# 14 Self-Organization of Very Large Document Collections

**Teuvo Kohonen, Samuel Kaski, Krista Lagus, Timo Honkela, Jarkko Salojärvi, Jukka Honkela, Vesa Paatero, Antti Saarela, and Antti Ahonen**

In the vast majority of SOM applications, the input data constitute high-dimensional real *feature vectors*. In the SOMs that form similarity graphs of *text documents*, models that describe collections of words in the documents may be used. The models can simply be weighted histograms of the words regarded as real vectors, but usually some dimensionality reduction of the histograms is carried out, as we shall see next.

## 14.1 Statistical models of documents

### 14.1.1 The primitive vector space model

In the basic *vector space model* [1] the stored documents are represented as real vectors in which each component corresponds to the frequency of occurrence of a particular word in the document: the model or document vector can be viewed as a weighted word histogram. For the weighting of a word according to its importance one can use the Shannon entropy over document classes, or the inverse of the number of the documents in which the word occurs ("inverse document frequency"). The main problem of the vector space model is the large vocabulary in any sizable collection of free-text documents, which means a vast dimensionality of the model vectors.

### 14.1.2 Latent semantic indexing (LSI)

In an attempt to reduce the dimensionality of the document vectors, one often first forms a matrix in which each column corresponds to the word histogram of a document, and there is one column for each document. After that the factors of the space spanned by the column vectors are computed by a method called the singular-value decomposition (SVD), and the factors that have the least influence on the matrix are omitted. The document vector formed of the histogram of the remaining factors has then a much smaller dimensionality. This method is called the *latent semantic indexing (LSI)* [2].

### 14.1.3 Randomly projected histograms

It has been shown experimentally that the dimensionality of the document vectors can be reduced radically by a random projection method [3], [4] without essentially losing the power of discrimination between the documents. Consider the original document vector (weighted histogram) $\mathbf{n}_i \in \Re^n$ and a rectangular random matrix $\mathbf{R}$, the elements in each column of which are assumed to be normally distributed. Let us form the document vectors as the *projections* $\mathbf{x}_i \in \Re^m$, where $m \ll n$:

$$\mathbf{x}_i = \mathbf{R}\mathbf{n}_i \ . \tag{78}$$

It has transpired in our experiments that if $m$ is at least of the order of 100, the similarity relations between arbitrary pairs of projection vectors $(\mathbf{x}_i, \mathbf{x}_j)$ are very good approximations of the corresponding relations between the original document vectors $(\mathbf{n}_i, \mathbf{n}_j)$, and the computing load of the projections is reasonable; on the other hand, with the radically decreased dimensionality of the document vectors, the time needed to classify a document is radically decreased.

### 14.1.4 Histograms on the word category map

In the "self-organizing semantic map" method [5] the words of free natural text are clustered onto neighboring grid points of a special SOM. Synonyms and closely related words such as those with opposite meanings and those forming a closed set of attribute values are often mapped onto the same grid point. In this sense this clustering scheme is even more effective than the thesaurus method in which sets of synonyms are found manually.

The input to the "self-organizing semantic map" usually consists of adjacent words in the text taken over a moving window. Let a word in the vocabulary be indexed by $k$ and represented by a unique random vector $\mathbf{r}_k$. Let us then scan all occurrences of word $(k)$ in the text in the positions $j(k)$, and construct for word $(k)$ its "average context vector"

$$\mathbf{x}_k = \begin{bmatrix} \mathrm{E}\left\{\mathbf{r}_{j(k)-1}\right\} \\ \varepsilon\, \mathbf{r}_{j(k)} \\ \mathrm{E}\left\{\mathbf{r}_{j(k)+1}\right\} \end{bmatrix} , \tag{79}$$

where E means the average over all $j(k)$, $\mathbf{r}_{j(k)}$ is the random vector representing word $(k)$ in position $j = j(k)$ of the text, and $\varepsilon$ is a scaling (balancing) parameter. Notice that this expression has to be computed only once for each different word, because the $\mathbf{r}_{j(k)}$ for all the $j = j(k)$ are identical.

