HYPERGEO: A DATA ORGANIZATION AND RETRIEVAL SYSTEM FOR TOURIST INFORMATION

A. GEORGAKIS, C. KOTROPOULOS, N. BASSIOU, I. PITAS
Department of Informatics
University of Thessaloniki
Thessaloniki 54006, Greece.
Tel: +3031-998225
E-mail: costas@zeus.csd.auth.gr

Abstract

Exploring data stored in a document organization system often requires the formation of a rather complicated query. Sometimes, restating that query in the case of poor or inaccurate results is proven to be even harder. The present search engines are based on keyword-based search techniques and carry some native drawbacks. This paper presents a method for data organization of domain specific text documents in an effort to overcome these difficulties and increase the overall performance of an information retrieval (IR) system. The method presented is partially based on the self-organizing map (SOM) architecture. SOMs are artificial neural networks that have the ability of creating a spatially organised representation for the feature vectors of the input space. The data used throughout the experiments have been extracted from the Internet and are related to the tourism domain.

Keywords: Document organization and retrieval, data mining.

1. Introduction

One of the most complicated tasks nowadays is the retrieval of the information stored inside a document organization system. A query aiming at exploring these data must fulfil certain criteria (keywords and Boolean logic). Boolean logic requires the keywords in the search query to be combined using Boolean operators such as *AND*, *OR* and *NOT*. However, Boolean logic has some native weaknesses, namely: it doesn't appear to be a good way to weight the significance of the terms in the query and furthermore, some users are not well trained in Boolean logic [1].

The search engines available over the Internet can be regarded as IR systems. Almost every search engine uses the Boolean logic in order to respond to a query. To perform this task, the search engine has to build some kind of database. For example, let us consider a popular search engine like Altavista or Yahoo [2,3]. Web crawlers are used to collect web pages for indexing. After the completion of document indexing, a user can generate a query. The search engine then presents the user with a list of ranked documents, ideally with those being most relevant to the query topic returned first, from which the user can obtain the information or answers to questions, he/she is seeking. In reality, the output of a search engine is more or less corrupted with noise, that is, it contains also information non-relevant to the original query.

Another problem is that despite the fact that some search engines can point pretty well to a URL or a document that probably contains an answer to the query, they cannot suggest web pages or documents that have similar context to a specific, user-defined, document. Furthermore, they can be easily tricked, in order, to index a page not relevant to a specific subject in a higher position.

Most of these problems are based on the fact that keyword-based document organization and retrieval strategies are used. A great portion of these keyword-based search information retrieval systems is based on one of the following indexing techniques: inverted files, suffix arrays, or signature files [1]. Nowadays, emphasis is given on the inverted file organization. The inverted file structure is based on two files: the vocabulary and the occurrences file, which are build both during the training phase of data indexing. The vocabulary file stores all the different words found in the text files, whereas the occurrences file stores the text positions where these words appear. Words assumed to contain no particular information are normally removed from the documents in a process called stopping.

In the recall phase, these two files along with the processed query (stemmed and stopped) are input to the retrieval engine. The words in the query are searched in the vocabulary file and their occurrences in the text files are retrieved. From these text files the search engine derives its answer. Although the user's query can be a word list, phrase, sentence or extended text, it is regarded

Acknowledgement
The work presented in this paper was supported by the IST European Project - HYPERGEO (IST-1999-11641).

as an unstructured list of terms, which is governed by Boolean operators.

In the Hypergeo project the main target group consists of the so-called mobile users. Anyone using modern equipment, such as Personal Digital Assistants (PDAs), mobile phones with Wireless Application Protocol (WAP) enabled, or even having access to the Internet can be regarded as a mobile user. Through these high-tech gadgets any user can obtain access to every single IR system available today. One question arising here is, if it is worth trying to issue queries, with these low-bandwidth, expensive to use devices. As stated above, the problems originate from the fact that the IR systems are keyword-based and do not take into consideration the document context. To increase the performance of an existing IR system, a different approach must be used.

The presented state-of-the-art technique models the semantics and pragmatics in natural language texts in an effort to overcome these problems. In the second and third section of the paper we will present a statistical approach towards that direction.

