Computing Science and Statistics 29:281-290, 1998
(Scott, D. W., ed.), Interface Foundation of NorthAmerica, Inc.: Fairfax Station, VA

Statistical Aspects of the WEBSOM System in Organizing Document Collections

S. Kaski, K. Lagus, T. Honkela, and T. Kohonen
Neural Networks Research Centre
Helsinki University of Technology
P.O.Box 2200, FIN-02015 HUT, Finland

Abstract

WEBSOM is a novel method for organizing document collections onto map displays to enhance the interactive browsing and retrieval of the documents. The map is organized automatically according to the contents of the full-text documents by the Self-Organizing Map algorithm. The map display provides a visual overview of the whole document collection. The overview, the map display, aids in the exploration since similar documents are located close to each other. In this paper we describe the WEBSOM system in a statistically oriented fashion and discuss its relations to other methods. Particular emphasis is put on how effective the methods are in treating large document collections. The two-phase architecture of the WEBSOM system makes it possible to build contextual information about the relations of words off-line into a word category representation, which can then be utilized rapidly on-line, when the documents are being encoded. The construction of large map displays from the encoded document representations is a computationally intensive operation when done in a straightforward manner. There exist, however, several effective computational shortcuts.

1 Introduction

One motivation in the construction of a new approach to information retrieval has been that the present systems seem to be of help only in rather narrowly defined tasks. In fact, the term information retrieval is often used to refer only to the process of finding a set of documents relevant to a given textual query. There exist, however, also other less precisely defined information retrieval tasks in which it may not be possible to formulate a satisfactory query. If the topic area is not yet familiar, like when studying a new topic, the relevant terms are not known and a satisfactory query cannot be formulated. It is also quite common that after finding a set of interesting documents the user wishes to quickly *browse* related topic areas. In most search engines this is not possible.

To clarify the differences between different ways of retrieving information we divide the tasks into *exploration, search*, and *filtering*. The narrow definition of information retrieval includes only the search tasks, although searching for specific information will probably most often involve both browsing and searching steps. Exploration, which refers to browsing a collection to find something useful or interesting, is needed especially when it is difficult to construct a satisfactory query. Information filtering refers to the extraction of interesting information from a stream of information.

WEBSOM (Honkela et al., 1996; Kaski et al., 1996; Kohonen et al., 1996; Lagus et al., 1996) is a new approach to the information retrieval tasks. Although the WEBSOM system aids in all the three tasks its most salient feature is the capability of aiding in the exploration of a document collection.

The WEBSOM system forms a map of the document collection. Close-by areas on the map display will represent documents that have similar contents. A neighborhood of an interesting location on the map is therefore likely to contain other interesting articles. When the map is equipped with a suitable interface it can be used for exploring the document collection. If an appropriate query can be constructed or if an interesting new document is available the map can also be used for searching related documents. The position of the interesting document on the map provides an ideal starting point for exploration. The map can also be utilized in filtering interesting articles from a stream of information. The use of the WEBSOM in all of these information retrieval tasks will be demonstrated in Section 4.

The processing of the documents consists of two stages (Fig. 1). First the textual document is encoded into a numerical vector denoted by $\mathbf{a}_k \in \mathbb{R}^n$ as will be described in Section 3. In the second stage the vector is mapped onto the document map display using a suitable mapping

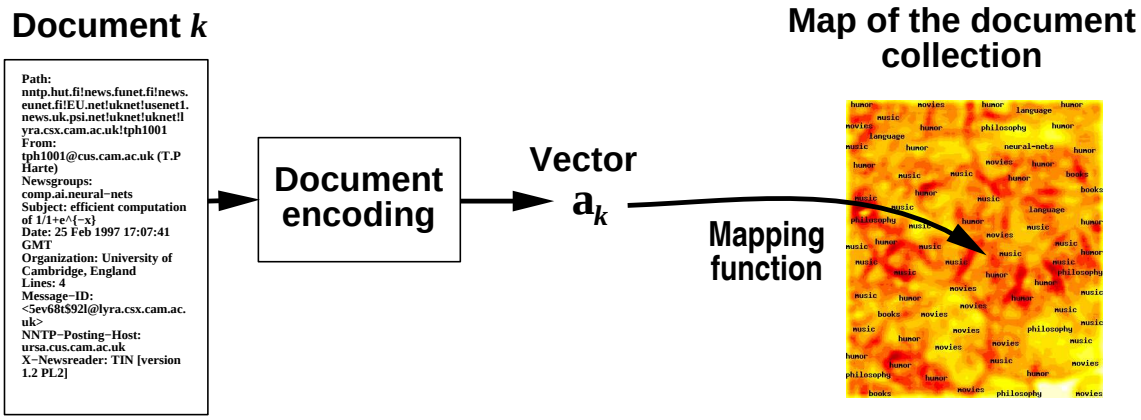

Figure 1: The basic building blocks of the WEBSOM system.

function that has been estimated based on a document collection. Demonstrations of the second stage will be presented in Section 4. The Self-Organizing Map algorithm (Kohonen, 1982; Kohonen, 1995) that plays a key role in both of the stages will be discussed in Section 2.

