A New Improved Term Weighting Scheme for Text Categorization

Nguyen Pham Xuan and Hieu Le Quang

Abstract. In text categorization, term weighting is the task to assign weights to terms during the document presentation phase. Thus, it affects the classification performance. In this paper, we propose a new term weighting scheme $logtf.rf_{max}$. It is an improvement to *tf.rf* – one of the most effective term weighting schemes to date. We conducted experiments to compare the new term weighting scheme to *tf.rf* and others on common text categorization benchmark data sets. The experimental results show that $logtf.rf_{max}$ consistently outperforms *tf.rf* as well as other schemes. Furthermore, our new scheme is simpler than *tf.rf*.

1 Introduction

The task of text categorization is to classify documents into predefined categories. Text categorization has been studied extensively in recent years [12]. In the vector model, each document is presented as a vector of terms. Each vector component contains a value presenting how much the term contributes to the discriminative semantics of the document. The goal of a term weighting method is to assign appropriate weights to terms in order to achieve high classification performance.

Term weighting methods can be divided into two categories, namely, supervised term weighting method and unsupervised term weighting method [8]. The traditional term weighting methods such as *binary*, *tf*, *tf.idf* [11], belong to the unsupervised term weighting methods. The other term weighting methods (for example, $tf.\chi^2$ [3]), that make use of the prior information about the membership of training documents in predefined categories, belong to the supervised term weighting methods.

Nguyen Pham Xuan · Hieu Le Quang
Faculty of Information Technology, VNU University of Engineering and Technology, Vietnam
e-mail: {nguyenpx.mi10,hieulq}@vnu.edu.vn

V.-N. Huynh et al. (eds.), *Knowledge and Systems Engineering, Volume 1*,
Advances in Intelligent Systems and Computing 244,
DOI: 10.1007/978-3-319-02741-8_23,

The supervised term weighting method *tf.rf* [8] showed consistently better performance than many other term weighting methods in experiments using SVM and kNN, two of the most commonly-used algorithms for text categorization [14]. The OneVsAll approach used in [8] transforms the multi-label classification problem of N categories into N binary classification problems, each of which associates with a different category. For each term, *tf.rf* requires N *rf* values, each for a binary classification problem.

In this paper we present a new term weighting scheme $logtf.rf_{max}$, an improvement to *tf.rf*. Our scheme requires a single *rf* value for each term for a multi-label classification problem. Moreover, it uses $logtf = log_2(1.0 + tf)$ instead of *tf*. Our experimental results show that our scheme is consistently better than *tf.rf* and others.

The rest of paper is organized as follows. Section 2 reviews related works. Our new improved term weighting method is described in Section 3. Section 4 and Section 5 report our experimental settings and results, as well as our disscusion. Finally, we draw conclusions in Section 6.

2 Related Works

In this section, we give a brief overview of the existing term weighting methods. We also describe the feature selection method CHI_{max} which relates to an improvement in our new term weighting scheme.

2.1 Traditional Term Weighting Methods

Generally, the traditional term weighting methods are rooted from the information retrieval field and they belong to the unsupervised term weighting methods. The simplest *binary* term weighting method assigns 1 to all terms in a document in the vector representation phase. The most widely-used term weighting approaches in this group is *tf.idf*. *tf* is the frequency of a term in a document and *idf* is the inverse document frequency of a term. *tf* has various variants which use the logarithm operation such as *log(tf)*, *log(1 + tf)*, *1 + log(tf)* [9]. The goal of logarithm operation is to scale down the effect of noisy terms.

