Authors:	Saarikoski Jyri, Laurikkala Jorma, Järvelin Kalervo, Juhola Martti
Name of article:	A study of the use of self-organising maps in information retrieval
Year of publication:	2009
Name of journal:	Journal of Documentation
Volume:	65
Number of issue:	2
Pages:	304-322
ISSN:	0022-0418
Discipline:	Natural sciences / Computer and information sciences
Language:	en
School/Other Unit:	School of Information Sciences

URN: http://urn.fi/urn:nbn:uta-3-740
DOI: http://dx.doi.org/10.1108/00220410910937633

Preprint from: Saarikoski, J, & Laurikkala, J. & Järvelin, K. & Juhola, M (2009). A study on the use of self-organising maps in information retrieval. Journal of Documentation 65 (2): 304-322. Full text at: http://www.emeraldinsight.com/Insight/viewPDF.jsp?contentType=Article&Filename=html/Output/Published/EmeraldFullTextArticle/Pdf/2780650207.pdf

A study on the use of self-organising maps in information retrieval

Jyri Saarikoski[1], Jorma Laurikkala[1], Kalervo Järvelin[2] and Martti Juhola[1]

[1]*Department of Computer Sciences, 33014 University of Tampere, Finland*

[2]*Department of Information Studies, 33014 University of Tampere, Finland*

Keywords *Self-organising maps, neural networks, information retrieval, document grouping*

Purpose *- We studied the applicability of self-organising maps for searching for information in a document collection.* ***Design / methodology / approach*** *– After conventional preprocessing, like transform into vector space, documents from a German document collection were trained for a neural network of Kohonen self-organising map type. Such an unsupervised network forms a document map from which relevant objects can be found according to queries.* ***Findings*** *- Self-organising maps ordered documents to groups from which it was possible to find relevant targets.* ***Research limitations / implications*** *- The number of documents used was moderate due to the limited number of documents associated to test topics. The training of self-organising maps entails rather long running times, which is their practical limitation. In future, our aim will be to build larger networks by*

This research was funded, in part, by the Academy of Finland, Project Nos. 120996, 200844, 202185, 204970 and 206568. The SNOWBALL stemmer by Martin Porter.

compressing document matrices, and develop document searching in them. ***Practical implications*** *- With self-organising maps the distribution of documents can be visualised and relevant documents found in document collections of limited size.* ***Originality / value*** *– This approach can be especially used to group documents and also for information search. So far self-organising maps have rarely been studied for information retrieval. Instead, they have been applied to document grouping tasks.*

1. Introduction

In information retrieval tasks, documents may be represented in a vector form, which can be considered in many ways to execute search or grouping tasks. One possibility is to apply machine learning methods which utilise similarity values or distances between documents. These methods include the traditional nearest neighbour searching as well as more sophisticated methods, such as neural networks. Our research objective was to explore the possibility to retrieve information with Kohonen self-organising maps, which are known to be effective to group objects according to their similarity or dissimilarity. The aim was interesting since they have seldom appeared in information retrieval literature and the articles encountered seem to consider rather organisation of documents, not their retrieval.

Perhaps the most important application of self-organising maps (Kohonen, 1995) connected to electronic documents is WEBSOM by Honkela (1997) and Lagus *et al*. (2004), who used them to organise large document collections. A user of WEBSOM can search for documents from a collection by exploring a self-organising map given as a two-dimensional representation. Words (concepts) describing different areas were inserted in such a map to aid exploration. Additionally, colours were used to emphasize the similarity of documents in adjacent map areas. WEBSOM was used to explore the map representation and supported access of browsing type. However, it was not used for an actual information retrieval evaluation as usually understood - applying a test collection of topics and relevant documents. They treated very large collections, even as large as approximately 6 840 000 English patent abstracts (Kohonen *et al*., 2000) for which a map of over one million nodes was built. Such a huge number inevitably required a compres-

sion of document vectors, which was performed by a random matrix projection (Kaski, 1998) to reduce a vector length of over 43 000 down to 500 words. Despite this, the computation took seven weeks.

