ADVANCES in NATURAL and APPLIED SCIENCES

ISSN: 1995-0772
EISSN: 1998-1090
2016 April 10(4): pages 64-73

Published BY **AENSI Publication**
http://www.aensiweb.com/ANAS
Open Access Journal

Efficient Feature Extraction for Text Mining

[1]Maruthu Pandi.J, M.E,(Ph.D), [2]Dr.Vimala Devi.K, M.E Ph.D and [3]Anitha.V, M.Tech

[1]Department of IT Mepco Schlenk Engg. College Sivakasi, Tamilnadu, India.
[2]Department of CSE PSR Engineering College Sivakasi, Tamilnadu, India.
[3]Department of IT Mepco Schlenk Engg. College Sivakasi, Tamilnadu, India.

Received 25 February 2016; Accepted 10 April 2016; Available 15 April 2016

Address For Correspondence:
Maruthu Pandi.J, M.E,(Ph.D), Department of IT Mepco Schlenk Engg. College Sivakasi, Tamilnadu, India.
E-mail: maruthupandi@mepcoeng.ac.in

ABSTRACT

Now-a-days growth of data is so fast that we have been put in a situation to handle voluminous amounts of data. To make use of the data for retrieving some valuable information, efficient strategies should be devised. In the present work, we focus on text mining for retrieving apt information from voluminous documents when given input keywords. The retrieval process uses appropriate features for selecting the necessary data accurately. Existing methods using term based and pattern based approaches either encounter problems like polysemy and synonymy or ineffectiveness on application to a vast range of documents. With a view to settle this botheration, given an input query, the training documents need to be split into relevant and irrelevant ones. Hence we propose to use, support and specificity values for the input query in every training document. A new pattern based feature clustering method is employed to divide the documents into three major categories using min and max specificity values. We use five standard performance metrics in text mining to ensure the efficiency of our approach. We also calculated different measures of Efficiency for performing micro reviews to retrieve the maximum number of right relevant features.

KEYWORDS: polysemy, synonymy, specificity, performance metrics.

INTRODUCTION

For performing mining in the text datasets, the need for useful feature extraction from both relevant and irrelevant documents is crucial, which is a challenging task in both ideological and experimental point of view. This botheration was a key source of attention in Web personalized applications, Data Mining, Machine Learning, Information Retrieval and Web Intelligence Communities [4]. Pattern mining approaches encounter two problems. The first one is regarding the deployed support. A lengthier pattern though more specific appears in documents with low support. But on reducing the support further only result in noisy patterns. The second one is that both support and confidence measures are not many times suitable.

Thus the method to explore the extracted features for calculating term weights lies as an open issue. Existing term based approaches like PTM (Pattern taxonomy mining) use sequential pattern mining for weighing useful features [3]. Concept based models(CBM) with natural language processing techniques used noun-verb structures for useful feature extraction. Though it showed improvements it lacked effective integration in both relevant and irrelevant documents. Hybrid approaches which combine document ranking, information filtering and text classification were used over the years. For extracting from relevant and unlabelled documents, a two stage classifier with Rocchio classifier at stage 1(to remove irrelevant documents from unlabelled sets) and SVM classifier at stage 2 (to classify text documents) were used. Larger weight terms tend to be more general in both relevant and irrelevant ones. For example, the word package is more general whereas the term sdk is more specific. Therefore the solution is to use specificity along with term distribution. Specificity describes the degree to which the focus is made on the topic which the user wants. Initially specificity score was calculated based on the term appearance in both relevant and irrelevant ones but with the need of iterative loops to calculate the accurate term weights.

As a revolution to the previous approaches, the first version of RFD used manual setting of parameters with a definition of specificity and calculated the term weights after grouping the terms into positive, negative and general terms. For large datasets, separate sliding window approach is used for calculating for each part of the full dataset and taking the average. The second version of RFD used FClustering approach to automatically cluster the terms into positive, negative and general terms. Irrelevant documents identification and weight revision is also made. Now we propose an enhanced version of RFD to increase the document count of relevant documents and use SVM to classify the features. The advantages of the proposed model includes,

- Rank calculation for the relevant documents in sorted order
- Dictionary approach to handle synonymy and polysemy
- SVM classification of the features
- Enhancement of relevant documents count using average of min and max support values
- Testing using Harmonic Mean, min, average, sack potency
- Multiple Criteria for relevancy such as support, term weight, rank and pointer
- Visualization using Tree Structure
- Micro review to find profit, price and selecting the audit with highest profit to-price ratio

Our proposed approach is tested using the standard performance metrics in text mining namely: F beta measure, IAP, MAP and break-even point. The results show an enhanced achievement over the state of art methods.