2.2 Supervised Term Weighting Methods

The supervised term weighting methods are the ones that make use of the prior information about the membership of training documents in predefined categories to assign weights to terms. One way to use this known information is to combine *tf* and a feature selection metric such as χ^2, Information Gain, Gain Ratio [3], [2].

tf.rf is the supervised term weighting method that combines *tf* and *rf* (relevance frequency) factor which proposed by Lan et al. [8]. As said in Introduction, OneVsAll transforms a multi-label classification problem into N binary classification problems, each relates to a category which is tagged as the positive category

and all other categories in the training set are grouped into the negative category. Each term t requires one rf value in each category C_i, and this value is computed as follows:

$$rf(C_i) = \log_2(2 + \frac{a}{c}). \quad (1)$$

where, a is the number of documents in category C_i which contains t and c is the number of documents not in category C_i which contains t. The purpose of rf is to give more weight to terms which help classify documents into the positive category.

2.3 *Feature Selection Method* CHI_{max}

CHI is a feature selection method using the χ^2 statistic, which is used to compute score of a term t in a category C_i as bellow:

$$\chi^2(t, C_i) = N * \frac{(a*d - b*c)^2}{(a+c)*(b+d)*(a+b)*(c+d)}. \quad (2)$$

a is the number of documents in category C_i which contain t, b is the number of documents in category C_i which do not contain t, c is the number of documents not in category C_i which contains t, d is the number of documents not in category C_i which do not contain t, N is the total number of documents in the training set.

For the multi-label classification problem which is transformed by the OneVsAll method, each feature is assigned a score in each category as described in 2. Then all these scores are combined into a single final score base on a function such as *max* or *average*. Finally, all features are sorted by final scores in descending order and top p highest score features are selected. The levels of p are optional. CHI_{max} and CHI_{avg} are two CHI feature selections that use *max* and *average* to combine all scores, respectively. Following [10], CHI_{max} performs better than CHI_{avg}.

3 Our New Improved Term Weighting Method

As said before, for each term in a multi-label classification problem, *tf.rf* uses N rf values, each of them for a different binary classifier. Meanwhile, our scheme, which also uses OneVsAll method, assigns a single rf_{max}(maximum of all rf) for each term for all binary classiffiers. By this way, the weight of a term is now corresponded to the category that this term repesents the most. This approach is similar to the one used in the CHI_{max} feature selection method described in Section 2. Our experimental results (see Section 5) show that this improvement helps to increase classification performance.

One consequence of making use of the highest value is that our scheme is simpler than *tf.rf*. For a *N-class* problems, the *tf.rf* requires N vector representations for each document, one for each binary classificatier. Meanwhile, our scheme need only one presentation for all N binary classifiers.

Our other observation is that *tf.rf* has the lower result than *rf* in some cases as described in [8]. We believe that the reason of the problem is the impact of noisy terms that repeated many times in a document. Table 1 illustrates this point.

Table 1 Examples of two terms which have different *tf* and $log_2(1 + tf)$

term	*tf*	$log_2(1+tf)$
song	2	1.58
the	10	3.45

Clearly, *benefit* (a common word occuring in many categories) is a noisy term. According to *tf* scheme, the ratio of weight of *the* and *song* is 5:1. Meanwhile, this ratio becomes 3.45:1.58 by $log_2(1+tf)$ scheme. In other words, the effect of *the* is scaled down by $log_2(1+tf)$ scheme. We tried to combine *rf* or rf_{max} with *logtf* instead of *tf* as to improve the classification results. In our experiments, schemes using *logtf* show better performance than others using *tf* on two data sets.

To sum up, our new improved term weighting method computed weight for a term *t* as follows:

$$logtf.rf_{max} = log_2(1.0 + tf) * \max_{i=1}^{N}\{rf(C_i)\} \quad (3)$$

where *tf* is the frequency of *t*, *N* is the total number of categories, $rf(C_i)$ is defined in equation 1 Section 2.

4 Experiments

This section describes the experimental settings, including data corpora, inductive learning algorithm and performance measures.

4.1 Data Corpora

We used the Reuters News corpus and the 20 Newsgroups corpus - the two commonly used benchmark data sets. We choose these data sets so as to our results can be compared with others, especially those reported in [8].

