5.20	84.16 ± 5.42	80.90 ± 6.44	92.16 ± 3.88
tr23	83.52 ± 8.14	80.24 ± 8.46	80.69 ± 8.18	89.98 ± 6.76	94.90 ± 4.62
tr31	96.98 ± 1.94	97.37 ± 1.82	97.37 ± 1.89	97.80 ± 1.82	97.84 ± 1.67
tr41	95.33 ± 2.16	94.54 ± 2.47	95.06 ± 2.33	95.96 ± 2.05	96.53 ± 1.70
tr45	92.57 ± 2.84	93.03 ± 2.44	93.25 ± 2.52	93.07 ± 3.11	95.00 ± 2.05
wap	82.88 ± 2.39	81.67 ± 2.67	82.50 ± 2.64	84.35 ± 2.58	84.67 ± 2.38
Average	86.95	86.54	86.81	87.82	89.60

Table 11
Classification accuracy comparisons of Wilcoxon tests with regard to MTF-MDFS based on MNB.

Algorithm	LogTF-RF$_{max}$	SqrtTF-IGM	SqrtTF-IGM$_{imp}$	TF-MDFS	MTF-MDFS
LogTF-RF$_{max}$	–			∘	∘
SqrtTF-IGM		–	∘	∘	∘
SqrtTF-IGM$_{imp}$		•	–	∘	∘
TF-MDFS		•		–	∘
MTF-MDFS	•	•	•	•	–

Table 13
Classification accuracy comparisons of Wilcoxon tests with regard to MTF-MDFS based on SVM.

Algorithm	LogTF-RF$_{max}$	SqrtTF-IGM	SqrtTF-IGM$_{imp}$	TF-MDFS	MTF-MDFS
LogTF-RF$_{max}$	–			•	∘
SqrtTF-IGM		–		•	∘
SqrtTF-IGM$_{imp}$			–	•	∘
TF-MDFS	∘	∘	∘	–	∘
MTF-MDFS	•	•	•	•	–

TF (88.09%), LogTF (90.02%) and SqrtTF (90.03%). In terms of LR, the averaged classification accuracy of MTF on 19 datasets is 90.73%, which is comparatively higher than those of TF (89.12%), LogTF (90.53%) and SqrtTF (90.51%). This indicates that our proposed MTF is very effective.

2. According to the Wilcoxon signed-rank tests results, our proposed MTF significantly outperforms all of its competitors, including TF, LogTF and SqrtTF. From these comparison results, we can conclude that an effective term frequency factor is very important for improving the performance of TC and our proposed MTF is the best one among all the term frequency factors used to compare.

Tables 10, 12 and 14 show the detailed classification accuracy comparisons for MTF-MDFS versus its competitors based on MNB, SVM and LR, respectively. Tables 11, 13 and 15 summarize the detailed comparison results of the Wilcoxon tests with regard to MTF-MDFS based on MNB, SVM and LR, respectively. From these comparison results, we can see the following:

1. In terms of MNB, the averaged classification accuracy of MTF-MDFS on 19 datasets is 89.60%, which is much higher than those of LogTF-RF$_{max}$ (86.95%), SqrtTF-IGM (86.54%) SqrtTF-IGM$_{imp}$ (86.81%) and TF-MDFS (87.82%).
2. In terms of SVM, the averaged classification accuracy of MTF-MDFS on 19 datasets is 91.46%, which is also higher than those

Table 12
Classification accuracy comparisons for MTF-MDFS versus its competitors based on SVM.

Dataset	LogTF-RF$_{max}$	SqrtTF-IGM	SqrtTF-IGM$_{imp}$	TF-MDFS	MTF-MDFS
fbis	86.64 ± 1.99	86.89 ± 1.91	86.87 ± 1.88	84.40 ± 1.73	87.28 ± 1.82
la1s	89.65 ± 1.73	90.03 ± 1.73	89.78 ± 1.67	88.30 ± 1.75	90.85 ± 1.67
la2s	91.35 ± 1.58	91.43 ± 1.62	91.44 ± 1.59	90.07 ± 1.69	92.12 ± 1.61
new3s	90.83 ± 0.91	90.88 ± 0.94	90.86 ± 0.96	88.32 ± 1.00	91.38 ± 0.85
oh0	93.05 ± 2.47	93.77 ± 2.27	93.62 ± 2.25	92.26 ± 2.47	93.06 ± 2.49
oh10	85.95 ± 3.10	86.06 ± 2.74	86.05 ± 2.88	83.28 ± 3.31	85.83 ± 3.06
oh15	87.58 ± 3.02	87.86 ± 3.10	87.98 ± 2.95	86.69 ± 3.27	87.98 ± 3.18
oh5	93.38 ± 2.52	93.66 ± 2.45	93.67 ± 2.47	92.21 ± 2.80	93.60 ± 2.32
ohscal	79.08 ± 1.26	79.35 ± 1.37	79.39 ± 1.36	77.32 ± 1.22	79.84 ± 1.40
re0	85.80 ± 2.72	85.16 ± 2.72	85.32 ± 2.74	84.96 ± 2.55	86.60 ± 2.34
re1	87.54 ± 2.54	87.07 ± 2.73	87.22 ± 2.61	87.14 ± 2.33	88.59 ± 2.20
tr11	93.24 ± 3.57	94.13 ± 3.44	94.11 ± 3.52	91.09 ± 3.77	94.40 ± 3.22
tr12	92.52 ± 4.64	92.05 ± 4.53	92.23 ± 4.44	91.30 ± 5.08	91.71 ± 5.33
tr21	94.84 ± 3.65	94.49 ± 3.63	94.64 ± 3.65	94.31 ± 4.05	96.31 ± 2.99
tr23	94.77 ± 4.68	95.16 ± 4.74	95.21 ± 4.71	94.81 ± 4.68	96.18 ± 4.23
tr31	99.34 ± 0.78	99.38 ± 0.80	99.35 ± 0.84	98.71 ± 1.17	99.32 ± 0.82
tr41	98.66 ± 1.23	98.80 ± 1.17	98.78 ± 1.21	97.26 ± 1.82	98.11 ± 1.48
tr45	97.42 ± 1.86	97.25 ± 1.98	97.22 ± 1.91	96.52 ± 2.15	97.46 ± 1.76
wap	86.21 ± 2.30	85.81 ± 2.35	86.17 ± 2.41	86.01 ± 2.29	87.11 ± 2.47
Average	90.94	91.01	91.05	89.73	91.46

Table 14
Classification accuracy comparisons for MTF-MDFS versus its competitors based on LR.

Dataset	LogTF-RF$_{max}$	SqrtTF-IGM	SqrtTF-IGM$_{imp}$	TF-MDFS	MTF-MDFS
fbis	88.21 ± 1.90	88.49 ± 1.75	88.49 ± 1.77	86.41 ± 1.71	88.42 ± 1.81
la1s	90.93 ± 1.56	91.18 ± 1.60	91.07 ± 1.59	89.77 ± 1.63	91.41 ± 1.47
la2s	92.43 ± 1.43	92.44 ± 1.48	92.32 ± 1.49	91.60 ± 1.56	92.55 ± 1.58
new3s	91.59 ± 0.82	91.55 ± 0.87	91.59 ± 0.87	89.51 ± 0.96	91.86 ± 0.84
oh0	93.45 ± 2.45	94.13 ± 2.28	94.03 ± 2.34	92.89 ± 2.35	93.40 ± 2.44
oh10	86.86 ± 2.84	87.18 ± 2.70	87.21 ± 2.75	84.91 ± 3.10	87.08 ± 2.83
oh15	88.23 ± 2.98	88.56 ± 2.81	88.46 ± 2.98	87.87 ± 3.18	88.55 ± 3.23
oh5	93.94 ± 2.32	94.06 ± 2.20	94.21 ± 2.18	92.89 ± 2.63	93.61 ± 2.26
ohscal	81.22 ± 1.20	81.30 ± 1.27	81.29 ± 1.30	79.96 ± 1.09	82.11 ± 1.18
re0	87.53 ± 2.35	87.21 ± 2.44	87.24 ± 2.44	86.98 ± 2.30	88.07 ± 2.23
re1	89.52 ± 2.25	89.14 ± 2.22	89.32 ± 2.09	89.59 ± 1.90	89.76 ± 2.02
tr11	93.05 ± 3.72	93.77 ± 3.47	93.75 ± 3.47	91.45 ± 3.80	94.16 ± 3.15
tr12	91.95 ± 4.14	91.66 ± 4.12	92.01 ± 4.16	91.59 ± 5.03	91.20 ± 5.18
tr21	94.25 ± 3.56	94.28 ± 3.49	94.16 ± 3.65	94.37 ± 3.66	95.74 ± 3.24
tr23	95.26 ± 4.30	95.12 ± 4.76	95.26 ± 4.58	95.11 ± 4.34	96.96 ± 3.62
tr31	99.42 ± 0.79	99.31 ± 0.87	99.29 ± 0.87	98.84 ± 1.11	99.46 ± 0.73
tr41	98.50 ± 1.31	98.51 ± 1.36	98.54 ± 1.31	96.98 ± 1.80	98.01 ± 1.45
tr45	97.22 ± 1.87	97.22 ± 1.92	97.17 ± 1.85	96.25 ± 1.96	97.38 ± 1.79
wap	86.56 ± 2.41	85.85 ± 2.48	86.24 ± 2.46	86.99 ± 2.16	87.21 ± 2.27
Average	91.59	91.63	91.67	90.73	91.94