In making the "semantic SOM" or the *word category map*, all the $\mathbf{x}_k$ from *all the documents* are input iteratively a sufficient number of times. After that each grid point is labeled by *all those words (k)*, the $\mathbf{x}_k$ of which are mapped to that point. In this way the grid points usually get multiple labels. A sample map is shown in Figure 24.

In forming the "word category histogram" for a document, the words of the document are scanned and counted at those grid points of the SOM that were labeled by that word. In counting, the words can be weighted by the Shannon entropy or the inverse of the number of documents in the text corpus in which this word had occurred (= "inverse document frequency").

The "word category histograms" can be computed reasonably fast, much faster than, e.g., the LSI.

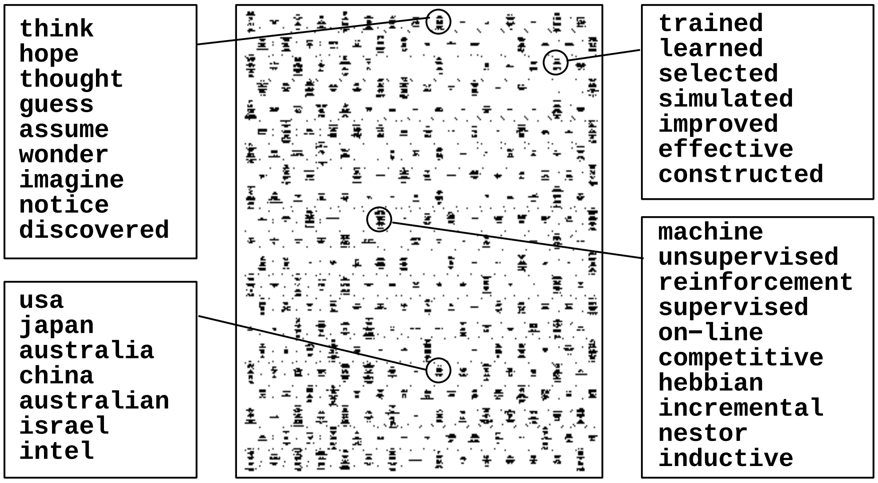


Figure 24: Examples of some clear "categories" of words on the word category map of the size of 15 by 21 nodes. The word labels of the map nodes have been shown with a tiny font on the map grid, and four nodes have been enlarged in the insets.

### 14.1.5 Randomly projected word category histograms

In a great number of experiments performed by us it has transpired that if the histograms on the word category maps are used as models, the ability of our method to discriminate between the documents is reduced if the grid points in the word category map contain more than, say, ten words on the average: specific information contained in the words is then lost. We have been interested in very large document collections that may contain, say, hundreds of thousands of unique words, and even after discarding very rare words, the remaining vocabulary may consist of tens of thousands of words. In order to keep the number of words on each point of the word category map at the tolerable level, the word category map therefore had to be reasonably large, for example 13,432 grid points in some of our latest experiments. The histograms of this dimensionality we then again projected randomly to form 315-dimensional statistical document vectors.

The combination of word categorization and random projection guarantees a certain degree of invariance with respect to the choice of, e.g., synonyms, while a high degree of discrimination between documents can still be maintained, for similar reasons as in the random projection method.

### 14.1.6 Construction of the random projections by pointers

There exists a special method for forming projections that gives as good results as the random projections discussed above but is computationally much more efficient for sparse input vectors (Table 5). The method has been discussed in Section 15.

## 14.2 Construction of the document map

Our original document-organization system named the WEBSOM (`http://websom.hut.fi/websom/`) used word-category histograms as statistical models of the documents. Certain reasons, among them the accuracy of classification, have recently led us to prefer the straightforward random projection (or its shortcut computation by the pointers) in forming the statistical models of the documents. We have carried out numerous experiments with maps of very different sizes; results from a sample comparison have been given in Table 5. In these experiments the word category map had 1598 grid points, and the dimension of the projected model was 270.

