A Novel Weighted Classification Approach using Linguistic Text Mining

Rajni Jindal
Computer Engineering Department
Delhi Technological University
New Delhi-110042, India

Shweta Taneja
Computer Engineering Department
Delhi Technological University
New Delhi-110042, India

ABSTRACT
Text categorization is the process of automatically assigning labels or categories to new or previously unseen text documents. The text documents may be unstructured or semi structured in nature. In our work, we have used concepts of natural language processing for text categorization. That is, a lexical approach for text categorization. We have developed an algorithm which automatically classifies articles into their categories. The algorithm identifies tokens and assigns them weights in the abstracts of journal articles. We have implemented our approach using K Nearest Neighbor (KNN) classifier as it is the most widely used classifier in research. The proposed algorithm Lexical KNN (LKNN) has been evaluated on two datasets. One is set of journal articles of computer science discipline and the other is a collection of medical documents (Ohsumed collection).The experimental results show that our proposed algorithm Lexical KNN (LKNN) performs better than the other existing classifiers.

Keywords
Text categorization, K Nearest Neighbor (KNN), Lexical Analysis, Tokens.

1. INTRODUCTION
There is a rapid increase in the amount of textual documents available on the Internet. This has led to information clutter and information explosion. So, there is a great need to organize and categorize this information. In this regard, text categorization has turned into a vital tool nowadays in research domain. The process of text categorization automatically assigns labels or categories to new or previously unseen documents ([1], [2], [3]). It is an emerging as well as active research area nowadays. The text documents may be unstructured (i.e. free text) or semi structured (like emails, html documents etc.) in nature.

There are various algorithms available for text classification. Some of them are Naïve Bayes [4], Support Vector Machines (SVM)[5], K Nearest Neighbor (KNN), Decision trees[6], Centroid based Classifiers[7], neural network[8] Rocchio [9] etc. Out of all, KNN [10] is the most widely used classifier in research. This is because it is simple and easy to understand. A lot of studies have been conducted on KNN algorithm.

In our work, we have improved the traditional version of KNN algorithm and included the concept of tokens in it. Our aim is to classify semi structured documents into distinct categories. We have taken a collection of articles as a case study. Our objective is to develop an algorithm which helps to classify journal articles of computer science in sub disciplines like data mining, compilers, computer networks, data encryption etc. For achieving this, we have combined Natural Language Processing (NLP) techniques with text categorization.

The structure of the paper is described as follows. Section 2 gives the background work done in the area of KNN algorithm. We highlight the improvements or modifications done in KNN algorithm both for structured data as well as unstructured data. Section 3 presents our proposed Lexical KNN (LKNN) algorithm along with flow diagram. Section 4 describes the datasets used in implementation. In the next section the results and performance comparison of our proposed algorithm is done with KNN algorithm. And then conclusion is given in the end.

2. BACKGROUND WORK

2.1 KNN Algorithm
The KNN algorithm was first given by Cover and Hart in 1967[11]. This algorithm is easy to implement, simple and effective. It is a lazy learner. The algorithm finds K nearest neighbors in the dataset first and then takes their classes [12]. Then the similarity score is calculated between test document and K nearest neighbor document. That is termed as weight of class of nearest neighbor. If same class is shared by many K nearest neighbors then their weights per neighbor are calculated and summed up. This sum is denoted as that class's score. These scores are sorted further.

The above rule of KNN algorithm can be written in Equation 1 as:

$$\text{Score}(d, c_i) = \sum_{dj \in KNN(d)} \text{Sim}(d, dj)\ \S\ (dj, c_i) \qquad (1)$$

where K nearest neighbors of a document d are shown by KNN (d).

And $\S\ (d_j, c_i) = 1$, if the document belongs to the class and 0 other wise.The test document is allocated to that class which has the maximum resultant weighted sum.

