NATIONAL JOURNAL OF ARTS, COMMERCE & SCIENTIFIC

RESEARCH REVIEW

NEURAL NETWORK BASED APPROACH FOR CLASSIFICATION OF TEXT DOCUMENTS

ADARSH MJ*[1]

Praveen kumar katigar*[2]

[1]Research Scholar

[2]Supervisor , Assistant Professor, Dept. of Comp. Science and Engg ,

Proudhadevaraya Institute of Technology

ABSTRACT

Internet is a pool of information, which contains billions of text documents which are stored in compressed format. In literature we can find many text classification algorithms which work on uncompressed text documents. In this paper, we propose a novel representation scheme for a given text document using compression technique. Further, neural network classifier is also designed for classification of text documents. For the purpose of compression, LZW compression technique is used and the compressed dictionary representation obtained by LZW technique is used as representative for the text document. Extensive experimentation is carried out on seven datasets, out of which three are our own datasets and remaining four are publically available datasets resulting with approximately 89% of F-measure.

Key words: *Text classification, Text compression, LZW compression technique.*

Introduction

Internet is the rapidly growing information gallery that contains rich textual information. This rapid growth makes it difficult for the users to locate relevant information quickly on the web. Document retrieval, categorization, routing and filtering systems are often based on text classification. Text classification problem can be stated as follows: given a set of labeled examples belonging to two or more classes, we classify a new test document to a class with the highest similarity. Text classification presents many challenges and difficulties. Firstly, it is difficult to capture high-level semantics and abstract concepts of natural languages just from a few key words and the same word can represent different meanings. Secondly, it is difficult to handle high dimensionality and variable lengths of text documents.

Text Documents are the most common type of information store house especially with the increased use of the internet. Internet web pages, e-mails, e-news feeds newsgroup messages have millions or billions of text documents. The web pages that are available in the internet are stored in the compressed format. Data mining activities such as document classification and clustering are carried out these data by decompressing the data and taking it back to the standard format. These processes of decompressing and performing mining activities consume more computational time. However to the best of our knowledge, nowhere in the literature we can find any works on classification of text documents in text compressed format. This motivated us to take up this work for design of text classification using text compression representation as a new representation method.

The rest of the paper is organized as follows. In section 2 a brief literature survey on the text classification is presented. In section 3 we present the proposed model based on LZW compression technique. In section 4 we discuss about experimentation details and comparative analysis. In section 5 we present conclusion along with future work.

2. Related Work

In automatic text classification, it has been proved that the term is the best unit for text representation and classification [1]. Though a text document expresses vast range of information, unfortunately, it lacks the imposed structure of traditional database. Therefore, unstructured data, particularly free running text data has to be transformed into a structured data. To do this, many pre-processing techniques are proposed in literature [2, 3]. After converting an unstructured data into a structured data, we need to have an effective document representation model to build an efficient classification system. 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) [4]. The major limitation of VSM is that the correlation and context of each term is lost which is very important in understanding a document. Li and Jain [5] 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. Another approach [6] 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 document. Wei et al., (2008) proposed an approach called Latent Semantic Indexing (LSI) [7] 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) [8] was proposed for document representation. The LPI discovers the local semantic structure of a document. Unfortunately LPI is not efficient in time and memory [9]. Choudhary and Bhattacharyya (2002) [10] 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 [11], K-NN classifier [12], Centroid Classifier [13], Decision Trees [14], Rocchio classifier [15], Support Vector Machines [16]. 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-labelled categorization results for well known datasets are based on BoW representation [17]. 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 [18]. 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 [10].

All the mentioned algorithms works on uncompressed documents. Whereas the challenging and required is to classify documents at compression level.

