

# SOM-based Document Image Retrieval

Stefano Faini Simone Marinai Emanuele Marino Giovanni Soda  
DSI - University of Florence (Italy)  
E-mail: marinai@dsi.unifi.it

## Abstract

*In this paper we discuss some applications of word image clustering (based on Self Organizing Maps, SOM) for tasks related to document image retrieval.*

*Two main applications are discussed: document retrieval and word retrieval. In document retrieval a document representation based on the vector model is obtained by computing the occurrences of words belonging to the SOM clusters in each document. In word retrieval the combination of the SOM clustering with a Principal Component Analysis based space reduction allows us to efficiently retrieve matching words from large documents collections.*

## 1. Introduction

Document Image Retrieval (DIR) aims at finding relevant documents relying on image features only. Important sub-tasks involve the retrieval by layout similarity and the text based retrieval. Digital Libraries are application domains where the use of DIR techniques is more and more appropriate. In this paper we address the text based retrieval by exploring the effectiveness of a SOM-based word image clustering, where words are grouped considering the image similarity. The idea of using this clustering technique is extended from a previous approach that exploited SOMs for character-like object clustering [?]. The main advantage of character (and word) clustering is the independence with respect to various language and fonts. In particular the approach seems to be more appropriate for dealing with old documents printed with non-standard fonts as well as for dealing with noisy documents (containing words with touching characters). However, one significant restriction of word clustering is the need to compute a new clustering when different documents need to be addressed. It is therefore more appropriate when uniform document collections (containing several documents with similar features) need to be stored.

In the proposed approach after a preliminary SOM training step, two main retrieval strategies are performed:

- In **Document Retrieval** the query is made by one page in the collection and more similar pages are ranked according to the estimated text similarity.
- In **Word Retrieval** we look for occurrences of a user-defined word in the collection.

The paper is organized as follows: in Section 2 we describe the use of SOM for word image clustering. In Section 3 and Section 4 we analyze the document retrieval and the word retrieval methods, respectively. Finally, some conclusions are drawn in Section 5.

## 2. Self Organizing Maps of word images

The Self Organizing Map (SOM [?]) is a special kind of artificial neural network that is based on competitive learning algorithms, where the output neurons of the network compete among themselves. In the Self-Organizing Map the neurons are arranged in a two dimensional lattice (feature map). Each neuron receives inputs from the input layer (vectors in  $R^n$ ) and from the other neurons in the map. During the learning the network performs clustering and the neurons are moved in the lattice so as to reflect cluster similarity by means of distances in the map. To each element in the SOM map it is associated one real vector (in  $R^n$ ) that can be considered as a prototype of the patterns in the cluster.

One advantage of the use of SOM for word clustering is the spatial organization of the feature map that is achieved after the learning process. Basically, more similar clusters are closer than more different ones. Consequently, the distance among prototypes in the output layer of the SOM can be considered as a measure of similarity between words in the clusters. This feature will be exploited particularly for word retrieval as discussed in Section 4.

**Figure 1. Distribution of words for each neuron of the 60x40 SOM related to partition number 4.**

In our model the SOM is used to build a word image database where the words are grouped in clusters contain-

ing similar words (from a graphical point of view). Relying on this collection it is possible to design the *Document Retrieval* and *Word Retrieval* systems.

## 2.1. Preprocessing

During the indexing each document image is analyzed by means of one layout analysis tool that identifies the text regions and extracts the words by means of an RLSA-based algorithm. The indexed words are then split into six disjoint partitions on the basis of the word aspect-ratio (the ratio between the word height and width). In this way the words in each partition have a similar aspect-ratio and clustering is performed on uniform data. For each partition we compute also the average word width and height, referred to as *normalized dimensions* for the  $i$ -th partition.

Neuron 59-6	Neuron 5-12

**Figure 2. Examples of the contents of two neurons. For each neuron we show the closest words (on the left) and the farthest words (on the right).**

During the indexing for each word its aspect-ratio is computed and considered to find the right partition. Next, the word image is linearly scaled to the normalized dimensions, obtaining a vectorial word representation where each vector item contains the average gray level of the pixels belonging to the corresponding grid cell. The main problem of this approach is the high vector size (hundreds of items) that is reflected into a long training time. However, it should be remarked that this size is a problem only for the training performed during indexing, but it is not important for the document retrieval and the word retrieval.