4.1.1 Reuters News Corpus

[1]. The Reuters-21578 corpus contains the 10794 news stories, including 7775 documents in the training set and 3019 documents in the test set. There are 115 categories that has at least 1 training documents. We have conducted experiments on Reuter

[1] Reuters-21578 corpus can be downloaded from
`http://www.daviddlewis.com/resources/testcollections/reuters21578/`

top ten (10 largest categories in this corpus) and each document may be categorized in more than one category. In text preprocessing phase, 513 stop words, numbers, words containing single char and words occurring less than 3 times in the training were removed. The resulting vocabulary has 9744 unique words (features). By using CHI_{max} for feature selection, the top $p \in \{500, 2000, 4000, 6000, 8000, 10000, 12000\}$ features are tried. Besides, we also used all words in the vocabulary.

The categories in the Reuters News corpus have the skewed distribution. in the training set, the most common category (*earn*) accounts for 29% of the total number of samples, but 98% of the other categories have less than 5% samples.

4.1.2 20 Newsgroups Corpus

[2]. The 20 Newsgroups corpus (20NG) is a collection of roughly 20,000 newsgroup documents, divided into 20 newsgroups. Each newsgroup corresponds to a different topic. After removing duplicates and headers, the remaining documents are sorted by date. The training set contains 11314 documents (60%) and 7532 documents (40%) belong to the test set. In text preprocessing phase, 513 stop words, words occurring less than 3 times in the training or words containing single char were removed. There are 37172 unique words in vocabulary. We used CHI_{max} for feature selection, the top $p \in \{500, 2000, 4000, 6000, 8000, 10000, 12000, 14000, 16000\}$ were selected.

The 20 categories in the 20 Newsgroups corpus have the rough uniform distribution. This distribution is different from the category distribution in the Reuters News corpus.

4.2 Inductive Learning Algorithm

We used linear SVM algorithm since it has shown better performance than other algorithms in the prior studies [4], [7]. In addition, for SVM methods, linear kernel is simpler but as good as other kernels like RBF [6]. The linear SVM library we used is LIBLINEAR 1.93 [5].

4.3 Performance Evaluation

Measures for a category are generally *precision* (p), *recall* (r) and F_1 [16]. F_1 is a combination of *precision* and *recall* and it is defined as bellow:

$$F_1 = \frac{2.p.r}{p+r}. \tag{4}$$

F_1 is used to balance out *precision* and *recall* since we can normally not have both high *precision* and *recall* at the same time. To evaluate the performance of all

[2] The 20 Newsgroups corpus can be downloaded from
`http://people.csail.mit.edu/jrennie/20Newsgroups/`

categories in a multi-label classification problem, we have two averaging methods for F_1, namely $micro - F_1$ and $macro - F_1$. $Micro - F_1$ is dependent on the large categories while $macro - F_1$ is influenced by the small categories as described in [12]. By using these measures, our results are comparable with other results, including those in [8].

5 Results and Discussion

We will describe the experimental results and discussion in this section. In addition to *binary, tf, tf.rf, rf, logtf.rf*, we used *tf.rf_max*, *rf_max*, *logtf.rf_max* that schemes use rf_{max} instead of *rf*. The experimental results of these eight term weighting methods with respect to $micro - F_1$ and $macro - F_1$ measure on the Reuter News corpus and the 20NG corpus reported from Figure 1 to Figure 4. Each line in the figures shows the performance of each term weighting method at different of features selection levels.

