The Significance Of Low Frequent Terms in Text Classification

Mayy M. Al-Tahrawi*
Al-Ahliyya Amman University, Amman Jordan

The significance of low frequent terms in text classification (TC) was always debatable. These terms were often accused of adding noise to the TC process. Nevertheless, some recent studies have proved that they are very helpful in improving the performance of text classifiers. This paper shows the significance of low frequent terms in enhancing the performance of English TC, in terms of precision, recall, F-measure, and accuracy. Six well-known TC algorithms are tested on the benchmark Reuters Data Set, once keeping low frequent terms and another time discarding them. These algorithms are the support vector machines, logistic regression, k-nearest neighbor, naive bayes, the radial basis function networks, and polynomial networks. All the experiments in this research have shown a superior performance of TC when the low frequent terms are used in classification. © 2014 Wiley Periodicals, Inc.

1. INTRODUCTION

Text classification (TC) is the ability of a computer system to assign a new—unseen before—document to one or more predefined classes or topics, like sports, news, and religion, etc. The need for accurate automatic TC systems increases every day, along with the continuous huge increment of the amounts of online textual information becoming available every day.

Term selection is a common routine in TC; it involves selecting part of the document terms as discriminating keywords to be used in building a classifier rather than using all the document terms for this task. Term selection aims mainly to reduce the size of the term set used in TC; this will result in more efficient text classifiers regarding computing resources. Furthermore, term selection was considered a process that aids in removing noisy data, and hence enhances classification accuracy to a great extent.

The significance of low frequent terms in TC performance was always debatable. A recent study has proved that keeping low frequent terms can enhance polynomial networks (PN)-based TC of the Reuters Data Set to a great extent,

*Author to whom all correspondence should be addressed; e-mail: mayy.tahrawi@gmail.com.

INTERNATIONAL JOURNAL OF INTELLIGENT SYSTEMS, VOL. 29, 389–406 (2014)
© 2014 Wiley Periodicals, Inc.
View this article online at wileyonlinelibrary.com. • DOI 10.1002/int.21643

regardless of the term-weighting scheme adopted or the term-reduction method used.[1] The enhancement on the accuracy recorded when keeping the low frequent terms in this research was great; it reached 17% in some experiments. Another recent study has shown the significance of low frequent terms in patent classification;[2] keeping low frequent terms in their experiments has shown to outperform the set of high frequent terms in classifying patent documents. Other studies in the literature have also concluded that low frequent terms are very helpful in improving the accuracy of TC.[3–6]

The research conducted in Ref. 1 is extended here to investigate the significance of low frequent terms in TC using other state-of-the-art TC algorithms. Furthermore, additional performance measures are used here to investigate the significance of low frequent terms in TC. Besides PNs, five of the top performers in English TC algorithms are selected: support vector machines (SVM), logistic regression (LR), k-nearest neighbor (kNN), naive bayes (NB), and the radial basis function (RBF) networks. Each of the six algorithms is used in this research to classify the Reuters Data Set, once keeping low frequent terms as a part of the terms used for building the classifier, and another time discarding them. All the experiments using all algorithms have proved that keeping low frequent terms has achieved a superior TC performance of the Reuters Data Set as compared with removing these terms, in terms of classification accuracy, precision, recall, and F-measure.

The paper is organized as follows: Section 2 presents an overview of the text classifiers used in this research, while Section 3 is devoted to explain, in brief, the data set used and the processing steps performed on the data set. The performance evaluation measures used in the experiments are explained in Section 4, and Section 5 of the paper presents a summary of the results reached in the experiments conducted in this research. Analysis of these results takes place in Section 6, and finally, conclusions and intended future work are presented in Section 7.

2. TC ALGORITHMS

Six of the top performers in TC are used in this research to investigate the significance of low frequent terms in TC: SVMs, LR, kNN, NB, the RBF networks, and PNs. Details of each of these classification algorithms are presented next.

2.1 SVMs

SVMs were introduced by Vapnik.[7–9] Empirical studies have shown that SVM is the state-of-the-art technique among other well-known TC algorithms.[10] Moreover, SVMs are fully automatic; no manual parameter tuning is needed. SVMs are based on the *Structural Risk Minimization* principle[9] from computational learning theory. The idea of structural risk minimization is to find a hypothesis h for which the lowest true error can be guaranteed. The true error of h is the probability that h will make an error on a random unseen test example. An upper bound can be used to connect the true error of a hypothesis h with the error of h on the training set and the complexity of H (measured by Vapnik–Chervonenkis, i.e., VC dimension); the

hypothesis space containing h.[9] SVMs find the hypothesis h which approximately minimizes this bound on the true error by controlling the VC dimension of H efficiently. For details of SVMs computations, and how to apply them in TC, the reader can refer to Refs. 10 and 11.

2.2 LR

LR has been a well-known statistical model suitable for probabilistic classification. Recently, logistic regression has been studied in statistical machine learning community.[12–15] It is a high-performance classifier that can be efficiently trained with a large number of labeled examples. Previous studies have shown that the logistic regression model is able to achieve similar performance of TC as SVMs.[12,15,16] Logistic regression can be applied to both real and binary data. It outputs the posterior probabilities for test examples that can be conveniently processed and engaged in other systems. In theory, given a test example x, logistic regression models the conditional probability of assigning a class label y to the example by[13]

$$P(y|x) = \frac{1}{1 + \exp\left(-y\alpha^T x\right)}. \tag{1}$$

where α is the model parameter.