In this paper we shall concentrate on a statistically oriented description of the WEBSOM methodology; especially the connections to related statistical methods will be discussed. In addition, we consider the computational complexity of the methods since the document collections are very large in several real-world information retrieval applications.

2 The Self-Organizing Map

2.1 The Algorithm

The Self-Organizing Map (SOM) algorithm (Kohonen, 1982; Kohonen, 1995) can be used to form a condensed description, a certain kind of a set of statistics, of a data set consisting of multivariate observations $\mathbf{x}_k \in \mathbb{R}^n$, $k = 1, \ldots, K$. The description can be visualized as a map display on which observations, also new ones, can be mapped. In general, similar observations become mapped near each other.

The SOM consists of an ordered set of *model vectors* $\mathbf{m}_i$, $i = 1, \ldots, M$. The order of the model vectors is determined by a *map lattice* onto which the vectors are attached at regularly spaced positions. An example where the models have been attached onto a hexagonal two-dimensional lattice is shown in Figure 2.

The SOM algorithm creates a regression of the ordered set of model vectors into the data space. The two-dimensional map lattice can intuitively be thought of as an "elastic network" that is being fitted to the input data. Each model vector determines the location of

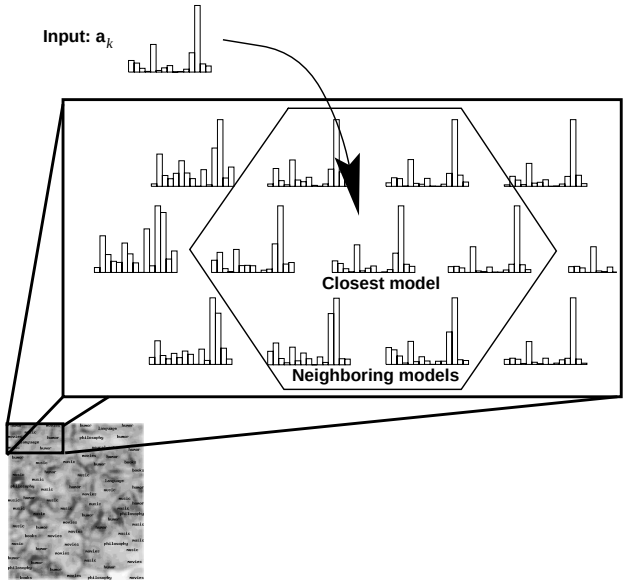

Figure 2: The Self-Organizing Map consists of a set of model vectors, depicted here as histograms where each bin corresponds to one component. The models are attached onto a *map lattice* at regularly spaced intervals. An example map display of a document collection is shown in the lower left corner, and the enlarged portion shows the map lattice that underlies the top left corner of the map display. The arrow illustrates the mapping of the document vector $\mathbf{a}_k$, and the hexagon denotes the neighborhood of the closest model.

the corresponding node of the network in the input data space. The network tries to follow the distribution of the input data while remaining organized in the sense that model vectors at neighboring positions on the map lattice are close to each other.

In the stochastic version of the SOM algorithm the model vectors are estimated in an iterative process. At each step of the iteration one data vector $\mathbf{x}$, i.e., a multidimensional observation, is chosen with replacement. The closest model vector, defined by the expression

$$\|\mathbf{x} - \mathbf{m}_c\| \leq \|\mathbf{x} - \mathbf{m}_i\| \ \ \forall i \ , \tag{1}$$

is then searched for. Here $c = c(\mathbf{x})$ indexes the closest model vector. The norm is usually Euclidean.

After the closest model has been found it is modified towards the current input, so that the model represents the input better. When this process is iterated, each time choosing randomly a new observation, the different model vectors gradually begin to specialize in representing different types of observations.

The algorithm described so far is essentially a stochastic version of the well-known K-means clustering approach. What distinguishes the SOM from the K-means is that in addition to the closest model *the neighboring models are updated as well.* The neighborhoods are defined in terms of closeness *on the map lattice* as demonstrated in Figure 2. The update of the neighboring models can be expressed in a general form by introducing a *neighborhood kernel* h_{ci} which is a decreasing function of the distance between the lattice points i and c. The kernel determines how much the model vector indexed with i will be modified when the model vector c is closest to the current observation $\mathbf{x}$. Using this notation the modification at step t of the stochastic iteration can be expressed as

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + h_{c(\mathbf{x}),i}(t)\,(\mathbf{x}(t) - \mathbf{m}_i(t)) \ . \tag{2}$$

When the iterative process defined by the equations (1) and (2) is continued, model vectors that are attached to neighboring positions on the map lattice gradually become similar. The process forms a neighborhood preserving mapping of the lattice into the input space.

Although the algorithm looks deceptively simple it has turned out to be very difficult to analyze mathematically in the general case (cf. Kohonen, 1991; Kohonen, 1995). In practical applications the data set is always finite, however, and in that case there exists a cost function that the SOM algorithm tries to minimize with stochastic approximation (Robbins and Monro, 1951). The cost function *for a fixed neighborhood function* h' is (Ritter and Schulten, 1988; Kohonen, 1991)

$$\frac{1}{2}\sum_k \sum_i h'_{c(\mathbf{x}_k),i}\,\|\mathbf{x}_k - \mathbf{m}_i\|^2 \ . \tag{3}$$

The update defined by equations (1) and (2) corresponds to one step towards the direction of a noisy negative gradient of the cost function. The noisy gradient is here the gradient that has been computed based on one randomly chosen input sample.

