Tampere University of Technology at TREC 2001

Ari Visa, Jarmo Toivonen, Tomi Vesanen, Jarno Mäkinen
Tampere University of Technology
P.O. Box 553
FIN-33101 Tampere, Finland
{ari.visa, jarmo.toivonen, tomi.vesanen, jarno.makinen}@cs.tut.fi

Abstract

In this paper we present the prototype based text matching methodology used in the Routing Sub-Task of TREC 2001 Filtering Track. The methodology examines texts on word and sentence levels. On the word level the methodology is based on word coding and transforming the codes into histograms by the means of Weibull distribution. On the sentence level the word coding is done in a similar manner as on the word level. But instead of making histograms we use a more simple method. After the word coding, we transform the sentence vectors to sentence feature vectors using Slant transform. The paper includes also description of the TREC runs and some discussion about the results.

1 Introduction

A common approach to topic detection and tracking is the usage of keywords, especially, in context of Dewey Decimal Classification [3, 2] that is used in United States to classify books. The approach is based on assumption that keywords given by authors or indexers characterize the text well. This may be true, but then one neglects the accuracy. There are also many automatic indexing approaches. A more accurate method is to use all the words of a document and the frequency distribution of words, but the comparison of frequency distributions is a complicated task. Some theories say that the rare words in the word frequency histograms distinguish documents [6]. Traditionally, information retrieval has roughly been based on a fixed list of index terms [6, 5], or vector space models [10, 9]. The latter ones miss the information of co-occurrences of words. There are techniques that are capable of considering the co-occurrences of words, as latent semantic analysis [7] but they are computationally heavy.

In this paper, we present our methodology and concentrate on tests of content based topic classification, which is highly attractive in text mining. The evolution of the methodology has been earlier

This research is supported by TEKES, the National Technology Agency of Finland (grant number 40943/99). The support is gratefully acknowledged.

discussed in several publications [11, 12, 13]. In the second chapter the applied methodology is described. In the third chapter the experiments with the Reuters database are described. The execution times are presented in chapter four. Finally, the methodology and the results are discussed.

2 Methodology

The methodology we applied to the TREC 2001 Routing runs is a multilevel one. It examines the contents of text documents on word and sentence levels.

The process starts with preprocessing of the training set text. This includes omitting extra spaces and carriage returns, and separating single words with single spaces. With the Reuters database, the preprocessing also includes the removal of the XML tags. The filtered text is next translated into a suitable form for encoding purposes. The encoding of words is a wide subject and there are several approaches for doing it. The word can be recognized and replaced with a code. This approach is sensitive to new words. The succeeding words can be replaced with a code. This method is language sensitive. Or, each word can be analyzed character by character and based on the characters a key entry to a code table is calculated. This approach is sensitive to capital letters and conjugation if the code table is not arranged in a special way.

We selected the last alternative, because it is accurate and suitable for statistical analysis. A word w is transformed into a number in the following manner:

$$y = \sum_{i=0}^{L-1} k^i * c_{L-i} \tag{1}$$

where L is the length of the character string (the word), c_i is the ASCII value of a character within a word w, and k is a constant.

Example: if the word is "**c a t**", then

$$y = k^2 * ascii(c) + k * ascii(a) + ascii(t) \tag{2}$$

The encoding algorithm produces a unique code number for each different word. After each word has been converted to a code number, we consider the distribution of these numbers and try to estimate their statistical distribution. Many distributions, e.g. Gamma distribution, are suitable for this purpose. However, it would be advantageous, if the selected distribution had only few parameters and it matched the observed distribution as well as possible. Based on tests with different types of text databases, we selected the Weibull distribution to estimate the distribution of the code numbers.

In the training phase the range from the logarithm of the minimum value to the logarithm of the maximum value of code numbers is examined. This range is divided to N_w equal bins. Next, the frequency count of words belonging to each bin is calculated. The bins' counts are normalized with the number of all words. Then the best Weibull distribution corresponding to the data is determined. Weibull distribution is compared with empirical distribution by examining both distributions' cumulative distributions. Weibull's Cumulative Distribution Function is calculated by:

$$CDF = 1 - e^{(((-2.6 * log(y/y_{max}))^b) * a)} \tag{3}$$

There are two parameters that can be varied in Weibull's CDF formula: a and b. A set of Weibull distributions are calculated with all the possible combinations of a's and b's using a selected precision.