2.3 kNN

kNN is a well-known statistical approach that has been applied to TC since the early stages of research. It is one of the top-performing methods on Reuters.[12,17] The algorithm is very simple; given a test document to classify, the classifier finds the kNNs of the test document, and majority voting among the neighbors is used to decide the category of the test document. Similarity is measured by the cosine between the vectors representing the documents. If a category is shared by more than one of the k neighbors, then the sum of the similarity scores of these neighbors is the weight of that shared category.

2.4 NB

NB is a well-known and highly practical probabilistic classifier that has been widely used in TC. It uses the joint probabilities of words and categories to estimate the probabilities of categories, given a test document. The naive part in this algorithm is the assumption of word independence: the probability of a word, given a category, is assumed to be independent from the conditional probabilities of other words given in that category; that is, it does not use word combinations as predictors. This naïve assumption results in saving the computation time to a great extent. Several studies show that NB performs surprisingly well in TC, despite this wrong independence assumption.[12,18] Zheng et al.[19], Lewis and Ringuette[20], and Tahrawi and Abu Zitar[12] found term selection to be very useful for Reuters when classified with NB. In TC,

the probability of a class C, given a document dj is calculated by Bayes' theorem as follows:[21]

$$P(C|dj) = \frac{P(dj|C)\,P(C)}{P(dj)} \tag{2}$$

$$= \frac{P(dj|C)\,P(C)}{P(dj|C)\,P(C) + P\left(dj|\bar{C}\right)P\left(\bar{C}\right)} \tag{3}$$

2.5 RBF Networks

RBF network is an artificial neural network model motivated by the locally tuned response observed in biological neurons.[22] RBF networks were used early for interpolation,[23,24] probability density estimation,[25–27] and approximations of smooth multivariate functions.[28] They have also been applied with success to classification.[12,29–32] The RBF network has a feed-forward structure consisting of a single hidden layer of a number of locally tuned units that are fully interconnected to an output layer of a number of linear units. All hidden units simultaneously receive the input vector. Hidden units outputs are calculated as the distance between the input vector and the weight vector of the hidden unit multiplied by a bias b. The bias allows the sensitivity of the radial basis unit to be adjusted. It determines the width of the area in the input space to which each hidden unit responds. A distinguishing feature of a RBF network is its adaptive nature, which generally allows it to utilize a relatively smaller number of locally tuned units.

2.6 PNs

PN classifiers have been known in the literature for many years.[33] They have been recently used in many areas like speaker verification and sign-language recognition.[34–38] More recently, PNs have proved to be competitive to the top performers in the field of English TC of the two benchmark data sets in TC: Reuters and 20Newsgroups.[1,12] More importantly, this performance was achieved in one shot training (non-iteratively) and using just 0.25%–0.5% of the corpora terms.[1,12]

The PN model adopted in this research consists of two layers. The first layer (the input layer) forms the monomial basis terms of the input vector x ($x_1, x_2, \ldots, x_N$), such as $1, x_1, x_2, {x_1}^2$, and so on, where N is the number of terms of x. A second layer then linearly combines the output of the first layer, that is, the data are first expanded into a high dimensional space in the first layer and then linearly separated using the second layer. The basic embodiment of a Kth-order PN consists of several parts. The N terms of one observation $x(x_1, x_2, \ldots, x_N)$ are used to form a basis function $p(x)$; one $p(x)$ is formed for each observation. The elements of $p(x)$ for a polynomial of degree K are monomials of the form[35]

$$\prod_{j=1}^{N} x_j^{k_j}, \quad \textit{where} \quad k_j \geq 0 \quad \textit{and} \quad 0 \leq \sum_{j=1}^{N} k_j \leq \quad K \tag{4}$$

The second layer of the PN linearly combines all inputs to produce weights of classes. The whole class is represented by one weight, which is computed during the training phase. Details of using PNs in TC are explained next. Polynomials of degree 2 were used in this research.

2.6.1 The Training Phase of PN Classifiers

A PN is trained to approximate an ideal output using mean squared error as the objective criterion. The polynomial expansion of the *i*th-class term vectors (documents) is denoted by[12,34]

$$M_i = [p(x_i, 1)\, p(x_i, 2) p(x_i, 3) \ldots p(x_i, N_i)]^t \tag{5}$$

where N_i is the number of training term vectors for class i, and $p(x_i,m)$ is the basis function of the mth term vector for class i. After forming M_i for each class i of the nc training classes, a global matrix M is obtained for the nc classes, by concatenating the individual M_i's computed for each class[12,35]

$$M = [M_1\, M_2\, M_3 \quad M_{nc}]^t \tag{6}$$

The training problem then reduces to finding an optimum set of weights w (one weight for each class) that minimizes the distance between the ideal outputs and a linear combination of the polynomial expansion of the training data such that[12,35]

$$w_i^{opt} = \arg \min_{w} \| M w - o_i \|_2 \tag{7}$$

where o_i is the ideal output. A class model w_i^{opt} can be obtained in one shot by applying the normal equations method, which is as follows:[12,35]

$$M^t\, M\, w_i^{opt} = M^t\, oi \tag{8}$$

2.6.2 Recognition Phase of PN Classifiers

Classification of a new unseen document consists of two parts: identification and verification. Identification involves finding the best matching class of an unseen document, given the term vector of this document. In the verification phase, the claim made in the identification phase is either accepted or rejected. The identification phase proceeds as follows in the PN algorithm: the term vector x of the unseen document is expanded into its polynomial terms $p(x)$ in a manner similar to what was done with the training inputs in the training phase. Then, the new unseen document is assigned to the class c such that[12,35]

$$c = \arg \max\, w_i^{opt} \cdot p(x) \quad \text{for } i = 1, 2, \ldots, nc \tag{9}$$

Copy Rights

Table I. Distribution of documents and terms among R8 classes.