Table 5: Classification accuracies in a sample comparison test

| | Matrix product | Pointer method |
|---|---|---|
| | | (3 pointers/column) |
| Random projection | 68.0 | 67.5 |
| Randomly projected word category histogram | 66.0 | 67.0 |

It must also be taken into account that with the word category map method we have to deal with an extra self-organizing process, whereas forming the random projection is a straightforward computation.

Our current method is a collection of programs that can be combined in different ways. A brief overview of the computing phases is given in the following.

**Preprocessing.** From the raw text, nontextual and otherwise nonrelevant information (punctuation marks, articles and other stopwords, message headers, URLs, email addresses, signatures, images, and numbers) was removed. The most common words, and words occuring rarely (e.g., less than 50 times in the corpus) were also discarded. Each remaining word was represented by a unique random vector of dimensionality 90.

For a language like Finnish that has plenty of inflections, we have used a *stemmer*. In our experiments we have so far regarded the various English word forms as different "words" in vocabulary, but a stemmer could be used for English, too.

**Formation of statistical models.** To reduce the dimensionality of the models, we have used both randomly projected word category histograms and randomly projected word histograms, weighted by the Shannon entropy or "inverse document frequency."

**Formation of the document map.** The document maps were formed automatically by the SOM algorithm, for which the statistical models of documents were used as input. The size of the SOM was determined so that on the average 10 to 15 articles were mapped onto each grid point; this figure was mainly determined for the convenience of browsing.

The speed of computation, especially of large SOMs can be increased by several methods. In our latest experiments we have used the Batchmap algorithm (discussed in Section 1) in which the winner search was accelerated. The search was started in the neighborhood of corresponding winners at the last cycle of iteration (discussed in Section 9). The distance computation, or in this case actually computation of inner products, can be speeded up even further by neglecting the components having the value zero; a large proportion of the components are zeroes since the input vectors are still sparse after the dimensionality has been reduced by the random projection with pointers.

The size (number of grid nodes) of the maps was increased stepwise during learning using the estimation procedure discussed in Section 9. After each increase the winners for all data vectors can be found quickly by utilizing the estimation formula that was used in increasing the map size, equation 75 in Section 9. The winner is the map unit for which the inner product with the data vector is the largest, and the inner products can be computed rapidly using the expression

$$\mathbf{x}^T\mathbf{m}_h^{(d)} = \alpha_h\mathbf{x}^T\mathbf{m}_i^{(s)} + \beta_h\mathbf{x}^T\mathbf{m}_j^{(s)} + (1-\alpha_h-\beta_h)\mathbf{x}^T\mathbf{m}_k^{(s)} \; . \tag{80}$$

Here $d$ refers to model vectors of the large map and $s$ of the small map, respectively. The expression (80) can be interpreted as the inner product between two *three-dimensional* vectors, $[\alpha_h;\beta_h;(1-\alpha_h-\beta_h)]^T$ and $[\mathbf{x}^T\mathbf{m}_i^{(s)};\mathbf{x}^T\mathbf{m}_j^{(s)};\mathbf{x}^T\mathbf{m}_k^{(s)}]^T$, *irrespectively of the dimensionality of* $\mathbf{x}$. If necessary, the winner search can be speeded up even further by restricting the search to the area of the larger map that corresponds to the neighborhood of the winner on the smaller map.

For very large maps it may be necessary to save the amount of memory needed for storing the maps. We have represented the model vectors with reduced accuracy and used probabilistic updating to maintain numerical accuracy in adapting the model vectors.

**User interface.** The document map was presented as a series of HTML pages that enable exploration of the grid points: when clicking the latter with a mouse, links to the document data base enable reading the contents of the articles. Depending on the size of the grid, subsets of it can first be viewed by zooming. Usually we use two zooming levels for bigger maps before reading the documents.

There is also an automatic method, discussed in Section 16, for assigning descriptive signposts to map regions; in deeper zooming, more signs appear. The signposts are words that appear often in the articles in that map region and rarely elsewhere.

**Content-addressable search.** The HTML page can be provided with a form field into which the user can type an own query in the form of a short "document." This query is preprocessed and a document vector (histogram) is formed in the same way as for the stored documents. This histogram is then compared with the "models" of all grid points, and a specified number of best-matching points are marked with a round symbol, the diameter of which is the larger, the better the match is. These symbols provide good starting points for browsing.