5.1 Results on the 20NG Corpus

Figure 1 shows the results in term of $micro - F_1$ on the 20NG corpus. Generally, the $micro - F_1$ values of all methods increase when the number of selected features increases. $logtf.rf_{max}$ and rf_{max} are consistently better than others at all feature selection levels. Almost all term weighting methods achieve their peak at a feature size around 16000 and the best three $micro - F_1$ values 81.27%, 81.23% and

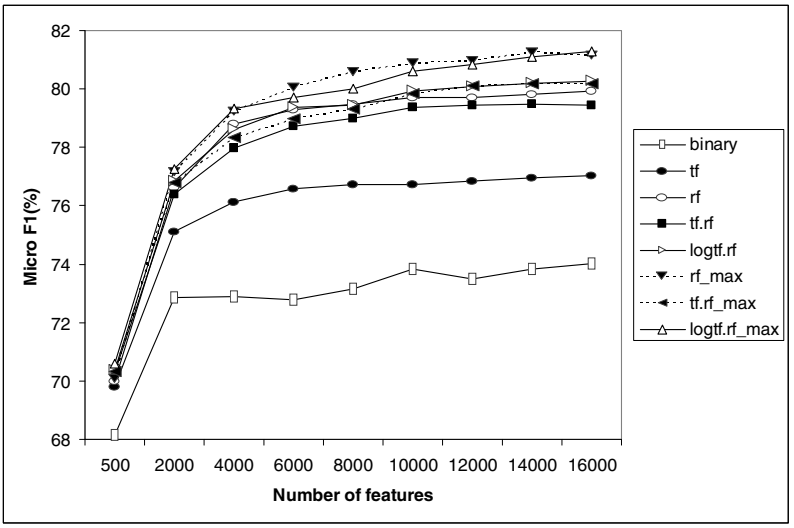

Fig. 1 $micro - F_1$ measure of eight term weighting schemes on 20NG with different numbers of features

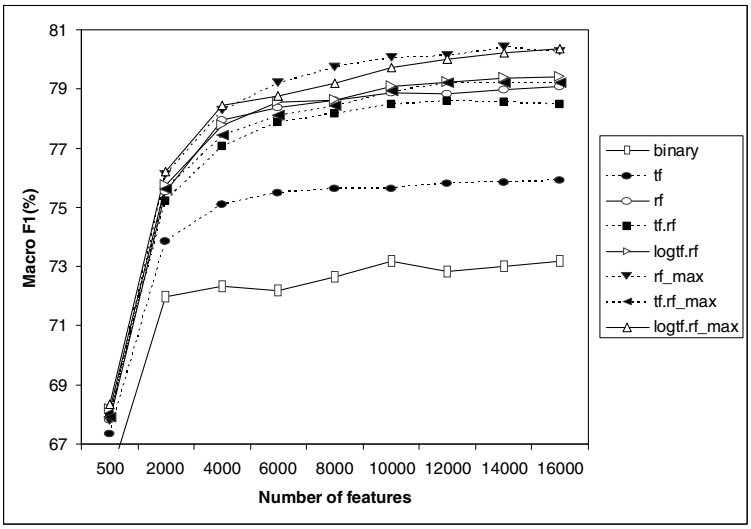

Fig. 2 $macro-F_1$ measure of eight term weighting schemes on 20NG with different numbers of features

80.27% are reached by using $logtf.rf_{max}$, rf_{max} and *logtf.rf*, respectively. *tf.rf* and *rf* reach their peak of 79.46% and 79.94%.

Figure 2 depicts the results in term of $macro-F_1$ on the 20NG corpus. The trends of the lines are similar to those in Figure 1. $logtf.rf_{max}$ and rf_{max} are better than other schemes at all different numbers of selected features.

5.2 Results on the Reuter News Corpus

Figure 3 shows the results with respect to $micro-F_1$ on the Reuters News corpus. From 6000 features onwards, the $micro-F_1$ values generally increase. $logtf.rf_{max}$ and $tf.rf_{max}$ are consistently better than others as the level of feature selection is bigger than 8000. Almost all term weighting methods achieve their peak at the full vocabulary. The best three $micro-F_1$ values 94.23%, 94.20% and 94.03% achieved by using $tf.rf_{max}$, $logtf.rf_{max}$ and *tf.rf*. Scheme rf_{max} and *rf* account for 93.50% and 93.10% at the full vocabulary.

