Classification of Text Documents Using B-Tree

B.S. Harish[*], D.S. Guru, and S. Manjunath

Department of Information Science and Engineering, SJCE, Mysore
Department of Studies in Computer Science, University of Mysore, Manasagangothri, Mysore, Karnataka, India
bsharish@ymail.com, dsg@compsci.uni-mysore.ac.in, manju_uom@yahoo.co.in

Abstract. In this paper, we propose an unconventional method of representing and classifying text documents, which preserves the sequence of term occurrence in a test document. The term sequence is effectively preserved with the help of a novel datastructure called 'Status Matrix'. In addition, in order to avoid sequential matching during classification, we propose to index the terms in B-tree, an efficient index scheme. Each term in B-tree is associated with a list of class labels of those documents which contain the term. Further the corresponding classification technique has been proposed. To corroborate the efficacy of the proposed representation and status matrix based classification, we have conducted extensive experiments on various datasets.

Keywords: **T**ext documents, Term sequence, B-Tree, Classification.

1 Introduction

Bag of Word (BoW) is one of the basic methods of representing a document. The BoW is used to form a vector representing a document using the frequency count of each term in the document. This method of document representation is called as a Vector Space Model (VSM) [1]. The major limitation of VSM is that the correlation and context of each term is lost which is very important in understanding a document. Jain and Li [2] used binary representation for given document. The major drawback of this model is that it results in a huge sparse matrix, which raises a problem of high dimensionality. Hotho et al., [3] proposed an ontology representation for a document to keep the semantic relationship between the terms in a document. This ontology model preserves the domain knowledge of a term present in a document. However, automatic ontology construction is a difficult task due to the lack of structured knowledge base. Cavanar., (1994) [4] used a sequence of symbols (byte, a character or a word) called N-Grams, that are extracted from a long string in a document. In a N-Gram scheme, it is very difficult to decide the number of grams to be considered for effective document representation. Another approach [5] uses multi-word terms as vector components to represent a document. But this method requires a sophisticated automatic term extraction algorithms to extract the terms automatically from a

[*] Corresponding author.

document. Wei et al., (2008) proposed an approach called Latent Semantic Indexing (LSI) [6] which preserves the representative features for a document. The LSI preserves the most representative features rather than discriminating features. Thus to overcome this problem, Locality Preserving Indexing (LPI) [7] was proposed for document representation. The LPI discovers the local semantic structure of a document. Unfortunately LPI is not efficient in time and memory [8]. Choudhary and Bhattacharyya (2002) [9] used Universal Networking Language (UNL) to represent a document. The UNL represents the document in the form of a graph with words as nodes and relation between them as links. This method requires the construction of a graph for every document and hence it is unwieldy to use for an application where large numbers of documents are present.

After giving an effective representation for a document, the task of text classification is to classify the documents to the predefined categories. In order to do so, many statistical and computational models have been developed based on Naïve Bayes classifier, K-NN classifier, Centroid Classifier, Decision Trees, Rocchio classifier, Neural Networks, Support Vector Machines [10].

Although many text document representation models are available in literature, frequency-based BoW model gives effective results in text classification task. Indeed, till date the best multi-class, multi-labeled categorization results for well known datasets are based on BoW representation [11]. Unfortunately, BoW representation scheme has its own limitations. Some of them are: high dimensionality of the representation, loss of correlation with adjacent words and loss of semantic relationship that exist among the terms in a document [12]. Also the main problem with the frequency based approach is that given a term, with lesser frequency of occurrence may be appropriate in describing a document, whereas, a term with the higher frequency may have a less importance. Unfortunately, frequency-based BoW methods do not take this into account [9]. Hence there is a need for developing a new scheme for document representation preserving the correlation among adjacent words. This motivated us to use a new datastrcuture called “Status Matrix” [13] which effectively represents a text document and thereby giving a better classification results.

The paper is organized as follows. The working principle of the proposed method is presented in section 2. Details of experimental settings and results are presented in section 3. The paper is concluded in section 4.

2 Proposed Method

In this section, we propose a new method of representing documents based on preserving the sequence of term occurrence in a document. Subsequently, we present the corresponding classification model.

2.1 Representation Stage

Let there be k number of classes each containing n number of documents. A simple text processing algorithm is employed to extract the terms (words) present in each document. From the extracted set of words, stop words are removed. For each class say C_j, $j = 1, 2, ..., k$, set of all words present in the documents of that class is formed. From these set of words an inverted list structure is formed for each of the word by associating the labels of the class of the documents that contain that particular word. The list of class labels associated with a word may contain many class labels as it is not uncommon that the documents of different classes contain the same word. The words and their associated lists of class indices are recommended to be stored in the knowledge base to support classification of an unknown test document.