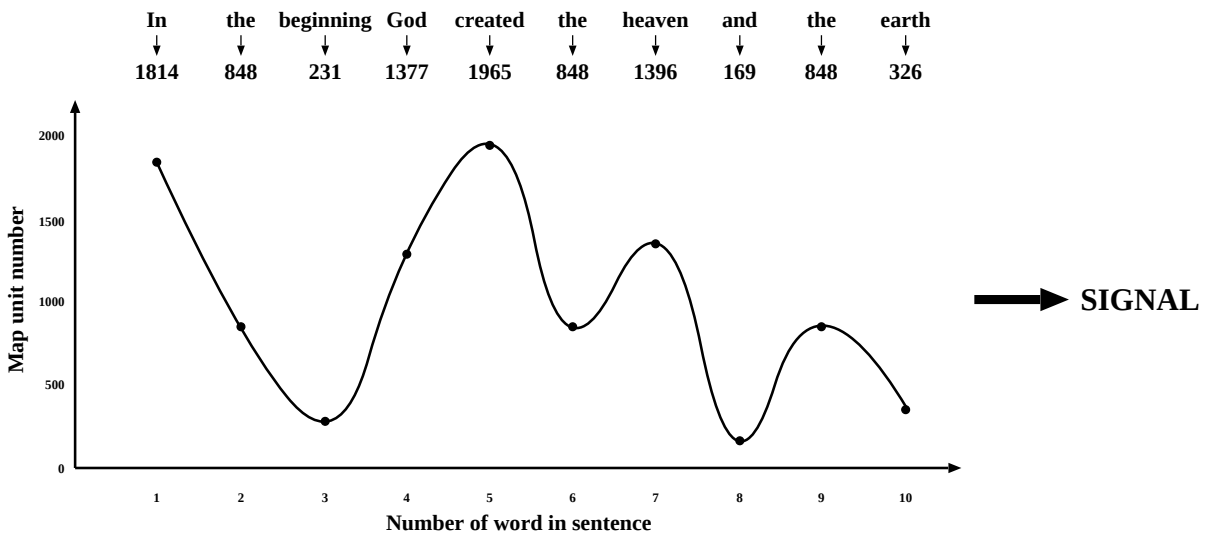

Figure 1. The transform from a sentence to a signal.

The possible values for the coefficients are restricted between realistic minimum and maximum values. The empirical cumulative distribution and Weibull's cumulative distribution are compared in the least square sum sense.

The best Weibull distribution is divided into N_w equal probable bins. Every word belongs now to a bin that can be found using the word code number and the best fitting Weibull distribution. Using this type of quantization the word can now be presented as the bin number, i.e. the number of the bin that it belongs to. Due to the selected coding method the resolution will be the best where the words are most typical to text (usually words with 2-5 characters). Rare words (usually long words) are not so accurately separated from each other.

Similarly at the sentence level every sentence has to be converted to numbers. Every word in a sentence is now replaced with a bin number. The bin number is generated with the method described earlier. Example of encoding a sentence :

I	**have**	**a**	**cat**	**.**
bn_0	bn_1	bn_2	bn_3	bn_4

where bn_i = bin number of the word i. Note, that in the encoding also the punctuation marks are encoded. When the sentence is encoded, it can be considered as a sampled signal. To illustrate this way of thinking, an example sentence and it's encoded form are presented in figure 1.

We use Slant transform matrix for transforming the sentence signals. Slant transform coding is commonly used in image processing. In detail Slant transform is explained e.g. in publications [1, 4, 8]. The size of the Slant matrix is, based on experiments, selected to be 32*32. If the sentences length is over 32 words, the rest of the sentence after 32 words is not considered. If the sentence is shorter than 32 words, then the missing words get zero as value.

First row of Slant matrix is multiplied with sentence vector and the result is stored to S_0. Second row

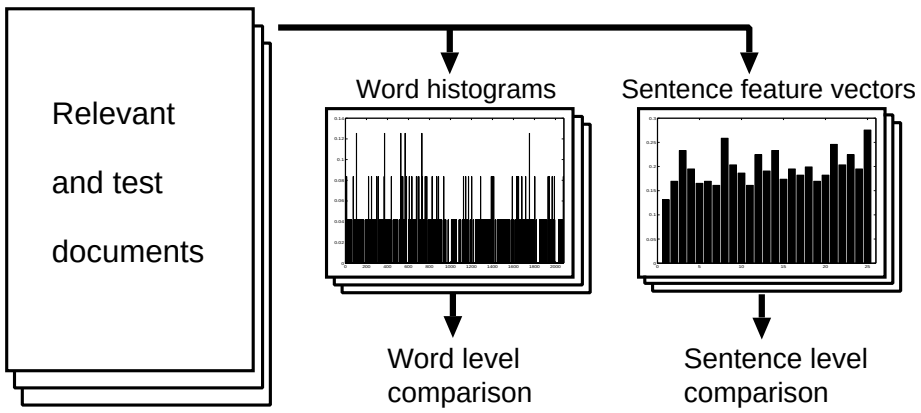

Figure 2. Process of comparing and analyzing documents based on extracted histograms and feature vectors.