2. Self-Organizing Map

SOMs [4,5,6,7] are single-layered, feed-forward artificial neural networks [8] that form a non-linear projection from a high-dimensional data manifold (the so-called feature vectors) onto a low-dimensional space (usually a two-dimensional or three-dimensional grid). The algorithm takes into account the relations of the feature vectors and computes an optimal representation that approximates these data in the sense of some error criterion, usually the mean square error (MSE). SOMs are used to cluster the vectors assigned to the words or the documents of the document collection [9,10,11] according to a specific feature.

The neural network consists of a set of processing units (neurons) arranged on a hexagonal or orthogonal lattice. Each neuron is equipped with a weight vector, which is initialized either randomly or by using randomly selected initial samples. SOMs work as follows; from the set of feature vectors, one randomly selected vector at a time is presented to each neuron of the network. Every neuron calculates an output value. This value is a metric between the input vector and the weight vector of the particular neuron. A possible choice for that metric is the Euclidean distance. To be specific, let $\boldsymbol{x}_i$ denote an *m*-by-*1* vector

$$\boldsymbol{x}_i=[\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2},\ldots, \boldsymbol{x}_{im}]^{\mathrm{T}} \tag{2.1}$$

whose elements are real. The Euclidean distance between a pair of *m*-by-*1* vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is defined by:

$$\begin{aligned} d^2\left(\boldsymbol{x}_i,\boldsymbol{x}_j\right) &= \left\|\boldsymbol{x}_i-\boldsymbol{x}_j\right\| \\ &= \left[\sum_{k=1}^{m}(x_{ik}-x_{jk})^2\right] \end{aligned} \tag{2.2}$$

where x_{ik} and x_{jk} are the *k*th elements of the input vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, respectively. Another alternative metric is the dot product or inner product. Given a pair of vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ of the same dimension, their inner product is $\boldsymbol{x}_i^T\boldsymbol{x}_j$, which can also be written in the following form:

$$\begin{aligned} \left(\boldsymbol{x}_i,\boldsymbol{x}_j\right) &= \boldsymbol{x}_i^T\boldsymbol{x}_j \\ &= \left[\sum_{k=1}^{m}x_{ik}x_{jk}\right] \end{aligned} \tag{2.3}$$

These two metrics are actually related to each other with the following relation: the smaller the Euclidean distance is, the larger the dot product will be. To prove this we must normalize the vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ to have unit length, that is,

$$\left\|\boldsymbol{x}_i\right\| = \left\|\boldsymbol{x}_j\right\| = 1 \tag{2.4}$$

From the following equation, we can see the relationship between these metrics:

$$\begin{aligned} d^2\left(\boldsymbol{x}_i,\boldsymbol{x}_j\right) &= \left(\boldsymbol{x}_i-\boldsymbol{x}_j\right)^T\left(\boldsymbol{x}_i-\boldsymbol{x}_j\right) \\ &= 2-2\boldsymbol{x}_i^T\boldsymbol{x}_j \end{aligned} \tag{2.5}$$

The previous equation shows that minimization of the Euclidean distance corresponds to maximization of the inner product and vise versa.

The neuron with the strongest response -smallest metric distance- to the input vector prints its output on a 2-D map. The weight vectors of the winning neuron along with its neighbour neurons are moved towards the input vector in order to produce a stronger response to the same input vector in the future. After a sufficient number of iterations some of the topological relations in the input space are retained and the weight vectors become an ordered mapping of it. Close-by areas on the map will represent (with high probability) related feature vectors. In the case under consideration, the collection of documents or corpus, as well as, the words used by these documents were represented as vectors. These vectors are the feature vectors used during the training phase of the application.

The application presented in the next section follows the connectionist and statistical approach in the IR community, which is observed during the last decade

towards the statistical techniques in natural language processing [12]. The motivation for that shift is the close cooperation of natural language understanding community with the speech recognition community where statistical techniques, such as Bayesian decision theory, entropy and cross entropy based techniques, Hidden Markov Models, vector quantization techniques, etc., are well founded and constitute a common practice.