However, this representation requires a linear time searching, which is not acceptable in real pragmatic scenario. Thus in order to speed up classification and to make the representation scheme dynamic supporting addition and deletion of documents, we recommend to index the documents using an existing indexing data structure. To do this task, one may think of many indexing structures like multidimensional binary trees [14], G-Tree [15], KDB Tree [16] and BD Tree [17]. However, each structure has got its own limitations [18] with respect to handling the data and storage methods. Thus, in our work we make use of B-Tree structure as it is simple and less complex. Moreover, B-Tree is used because of its availability, its simplicity and less complexity in addition to its balanced nature.

The proposed B-tree based system can be easily extended towards dynamic databases, as it is very easy to include new documents. In addition, insertion of new documents is as simple as just the insertion of set of words into the existing set and updating the associated term lists. With respect to the proposed representation scheme, insertion of a document into the database is simply a process of inserting the terms present in the document into the B-tree. In order to insert a term T corresponding to the document to be inserted, the B-tree is accessed through to find out the location of T in the B-tree. If T is already present in the database, the insertion problem is reduced to the problem of getting the list of documents updated by appending the index of the document to be inserted. On the other hand, if a term T is not present in the database then no doubt we are at the node U where T is expected to be present. If U contains fewer than $(r-1)$ terms (r is the order of the B-tree), T is simply inserted into U in a sorted order. Otherwise, unlike conventional B-tree insertion procedure where the node is eventually split into two nodes, in our work, we recommend to look at the siblings of the node to find if any a free location, so that by data movements we can get the T term accommodated at the node U itself without splitting it up. Indeed this modification suggested to the conventional B-tree insertion process significantly enhances the efficacy of the insertion procedure particularly on a very large B-tree. The complexity of using the B-tree is of $O(\log_r t)$, where t is the number terms stored in the B-tree and r is the order of the B-tree.

For an illustrative purpose, we consider four different classes of documents. For each class we have created a knowledge base as follows. Given a set of training documents of an individual class, stopwords from each training documents are eliminated and the terms are pooled to form a knowledge base. The knowledge base obtained for four different classes are given below:

K1: categorization,documents,implement,metric,similarity,text

K2: algorithms,categorization,mining,similarity,video

K3: algorithms,efficient,enhancements,filter,image

K4: algorithms,congestion,networks,protocols,routing.

The terms present in the knowledge base along with their class labels are stored in a B-tree for the purpose of fast retrieval.

A B-tree of order $r = 3$ is constructed (Figure 1) to store the distinct terms and each term in the B-tree is attached with its respective list of class indices. The index table containing all terms for each of the documents to be stored is created as shown in Table 1.

2.2 Classification

Sequence of occurrence of words in any text plays a major role in understanding the text document. However, most of the existing methods do not preserve the sequence of occurrence of words as they assume that the word occurrence is independent of text representation.

Simple method to check the sequence of occurrence of words is same as common longest substring matching. Thus, one can think that the problem of classifying a test document is reduced to the problem of finding out a common longest subsequence of terms in the database. In practice, this is not acceptable as the process of substring matching has non-deterministic polynomial time complexity.

Hence, in this section we propose an alternative method of matching and classification of text documents. For the purpose of preserving the sequence of occurrences of words in a test document we recommend to use the concept of status matrix. Status matrix representation was proposed for the purpose of recognition of partially occluded object recognition, where status matrix representation is a binary matrix preserving the order in which the information occurs.

2.3 Computational Complexity of Classification

As there are k classes and a query document contains t_q terms, we require $O(t_q \log_r t)$ computations to create a status matrix of size M .

A status matrix is a binary matrix where the entries are either 0 or 1. The status matrix is of dimension $k \times t_q$ where, k is the number of classes, and t_q is the number of terms in the query text document after preprocessing.