multiplication with sentence vector is stored to S_1. I.e.

$$[Slant\ matrix] * [Sentence\ vector]^T \tag{4}$$

creates the result vector of the sentence $[S_0, S_1, S_2...S_{31}]$. Now there are 32 real numbers in a vector that describe the sentence. The numbers can be negative or positive.

This method have to go through all the sentence vectors of the text and the result is summed up with previous result. After every sentence of the text is transformed with Slant transform and then normalized, then we have 32-dimensional feature vector that describes the text.

When examining the test set documents on the word level, we create histograms of the relevant and test set documents' word code numbers. The filtered text from a single document is encoded word by word. Each word number is quantized using word quantization created with all the words of the training set. The quantization value is determined, an accumulator corresponding to the value is increased, and thus a word histogram A_w is created. The histogram A_w consisting of N_w bins is finally normalized by the length of the histogram vector. On the sentence level the relevant and test documents are encoded to straight to feature vectors as described earlier.

With the histograms and feature vectors derived from all the relevant and test documents in the database it is possible to compare and analyze the test documents' text on the word and sentence levels against the relevant texts. The histogram and vector creation and comparison processes are illustrated in Figure 2. Note, that it is not necessary to have any prior knowledge of the actual text documents to use this methodology. The training set words define the distribution formula that is used with the quantization of words. This information is transferred to the sentence level, where the relations and order of the words are taken into account. No linguistic methods are used in the process.

3 Runs with Reuters database

For TREC 2001 we did two runs, one on the word level (VisaWordT10) and one on the sentence level (VisaSentT10). This is our first time in the TREC conference, so we were left with very little time for doing the actual runs. This led to simplifying the training and testing processes.

In the runs for TREC, 2080 was selected for bin number N_w. The amount of bins was selected on the grounds of earlier experiments with English text databases. On the word level run, every test word histogram is compared with randomly chosen 969 relevant document histograms. The number of relevant documents is reduced from 15261 to 969 because it speeds up the test comparison time. Otherwise, the computing time would have risen unreasonable high. To all 84 topics 13 relevant documents have been chosen by random. If a topic has less than 13 relevant documents the number of chosen relevant documents is the amount that exist in that topic. Euclidean distance metric is used in comparing word histograms. The topic of the test document becomes the topic of the relevant document which have the smallest Euclidean distance with the test document. Only the best match relevant document is considered.

On the sentence level run, feature vector of every sentence is compared with all 15261 feature vectors of relevant documents. Since the sentence feature vector is only 32-dimensional, the comparisons are very fast. Comparison was done using Euclidean distance between the sentence vectors. Distances to every topic are calculated for each test document. On the sentence level, all the distances between the test document and the relevant documents are taken into consideration.

On both runs, top thousand test documents are chosen from every topic as the final result. The actual value of the distances are not take into consideration, there are no thresholds for belonging to a topic.

4 Execution times

The applied methodology is very fast even with a database as large as the Reuters database. In table 1 we present the execution times we calculated for the two runs. Making histograms (word level) execution time include finding the best Weibull distribution and creating the word histograms for test documents. Making feature vectors (sentence level) execution time consists of the encoding of the sentences and creation of the 32-dimensional feature vectors. The comparing execution times are the times that it took to compare the test histograms and vectors with the relevant documents' histograms and vectors. The

Table 1. Execution times rounded up to the nearest hour.

	Making histograms/ feature vectors	**Comparing**	**Altogether**
Word level	3 h	30 h	33 h
Sentence level	4 h	26 h	30 h

computer used in the runs was a PC with a Intel® 550 MHz Pentium® III processor and 128 Mb of memory. The operating system was Linux.

5 Conclusions

There were some general difficulties when using the methodology on the Reuters database. The selection of documents for the given training set turned out to be disadvantageous. Firstly, it seemed that the set was too unevenly distributed in topics for our methodology. When some topics have under ten relevant documents and some hundreds, statistical methods are in trouble. There is not enough information in just few short relevant documents for this type of methods to be successful. Uneven division in topics also lead to give more weight to topics that have more relevant documents. Secondly, because the training set was from few days period of a single month, the vocabulary in the relevant documents seemed not to vary enough. The type of methodology we used requires a good set of representative word and sentence samples from the whole database. The training set vocabulary was restricted in the sense of yearly cycle, to one month in autumn of 1996. This type of difficulties are, on the other hand, very common in real life tasks.