Table 15
Classification accuracy comparisons of Wilcoxon tests with regard to MTF-MDFS based on LR.

Algorithm	LogTF-RF$_{max}$	SqrtTF-IGM	SqrtTF-IGM$_{imp}$	TF-MDFS	MTF-MDFS
LogTF-RF$_{max}$	–			•	∘
SqrtTF-IGM		–		•	∘
SqrtTF-IGM$_{imp}$			–	•	
TF-MDFS	∘	∘	∘	–	∘
MTF-MDFS	•			•	–

of LogTF-RF$_{max}$ (90.94%), SqrtTF-IGM (91.01%) SqrtTF-IGM$_{imp}$ (91.05%) and TF-MDFS (89.73%).

3. In terms of LR, the averaged classification accuracy of MTF-MDFS on 19 datasets is 91.94%, which is comparatively higher than those of LogTF-RF$_{max}$ (91.59%), SqrtTF-IGM (91.63%) SqrtTF-IGM$_{imp}$ (91.67%) and TF-MDFS (90.73%).
4. According to the Wilcoxon signed-rank tests results, although TF-MDFS performs better than SqrtTF-IGM on MNB, its performance on SVM and LR is significantly worse than all the other competitors. This indicates that the classification performance of directly using the raw term frequency as a term frequency factor is limited.
5. According to the Wilcoxon signed-rank tests results, our MTF-MDFS significantly outperforms all the other competitors in terms of MNB and SVM, and is notably better than LogTF-RF$_{max}$ and TF-MDFS in terms of LR.
6. According to all above experimental results, we can roughly rank the performance of all these term weighting schemes as: MTF-MDFS >SqrtTF-IGM$_{imp}$ >SqrtTF-IGM >LogTF-RF$_{max}$ > TF-MDFS.

In order to further validate the effectiveness of our new proposed MTF and MTF-MDFS, we reevaluated the above experimental results from the perspective of weighted average of F_1. All the weighted average of F_1 estimates were obtained by averaging the results from 10 separate runs of stratified 10-fold cross-validation. The weighted average of F_1 is used to measure the overall performance of a algorithm in terms of classification performance of different classes in multiple types of problems, and its specific calculation formula is as follows:

$$F_1 = \frac{\sum_{j=1}^{q} F_{1j} d(c_j)}{n}, \tag{14}$$

where F_{1j} is the F_1 value when the jth class samples are taken as the positive samples and the rest as the negative samples. F_{1j} is calculated by the following formula:

$$F_{1j} = \frac{2 \cdot p_j \cdot r_j}{p_j + r_j}, \tag{15}$$

Table 16
Weighted average of F_1 comparisons for MTF versus its competitors based on MNB.

Dataset	TF	LogTF	SqrtTF	MTF
fbis	77.40 ± 2.48	77.87 ± 2.17	77.83 ± 2.16	79.46 ± 2.12
la1s	88.21 ± 1.63	88.36 ± 1.60	88.36 ± 1.55	88.97 ± 1.54
la2s	89.74 ± 1.58	89.81 ± 1.61	89.62 ± 1.64	90.55 ± 1.69
new3s	78.47 ± 1.17	78.31 ± 1.22	77.62 ± 1.28	80.12 ± 1.27
oh0	89.49 ± 2.84	89.70 ± 2.94	89.61 ± 2.99	89.98 ± 2.85
oh10	80.05 ± 3.29	81.21 ± 3.21	81.03 ± 3.31	81.37 ± 3.29
oh15	83.25 ± 3.25	83.41 ± 3.72	83.25 ± 3.85	83.17 ± 3.77
oh5	86.55 ± 3.08	85.86 ± 3.50	85.74 ± 3.62	85.77 ± 3.53
ohscal	74.61 ± 1.18	74.63 ± 1.28	74.50 ± 1.29	74.60 ± 1.31
re0	79.91 ± 2.89	79.14 ± 2.83	78.95 ± 2.72	79.66 ± 2.84
re1	81.01 ± 3.05	78.01 ± 2.85	77.04 ± 2.86	77.71 ± 2.81
tr11	83.81 ± 4.94	83.10 ± 5.11	82.61 ± 4.94	84.42 ± 4.87
tr12	79.76 ± 6.63	84.06 ± 6.02	83.11 ± 5.97	84.32 ± 5.76
tr21	63.03 ± 8.91	79.25 ± 6.47	78.80 ± 6.53	85.38 ± 5.55
tr23	72.13 ± 10.70	74.74 ± 10.35	74.89 ± 9.82	90.46 ± 6.00
tr31	94.42 ± 2.54	95.61 ± 2.55	95.62 ± 2.55	95.89 ± 2.46
tr41	94.44 ± 2.28	93.77 ± 2.15	93.70 ± 2.14	93.85 ± 2.03
tr45	82.97 ± 4.55	87.32 ± 3.45	86.86 ± 3.62	90.09 ± 3.14
wap	78.66 ± 2.83	76.42 ± 2.70	75.79 ± 2.76	76.26 ± 2.63
Average	82.00	83.19	82.89	84.84

Table 17
Weighted average of F_1 comparisons of Wilcoxon tests with regard to MTF based on MNB.

Algorithm	TF	LogTF	SqrtTF	MTF
TF	–			∘
LogTF		–	•	∘
SqrtTF		∘	–	∘
MTF		•	•	–

Table 18
Weighted average of F_1 comparisons for MTF versus its competitors based on SVM.

Dataset	TF	LogTF	SqrtTF	MTF
fbis	82.80 ± 2.01	86.40 ± 2.13	86.39 ± 2.05	86.24 ± 1.94
la1s	88.29 ± 1.84	89.30 ± 1.77	89.50 ± 1.82	90.35 ± 1.60
la2s	90.12 ± 1.43	91.09 ± 1.55	91.20 ± 1.48	91.99 ± 1.49
new3s	87.30 ± 1.04	90.22 ± 0.98	90.29 ± 0.95	90.38 ± 0.97
oh0	89.29 ± 2.64	91.26 ± 2.58	91.27 ± 2.63	91.58 ± 2.63
oh10	79.98 ± 3.76	83.52 ± 3.19	83.61 ± 3.30	83.75 ± 3.27
oh15	83.97 ± 3.77	85.63 ± 3.54	85.51 ± 3.38	86.17 ± 3.39
oh5	89.59 ± 2.92	91.36 ± 2.82	91.25 ± 2.85	91.82 ± 2.57
ohscal	75.66 ± 1.29	78.07 ± 1.26	78.10 ± 1.24	78.15 ± 1.23
re0	84.77 ± 2.45	86.26 ± 2.65	86.13 ± 2.58	86.21 ± 2.55
re1	85.16 ± 2.68	86.77 ± 2.48	86.79 ± 2.40	86.47 ± 2.47
tr11	88.48 ± 4.44	90.32 ± 4.30	90.15 ± 4.47	89.57 ± 4.47
tr12	88.24 ± 5.66	90.52 ± 5.69	91.23 ± 5.42	91.08 ± 5.33
tr21	90.38 ± 4.84	94.38 ± 3.78	93.99 ± 3.90	95.05 ± 3.73
tr23	89.71 ± 6.92	90.18 ± 6.72	90.14 ± 6.65	91.52 ± 6.72
tr31	98.27 ± 1.48	98.86 ± 1.33	98.78 ± 1.29	98.65 ± 1.36
tr41	96.32 ± 2.15	97.83 ± 1.46	97.88 ± 1.52	98.08 ± 1.38
tr45	94.02 ± 2.99	96.53 ± 2.29	96.31 ± 2.35	96.36 ± 2.31
wap	84.32 ± 2.54	84.53 ± 2.49	84.58 ± 2.45	85.35 ± 2.47
Average	87.72	89.63	89.64	89.93

Table 19
Weighted average of F_1 comparisons of Wilcoxon tests with regard to MTF based on SVM.