Class number	Class	Number of train documents	Number of test documents	Total number of documents	Number of terms
1	Acq	1,596	696	2,292	7,323
2	Crude	253	121	374	2,751
3	Earn	2,840	1,083	3,923	7,188
4	Grain	41	10	51	1,038
5	Interest	190	81	271	1,448
6	Money–fx	206	87	293	1,992
7	Ship	108	36	144	1,676
8	Trade	251	75	326	2,652
	Total	5,485	2,189	7,674	13,891 (after removing duplicates among classes)

3. DATA SET

The Reuters-21578 benchmark subset suitable for single-label TC—R8[39]—was used in this research. The whole processing steps performed on the data sets can be summarized as follows:

1. Only letters, hyphens '-', and underscores '_' are kept; any other character is eliminated.
2. All letters are converted to lowercase.
3. Tabs, new lines, and RETURN characters are replaced by single spaces.
4. The Porter stemmer[40] was used, with the following modification: an ignore list of more than 1000 stop words is defined and used to reduce the number of terms in the data set.
5. Then, any remaining word consisting of just one character is removed.

The distribution of documents and terms, per class, for R8 is shown in Table I.

3.1 Term Selection

Chi Square (χ^2) was used to compute the discriminating power of each term in the corpus in this research. Chi square has shown to yield good results in classification, compared with other term-selection methods.[1,12,41–44] The chi-square score measures the correlation dependency between the term and its containing class. The higher this score is, the more discriminating the term is for that class. The chi-square measure is computed for each term t in each class c_i as follows:[45]

$$\chi^2(\mathrm{t}, \mathrm{c_i}) = \frac{\mathrm{N} \times (\mathrm{AD} - \mathrm{CB})^2}{(\mathrm{A} + \mathrm{C}) \times (\mathrm{B} + \mathrm{D}) \times (\mathrm{A} + \mathrm{B}) \times (\mathrm{C} + \mathrm{D})} \tag{10}$$

where:

1. N is the total number of training documents in the data set,
2. A is the number of documents belonging to class c_i and containing t,
3. B is the number of documents belonging to class c_i but not containing t,
4. C is the number of documents not belonging to class c_i but containing t, and
5. D is the number of documents neither belonging to class c_i nor containing t.

The chi square measure can be globalized for terms that appear in more than one class (usually with different chi-square measures in different classes) in one score by choosing the maximum or the average score. Finally, the reduced term set is chosen from the topmost chi-square measure terms, ignoring terms with zero or small measures. Different term-reduction criteria can be used, as explained in the next section.

3.2 Term-Reduction Methods

Three different methods were used to reduce the resulting set of terms. Details of these methods are explained in the following subsections. Each of these methods was tried once keeping the low frequent terms in each document, and another time discarding them. Low frequent terms are the terms that are selected for building the classifier and occur in a training or testing document with a frequency of three or less. This aims mainly to investigate the role of low frequent terms in enhancing document-classification performance. Detailed results of using these reduced term sets in classification, as well as an analysis of these results, will follow in the subsequent sections.

3.2.1 Selecting the Topmost Terms from the Corpus as a Whole

First, the topmost 100 chi square measure terms from the corpus as a whole (0.72% of the corpus terms) were selected. Another reduced term set is formed by selecting the topmost 70 chi-square measure terms from the corpus as a whole (0.5% of the corpus terms). This term set was created to compare the classification performance using the same reduction method (the corpus topmost terms) but with a smaller number of terms.

3.2.2 Selecting an Equal Number of Terms from Each Class in the Corpus

An equal number of terms is chosen from each class as a second term-reduction strategy. This aims to overcome the problem of the variation in the number of terms chosen from each class to build the classifier, when the previous method is adopted. The topmost 13 chi-square measure terms were selected from each class, and these 104 terms were reduced to 96 terms (0.7% of the corpus terms) after the elimination of duplicates.

3.2.3 Selecting an Equal Percentage of the Topmost Chi-Square Measure Terms from Each Class

The last term-reduction strategy was to select an equal percentage of the topmost chi-square measure terms from each class (0.5% of each class). The term set had 131 terms and these 131 terms were reduced to 108 terms after the elimination of duplicates.

3.3 Term Weighting

In TC, each document is represented by a vector of term weights, which represent the strength or significance of these terms in this document. These weights are usually numbers that fall in the [0,1] interval. Several term-weighting schemes were used in the literature of TC, such as document frequency, term frequency, normalized tf.idf, binary weights, information gain, and weighted inverse document frequency. Normalized document frequency was used for term weighting in all classification algorithms experimented in this research, except NB, which uses binary weights. Normalized document frequency was selected, because previous researches on the same data set[1,12] have shown this term-weighting scheme to record the best performance on this data set.

4. PERFORMANCE EVALUATION

All classifiers in this research are evaluated by measuring their accuracy, micro- and macro-averaged precision, recall, and F1 measure.

4.1 Accuracy

Accuracy of a class c_i, Acc_i, is computed as follows:

$$Acc_i = \frac{TP_i}{TP_i + FN_i + FP_i}. \tag{11}$$

where

1. TP_i: true positives with respect to a category c_i; the number of documents correctly claimed by the classifier as belonging to category c_i.
2. FP_i: false positives with respect to c_i; the number of documents incorrectly claimed by the classifier as belonging to c_i.
3. FN_i: false negatives with respect to c_i; the number of documents incorrectly claimed by the classifier as not belonging to c_i.

4.2 Precision

Precision refers to the proportion of test files classified into a class that really belong to that class. Precision of a class c_i, P_i can be defined as follows:[46]

$$P_i = \frac{TP_i}{TP_i + FP_i}. \tag{12}$$

4.3 Recall

Recall is the proportion of test files belonging to a class and are claimed by the classifier as belonging to that class. Recall of a class c_i (R_i) can be computed using

Copy Rights

Table II. Detailed results when removing low frequent terms.