There are only a few applications of self-organising maps in information retrieval. As early as in 1991 Lin *et al.* introduced an information retrieval system, which utilised self-organising maps for placing 150 documents on a map for browsing. Later it was extended as a general information representation tool (Lin, 1997). Proper nouns and other words were used to form two maps for retrieval from a Spanish collection of 454 042 documents (Fernández *et al.*, 2004). Two maps were used for the classifications of both words and documents (Lee and Yang, 1999). Moreover, Chowdhury and Saha (2005) classified 400, 500 and 600 sports articles, while Guerrero-Bote *et al.* (2002) 202 documents. Moya-Anegón *et al.* (2006) clustered scientific documents on the basis of self-organising maps.

Altogether, the preceding articles can be chiefly seen to consider document grouping. The reports found that were slightly closer to information retrieval were the work of Fernández *et al.* (2004) and that of Lagus (2002) in which a collection of 1460 documents was explored by searches based on self-organising maps. Nevertheless, the average length of its documents was short, merely 115 words, but queries were relatively long, even a half of document lengths. Thus, its approach did not follow an ordinary information retrieval situation. In spite of the shortage of actually comparable studies, the former articles encouraged us that self-organising maps would be promising for information retrieval. On the other hand, they seemed to be a fairly unexplored area as to information retrieval.

The rest of this paper is arranged in the following way. Section 2 presents the test data used in this study. Section 3 presents the creation of self-organising maps in the current context. Sections 4 and 5 describe results obtained. Section 6 discusses the outcomes and compares them to results computed with two clustering techniques. Section 7 concludes the research.

2. Test data and its preprocessing

We applied a German document collection including 294 809 news articles originated from CLEF 2003 (Airio, 2006) from 1994 and 1995. The articles were published, among others, in Frankfurter Allgemeine and Der Spiegel. Altogether, 60 available test topics were associated to the collection. Each topic had a pool of relevant documents. Both relevant and non-relevant documents to the tests topics were incorporated into our tests. For the ease of processing, a subset of 1160 documents was randomly chosen as follows. At first, 20 topics were randomly taken from the collection. Then all 580 known relevant documents associated to these topics were taken. Each topic included 6-87 relevant documents. Thus on average 29 documents were obtained for each topic. Thereafter, 580 non-relevant documents, not related to any of the 20 topics, were randomly drawn from the collection.

In the context of the present research, 'non-relevance' means that non-relevant documents are non-topically related, i.e. they have not been found as relevant in any earlier tests or research with the current data for the 20 topics selected randomly. Of course, it was not possible to study all the large majority of such 'non-relevant' documents one by one to verify their non-relevance. It is really vital for self-organising maps, or any machine learning method, that there is such a non-relevant class of documents for the learn-

ing purpose of the method. Otherwise, the network could not be able to separate topically related documents from the non-topically related ones.

Both documents and test topics were of XML form. The following snippet exemplifies an XML topic representation.

```
<top>
<num> C171 </num>
<DE-title> Eishockeyfinale in Lillehammer </DE-title>
<DE-desc> Welche Mannschaften spielten im Eishockeyfinale der Olympischen Spiele
von Lillehammer 1994?
</DE-desc>
<DE-narr> Relevante Dokumente berichten darüber, welche Teams im Eishockeyfinale
der Olympischen Spiele von Lillehammer 1994 spielten. Dokumente, die darüber
informieren, welches Team den ersten und zweiten Platz der Veranstaltung
belegte, ohne speziell das Finale zu erwähnen, sind ebenfalls relevant.
</DE-narr></top>
```

Our test queries were not based on the whole topics. The text between <DE-title> and the tags </DE-desc> was the origin of each query. The <DE-narr> part was not applied since it could, in principle, include even “disinformation”, such that is explicitly expressed to be adverse to the topic.