During the stochastic approximation the length of the step, $\alpha(t)$, decreases gradually towards zero. In equation (2) the length is incorporated into the neighborhood function $h(t) = \alpha(t)h'(t)$.

To guarantee global organization of the map the "width" of h will be made to decrease as well. It is advisable to start the learning with a wide neighborhood function which narrows rapidly. After that the final values of the model vectors can be estimated using a narrower neighborhood function.

2.2 Properties Useful for Data Exploration

The SOM can be used as a tool for creating an *overview* of a data set. The data set can be visualized on the SOM display by projecting each sample $\mathbf{x}_k$ onto the location of the map lattice that corresponds to the closest model vector. Nearby locations on the map display represent similar observation vectors, which makes the overview suitable for data exploration.

The SOM display can additionally be used as an *ordered groundwork* for visualizing, for instance, the *clustering tendency* in the data set. The distances of the neighboring model vectors can be visualized on the map display as gray levels (Ultsch, 1993). Light shades mean that in the corresponding area the observation vectors are relatively more similar than in the dark areas. The shades on such a display can be interpreted to reflect the clustering tendency since the model vectors of the SOM are more dense in the dense areas of the data space.

In this concise introduction we have discussed only the features of the SOM that are most relevant to the WEBSOM system. The use of the SOMs for data exploration is discussed in more detail elsewhere (Kaski, 1997b; Kohonen, 1995).

2.3 Relations to Other Methods

As was already mentioned, the SOM algorithm is closely related to the classical K-means clustering algorithms (Forgy, 1965; Lloyd, 1982; Linde et al., 1980; MacQueen, 1967). A SOM without the neighborhood function (i.e.,

when $h_{ci} = \delta_{ci}$ where δ denotes the Kronecker delta), is equivalent to K-means.

Principal curves (Hastie and Stuetzle, 1989) are defined to be smooth curves on which every point is the average of those data vectors that project onto it. In the SOM, each model finds the mean of the data that projects onto it or to the neighboring nodes, weighted by the neighborhood kernel. In this sense the SOMs can be thought to be analogous to the principal curves.

The multidimensional scaling (MDS) algorithms (cf., e.g., Kruskal and Wish, 1978; Sammon, Jr., 1969) can be used for producing a nonlinear projection from the original data space to a lower-dimensional space. The mapping tries to preserve the *distances*, or some functions of the distances, between the original observation vectors as faithfully as possible. The SOM, in contrast, does not try to preserve the distances but the neighborhood relations on the map lattice; nearby locations on the lattice represent similar data items. In this sense the SOM is topology preserving instead of distance preserving.

Another difference between the SOM and the MDS is that the SOM estimates an explicit mapping function based on a data set. The function can be later used for mapping new data items. The traditional MDS methods find the mapping of a specific data set in an iterative process which must be performed again every time when new data samples become available.

2.4 Computational Complexity

The computational complexity of mapping one observation on a SOM is $\mathcal{O}(M)$, where M denotes the number of the model vectors. The construction of the mapping requires more computations. Finding the best representation according to (1) involves $\mathcal{O}(M)$ steps, and a sufficient number of iterations must be performed to warrant good estimates for the model vectors. One simple rule of thumb states that a suitable constant number of data items should be presented for the estimation of each model vector; the model vectors are essentially means of subsets of the data. In total the computational complexity of constructing the mapping function is then $\mathcal{O}(M^2)$. Note that in the important special case where the ratio between the "width" of the neighborhood kernel and the map size is fixed the computational complexity is only $\mathcal{O}(M)$.

There exists a tradeoff between the computing time and the size of the map M which determines the resolution of the mapping. In case a large map size would be needed, like for example in WEBSOM, computational shortcuts can be used for reducing the time spent for constructing the map (Kohonen, 1996). For example, it is possible to compute first a smaller SOM and then to estimate a larger one based on the model vectors of the smaller SOM. Furthermore, the search for the closest model vector (equation 1) can be restricted to the neighborhood of the model that was closest to the data vector when it was chosen previously. Such a restricted search is reasonably accurate if the map has not changed much since the previous presentation of the vector. This is the case especially towards the end of the learning process.

3 Document Encoding

The goal of document encoding, which could be called *feature extraction* or *variable selection* as well, is to choose a set of variables that would convey the contents, the "meaning" of the documents.

Several different approaches for the encoding of documents have been suggested in the information retrieval literature. All the approaches that aim at a representation of textual documents as numerical vectors could in principle be used as the document encoding stage of the WEBSOM. In the following we shall describe a particularly fast approach (Honkela et al., 1996) that we have used so far, and discuss its relations to other alternatives.