Algorithm	Number of terms	Accuracy	Micro-averaged results			Macro-averaged results		
			Precision	Recall	F-measure	Precision	Recall	F-measure
LR	70	76.5190	76.5190	76.5190	76.5190	84.4685	67.4734	73.252
	96	81.8	81.7725	81.7725	81.7725	82.8077	69.9821	74.6656
	100	81.8182	81.8182	81.8182	81.8182	82.8731	69.3687	74.1667
	108	84.011	84.011	84.011	84.011	84.3793	72.0425	76.5068
KNN	70	83.8739	91.2746	91.2746	91.2746	87.3481	83.5973	85.1837
	96	83.5998	92.2796	92.2796	92.2796	91.9785	79.3759	84.1370
	100	84.2851	92.2339	92.2339	92.2339	92.8585	79.3249	84.7314
	108	83.5541	83.5541	83.5541	83.5541	89.4295	68.9951	75.9272
RBF	70	61.6263	63.1850	61.6263	62.3959	53.0966	46.0355	42.2997
	96	68.3874	70.2158	68.3874	69.2895	46.7859	38.1385	38.1091
	100	68.3417	70.6660	68.3417	69.4844	46.9411	41.2100	40.0376
	108	67.9762	70.6553	67.9762	69.2899	49.2058	42.7290	41.2102
NB	70	83.5541	83.5541	83.5541	83.5541	84.5835	69.5958	75.0106
	96	84.2394	84.2394	84.2394	84.2394	84.2359	69.6763	74.9712
	100	84.1937	84.1937	84.1937	84.1937	82.1494	69.4310	73.8895
	108	83.8282	83.8282	83.8282	83.8282	85.5667	68.5771	74.8927
SVM	70	76.5190	76.519	76.519	76.519	87.2895	67.098	73.362
	96	82.0923	82.0923	82.0923	82.0923	87.4182	70.4238	76.5063
	100	82.5948	82.5948	82.5948	82.5948	88.6216	70.7407	77.0266
	108	84.6505	84.6505	84.6505	84.6505	90.2484	72.0904	78.7368
PN	70	76.519	76.5190	76.5190	76.5190	84.43	67.18	72.98
	96	80.9959	80.9959	80.9959	80.9959	80.6931	66.9159	72.0611
	100	81.0873	81.0873	81.0873	81.0873	81.7901	68.1491	73.0262
	108	82.9146	82.9146	82.9146	82.9146	86.0144	69.4407	75.6344

the following formula:[46]

$$R_i = \frac{TP_i}{TP_i + FN_i}. \tag{13}$$

4.4 F1 measure

The F1 measure, introduced by Rijsbergen[47] is the harmonic average of both precision and recall. F1 is computed as follows:[46]

$$F1 = \frac{2 * Recall * Precision}{Recall + Precision} \tag{14}$$

$$= \frac{2 * TP_i}{2 * TP_i + FN_i + FP_i} \tag{15}$$

Individual results of categories can be either micro-averaged or macro-averaged to give an idea of the classification performance on the corpus as a whole. The six classifiers are evaluated using accuracy, precision, recall, and F1. Micro- and macro-averaged results are presented to give a clear idea of the classifiers performance. For

Table III. Detailed results when keeping low frequent terms.

Algorithm	Number of terms	Accuracy	Micro-averaged results			Macro-averaged results		
			Precision	Recall	F-measure	Precision	Recall	F-measure
LR	70	93.2846	93.2846	93.2846	93.2846	88.4652	86.7366	87.2397
	96	94.4267	94.4267	94.4267	94.4267	91.4317	88.8195	89.8624
	100	94.1983	94.1983	94.1983	94.1983	88.8182	86.6119	87.3856
	108	95.0662	95.0662	95.0662	95.0662	91.2302	89.2222	89.9377
KNN	70	92.73641	92.73641	92.73641	92.73641	92.49809	83.67196	87.14097
	96	92.28	92.27958	92.27958	92.27958	90.78793	80.2498	84.45063
	100	92.64504	92.64504	92.64504	92.64504	91.67699	82.03712	86.15902
	108	92.73641	92.73641	92.73641	92.73641	91.2589	83.19586	86.53073
RBF	70	76.42841	77.41331	75.46825	76.42841	51.54494	51.88161	47.22399
	96	77.38	78.36066	76.42759	77.38205	51.3376	54.35716	48.46213
	100	76.63	77.63713	75.65098	76.63119	52.62719	57.04643	49.31039
	108	77.94	79.96156	76.01645	77.93911	55.44916	56.82075	51.65044
NB	70	92.3253	92.3253	92.3253	92.3253	84.3784	84.4005	84.0656
	96	92.2796	92.2796	92.2796	92.2796	85.1718	82.7446	83.7172
	100	91.7314	91.7314	91.7314	91.7314	83.8816	81.2529	82.3956
	108	92.65	92.645	92.645	92.65	85.5184	82.4948	83.8347
SVM	70	93.1932	93.19324	93.19324	93.19324	92.56581	82.71749	86.40631
	96	93.6958	93.69575	93.69575	93.69575	91.73871	82.70901	86.21263
	100	94.11	94.11	94.11	94.11	92.16376	83.82417	87.1019
	108	95.3403	95.34034	95.34034	95.34034	93.38495	87.19103	89.80104
PN	70	92.4166	92.4166	92.4166	92.4166	86.5387	85.416	85.4169
	96	91.5943	91.5943	91.5943	91.5943	85.5336	83.1539	83.9597
	100	91.7314	91.7314	91.7314	91.7314	86.8798	82.5371	84.4752
	108	93.6044	93.6044	93.6044	93.6044	90.0878	87.8713	88.8039

detailed formulae for computing micro- and macro-averaged results, the reader can refer to Ref. 46.