If the document map is very large, the comparison between the document vector and all the model vectors is time-consuming. It is, however, possible to make rapid approximations by restricting the comparisons in such subspaces of the original space that best represent the (local) organization of the map.

A problem may be encountered if the user wants to use a single keyword or a few keywords only as a "key document." Such queries make very bad "histograms." In this case it is more advisable to use *two different modes of use* of the WEBSOM: the user must then specify whether a document-type or keyword-type query has to be used. In the former case the operation is like described before; in the latter case one has to index each word of the vocabulary by pointers to those documents where these words occur, and use a rather conventional indexed search to find the matches.

## 14.3 Examples

### 14.3.1 The largest published map

The biggest document map we have published so far consists of 104,040 grid points. Each model is 315-dimensional, and has been made by projecting a word category map with 13,432 grid points randomly onto the 315-dimensional space. The text material was taken from 80 very different Usenet newsgroups and consisted of 1,124,134 documents with average length of 218 words. The size of the finally accepted vocabulary was 63,773 words. The words were weighted by the Shannon entropy computed from the distribution of the words into 80 classes (newsgroups). It took about 1 month to process the two SOMs without our newest speedup methods; searching occurs in nearly real time.

The accuracy of classifying a document into one of the 80 groups was about 80 per cent.

Fig. 1 exemplifies a case of content-addressable search. The document map has been depicted in the background, and the shades of gray correspond to document clusters. The 20 grid points, the models of which matched best with the short query, are visible as a small black heap on the left-hand side of the document map. Using a browser, the documents mapped to grid points of the document map can be read out from the HTML page. Two title pages are shown.

Actually there is only one article in Fig. 1 that deals with NN chess. However, the other computer chess documents were so similar that they were returned, too. About one fourth of the found documents obviously does not deal with chess.

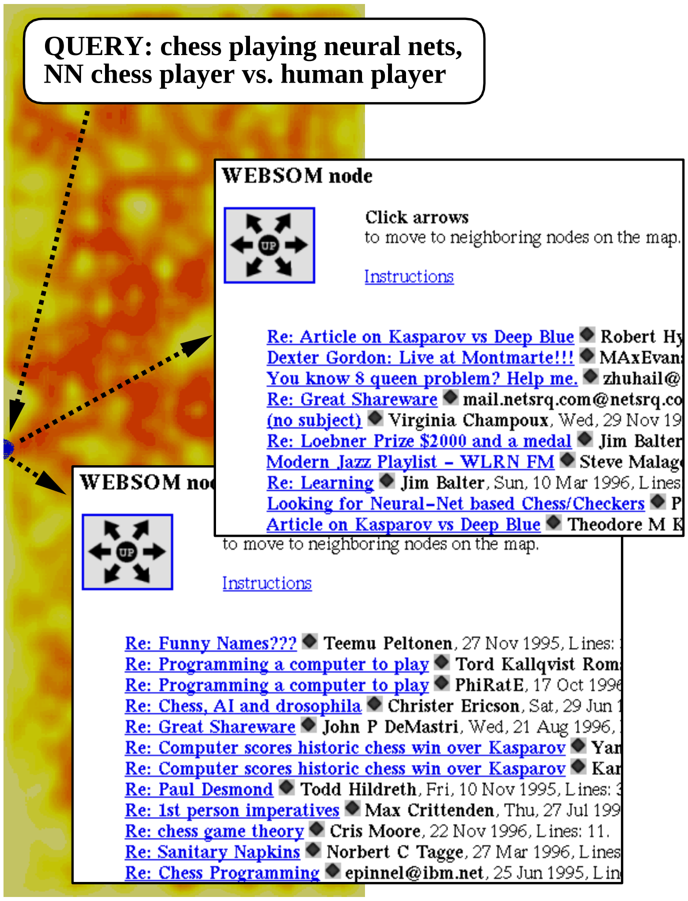
### 14.3.2 The largest map being processed

We are currently finishing the computation of a map of all of the patent abstracts in the world that are available in electronic form, about 7,000,000 in total. The map consists of about 1,000,000 units.