2.2 Improvements made in Traditional KNN Algorithm
The traditional KNN algorithm suffers from the drawback of taking large storage space and high time complexity. Researchers have suggested various improvements ([13], 14]). The KNN algorithm is based on standard Euclidean distance method. It gives equal participation to all the attributes. But in case of large number of irrelevant attributes this feature leads to problems. This problem is called as Curse of dimensionality [10]. To solve this problem, degree of importance or weight is allocated to all attributes. Equation 2 below gives weighted Euclidean distance function:

$$d(x_i, x_j) = \sqrt{\sum w_i (a_r(x_i) - a_r(x_j))x^2} \qquad (2)$$

here attribute A_i has weight w_i and i =1,2,.....,n. Thus in case of nominal attributes, we can define the distance function [15] in Equation 3 as:

$$c(x)= \sum_{i=1}^{n} I_p (A_i : C)\delta(a_i(x), a_i (y)) \quad (3)$$

here A_i is an attribute having mutual information Ip(Ai : C) and C is the class.

$$\delta(a_i(x), a_i (y)) = 0 \text{ if } ai(x) = ai(y)$$

$$\text{and } \delta(ai (x), ai (y)) =1 \text{ otherwise.}$$

Another improvement proposed in literature is known as KNN with distance weights. In this method, k nearest neighbors are assigned votes differently depending on their distance. Equation 4 below gives the weighted class probability estimation method [16]:

$$c(x) = \arg\max \sum_{c\in C, i=1}^{k} w\ \delta\big(c, c (y_i)\big) \quad (4)$$

where k nearest neighbors are y1 , y2 , …, yk and C is a set of class labels.

$$\delta(c, c(yi)) = 1 \text{ if } c =c(yi)$$

$$\text{and } \delta(c, c(yi)) = 0 \text{ otherwise.}$$

The traditional KNN algorithm has also been improved by using the concept of rough sets and fuzzy sets in research. Wang in 2005 [17] gave a new nearest neighbor algorithm that uses the concept of rough sets and fuzzy sets. Also they have used Clustering algorithm and selected cluster centers. Further, [18] suggested the idea of Fuzzy KNN algorithm. The proposed algorithm provides fuzzy memberships to data samples instead of crisp memberships. They introduced the idea of optimal weights that are combined with fuzzy membership values. Thereafter an Adaptive version of Fuzzy KNN algorithm was proposed by Shang et al. [19]. They focused on selection of dimensions and improvement of decision rule. Again in [20], authors gave a new fuzzy based KNN algorithm. This method differs from the already existing Fuzzy KNN algorithm in terms of its initialization procedure. The procedure can manage imprecise inputs. Also the algorithm can be applied upon different types of problems.

Several improvements in the traditional KNN have been suggested for unstructured data as well. The authors in [12] proposed neighbor weighted KNN algorithm to deal with unbalanced text data. The text data is represented using Vector Space Model. The author calculated the weighted sum of distances between the K nearest neighbors. A tree based KNN algorithm or TKNN was proposed in [21] to overcome the drawbacks of KNN algorithm. The proposed algorithm builds the tree structure from the training set, then updates the tree if test data arrives. And then it performs classification by calculating the scores of K nearest neighbors. The most popular approach used is a hybrid approach by combining KNN with other algorithms. The authors in [16] have used a clustering algorithm to combine KNN classifier and the Rocchio classifier. They conducted experiments on Reuters-21578 dataset, various other Chinese and English text corpora.

Text Classification has become an important research area nowadays. In [22], concept of syntactic parse tree is used as a similarity measure to detect weak semantic signals. Many authors worked on hybrid techniques using KNN algorithm. In [23], authors Miao et al. combined the rough set approach with KNN algorithm. The algorithm achieved good performance improvement.

Ours is a novel approach that uses the concept of tokens for text categorization. It identifies tokens from the text documents. These tokens help in the process of classification of documents. Our proposed approach is given in the next section.

3. PROPOSED LEXICAL KNN ALGORITHM (LKNN)

We have taken a collection of articles from reputed journals. There are 80 articles taken from computer science disciplines like compilers, networks, databases, security, encryption etc. This is a small dataset taken for a pilot study. This number may be increased in future. Our approach is based on lexical analysis, where we find tokens from abstracts of journals. The Lexical Analysis module reads the Abstract of the article and identifies the tokens. Tokens here are the keywords present in the abstract of the article. Each journal may be represented by two things: tokens of a journal and their weights. The value of frequency of a token is taken as its weight. We have used distance between the tokens as a metric in our algorithm. And thereafter we use the formula given in equation 5 below to compute predicted class. The proposed LKNN algorithm is given below in figure 1.

Step 1: Build Classification Model or Classifier using a Training set

// Input a set of journal articles. Let J = { j1, j2, j3,…….jn) , where n is the maximum number of journal articles taken.