5. EXPERIMENTS AND RESULTS

Each of the six classification algorithms was experimented using the four reduced term sets once keeping low frequent terms and another time discarding them. Various parameters settings were tested for SVM, kNN, and RBF networks; the best results are presented here. Experiments results are summarized in Tables II and III. Figures 1–18 compare each algorithm performance when keeping low frequent terms versus removing them.

6. ANALYSIS OF RESULTS

It is apparently clear from all the experiments of this research that keeping low frequent terms has achieved superior precision, recall, F-measure, and accuracy compared with the results of the same experiment settings with the low frequent terms being removed. This is valid for all the classification algorithms tested in this

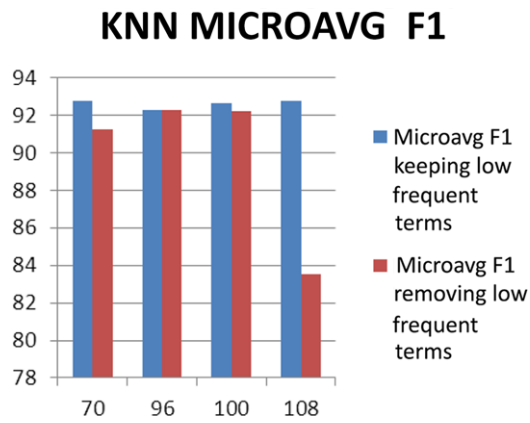

Figure 1. MicroAverage F1 of kNN: keeping vs removing low frequent terms

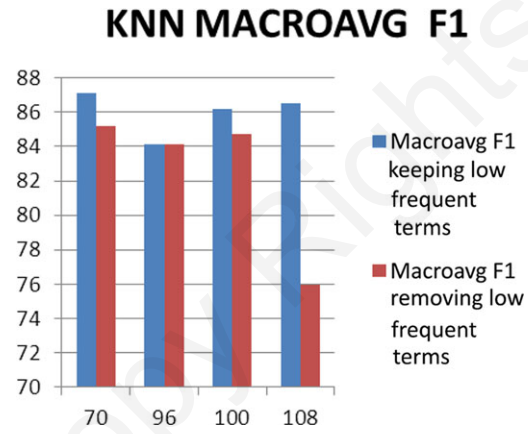

Figure 2. MacroAverage F1 of kNN: keeping vs removing low frequent terms

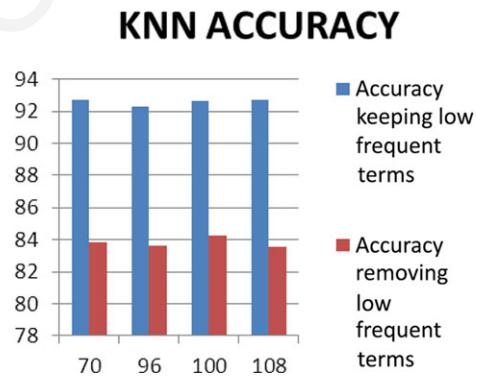

Figure 3. Accuracy of kNN: keeping vs removing low frequent terms

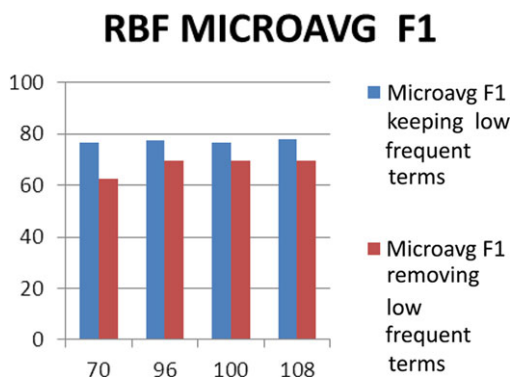

Figure 4. MicroAverage F1 of RBF: keeping vs removing low frequent terms

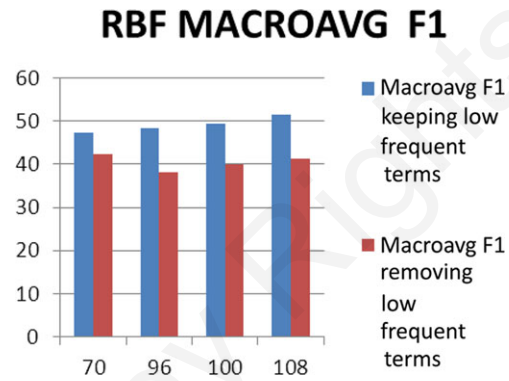

