ISSN: 2277-3754
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 1, Issue 3, March 2012

Study and Performance Evaluation of Various Term Weighing Methods for Automated Text Categorization Using SVM

Sandeep S. Patil, Rashmi R. Rathi, Ankita H. Tidake

Abstract*—The term weighting method is an important step to improve the effectiveness of text categorization (TC) by assigning appropriate weights to terms. The traditional term weighting methods borrowed from information retrieval(IR), such as binary, term frequency (tf), tf:idf, and its various variants, belong to the unsupervised term weighting methods as the calculation of these weighting methods do not make use of the information on the category membership of training documents. Generally, the supervised term weighting methods adopt this known information in several ways. Therefore, the fundamental question arise here, "Does the difference between supervised and unsupervised term weighting methods have any relationship with different learning algorithms?", and if we consider normalized term frequency instead of term frequency along with relevant frequency the new method will be ntf.rf. but will this new method is effective for text categorization? So we would like to answer these questions by implementing new supervised term weighing method (ntf.rf) using Support Vector Machines (SVM).

***Index Terms*—Text Categorization, Information Retrieval, Term weighting, SVM.**

I. Introduction

Text Categorization (TC) is the task of automatically classifying unlabelled natural language documents into a predefined set of semantic categories. As the first and a vital step, text representation converts the content of a textual document into a compact format so that the document can be recognized and classified by a computer or a classifier. This problem has received a special and increased attention from researchers in the past few decades due to many reasons like the gigantic amount of digital and online documents that are easily accessible and the increased demand to organize and retrieve these documents efficiently. The fast expansion of the Internet globally also has increased the need for more text categorization systems. Efficient text categorization systems are beneficial for many applications, for example, information retrieval, classification of news stories, text filtering, categorization of incoming e-mail messages and memos, and classification of Web pages. A large number of machine learning, knowledge engineering, and probabilistic-based methods have been proposed for TC. The proposed TC method has been implemented and evaluated by a large number of experiments on two benchmark text collections 20NewsGroups and Reuters. The most popular methods include Bayesian probabilistic methods, regression models, example-based classification, decision trees, decision rules, Rocchio method, neural networks, support vector machines (SVM), and association rules mining In the vector space model (VSM), the content of a document is represented as a vector in the term space, i.e., d = {w1; . . . ; w_k}, where k is the term (feature) set size. Terms can be at various levels, such as syllables, words, phrases, or any other complicated semantic and/or syntactic indexing units used to identify the contents of a text. Different terms have different importance in a text, thus an important indicator w_i (usually between 0 and 1) represents how much the term t_i contributes to the semantics of document d. The term weighting method is such an important step to improve the effectiveness of TC by assigning appropriate weights to terms. Although TC has been intensively studied for several decades, the term weighting methods for TC are usually borrowed from the traditional information retrieval (IR) field, for example, the simplest binary representation, the most famous tf:idf, and its various variants. Recently, the study of term weighting methods for TC has gained increasing attention. In contrast to IR, TC is a supervised learning task as it makes use of prior information on the membership of training documents in predefined categories. This known information is effective and has been widely used for the feature selection [1] and the construction of text classifier to improve the performance of the system. In this study, we group the term weighting methods into two categories according to whether the method involves this prior information, i.e., supervised term weighting method (if it uses this known membership information) and unsupervised term weighting method (if it does not use this information). Generally, the supervised term weighting methods adopt this known information in several ways. One approach is to weight terms by using feature selection metrics they are naturally thought to be of great help to assign appropriate weights to terms in TC. Another approach is based on statistical confidence intervals [4], which rely on the prior knowledge of the statistical information in the labeled training data. Nevertheless, there is another approach that combines the term weighting method with a text classifier [5]. Similar to the idea of using feature selection scores, the scores used by the text classifier aim to distinguish the documents. Since these supervised term weighting methods take the document distribution into consideration, they are naturally expected to be superior to the unsupervised (traditional) term weighting methods. However, not much work has been done on their comprehensive comparison with unsupervised term weighting methods. Although there are partial comparisons in [3] and [4], these supervised term weighting methods have been shown to have mixed results. .