3. Application

Our work is focused on fulfilling the needs for tourist information. For this purpose, a collection of 633 full-text documents containing a total number of nearly 124000 words (word tokens) is made. The corpus consisted of web pages manually selected over the Internet following the following criterion:

- Representativeness (most important)
- English documents
- Finite size

At that point we must note that it is a biased corpus, meaning that the web pages that are contained in it are contextually related with Greece, Spain and Germany. That was done for comparison reasons only.

Before applying the method, the corpus was processed in a series of steps in order to fulfil certain criteria. Firstly, it was annotated following certain annotation rules about names, tourist objects, terminology words and acronyms. Then all the HTML tags and unwanted textual part (e.g., numbers, symbols, articles, determiners, complementizers, prepositions, pronouns and punctuation marks) were removed. Some rarely seen words were removed as well. The non-English terms found in the text files were not removed automatically, rather, by hand. This was done, because many of those words are actually the names of places or local foods, and this information is extremely valuable to be lost in an automatic process.

After that, the word suffixes were eliminated leaving behind only the roots of the words in a process known as stemming [13]. A total number of 40857 word stems (stem tokens, i.e., occurrences of stem types) remained after that step, in which the Porter stemmer was used [14]. Using the word stems instead of the word tokens reduces the total size of the vocabulary and therefore speeds up the overall process.

A further step is the computation of the contextual statistics related to each word i in the corpus. For that reason one of the following vectors can be used in calculating those statistics [15,16]:

$$\widetilde{\boldsymbol{x}}_i = \frac{1}{N_i}\sum_l n_{li}\boldsymbol{e}_l \tag{3.1}$$

or

$$\widetilde{\boldsymbol{x}}_i = \frac{1}{N_i}\begin{bmatrix} \sum_{\substack{l=1 \\ l\neq i}}^{N_w} n_{li}\boldsymbol{e}_l \\ \varepsilon \boldsymbol{e}_i \\ \sum_{\substack{m=1 \\ m\neq i}}^{N_w} n_{im}\boldsymbol{e}_m \end{bmatrix} \tag{3.2}$$

where n_{li} is the number of times the l-th word occurred before the i-th word. By $\boldsymbol{e}_l$ and $\boldsymbol{e}_m$, we denote the vectors that in principle have unit in the l-th and m-th component, respectively, and zeros elsewhere. Finally, N_w is the number of unique stems found in the corpus (N_w=8709) and ε is a small scalar parameter. Equations (3.1) and (3.2) rely on the concept of bigrams. Equation (3.2) was used in order to calculate a feature vector for each word stem.

The dimension of the feature vectors derived from the equation (3.2) is equal to $3N_w$, which is exceptionally high. Therefore, a further step towards dimensionality reduction is taken. A wealth of techniques can be applied ranging from feature extraction to multi-dimensional scaling. Among them are factor analysis, Latent Semantic Analysis (LSI), linear and non-linear Principal Component Analysis (PCA) and Independent Component Analysis (ICA). Unfortunately, all these methods cost in computations. On the other hand, random mapping [17,18] performs equally well without being computationally expensive. In random projection or random mapping each $3N_w$-by-1 dimensional feature vector is transformed into a new m-by-1 dimensional vector using a m-by-$3N_w$ random matrix $\boldsymbol{R}$. Random projection is defined in the following equation:

$$\boldsymbol{x}_i = R_{m\times 3N_w}\,\widetilde{\boldsymbol{x}}_i \tag{3.3}$$

If $m<3N_w$ then the output of equation (3.3) is a vector whose dimension is lower than the dimension of the original space.

All the feature vectors are presented iteratively an adequately number of times to the network. The network, in two successive steps, finds the winning neuron -the most correlated neuron on the lattice in respect to the input vector. Let us, denote by $\boldsymbol{m}_i(t)$ the weight vector of the winning neuron, where t=0,1,2, … is an integer, the discrete-time coordinate, and $\boldsymbol{x}_i$ is the

feature vector used for updating. The index number is given by:

$$c = arg \min_{\forall j} \left\{ \left\| \boldsymbol{x}_i - \boldsymbol{m}_j(t) \right\| \right\} \qquad (3.4)$$