In literature we can find many compression techniques which are used for the effective representation of data in compressed format. In this paper we consider only the lossless compression schemes. Run Length Encoding (RLE) [19] is a simple and popular data compression algorithm. It is based on the idea to replace a long sequence of the same symbol by a shorter sequence. Huffman compression [20] it is a statistical lossless compression method that converts characters into variable length bit strings. Huffman compression technique works on frequency of individual symbol. The Huffman algorithm is a so-called "greedy" approach to solving this problem in the sense that at each step, the algorithm chooses the best available option. Lempel–Ziv–Welch (LZW) is a universal lossless data compression algorithm created by Abraham Lempel, Jacob Ziv, and Terry Welch. The LZW compression algorithm organized around a translation table or string table, that maps input characters into the fixed length codes [21]. Among different compression techniques, we have used LZW compression technique. LZW compression is used as the foremost technique, mainly because of its versatility and simplicity. Typically, the LZW compression can compress executable code, text, and similar data files to almost one-half of their original size. It usually uses single codes to replace strings of characters, thereby compressing the data. LZW also gives a good performance when extremely redundant data files are presented to it like computer source code, tabulated numbers and acquired signals. The common compression ratio for these cases is almost in the range of 5:1.Though RLE and Huffman compression techniques are also very simple; they are not suitable for text documents and also these two methods does not provide good compression ratio like LZW method.

3. Proposed Method

In this paper we are proposing a novel method used for classification of compressed text documents. Normally text documents are available in several formats such as html, xhtml, pdf, plain text etc. The first step is to pre-process the text document, hence to bring them to a common format before processing the text. In the literature we can find many techniques for pre-processing for text documents. They are stop word elimination, stemming, pruning etc as pre-processing steps.[27]. In this work we have used only stop word elimination technique. Once the pre-processing is done on training data, the text documents are compressed using LZW compression scheme and a compressed training document library is created. The working principle of LZW compression technique is given as follows.

LZW is a universal lossless compression algorithm which is organized around string table. String table contains strings that have been encountered previously in the text being compressed. It consists of a running sample of strings in the text, so the available strings reflect the statistics of the text. It uses greedy parsing algorithm, where the input string is examined character-serially on one pass, and the longest recognized input string is parsed off each time. A recognized string is one that exists in the string table. Each such added string is assigned a uniquely identified by code value.

The proposed model is of two stages, in which stage one is of creation of knowledgebase in which all pre-processed text data are compressed and preserved, stage two is classification stage in which given unknown sample is classified into its corresponding class label using compression technique.

Algorithm: LZW text compression.
Input: Pool of text data
Output: Pool of compressed text data, String table.
Method:

1. Initialize table to contain single character strings.
2. Prefix string ω ← Read first input character.
3. K← Read next input character
   If no such K (input exhausted) : code (ω) – output; EXIT
4. If ωK exists in string table : ωK- ω; repeat 3;
5. else ωK not in string table : code (ω) – output;
6. ωK – string table;
7. K – ω; repeat Step.

Algorithm end.

At each execution of the basic step an acceptable input string ω has been parsed off. The next character K is read and the extended string ωK is tested to see if it exists in the string table. For each training document we obtain a string table which is referred as dictionary representation and stored in the library. Further, given a test document we obtain dictionary representation and neural network approach classification technique. We classify the test document and class label is assigned. The block diagram of the proposed model is as shown in fig 1.

3.1 THE NEURAL NETWORKS CLASSIFIER

The basic design of the classifier discussed in this paper is a feed-forward neural network. In order to achieve good classification accuracy, we used the design of a feed-forward network with a "bottleneck", we choose a three level network with m inputs, m outputs and k neurons on the hidden level, where k < m. Then the network is trained, under standard back-propagation to learn the identity function on the sample examples. The idea is that while the bottleneck prevents learning the full identity function on m-space; the identity on the small set of examples is in fact learnable. Then the set of vectors for which the network acts as the identity function is a sort of sub-space which is similar to the trained set. This avoids the "saturation" problem of learning from only positive examples. Thus the filter is defined by applying the network to a given vector; if the result is the identity, then the vector is "interesting". Training proceeded according to standard back-propagation with learning parameter .75 and momentum coefficient .08 until the mean-square error fell below a pre-determined level. For acceptance threshold determination, a sophisticated method was used, based on a combination of variance and calculating the optimal F1 measure. During training, we checked, at different levels of error, the F1 values of the test set. We stop the training at the point which F1 started a steep decline. Then we did a secondary analysis to determine an optimal real multiple of the standard deviation of the average error to serve as the threshold.