Algorithm	TF	LogTF	SqrtTF	MTF
TF	–	∘	∘	∘
LogTF	•	–		∘
SqrtTF	•		–	∘
MTF	•	•	•	–

Table 20
Weighted average of F_1 comparisons for MTF versus its competitors based on LR.

Dataset	TF	LogTF	SqrtTF	MTF
fbis	85.07 ± 1.92	87.44 ± 1.93	87.57 ± 2.00	87.51 ± 1.95
la1s	89.77 ± 1.70	90.92 ± 1.55	90.96 ± 1.55	91.34 ± 1.51
la2s	91.37 ± 1.47	92.11 ± 1.37	92.04 ± 1.43	92.49 ± 1.43
new3s	88.69 ± 1.01	91.07 ± 0.88	91.04 ± 0.88	90.93 ± 0.87
oh0	89.90 ± 2.56	91.58 ± 2.55	91.49 ± 2.59	91.64 ± 2.71
oh10	81.45 ± 3.65	84.13 ± 3.30	84.21 ± 3.50	84.37 ± 3.45
oh15	84.90 ± 3.58	86.15 ± 3.35	86.06 ± 3.38	86.02 ± 3.24
oh5	90.59 ± 3.01	92.01 ± 2.56	91.96 ± 2.45	92.04 ± 2.45
ohscal	78.25 ± 1.20	80.43 ± 1.25	80.50 ± 1.20	80.52 ± 1.19
re0	86.50 ± 2.36	87.10 ± 2.23	87.10 ± 2.22	87.70 ± 2.26
re1	86.77 ± 2.63	87.18 ± 2.45	87.12 ± 2.50	87.05 ± 2.46
tr11	88.32 ± 4.32	88.99 ± 4.19	88.87 ± 4.21	89.07 ± 4.19
tr12	89.06 ± 5.40	90.62 ± 5.27	91.19 ± 5.00	90.60 ± 5.05
tr21	90.55 ± 4.73	92.12 ± 4.26	91.87 ± 4.44	93.70 ± 3.75
tr23	89.43 ± 7.45	90.87 ± 6.62	90.19 ± 6.95	90.91 ± 6.84
tr31	98.03 ± 1.53	98.83 ± 1.34	98.77 ± 1.28	98.68 ± 1.34
tr41	96.28 ± 2.01	97.78 ± 1.59	97.86 ± 1.47	97.96 ± 1.54
tr45	94.00 ± 2.84	95.43 ± 2.27	95.36 ± 2.22	95.57 ± 2.27
wap	84.44 ± 2.57	84.83 ± 2.57	84.84 ± 2.45	85.39 ± 2.53
Average	88.60	89.98	89.95	90.18

Table 21
Weighted average of F_1 comparisons of Wilcoxon tests with regard to MTF based on LR.

Algorithm	TF	LogTF	SqrtTF	MTF
TF	–	∘	∘	∘
LogTF	•	–		∘
SqrtTF	•		–	∘
MTF	•	•	•	–

where p_j is the precision and r_j is the recall for the jth class.

Tables 16, 18 and 20 show the detailed weighted average of F_1 comparisons for MTF versus its competitors based on MNB, SVM and LR, respectively. Tables 17, 19 and 21 summarize the detailed comparison results of the Wilcoxon tests with regard to MTF based on MNB, SVM and LR, respectively. Tables 22, 24 and 26 show the detailed weighted average of F_1 comparisons for MTF-MDFS versus its competitors based on MNB, SVM and LR, respectively. Tables 23, 25 and 27 summarize the detailed comparison results of the Wilcoxon tests with regard to MTF-MDFS based on MNB, SVM and LR, respectively. From these comparison results, we can find that our proposed MTF and MTF-MDFS are still significantly better overall than their competitors. The experimental results on the weighted average of F_1 are basically consistent with the corresponding classification accuracy results. This shows that our proposed algorithms have good stability on different performance metrics.

4.2. Experiments on real-world text datasets

In order to further explore the effectiveness of our proposed MTF-MDFS on different types of datasets, we observe its performance on 6 different real-world text classification datasets, which include Amazon Commerce Reviews,[3] Movie Review,[4] WebKB,[5] 20 Newsgroups,[6] Reuters-21578[7] and RCV1.[8]

Amazon Commerce Reviews: This dataset is derived from the customers reviews in Amazon commerce website for 355 authorship identification and contains 1500 documents. The goal is to identify the 50 most active customers who frequently posted reviews in these newsgroups. The Amazon commerce reviews dataset has a small number of documents per class. The number of documents for each class is 30. The corpus has a vocabulary of 10000 words.

Movie Review: We choose the polarity dataset v2.0, which is a binary dataset containing 1000 positive and 1000 negative processed reviews. Stop-word removal and stemming (Porter, 1980) are carried out in the preprocessing phase. To save training time, rare words which occur less than 10 times in the dataset, numbers, punctuation marks and other non-alphabetic characters are removed. At the same time, the letters are converted to lower case. The resulting corpus has a vocabulary of 7103 words.

WebKB: This dataset contains web pages collected from computer science departments of various universities. We follow the common practice of using four classes: course, faculty, student and project. Among the selected 4199 documents, the number of documents contained in each class is 930, 1124, 1641 and 504, respectively. By preprocessing in the same way as above for the Movie Review dataset, the resulting corpus has a vocabulary of 8791 words.

20 Newsgroups: This dataset contains 19997 documents of newsgroup messages, which are divided into 20 classes. Except for 997 documents in one class, there are 1000 documents in each of the remaining 19 classes. By preprocessing in the same way as above for the Movie Review dataset, the resulting corpus has a vocabulary of 20746 words.

[3] http://archive.ics.uci.edu/ml/datasets/Amazon+Commerce+reviews+set
[4] http://www.cs.cornell.edu/people/pabo/movie-review-data/
[5] http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/
[6] http://archive.ics.uci.edu/ml/datasets/Twenty+Newsgroups
[7] http://archive.ics.uci.edu/ml/datasets/Reuters-21578+Text+Categorization+Collection
[8] http://archive.ics.uci.edu/ml/datasets/Reuters+RCV1+RCV2+Multilingual,+Multiview+Text+Categorization+Test+collection#

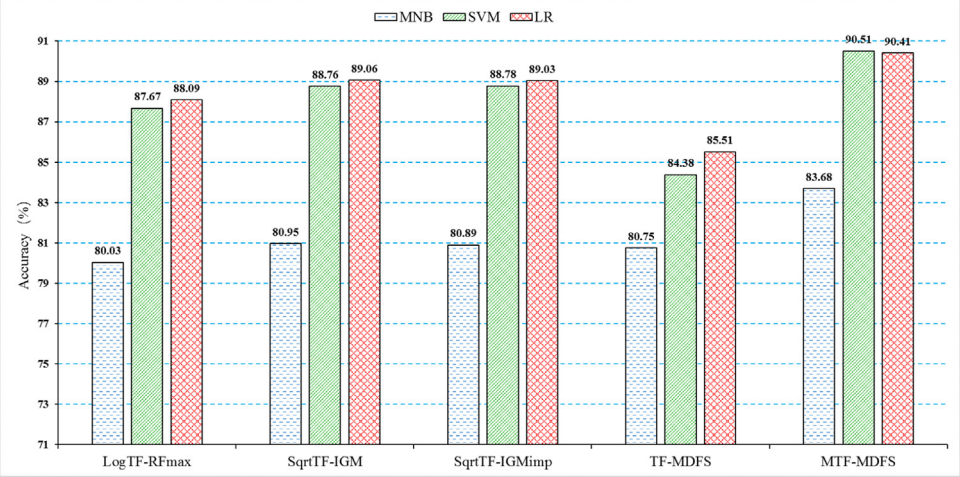


Fig. 1. Classification accuracy comparisons on the Amazon Commerce Reviews dataset.