To implement the designed search engine the following subtasks had to be solved. The SNOWBALL German stemmer was run to identify word stems, e.g. from ‘Reisimporte’ to ‘reisimport’, ‘Olympische’ to ‘olymp’ and ‘Antike’ to ‘antik’. A list of 1320 German stopwords was used from prepositions, pronouns, adverbs etc., which are typically uninflecting words. Their occurrences were removed from the documents. After stemming, short parts (smaller than two letters) of words were also deleted. Word frequencies of each document were then calculated for remaining word stems.

3. Creation of self-organising maps

After the initial processing, we continued to construct suitable self-organising maps. First we experimented with a program called MATLAB SOM Toolbox (Vesanto *et* al., 2000).

However, the Matlab environment was inappropriate since our matrices were too large to be run with it. Therefore, we chose another program, SOM_PAK (Kohonen *et* al., 1996), which has been written in C and which allows larger data quantities and is far faster compared to MATLAB. It supports the basic operations for a self-organising map like initialisation, learning and evaluation.

Our objective was to design a fairly straightforward search engine prototype in order to form two-dimensional search networks on the basis of self-organising maps as follows. Firstly, text documents are input to the system in the XML form. Secondly, a self-organising map is built after the preceding preprocessing subtasks. Thirdly, a best fit match (node) is searched for from the map and documents contained by such a node and by nodes in the close neighbourhood are retrieved. This means that the best fit node is searched for a given query from the map and the documents of the best fit node and possibly those of its close neighbourhood are produced as the outcome for the query. Lastly, topics could be marked as words on a map. A user could browse a collection included in the map and also search by words.

After preprocessing of the data chosen, the frequency calculation of the remaining words was accomplished. The following procedure was used to remove very frequent and rare words:

1. Frequency information was computed for all words of the document set in how many documents each word occurred.
2. The words were sorted to the list of descending order along with their frequencies.
3. An appropriate quantity of words, e.g. 1000, was selected from the centre of the list.

Our aim was to exclude such words that occur in all or most documents or only in few documents.

Next, document vectors were created. Note that document lengths naturally varied. We employed document vectors from 500 to 5000 words, which were shorter than the original documents. Main results to be described were obtained with the vectors of 1000 words, but according to our preliminary experiments results did not essentially depend on this choice.

The document vectors were encoded with binary, frequency and *tf.idf* weights (Baeza-Yates and Ribeiro-Neto, 1999), but we shall only show main results for the last one since these were slightly better than those of the others. To compute *tf.idf* weights, the frequencies of words (terms) were calculated according to

$$tf_{ik} = \frac{freq_{ik}}{\max_j \{freq_{ij}\}}$$

where $freq_{ik}$ is the number of occurrences of word k in document D_i and $freq_{ij}$ is for all words of D_i. The whole document collection is dealt in this way. The inverse document frequency is obtained by

$$idf_k = \log \frac{N}{n_k}$$

where N is the number of all documents in the collection and n_k the number of documents containing the word k. These formulas give a *tf.idf* value for word k in document D_i

$$a_{ik} = tf_{ik} \cdot idf_k .$$

The document vectors were then used under SOM_PAK for learning of several different self-organising maps depending on their system parameters: the lengths of document vectors, number of nodes, initialisation of node values, neighbourhood computation type and number of learning iterations. Thereafter document locations on a map were computed with SOM_PAK.

4. Queries and runs

To assess the self-organising maps constructed, we ran queries to see how and where relevant documents were distributed in the maps. The queries were constructed like the document vectors previously. The queries were formed automatically on the basis of the topics as described above. In other words, a query vector was prepared from the tags <DE-title> and <DE-desc> of each topic. Interrogative words were eliminated, other words were stemmed, stopwords were excluded, word frequencies calculated, and finally binary query vectors formed from the words of the document set. If a word appears in a query, its value is 1, otherwise 0. Let k denote a word and j denote query Q_j with vector component

$$q_{jk} = \begin{cases} 1, \text{ if } freq_{jk} > 0 \\ 0, \text{ if } freq_{jk} = 0 \end{cases}$$

in which $freq_{jk}$ is equal to the number of the occurrences of word k in query Q_j.

After building the query vectors, the best matches were searched for from each map computed. All nodes of a map were explored and the product of the weight values corresponding to the words of a query was computed for each node. The greatest product yielded the best match.