ISSN: 2277-3754
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 1, Issue 3, March 2012

II. LITERATURE REVIEW

There are various works done in text categorization till date, as text categorization can be done in both ways supervised and unsupervised there are various ways such as Racchio, decision trees, Naïve Bayes, SVM, etc. but according to results of all these methods it is proven that SVM is one the best method used for Text Categorization by using bag-of-words. Decision Tree as one of the method uses information gain factor for text categorization which categorizes documents on the basis of combination of word occurrences for which it uses the tree based methods such as ID-3 and C4.5. but in this particular way as it is using combination of words most of words in English language are used in different ways according to the need of statement, hence in such cases it may give the proper results word-wise but fails to work in various situations also so we cannot state this as an effective one. These methods generate classifiers by inductive learning rule [4]. Decision tree induction classifiers are biased towards frequent classes [7].

Naïve Bayes is again one of the method of text categorization in which user can get very reasonable performance in an attractive framework [5]. In this method document is considered as a binary feature vector which is the representation of whether term is present or not. and is also known as multivariate Bernoulli naïve Bayes. but in this method there are two problems first is its rough parameter estimation and calculations are done by taking all positive documents into consideration and second is in handling categories of rare terms or insufficient data where it cannot work well[6]. Olex again one of the novel method for text categorization specially in case of automatic induction of rule based text classifier. Which needs documents to be classified into positive and negative literals? This rule allows prediction about belongingness about the terms in document, and also is an optimization problem. Non informative words are removed from the documents in order to increase the time efficiency; this uses chi^2 and information gain as one of the key method for calculating the efficiency and term goodness.

On one hand it is proved to be both effective and efficient and on other hand its local one-term-at-a-time greedy search strategy prevents it to cope with term interaction, as no two or more terms are calculated and evaluated at a time as a whole. Another problem is inconvenience with rule generation stems from the way how greedy heuristics works [7]. KNN which are the instance-based classifier do not relay on statistical distribution of training data, they cannot good positive examples. In this two different documents may be nearest neighbor even they are of different category and a vice-versa can also occur.

Racchio method also performs well after sorting the data into positive and negative categories but does not give that much efficiency this can also be proven by Table. 1 with the help of a labeling heuristic, called PNLH (Positive examples and Negative examples Labeling Heuristic) which is an extension of preliminary work in [8]. Fig. 1 shows the general framework of PNLH. It consists of two steps: extraction and enlargement [8].

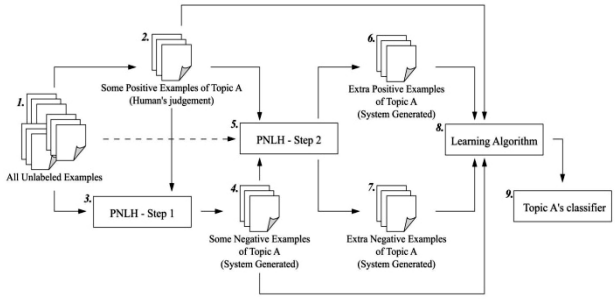

Fig. 1. General framework of PNLH

TABLE I. Results of 20Newsgroup

%	Without *PNLH*			With *PNLH*		
	SVM	Rocchio	NB	SVM	Rocchio	NB
1	0.000	**0.001**	0.000	0.000	0.000	0.000
2	0.000	0.007	0.001	**0.026**	**0.022**	**0.013**
3	0.000	0.016	0.002	**0.040**	**0.032**	**0.019**
4	0.000	0.034	0.005	**0.069**	**0.064**	**0.032**
5	0.000	0.036	0.010	**0.094**	**0.105**	**0.072**
6	0.000	0.055	0.021	**0.185**	**0.182**	**0.141**
7	0.000	0.044	0.024	**0.237**	**0.205**	**0.215**
8	0.000	0.049	0.034	**0.274**	**0.254**	**0.252**
9	0.000	0.069	0.046	**0.314**	**0.305**	**0.272**
10	0.001	0.103	0.070	**0.343**	**0.405**	**0.352**
20	0.013	0.290	0.185	**0.423**	**0.451**	**0.385**
30	0.029	0.365	0.299	**0.444**	**0.461**	**0.401**
40	0.073	0.514	**0.455**	**0.437**	**0.525**	0.452
50	0.162	0.618	0.532	**0.450**	**0.615**	**0.544**
60	0.301	0.659	0.610	**0.481**	**0.665**	**0.613**
70	0.438	**0.691**	**0.643**	**0.535**	0.684	0.623
80	0.573	**0.702**	**0.671**	**0.632**	0.684	0.642
90	**0.671**	**0.711**	**0.703**	0.662	0.695	0.654
100	**0.761**	**0.711**	**0.724**	0.722	0.701	0.695