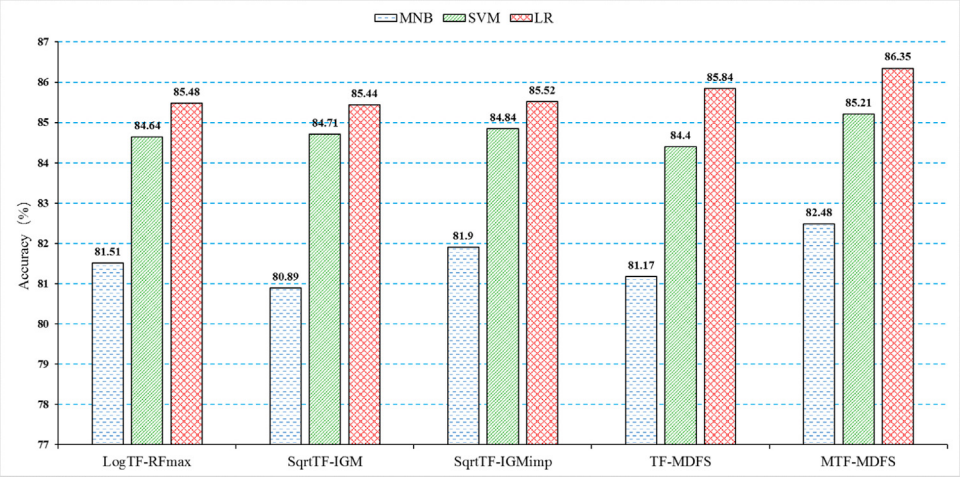


Fig. 2. Classification accuracy comparisons on the Movie Review dataset.

Table 22
Weighted average of F_1 comparisons for MTF-MDFS versus its competitors based on MNB.

Dataset	LogTF-RF$_{max}$	SqrtTF-IGM	SqrtTF-IGM$_{imp}$	TF-MDFS	MTF-MDFS
fbis	79.46 ± 2.32	79.52 ± 2.36	79.52 ± 2.36	80.35 ± 2.49	82.00 ± 2.18
la1s	89.60 ± 1.66	89.88 ± 1.71	89.83 ± 1.63	88.64 ± 1.67	89.40 ± 1.67
la2s	90.72 ± 1.44	90.60 ± 1.57	90.77 ± 1.50	90.10 ± 1.53	90.91 ± 1.66
new3s	81.15 ± 1.31	82.04 ± 1.28	81.93 ± 1.30	82.19 ± 1.30	84.34 ± 1.22
oh0	93.86 ± 2.32	93.45 ± 2.47	93.72 ± 2.34	93.59 ± 2.20	94.49 ± 2.11
oh10	84.71 ± 3.01	84.13 ± 3.16	84.48 ± 2.93	84.42 ± 2.98	85.87 ± 3.11
oh15	87.01 ± 3.26	86.66 ± 3.24	86.97 ± 3.14	88.22 ± 2.68	88.26 ± 2.87
oh5	91.44 ± 2.70	91.19 ± 2.70	91.45 ± 2.79	93.04 ± 2.78	93.52 ± 2.50
ohscal	76.88 ± 1.24	76.49 ± 1.19	77.10 ± 1.21	79.24 ± 1.18	80.01 ± 1.26
re0	79.70 ± 2.89	78.80 ± 3.12	79.28 ± 3.01	82.82 ± 2.89	82.35 ± 2.73
re1	83.83 ± 2.31	84.10 ± 2.47	84.62 ± 2.37	87.57 ± 2.20	86.77 ± 2.18
tr11	85.64 ± 4.62	86.48 ± 4.45	86.03 ± 4.46	87.56 ± 4.58	88.80 ± 4.84
tr12	88.55 ± 5.40	87.13 ± 5.77	87.74 ± 5.53	86.53 ± 5.97	89.70 ± 4.49
tr21	81.67 ± 5.72	84.18 ± 5.32	83.26 ± 5.63	82.60 ± 5.81	91.45 ± 4.28
tr23	84.28 ± 7.79	81.32 ± 8.25	81.66 ± 8.01	90.51 ± 6.49	94.72 ± 4.87
tr31	96.91 ± 2.01	97.29 ± 1.87	97.29 ± 1.95	97.72 ± 1.87	97.79 ± 1.72
tr41	95.21 ± 2.15	94.87 ± 2.29	95.20 ± 2.24	95.95 ± 2.04	96.46 ± 1.74
tr45	91.63 ± 3.05	92.56 ± 2.61	92.62 ± 2.67	92.69 ± 3.30	94.19 ± 2.33
wap	80.62 ± 2.57	80.59 ± 2.62	81.06 ± 2.70	83.24 ± 2.68	82.98 ± 2.50
Average	86.47	86.38	86.55	87.74	89.16

Reuters-21578: This is a collection of documents that appeared on Reuters newswire. The top-10 largest classes of the celebrated Reuters-21578 were selected in our experiment. Among the reduced 9980 documents, the largest class (earn) contains 3964 documents and the smallest class (corn) contains 237 documents. By preprocessing in the same way as above for the Movie Review dataset, the resulting corpus has a vocabulary of 4854 words.

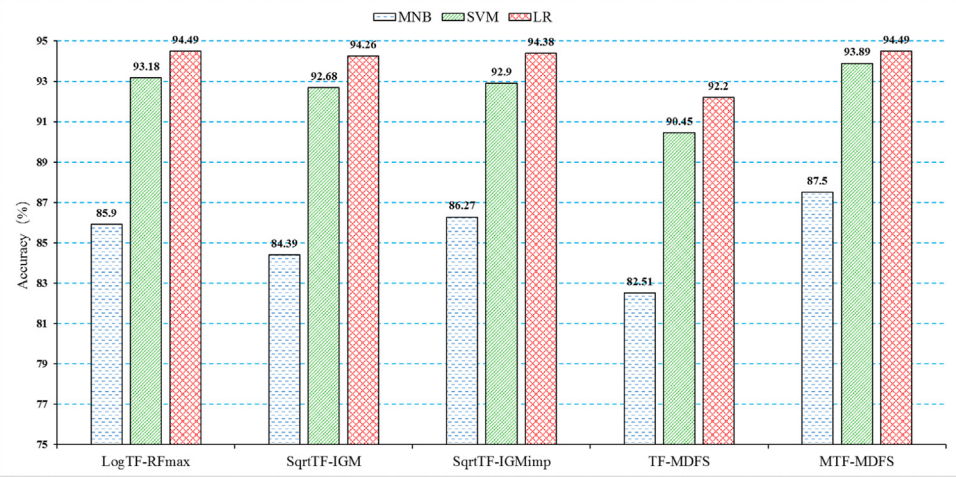


Fig. 3. Classification accuracy comparisons on the WebKB dataset.