In detail, the best matched node was computed as follows. To compute the best matched node of a binary query vector, the query vector and model vectors of the self-organising map are compared. There is a model vector for every node of the map. The dimension of a model vector is equal to that of the document vectors. Model vectors are initialised either randomly or with good estimates. In the learning phase of a self-organising map model vectors are compared to the learning data and their component values are appropriately changed during learning. Each node includes model vector M_p which has the same number of components as the input data, query Q_j.

A product is computed for query Q_j and every node M_p as

$$Prod\ (Q_j, M_p) = (1 + m_{p1})^{q_{j1}} (1 + m_{p2})^{q_{j2}} \times (1 + m_{pt-1})^{q_{jt-1}} (1 + m_{pt})^{q_{jt}}$$

where $q_{jk,}$ for k=1,…,t (all words of the document set), is the kth component of the vector of query Q_j and m_{pk} is the component of the model vector of M_p. The node of the greatest product will be the best matched node.

Random or linear initialisation was used for model vectors. In the former way, model vector components are set to random values uniformly distributed. In the latter, model vectors are initialised along with a two-dimensional subspace spanned by the two principal eigenvectors of the learning data vectors. Learning, in other words changing model vector components after the initialisation, followed the general principle of self-organising maps to modify model vectors so that a group of nodes close to each other gradually begins to represent some type of input vectors (document vectors) and finally groups or areas of nodes on a map correspond to certain document vector types, i.e. documents that are somewhat similar to each other. An individual learning iteration was as follows:

1. An input vector was selected randomly.
2. It was compared to the model vectors of a map using Euclidean distance.
3. The best matching node (model vector) was taken.
4. The components of the taken node (model vector) and its closest neighbourhood nodes were modified toward the input vector given.

We applied bubble and Gaussian neighbourhood weighting types (Kohonen *et al*., 1996). In the former, the closest nodes, next closest nodes etc. of the best matching node (Fig. 1) are taken and changes of their model vectors are multiplied by weights depending on their closeness to the best matching node. Straightforwardly a step function was used here: the weight is equal to 1 if a node is within the bubble, otherwise 0. In the latter type, weighting is given by a normal (Gaussian) distribution centred in the best matching node.

Our goal was to pursue a situation on a map that most documents relevant for a topic would be fairly closely located around some node. To evaluate typical performance of self-organising maps we computed five different versions for each map type (number of nodes etc.) test. Their median was calculated on the basis of quantisation errors computed between the weight vectors or model vectors of the best fit node and the input (document) vectors to represent average performance. Self-organising maps were built using the following parameter settings: the lengths of document vectors equal to 500, 1000, 2000, 3000, 4000 and 5000 unique words, maps of 9×9, 11×11, 13×13, 15×15 and 17×17 nodes, either random or linear initialisation, either bubble or Gaussian neighbourhood computation type and ratio of the learning iteration numbers of ordering phase and tuning phase 1/5, 2/10, 3/15 and 4/20. In the ratio 1/5 the number 1 corresponds to the situation where all documents are learnt once in the ordering phase of the network construction

and 5 that fine-tuning epochs are made five times for the whole document set. All alternatives are equal to the ratio 1/5 so that the total number of learning iterations could be tested without other system parameter changes. To evaluate the similarity of vectors Euclidean distances were computed between them.

5. Results

Results were estimated as conventional recall and precision values (Belew, 2000). The precision ($Precision_0$) and recall ($Recall_0$) of each best matching node and the number of documents ($Documents_0$) in such a node were computed as the mean of 20 queries. Second, the values ($Precision_1$, $Recall_1$, and $Documents_1$) were computed by also taking the closest four neighbours of a best matching node into account. In the following, we consider the parameters of self-organising maps one by one in order to find proper settings for them. To investigate the most suitable parameter settings we applied the basic selections: binary weights, document vector length of 1000 words, map size of 11×11, random initialisation, bubble neighbourhood computation and learning iteration ratio 2/10. In Tables 1-6, each of these six parameters is varied whereas the other five are fixed according to the afore-mentioned basic selections.