3.2 TEXT REPRESENTATION

To implement the bottleneck filter, we ran experiments with several different representations. In addition, as a comparison, we ran a different algorithm that simply takes the average vector of the sample examples as a prototype vector and defines the tolerance as the largest angle between this prototype and all sample examples. The filter then accepts all documents whose vector representation is within this tolerance.

One can use, instead of the word frequency, the ***tf-idf*** (term-frequency-inverse-document-frequency) representation which is given by the following formula (where ***f(word)*** means the frequency of the word in the document and ***N(word)*** means the number of documents the word appears in):

$$tf - idf = f(word)[\log \frac{n}{N(word)} + 1]$$

To explain our heuristic mix of neural network encoding and heuristic choice of representation, we will need a few definitions:

Let *C*, the *"corpus"* be the set of documents to be classified. Let *T* be a subset of *C* the class of "interesting" documents. Let *E* be a subset of *T*, the positive examples. The problem is to define a function (or "filter"), using only information from *E* that distinguishes *T* from the complement of *T*.

We proceed as follows: Let *D* be the *compressed dictionary* of all words in $\cup$E; with each word is associated its frequency in the list. Heuristically, we eliminate words whose document frequency is less than 3; and use standard algorithms to (i) eliminate common words and (ii) stemming of words.

From this dictionary, we then chose the *m* words that appear in the most documents of *E*. We call these "key-words"; however they are chosen automatically. This choice of *m* is influenced by the comparisons with different choices of *m*. Then we define V_E as the *m*-dimensional vector consisting of the frequencies of appearance of each of the keywords throughout the compressed dictionary. Then for each document e$\in$E we associate a vector of dimension *m*, which we will continue to designate as *e*. Here e_i is the frequency of the i^{th} chosen keyword in the document

Algorithm : Compression based sentimental analysis
Input : Compressed User Reviews
Method :

Step1 : Input collection of compressed text documents.
Step 2: Apply text representation on compressed text documents.
Step 3: Develop a database of compressed text documents (dictionary).
Step 4: Input a compressed text documents for evaluation.
Step 5: Apply text representation on compressed text documents.
Step 6: Apply neural networks classifier and define layers with its parameters.
Step 8: Give a class label to the testing document.

Algorithm ends.

