NGramsSA Algorithm for Text Categorization

S.Ramasundaram

Associate Professor

MADURAI Kamaraj University College

Madurai , Tamilnadu, India

s_ramasundaram@yahoo.com

S.P.Victor

Associate Professor

St. Xavier's College Tirunelveli,

Tamilnadu,India

victorsp@rediffmail.com

ABSTRACT

Text handling has become an important task in the current age of information processing. In this text categorization (TC) is the fundamental task of handling the text documents. There are several algorithms available for implementing text categorization. This paper concentrates on proposing a new algorithm for text categorization termed as NGramsSA algorithm. The performance of this algorithm is also compared with NGrams algorithm and the results are provided at the end of this paper.

Keywords

N-Grams, Test Profile, Simulated Annealing, Precision, Execution Time

1. Introduction

The biggest problem that is found in the Internet is the huge chunks of the Information available. This information is stored in the form text documents in majority of the cases. Now the problem moves to the next level. How to sort out this text for further processing is the next level of the problem. The answer for this is going to be finding an efficient text categorization algorithm. Currently algorithms like NGrams, K-Nearest Neighbourhood produce better results in categorizing the text. But still there is a need for an improvised algorithm for the Text categorization problem. An efficient algorithm will have a very good impact on this real world problem. With these things in the background a new algorithm termed as NGramsSA is proposed in this chapter. Along with the details of this algorithm, its performance details have also been compared against the NGrams algorithm, whose performance was found to be better than other algorithms.

2. Definition

Classification problem can be stated like this in general. If a set of objects are present, it has to be divided into different groups. This task of grouping objects into different groups based on their characteristics and properties is called as classification. While discussing about classification it is important to distinguish it from clustering. In classification the objects are assigned to predefined categories. These predefined categories are called as classes. But in clustering there are no such predefined classes existing. The clustering module has to identify such classes and group the objects accordingly.

In computer based classification, an object is usually represented by a set of attributes or features. Each feature corresponds to the property of the object. Usually the set of features are grouped to form a feature vector,

in which each vector component corresponds to a feature. For example, the following is a feature vector(X) with n features (f1,f2,…,fn):

X=< f1,f2,…,fn >

Consider a general model of computer based classification task. In this model, a set of n-dimensional feature vectors representing the objects to be classified is given to the classifier as input. Based on the features contained in each input feature vector, the classifier will make the decision of class assignment for the objects. This decision is represented by an m dimensional output vector, where m is equal to the number of pre-defined classes in particular classification task. Here classifier is a function which maps an n-dimensional feature vector into an m dimensional output vector. The output vector from the classifier is sometimes called a classification vector.

3. Motivation for the new algorithm

At present one algorithm that stands out in terms of execution time and precision is NGrams algorithm (Abdellatif Rahmoun and Zakaria Elberrichi, 2007). Hence if a search for more efficient algorithm is done it has to start from the NGrams algorithm. An improvisation to this algorithm or modification made to this algorithm will definitely provide an improved algorithm for the TC. This idea gave the direction for finding the better algorithm for TC. So keeping NGrams algorithm as the base the improvisation has to be planned.

Now the question is how to make an improvisation or modification to this algorithm to get better results. If the algorithms used earlier are considered K Nearest Neighbourhood-Simulated Annealing (KNN-SA) algorithm is one among them. Basically KNN algorithm is found to be a lazy algorithm but when it is combined with the simulated annealing it provides far more improved results (Chuan Yao Yang et al, 2007). On the other hand simulated annealing (SA) is a random search technique which tries to find good solutions to an optimization problem by making random variations of the current solution. Now this idea of SA is applied to the NGrams algorithm to create an improvisation to this algorithm. After forming this idea a new algorithm termed as NGramsSA has been formulated and discussed in this paper.

4. Background

Few ideas that will help in understanding the algorithm NGramsSA is provided here. Like the four algorithms which have been discussed in the previous chapter, NGramSA algorithm also categorizes the text documents according to the predefined categories. These categories are inputted into this system. Then the profiles for these categories are created. These profiles are called as test profiles. It means that all the terms or n grams belonging to the specific category is included in this profile. These profiles play an important part in the algorithm. This works as a testing data in the algorithm. After this the document profile is also created in the same manner mentioned above.

χ_2 statistical supervised method is used for selecting the NGrams belonging to a particular category. To help this, the frequency of the n grams is taken into account. After this the computation of the weights for the n grams is computed and stored in the corresponding profiles.