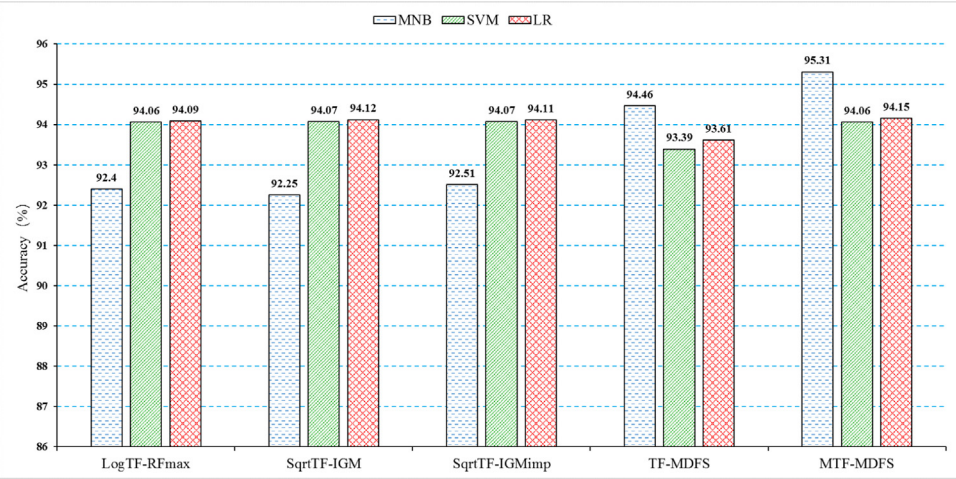


Fig. 4. Classification accuracy comparisons on the 20 Newsgroups dataset.

Table 23
Weighted average of F_1 comparisons of Wilcoxon tests with regard to MTF-MDFS based on MNB.

Algorithm	LogTF-RF$_{max}$	SqrtTF-IGM	SqrtTF-IGM$_{imp}$	TF-MDFS	MTF-MDFS
LogTF-RF$_{max}$	–			∘	∘
SqrtTF-IGM		–	∘	∘	∘
SqrtTF-IGM$_{imp}$		•	–	∘	∘
TF-MDFS	•	•	•	–	∘
MTF-MDFS	•	•	•	•	–

RCV1: This dataset contains feature characteristics of documents originally written in five different languages and their translations. We only choose the original English documents containing 18758 documents in our experiment. This dataset contains 6 classes: C15, CCAT, E21, ECAT, GCAT and M11. The number of documents for each class is 5102, 4331, 1234, 2055, 4829 and 1207, respectively. After removing a word with negative values, the corpus has a vocabulary of 21530 words.

Table 28 summarizes the characteristics of these 6 datasets. The same as the experiments on benchmark text datasets, all comparison results were also obtained by averaging the results from 10 separate runs of stratified 10-fold cross-validation. Figs. 1–6 show the detailed classification accuracy comparison results based on MNB, SVM and LR, respectively. Figs. 7–12 show the detailed weighted average of F_1 comparison results based on MNB, SVM and LR, respectively. From these comparisons, we can see that:

1. For the classification accuracy and the weighted average of F_1, the comparison results of MTF-MDFS on the real-world text datasets are still consistent.
2. From the experimental results on the Amazon Commerce Reviews, Movie Review and WebKB datasets, MTF-MDFS is significantly better than all the other existing competitors.
3. From the experimental results on the 20 Newsgroups dataset, MTF-MDFS significantly outperforms all the other competitors in terms of MNB, and is comparable to LogTF-RF$_{max}$, SqrtTF-IGM and SqrtTF-IGM$_{imp}$ in terms of SVM and LR.
4. From the experimental results on the Reuters-21578 and RCV1 datasets, the performance of MTF-MDFS on MNB is not outstanding, but the performance of MTF-MDFS on SVM and LR is better than all the other competitors.
5. From the above comparison results, the performance of MTF-MDFS on MNB varies greatly with respect to different types of datasets, while the performance of MTF-MDFS on SVM and LR is more stable.
6. In almost all cases, the performance of MTF-MDFS is better than that of TF-MDFS. This shows that an appropriate term frequency factor is very important for improving the performance of TC.

Table 24
Weighted average of F_1 comparisons for MTF-MDFS versus its competitors based on SVM.

Dataset	LogTF-RF$_{max}$	SqrtTF-IGM	SqrtTF-IGM$_{imp}$	TF-MDFS	MTF-MDFS
fbis	86.39 ± 2.04	86.62 ± 1.97	86.61 ± 1.94	84.26 ± 1.71	86.99 ± 1.89
la1s	89.62 ± 1.72	89.99 ± 1.72	89.74 ± 1.66	88.29 ± 1.73	90.81 ± 1.67
la2s	91.35 ± 1.58	91.43 ± 1.62	91.43 ± 1.58	90.07 ± 1.68	92.10 ± 1.61
new3s	90.80 ± 0.91	90.84 ± 0.95	90.83 ± 0.97	88.29 ± 1.01	91.33 ± 0.86
oh0	92.97 ± 2.50	93.71 ± 2.30	93.54 ± 2.30	92.22 ± 2.49	93.00 ± 2.53
oh10	85.63 ± 3.34	85.78 ± 2.94	85.77 ± 3.11	82.99 ± 3.50	85.54 ± 3.24
oh15	87.39 ± 3.06	87.71 ± 3.15	87.81 ± 3.00	86.44 ± 3.34	87.82 ± 3.18
oh5	93.32 ± 2.56	93.66 ± 2.45	93.66 ± 2.48	92.16 ± 2.82	93.56 ± 2.35
ohscal	79.08 ± 1.25	79.35 ± 1.35	79.39 ± 1.34	77.30 ± 1.21	79.84 ± 1.38
re0	85.62 ± 2.76	84.98 ± 2.76	85.12 ± 2.80	84.77 ± 2.59	86.32 ± 2.44
re1	86.89 ± 2.56	86.55 ± 2.68	86.66 ± 2.56	86.67 ± 2.26	87.87 ± 2.22
tr11	92.40 ± 4.10	93.43 ± 3.87	93.51 ± 3.86	90.63 ± 3.87	93.92 ± 3.64
tr12	91.85 ± 5.09	91.46 ± 4.82	91.60 ± 4.80	90.75 ± 5.49	91.09 ± 5.74
tr21	94.06 ± 4.32	93.80 ± 4.21	93.96 ± 4.25	93.38 ± 4.83	95.72 ± 3.61
tr23	94.28 ± 5.05	94.77 ± 5.13	94.83 ± 5.10	93.97 ± 5.34	95.79 ± 4.74
tr31	99.23 ± 0.93	99.26 ± 0.95	99.24 ± 0.99	98.60 ± 1.28	99.21 ± 0.96
tr41	98.56 ± 1.30	98.71 ± 1.26	98.69 ± 1.29	97.15 ± 1.89	97.95 ± 1.58
tr45	97.22 ± 1.96	97.01 ± 2.13	96.99 ± 2.03	96.29 ± 2.33	97.28 ± 1.86
wap	85.27 ± 2.41	84.88 ± 2.43	85.20 ± 2.56	85.32 ± 2.47	86.21 ± 2.63
Average	90.63	90.73	90.77	89.45	91.18

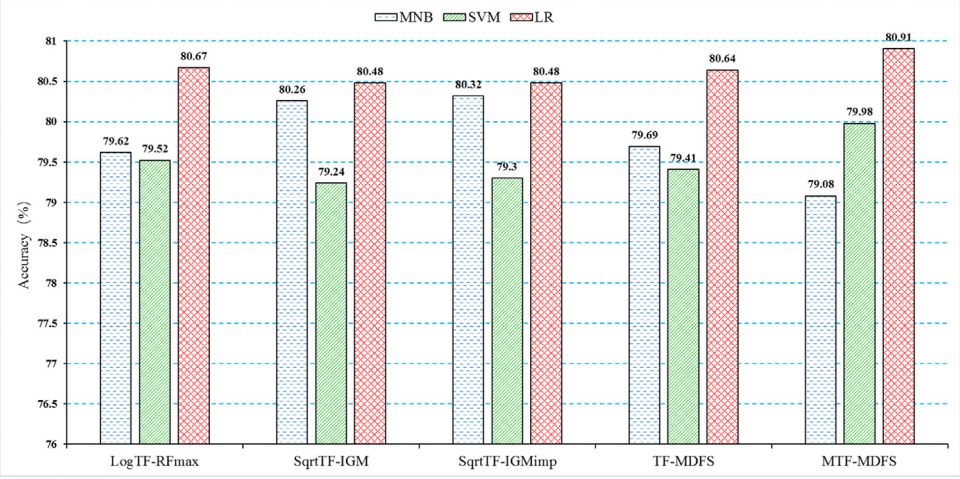


Fig. 5. Classification accuracy comparisons on the Reuters-21578 dataset.