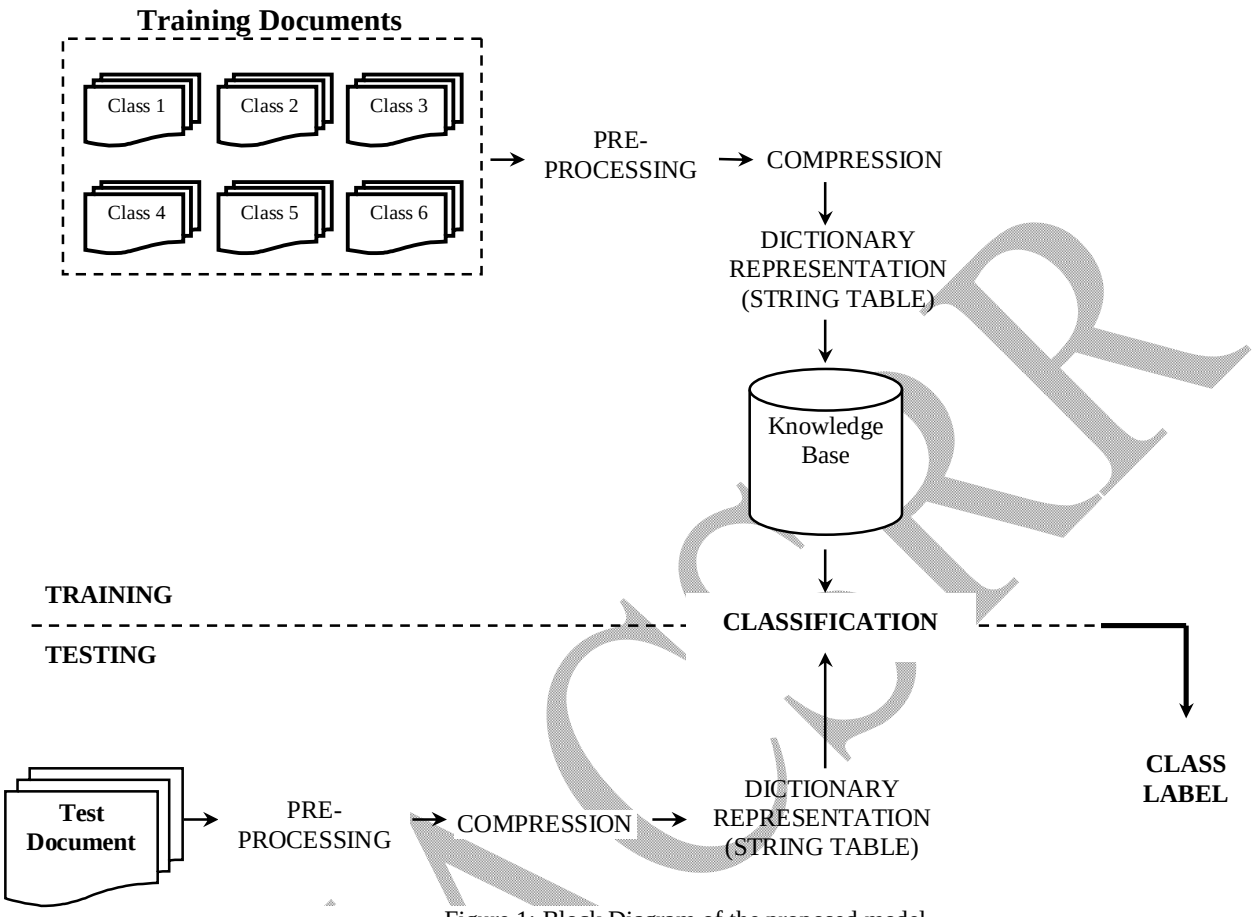

Figure 1: Block Diagram of the proposed model.

4. Experimentation

In this section, we present the details of the experiments conducted to represent the effectiveness of the proposed method on seven datasets. We have created three datasets of our own and four publically available datasets to evaluate the performance of the proposed model. First dataset consists of three classes and each class consists of five documents. Second dataset consists of five classes and each class consists of ten documents. Third dataset consist of 1000 documents from 10 different classes. Fourth dataset is Google news group dataset which contains one thousand documents from ten different classes and fifth dataset is vehicles Wikipedia [22] used to evaluate a prototype system used for the evaluation of classification performance. Seventh dataset is 20mini newsgroup dataset. Sixth and seventh datasets are the 20 newsgroups mini and 20 newsgroup large dataset which are publically available dataset consisting collection of 2000 and 20000 newsgroup documents, partitioned evenly across 20 different classes. The first three datasets consists of documents which do not have overlap compared to other publically available datasets. This is considered to study the performance of the proposed model in case of less overlap and large overlap.

We have conducted two sets of experiments; where each set contain three different trails. In first set of experiments, we have used 40% of the database for training and remaining 60 % is used for testing. In second set of experiments, we have used 60 % training and 40 % for testing. Each set of experiments

contain three different trials. In each trail we have selected training and testing document randomly. For the purpose of evaluation of results, we have calculated precision, recall and f-measure for each trail. The details of the experiments are shown in the following table 1.

The quantitative evaluation of the proposed method is carried out with existing different methods of text classifiers. The proposed method with different type classification techniques are analyzed in qualitative comparative analysis is also presented. Table 3 reveals that, all the existing works in the literature are done on the uncompressed text documents. But the proposed model classifies the documents in compressed format also.