3.1 Document Encoding by Clustering the Words

Introduction. The *vector space model* (Salton and McGill, 1983) is a traditional, effective method for encoding text documents for information retrieval. The document indexed by k is represented by a vector $\mathbf{a}_k \in \mathbb{R}^n$. Each component of $\mathbf{a}_k$ corresponds to the frequency of occurrence of one word in the document or to some function of the frequency of occurrence. The components may additionally be weighted by some measures of the importance of the words.

The vector $\mathbf{a}_k$ can be visualized as a *word histogram* where each bin corresponds to the frequency of occurrence of one word (see Figure 2).

Straightforward application of the vector space model to large document collections is problematic since the dimensionality of the document vectors may be very large. There have been tens of thousands of words in some of the collections we have used, even after the rarest words were discarded.

Another problem in the vector space model is that it does not take into account any relations between the meanings of the words. Synonyms, for example, are treated as quite unrelated dimensions.

One Solution: Clustering of the Words Off-Line. One solution for reducing the dimensionality of the word histograms is to cluster the words. If words having similar meaning are clustered together the accuracy of the encoding may not even suffer in the dimensionality reduction.

It is usually not feasible to try to incorporate direct information about the meanings of the words in an automatic system. There is, however, plenty of textual *contextual information* available in the document collections. A text corpus can be used for gathering statistics about the contexts in which each word occurs. Assuming that words occurring in similar contexts are related enough to warrant grouping them together, we can cluster the words based on suitable contextual statistics.

The contextual statistics relating to word i will be collected into the vector $\mathbf{x}_i$; the computation of the statistics will be discussed in more detail in Section 3.2. The vectors $\mathbf{x}_i$ can then be clustered with the SOM to form a self-organizing semantic map (Ritter and Kohonen, 1989). In the original article the method was motivated from a somewhat different viewpoint; here we shall concentrate on a statistically oriented computational description. The resulting clusters will be called *word categories*, i.e., categories comprising of words that tend to occur in similar contexts.

After the word categories have been constructed off-line utilizing a suitable text corpus they can be used on-line for encoding documents. The word histogram of the vector space model will be replaced by a *word category histogram*. We shall next discuss the steps of the document encoding in more detail.

3.2 The Contextual Statistics

The contextual statistics that are used in clustering the words are based on the co-occurrences of the words in consecutive positions in the word stream. The principle is introduced below using a particularly simple example.

Denote by n_{il} the number of times the word indexed by l has occurred in the text corpus immediately before the word indexed by i. Denote by N_i the total number of times the word i has occurred and by n the total number of words. The contextual statistics that are used for encoding the word i are then

$$x_{il} = n_{il}/N_i \ , \quad l = 1, \ldots , n.$$

The lth component of the vector $\mathbf{x}_i$ containing the contextual statistics can be interpreted as an estimate of the probability that the word l occurs immediately before the word i.

The vector $\mathbf{x}_i$ can be represented equivalently with the following vector sum:

$$\mathbf{x}_i = \frac{1}{N_i} \sum_l n_{il} \mathbf{e}_l \ .$$

Here $\mathbf{e}_l$ denotes the unit vector that has one in the lth component and zeros elsewhere. In practice the dimensionality of $\mathbf{x}_i$ may be very high; it is equal to the dimensionality of the word histograms. Therefore lower-dimensional *random vectors* are used instead of the $\mathbf{e}_l$ to reduce the computational load (Ritter and Kohonen, 1989).

The above discussion involved only the words that have occurred immediately before the word i. It is, however, straightforward to include also information about the words that have followed the word i, and it is possible to compute other kinds of contextual statistics as well. In the simulations reported in this article we have used a context consisting of one word on each side of the word that is being encoded.

Some examples of the word categories that are obtained by clustering the vectors $\mathbf{x}_i$ containing the contextual statistics are shown in Figure 3.

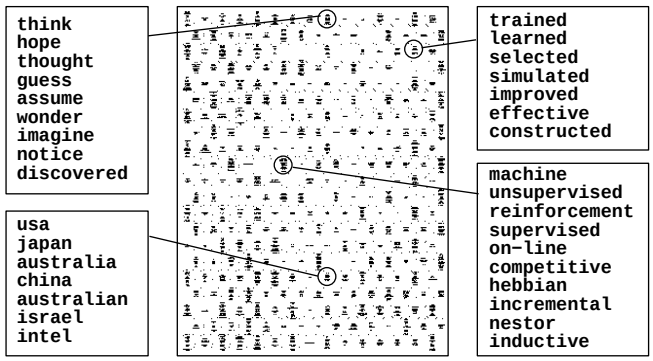

Figure 3: Sample word categories on a self-organized semantic map. Each inset shows the words that have been mapped into one word category. The text corpus consisted of articles from the Usenet newsgroup `comp.ai.neural-nets`.

3.3 Formation of the Word Category Histograms

One clear advantage of reducing the dimensionality of the word histograms by clustering is that the encoding of a document can be carried out very rapidly. The words can be clustered into word categories off-line. During the actual encoding we then only need to find the category of each word in the document and to update the

corresponding bin in the word category histogram. The correct category can be found rapidly with hash addressing and a table lookup.