The paper shows related works in Section 2.Section 3 contains the explanation of the proposed enhanced version of RFD. Section 4 describes the effective algorithm for Feature clustering and Section 5 shows the evaluation and performance tests.

Related Work:

Feature selection is different from feature extraction.Before performing classification or clustering [2], accurate feature selection is necessary [8],[10]. Classifiers like Rocchio, SVM, Naive Bayes, kNN [7] have been invented. A solution for multi class problem is to use composition of binary classifiers then assigning it to one of the two(relevant or irrelevant)[1].Bag of words (multiset) is also used in traditional approaches for multi class problem. Feature selection criteria also include DF,global IDF, information gain, mutual information, Chi-square and term strength. Selection of relevant features being a single class problem was not solved using traditional approaches [11] [12]. Relevance score which is given using term weights reveals the aptness of retrieved results for an input query. The existing works of term based ranking approaches include tf-idf, rocchio, probabilistic models and okapi BM25.

Redundancy and noise are the two main problems of relevance feature discovery. Text features might range from simple words to complex structures for example, n-grams which carried better semantics than words considered using unigram, bigrams or trigrams for calculating weights[9][6].CBM model used three steps: first to analyze the sentence structure, second to use an ontological graphical representation[5] and lastly to construct feature vectors. Pattern based approaches used algorithms like apriori, prefix span, FP-tree, SPADE, SLPMiner and GST. Again the challenge is to handle vast and redundant terms. Comparing with traditional IR models, closed patterns with pattern deployment method to map into a term vector proved better beneficial results[3] [13]. But still low frequency problem exists. Dimensionality problem is reduced by using a LDA clustering based feature selection methods.

To summarize the discussion into three categories, the first approach estimates the weights of those terms which appear in both relevant and irrelevant. The second one focuses on the frequency of term appearance and the third one groups the features into three clusters-relevant,irrelevant and general terms. Our proposed approach enhances the third one.

RFD-Module Description:

Here in this subdivision, we discuss about the proposed innovation of discovering the features of relevancy. The training documents were first split into relevant and irrelevant ones based on the appearance and frequency of the query term in the documents and then the support and specificity of the query term in each of the training documents is calculated. We use FClustering algorithm to divide the training set into three categories: positive specific terms(T^+), general terms(G) and negative specific terms(T^-). Here, G, T^+, T- are disjoint sets and hence specificity is a good partitioning measure. Then we calculate the term weights followed by evaluation using standard performance metrics in text mining.

Coverage$^+$ denotes the set of relevant documents that contain term t

Coverage$^-$ denotes the set of irrelevant documents that contain term t

Positive Specific Terms (T^+) denotes the set of Terms frequently occurring in relevant documents

Negative Specific Terms(T^-)denotes the set of Terms frequently occurring in irrelevant documents

General Terms (G) denotes the set of Terms frequently occurring in both relevant and irrelevant documents

Given a query, example ”flower”, we can infer related terms to flower as tree structure is used.

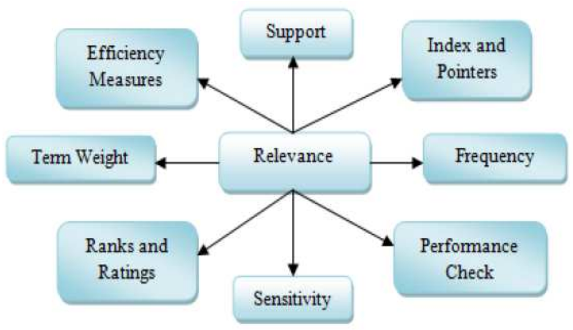


Fig. 1: Efficient Feature Selection Criteria

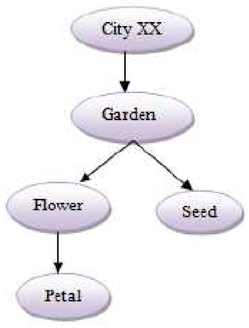


Fig. 2: Tree Structure for Query "Flower".

3.1 Training:

The input documents were read and separated into sentences. The sentences were separated into words. Data cleaning and preprocessing is done to eliminate stop words.