Finally to classify the documents belonging to a particular category cosine distance between the category profile and the document profile is calculated based on the weights assigned for the n grams. Apart from these aspects in the algorithm annealing schedule has to be incorporated into it.

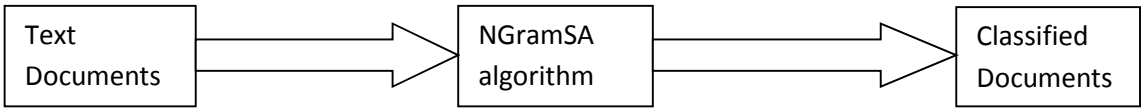

Figure 1 : Role of NGramsSA algorithm

An annealing schedule consists of the following things:

1. Initial Temperature
2. Final Temperature or the stopping criterion
3. A length for the Markov's chains
4. A rule for the decrementing temperature.

In this algorithm the initial temperature is the highest test category profile and then getting the corresponding R n grams. Here R refers to a constant numeric value. Then the cosine distance of this test instance and the R n grams are computed. This is the initial temperature.

The final temperature or stopping criterion is about finding the next R n grams and computing the cosine distances for them. Then if these new cosine distances do not replace earlier R NGrams then it can be concluded that this is the final temperature. This also means that the system has found the stable state.

The next aspect in the annealing schedule is the length of the Markov's chain. The length of the Markov's chain is set as R. Theoretically the importance of the Markov's chain is that if the system is in testing state, it finds n constitute, then the energy to gain is *f(n)* × Markov. It means that if the system is less stable then more tests have to be conducted. This gives more opportunity to jump out of the trap.

All the above mentioned elements have to be incorporated in the NGramsSA algorithm.

5. NGramsSA Algorithm

Several details about these algorithms have been provided in section 6.3. In continuation of these details this algorithm provided here.

Step 1: N grams generation based on documents for classification is carried out.

Step 1.1: The weight for each n gram is calculated based on the TFIDF formula.

Step 1.2: Now the n grams generated are stored in array called as feature array.

Step 2: Selection of the characteristic n grams is carried out.

Step 2.1: Input the categories under which the classification of documents should take place.

Step 2.2: Now the test profile of each category is created. For this all the n grams belonging to this category are selected. The weight of each test instance is calculated.

Step 2.3: For selecting the n grams under a particular category χ_2 statistic supervised method is used. In this method frequency of the n grams is taken into account.

Step 2.4: Compute the weight of each n grams and store them in the profile. Along with the weight W_i, a term D_i is used. The D_i means the number of the instance which contains this feature. (E.g. suppose the considered feature is "Soccer", D_i refers how many times "Soccer" has appeared)

Step 2.5: Compute the feature weight of test instance and store it in test profile.

Step 2.6: Calculate $\sqrt{\sum_{i=1}^{n} x_i^2}$ of each training instance *x*, where i is denoted as feature of the instance.

Step 3: Choose n grams with the highest weight from the test category. Then choose out the first R n grams directed by this feature from the training category profile.

Step 4: Compute the cosine distance between the test instance and R instances and insert the result into candidate set.

Step 5: Markov = R;

Step 6: while (flag)

Step 7:{

Step 8: Fetch the highest feature M by its weight of the test instance.

Step 9: Fetch the next Markov n grams from the training category that are not already present in the result set.

Step 10: Compute the cosine distance of testing instance and n grams. Store the result in temp result set.

Step 11: n=count the cosine distances in temp result set which are larger than those in candidate result set.

Step 12: Substitute these n grams and their cosine distances to the candidate result set; Set flag=TRUE

Step 13: if n=0 then flag=FALSE

Step 14: else Markov= (log(n/R×10+0.1))×R;

Step 15: }//end of while loop

This algorithm provided above requires certain explanations. The step 1.1 refers to the weight calculation of the NGrams. One of the methods used for giving weight to the terms is the Inverse document frequency (IDF). IDF is a factor used for reflecting the discrimination value of each term. A commonly used definition of the IDF is given below:

$IDF_i = \log(N/n)$

Where IDF_i is the inverse document frequency of the ith indexing term, N is the number of documents in the document set, and n is the number of documents in which the ith indexing term appears. By this definition, a term appears in fewer documents will have a higher IDF. The assumption behind this definition is that the terms concentrated in a few documents are more helpful in distinguishing between documents with different topics. For term weighting, the term weight is equal to the product of the term frequency (TF) and the inverse document frequency (IDF):

$$tfidf\,(t_k, c_i) = tf\,(t_k, c_i) \times Log\left(\frac{|C|}{df(t_k)}\right)$$