5. Conclusion

Novel LZW compression based technique for classification of text documents is presented. The proposed method uses LZW compressed dictionary representation scheme for representation of text documents. To check the efficiency and the robustness of the proposed models, an extensive experiment is carried out on all the seven dataset. The performance evaluation of the proposed method is carried out by performance measures such as precision, recall and f-measure. Even though, the results are not better than other uncompressed based techniques, they are comparatively equal to them, i.e., approximately 89% of classification accuracy. In this paper we have put forward a new representation model for text classification using compression technique, which is first of its kind. Further, we explore novel proximity measures for comparing text in compressed format which may improve the classification accuracy.

References

[1] Rigutini L., 2004. Automatic text processing: Machine learning techniques. Ph.D. Thesis,University of Siena.

[2] Porter M. F., 1980. An algorithm for suffix stripping. Program, vol. 14, no. 3, pp. 130 –137.

[3] Hotho A., A. Nurnberger and G. Paab, 2005. A brief survey of text mining. Journal for Computational Linguistics and Language Technology, vol. 20, pp. 19 – 62.

[4] Salton G., A. Wang and C. S. Yang, 1975. A vector space model for automatic indexing.Communications of the ACM, vol. 18, no. 11, pp. 613 – 620.

[5] Li Y. H and A. K. Jain, 1998. Classification of text documents. The Computer Journal,vol. 41, no. 8, pp. 537 – 546.

[6] Shafiei M, W. Singer, R. Zhang, E. Milios, T. Bin, J. Tougas and R. Spiteri., 2007. Document Representation and Dimension Reduction for Text Clustering. Proceedings of the IEEE 23rd International Conference on Data Engineering, USA, pp. 770 – 779.

[7] Wei, C.P., Yang, C.C., Lin, C.M. (2008) Journal of Decision Support System. 606—620

[8] He X., D. Cai., H. Liu and W. Y. Ma, 2004 (a). Locality preserving indexing for document representation. Proceedings of the International Conference on Research and Development in Information Retrieval, UK, pp. 96 – 103.

[9] Cai D., X. He and J. Han, 2005. Document clustering using locality preserving indexing.IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 12, pp. 1624 –1637.

[10] Choudhary B and P. Bhattacharyya, 2002. Text clustering using universal networkinglanguage representation. Proceedings of the 11th International Conference on World Wide Web, USA (http://www-clips.imag.fr/geta/User/wang-ju.tsai/articles/BhChPBh-UNL01.PDF).