5.1 Weights and lengths of document vectors

In Table 1 there are precision and recall results for binary, frequency and *tf.idf* weights. As mentioned above, the *tf.idf* weights were selected since they expectedly yielded the best results from the three alternatives. Subscript 0 for precision and recall values corresponds to each best matching node and subscript 1 also consists of its closest four

neighbours used in all subsequent tables. Thus, our choice for later main tests after the current tests of parameter settings will be *tf.idf* weights which gave considerably better results than those of the binary and frequency weights.

The document vector lengths of 500, 1000, 2000, 3000, 4000 and 5000 were used in Table 2. Comparing the average precision and recall values obtained pairwise for these settings, the selection of 1000 words is better than the others except that of 2000 words, which yielded virtually similar results. Thus, we selected the shorter document vector length of 1000 words for our tests, because the shorter length means less computation.

5.2 Size and initialisation of self-organising maps

The size of self-organising maps was varied in Table 3 for five different alternatives. Considering the averages of both two precision and two recall values on the basis of Table 3, increasing the size is worthwhile. Therefore, we shall use 17×17 nodes in our later main tests. Correspondingly, the random initialisations produced better results compared with those of the linear initialisations in Table 4. This supported the use of random initialisations.

5.3 Neighbourhood type and learning iteration ratio

The neighbourhood computation was executed by applying the bubble and Gaussian neighbourhoods. The results are presented in Table 5. The average of the precisions and recalls of the former were better. Consequently, it was employed. Ultimately, the learning iteration ratios of 1/5, 2/10, 3/15 and 4/20 (Table 6) were tested. An increase of the iteration numbers was productive. We shall use 3/15 in the main tests.

5.4 Main tests

We continued after the preceding selecting parameter setting: *tf.idf* weights, document vector length 1000, map of 17×17 nodes, random initialisations, neighbourhood type bubble and learning iteration ratio 3/15.

The following figures show the map of 17×17 nodes. Black balls represent nodes and lines the links between the nodes. Colours from dark (red) to light (white) correspond to distances between the nodes along with the flanked scale bar. The dark areas denote nearness, and the light areas represent cluster borders and great distances between the nodes. For instance, see Fig. 2.

The following four figures are shown similarly to Fig. 2 extended with the numbers of relevant documents of a topic in nodes. The size of black boxes depicts the numbers of documents: the larger a node, the more documents. Figs. 3-6 show the locations of documents for four topics. Fig. 3 shows the Topic 2: 'Ölunfälle und Vögel' (oil accidents and birds). Fig. 4 shows Topic 6: 'Olympische Spiele und Frieden' (Olympic Games and peace). Fig. 5 depicts the documents of Topic 10: 'Eishockeyfinale in Lillehammer' (Ice hockey final in Lillehammer). Fig. 6 yields Topic 19: 'EU und baltische Länder' (the EU and Baltic countries). Figs. 4 and 5 were chosen since both consider the sports and Olympic Games to see whether they are close to each other. The other two topics, which do not represent sports, were taken to see whether they are further away from the sports nodes. Indeed, the sports topics are near each other, but especially the topic on the EU and Baltic countries are apart from the preceding two.

Since the retrieval results clearly depended on a topic, detailed precision and recall values associated to the topic numbers are shown in Table 7. The results of the closest two neighbourhoods according to the linked distance (Fig. 1) are presented. The results indicate how the documents of some topics are well found and those of the other are poorly or not at all found. To achieve a good result, it is not enough that the relevant documents of a topic are concentrated on a consistent area of nodes, but the query of the topic should also hit this area. The queries missed the relevant nodes for some topics. For example, recall and precision were zero for Topic 2 (Table 7), while Fig. 3 shows that the documents grouped well. The documents of a topic were seldom widely dispersed.