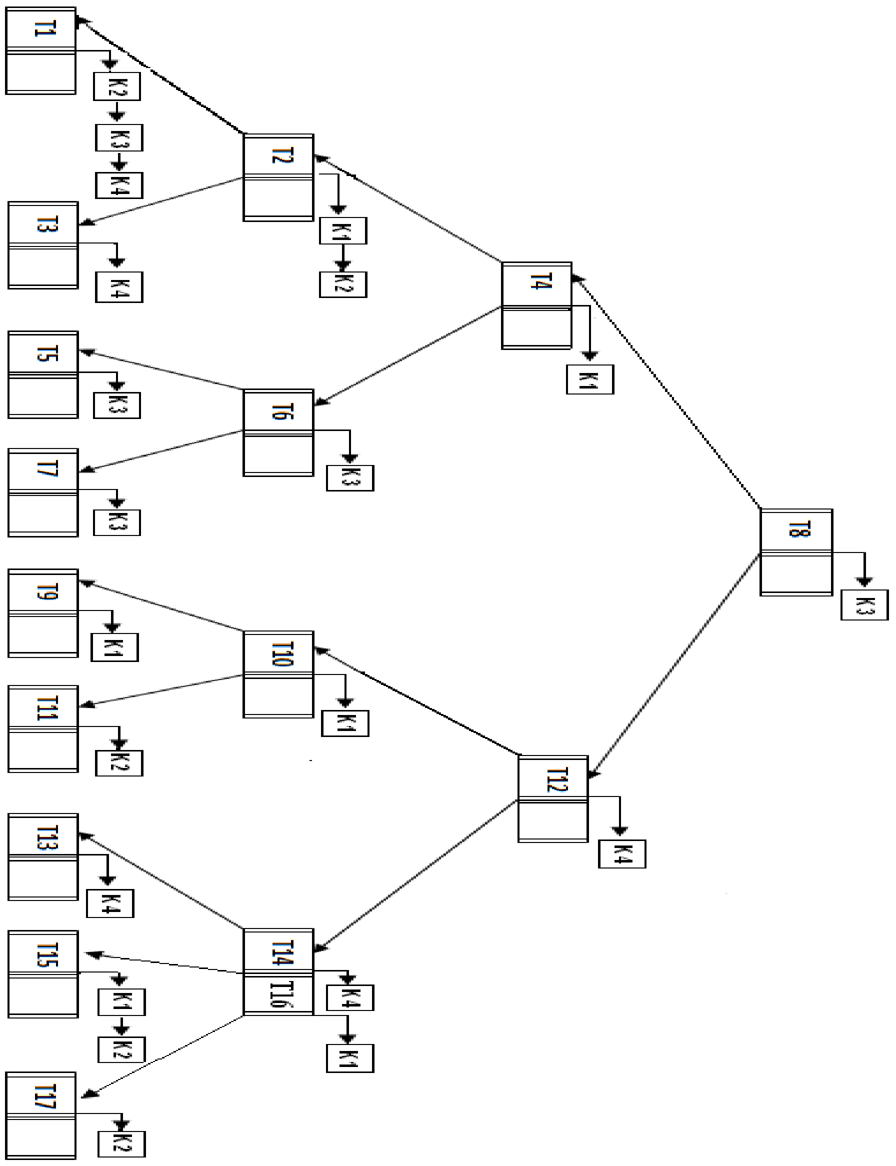

Fig. 1. A B-tree representation to the knowledge base

Table 1. The index table for the illustrated example

Index	Terms	Index	Terms
T1	Algorithms	T 10	Metric
T2	Categorization	T 11	Mining
T3	Congestion	T 12	Networks
T 4	Documents	T 13	Protocols
T 5	Efficient	T 14	Routing
T 6	Enhancements	T 15	Similarity
T 7	Filter	T 16	Text
T 8	Image	T 17	Video
T 9	Implement		

The B-tree is accessed through in search of each term and the lists of document indices corresponding to that term are retrieved from the database. If the i^{th} term T_i of the query document is present in the knowledge base of the class C_j , then the entry corresponding to the row of C_j and the column T_i in the status matrix is set to 1, otherwise it is set to 0. That is, if M is a status matrix, then, M is given by

$$M_{ij} = \begin{cases} 1 & if\ T_i \in C_j \\ 0 & otherwise \end{cases} \tag{1}$$

Assuming each row of the status matrix as a binary string, we then look for a row with a longest substring containing only 1s. The class corresponding to that row is declared to be the class of the test document.

As an illustration, let us consider the following paragraph as a query document d_q .

"Text categorization is not a trivial problem. The complexity of the problem lies in how to define a similarity metric between the documents, and then how to implement a computationally efficient algorithm to solve the problem given this similarity metric".

In order to classify this document we first eliminate stopwords present in it, which results with the following set of terms.

{text, categorization, trivial, problem, complexity, similarity, metric, documents, implement, computationally, efficient, algorithms, similarity, metric}.

This query document totally contains 14 terms. Now it is understood that as there are 4 classes and the query document has 14 terms, we have the status matrix of size 4×14 as shown in Table 2.