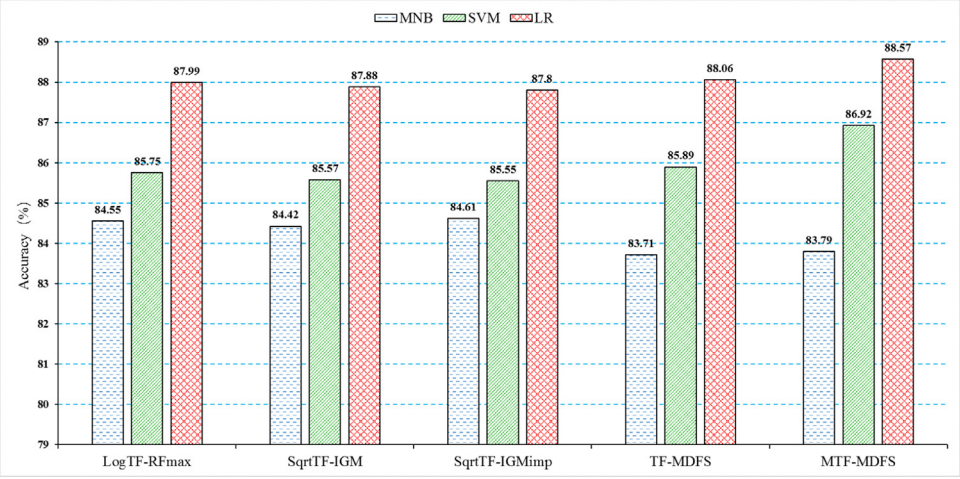


Fig. 6. Classification accuracy comparisons on the RCV1 dataset.

7. Based on all of these comparison results, we can see that our MTF-MDFS is overall the best one among all the term weighting schemes used to compare.

5. Conclusions and future work

In this paper, we conducted a comprehensive survey on the existing well-known term weighting schemes and found that most of

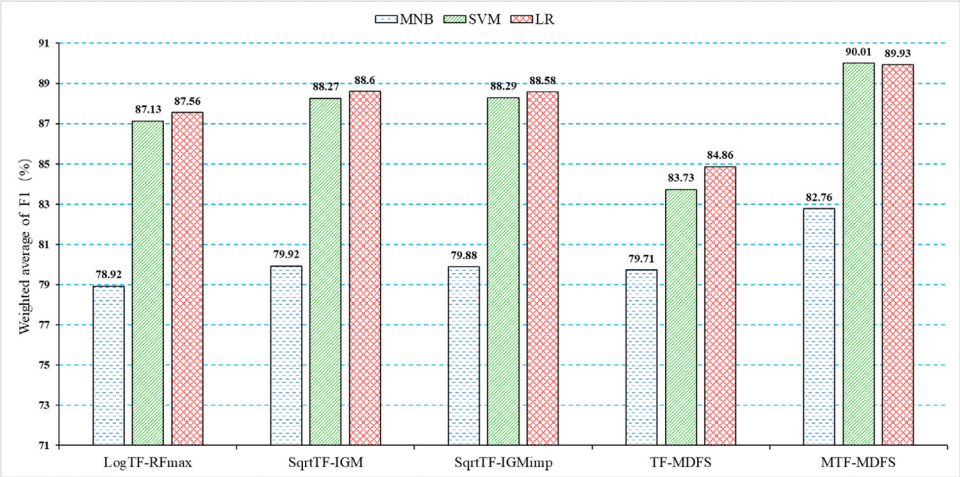


Fig. 7. Weighted average of F_1 comparisons on the Amazon Commerce Reviews dataset.

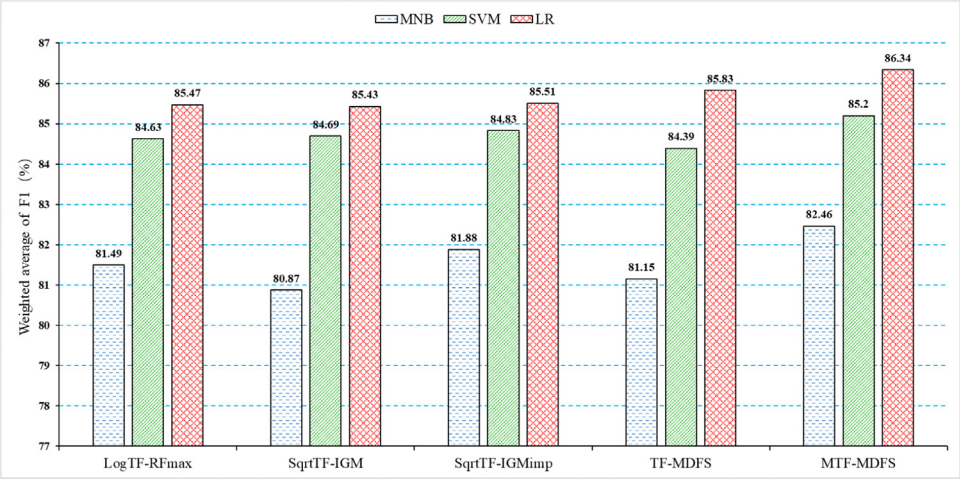


Fig. 8. Weighted average of F_1 comparisons on the Movie Review dataset.

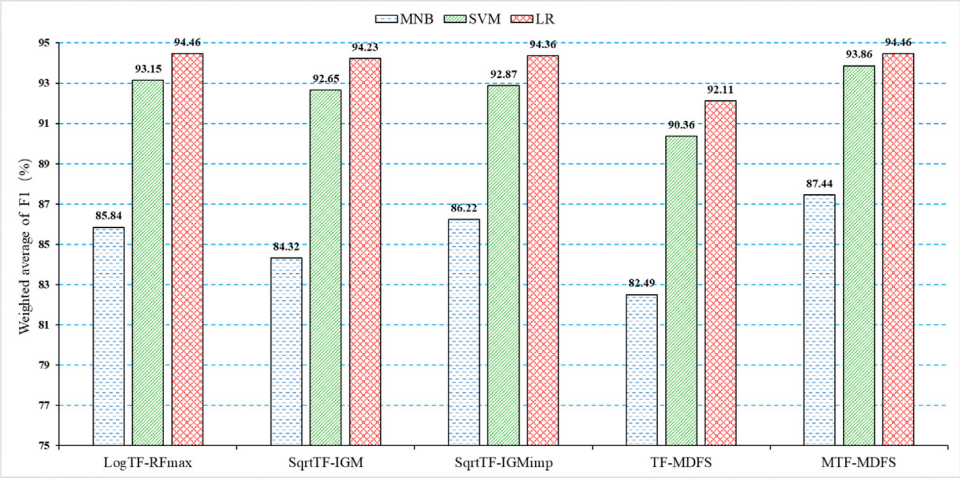


Fig. 9. Weighted average of F_1 comparisons on the WebKB dataset.

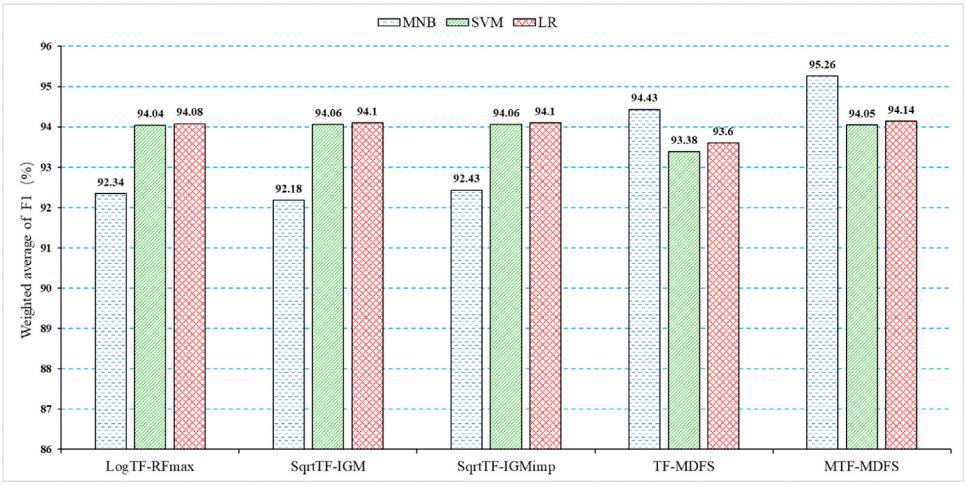


Fig. 10. Weighted average of F_1 comparisons on the 20 Newsgroups dataset.