For i = 1 to n

Repeat

i Scan the abstract section of the journal article (ji) and remove stop words from it. The standard list of stop words that we have used is given in [29].

ii Using Lexical Analysis, scan the abstract and identify the tokens in the abstract of the article ji. The standard list of keywords (tokens) is specified as ACM Computing Classification System in 2012 [30]. Also, find the weight of the token. The weight w_i of a token t_i = $freq_i$, where $freq_i$ is the frequency of occurrence.

iii Create a table of tokens to record the name of token and its weight.

iv. Each journal article ji called an instance, is represented as a vector : <w1 (j), w2 (j), w3 (j), w (j)...> where wi(j) is the weight of the ith term. That weight is set according to its frequency of occurrence.

v To build KNN Classifier, we use distance as a basis to calculate the contribution of each k neighbor in the class allocation process.

vi We define the predicted class of a journal article ji belonging to class c as:

$$\text{Pred}_{\text{class}}\ (c, j_i) = \frac{\sum_{k_i \in K[\text{Class}\ (k_i=c)]} \text{Sim}\ (k_i, j_i)}{\sum_{k_i=K} \text{Sim}\ (k_i, j_i)} \quad (5)$$

Where Sim is a similarity function which returns a value after comparing an article with its neighbor. That is, we sum up the similarities of each neighbor belonging to a particular class c and divide by all similarities of k neighbors irrespective of the class.

Step 2: Text Categorization

// Classify the test journal article using the KNN Classifier

To compare article j with instance i, we define the CosSim function which is defined using our token weight approach as follows:

$$\text{CosSim}(i, j) = \frac{S}{\sqrt{A * B}} \quad (6)$$

where S is the number of terms that i and j have in common, A is the number of terms in i and B the number of terms in j.

Fig 1: Proposed Lexical KNN Algorithm

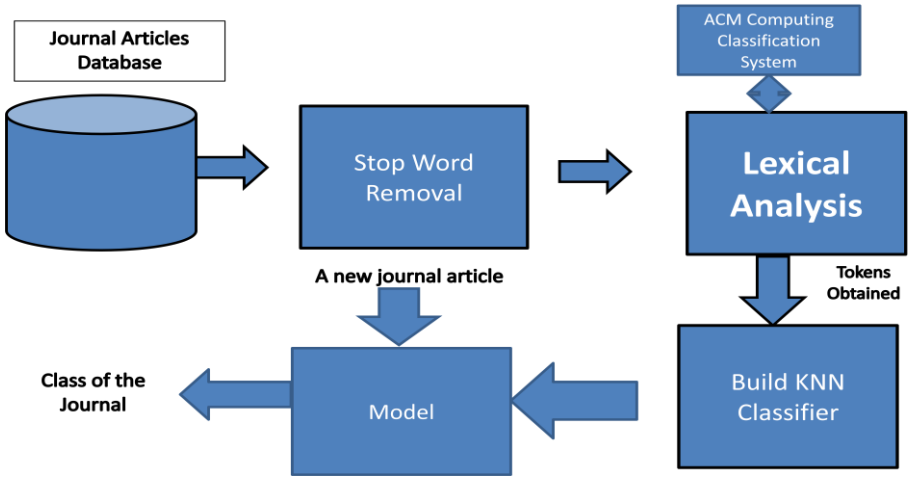

Fig 2: Flow diagram of the proposed linguistic Approach

The working of the proposed algorithm may be explained as follows: Firstly, journal articles (after removing stop words) are taken. They are fed as input to Lexical Analysis. The lexical analysis scans the journal article character by character and groups them in tokens. Keywords are taken as tokens. The standard ACM Computing Classification System (2012) is used for this purpose. The output of this part is a list of tokens. Then KNN Classifier is developed using the tokens along with the weights. The flow diagram of LKNN algorithm is given in Figure 2.

4. DATASETS USED

The proposed algorithm is implemented using JDK 1.6[24] on two datasets. One is a set of 80 journal articles of computer science discipline. These articles belong to reputed journals. The other dataset is a collection of medical documents (Ohsumed Collection) used by Hersh et al. [25] which is a subset of MEDLINE database [26]. This dataset is made up of abstracts from medical documents of the year 1991. The dataset we have taken (Joachims [27]) contains total 20,000 documents in which 10,000 documents are used for training and rest 10,000 is used for testing. We have taken cardiovascular diseases. The table 1 shows the details of computer science journals dataset.