To have an idea of the trained SOM we report in Figure 1 the distribution of words assigned to each neuron. We can remark that the words are homogeneously assigned to neurons and that peaks on the corners (a typical problem of the SOM training) are not too high. We designed one tool that allows us to display the words corresponding to user defined neurons (the words closest to the neuron centroids). By using this tool we verified that it is very unlikely that one cluster contains only instances of one unique word (with the exception of few clusters corresponding to stop words). In general the farthest words are loosely related with those closer to the centroid (as an example see Figure 2). To mitigate this problem we slightly modified the SOM training algorithm as described in the following.

## 2.2. Modified training algorithm

For the SOM training we use the SOM\_PAK package<sup>1</sup> that implements the standard incremental learning algorithm ([?] page 109). Basically, after each training step the prototype of each neuron is “moved” in the  $R^n$  space so as to better represent the words belonging to its cluster and to its neighborhoods. This is obtained by replacing the prototype with the arithmetic mean of the vectors of the patterns belonging to the clusters in the neighborhood. In so doing the code vectors usually do not correspond to “actual words” unless the corresponding words are very homogeneous as in the case of stopwords (e.g. see the word ‘une’ in Figure 3).

**Figure 3. Eight prototypes obtained from a SOM map trained with the standard training algorithm.**

One solution to the above mentioned problem relies on the update of each prototype with the closest training vector among all the patterns in the neighborhood (instead of computing the arithmetic mean). This prototype update is made after each training epoch<sup>2</sup> and also at the end of the training. The update is made by scanning the training set associating each input pattern to the closest prototype; when all the input patterns have been associated then we replace the prototype with the closest associated pattern. This solution provides good results since the final map is uniform and the prototypes usually represent the most frequent words in the dataset. The prototypes shown in Figure 4 are obtained with the modified training algorithm.

**Figure 4. Some prototypes corresponding to a map trained with the modified training algorithm.**

## 3. Document Retrieval

In this section we describe the use of the SOM-based word image clustering to perform document retrieval. The proposed strategy starts from well known methods addressed in text-based Information Retrieval.

One well known approach to perform document retrieval is the *vector model* (see [?], Chapter 2) that is based on a vectorial description of the document textual contents. The vector items are related to the occurrences of index terms, that usually correspond to words in the document. Vector values are weighted in order to provide more importance to

<sup>1</sup> The package can be downloaded at: [http://www.cis.hut.fi/research/som\\_lvq\\_pak.shtml](http://www.cis.hut.fi/research/som_lvq_pak.shtml).  
<sup>2</sup> The epoch is a cycle of presentation of the patterns to the network.

most discriminant terms. To this purpose one common approach relies on the well known *tf-idf* weighting scheme. Basically, index terms that are present in many documents of the collection have a lower weight since their presence is not discriminant. With the *tf-idf* approach the weight assigned to the  $k$ -th word in the document  $D_i$  is computed by Eq. (1)

$$w_{i,k} = f_{i,k} \cdot \log\left(\frac{N}{n_k}\right) \quad (1)$$

where  $f_{i,k}$  is the frequency of the  $k$ -th word in  $D_i$  normalized with respect to the maximum word frequency in  $D_i$ ,  $N$  is the total number of documents, and  $n_k$  is the number of documents containing the  $k$ -th word. The vector model has been designed in order to deal with symbolically encoded documents where the word identification and grouping (with possible stemming and stopword removal) is quite straightforward.

The application of SOM-based word clustering to perform document retrieval relies on the use of SOM clusters in lieu of words in the previous method based on the *tf-idf* weighting approach. Basically, a document (in our case a page) is represented by a vector whose items correspond to the neurons of the six SOMs obtained for each partition. Each item contains the number of words in the page that are assigned to the corresponding neuron (e.g. the words that are closer to the neuron prototype). Similarly to the vector model we apply the *tf-idf* weighting to this representation. It is worth to remark that even if the combination of the partitions provides a large number of neurons, the size of the overall vector is smaller than the typical size of a dictionary considered with the vector model (for instance for the well known *Reuters 21578* corpus the dictionary contains around 19,000 terms).

### 3.1. Retrieval

At the end of the indexing each page is represented with a vector containing occurrences of words corresponding to the SOM neurons of the six partitions. A similar vector is computed from the query page and the similarity of the pages in the database is evaluated by comparing the query page with each indexed page. The similarity between  $\vec{a}$  and  $\vec{b}$  is computed by taking into account the *cosine of the angle* between the two vectors Eq. (2)

$$\text{sim}(a,b) = \frac{\sum_{i=0}^{n-1} (a_i \cdot b_i)}{\sqrt{\sum_{i=0}^{n-1} a_i^2} \cdot \sqrt{\sum_{i=0}^{n-1} b_i^2}} \quad (2)$$

To obtain a global ranking of the indexed words we compute the similarity of all the pages with respect to the query and we sort the pages on the basis of the measure computed in Eq. (2).