[11] Mccallum A. K and Nigam K., 1998. Employing EM in pool-based active learning fortext classification. Proceedings of the 15th International Conference on MachineLearning, USA, pp. 350 – 358.
[12] Tan S., 2007. An effective refinement strategy for k-NN text classifier. Journal of Expert Systems with Applications, vol. 30, no. 2, pp. 290 – 298.
[13] Tan S., 2008. An improved centroid classifier for text categorization. Journal of Expert System with Applications, vol. 35, no. 2, pp. 279 – 285.
[14] Wang J., Y. Yao and Z. J. Liu, 2007. A new text classification method based on HMMSVM. Proceedings of the 7th International Symposium on Communications and Information Technologies, Australia, pp. 1516 – 1519.
[15] Lewis D. D., 1992. An evaluation of phrasal and clustered representations on a text categorization task. Proceedings of the 15th Annual International Conference on Research and Development in Information Retrieval, Denmark, pp. 37 – 50.
[16] Mitra V., C. J. Wang and S. Banerjee, 2007. Text classification: A least square support vector machine approach. Journal of Applied Soft Computing, vol. 7, no. 3, pp. 908– 914.
[17] Bekkerman R and J. Allan, 2003. Using bigrams in text categorization. CIIR Technical Report, University of Massachusetts.
[18] Bernotas M., K. Karklius., R. Laurutis and A. Slotkiene, 2007. The peculiarities of the text document representation, using ontology and tagging-based clustering technique. Journal of Information Technology and Control, vol. 36, no. 2, pp. 217 – 220.
[19] P.A. Franaszek (1972), U.S. Patent 3,689,899.
[20] D. A. Huffman., 1952. A method for construction of minimum-redundancy codes. Proceeding of the Institute of Electrical and Radio Engineers, 40(9) pp. 1090 – 1101.
[21] Ziv. J and Lempel A., 1978. Compression of Individual Sequences via Variable-Rate Coding, IEEE Transactions on Information Theory 24 (5), pp. 530–536.
[22] Isa D., L. H. Lee., V. P. Kallimani and R. Rajkumar., 2008. Text document preprocessing with the bayes formula for classification using the support vector machine. IEEE Transactions on Knowledge and Data Engineering, vol. 20, no. 9, pp. 23 – 31.
[23] B.S. Harish, D.S. Guru, and S. Manjunath." Representation and Classification of Text Documents: A Brief Review". IJCA Special Issue on "Recent Trends in Image Processing and Pattern Recognition" RTIPPR, 2010.
[24] Xue X. B and Z. H. Zhou., 2009. Distributional features for text categorization. IEEE Transactions on Knowledge and Data Engineering, vol. 21, no. 3, pp. 428 – 442.
[25] D.S. Guru, B.S. Harish, and S. Manjunath, "Symbolic Representation of Text Documents" Proceedings of the International ACM Conference, Compute 2010, Bangalore, India.
[26] Dinesh, R., Harish, B. S., Guru, D.S., and Manjunath, S. 2009. Concept of Status Matrix in Text Classification.In the Proceedings of Indian International Conference on Artificial Intelligence, Tumkur, India, pp. 2071 – 2079.
[27] Harish B S., 2011. Classification of Large Textual Data. Ph.D.Thesis, University of Mysore.

Table 1 : Classification result table on different dataset using proposed model

Dataset	Trails	40 : 60				60:40:00			
		Precision	Recall	F Measure	Avg F-Measure	Precision	Recall	F Measure	Avg F-Measure
DATASET 1	1	0.8500	0.8500	0.85		0.8800	0.8900	0.88	
	2	0.8600	0.8600	0.86		0.8800	0.8900	0.88	
	3	0.8300	0.8300	0.83	0.85	0.8700	0.8900	0.88	0.88
DATASET 2	1	0.8600	0.8650	0.86		0.8900	0.8900	0.89	
	2	0.8500	0.8700	0.86		0.8800	0.8900	0.88	
	3	0.8800	0.8900	0.88	0.87	0.8700	0.8900	0.88	0.88
DATASET 3	1	0.8610	0.8510	0.86		0.8900	0.8800	0.88	
	2	0.8950	0.8711	0.88		0.8810	0.8800	0.88	
	3	0.8500	0.8600	0.85	0.86	0.8900	0.9000	0.89	0.89
DATASET 4	1	0.8610	0.8700	0.87		0.8600	0.8650	0.86	
	2	0.8600	0.8700	0.86		0.8500	0.8700	0.86	
	3	0.8550	0.8600	0.86	0.86	0.8800	0.8900	0.88	0.87
DATASET 5	1	0.9000	0.9100	0.90		0.9125	0.9100	0.91	
	2	0.8925	0.9052	0.90		0.9000	0.9052	0.90	
	3	0.9122	0.8800	0.90	0.90	0.9200	0.8900	0.90	0.91
DATASET 6	1	0.8900	0.8900	0.89		0.8900	0.8800	0.88	
	2	0.8800	0.8900	0.88		0.8810	0.8800	0.88	
	3	0.8700	0.8900	0.88	0.88	0.8900	0.9000	0.89	0.89
DATASET 7	1	0.8800	0.8900	0.88		0.9000	0.9100	0.90	
	2	0.8800	0.8900	0.88		0.8925	0.9052	0.90	
	3	0.8700	0.8900	0.88	0.88	0.9122	0.8800	0.90	0.90