Figure 4 depicts the results in term of $macro-F_1$ on the Reuters News corpus. The performances of eight schemes fluctuate as the number of selected features is smaller than 8000. From this point onwards, $logtf.rf_{max}$ and *logtf.rf* are schemes that are consistently better than others.

Our experimental results confirm the classification results of *tf.rf* and *rf* (the peaks and trends) as reported in [8]. Firstly, *tf.rf* shows consistently better than *rf*, *tf* and *binay* on the Reuter News corpus (Figure 3). Moreover, the performance of *rf* is better than *tf.rf*, *tf* and *binary* on the 20NG corpus.

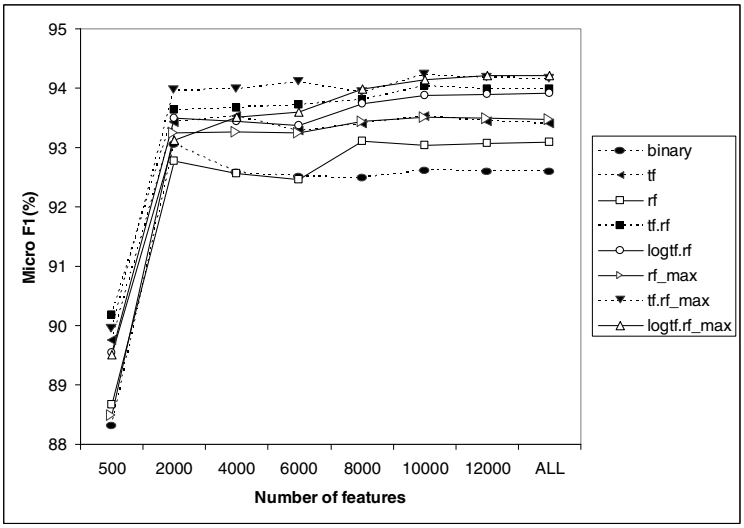

Fig. 3 $micro - F_1$ measure of eight term weighting schemes on Reuter with different numbers of features

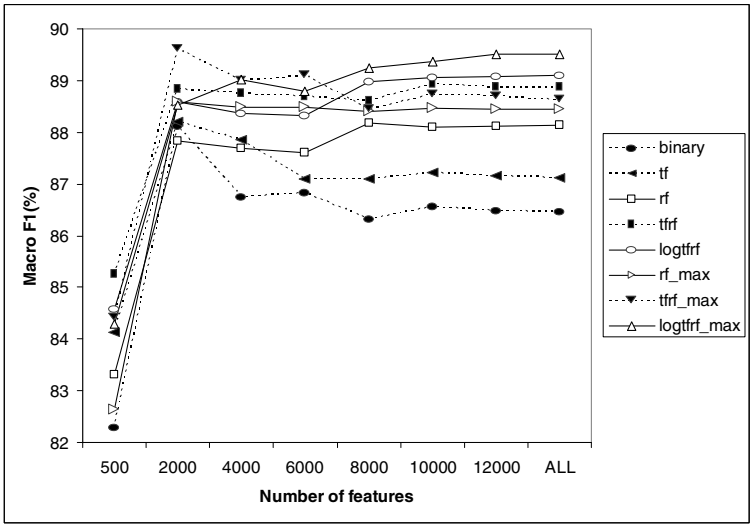

Fig. 4 $macro - F_1$ measure of eight term weighting schemes on Reuter with different numbers of features

5.3 Observations

There are some our observations of schemes with our proposed improvements:

- The schemes used rf_{max} factors are better than other schemes with *rf* factor. Specifically, $tf.rf_{max}$, $logtf.rf_{max}$ and rf_{max} are better than *tf.rf*, *logtf.rf* and *rf*, respectively in all Figures.
- The schemes used *logtf* factor yield better performance than others used *tf* factor on the 20NG corpus (Figure 1 and Figure 2). On the Reuter News corpus, the schemes used *logtf* have a comparably good performance as the schemes used *tf* (Figure 3 and Figure 4).
- $logtf.rf_{max}$, a combination of two improvements, has a comparably good performance as $tf.rf_{max}$ and rf_{max}, two best schemes on the Reuter News corpus and the 20NG corpus, respectively.
- $logtf.rf_{max}$ shows significantly better than *tf.rf* on the 20NG corpus and consistently better than *tf.rf* on the Reuter News corpus as the level of feature selection exceed 6000.