The histograms may be additionally postprocessed to enhance invariance to small changes in the documents. Since the categories on the SOM are *ordered*, i.e., closeby models on the map lattice are similar, similar words will mostly occur in either the same category or in closeby categories. Invariance to the choices of the words made by the authors of the documents can thus be enhanced by *smoothing* the histogram *spatially* on the map lattice. There exists, however, some evidence that at least in certain situations the smoothing has only a small effect (cf. Kaski, 1997a).

3.4 Other Possible Encoding Methods

Latent Semantic Indexing (LSI) (Berry et al., 1995; Deerwester et al., 1990) is one possible alternative document encoding method. One way of interpreting the LSI is that it represents the kth document by the vector

$$\mathbf{a}'_k = \sum_i a_{ki}\mathbf{x}'_i \ , \tag{4}$$

where a_{ki} denotes the number of times the word i occurs in the kth document. The $\mathbf{x}'_i$ is the code that the LSI forms for the ith word by investigating the cooccurrences of the words within the documents. The term by document matrix, the matrix where each column is the word histogram corresponding to one document, is decomposed into a set of factors (eigenvectors) using the singular-value decomposition (SVD). The factors that have the least influence on the matrix are then discarded. The motivation behind omitting the smallest factors is that they are most likely to consist of noise. The vectors $\mathbf{x}'_i$ can then be formed by using the remaining factors.

LSI differs from the approach we have used in two respects. In LSI the context from which the codes of the words are estimated consists of the whole documents whereas we have used shorter contexts. Another difference is that we use clustering to reduce the dimensionality of the codes whereas LSI uses the linear SVD method.

The MatchPlus method of HNC (Gallant et al., 1992) encodes the documents as sums of the vectorial representations of the words quite like the LSI (equation 4). However, in the MatchPlus system the vectors $\mathbf{x}'_i$ are constructed differently.

3.5 On-Line Computational Complexity of the Document Encoding

Denote by N the average number of words in a document, and let n denote the dimensionality of the vectors representing the documents. Using these notations the computational complexity of the document encoding is $\mathcal{O}(N) + \mathcal{O}(n)$ in the clustering approach, and in LSI it is $\mathcal{O}(N)\mathcal{O}(n)$. The former expression includes a smoothing of the word category histogram with a local smoothing kernel. Both of the computational complexities take into account only the on-line parts of the encoding. The parts that can be carried out off-line, i.e., the clustering of the words and the computation of the SVD, have been neglected. The clustering approach is thus clearly faster on-line.

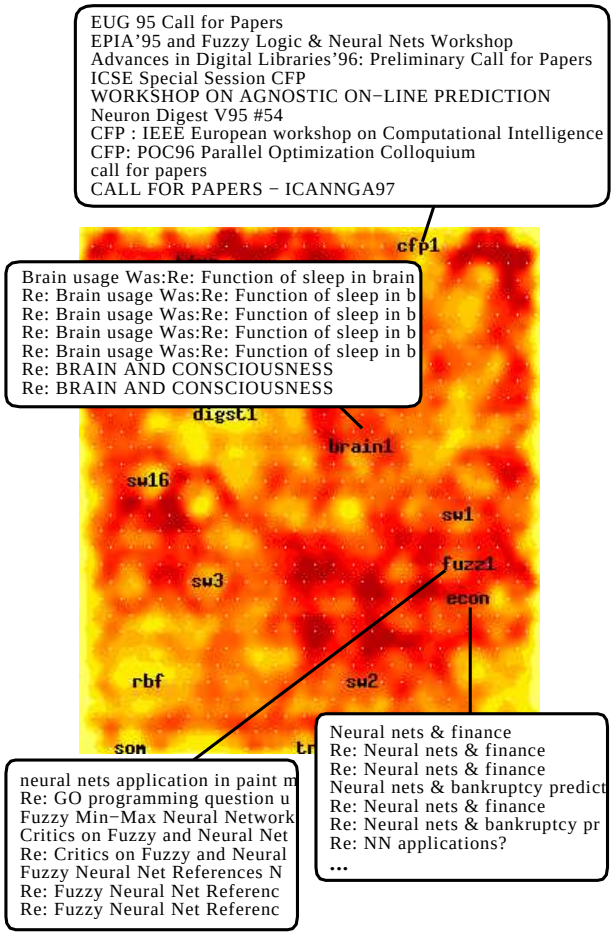

Figure 4: A map of neural network discussions organized with the WEBSOM method. The short, manually inserted labels, "bookmarks", on the map describe the underlying map area (their explanations have been omitted here for brevity). The titles of the documents found in four sample map nodes are shown in the boxes.