Once the status matrix is constructed we search through the status matrix in search of longest matched sequence. Now, the test document is assigned to the class which has longest matched sequence of terms present in the query document. Here in this example the query document is given the class label 1.

Table 2. Status matrix obtained for query document d_q

	T1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}	T_{11}	T_{12}	T_{13}	T_{14}
k_1	1	1	0	0	0	**1**	**1**	**1**	**1**	0	0	0	1	1
k_2	0	1	0	0	0	1	0	0	0	0	0	1	1	0
k_3	0	0	0	0	0	0	0	0	0	0	1	1	0	0
k_4	0	0	0	0	0	0	0	0	0	0	0	1	0	0

3 Experimental Setup

3.1 Dataset

To test the efficacy of the proposed model, we have used the following five datasets. The first dataset is standard 20 Newsgroup Large [19]. It contains 20000 documents categorized into 20 classes. The second dataset consists of vehicle characteristics extracted from wikipedia pages (vehicles- wikipedia) [20]. The dataset contains 4 categories that have low degrees of similarity. The dataset contains four categories of vehicles: Aircraft, Boats, Cars and Trains. All the four categories are easily differentiated and every category has a set of unique key words. The third dataset is a standard 20 mini newsgroup dataset which contains about 2000 documents evenly divided among 20 Usenet discussion groups. This dataset is a subset of 20 newsgroups which contains 20,000 documents. In 20 MiniNewsgroup, each class contains 100 documents in 20 classes which are randomly picked from original dataset. The fourth dataset is constructed by a text corpus of 1000 documents that are downloaded from Google-Newsgroup. Each class contains 100 documents belonging to 10 different classes (Business, Cricket, Music, Electronics, Biofuels, Biometrics, Astronomy, Health, Video Processing and Text Mining). The fifth dataset is a collection of research article abstracts. All these research articles are downloaded from the scientific web portals. We have collected 1000 documents from 10 different classes. Each class contains 100 documents.

3.2 Experimentation

In this section, we present the results of the experiments conducted to demonstrate the effectiveness of the proposed status matrix based method on all the five datasets viz., 20 newsgroup large, vehicles wikipedia, 20 mini newsgroup, google newsgroup and research article abstracts. During experimentation, we conducted two different sets of experiments. In the first set of experiments, we used, 50% of the documents of each class of a dataset to create training set and the remaining 50% of the documents for testing purpose. On the other hand, in the second set of experiments, the numbers of training and testing documents are in the ratio 60:40. Both experiments are repeated 5 times by choosing the training samples randomly. The minimum, maximum and the average value of the classification accuracy of all the 5 trials are presented in Table 3.

Table 4 shows the comparative analysis of the results on five datasets mentioned above. It is clear from the Table 4, that the proposed model achieved good classification accuracy on three benchmark datasets and also on our own dataset (Google Newsgroup and Research article abstracts), when compared with other well known classifiers viz. Naïve Bayes, KNN and SVM classifier. This is because of the proposed model has a capability of classifying the documents at two stages (one is at voting stage and the other is at term sequence stage). It is also worth mentioning that the incorporation of status matrix improves the performance of a voting classifier.

Table 3. Classification accuracy of the proposed method on different data sets

Dataset	**Training Vs Testing**	**Minimum Accuracy (5 Trials)**	**Maximum Accuracy (5 Trials)**	**Average Accuracy (5 Trials)**
20 Newsgroup Large	50 vs 50	79.65	84.20	82.64
	60 vs 40	84.35	87.85	86.35
Vehicles Wikipedia	50 vs 50	70.65	72.85	71.60
	60 vs 40	74.95	76.00	75.45
20 Mini Newsgroup	50 vs 50	64.65	68.95	66.91
	60 vs 40	71.00	76.85	73.95
Google Newsgroup	50 vs 50	86.70	89.70	88.74
	60 vs 40	89.85	96.00	93.33
Research Article Abstracts	50 vs 50	86.25	90.25	88.52
	60 vs 40	89.00	91.25	90.13

Table 4. Comparative analysis of the proposed method with other classifiers

Dataset Used	Voting Classifier	Voting + Status Matrix Classifier **(Proposed Method)**	Naïve Bayes Classifier	KNN Classifier	SVM Classifier
20 Newsgroup Large	82.55	87.85	86.50	70.00	85.65
Google Newsgroup	93.50	96.00	80.00	46.25	48.25
Vehicle-Wiki	67.50	76.00	74.00	64.50	63.00
20 Mini Newsgroup	66.25	71.12	66.22	38.73	51.02
Research Article Abstracts	86.75	91.25	-	-	-