In brief, $logtf.rf_{max}$ steadily has higher performance than other shemes in our experiments.

6 Conclusions

In this study, we have introduced a newly term weighting scheme $logtf.rf_{max}$ that apply two improvements to *tf.rf* - one of the best term weighting schemes to date. Firstly, our scheme requires a single *rf* value for each term while *tf.rf* requires many *rf* values in a the multi-label classification problem. Second, our scheme used *logtf* instead of *tf*. The experimental results show that our newly term weighting scheme is consistently better than *tf.rf* and others on two data sets with the different category distribution.

For future works, we will use other classification methods (for example kNN) as well as text corpuses to further validate $logtf.rf_{max}$.

References

1. Buckley, C., Salton, G., Allan, J., Singhal, A.: Automatic Query Expansion Using SMART: TREC 3. In: NIST SPECIAL PUBLICATION SP, pp. 69–69 (1995)
2. Debole, F., Sebastiani, F.: Supervised term weighting for automated text categorization. In: Sirmakessis, S. (ed.) Text Mining and its Applications. STUDFUZZ, vol. 138, pp. 81–97. Springer, Heidelberg (2004)
3. Deng, Z.-H., Tang, S.-W., Yang, D.-Q., Li, M.Z.L.-Y., Xie, K.-Q.: A comparative study on feature weight in text categorization. In: Yu, J.X., Lin, X., Lu, H., Zhang, Y. (eds.) APWeb 2004. LNCS, vol. 3007, pp. 588–597. Springer, Heidelberg (2004)
4. Dumais, S., Platt, J., Heckerman, D., Sahami, M.: Inductive learning algorithms and representations for text categorization. In: The Seventh International Conference on Information and Knowledge Management (1998)
5. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: LIBLINEAR: A Library for Large Linear classification. The Journal of Machine Learning Research 9, 1871–1874 (2008), Software available at
`http://www.csie.ntu.edu.tw/~cjlin/liblinear`

6. Hsu, C.W., Chang, C.C., Lin, C.J.: A practical guide to support vector classification (2003)
7. Joachims, T.: Text categorization with support vector machines: Learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998)
8. Lan, M., Tan, C.L., Su, J., Lu, Y.: Supervised and traditional term weighting methods for automatic text categorization. IEEE Transactions on Pattern Analysis and Machine Intelligence 31(4), 721–735 (2009)
9. Leopold, E., Kindermann, J.: Text categorization with support vector machines. How to represent texts in input space? Machine Learning 46(1-3), 423–444 (2002)
10. Rogati, M., Yang, Y.: High-performing feature selection for text classification. In: The Eleventh International Conference on Information and Knowledge Management, pp. 659–661. ACM (2002)
11. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing and Management 24(5), 513–523 (1988)
12. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys (CSUR) 34(1), 1–47 (2002)
13. Wu, H., Salton, G.: A comparison of search term weighting: term relevance vs. inverse document frequency. ACM SIGIR Forum 16(1), 30–39 (1981)
14. Yang, Y., Liu, X.: A re-examination of text categorization methods. In: The 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 42–49. ACM (1999)
15. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: Machine Learning-International Workshop Then Conference, pp. 412–420. Morgan Kaufmann Publishers, Inc. (1997)
16. Yang, Y.: An evaluation of statistical approaches to text categorization. Information Retrieval 1(1-2), 69–90 (1999)