And, then modifies the weight vectors towards $\boldsymbol{x}_i$. The following equation gives an example of the updating process:

$$\boldsymbol{m}_j(t+1) = \boldsymbol{m}_j(t) + h_{cj}(t)\left[\boldsymbol{x} - \boldsymbol{m}_j(t) \right] \qquad (3.4)$$

In equation (3.4), the function $h_{cj}(t)$ is the so-called neighborhood function, which is a smoothly decaying kernel centered on the winning neuron. The neighborhood function used during the training phase has the following form:

$$h_{cj}(t) = a(t) exp\left(-\frac{\left\| r_c - r_j \right\|^2}{2\sigma^2(t)} \right) \qquad (3.5)$$

where $a(t)$ is a time varying scalar-valued factor, the so-called learning rate, and $\sigma(t)$ is a time varying factor also, which denotes the diameter of the updating kernel. Finally, r_c and r_j is the location on the lattice of the winner neuron and the neuron being updated, respectively. The impact of the updating kernel and the diameter of the kernel is the following: iterative repetitions of the training phase leads to global ordering of the map.

The above-described procedure yields the so-called *word category map* (WCM). Each node in the WCM is labeled by the words that are mapped on it (Fig. 1). The various levels of grey colour on the map correspond to different word densities in various neurons. Greyl evels near to 255 (white colour) imply that a few stems have been assigned to the node, whereas, levels of grey near 0 (black colour) mean heavier densities.

After that, for each document in the corpus a word category histogram is computed. In order for the histogram to be calculated, the document is stemmed and then for each word stem in the document the winning neuron in the WCM is found. The neuron's position on the lattice (index number) specifies the category or bin on the histogram where it belongs. The histogram is then updated for that word category. After the construction of the histogram, smoothing is applied in order to enhance the invariance to small changes. The process of smoothing can be described in the following way: all the neighbor bins corresponding to the histogram peak are modified according to their distance from the histogram peak with the help of a Gaussian kernel. In order to achieve this, the histogram is convolved with the kernel. By normalizing

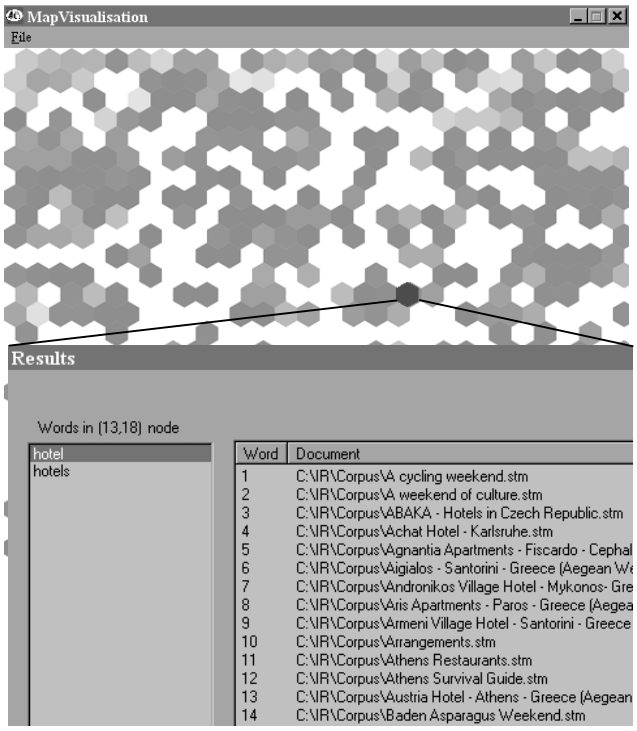

Fig. 1: The WCM for 8709 stem on 27x27 lattice. In the overlaid table one may see which word are assigned in the (13,18) node, and the documents that containing these words.

the document histogram we produce the document vector $\boldsymbol{a}_k$. An important note here is that the dimensionality of the document vector equals the total number of neurons on the first neural network used for the construction of the WCM (number of neurons on the grid).

When the construction of the document vectors is finished, the SOM algorithm is used once again to construct a map for the collection of the document vectors. The output of this step is the so-called *document map* (DM) which can be seen in Fig. 2. In this step, the neurons constituting the lattice are less than those used to create the WCM. In our experiments we used varying dimensional grids: from 15 to 27 neurons per side for the WCM and 5 to 9 neurons for the DM.