From the above table it can be concluded that PNLH really affect the contents of the document for getting classified even if we use the same classifier techniques and same term weighting methods. These findings are again verified through one more method introduced by Hisham Al- Mubaid and Syed A. umair, Lsquare using distributional clustering and learning logic. This is mainly focused on word feature and feature clustering, generates different separating and nested sets. This gives the results as good as the SVM with the differentiation of very few points [4].

TABLE II.Accuracy for Lsquare and SVM on hundred examples of six categories of 20NG

Training Size = 100 doc	Lsquare	SVM
Category	**Accuracy**	**Accuracy**
alt.atheism	98.74	98.46
comp.windows.x	99.61	98.76
rec.autos	99.47	99.33
rec.sports.hockey	99.72	99.21
sci.electonics	97.60	99.74
talk.politics.guns	98.64	97.79
Average	**98.96**	**98.88**

There are total 20 categories of 20 newsgroup database out of which they have selected only 6 categories which are relevant to some other categories as well but we cannot say that they are very closely related to each other. And even the

ISSN: 2277-3754
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 1, Issue 3, March 2012

Lsquare method gives the better results than the SVM as mentioned above but they are not based on all the 20 categories that is why it may happen that the remaining categories will give the better results in SVM and will increase the average.

III. Implementation Design

A. Why SVM for Text Categorization

Even though there are various methods for text categorization which works differently according to the method parameters, but this is also true that these values changes or works according to the text hence it is necessary to check the properties of text. Some of which are listed below:

- Linear separability:most of categories of assumed database are linearly separable hence classifier need to find it out first and so are many of the Reuters tasks also where the idea of SVM is to find such linear separators.
- Document vectors are sparse: for each document there are some entries which are not zero. Kivinen, Warmuth and Auer [8] gives both the theoretical and empirical evidences for the mistakes bound model that additive algorithms, which have similar inductive bias like SVMs are well suited for problems like dense concepts and sparse instances.
- High dimensional input space: When we actually categorize the text we come across many features and SVM gives over fitting protection which does not depends on the number of features. They have potential to handle large number of feature spaces.
- Irrelevant features: To avoid the above stated problem one way is this and will done through feature selection. Through text categorization we get very few relevant features according to their information gain factor also many time even word occurring very few times gives the more relevant information. So the good classifier must combine many features and this aggressive selection may lead to loss of information and SVM gives many parameters for feature selection, which may avoid this up to great extent[5][12].

Another two reasons to use SVM are one; it is based on simple ideas and provides clear intuition of what we are exactly learning from those examples. Second is it performs very well in practical applications and complex algorithm of feature extraction. It contains large class of neural nets, radial basic function and polynomial classifier as special cases also maps data into some other feature space. It uses inductive learning technique to automatically construct classifier using labeled training data and provide good accuracy. This is very effective in real world classification [9]. These parameters are very important because categorization goes through various hierarchical classification method such as threshold reduction, restricted voting and extended multiplicative. In which the blocking reduction techniques are also used [11].

B. Term Weighting Methods: A Brief Review

In text representation terms are words or phrases or any indexing term and each is represented by a value whose measure gives the importance of that term. All documents are categorized to get the features and those features acts as an keyword. The frequency of keyword that is the number of times the particular term occurs in the document is denoted by tf (term frequency), likewise there are various terms which gives frequency count of keywords which are given in table II as given below.