Figure 5. MacroAverage F1 of RBF: keeping vs removing low frequent terms

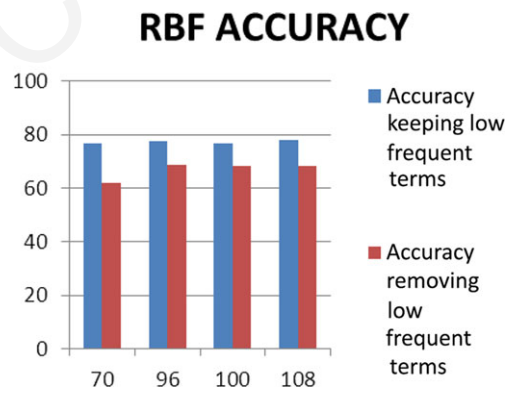

Figure 6. Accuracy of RBF: keeping vs removing low frequent terms

Copyrights

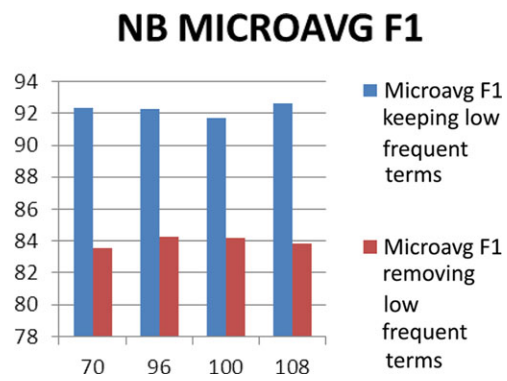

Figure 7. MicroAverage F1 of NB: keeping vs removing low frequent terms

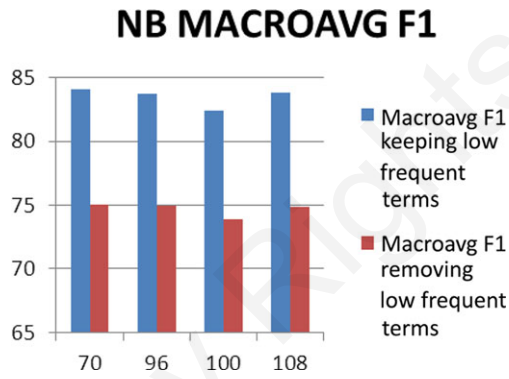

Figure 8. MacroAverage F1 of NB: keeping vs removing low frequent terms

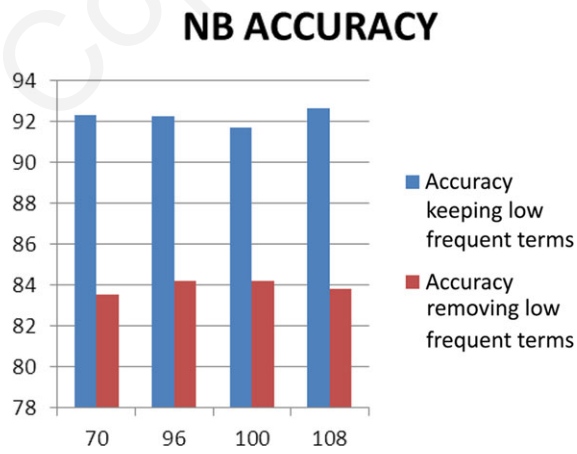

Figure 9. Accuracy of NB: keeping vs removing low frequent terms

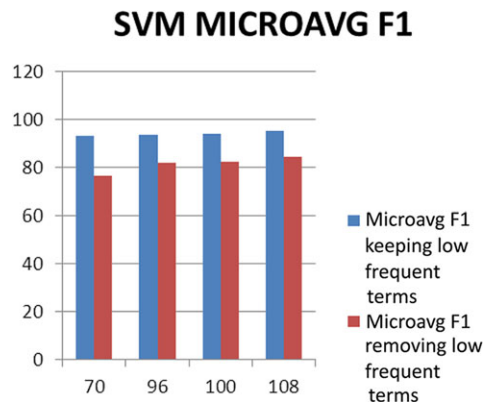

Figure 10. MicroAverage F1 of SVM: keeping vs removing low frequent terms

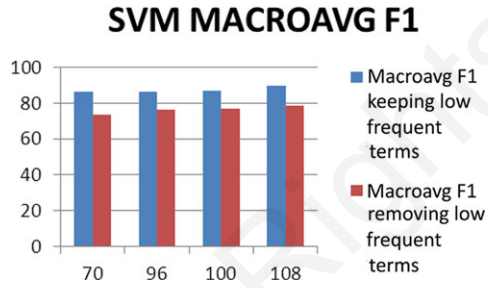

Figure 11. MacroAverage F1 of SVM: keeping vs removing low frequent terms

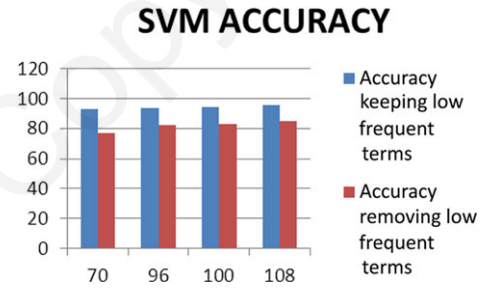

Figure 12. Accuracy of SVM: keeping vs removing low frequent terms

research regardless of the term-reduction method used. The enhancement on the accuracy recorded when keeping the low frequent terms is great; it lies between 16% and 17% for PN, SVM, and LR, and lies between 9% and 11% for NB, RBF, and kNN.

It is also clear that the 108-terms–reduced term set has the optimum performance among the other term sets. This term set was constructed by selecting an