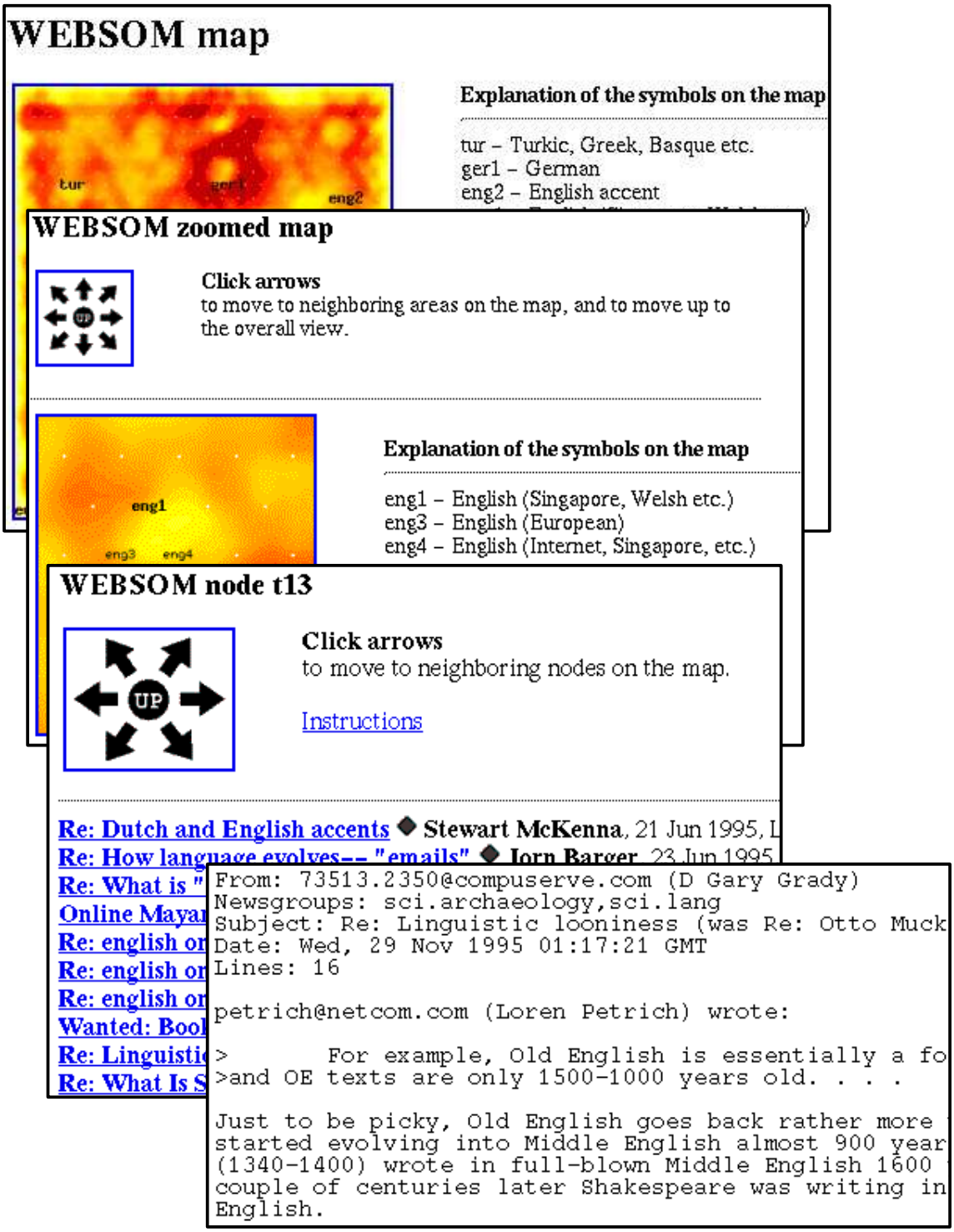

Figure 5: The four levels of the WEBSOM interface. The top level offers an overview of the collection. The user may zoom in on the collection by clicking on the map display, and finally reach the contents of individual documents. Once an interesting document or map node is found, the arrows can be used for moving in the surrounding map area to find similar, potentially interesting documents. In this example the documents come from the Usenet newsgroup `sci.lang`.

4 Document Maps

Once the documents have been encoded, so that similarity of the texts is paralleled with similarity of the respective document vectors, the remaining task is to determine the mapping function sketched in Figure 1. The function is estimated by the SOM algorithm that constructs a map of a document collection. The resulting document map provides the mapping function: for any given document vector the closest model vector is searched for, and the document is placed on the corresponding point on the map lattice. The function can be used for mapping previously unseen documents as well, as long as the new documents treat similar topics as the old ones.

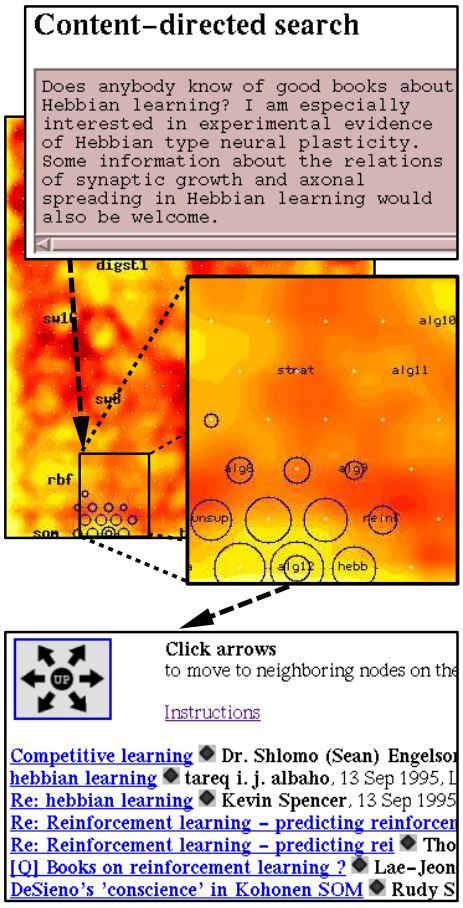

Figure 6: A new document or any written description can be used for finding related documents. The circles on the map display denote the locations that best represent the textual description shown in the topmost display. Titles of some of the articles found in the best matching map location are shown in the display at the bottom.