3.2 D_Support:

If SP_1, SP_2….$SP_{|D+|}$ represent the sets of identified closed sequential patterns for all documents such that $d_i \in D^+$(i=1…n. Here, n represents the set of relevant documents D^+). The deployment support is calculated as

$$d\_supp(t,D^+)=\sum_{i=1}^{n} \frac{|\{p|p \in SPi, t \in p|}{\sum_{p \in SPi} |p|} \quad (1)$$

The numerator sums the terms belonging to Sequential Pattern and the denominator gives the summation of the matching term count

3.3 Specificity:

The specificity value for the query term in each document is calculated after separating the input documents into D^+ cluster and D^- cluster. The D^+ cluster refers to the documents containing relevant information regarding the input query. The D^- cluster refers to the documents containing irrelevant information regarding the input query.

Specificity is the degree to which a term focuses on the subject the user needs. If a set of terms from D^- is represented as $T_{2,}$such that $T=T_1 \cup T_2$ and $D=D^+ \cup D^-$,then for a term t that belongs to T, the specificity in each document is calculated using:

$$Spe(t)=|\{d|d \in D^+,t \in d\}|-|\{d|d \in D^-,t \in d\}| /n \quad (2)$$

$|\{d|d \in D^+, t \in d\}|$ denotes the documents that belongs to $D^+\{d|d \in D^-, t \in d\}|$ denotes the documents that belongs to D^-n denotes number of relevant documents in the training set

$Coverage^+(t)=\{d|d \in D^+|t \in d\}$, $Coverage^-(t)=\{d|d \in D^-|t \in d\}$ and $n=|D^+|$. If spe(t) > 0 it means that the term t is a frequently occurring one in the relevant documents than the irrelevant ones. This relative specificity is fair when the absolute specificity of T^+ is greater that of G

3.4 F-Clustering:

The documents were then divided into three clusters T^+, T^- and G. The T^+ cluster contains the more related documents from the input documents. The T^- cluster contains the most irrelevant document from the given dataset. The G cluster contains the common documents that contains the terms related to input document.

Calculating The Cluster Difference:

If c_i and c_j denote two clusters, their cluster difference is calculated as:

$dif(c_i,c_j)=min\{|max_{spe}(c_i),min_{spe}(c_j)|,|max_{spe}(c_j)-min_{spe}(c_i)|\}$ (3)

$max_{spe}(c_i)$ denotes the Maximum Specificity Value

$min_{spe}(c_j)$denotes the Minimum Specificity Value

3.5 Weight Calculation:

The weight w(t) will take any of the equations below based on the type of cluster they belong.

$d\_sup(t,D^+)$ $(1+spe(t))$ if $t \in T+$ $d\_sup(t,D^+)$ if $t \in G$

$d\_sup(t,D^+)$ $(1-|spe(t)|)$ if $t \in T1$ $-d\_sup(t,D^+)$ $(1+|spe(t)|)$ otherwise (4)

3.6 Rank Calculation:

A major challenge is to select the irrelevant documents from the training set, since the sample size is very large. As for a case, consider a normal web search using a search engine. A billion documents are retrieved but only a few of them are actually the ones which the user is in need. So the normal approach is to give ranks to the retrieved results based on the term‘s appearance in the documents.

$rank(d)= w (t) . \tau(t, d)$ (5)

The function $\tau(t, d)$ is defined as follows:

$\tau(t,d)=1$ if $t \in d$,

$\tau(t,d)=0$ otherwise

An offender is defined as an irrelevant document with high rank.

3.7 Audits And Tips:

The set of tips T with t tips covered by atleast a single sentence s of audit A having a co-incident function F,

$TA= \{t \in T :s \in A, F(s,t) =1 \}$ (6)

The coverage of the collection of audits C is given by

$Cov (C) = |UA \in C\ TA| / |T|$ (7)

3.8 Potency:

The potency measure is to detect the highest number of relevant features from the dataset for the given query. If for a audit, Relr represents the collection of relevant sentences covering a minimum of one tip, potency is given as,

$Pot(A) = |Relr| / |A|$ (8)

3.9 Min Potency:

Min potency takes the audit A with minimum potency value in the audit collection C.

$Potmin (C) = minA \in C\ Pot (A)$ (9)

3.10 Average Potency:

Average potency of a audit collection C is given by dividing the summation of the potency values of the audit by the number of audits in the collection.

$Potavg (C) = A \in C\ Pot(A) / |C|$ (10)

3.11 Sack Potency:

If xi is a binary integer variable associated with an audit, with a parameter $\propto$the the sack potency is given as

$Sack\ Pot(A) = A - \propto A\ xi \geq 0\ ni =1$ (11)

3.12 Harmonic Mean:

Harmonic mean represents the ratio of the average of coverage to the average potency of the input.