Table 1. Computer Science Journal Articles Dataset

S .No	Sub discipline	No. of articles
1	Data bases	20
2	Networks	20
3	Compilers	20
4	Security	10
5	Encryption	10

5. RESULTS AND PERFORMANCE EVALUATON

The proposed LKNN algorithm is compared with the standard KNN algorithm. The performance measures used are Recall, Precision, F1 – measure and Accuracy [28]. These can be calculated in following equations 7, 8 and 9 respectively:

$$\text{Recall} = A/(A+B) \quad (7)$$

$$\text{Precision} = A/(A+C) \quad (8)$$

$$\text{F1-measure} = (2 * \text{Precision} * \text{Recall})/(\text{Precision} + \text{recall}) \quad (9)$$

The following tables 2, 3, 4 and 5 show the recall, precision, F1- measure and accuracy values for various values of K in Ohsumed Collection. The figures 3, 4, 5 and 6 also show the values of these parameters for Ohsumed Collection respectively. The results obtained are the best ones that we get. Due to limitation of space, we have shown the results for Ohsumed Dataset only.

From the results, it can be analyzed that the values of recall, precision, F1- measure as well as accuracy are better for LKNN algorithm as compared to the traditional KNN algorithm. Also, it is noticed that as the value of k is increased, the proposed algorithm LKNN shows better performance than KNN algorithm.

Table 2. Values of Recall for different values of K in Ohsumed Collection

Values of K	KNN	LKNN
1	0.402	0.405
3	0.456	0.501
5	0.437	0.498
7	0.501	0.520
10	0.521	0.550

Table 3. Values of Precision for different values of K in Ohsumed Collection

Values of K	KNN	LKNN
1	0.566	0.586
3	0.601	0.621
5	0.631	0.721
7	0.660	0.800
10	0.566	0.956

Table 4. Values of F1-measure for different values of K in Ohsumed Collection

Values of K	KNN	LKNN
1	0.470	0.472
3	0.517	0.546
5	0.516	0.589
7	0.569	0.630
10	0.542	0.698

Table 5. Values of Accuracy for different values of K in Ohsumed Collection

Values of K	KNN	LKNN
1	0.625	0.625
3	0.710	0.750
5	0.715	0.800
7	0.635	0.867
10	0.581	0.947

In case of Ohsumed dataset, the values of Recall, Precision, F1 measure increase linearly with the value of K as shown in figures 3, 4, 5 and 6 respectively. As shown in table 2, value of Recall improves from 0.402 to 0.405 in case of the proposed algorithm (LKNN). Similarly, as the value of K increases value of Recall also increases. This same pattern is observed in case of Precision values shown in table 3. That is, for small value of K there is not much difference in both the algorithms. But as K increases the values of Precision also increases. This similar pattern is observed in F1 measure and Accuracy shown in tables 4 and 5 respectively.