### 3.2. Stopwords

Textual information retrieval systems usually address in some specific way the stopwords. When dealing with most Western languages the stopwords are frequently removed from the index in order to reduce the size of the dictionary. We extended this process to our approach generating a graphical model of each stopword by means of the  $\text{\LaTeX}$  package. Afterwards, we represent the word with the approach described in Section 2 and we remove from the collection the words that are closer to the image of the stopword than a given threshold.

### 3.3. Experiments

The experiments described in this paper are made on two books (containing 1280 pages) that are part of an encyclopedia of the XIX<sup>th</sup> Century<sup>3</sup>. This encyclopedia addresses machineries and techniques of the industry of the XIX<sup>th</sup> Century. The images are quite clean and the OCR works well on these documents. From the document retrieval point of view the relevance of this dataset lies on the homogeneity of the contents of pages in each chapter, so that the evaluation is simplified.

An accurate evaluation of document retrieval systems is frequently based on the judgment of relevance provided by human experts. In our application this judgment is not available and we evaluate the system in two ways. First we made a rough evaluation of the whole data-set by performing several queries for each chapter; then we carefully analyzed some queries by checking the contents of the neurons providing the highest contribution to the similarity.

**3.3.1. Precision-recall plot** To roughly evaluate the document retrieval we made one simple assumption, defining one page as relevant for a given query if the two pages belong to the same chapter. Even if not appropriate for providing an overall score for the performance of the document retrieval, this approach seems to be suitable to compare different strategies and assess the optimal values of some parameters. The experimental results have been evaluated by taking into account one set of query pages for each chapter that have been selected considering only pages with enough information (thus avoiding pages with few text such as picture pages).

The *precision-recall* plots are computed considering the first 50 pages ranked by the program, computing a correct answer for the pages belonging to the chapter of the query. The plots reported in Figure 5 are obtained by averaging the plots computed for 7 queries made by randomly selecting pages in the corresponding chapter. These plots are aimed

<sup>3</sup> *Les Merveilles De l'Industrie*: downloaded from the web site of the National Library of France.

at evaluating the effect of the stopword removal. We can remark that for most chapters the precision-recall plots are improved when removing the stopwords.

**3.3.2. One example query** To complement the analysis provided by the precision-recall plots we analyze in the following the results obtained with one query identifying the most important neurons for each selected page.

The query page (number 452) belongs to chapter 4 'sodium and potassium' and contains the description of one specific machinery with several technical terms (e.g. *cylinder, wheels, oven, movement, combustion, handle, heat, worker, rotation*). This machinery is used for the production of sodium and therefore the following words are contained in the page: *sulfate, reaction, carbonate, chaux* (lime).

Rank	Page	Sim.	Most frequent words
1	448	0.1624	<i>carbonate, charbon, sulfate</i> ( <i>carbonate, coal, sulphate</i> )
2	446	0.1492	<i>four</i> ( <i>oven</i> )
3	1007	0.1368	<i>mouvement, cylindre, ouvrier, roues</i> ( <i>movement, cylinder, worker, wheels</i> )
4	822	0.1348	–
5	1164	0.1320	similar to 1007
6	1254	0.1313	–
7	455	0.1309	<i>ammoniaque, sulfate</i> ( <i>ammonia, sulfate</i> )

**Table 1.** Most frequent words in the top ranked pages of the example query.

Among the ranked pages (see Table 1) there are few pages belonging to chapter 4 (pages 419-480), since in the query page there are few references to sodium and potassium.

Table 1 reports the most frequent words in the neurons providing the higher contribution to Eq. 2 for the first ranked pages. Pages 448 and 455 belong to the same chapter as the query page. Page 446 describes a machinery very similar to the machinery 'four tournants' addressed in the query. Likewise, the machineries described at page 1007 and 1164 are very similar to the query, but are designed to work with other materials.