TABLE III.Term Frequency Factor

Term frequency factor	Denoted by	Description
1.0	Binary	Binary weight =1 for terms present in a vector
Tf alone	Tf	Row term frequency(no of times term occurs in a document)
Log(1+tf)	Log tf	Logarithm of a term frequency
1-(1/(1+tf))	ITF	Inverse term frequency usually tf-1
ntf	ntf	Normalized term frequency

In Table I above four are commonly used term frequency factor in which binary term gives only the presence of term by 0 and 1 but it does not give importance of term hence we cannot use this in feature generation this is used in Naïve Bayes and Decision Trees also. Next is most popular term frequency representation which adopts raw term frequency however different variants of this also gives log(1+tf) which is nearly as same as log(tf) this is used to scale the unfavorably high term frequency in the documents[1], this is again reduced to certain extend by formula 1-(1/(1+tf)) known as inverse term frequency but this frequency factor is not as effective as an term frequency, no doubt it reduces the value of term frequency when it is high but it does not support to the new input document of classifier to categorize if it is not exactly the keyword but very close to the keyword. In that case we need to use the term frequency but if we do so again we need to minimize the unfavorable high value so another solution that we are proposing in this is normalized term frequency factor denoted as ntf. Which is given by the equation (1) where i is the keyword that we want to search for and j is the document in which it occurs, while k_i is the maximum times occurring keyword in that document which may be or may not be same as i.

$$ntf = \frac{freq(i,j)}{\max k \in T\ freq(k_i,j)} \quad (1)$$

This gives the normalized term frequency of the document. There are various term weighting methods used for text categorization before this which are listed in table IV, if we go according to that we need to calculated few parameters such as information gain (ig), odds ratio (OR), chi square(x^2), relevant frequency (rf) and inverse document frequency (idf) this calculation is done by dividing the documents in to positive and negative categories and all the calculations are done on the term basis. Suppose there are six terms t1, t2, t3, t4, t5 and t6 as shown in Figure 2 Given one chosen positive

ISSN: 2277-3754
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 1, Issue 3, March 2012

category on data collection. Each column represents document distribution in the corpus for each term and height is number of documents.

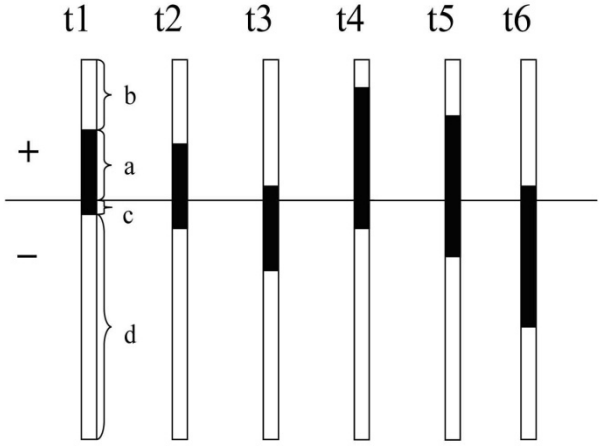

Fig. 2. Examples of different distributions of documents that contain six terms in the whole collection

The horizontal line divides these documents into two categories the positive (above) and negative (below) the heights of the column above and below the horizontal line denote the number of documents in the positive and negative categories, respectively. The height of the shaded part is the number of documents that contains this term we use a, b, c and d to denote the number of different document as listed below

- a to denote the number of documents in positive category that contains this term
- b to denote the number of documents in positive category that do not contains this term
- c to denote the number of documents in negative category that contains this term
- d to denote the number of documents in negative category that do not contains this term[1]

$$x^2 = N * \frac{[(a*d)-(b*c)]^2}{(a+c)(b+d)(a+b)(c+d)} \quad (2)$$

$$gr = \frac{ig}{-\frac{(a+b)}{N} * log\left(\frac{a+b}{N}\right) - \frac{c+d}{N} - log\left(\frac{c+d}{N}\right)} \quad (3)$$

$$ig = N * \frac{a}{N} * \frac{\log(a*N)}{(a+c)(a+b)} + \frac{b}{N} * \frac{\log(b*N)}{(b+d)(a+b)} + \frac{c}{N} * \frac{\log(c*N)}{(a+c)(c+d)} + \frac{d}{N} * \frac{\log(d*N)}{(b+d)(c+d)} \quad (4)$$