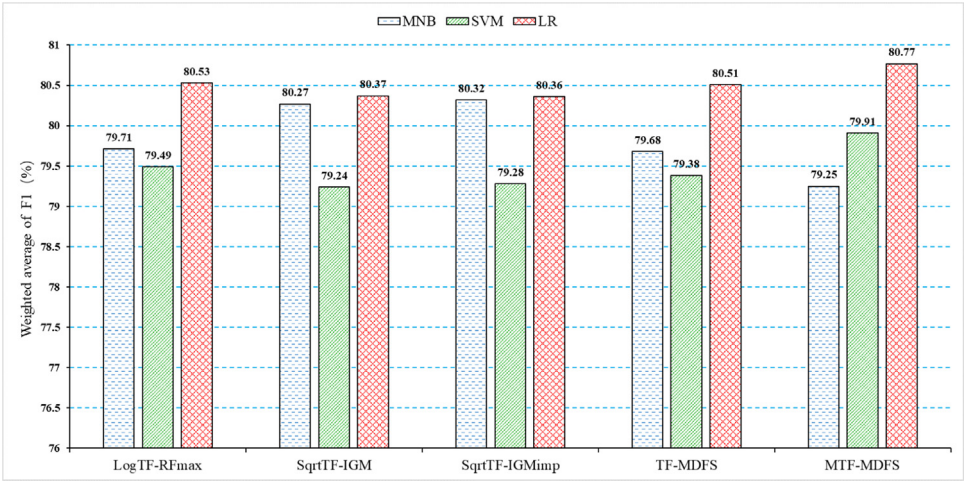


Fig. 11. Weighted average of F_1 comparisons on the Reuters-21578 dataset.

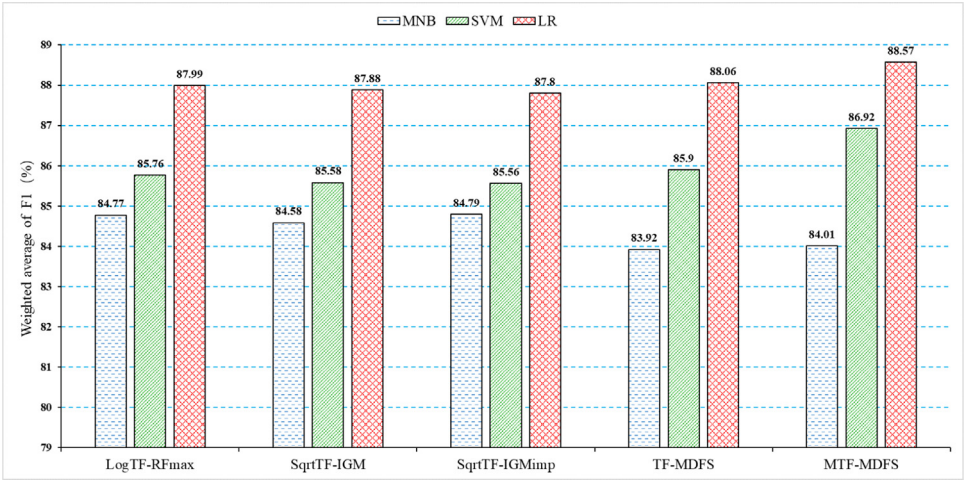


Fig. 12. Weighted average of F_1 comparisons on the RCV1 dataset.

Table 25
Weighted average of F_1 comparisons of Wilcoxon tests with regard to MTF-MDFS based on SVM.

Algorithm	LogTF-RF$_{max}$	SqrtTF-IGM	SqrtTF-IGM$_{imp}$	TF-MDFS	MTF-MDFS
LogTF-RF$_{max}$	–			•	○
SqrtTF-IGM		–		•	○
SqrtTF-IGM$_{imp}$			–	•	○
TF-MDFS	○	○	○	–	○
MTF-MDFS	•	•	•	•	–

Table 27
Weighted average of F_1 comparisons of Wilcoxon tests with regard to MTF-MDFS based on LR.

Algorithm	LogTF-RF$_{max}$	SqrtTF-IGM	SqrtTF-IGM$_{imp}$	TF-MDFS	MTF-MDFS
LogTF-RF$_{max}$	–			•	○
SqrtTF-IGM		–		•	○
SqrtTF-IGM$_{imp}$			–	•	○
TF-MDFS	○	○	○	–	○
MTF-MDFS	•			•	–

Table 28
The real-world datasets used in our experiments.

Dataset	#Documents	#Words	#Classes
Amazon commerce reviews	1500	10000	50
Movie review	2000	7103	2
WebKB	4199	8791	4
20 Newsgroups	19997	20746	20
Reuters-21578	9980	4854	10
RCV1	18758	21530	6

them focus on finding a more effective collection frequency factor, but rarely pay attention to finding a new term frequency factor. We argue that an appropriate term frequency factor can also bring significant improvements to the performance of TC. Based on this premise, we introduced the length information of all training documents into the TF factor and proposed a new term frequency factor called modified term frequency (MTF). Then we combined MTF with an existing collection frequency factor called modified distinguishing feature selector (MDFS) and proposed a new term weighting scheme. We denoted our scheme by MTF-MDFS (MDFS-based MTF). The extensive comparison results validate the effectiveness of our proposed MTF and MTF-MDFS in terms of the classification performance of widely used base classifiers.

The term frequency factor and the collection frequency factor are equally important for term weighting, both of which are helpful to improve the performance of TC. Currently, we simply combined MTF and MDFS without considering whether they maybe counteract each other to generate inappropriate weights. For example, when the length difference of training documents is particularly large, there is likely to be a situation that the term frequency factor plays a dominant role in term weighting and the role of the collection frequency factor is greatly reduced. On the other hand, when the specificity scores of terms for classes varies too much, it may also cause adverse effects. We believe that the use of more sophisticated combination methods could improve the performance of the current MTF-MDFS and make its advantage stronger. This will be a major direction for our future work. In addition, applying more sophisticated optimization methods (Gong and Cai, 2013; Yan et al., 2018; Lu et al., 2018; Gong et al., 2020), to directly search term weights is another interesting topic for future work.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The work was partially supported by the National Natural Science Foundation of China (U1711267), the Fundamental Research Funds for the Central Universities (CUGGC03), and the foundation of Key Laboratory of Artificial Intelligence, Ministry of Education, P.R. China (AI2020002).

Table 26
Weighted average of F_1 comparisons for MTF-MDFS versus its competitors based on LR.

Dataset	LogTF-RF$_{max}$	SqrtTF-IGM	SqrtTF-IGM$_{imp}$	TF-MDFS	MTF-MDFS
fbis	87.75 ± 2.04	88.06 ± 1.86	88.06 ± 1.89	86.15 ± 1.76	87.99 ± 1.89
la1s	90.88 ± 1.57	91.14 ± 1.60	91.02 ± 1.59	89.73 ± 1.63	91.36 ± 1.48
la2s	92.40 ± 1.44	92.41 ± 1.49	92.29 ± 1.50	91.58 ± 1.55	92.51 ± 1.59
new3s	91.52 ± 0.83	91.47 ± 0.89	91.52 ± 0.89	89.45 ± 0.97	91.77 ± 0.85
oh0	93.38 ± 2.47	94.07 ± 2.32	93.96 ± 2.38	92.84 ± 2.36	93.35 ± 2.45
oh10	86.47 ± 3.07	86.83 ± 2.89	86.85 ± 2.94	84.52 ± 3.33	86.75 ± 3.02
oh15	88.04 ± 3.05	88.39 ± 2.86	88.29 ± 3.03	87.66 ± 3.24	88.38 ± 3.27
oh5	93.91 ± 2.32	94.04 ± 2.21	94.17 ± 2.21	92.82 ± 2.67	93.56 ± 2.29
ohscal	81.22 ± 1.19	81.30 ± 1.26	81.29 ± 1.28	79.95 ± 1.09	82.12 ± 1.16
re0	87.14 ± 2.48	86.86 ± 2.52	86.88 ± 2.54	86.62 ± 2.44	87.64 ± 2.39
re1	88.44 ± 2.39	88.22 ± 2.24	88.35 ± 2.12	88.81 ± 2.01	88.76 ± 2.13
tr11	92.02 ± 4.30	92.90 ± 3.89	92.88 ± 3.97	90.93 ± 3.94	93.69 ± 3.49
tr12	91.20 ± 4.55	90.89 ± 4.45	91.24 ± 4.53	90.98 ± 5.46	90.42 ± 5.67
tr21	92.98 ± 4.38	93.18 ± 4.17	93.05 ± 4.35	93.34 ± 4.46	94.94 ± 3.95
tr23	94.70 ± 4.76	94.67 ± 5.21	94.85 ± 5.02	94.27 ± 5.15	96.67 ± 4.09
tr31	99.31 ± 0.94	99.20 ± 1.03	99.17 ± 1.02	98.72 ± 1.24	99.35 ± 0.88
tr41	98.28 ± 1.46	98.33 ± 1.48	98.37 ± 1.43	96.78 ± 1.92	97.85 ± 1.54
tr45	96.96 ± 2.04	96.94 ± 2.09	96.90 ± 2.01	95.91 ± 2.15	97.19 ± 1.89
wap	85.23 ± 2.60	84.51 ± 2.68	84.92 ± 2.68	86.08 ± 2.26	85.97 ± 2.48
Average	91.15	91.23	91.27	90.38	91.59