$HMean =2 * Cov\ C * Pot\ avg\ (C)\ Cov\ C + Pot\ \ avg\ (C)$ (12)

4 Methodology:

4.1 Algorithm I:

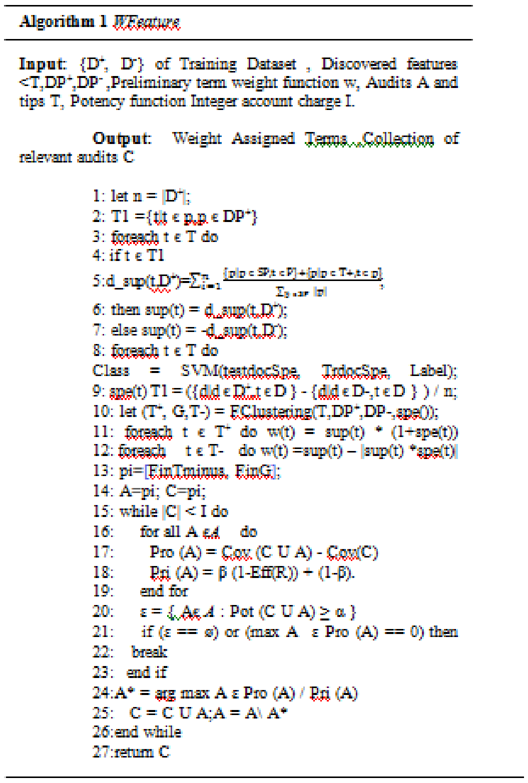
Wfeature Algorithm ( ):

Steps:

- Evaluate Support and Specificity
- Make function call to FClustering Algorithm for obtaining 3 term categories

- Weight Calculation using weight function
- Find profit and price and obtain A*,the maximum relevant features

4.1.1 Algorithm I Pseudo Code:



Algorithm 1 *WFeature*

Input: $\{D^+, D^-\}$ of Training Dataset , Discovered features <T,DP$^+$,DP$^-$,Preliminary term weight function w, Audits A and tips T, Potency function Integer account charge I.

Output: Weight Assigned Terms, Collection of relevant audits C

```
1: let n = |D+|;
2: T1 ={t|t ϵ p,p ϵ DP+}
3: foreach t ϵ T do
4: if t ϵ T1
5:d_sup(t,D+)=∑ [illegible] ;
6: then sup(t) = d_sup(t,D+);
7: else sup(t) = -d_sup(t,D-);
8: foreach t ϵ T do
Class = SVM(testdocSpe, TrdocSpe, Label);
9: spe(t) T1 = ({d|d ϵ D+,t ϵ D } - {d|d ϵ D-,t ϵ D } ) / n;
10: let (T+, G,T-) = FClustering(T,DP+,DP-,spe());
11: foreach t ϵ T+ do w(t) = sup(t) * (1+spe(t))
12: foreach   t ϵ T-   do w(t) =sup(t) – |sup(t) *spe(t)|
13: pi=[FinTminus, FinG];
14: A=pi; C=pi;
15: while |C| < I do
16:     for all A ε A    do
17:       Pro (A) = Cov (C U A) - Cov(C)
18:       Pri (A) = β (1-Eff(R)) + (1-β).
19:     end for
20:     ε = { Aε A : Pot (C U A) ≥ α }
21:     if (ε == ∅) or (max A  ε Pro (A) == 0) then
22:   break
23:   end if
24:A* = arg max A ε Pro (A) / Pri (A)
25:   C = C U A;A = A\ A*
26:end while
27:return C
```

Assign n to the count of the number of positive documents. Then every term t that belongs to a positive pattern DP$^+$ is assigned to T1.Calculate support values using the deployment support formula depending on whether the term belongs to D$^+$ or D$^-$but for all terms belonging to the term set T. Note that we use the modified d_supp formula. The time complexity is O(n*|p|) where |p| is the average size of a pattern, and |p|<= |d. Use SVM, a promising classifier to find the appropriate class. Then calculate the specificity value for which the time complexity is O(n*|d|) to calculate spe function. Therefore, the time complexity is O((|T|*n*|p|)+(|T|* n*|d|)) = O(|T|*|d|*n).Make a function call to FClustering algorithm() to obtain the term categories. Then calculate the term weights depending on the type of term cluster T$^+$ or T$^-$. Thus WFeature( ) algorithm take a time of O(|T|*|d|*n+|T|^2) where the average document size is |d| and the count of the relevant documents is n. The algorithm has many recursions increasing the audit collection C count with new audit A. At each recursion, for a given audit calculate the profit (Pro) and price (Pri). Select A* which has the greatest Profit-to-price ratio. Here |T| < |d|; so, Algorithm WFeature is efficient.