With the help of above parameters one will be able to calculate the term weighting method. After observing the figure 2 and parameters such as a, b, c and d we also come to know that along with the term frequency it is necessary to find inverse document frequency as well. This can be calculated in two different ways either by equation (5) or (6)

$$idf = \log \frac{N}{(a+c)} \quad (5)$$

$$idf = \log \frac{N}{n} \quad (6)$$

TABLE .IV.Summary of Nine Different Term Weighting Methods

Methods	Denoted by	Description
Unsupervised Term Weighting	Binary	1 for presence, 0 for absence
	tf	Term frequency alone
	tf.idf	Classic tf and idf
Supervised Term Weighting	tf.rf	Term and relevant frequency
	rf	relevant frequency alone
	tf.x^2	Tf and chi square
	tf.ig	Tf and information gain
	tf.logOR	Tf and logarithm of odds ratio
	ntf.idf	Proposed method

Now after getting normalized term frequency you need to find for relevant frequency which need two main parameters one that the number of documents in positive category that contains this term that is a, and the number of documents in negative category that contains this term that is c, based on these two parameters relevant frequency is calculated as given in equation (7)

$$rf = \log\left(2 + \frac{a}{c}\right) \quad (7)$$

By looking at above equation in worst case it may happen that there are no such documents in which the given term is not occurring at all in that case the denominator will become zero and will lead to divide by zero error hence in that case the another option is chose one instead of c, and equation (7) will be modified as equation (8) as given below

$$rf = \log\left(2 + \frac{a}{\max.(1,c)}\right) \quad (8)$$

As the term weighting methods described in table 4 we can go to the calculation of our proposed term weighting method that calculating ntf and rf and multiplying the both which will generate the new term weighting method. When we make term frequency as an normalized one it gives the frequency in range of 0 to 1 which we can call as an normalized one and restrict to unfavorable term count values that is why chosen normalized frequency rather that term frequency when we combine that with idf we get the weights of the terms which generate the vector space model [1].

IV. IMPLEMENTATION DESIGN

The document to be get categorized needs to be given to classifier and to do these weights of the terms should be calculated which generates the vector space model (VSM) and this will act as an input to the classifier. For processing the text documents when input is read word by word and filtering is done from this filtered words stop words should be removed as they does not make any sense for the categorization of documents , after removing stop words all the words should be brought into their actual verb forms so

ISSN: 2277-3754
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 1, Issue 3, March 2012