5.5 Succeeded and failed queries

The 8 problematic topics included 6, 7, 10, 11, 21, 24, 29 and 40 relevant documents (Table 7). The 12 successful topics included 8, 24, 27, 29, 34, 45 or more documents. Thus, the topics with small numbers of relevant documents were generally difficult to retrieve. This is shown by the results in Table 8, which are the same as in Table 7, but grouped into two classes of the equal size (ten topics in each) including either infrequent or frequent relevant topics, i.e. the average less than 27 relevant documents, or equal to or greater than 27 relevant documents. The average number of relevant documents strongly affected the results: the more relevant documents, the better results. For these two classes precision improved according to the ratio of 1:2.4 or 1:2.8 and recall to that of 1:1.4.

From Table 7 we can also find the failed queries which have no hits in the largest neighbourhood considered, Neighbourhood_2. There were seven such queries. Their average number of the relevant documents was only 15.4, whereas that of the other 13 queries

was 36.3. The average numbers of the query words (the number of 1's in the query vector) were 2.7 for the failed queries and 4.8. for the others. For all queries the average of the relevant documents was 29 and that of the query words 4.1. This shows that both the number of relevant documents per topic and that of the query words had a positive impact on the results, which is hardly surprising.

If few relevant documents, say less than 10, belong to a topic, it is probable that most words connected with such a topic are discarded from the set of words chosen while creating document vectors. A greater number of relevant documents guarantees the words of a topic a better opportunity to remain in the chosen word set and, thus, to become good search keys. When a majority of the words of a topic with a small number of relevant documents is not included in the chosen word set, it is probable that the document vectors and a query vector of the topic consist of only few words, which is not necessarily enough so that the documents would be grouped into a compact area on the map resulting in difficulties to find a good hit.

Considering the eight failed topics in Table 7, in all of them their queries did not hit right nodes. In addition, in four of them relevant documents were dispersed on the map.

6. Discussion and comparison with clustering experiments

The results showed that it is possible to concentrate relevant documents on compact areas on a self-organising map. For instance, in Fig. 4 most of the documents relevant to the Topic 6 were in four adjacent nodes. The means of the precision and recall values in the best matched node and its closest four neighbour nodes were satisfactory, 26-50 %. The majority of the documents of each topic were inside nearby area demonstrated by Figs. 3-

6. When on average there were 29 relevant documents per query, approximately the quantity of 1/40 of all documents was relevant for each topic. If we took 30 documents fully randomly from the 1160 documents and compared them to a given query, the expectation of relevant documents would be 0.75. This means that the self-organising maps giving expectations 7.5 – 14.5 were able to produce an outcome better than ten times a random search.

The results indicated that it was not easy to find the documents of the current 20 topics applied. Moreover, the detailed results in Table 7 revealed how strongly this outcome depended on a topic. This may denote such a feature that it was not possible to separate the "lost" topics from other documents on the basis of the used variables, i.e. the chosen words. However, our technique to choose words could possibly be developed. At the moment, we discarded, after stemming, words shorter than two letters. Perhaps by eliminating words shorter than three or four letters would be more effective. On the other hand, even after stemming most German words are longer than three letters.

In order to compare the results obtained with the self-organising maps, we clustered our test data using *k*-means and hierarchical Ward's algorithms with the Euclidean measure. Along with the tests of the preceding section, we used the same 20 topics and computed average results for 10 runs. We computed these results for cluster numbers *k* of 20, 30, 40, ..., 290 and 300. In addition, we computed results for quadratic self-organising maps of sizes 4×4=16, 5×5=25, 6×6=36, …, 16×16=256 and 17×17=289. Cluster numbers that were closest to the node numbers of the self-organising maps were selected and their results were compared to those of the maps. Regarding the two clustering techniques, we selected such clusters to a neighbourhood set whose centroids were closest to the query

vector as measured with the Euclidean distance. Since the neighbourhoods of the self-organising maps consisted of 1, 5 and 13 nodes, the closest cluster, 5 closest clusters and 13 closest clusters were selected.