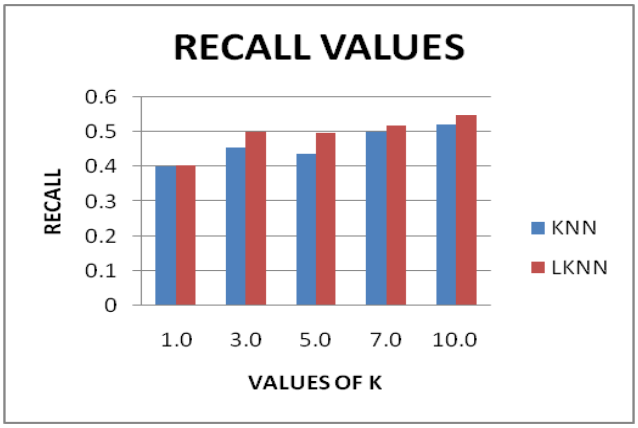

Fig 3: Comparison of Recall values between KNN and LKNN

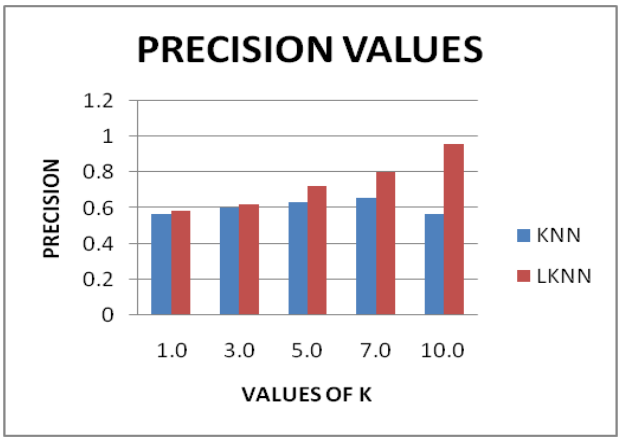

Fig 4: Comparison of precision values between KNN and LKNN

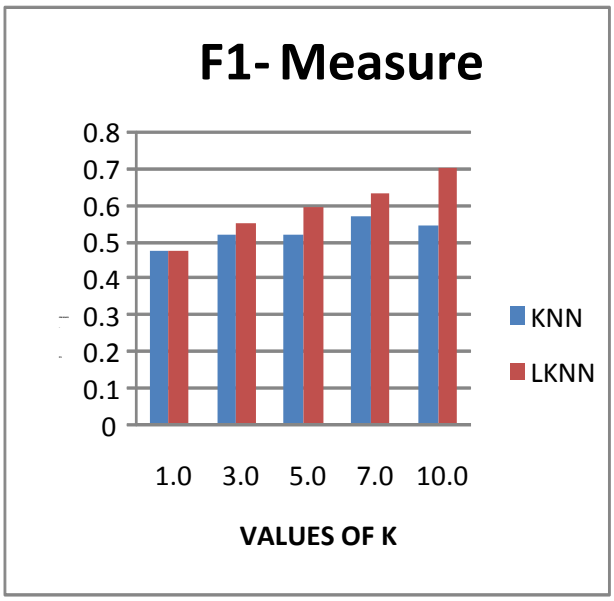

Fig 5: Comparison of F1-measure values between KNN and LKNN

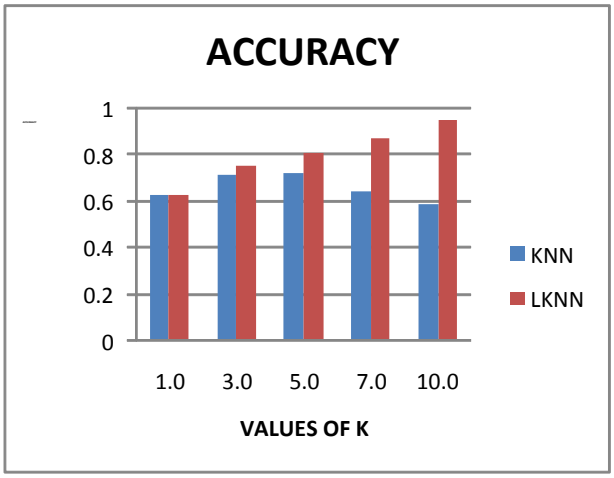

Fig 6: Comparison of Accuracy values between KNN and LKNN

6. CONCLUSION

In this work, the concepts of NLP are applied in field of text categorization. The proposed algorithm Lexical KNN (LKNN) helps in the automatic categorization of journal articles into various categories. Every journal article can be denoted by a group of tokens. KNN algorithm is used in the work. The proposed algorithm is tested on two datasets: One is a collection of journal articles of computer science discipline and other is collection of medical documents. The performance of the proposed algorithm is experimentally proved to be better than the traditional KNN for both the datasets. This is proved by calculating the performance metrics like Recall, Precision, F1-measure and Accuracy values. In future, the proposed algorithm can be tested on journal articles of other disciplines.

7. REFERENCES

[1] Dong, T., Cheng, W. and Shang 2012. . The Research of kNN Text Categorization Algorithm Based on Eager Learning. Proceedings of International Conference on Industrial Control and Electronics Engineering, Xi'an, IEEE Xplore, 1120-1123.

[2] Aggarwal, Charu, C., Zhai, Cheng, X. (Eds.) 2012. Mining Text Data, Springer.

[3] Weiss, Sholom, M., Indurkhya, Nitin, Zhang, Tong 2015. Fundamentals of Predictive Text Mining. 2nd Edition, Springer.

[4] Wei, D., Yang, L.X. 2010. Weighted Naive Bayesian Classifier Model Based on Information Gain. Proceedings of International Conference on Intelligent System Design and Engineering Application ,Changsha, 2, IEEE Xplore digital library.

[5] Liu, Q., He, Q., and Shi, Z. 2008. Extreme Support Vector Machine Classifier. Proceedings of PAKDD, Lecture Notes in Computer Science , 5012, Springer, 222-233.

[6] Lewis, D. D. and Ringuette, M. 1994. Comparison of two learning algorithms for text categorization. In Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval (SDAIR'94).

[7] Han, E. and Karypis, G. 2000. Centroid -based document classification analysis and experimental results. Technical Report. University of Minnesota. http://www.cs.umn.edu/wkarypis.