4.1 Results

We have created document maps of both small, relatively focused topic areas and of large collections involving several quite different topic areas. A map of documents discussing a single rather narrow subject area, neural networks, is shown in Figure 4 (Honkela et al., 1996; Kaski et al., 1996). The document collection consisting of 4600 short colloquial texts from the newsgroup `comp.ai.neural-nets` was organized on a map consisting of 768 model vectors.

The method has also been applied to forming an overview of a large collection of Usenet newsgroup discussions dealing with several very different topics such as artificial intelligence, music, movies, and linguistics. A map consisting of 49,152 model vectors was used for organizing 131,500 documents (Kohonen et al., 1996). In this study it was possible to automatically enhance the separability of different topic areas. Information-theoretic *entropy* of the distribution of each word in the different topic areas, here the different Usenet discussion groups, was used as an indication of the importance of the word. The contribution of each word in the representations of the documents was then weighted with an entropy-based weight.

4.2 How to Use the Maps

The exploration interface of the WEBSOM is shown in Figure 5. Using the interface one may explore the document collection and read the documents.

A query or an interesting document can be mapped to the display quite like any other document. The mapping of new documents is fast and can be carried out in real time. The position of the new document on the map then provides an ideal starting point for exploring related documents; the map functions as a genuine content-addressable memory. Such starting points are especially useful when the user is not yet familiar with the map. A sample search is shown in Figure 6.

Later, when the user has become more familiar with the document map, the map can be utilized as a filter. Any new documents may be mapped onto the map display. When the users are familiar with the map and therefore know which areas of the map are interesting, they can concentrate on the documents that become mapped to those areas (Fig. 7). It is also possible to set "traps" or "mailboxes" on the map: new documents that become mapped on these areas become routed to their destination while the rest will be discarded.

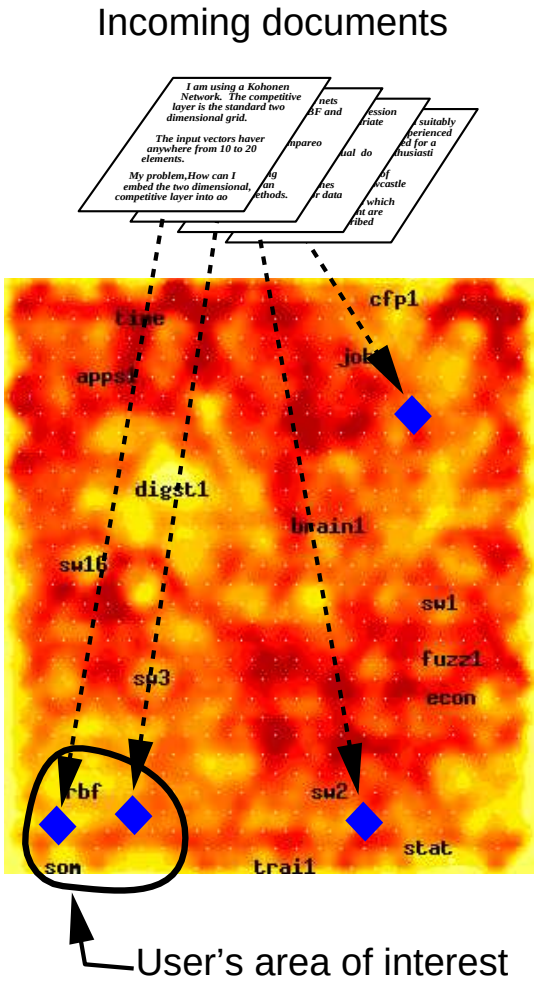

Figure 7: The document map can be used as a filter that notifies the user of interesting documents. Any interesting area on the map can be marked as a trap which keeps the new documents.

5 Conclusions

In this paper we have provided an overview of the WEBSOM method and demonstrated how it can be used in information retrieval from textual document collections. The system consists of two stages, the document encoding stage and the construction of a map of the encoded documents. The current encoding stage could potentially be replaced with alternative schemes but the creation of the map of the document collection with the SOM algorithm is a central part of the system. SOMs of document collections have also been studied by others (Chen et al., 1996; Lin et al., 1991; Merkl, 1993; Scholtes, 1993).

The maps that have been used in the illustrations are available for exploration in the WWW at the address `http://websom.hut.fi/websom/`

References

Berry, M. W., Dumais, S. T., and O'Brien, G. W. (1995). Using linear algebra for intelligent information retrieval. *SIAM Review*, 37:573–595.