The next aspect is found in the step 2.3. For the purpose of Dimensionality reduction the multivariate χ_2 method is used. The χ_2 measures the degree of association between the feature and the category. This method is based on the assumption that a feature whose frequency strongly depends on the category in which it occurs will be helpful in discriminating among the categories. For the purpose of dimensionality reduction smaller χ_2 are discarded (Abdellatif Rahmoun et al, 2007). This χ_2 multivariate is supervised method. This allows selection of features by taking into account not only their frequencies in each category but also the interaction of features between them and interactions between the features and categories. The principle consists in extracting R better features characterizing the best category compared to the others. This is done for each category. The contributions of the features in discriminating categories are calculated by the following equation:

$$C_{jk}^{x^2} = N\frac{(f_{jk} - f_j \cdot f_k)^2}{f_j \cdot f_k} \times sign(f_{jk} - f_j \cdot f_k)$$

With $f_{jk}=\frac{N_{jk}}{N}$ representing the relative frequencies of the occurrence. Here N refers total sum of occurrences. The values N_{jk} represent the frequency of the feature X_j in the category e_k. The evaluation of sign in this equation makes it possible to determine the direction of the contribution of the feature in discriminating category. If the value is positive it indicates that it is the presence of the feature that contributes in the discrimination while a negative value reveals that its absence in it that contributes in it.

In step 4 computing the cosine distance between the test instance and the R instances have been mentioned. Computing the cosine distance is the very important aspect of this algorithm. This is because if the cosine distance between a feature NGram and the category is very closer then it can be concluded that the particular NGram belong to that category. The cosine distance between the two instances is defined by the following equation.

$$d(x_i,x_j) = \frac{\sum_{t=1}^{n} x_{it} \times x_{jt}}{\sqrt{\sum_{t=1}^{n} x_{it}^2 \sum_{t=1}^{n} x_{jt}^2}}$$

It should be noted that an arbitrary instance *x* be described by the feature vector

$(a_1(x), a_2(x),...a_n(x))$

where $a_n(x)$ denotes the value of the n^{th} attribute of instance *x*.

Now going into the details of the NGramSA algorithm further following things can be found. Initially steps 1, 1.1 and 1.2 talks about the generation of n grams and assigning weights to the n grams. It is found to produce better results when N=5. N represents the length of the n gram. An n gram is nothing but sequence of characters found in a text document. Then steps 2 to 2.6 tell about inputting the different categories, creation of the test profiles along with the weight calculation, using multivariate χ_2 for the dimensionality reduction in n grams and choosing R n grams from the training profile.

From step 3 onwards the algorithm talks about the computation of cosine distance calculation between the testing instance and R training instances, fixing the Markov's length as R and initiating the looping process. Inside the loop the next R n grams are selected and the cosine distances between them are calculated. Then it is compared with the values of temp result set, which has earlier cosine distances. Now if the current values are smaller than the previous values then it means that the system has reached stable state and the loop is terminated. If otherwise, the current values are substituted in the temp result set and Markov length is set as Markov=(log(n/R×10+0.1))×R. The loop continues till the stable state is reached. Now whether the n gram belongs to the particular category is determined by the cosine distance. After this text categorization comes to an end.

6. Results

NGramsSA algorithm was implemented in a Pentium IV system which had a clock speed of 2.4GHz. The software environment in which this algorithm was tested is as follows:

Windows XP Professional is the operating system used. VB.NET and SQL Server2005 was the front-end and Back- end respectively.

In the result a comparison between the NGrams algorithm and the NGramsSA algorithm is conducted, as this comparison will help to appreciate the improvisation in the latter algorithm over the former. These algorithms are compared on the basis execution time and the precision values. On these two counts NGramSA algorithm scores higher than the NGrams algorithm.

Initially the execution time of the NGramsSA and the NGrams algorithms are considered. The execution time of these algorithms is given in the table 1. Looking at the table it can be seen that NGramsSA algorithm's execution time is lesser than the execution time of NGrams' algorithm in all the cases. Looking at the variation in the execution time between NGrams and NGramsSA algorithms, the maximum variation is found in document 19 and it comes to 1.25 seconds. Next to that, in the document 11 the variation in the execution time comes to 0.99 seconds. The minimum variation is present in document 1. Here the variation comes to 0.06 seconds. As stated earlier, invariably in all the cases NGramsSA algorithm consumes lesser time. These details are further represented in the Chart 1. There also one can note that the minimal time taken by the NGramsSA algorithm when compared with the NGrams algorithm.