[8] Ruiz, M. E., and Srinivasan, P. 2002. Hierarchical text categorization using neural networks. Information Retrieval, 5(1), 87–118.

[9] Rocchio, J. 1971. Relevance feedback in information retrieval. In G. Salton, editor, The SMART Retrieval System: Experiments in Automatic Document Processing, 313–323. Prentice-Hall Inc.

[10] Tan, 2006. An effective refinement strategy for KNN text classifier. Journal of Expert Systems with Applications, 30, 290–298.

[11] Cover, T.M. and Hart, P.E. 1967. Nearest Neighbor Pattern Classification. IEEE Transactions Information Theory, 13, 21-27.

[12] Tan, S. 2005. Neighbor-weighted K-nearest neighbor for unbalanced text corpus. Expert Systems with Applications, 28, 667–671.

[13] Sun, S. and Huan, R. 2010. An Adaptive k-Nearest Neighbor Algorithm. Proceedings of Seventh International Conference on Fuzzy Systems and Knowledge Discovery, Yantai, Shandong, IEEE Xplore, 91-94.

[14] Wu, J., Cai, Z. and Gao, Z. 2010. Dynamic K-Nearest-Neighbor with Distance and Attribute Weighted for Classification. Proceedings of International Conference on Electronics and Information Engineering, Kyoto, IEEE Xplore, 356-360.

[15] Bo, S., Junping, D. and Tian, D. 2009. Study on the Improvement of K-Nearest-Neighbor Algorithm. Proceedings of International Conference on Artificial Intelligence and Computational Intelligence , Shanghai , IEEE Xplore, 390-393.

[16] Pang, G. and Jiang, S. 2013. A generalized cluster centroid based classifier for text categorization. Journal of Information Processing & Management, Volume 49, Issue 2, 576-586.

[17] Wang, X. et al. 2005. Fuzzy-Rough Set Based Nearest Neighbor Clustering Classification Algorithm. In Proceedings of FSKD 2005, LNAI 3613, 370 – 373.

[18] Pham, T.D. 2005. An Optimally Weighted Fuzzy k-NN Algorithm. In Proceedings of ICAPR 2005, LNCS 3686, 239–247.

[19] Shang, W. et al. 2006. An Adaptive Fuzzy kNN Text Classifier. In Proceedings of ICCS. Part III, LNCS 3993, 216 – 223.

[20] Chua, T. and Tan, W. 2009. New Fuzzy Rule-Based Initialization Method for K-Nearest Neighbor Classifier. Published in Proceedings of Fuzz-IEEE , Korea.

[21] Juan, L. 2011. TKNN: an improved KNN algorithm based on tree structure. Seventh International Conference on Computational Intelligence and Security.

[22] Galitsky B. 2013. Machine learning of syntactic parse trees for search and classification of text. Engineering Applications of Artificial Intelligence, 26, 1072–1091.

[23] Miao, D, Duan, Q., Zhang, H. and Jiao, N. 2009. Rough set based hybrid algorithm for text classification. Expert Systems with Applications,36, 9168–9174.

[24] Schildt, H. 1996. Java: The Complete Reference, Ninth Edition, Oracle press.

[25] Hersh, W., Buckley, C., Leone, T., Hickman, D. 1994. Ohsumed: An interactive retrieval evaluation and new large text collection for research. 17th ACM International Conference Research and Development in Information Retrieval, 192–201.

[26] Mehnert, R. 1997. Federal agency and federal library reports. .Province, NJ: National Library of Medicine. National Library of Medicine: http://www.nlm.nih.gov/.

[27] Joachims, T. 1998. Text categorization with support vector machines: Learning with many relevant features. Tenth European conference onmmachine learning, 137–142.

[28] Yuan, P., Chen, Y., Jin, H. and Huang, L. 2008. MSVM-kNN: Combining SVM and k-NN for Multi-Class Text Classification. Proceedings of IEEE International

Workshop on Semantic Computing and Systems, Huangshan, .133-140.

[29] www.lextek.com/manuals/onix/stopwords2.html

[30] Mirkin, Boris, Nascimento, Susana, Pereira, Luis Moniz, 2008. Representing a Computer Science Research Organization on the ACM Computing Classification System, in Proceedings of the 16th International Conference on Conceptual Structures (ICCS-2008) , CEUR Workshop Proceedings, 354, RWTH Aachen University,57–65.