Chen, H., Schuffels, C., and Orwig, R. (1996). Internet categorization and search: a self-organizing approach. *Journal of the Visual Communication and Image Representation*, 7:88–102.

Deerwester, S., Dumais, S. T., Furnas, G. W., and Landauer, T. K. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407.

Forgy, E. W. (1965). Cluster analysis of multivariate data: efficiency vs interpretability of classifications. *Biometrics*, 21:768–769.

Gallant, S. I., Caid, W. R., Carleton, J., Hecht-Nielsen, R., Pu Qing, K., and Sudbeck, D. (1992). HNC's MatchPlus system. *ACM SIGIR Forum*, 26(2):34–38.

Hastie, T. and Stuetzle, W. (1989). Principal curves. *Journal of the American Statistical Association*, 84:502–516.

Honkela, T., Kaski, S., Lagus, K., and Kohonen, T. (1996). Newsgroup exploration with WEBSOM method and browsing interface. Technical Report A32, Helsinki University of Technology, Laboratory of Computer and Information Science, Espoo, Finland.

Kaski, S. (1997a). Computationally efficient approximation of a probabilistic model for document representation in the WEBSOM full-text analysis method. *Neural Processing Letters*, 5:139–151.

Kaski, S. (1997b). Data exploration using self-organizing maps. *Acta Polytechnica Scandinavica, Mathematics, Computing and Management in Engineering Series No. 82*. DTech Thesis, Helsinki University of Technology, Finland.

Kaski, S., Honkela, T., Lagus, K., and Kohonen, T. (1996). Creating an order in digital libraries with self-organizing maps. In *Proceedings of WCNN'96, World Congress on Neural Networks*, pp. 814–817. Lawrence Erlbaum and INNS Press, Mahwah, NJ.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69.

Kohonen, T. (1991). Self-organizing maps: optimization approaches. In Kohonen, T., Mäkisara, K., Simula, O., and Kangas, J., editors, *Artificial Neural Networks*, vol. II, pp. 981–990. North-Holland, Amsterdam.

Kohonen, T. (1995). *Self-Organizing Maps*. Springer, Berlin.

Kohonen, T. (1996). The speedy SOM. Technical Report A33, Helsinki University of Technology, Laboratory of Computer and Information Science, Espoo, Finland.

Kohonen, T., Kaski, S., Lagus, K., and Honkela, T. (1996). Very large two-level SOM for the browsing of newsgroups. In von der Malsburg, C., von Seelen, W., Vorbrüggen, J. C., and Sendhoff, B., editors, *Proceedings of ICANN96, International Conference on Artificial Neural Networks*, Lecture Notes in Computer Science, vol. 1112, pp. 269–274. Springer, Berlin.

Kruskal, J. B. and Wish, M. (1978). *Multidimensional Scaling*. Number 07-011 in Sage University Paper series on Quantitative Applications in the Social Sciences. Sage Publications, Newbury Park, CA.

Lagus, K., Honkela, T., Kaski, S., and Kohonen, T. (1996). Self-organizing maps of document collections: a new approach to interactive exploration. In Simoudis, E., Han, J., and Fayyad, U., editors, *Proceedings of KDD-96, Second International Conference on Knowledge Discovery and Data Mining*, pp. 238–243. AAAI Press, Menlo Park, CA.

Lin, X., Soergel, D., and Marchionini, G. (1991). A self-organizing semantic map for information retrieval. In *Proceedings of 14th Annual International ACM/SIGIR Conference on Research & Development in Information Retrieval*, pp. 262–269.

Linde, Y., Buzo, A., and Gray, R. M. (1980). An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28:84–95.

Lloyd, S. P. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28:129–137. (Originally an unpublished memorandum, Bell Laboratories, 1957).

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In Le Cam, L. M. and Neyman, J., editors, *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability. Vol. I: Statistics*, pp. 281–297. University of California Press, Berkeley and Los Angeles, CA.

Merkl, D. (1993). Structuring software for reuse—the case of self-organizing maps. In *Proceedings of IJCNN-93 (Nagoya), International Joint Conference on Neural Networks*, vol. III, pp. 2468–2471, IEEE Service Center, Piscataway, NJ.

Ritter, H. and Kohonen, T. (1989). Self-organizing semantic maps. *Biological Cybernetics*, 61:241–254.

Ritter, H. and Schulten, K. (1988). Kohonen's self-organizing maps: exploring their computational capabilities. In *Proceedings of ICNN'88, IEEE International Conference on Neural Networks*, vol. I, pp. 109–116. IEEE Service Center, Piscataway, NJ.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407.

Salton, G. and McGill, M. J. (1983). *Introduction to modern information retrieval*. McGraw-Hill, New York.

Sammon, Jr., J. W. (1969). A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, 18:401–409.

Scholtes, J. C. (1993). *Neural Networks in Natural Language Processing and Information Retrieval*. PhD thesis, Universiteit van Amsterdam, Amsterdam.

Ultsch, A. (1993). Self-organizing neural networks for visualization and classification. In Opitz, O., Lausen, B., and Klar, R., editors, *Information and Classification*, pp. 307–313. Springer-Verlag, Berlin.