Table 1: Execution Time of NGramsSA algorithm in comparison with NGrams algorithm

Documents	Total Words	After stop words removal	E.Time in NGrams	E.Time in NGramsSA
Document1	77	40	0.6288	0.56529
Document 2	89	45	0.7074	0.63595
Document 3	98	41	0.64452	0.57942
Document 4	116	50	0.786	0.70661
Document 5	139	69	1.08468	0.97513
Document 6	233	95	1.4934	1.34257
Document 7	247	148	2.32656	2.09158
Document 8	256	132	2.07504	1.86546
Document 9	345	184	2.89162	2.59957
Document 10	352	154	2.42088	2.17637
Document 11	450	286	9.80122	8.8113
Document 12	483	174	2.73528	2.45902
Document 13	561	245	8.39662	7.54856
Document 14	569	325	5.64152	5.07173
Document 15	658	345	5.9892	5.38429
Document 16	724	241	8.25907	7.4249
Document 17	750	329	5.71144	5.13458
Document 18	758	458	7.95088	7.14784
Document 19	850	650	12.39043	11.139
Document 20	850	485	9.2441	8.31045

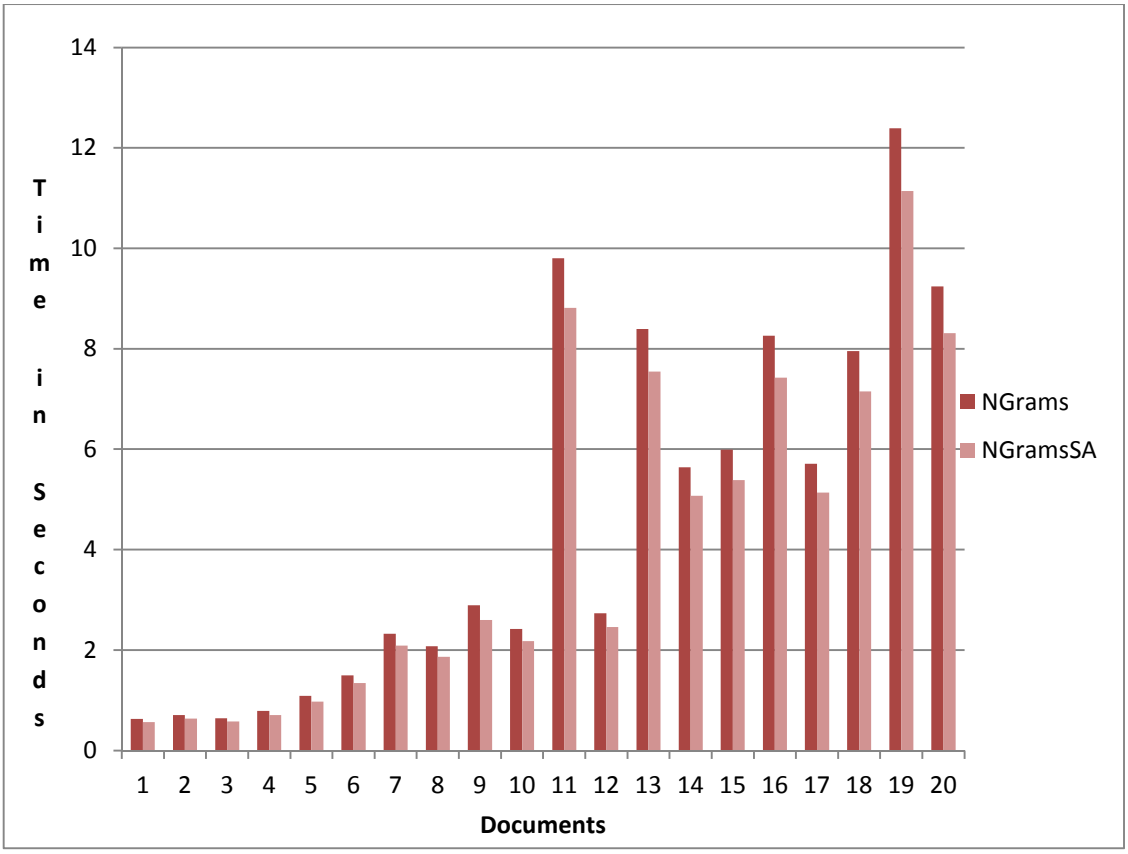

Chart 1: Execution Time of NGramsSA algorithm in comparison with NGrams algorithm