4 Conclusion

In this paper, we have proposed the classification of text documents using B-Tree. Further, we have presented a new datastructure called status matrix through which, we make an attempt to preserve the sequence of term occurrence in the query document. In addition, in order to avoid sequential matching during classification, we propose to index the terms in B-tree, an efficient index scheme. In order to investigate the effectiveness and robustness of the proposed method, experimentation is conducted on five different datasets. The experimental results tabulated in Table 4, indicate that the proposed method offers better performance results than other three well-known classifiers. In the proposed method we have pooled the terms of training documents of each class to create a knowledge base. For a given query document we create the status matrix to preserve the sequence of the term appearance in the query document. As we have pooled the terms in the knowledge base we are not preserving the term sequence during training stage. Along with this the presence of continuous 1's in status matrix do not ensure that the database contains any document having same sequence of terms present in the test document. Hence in our future work we try to study the sequence of the term appearance using the concept of status matrix even on training documents and there by preserving the topological sequence of term occurrence in a document useful for semantic retrieval.

References

1. Salton, G., Wang, A., Yang, C.S.: A Vector Space Model for Automatic Indexing. Communications of the ACM 18, 613–620 (1975)
2. Li, Y.H., Jain, A.K.: Classification of Text Documents. The Computer Journal 41, 537–546 (1998)
3. Hotho, A., Maedche, A., Staab, S.: Ontology-based text clustering. In: International Joint Conference on Artificial Intelligence, USA, pp. 30–37 (2001)
4. Cavnar, W.B.: Using an N-Gram based document representation with a vector processing retrieval model. In: The Third Text Retrieval Conference (TREC-3), pp. 269–278 (1994)
5. Milios, E., Zhang, Y., He, B., Dong, L.: Automatic term extraction and document similarity in special text corpora. In: Sixth Conference of the Pacific Association for Computational Linguistics (PACLing 2003), Canada, pp. 275–284 (2003)
6. Wei, C.P., Yang, C.C., Lin, C.M.: A Latent Semantic Indexing-based approach to multilingual document clustering. Journal of Decision Support System 45, 606–620 (2008)
7. He, X., Cai, D., Liu, H., Ma, W.Y.: Locality Preserving Indexing for document representation. In: SIGIR, pp. 96–103 (2004)
8. Cai, D., He, X., Zhang, W.V., Han, J.: Regularized Locality Preserving Indexing via Spectral Regression. In: ACM International Conference on Information and Knowledge Management (CIKM 2007), Portugal, pp. 741–750 (2007)
9. Choudhary, B., Bhattacharyya, P.: Text clustering using Universal Networking Language representation. In: Eleventh International World Wide Web Conference (2002)
10. Seabastiani, F.: Machine Learning in Automated Text Categorization. ACM Computing Surveys 34, 1–47 (2002)

11. Bekkerman, R., Allan, J.: Using Bigrams in Text Categorization. CIIR Technical Report, IR – 408 (2004)
12. Bernotas, M., Karklius, K., Laurutis, R., Slotkiene, A.: The peculiarities of the text document representation, using ontology and tagging-based clustering technique. Journal of Information Technology and Control 36, 217–220 (2007)
13. Dinesh, R.: POOR: Partially Occluded Object Recognizers – Some Novel Techniques. Ph.D. Thesis, University of Mysore (2006)
14. Bentley, J.L.: Multidimensional binary search trees used for associative searching. Communications of ACM 18(9), 509–517 (1975)
15. Kumar, A.: G – tree: A new datastructure for organizing multidimensional data. IEEE Transactions on Knowledge and Data Engineering 6(2), 341–347 (1994)
16. Robinson, J.T.: The KDB tree: A search structure for large multidimensional dynamic indexes. In: Proceedings of ACM SIGMOD Conference, Ann Arbor, MI, pp. 10–18
17. Dandamudi, S.P., Sorenson, P.G.: An empirical performance comparison of some variations of the k-d tree and bd tree. Computer and Information Sciences 14(3), 134–158 (1985)
18. Punitha, P.: IARS: Image Archival and Retrieval Systems. Ph.D. Thesis, University of Mysore (2005)
19. `http://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.html`
20. Isa, D., Lee, L.H., Kallimani, V.P., Rajkumar, R.: Text document preprocessing with the Bayes formula for classification using the support vector machine. IEEE Transactions on Knowledge and Data Engineering 20, 23–31 (2008)