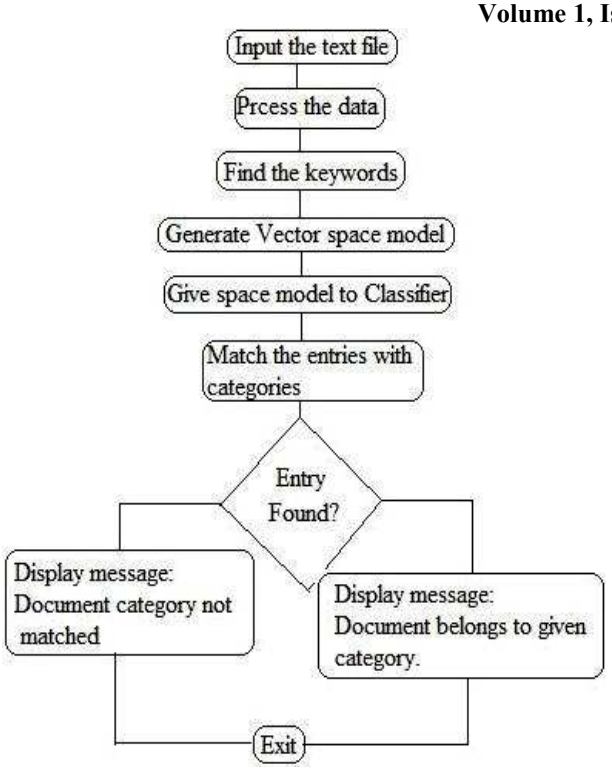

Fig. 3. Flow of the categorization process

that it will be easy to count the number of words in proper format because the word connecting and connected points to the same verb connect, likewise presenting and presented both points to the same word to avoid this type of ambiguity into words the stemming is the most important phase of the preprocessing the documents. These steps should be followed in proper sequence as shown in figure 3. And then the term frequency which will be the actual count of number of times the particular term occurring into the document. Once you calculate the term frequency one can go for normalized term frequency (ntf). By looking at the document as whole we can also find the inverse document frequency (idf) and multiplication of both ntf and idf we get the vector space model as shown in figure 4. And the weights are given to Support Vector Machines for further processing and results are calculated and seen.

V. RESULTS

A. Result on 20Newsgroups

The 20 Newsgroups corpus3 is a collection of approximate20, 000 newsgroup documents nearly evenly divided among 20 discussion groups. Some newsgroups are very closely related to each other, for example, the category of comp.sys.ibm.pc.hardware and category of comp.sys.mac.hardware. However, others are highly unrelated, for example, the category of misc.forsale and the category of soc.religion.Christian. After removing duplicates and headers, the remaining 18,846 documents are sorted and partitioned by date into 11,314 training documents (about 60 percent) and 7,532 test documents (about 40 percent). Therefore, compared with the skewed category distribution in the Reuters corpus, the 20 categories in the 20 Newsgroups corpus are of approximate uniform distribution. The resulting vocabulary, after removing stop words (513 stop words) and words that occur less than 3 and 6 times in the positive and negative categories, respectively, has 50,088 words. According to the x^2 statistics metric, the top p Є {5, 25, 50, 75, 100, 150, 200, 250, 300, 400, and 500} features are tried [1] [7].

TABLE V.Results of classification accuracy on 20 Newsgroup dataset

Category	SVM	kNN
alt.athism	8.5	7.6
c.graphics	8	7.6
c.windows.misc	7.7	6.7
c.pc.hardware	7.4	6.1
c.mac.hardware	8.7	7.5
c.windows.x	8.9	8
misc.forsale	8.9	7.7
rec.autos	9.3	9
rec.motorcycle	9.8	9.5
rec.s.baseball	9.8	9.5
rec.hockey	9.9	9.5
sci.crypt	9.6	9.3
sci.electronics	7.7	6.2
sci.med	9.3	8.9
sci.space	9.6	9.1
soc.r.christian	9	8.1
talk.p.guns	8	8.3
talk.p.midest	4.5	9.4
talk.p.misc	7	6.6
talk.rel.misc	6.4	5.7

Here the comparison on all categorize of SVM is done by two different classification techniques, SVM and kNN taking all categories of SVM into consideration.

B. Result on Reuters-21578

The documents from the top 10 largest categories of the Reuters-21578 document collections are used. According to the ModApte split, 9,980 news stories have been partitioned into a training set of 7,193 documents and a test set of 2,787 documents. Stop words (292 stop words), punctuation, and numbers are removed. The Porter's stemming is done to reduce words to their base forms. The resulting vocabulary has 15,937 words. By using the x^2 metric, the top p Є {25, 50, 75, 150, 300, 600, 900, 1200, 1800, 2400} features per category are tried. Since SVMs have the capability to deal with high-dimensional features, the previous studies showed that feature selection does not improve or even slightly degrades the SVM performance in and. We also conduct experiments by inputting the full words (after removing stop words, stemming, and setting minimal term length as 4) without feature selection. One issue of the Reuters corpus is the skewed category distribution problem. Among the top 10 categories which have 7,193 training documents, the most common category (earn) has a training set frequency of 2,877 (40 percent), but 80 percent of the categories have less than 7.5 percent instances [1] [7].