References

Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., 2011. KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. Mult.-Valued Logic Soft Comput. 17 (2–3), 255–287.

Amati, G., van Rijsbergen, C.J., 2002. Probabilistic models of information retrieval based on measuring the divergence from randomness. ACM Trans. Inf. Syst. 20 (4), 357–389.

Aseervatham, S., Gaussier, É., Antoniadis, A., Burlet, M., Denneulin, Y., 2012. Logistic regression and text classification. In: Textual Information Access: Statistical Models. pp. 61–84.

Chen, L., Jiang, L., Li, C., 2021. Modified DFS-based term weighting scheme for text classification. Expert Syst. Appl. 168, 114438.

Chen, K., Zhang, Z., Long, J., Zhang, H., 2016. Turning from TF-IDF to TF-IGM for term weighting in text classification. Expert Syst. Appl. 66, 245–260.

Cortes, C., Vapnik, V., 1995. Support-vector networks. Mach. Learn. 20 (3), 273–297.

Debole, F., Sebastiani, F., 2003. Supervised term weighting for automated text categorization. In: Proceedings of the 2003 ACM Symposium on Applied Computing. SAC 2003, Melbourne, FL, USA, March 9–12. pp. 784–788.

Demsar, J., 2006. Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. 7, 1–30.

Dogan, T., Uysal, A.K., 2019. Improved inverse gravity moment term weighting for text classification. Expert Syst. Appl. 130, 45–59.

Fan, R., Chang, K., Hsieh, C., Wang, X., Lin, C., 2008. LIBLINEAR: A library for large linear classification. J. Mach. Learn. Res. 9, 1871–1874.

Gangavarapu, T., Jaidhar, C.D., Chanduka, B., 2020. Applicability of machine learning in spam and phishing email filtering: review and approaches. Artif. Intell. Rev. 53 (7), 5019–5081.

Garcia, S., Herrera, F., 2008. An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. J. Mach. Learn. Res. 9, 2677–2694.

Gong, W., Cai, Z., 2013. Parameter extraction of solar cell models using repaired adaptive differential evolution. Sol. Energy 94, 209–220.

Gong, W., Wang, Y., Cai, Z., Wang, L., 2020. Finding multiple roots of nonlinear equation systems via a repulsion-based adaptive differential evolution. IEEE Trans. Syst. Man Cybern. A 50 (4), 1499–1513.

Han, E., Karypis, G., 2000. Centroid-based document classification: Analysis and experimental results. In: Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery. PKDD 2000, Lyon, France, September 13–16. pp. 424–431.

Hassonah, M.A., Al-Sayyed, R.M.H., Rodan, A., Al-Zoubi, A.M., Aljarah, I., Faris, H., 2020. An efficient hybrid filter and evolutionary wrapper approach for sentiment analysis of various topics on Twitter. Knowl. Based Syst. 192, 105353.

Jiang, L., Cai, Z., Zhang, H., Wang, D., 2013. Naive Bayes text classifiers: a locally weighted learning approach. J. Exp. Theor. Artif. Intell. 25 (2), 273–286.

Jiang, L., Li, C., Wang, S., Zhang, L., 2016a. Deep feature weighting for naive Bayes and its application to text classification. Eng. Appl. Artif. Intell. 52, 26–39.

Jiang, L., Wang, S., Li, C., Zhang, L., 2016b. Structure extended multinomial naive Bayes. Inform. Sci. 329, 346–356.

Jiang, L., Zhang, L., Li, C., Wu, J., 2019a. A correlation-based feature weighting filter for naive Bayes. IEEE Trans. Knowl. Data Eng. 31 (2), 201–213.

Jiang, L., Zhang, L., Yu, L., Wang, D., 2019b. Class-specific attribute weighted naive Bayes. Pattern Recognit. 88, 321–330.

Lan, M., Tan, C.L., 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.

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

Lu, C., Gao, L., Yi, J., 2018. Grey wolf optimizer with cellular topological structure. Expert Syst. Appl. 107, 89–114.

McCallum, A., Nigam, K., 1998. A comparison of event models for naive bayes text classification. In: AAAI-98 Workshop on Learning for Text Categorization, Vol. 752, No. 1. pp. 41–48.

Porter, M.F., 1980. An algorithm for suffix stripping. Program 14 (3), 130–137.

Ren, F., Sohrab, M.G., 2013. Class-indexing-based term weighting for automatic text classification. Inform. Sci. 236, 109–125.

Salton, G., Buckley, C., 1988. Term-weighting approaches in automatic text retrieval. Inf. Process. Manage. 24 (5), 513–523.

Salton, G., McGill, M., 1984. Introduction to Modern Information Retrieval. McGraw-Hill Book Company.

Silva, R.M., Santos, R.L.S., Almeida, T.A., Pardo, T.A.S., 2020. Towards automatically filtering fake news in portuguese. Expert Syst. Appl. 146, 113199.

Uysal, A.K., Günal, S., 2012. A novel probabilistic feature selection method for text classification. Knowl.-Based Syst. 36, 226–235.

Wang, T., Cai, Y., Leung, H., Cai, Z., Min, H., Entropy-based term weighting schemes for text categorization in VSM. In: Proceedings of the 27th IEEE International Conference on Tools with Artificial Intelligence. ICTAI 2015, Vietri Sul Mare, Italy, November 9–11. pp. 325–332.

Wang, S., Jiang, L., Li, C., 2015a. Adapting naive Bayes tree for text classification. Knowl. Inf. Syst. 44 (1), 77–89.

Wang, D., Zhang, H., 2013. Inverse-category-frequency based supervised term weighting schemes for text categorization. J. Inf. Sci. Eng. 29 (2), 209–225.

Witten, I.H., Frank, E., Hall, M.A., 2011. Data Mining: Practical Machine Learning Tools and Techniques, third ed. Morgan Kaufmann, Elsevier.

Xuan, N.P., Quang, H.L., 2013. A new improved term weighting scheme for text categorization. In: Proceedings of the 5th International Conference on Knowledgeand Systems Engineering. KSE 2013, Hanoi, Vietnam, October 17–19. pp. 261–270.

Yan, X., Yang, K., Hu, C., Gong, W., 2018. Pollution source positioning in a water supply network based on expensive optimization. Desalin. Water Treat. 110, 308–318.

Zhang, L., Jiang, L., Li, C., Kong, G., 2016. Two feature weighting approaches for naive Bayes text classifiers. Knowl. Based Syst. 100, 137–144.

Zhang, H., Jiang, L., Yu, L., 2020. Class-specific attribute value weighting for Naive Bayes. Inform. Sci. 508, 260–274.

Zhang, T., Oles, F.J., 2001. Text categorization based on regularized linear classification methods. Inf. Retr. 4 (1), 5–31.