4.1.2 Algorithm I Explanation:

Assign n to the count of the number of positive documents. Then every term t that belongs to a positive pattern DP$^+$ is assigned to T1.Calculate support values using the deployment support formula depending on whether the term belongs to D$^+$ or D$^-$ but for all terms belonging to the term set T. Note that we use the modified d_supp formula. The time complexity is O(n*|p|) where |p| is the average size of a pattern, and |p|< |d|. Then calculate the specificity value for which the time complexity isO(n*|d|) to calculate spe function. Therefore, the total time complexity is given as O((|T|*n*|p|)+(|T|*n*|d|))= O(|T|*|d|*n).Make a function call to FClustering( ) algorithm to obtain the term categories. Then calculate the term weights depending on the type of term cluster T$^+$ or T$^-$.Thus WFeature( ) algorithm take a time of O(|T|*|d|*n+|T|^2) where the average document size is |d| and the count of the relevant documents is n. Here |T| <|d|; so, Algorithm WFeature is efficient. The term count T is less than 300.

4.2 Algorithm II:
Fclustering algorithm( )
Steps:

- First assign T$^+$,T$^-$ and G to empty values
- Every term which does not belong to the set of positive terms T$^+$ is initially assigned to T$^-$ cluster.

- For now each of the m remaining terms are treated as individual clusters.
- According to $\min_{spe}$ values, sort the clusters
- Do recursive cluster merging till only 3 clusters remain

4.2.1 Algorithm II Pseudocode:

Algorithm 2 *FClustering*

Input: Identified features $< T, DP^+, DP^- >$ and spe values **Output**: Terms divided into three parts T^+, G and T^-

```
1.  $G=0, T^+=0, T^-=0;$
2.  foreach $t_i \in T$ do
3.     if $t_i \in \{t | t \in P, P \in DP^+\}$
4.     then $T^- = T^- \cup \{t_i\};$
5.  foreach $t_i \in T-T^-$ do
6.   { $C_i=\{t_i\}$;  $\max_{spe}(c_i)=\min_{spe}(c_i)=spe(t_i);$ }
7.  let $m=|T-T^-|;$
8.  let $C=\{c_1,c_2,\ldots c_m\}$
9.  $\min_{spe}(c1) >= \ldots >= \min_{spe}(c_m);$
10. while($|C|>3$)
11. {
12.     let $k=1$ and $mind=dif(c_1,c_2);$
13.     for $i=2$ to $m-1$ do
14.       if $dif(c_i,c_{i+1}) < mind$
15.         then {$k=i$; $mind = dif(c_i,c_{i+1});$}
16.     let $c_k=c_k \cup c_{k+1};$
17.    if $\min_{spe}(c_{k+1}) < \min_{spe}(c_k)$
18.     then $\min_{spe}(c_k)=\min_{spe}(c_{k+1});$
19.    if $\max_{spe}(c_{k+1}) > \max_{spe}(c_k)$
20.     then $\max_{spe}(c_k)=\max_{spe}(c_{k+1});$
21.    for $i=k+1$ to $m-1$ do
22.      let $c_i = c_{i+1};$
23. }
24. if $|C|=1$ then $T^+=c1$
25. else if $|C|=2$ then { $T^+=c1; G=c_2$}
26. else {$T^+=c_1; G=c_2; T^-=T^- \cup c_3$};
```

4.2.2 Algorithm II Explanation:

A major issue was to effectively partition the dataset into three categories, relevant, irrelevant and general for every input query term. When we do this, we can accurately discriminate the relevant documents from the irrelevant ones. To automate this category generation for every input query, we use specificity function. To start with, assign all the three clusters T^+,T^- and G to null. For every term t, belonging to the set of terms T, assigns all the terms which do not belong to DP^+ to T^- in the beginning for which t time complexity is $(O|T^2|)$. But for T-T^- (i.e.) for the terms in T^+, every term is an individual cluster. Sort these clusters based on the $\min_{spe}$ values in a time of O(mlogm). Now start the merging process which takes $O(m^2)$ time for the set of C clusters while it remains greater than 3.Assign k value as 1 and use the cluster difference formula to find the value of mind variable. When incrementing the i values to traverse the clusters, check whether the calculated cluster difference is less than the mind value and make the appropriate assignments and also remove clusters from C. If c_k is the cluster obtained by merging the two clusters c_i and c_j, we obtain the min and max values as

$$\min_{spe}(c_k)=\min\{\min_{spe}(c_i),\min_{spe}(c_j)\} \quad (12)$$

$$\max_{spe}(c_k)=\max\{\max_{spe}(c_i),\max_{spe}(c_j)\} \quad (13)$$

Finally if we have the C value as 1,the cluster is named as T^+ cluster. If the C value is 2,we obtain both T^+ and G cluster. If we obtain 3 clusters, they are T^+, G and T^- clusters. Thus the total time complexity is given as $O(|T| + m\log m + m^2) = O(|T| + m^2) = O(|T|^2)$