The second aspect to be considered here is the precision value. It means that the two algorithms mentioned above are compared on the basis of this precision value. Precision is calculated based on the following formula:

$$\text{Precision} = \frac{Categories_{matched_{correctly}}}{Total_{categories_{found}}}$$

The precision values of these two algorithms are provided in the table 2. In this table too one can find that the NGramsSA algorithm has higher precision values than the NGrams algorithm in all cases. Precision value is an assessment that how far these algorithms accurately categorize the particular document to a correct predefined category. When an algorithm has higher precision value it means that the algorithm has been more accurate in categorizing a particular text document.

Table 2 : Precision values for NGrams and NGramsSA algorithms

Documents	NGrams Precision	NGramsSA Precision
Document 1	0.76	0.8056
Document 2	0.69	0.7314
Document 3	0.71	0.7526
Document 4	0.72	0.7632
Document 5	0.81	0.8586
Document 6	0.68	0.7208
Document 7	0.85	0.901
Document 8	0.77	0.8162
Document9	0.79	0.8374

Document 10	0.7	0.742
Document 11	0.7	0.742
Document 12	0.65	0.689
Document 13	0.72	0.7632
Document 14	0.69	0.7314
Document 15	0.86	0.9116
Document 16	0.8	0.848
Document 17	0.79	0.8374
Document 18	0.73	0.7738
Document 19	0.84	0.8904
Document 20	0.82	0.8692