To condense results we combined precision P and recall R according to F value by Manning and Schűtze (2003):

$$F = \frac{2PR}{P+R}100\%.$$

In the following figures subscript 0 is for each best matching node, and subscripts 1 and 2 include its two closest afore-said neighbourhoods. Fig. 7 consists of the results of Neighbourhood_0, Fig. 8 those of Neighbourhood_1 and Fig. 9 those of Neighbourhood_2. The average results of the self-organising maps with Neighbourhood_0 are superior to those of *k*-means and Ward's clustering techniques. The best average results of *k*-means and Ward's clustering with Neighbourhood_1 in Fig. 8 (between 81 and 144 nodes) and with Neighbourhood_2 in Fig. 9 (over 196 nodes) are approximately 2-5 % higher than the best of the self-organising maps. Instead, over 169 nodes in Fig. 8 the self-organising maps gave 2-5 % better results.

Figs. 6-9 also include dashed curves which depict the means of the counts of relevant and non-relevant documents in result sets. These means naturally decrease when the numbers of nodes or clusters increase since the documents obtained are then spread over more nodes compared to the smaller numbers. Note also the differences between the three figures. The larger the neighbourhood was, the greater result sets were obtained.

To robustly compare the F values of the self-organising maps and clustering, we performed Friedman test, as usual, by calculating p value and if this was significant, pairwise differences between the methods were still dealt with. For Neighbourhood_0, the p

value of 0.18 was not significant, but trendsetting. From the three pairs, that of the self-organising maps and *k*-means clustering was nearest to the bound 0.05 of significance, which is seemingly surprising while looking at Fig. 7 advocating the self-organising maps. The explanation is that there are the means in Figs. 7-9, but the Friedman test utilises a test statistic calculated from the rank-orders of the methods within the queries. For Neighbourhood_1, *p* value was 0.57 showing no significant difference. For Neighbourhood_2, *p* value was 0.001, where the tests indicated that the *k*-means clustering was superior to the Ward's clustering and self-organising maps at the significance level of 0.05. Accordingly, the self-organising maps are able to produce as good results as the two traditional clustering techniques in such circumstances as Neighbourhood_0 and Neighbourhood_1.

Since the current precision and recall values of the self-organising maps and the two clustering techniques were not high, this may reflect the property that the document set would be difficult for any search method. In addition, there were only a few topics (20), some of which included a small number of relevant documents, the minimum being six documents. The neighbourhood forming of a self-organising map could also be developed, e.g. like using the same idea as with the clustering techniques by selecting the nodes of the best matching results and then searching for their close nodes. Viz., with the current neighbourhood form the good nodes with relevant documents might be inside the neighbourhood, but near its border and some other good nodes on the opposite side of the border. For instance, the nodes with 7 and 6 in Fig. 3 are not in the same small neighbourhood. We could sort documents inside the nodes according to their matching

property or sort documents in the vicinity of the best matching node. Sorting could furthermore be applied to several well matching nodes and their vicinities.

In future research we shall also study larger document sets than the current case. This, naturally, means that running times of the learning process in a self-organising map will grow from the current 52 s of the 17×17 map (Pentium D CPU with 3.2 GHz). On the other hand, the learning process is usually run only once and the map can then be run for searches arbitrarily many times. A new learning process is not needed until the collection is significantly extended or updated. Since a document matrix cannot, nevertheless, be increased extensively because of the relatively slow learning algorithms of self-organising maps and particularly huge sizes of such matrices which may require the use of a supercomputer, a subsequent step will be to reduce such a matrix with a suitable technique, e.g. principal component analysis.

7. Conclusions

We constructed self-organising maps to execute information retrieval and document grouping for a collection of German newspaper articles. The collection of 1160 documents was stemmed by SNOWBALL and pruned by a stopword list. The *tf.idf* values were computed for remaining search keys. Precision and recall values and number of relevant documents were evaluated for the best matching node of each of 20 queries. They were also computed for the close two neighbourhoods of those nodes measured according to link distance.