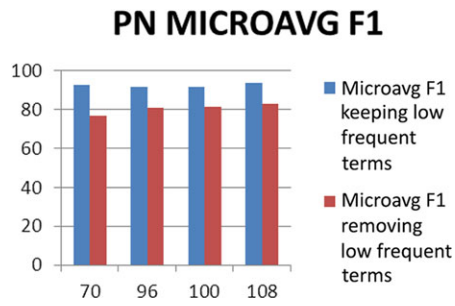

Figure 13. MicroAverage F1 of PN: keeping vs removing low frequent terms

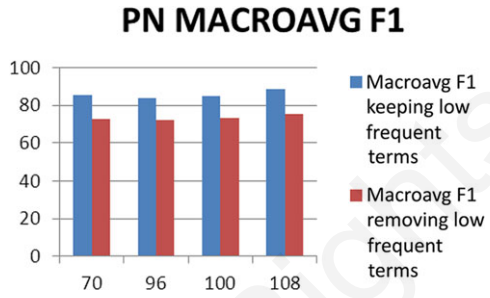

Figure 14. MacroAverage F1 of PN: keeping vs removing low frequent terms

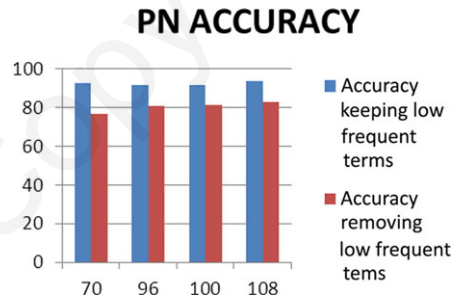

Figure 15. Accuracy of PN: keeping vs removing low frequent terms

equal percentage of the topmost terms in each class in the corpus. In fact, this complies with the conclusions in Refs. 1 and 12, that found that using equal percentage of class terms resulted in the best classification performance in several classifiers, compared with using equal number of terms from each class, or just choosing a specified number of the corpus topmost terms, as this guarantees that all classes participate evenly in building the classifier.

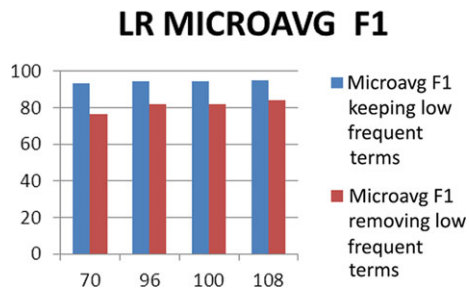

Figure 16. MicroAverage F1 of LR: keeping vs removing low frequent terms

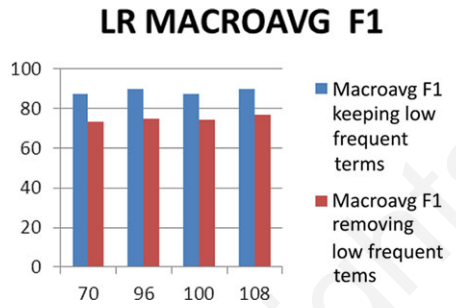

Figure 17. MacroAverage F1 of LR: keeping vs removing low frequent terms

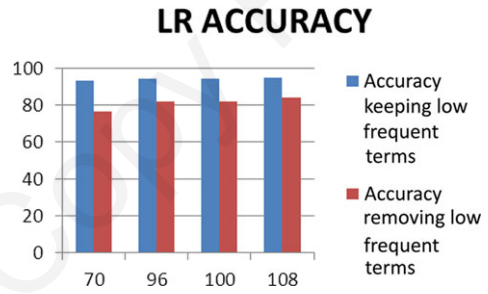

Figure 18. Accuracy of LR: keeping vs removing low frequent terms

7. CONCLUSIONS

In this paper, the significance of low frequent terms in TC was investigated. Six famous TC algorithms were tested. Each algorithm was tested using four reduced term sets that were selected using different reduction methods. On the basis of the results of the experiments conducted in this research, we strongly recommend keeping low frequent terms (after applying efficient term selection and reduction criteria) in classifying the Reuters Data Set, due to their remarkable effect in enhancing the accuracy, precision, recall, and F-measure of text classifiers. The intended near

future work is to extend the work conducted in this research to study the effect of keeping low frequent terms on classifying other benchmark data sets.

References

1. Al-Tahrawi MM. The role of rare terms in enhancing the performance of polynomial networks based text categorization. J Intell Learn Syst Appl 2013;5:84–89.
2. Khattak AS Heyer G. Significance of low frequent words in patent classification. The Sixth Int Multi-Conf Computing in the Global Information Technology, ICCGI 2011, Luxembourg city, Luxembourg. pp. 8–13.
3. Koller D, Sahami M. Hierarchically classifying documents using very few words. The Fourteenth Int Conf Machine Learning (ICML'97), Morgan Kaufmann Publishers Inc. San Francisco, CA, USA 1997. pp. 170–178.
4. Wang D, Zhang H. Inverse-category-frequency based supervised term weighting scheme for text categorization. J Inf Sci Eng 2013;(2):209–225.
5. Deisy C, Gowri M, Baskar S, Kalaiarasi SMA, Ramraj N. A novel term weighting scheme MIDF for text categorization. J Eng Sci Technol 2010;5(1):94–107.
6. Schonhofen P, Benczur AA. Exploiting extremely rare terms in text categorization. In: ECML'06 Proc 17th European Conf Machine Learning, Springer–Verlag, Berlin, Heidelberg, Germany; 2006. pp. 759–766.
7. Boser B, Guyon M, Vapnik V., A training algorithm for optimal margin classifiers. Conf Computational Learning Theory (COLT) ACM, New York, NY, USA (1992). pp. 144–152.
8. Cortes C, Vapnik V. Support vector networks. Mach Learn 1995;20:273–297.
9. Vapnik V. The nature of statistical learning theory. New York: Springer; 1995.
10. Joachims T. Text categorization with support vector machines: learning with many relevant features. Proc 10th Euro Conf Machine Learning (ECML) Springer-Verlag London, UK 1998;1398:137–142.
11. Burges CJC. A tutorial on support vector machines for pattern recognition. Data Min Knowl Discovery 1998;2(2):121–167.
12. AL-Tahrawi MM, Abu Zitar R. Polynomial networks versus other techniques in text categorization. Int J Patt Recog Artif Intell 2008; 22(2):295–322.
13. Hoi SCH, Jin R, Lyu MR. Large-scale text categorization by batch mode learning. In: Proc 15th Int World Wide Web conference (WWW2006), Edinburgh, England, UK, May, 2006.
14. Vapnik V. Statistical learning theory. New York: Wiley; 1998.
15. Zhang J, Jin R, Yang Y, Hauptmann A. Modified logistic regression: an approximation to SVM and its applications in large-scale text categorization. In: Proc Twentieth Int Conf Machine Learning (ICML 2003), Washington, DC USA, August, 2003; pp. 21–24.
16. Komarek P, Moore A. Making logistic regression a core data mining tool: a practical investigation of accuracy, speed, and simplicity. Technical Report TR-05—27, Robotics Institute, Carnegie Mellon University, May 2005.
17. Yang Y, Liu X. A re-examination of text categorization methods. SIGIR'99, ACM; 1999. pp. 42–49.
18. Domingos P, Pazzari MJ. On the optimality of the simplest Bayesian classifier under zero-one loss,. Mach. Learn. 1997;29(2/3):103–130.
19. Zheng Z, Wu X, Srihari R. Feature selection for text categorization on imbalanced data. SIGKDD Explorations 2004;6(1):80–89.
20. Lewis DD, Ringuette M. A comparison of two learning algorithms for text categorization. Proc Third Ann Symp Document Analysis and Information Retrieval (SDAIR'94) 1994: pp. 81–93.
21. Kim SB, Seo HC, Rim HC. Poisson Na¨ıve Bayes for text classification with feature weighting. Proc 6th Int. Workshop on Asian Language, Sapporo, Japan, ACL, July 2003: pp. 33–40.
22. Hassoun MH. Fundamentals of artificial neural networks. The MIT Press. 1995.