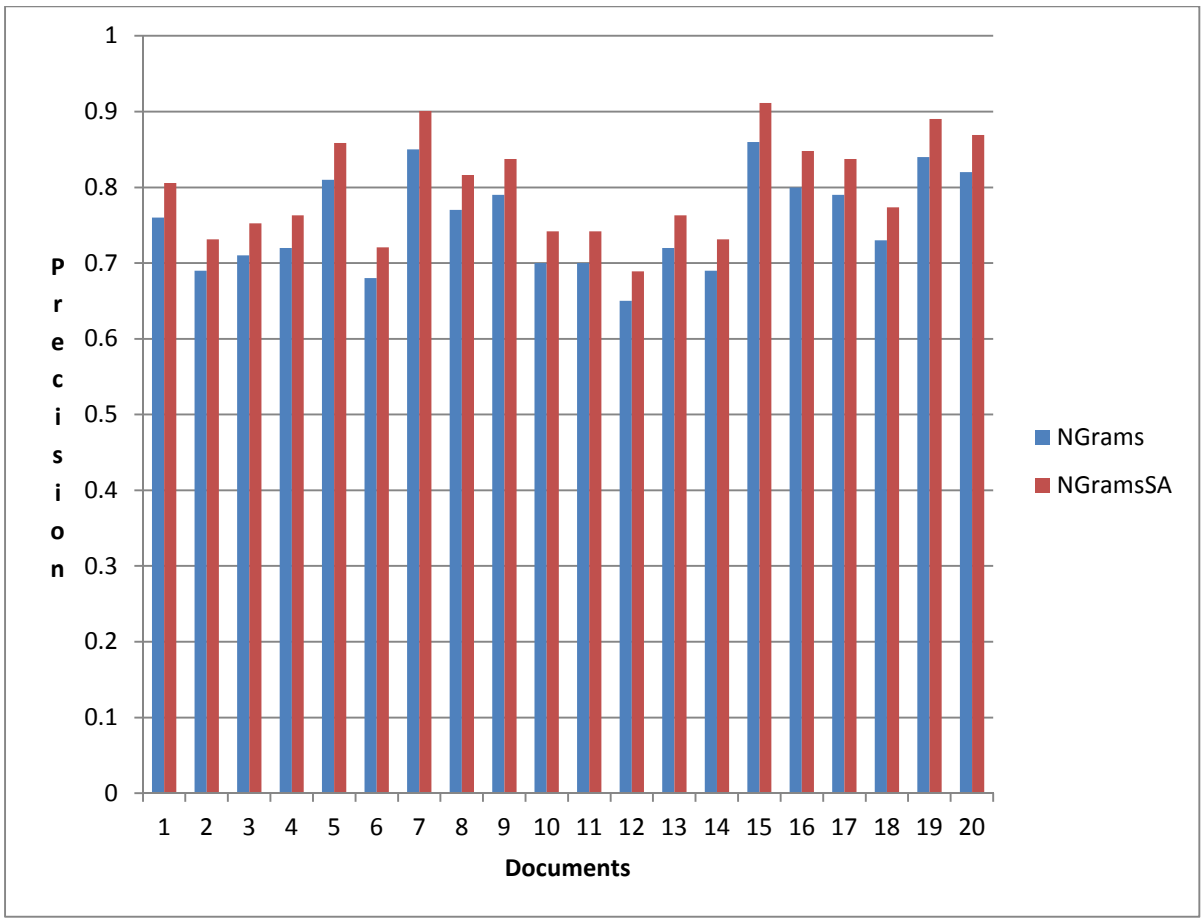

Chart 2: Precision values for NGrams and NGramsSA algorithms

In table 2, the maximum variation in the precision value is present in the document 7 and document 15. The variation is 0.051. This minimum variation is found in the document 12 and it is 0.039. Also it should be noted from the table 2 that the average precision value for the NGrams algorithm is 0.75 and the average precision value for the NGramsSA algorithm is 0.80. These details are pictorially well represented in the chart 2. Looking at this chart one can note that out of the 20 documents 10 times NGramsSA algorithm's precision value is over 0.8. This is a good sign for any algorithm.

Now if you look at the time complexity of the NGrams algorithm it is O(M×T×N) , where M is the number of training documents, T is the average length of the training documents and n is the maximum length of generated sequences. NGramsSA algorithm has a time complexity O(R×M×T×N), where other parameters remain same as the above R refers to the number of n grams considered during the iteration.

7. Conclusion

NGramsSA algorithm explained above shows improvisation over the NGrams algorithm and provides better results. This algorithm inherits the various aspects found in the NGrams algorithm and Simulated Annealing Algorithm. Apart from these things the performance of the algorithm depends on the statistical multivariate χ_2 method used for dimensionality reduction among the generated n grams.

Considering the fact that Text Categorization is a very relevant field in organizing the textual data, finding an improvised algorithm for it is very important. In this direction NGramSA algorithm is another step. Importantly these algorithms are found to be independent of the language that has been used. This is because for different operating systems and languages like Java, VB, ASP, .NET etc. efficiency of the algorithm should not suffer. But already NGrams has been proved to be language independent and small, fast and robust (William B Cavnar et al, 1994). This property applies for the NGramsSA algorithm too.

8. References

1. Abdellatif Rahmoun and Zakaria Elberrichi, (2007). *Experimenting N-Grams in Text Categorization.* The International Arab Journal of Information Technology, Vol.4,No.4.
2. A.Salappa, M.Doumpos and C.Zopoundis, (2007). *Feature Selection algorithms in classification problems: an experimental evaluation*, Taylor & Francis, Optimization Methods and Software Vol.22, No. 1, pp 199-214
3. Ben-Bassat, M., (1982). *Pattern recognition and reduction of dimensionality*. In: Handbook of Statistics, (P. R. Krishnaiah and L. N. Kanal, eds.), North Holland, p.p.773–791.
4. Cavnar, William B. AND Vayda , Alan J.,(1993). "*N-gram based matching for multi-field database access in postal applications*", Proceedings of the 1993 Symposium on Document Analysis and Information Retrieval, University of Nevada, Las Vegas
5. Chuanyao Yang, et.al., (2007). *A Fast KNN Algorithm based on simulated Annealing.* Proceedings of the International Conference on Data Mining, pp.46-51.
6. Damashek, M., (1995), *"Gauging similarity with n-grams: language-independent categorization of text?"*, Science, Vol. 267 No. 10, pp. 843-8.
7. Fabrizio Sebastiani, (2002). *Machine Learning in automatic text categorization*, ACM Computing Surveys, Vol. 34, No. 1, , pp. 1–47.
8. FRASCONI, P., et.al., (2002). *Text categorization for multi-page documents: A hybrid naive Bayes HMM approach.* J. Intell. Inform. Syst. 18, 2/3 (March–May), p.p. 195–217
9. Fuchun Peng, Xiangji Huang, (2006). *Machine Learning for Asian language Text Classification*, Journal Documentation.
10. Furnkranz J., (1998). *"A study using n-gram features for Text Categorization*", Technical Report OEFAI – TR -98-30, Austrian research Institute for artificial Intelligence, Austria.