The major advantage of the above-described method is that apart from keyword-based searching on the clustered documents, it allows searching of the document collection for documents contextually relative to a given, user-defined, document. A brief description of the recall phase is the following:

- In keyword based search, the stems of the words in the query are searched on the WCM. Then with the help of inverted files these categories are searched in the collection of the documents. The files retrieved at

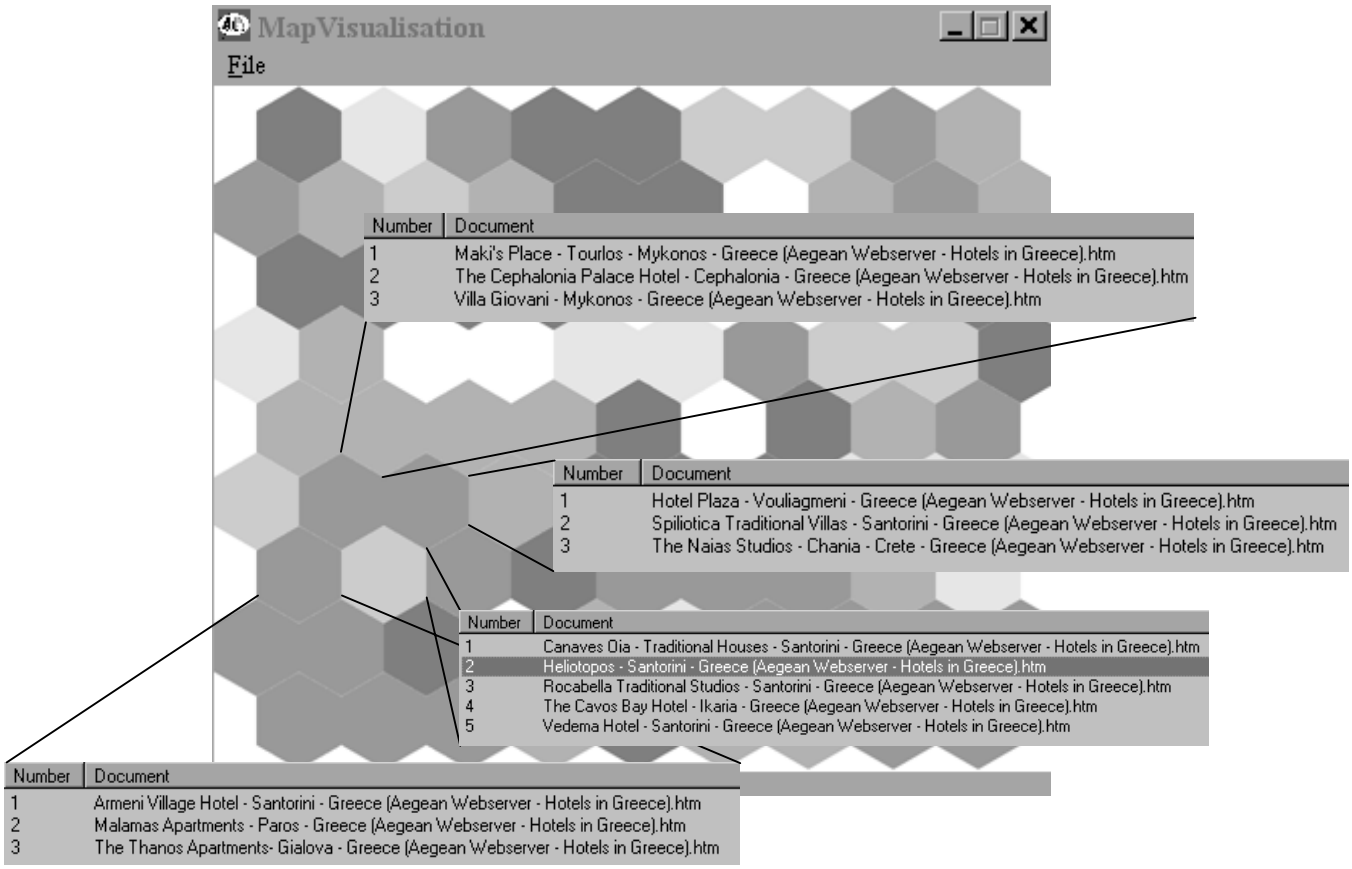

Fig. 2: The DM for a 9x9 and 633 document vectors. A visual inspection of the map shows well separated areas containing contextually relevant documents.

this stage contain not only the keywords in the query but also words that are semantic relative to those words.