23. Micchelli CA. Interpolation of scattered data: distance and conditionally positive definite functions. Constr Approximation. 1986;2(1):11–22.
24. Powell MJD. Radial basis functions for multivariate interpolation: a review. In: Mason JC, Cox MG, editors. Algorithms for the approximation of functions and data. Oxford, England: Clarendon Press; 1987. pp. 143–167.
25. Duda RO, Hart PE. Pattern classification and scene analysis. New York: Wiley; 1973.
26. Parzen E. On estimation of a probability density function and mode. Ann Math Stat 1962; 33:1065–1076.
27. Specht DF. Probabilistic neural networks. Neural Network 1990; 3(1):109–118.
28. Poggio T, Girosi F. Networks for approximation and learning. Proc IEEE 1990;78(9):1481–1497.
29. Lee Y. Handwritten digit recognition using k-nearest neighbor, radial basis functions, and backpropagation neural networks. Neural Comput 1991;3(3):440–449.
30. Niranjan M, Fallsise F. Neural networks and radial basis functions in classifying static speech patterns. Computer Speech & Language, 1990;43:275–289.
31. Nowlan SJ. Maximum likelihood competitive learning. In D. S. Touretzky, editor. Advances in neural information processing systems 2, San Mateo, CA: Morgan Kaufmann; 1990. pp. 574–582.
32. Wettschereck D, Dietterich T. Improving the performance of radial basis function networks by learning center locations. In: J.E.Moody, S.J.Hanson, and R.P.Lippmann, editors. Advances in neural information processing systems 4. San Mateo, CA: Morgan Kaufmann; 1992. pp. 1133–1140.
33. Fukunaga K. Introduction to statistical pattern recognition. New York: Academic Press; 1990.
34. Campbell WM, Assaleh KT, Broun CC. A novel algorithm for training polynomial networks. Int NAISO Symp Information Science Innovations ISI'2001 Dubai, UAE; March 2001.
35. Assaleh K, Al-Rousan M. A new method for Arabic sign language recognition. EURASIP J Appl Signal Processing. New York: Hindawi Publishing Corporation; 2005. pp. 2136–2145.
36. Campbell WM, Broun CC. Using polynomial networks for speech recognition. Neural Networks Signal Proc X. 2000; 795–803.
37. Campbell WM, Assaleh KT. Polynomial classifier techniques for speaker verification. In: Proc Int Conf Acoustics, Speech, and Signal Processing, IEEE International Conference-Volume 01, IEEE Computer Society, 1999: pp. 321–324.
38. Assaleh KT, Campbell WM. Speaker identification using a polynomial-based classifier. Int Symp Signal Processing and its Applications; Brisbane, Qld., 1999: Vol. 1, pp. 115–118.
39. Ana site for data sets suitable for single-label text categorization. http://web.ist.utl.pt/∼acardoso/datasets/
40. Porter MF. An algorithm for suffix stripping. Program 14(3); 1980: 130–137.
41. Forman G. An extensive empirical study of feature selection metrics for text classification. J Mach Learn Res 2003;3:1289–1305.
42. Yang Y, Pederson J. A comparative study on term selection in text categorization. Proc Fourteenth Int Conf on Machine Learning. San Francisco, CA: Morgan Kaufmann, 1997. pp. 412–420.
43. Fuka K, Hanka R. Term set reduction for document classification problems. IJCAI-01 Workshop: text learning: beyond supervision. Seattle, WA; 2001.
44. Rogati M, Yang Y. High- performing term selection for text classification. CIKM' McLean, Virginia, USA, November, 2002, pp. 4–9.
45. Zheng Z, Wu X, Srihari R. Term selection for text categorization on imbalanced data. IGKDD Explorations 2004;6(1):80–89.
46. Debole F, Sebastiani F. An analysis of the relative hardness of Reuters-21578 subsets. JASIS 2005; 56(6):584–596.
47. Van Rijsbergen CJ. Information retrieval. 2nd edn. London: Butterworths; 1979.