ISSN: 2277-3754
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 1, Issue 3, March 2012

TABLE VI.Results of classification accuracy on Reuters dataset

Category	SVM	kNN
acq	9	6.3
bop	7.5	4.4
carcass	8	6.9
cocoa	9.8	9.5
coffee	9.3	8.1
corn	9.3	7.8
cpi	7	5.5
crude	9	8.8
dir	7.6	6.5
earn	8.5	8
grip	8.7	7.6
gold	8	6.2
grain	9.8	8.7
interest	7.9	5.8
livestock	4.3	3.9
money	7.2	6.2
money supply	7.8	4.1
nat gas	6	6
oil seed	6.9	6.4
reserves	8.5	7
ship	8.9	8
soybean	8	6.9
sugar	9	8.1
trade	7	5.2
veg oil	7.8	6.1
wheat	9.2	8.1

VI. CONCLUSION

The following conclusion gives the answers to questions stated in the very first section. Answer to first question that is the performance of term weighting methods specially unsupervised term weighting methods has close relationship with learning algorithms and data corpora and we can also state that ntf.rf and ntf.idf gives better performance than tf.rf and tf has no relationship with algorithm and data corpora. Ntf performs very well in all cases. Answer to third question there is no doubt that tf.rf performs well is proved by all evidences but this is not well suited when there are large number of categories and more number of keywords hence in that case ntf.rf and even more that that ntf.idf is well doing. And a good text categorization can be performed.

REFERENCES

[1] Man lan, Chew lim Tan, senier member, IEEE, Jian Su and Yue Lu, member, IEEE "Supervised and traditional term weighting method for Automatic Text Categorization" IEEE Trans. on pattern Analysis and Machine Intelligence, Vol 31, no 4, April 2009

[2] Z-H. Deng, S.-W. Tang, D.-Q. Yang, M.- M. Zhang, L.-Y. Li, and K.Q. Xie, "A Comparative Study on Feature Weight in Text Categorization," Proc. Asia-Pacific Web Conf., vol. 3007, pp. 588-597, 2004.

[3] F Debole and F. Sebastini, "Supervised term weighting method for automated text categorization" Proc. ACM Symp, Applied Computing, pp. 784-788, 2003

[4] P.Soucy and G. W. Mineau, "beyond TFIDF weighting for text categorization in vector space model", Proc. Int'l Joint Conf. Artifial Intelligence. Pp. 1130-1135, 2005

[5] Thorsten Joachims, "Text Categorization with Support Vector Machines: Learning with many relevant features" Universitat Dortmund informatics ls8, baropar, 44221 Dortmund, Germany

[6] Sang-Bum, Kim, Kyong-Soo Han, hai-Chang Rim, and Sung Hyon Myaeng," Some effective techniques for naïve Bayes text Classification" IEEE Tran. On Knowledge and Data Engineering Vol 18, no 11, Nov 2006

[7] Pasquale Rullo, Veronica Lucia Policicchio, Chiara Cumbo, and Salvature liritano," Olex: Effective Rule Learning for Text Categorization", IEEE Tran. On Knowledge and Data Engineering Vol 21, no 8, Aug 2009

[8] Hisham al-Mubaid and Syed A. Umair,"A new Text Categorization Technique using Distributional Clustering and learning Logic" IEEE Tran. On Knowledge and Data Engineering Vol 18, no 9, Sept 2006

[9] Marti A. Hearst, University of California, Berkeley, "Support Vector Machine"

[10] Thornsten Joachims and Megh Meel Gure "Text Categorization with support vector machines: Learning with Many Relevant Features"

[11] Aixin Sun, Ee-Peng Lim, senier member, IEEE, Wee-Keong Ng, Member, IEEE Computer Society and Jaideeep srivastava, Fellow, IEEE, " Blocking Reduction Strategies in Hierarchical text Classification", IEEE trans. On Knowledge and Data Engineering Vol 16, no 10, Oct 2004

[12] Axin Sun, Ee-Peng Lim and Ying Liu, "What makes Categories difficult to Classify? A study on predicting classification performance for Categories"