## 4. Word Retrieval

**Figure 6.** Main steps in the *Word Retrieval*.

In this section we describe the word retrieval system built on the basis of the SOM word clustering. Without taking into account efficiency issues, one simple approach would rely on the linear comparison of the indexed words with a query word image. However, this approach is not feasible when large word databases are considered. The proposed

method takes into account the main features of SOM clustering in order to efficiently address the word retrieval problem, as described in the following. When dealing with large document collections we can expect to have many occurrences of words in each neuron. The search for the words closest to the query is therefore a challenging problem, due to the high-dimensionality of data (for instance the words in partition 4 are encoded with 684 dimensional vectors). Efficient search in high dimensional vector spaces is still the subject of active research. When dealing with low dimensional spaces, then the R-tree [?] (and its variants) can be adopted to reduce the search cost from the linear one. Some methods (e.g. X-Tree [?] or cluster tree [?]) have been proposed to search in high dimensional spaces, however these methods degenerate to the linear complexity when dealing with spaces having a few dozens of dimensions. We therefore suggest to reduce the search complexity by projecting the data in each cluster with Principal Component Analysis (PCA) and then perform an efficient search in the projected space with an appropriate search algorithm (e.g. the X-tree). Principal Component Analysis (PCA) projects  $n$ -dimensional data onto a lower dimensional subspace in a way that is optimal in a sum-squared error sense.

Let us describe the four main steps of the word retrieval algorithm (sketched also in Fig. 6).

### 1. The query

From the user point of view the queries are made with a simple text-based interface. Starting from the ASCII word one word image prototype is produced with  $\text{\LaTeX}$  software. This word image is then used similarly to the indexed words: we identify the partitions to be considered, and we scale the word image according to the average dimensions of the two partitions obtaining a vectorial word representation (Fig. 7).

**Figure 7.** From top: prototype generated with  $\text{\LaTeX}$  and the two scaled images required for partition 4 and 5, respectively.

### 2. Cluster search

In the second step we identify the three SOM clusters closer to the query to restrict the subsequent search. To this purpose we compare the query with all the cluster centers.

Since we compare the stored words with a query generated by  $\text{\LaTeX}$  (and due to the presence of noise in indexed words) it is unlikely that the most similar indexed words are contained in the closest neuron. To solve this problem, we take into account the three closest neurons (the number of neurons to retain has been

**Figure 5.** Comparison of precision-recall plots with (red) and without (green) the stopwords. From left to right: chapter 5, 6, 8, respectively.

obtained after some preliminary tests). On the average we reduce the number of patterns to be processed by the subsequent step by a factor 800 (the total number of neurons is 2400).

**Figure 8.** Left to right: the first words in the three clusters closest to the prototype generated for the query *al-cool*.

As we can expect some correct words can belong to some additional clusters and not to the three selected in this step. This is due to several problems such as differences between the indexed words and the L<sup>A</sup>T<sub>E</sub>X generated query and the presence of words printed with different fonts. To illustrate the need for the multiple cluster search we show in Figure 8 the first words in the three clusters closest to the prototype generated for the query *al-cool*. The closest cluster (the leftmost) contains only a few instances of the word *al-cool*, the second cluster does not contain any occurrence of the word, whereas the last one is the most populated.

### 3. Search in clusters

After obtaining the closest neurons we need to search among the patterns corresponding to the cluster by means of PCA.

Let us briefly recall the basic approach in Principal Component Analysis (e.g.[?], pag 568). The following procedure is repeated for each SOM cluster during indexing in order to build a lower dimensional hyper-plane for each cluster. We first compute the mean vector  $\mu$  and the  $n \times n$  covariance matrix  $\Sigma$  of the data in the cluster. The eigenvectors and eigenvalues of  $\Sigma$  are computed and the eigenvectors are sorted according to decreasing eigenvalue. Let  $e_1, e_2, \dots, e_h$  be the first  $h$  eigenvectors ( $\lambda_1, \lambda_2, \dots, \lambda_h$ , are the corresponding eigenvalues) that can be combined as columns of the  $n \times h$  matrix  $A$ . At this point it is possible to project the data in the cluster onto the  $h$ -th dimensional sub-space according to

$$x' = A^t(x - \mu) \quad (3)$$

After the SOM training, the patterns belonging to each cluster are used to compute the mean vector  $\mu$  and eigenvector matrix  $A$  and next we project the patterns in the cluster.

During the search in the three clusters we look for the  $k$  nearest neighborhoods in the projected space. As

we can expect the PCA projection approximates the proximity in the  $n$  dimensional space thus requiring a refinement in the next step. In Figure 9 we compare the rankings obtained in the projected space and in the original one for the words belonging to two clusters when looking for the word *fromage* that is very rare in this collection. The cluster (24,18) contains one occurrence of the word that is put in the first position in the  $R^{684}$  space, but is put in the 10th position in  $R^{10}$ . Two correct words in the other cluster are put in the first positions in both spaces.