5 Evaluation:

5.1 Performance Analysis:

For testing the real time performance, we used standard performance metrics in text mining namely: top-20 documents, F_1 measure, mean average precision (MAP), the break-even point (b/p), and interpolated average precision (IAP) on 11-points

First we calculate the precision by using true positive and false negative values using the formula:

$$\text{Precision}, P = tp / (fp + tp) \quad (14)$$

Similarly recall used false negative values.

$$\text{Recall}, R = tp / (fn + tp) \quad (15)$$

F-beta measure combined both precision and recall. Here we took the value of beta as 1which gives equal weights to both precision and recall.

$$F_b=2(P.R)/(P+R) \quad (16)$$

Mean Average Precision which is a measure of precision at relevant documents combines precision, recall with ranking. Break Even Point tells the value of precision or recall at which the precision recall intersects the precision=recall line.11-point measure took average precision at 11 standard recall levels.

5.2 Tabulations:

The dataset used here is RCV1. We tabulated the performance metric values using RFD_1 and RFD_2.The Sliding window approach is used to divide the datasets into multiples of 25 documents and then taking the average of all. We have also tabulated our results for the retrieved training documents, number of extracted terms, and weight of extracted by taking for different K values as $D^+/2$, D^+ and D^-

Table 1: Comparing RFD_1&RFD_2.

	RFD_1	RFD_2
TOP-20	0.55	0.55
F_1	0.46	0.46
B/P	0.47	0.47
IAP	0.51	0.51
MAP	0.49	0.51

Table 2: Sliding Window Approach For RFD_1.

	RFD_1 (1)	RFD_1 (2)	RFD_1 (3)	RFD_1 (4)	AVG RFD_1
TOP-20	0.57	0.55	0.56	0.53	0.55
F_1	0.47	0.47	0.46	0.46	0.46
B/P	0.48	0.48	0.47	0.46	0.47
IAP	0.52	0.51	0.51	0.50	0.51
MAP	0.50	0.50	0.49	0.47	0.49

Table 3: Sliding Window Approach for RFD_2.

	RFD_2 (1)	RFD_2 (2)	RFD_2 (3)	RFD_2 (4)	AVG RFD_2
TOP-20	0.57	0.55	0.56	0.54	0.55
F_1	0.47	0.47	0.46	0.47	0.46
B/P	0.48	0.48	0.47	0.48	0.47
IAP	0.52	0.52	0.51	0.52	0.51
MAP	0.52	0.52	0.51	0.52	0.51

Table 4: Sliding Window Approach for RFD_2+SVM.

	RFD_2 + SVM (1)	RFD_2+ SVM (2)	RFD_2+ SVM (3)	RFD_2 + SVM (4)	AVG RFD_2+SVM
TOP-20	0.60	0.57	0.58	0.57	0.72
F_1	0.49	0.49	0.48	0.49	0.48
B/P	0.50	0.50	0.49	0.50	0.49
IAP	0.54	0.54	0.53	0.54	0.53
MAP	0.52	0.52	0.51	0.52	0.51

Table 5: Average Number Of Training Documents.

K VALUE	$D^+/2$	D^+	D^-
RELEVANT	12.6	12.6	12.6
IRRELEVANT	40.2	40.2	40.2
OFFENDERS	6.4	9.2	37.6

Table 6: Average Number of Extracted Terms.

K VALUE	$D^+/2$	D^+	D^-
T+	23.3	27.5	30.1
G	21.2	23.3	7.4
T-	230.5	265.6	520.7

Table 7: Average Weight Of Extracted Terms.

K VALUE	$D^+/2$	D^+	D^-
$W(T^+)$	3.7	3.2	2.1
W(G)	1.27	1.21	0.45
$W(T^-)$	-0.32	-0.18	-68.7

Table 8: Different Efficiency Measures.

EFFICIENCY	MIN EFFICIENCY	BAG EFFICIENCY	AVG EFFICIENCY
0.845	0.2	0.85	0.844
0.875		0.75	
0.825		0.75	
0.85		0.75	
0.825		0.845	

Table 9: Harmonic Mean Values For Different Categories.