- In order to search the collection of the documents using a user-defined document, that document is processed according to the series of the steps described in the construction of the WCM. Then, with the help of the WCM, which has been constructed in the training phase, we construct the corresponding document vector. This vector is then compared with the weights of the neurons in the DM. The winning neuron is located and the documents assigned to that neuron are expected to be contextually related to the testing document.

4. Conclusion

The main purpose of this paper is to present the work done so far in the Hypergeo project towards document organization and retrieval for tourist information. Although the training set has a small size, the results ere encouraging. The corpus contained 15 well separable groups of documents -if they were clustered by a human- and the network separated them almost accurately. In the future, when larger data sets of domain specific material will be available, large-scale experiments will be performed. Also, tasks that were performed manually in the first experiments will be dealt in a more automatic way in the future (web crawlers collecting domain specific material, annotation etc).

5. References

[**1**] R. Baeza - Yates and B. Ribeiro - Neto, *Modern Information Retrieval* (ACM, 1999).

[**2**] M. W. Berry and M. Browne, *Understanding Search Engines: Mathematical Modelling and Text Retrieval* (SIAM, 1999).

[**3**] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval* (McGraw Hill, 1983).

[**4**] S. Kaski, K. Lagus, T. Honkela, and T. Kohonen, Statistical aspects of the WEBSOM System in Organizing Document Collections, *Computing Science and Statistics*, 29, 1998, 281-290.

[**5**] T. Kohonen, *Self-Organizing Maps* (Springer Verlag, 1997).

[**6**] T. Kohonen, Self-organization of very large document collections: State of the art, *Proc. of the 8th Int. Conf. on Artificial Neural Networks*, 1, Springer, 1998, 65-74.

[**7**] H. Ritter and T. Kohonen, Self-Organizing Semantic Map, *Biol. Cybernetics*, 61, 1989, 241-254.

[8] S. Haykin, *Neural Networks, A Comprehensive Foundation* (Prentice-Hall, Inc, 1999).

[**9**] T. Kohonen and S. Kaski and K. Lagus and J. Salojarvi and J. Honkela, and V. Paatero and A. Saarela, Self-Organization of Massive Document Collection, *IEEE Trans. On Neural Networks*, 11:3, 2000, 574-585.

[**10**] T. Honkela, Self-Organizing Maps in natural language processing, *PhD Thessis*, Helsinki University of Technology, 1997.

[**11**] S. Kaski, Data exploration using self-organizing maps, *PhD Thessis*, Helsinki University of Technology, 1997.

[**12**] D. Manning and H. Schutze, *Foundation of Statistical Natural Language Processing* (MIT Press, 1999).

[**13**] W. B. Frakes and R. Baeza - Yates, *Information Retrieval: Data Structures and Algorithms* (Prentice-Hall, Inc, 1992).

[**14**] M. F. Porter, An algorithm for suffix stripping, *Program*, 14, 1980, 130-137.

[**15**] C. Becchetti and L. P. Ricotti, *Speech Recognition: Theory and C++ Implementation* (Wiley, 1998).

[**16**] M. P. Oak, *Statistics for Corpus Linguistics* (University Press, 1998).

[**17**] S. Kaski, Dimensionality reduction by random mapping: Fast similarity computation for clustering, *Proc. of the 8th Int. Conf. on Neural Networks*, IEEE, 1, 1998, 413-418.

[**18**] S. Kaski, Fast winner search for SOM-based monitoring and retrieval of high-dimensional data, *Proc. of the 9th Int. Conf. on Artificial Neural Networks*, Edinburgh, U.K., September 1999.