More precise problematic issues using the methodology include the selection of quantization method and distance metrics. Using the word quantization method the way presented here has also some disadvantages. Some accuracy is lost when the word codes are quantized with the Weibull distribution. Words that are quite different may get the same value in the quantization, since strict division of the distribution is used to quantize the codes. Perhaps this could be prevented by using a quantization method that would use the code number values to create a more natural classification. Such a method would be very simple and would perhaps create a more precise and truthful quantization of words. Selection of distance metrics is perhaps even more problematic area. Euclidean distance, which was used here, seems not to be taking into account the shape of the histograms and feature vectors. Euclidean distance compares just the values that are in the same place in two vectors. A distance metric that would allow more variation in the position of values in the histograms would perhaps be more suitable. This kind of metrics are for example Levenshtein metrics or some metrics utilizing the angle between vectors.

Our methodology seemed to work well only in few topics. However, the methodology was competitive with other competitors on some topics. The runs were designed so that only a very basic form of the methodology was used. The methods used are very fast and it seems that the speed in these runs was gained at the cost of accuracy. It should be noted, that the methodology uses no external aids in the process. Dictionaries, stemming algorithms, transformation to base form, or linguistic information were not used. The methodology does not depend on the language. The comparisons can be performed as long as the training data and the test data are written in the same language.

Future improvements in the methodology include finding a way to balance the effects of different amounts of relevant documents in topics. One of our future aims is to more efficiently take into account the advantage from the given relevancy information. This would hopefully help to create a specific view of the topic, i.e. the knowledge of the words and sentences that separate the topics. Also the size of the word histograms and the sentence feature vectors should be more properly optimized for the amount and type of the text in the database.

References

[1] N. Ahmed and K. Rao. *Orthogonal Transforms for Digital Signal Processing.* Springer Verlag, 1975.

[2] M. Dewey. *A Classification and subject index for cataloguing and arranging the books and pamphlets of a library.* Case, Lockwood & Brainard Co., Amherst, MA, USA, 1876.

[3] M. Dewey. Catalogs and Cataloguing: A Decimal Classification and Subject Index. In *U.S. Bureau of Education Special Report on Public Libraries Part I*, pages 623–648. U.S.G.P.O., Washington DC, USA, 1876.

[4] A. Jain. *Fundamentals of Digital Image Processing.* Prentice Hall, 1989.

[5] T. Lahtinen. *Automatic indexing: an approach using an index term corpus and combining linguistic and statistical methods.* PhD thesis, Department of General Linguistics, University of Helsinki, Finland, 2000.

[6] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing.* The MIT Press, Cambridge, Massachusetts, 1999.

[7] D. W. Oard and G. Marchionini. A conceptual framework for text filtering. Technical Report CS-TR3643, University of Maryland, May 1996.

[8] W. K. Pratt, L. R. Welch, and W. H. Chen. Slant transform image coding. *IEEE Trans. on Communications*, 22:1075–1093, August 1974.

[9] G. Salton. *Automatic Text Processing.* Addison-Wesley, 1989.

[10] G. Salton, A. Wong, and C. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

[11] A. Visa, J. Toivonen, S. Autio, J. Mäkinen, H. Vanharanta, and B. Back. Data Mining of Text as a Tool in Authorship Attribution. In B. V. Dasarathy, editor, *Proceedings of AeroSense 2001, SPIE 15th Annual International Symposium on Aerospace/Defense Sensing, Simulation and Controls. Data Mining and Knowledge Discovery: Theory, Tools, and Technology III*, volume 4384, Orlando, Florida, USA, April 16–20 2001.

[12] A. Visa, J. Toivonen, B. Back, and H. Vanharanta. Improvements on a Knowledge Discovery Methodology for Text Documents. In *Proceedings of SSGRR 2000 – International Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet*, L'Aquila, Rome, Italy, July 31–August 6 2000. (CD–ROM).

[13] A. Visa, J. Toivonen, H. Vanharanta, and B. Back. Prototype Matching – Finding Meaning in the Books of the Bible. In J. Ralph H. Sprague, editor, *Proceedings of the Thirty-Fourth Annual Hawaii International Conference on System Sciences (HICSS-34)*, Maui, Hawaii, USA, January 3–6 2001. (CD–ROM).