ASPECTS	SENTENCES	ASPECTS PER SENTENCES	HMEAN
4.5	114.8	0.15	0.785 0.675 0.725 0.75 0.725

5.3 Graphs:

The dataset used here is RCV1 plotsare Precision Versus Recall. Precision and recall values are calculated using the tp, fp, tn and fn values. First we plotted the graph for a single query followed by plotting for different categories RFD_2,T^+ U G,T^+ U T^-,T^+,T^- U G, T^-, and G.Here U stands for Union.
In Figure 1, X axis represents Recall and Y axis represent Precision values
In Figure 2, X axis represents RCV1 topics and Y axis represents specificity values.

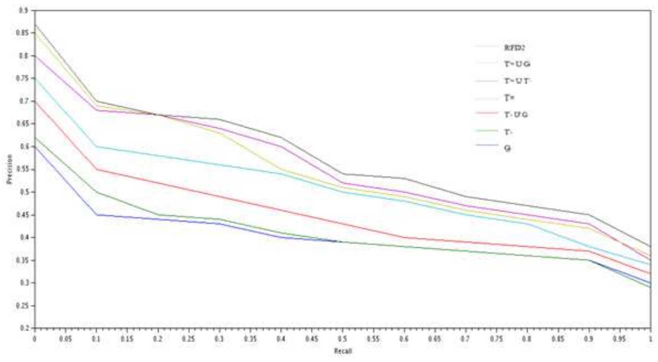


Fig. 1: Precision Recall Graph for different term categories.

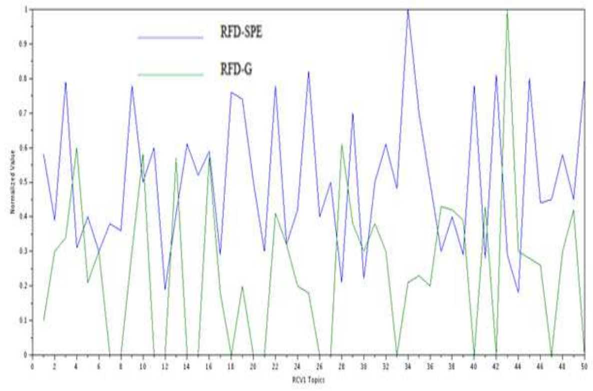


Fig. 2: RFD_2 Spe Values forSpecific and General terms.

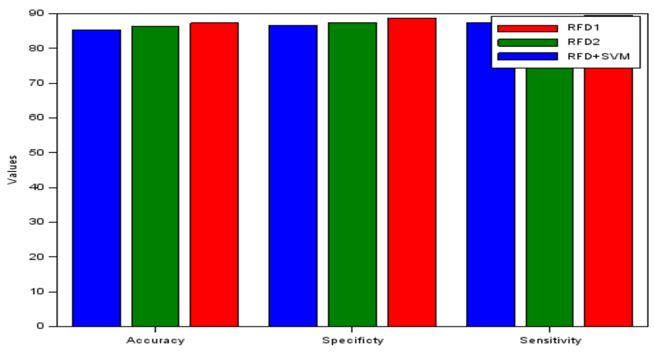


Fig. 3: Comparison of the count of the correct relevant documents retrieved using RFD_1 , RFD_2 and RFD_2 + SVM.

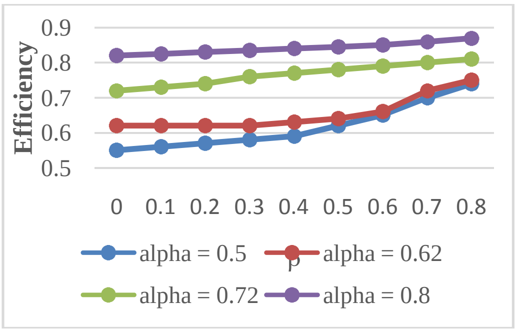


Fig. 4: Efficiency/ Potency values for different alpha and beta values.

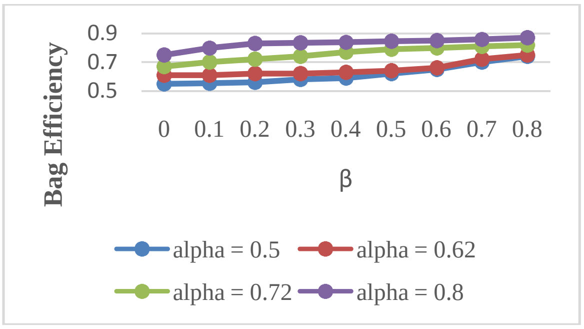


Fig. 5: Bag Efficiency/ Sack Potency Values for different values of alpha and beta.