Neuron (24,18)		Neuron (25,19)	
$R^{10}$	$R^{684}$	$R^{10}$	$R^{684}$

**Figure 9.** Ranking of the 20 words closest to the query *fromage*. We show two neurons and the ranking in the original space ( $R^{684}$ ) and in the projected one ( $R^{10}$ ).

### 4. Final ranking

The previous step allows us to identify the most similar words in the projected spaces of the three closest neurons. In order to merge the three lists and refine the final ranking we compute the distance between the query word and each word in the three lists in the original space. It is worth to remark that the computation in the original space is performed for a small number of words and therefore it is not problematic from the computational point of view. In Figure 10 we summarize the final ranking obtained by merging the three lists for the query *fromage*.

**Figure 10.** Query *fromage*: left the rankings corresponding to the three main clusters; right: the final ranking.

## 4.1. Experiments

In order to evaluate the word retrieval system we built a sort of ground-truth by running one commercial OCR on the dataset previously mentioned. The OCR has high recognition performance on this data-set and can be considered as a reasonable approximation of a human validated ground-truth.

The words to be used as queries have been selected from each partition, with a particular emphasis on longer words.

Basically, we had two types of query words: rare words (a few occurrences among the 1280 pages) and frequent words (occurring hundreds of times in the data-set). Tables 2 and 3 summarize the results obtained for some of the test words. In these tables OK denotes the number of correct words found in the list of the 20 most similar words; PR denotes the precision, e.g. the number of subsequent correct words reported at the top of the list; OCR denotes the number of words reported by the OCR (> 20 means more than 20 occurrences). For the following words all the first 20 answers are correct ones: *Canada, alcool, peaux, violet, France, Louis, nombre, cylindre, verre, chaque, lorsqu, aniline, volume, soluble, lessive, culture, sulfure, production.*

Word	OK	PR	OCR
provient	17	15	>20
femme	19	19	>20
raison	17	9	>20
porter	19	17	>20
terrain	17	17	>20
baguettes	13	11	>20
Bontemps	13	13	>20
Egypte	13	8	>20
chaud	15	1	>20
outil	15	0	>20
vive	11	10	>20
vent	11	1	>20
large	4	4	>20
moulage	18	16	>20
graines	13	8	>20
lignes	11	3	>20
abondance	14	14	>20
demande	11	8	>20
enfants	16	15	>20
incolore	19	15	>20
bambou	10	10	>20
explication	10	10	>20
proportion	17	17	>20
bronze	16	15	>20
Deux	19	15	>20

**Table 2.** Performance of word retrieval for some frequent words.

## 5. Conclusions

We described a general system for performing document image retrieval by means of a SOM-based word image clustering. The aim is to build some tools able to deal with different languages and fonts.

The two proposed applications are in the fields of document retrieval and word retrieval. In the case of document retrieval we adapted the vector model replacing the concept of word with the concept of word cluster. Even if this process has some lack of information, the preliminary results here reported are encouraging. The proposed approach for word retrieval addresses the efficiency issues related to the proximity search of large quantities of high dimensional

Word	OK	PR	OCR
flacons	15	9	20
provinces	12	11	20
lution	5	5	20
Trois	8	6	18
Giraud	2	2	18
pressions	7	5	17
boule	5	5	17
redable	2	2	15
Savon	10	6	14
intime	11	9	12
drainage	8	8	12
tournesol	1	1	11
chromate	0	0	11
grillage	7	7	10
assemblage	7	6	10
horizon	4	3	10
Marie	2	0	10
Lacroix	1	1	9
lande	5	5	8
Joseph	4	4	8
herbes	4	0	8
Elton	5	4	7
abattre	2	2	6
condiment	2	0	6
japonaise	0	0	6
fromage	3	3	5
mandarin	0	0	5
dosages	4	4	4
barbare	1	1	4
chirurgie	1	1	4
Danemark	1	1	3
kilos	1	1	3
Soyer	1	1	3
Suger	2	2	2
fumage	1	1	2
vasion	1	1	2
Lorenzo	1	1	2
idiot	1	1	2
Buch	2	0	2
lagunes	1	1	1
Elliot	1	1	1
geoises	0	0	1
argentier	0	0	1

**Table 3.** Performance of word retrieval for some rare words.

vectors. To this purpose the SOM-based clustering is used in conjunction with PCA projection of the data in each cluster.

Some points will require additional investigations, namely the introduction of more appropriate word distances as well as the use of efficient algorithms to search in the projected spaces (e.g. the X-tree algorithm). We are addressing also other datasets more appropriate for the proposed approach.