## 14.4 Conclusions

We have demonstrated that it is possible to scale up the SOMs in order to tackle very large-scale problems. Additionally, it has transpired in our experiments that the encoding of documents for their statistical identification can be performed much more effectively than believed a few years ago [2]. In particular, the various random-projection methods are as accurate in practice as the ideal theoretical vector space method, but much faster to compute than the eigenvalue methods (e.g., LSI) that have been used extensively to solve the problem of large dimensionality.

The content-addressable search must obviously be implemented differently when complete new "documents" are used as key information vs. when only a few key-



QUERY: chess playing neural nets, NN chess player vs. human player

**WEBSOM node**

**Click arrows**
to move to neighboring nodes on the map.

Instructions

Re: Article on Kasparov vs Deep Blue ◆ Robert Hy
Dexter Gordon: Live at Montmarte!!! ◆ MAxEvan
You know 8 queen problem? Help me. ◆ zhuhail@
Re: Great Shareware ◆ mail.netsrq.com@netsrq.co
(no subject) ◆ Virginia Champoux, Wed, 29 Nov 19
Re: Loebner Prize $2000 and a medal ◆ Jim Balter
Modern Jazz Playlist – WLRN FM ◆ Steve Malag
Re: Learning ◆ Jim Balter, Sun, 10 Mar 1996, Lines
Looking for Neural–Net based Chess/Checkers ◆ P
Article on Kasparov vs Deep Blue ◆ Theodore M K

**WEBSOM no**

to move to neighboring nodes on the map.

Instructions

Re: Funny Names??? ◆ Teemu Peltonen, 27 Nov 1995, Lines:
Re: Programming a computer to play ◆ Tord Kallqvist Rom
Re: Programming a computer to play ◆ PhiRatE, 17 Oct 1996
Re: Chess, AI and drosophila ◆ Christer Ericson, Sat, 29 Jun
Re: Great Shareware ◆ John P DeMastri, Wed, 21 Aug 1996,
Re: Computer scores historic chess win over Kasparov ◆ Yar
Re: Computer scores historic chess win over Kasparov ◆ Kar
Re: Paul Desmond ◆ Todd Hildreth, Fri, 10 Nov 1995, Lines: 3
Re: 1st person imperatives ◆ Max Crittenden, Thu, 27 Jul 199
Re: chess game theory ◆ Cris Moore, 22 Nov 1996, Lines: 11.
Re: Sanitary Napkins ◆ Norbert C Tagge, 27 Mar 1996, Lines
Re: Chess Programming ◆ epinnel@ibm.net, 25 Jun 1995, Lin

Figure 25: Content-addressable search from a 1,124,134-document WEBSOM

words are used. To this end one must first identify the users' needs, e.g., whether background information to a given article is wanted, or whether the method is used as a kind of keyword-directed search engine.

Finally it ought to be emphasized that the order that ensues in the WEBSOM may not represent any taxonomy of the articles and does not serve as a basis for any automatic indexing of the documents; the similarity relationships better serve "finding" than "searching for" relevant information.

## References

[1] Salton G, McGill MJ. *Introduction to modern information retrieval.* McGraw-Hill, New York, 1983

[2] Deerwester S, Dumais S, Furnas G, Landauer K. Indexing by latent semantic analysis. *J Am Soc Inform Sci*, 1990; 41:391-407

[3] Kaski S. Data exploration using self-organizing maps. Acta Polytechnica Scandinavica, Mathematics, Computing and Management in Engineering Series No 82, 1997. Dr Tech Thesis, Helsinki University of Technology, Finland

[4] Kaski S. Dimensionality reduction by random mapping. In: Proc of IJCNN'98, Int Joint Conf on Neural Networks. IEEE Press, Piscataway, NJ, 1998, pp 413-418

[5] Ritter H, Kohonen T. Self-organizing semantic maps. *Biol Cyb*, 1989; 61:241-254