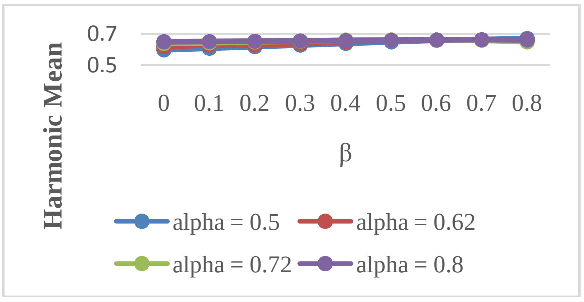


Fig. 6: Harmonic Mean values for different values of alpha and beta.

Conclusions And Future Work:

In this proposed research work we not only automated the categorization of datasets into relevant, irrelevant and general based on the input query term, we correctly removed the offenders efficiently preventing noise and irrelevancy. This helps us to discriminate the relevant documents from the irrelevant ones but also saves the user from exploring a vast range of irrelevant documents. We used FClustering algorithm to divide the training set into three categories: positive specific terms (T^+), general terms(G) and negative specific terms (T^-). Here, G, T^+,T^- are disjoint sets and hence specificity is a good partitioning measure. Then we calculated the term weights followed by evaluation using standard performance metrics in text mining and the results proved to be beneficial over the existing methodologies. Also as an enhancement in the proposed approach we have devised strategies to handle polysemy and synonymy using dictionary approach which alerts the user about the ambiguity of the query term's meaning and also retrieving documents of the query term's synonyms. We have also enhanced the frequency count of the relevant documents after ranking them by taking of min and max specificity values and classifying using SVM. We have calculated the harmonic mean, min, average and sack potency values after finding the profit, price to extract the maximum number of relevant features .Our work can be extended to handle semi supervised approach of classification where only part of the datasets have class labels.

REFERENCES

1. A novel fuzzy based clustering for multi label text categorization, 2015. Shalini Vincent M, ,Maruthu Pandi.J, Dr.Vimala Devi.K, IJAER, 10(2): 1642-1647.

2. A novel multi perspective based similarity for text categorization and clustering, 2015. Kavi Priya M, Maruthu Pandi. J, Dr.Vimala Devi.K, IJAER, 10(2): 1634-1638.
3. Zhong, N., Y. Li and S.T.Wu, 2012. "Effective pattern discovery for text mining," in IEEE Trans. Knowl. Data Eng., 24(1): 30-44.
4. Li, Y., A. Algarni and N. Zhong, 2010. "Mining positive and negative patterns for relevance feature discovery," in Proc. ACM SIGKDD Knowl. Discovery Data Mining, pp: 753-762.
5. Tao, X., Y. Li and N. Zhong, 2011. "A personalized ontology model for web information gathering," in IEEE Trans. Knowl. Data Eng., 23(4): 496-511.
6. Wang, X., H. Fang and C. Zhai, 2008. "A study of methods for negative relevance feedback," in Proc. Annu.Int. ACM SIGIR Conf. Res.Develop. Inf. Retrieval, pp: 219-226.
7. Li, X.L., B. Liu and S.K. Ng, 2007. "Learning to classify documents with only a small positive training set," in Proc. 18th Eur. Conf.Mach.Learn., pp: 201-213.
8. Scott, S. and S. Matwin, 1999. "Feature engineering for text classification," in Proc. Annu. Int. Conf. Mach. Learn., pp: 379-388.
9. Song, F. and W.B. Croft, 1999. "A general language model for information retrieval," in Proc. ACM Conf. Inf. Knowl.Manage., pp: 316-321.
10. Blum, A. and P. Langley, 1997. "Selection of relevant features and examples in machine learning,", Artif.Intell., 97(1/2): 245-271.
11. Buckley, C., G. Salton and J. Allan, 1994. "The effect of adding relevanceinformation in a relevance feedback environment," in Proc. Annu.Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval, pp: 292-300.
12. Jindal, N. and B. Liu, 2006. "Identifying comparative sentences in textdocuments," in Proc. Annu. Int. ACM SIGIR Conf. Res. Develop. Inf.Retrieval, pp: 244-251.
13. Wu, S.T., Y. Li and Y. Xu, 2006. "Deploying approaches for patternrefinement in text mining," in Proc. IEEE Conf. Data Mining, pp: 1157-1161.