To our knowledge, the present paper is the first one to use self-organising maps literally for information retrieval. The results indicated that self-organising maps are a reasonable

means for information retrieval and document grouping. The self-organising maps coped with the test data approximately as well as *k*-means and Ward's clustering methods. Besides, the former seem to offer a good graphic means especially to express nearness and remoteness of documents in the vector space formed on the basis of *tf.idf* values. For retrieval, a subset of the topics proved hard: the query could not be placed in the vicinity of relevant documents on the map and nothing was found. These topics were those of the small numbers of relevant documents. For a few topics the scarce relevant documents were dispersed across the map, unsuccessfully from the viewpoint of the retrievals. This can also reflect the property of neural networks that they may not learn efficiently such occurrences (documents) that are infrequent in the data. For the majority of the topics, the queries gave satisfactory results.

References

Airio, E. (2006), "Word normalization and decompounding in mono- and bilingual IR", *Information Retrieval*, Vol. 9 No. 3, pp. 249-271.

Baeza-Yates, R. and Ribeiro-Neto, B. (1999), *Modern Information Retrieval*, ACM Press and Addison-Wesley, New York.

Belew, R.K. (2000), *Finding Out About, A Cognitive Pespective on Search Engine Technology and WWW*, Cambridge University Press, Cambridge, UK

Chowdhury, N. and Saha, D. (2005), "Unsupervised Text Classification Using Kohonen's Self Organizing Network", in *Lecture Notes in Computer Science, Computational Linguistics and Intelligent Text Processing*, Vol. 3406, Springer-Verlag, pp. 715-718.

Fernández, J., Mones, R., Díaz, I., Ranilla, J. and Combarro, E.F. (2004), "Experiments with Self Organizing Maps in CLEF 2003", in *Lecture Notes in Computer Science, Comparative Evaluation of Multilingual Information Access Systems*, Vol. 3237, Springer-Verlag, pp. 358-366.

Guerrero-Bote, V.P., Moya-Anegón, F. and Herrero-Solana, V. (2002), "Document organization using Kohonen's algorithm", *Information Processing and Management*, Vol. 38, pp. 79-89.

Honkela, T. (1997), *Self-Organizing Maps in Natural Language Processing*, Academic Dissertation, Helsinki University of Technology, Finland.

Kaski, S. (1998), "Dimensionality reduction by random mapping: fast similarity computation for clustering", in *Proceedings of IEEE International Joint Conference on Neural Networks IJCNN'98*, IEEE, Vol. 1, pp. 413-418.

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

Kohonen, T., Hynninen, J., Kangas, J. and Laaksonen, J. (1996), "SOM_PAK: The Self-Organizing Map Program Package", Helsinki University of Technology, Finland. http://www.cis.hut.fi/research/papers/som_tr96.ps.Z

Kohonen, T., Kaski, S., Lagus, K. Salojärvi, J., Honkela, J., Paatero, V. and Saarela, A. (2000), "Self organization of a massive document collection", *IEEE Transactions on Neural Networks,* Vol. 11 No. 3, pp. 574-585.

Lagus, K. (2002), *Text Mining with WEBSOM,* Academic Dissertation, Helsinki University of Technology, Finland.

Lagus, K., Kaski, S. and Kohonen, T. (2004), "Mining massive document collections by the WEBSOM method", *Information Sciences,* Vol. 163 No. 1-3, pp. 135-156.

Lee, C.-H. and Yang, H.-C. (1999), "A Web Text Mining Approach Based on Self-Organizing Map", in *Proceedings of 2nd international workshop on Web Information and Data Management,* ACM, New York, NY, USA, pp. 59-62.

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 and Development in Information Retrieval SIGIR'91,* ACM Press, pp. 262-269.

Lin, X. (1997), "Map Displays for Information Retrieval", *Journal of the American Society for Information Science,* Vol. 48 No. 1, pp. 40-54.

Manning, C.D. and Schűtze, H. (2003), *Foundations of Statistical Natural Language Processing,* The MIT Press, Cambridge, MA.

Moya-Anegón, F., Herrero-Solana, V. and Jiménez-Contreras, E. (2006), "A connectionist and multivariate approach to science maps: the SOM, clustering and MDS applied to library and information science research", *Journal of Information Science*, Vol. 32 No. 1, pp. 63-77.

Vesanto, J., Himberg, J., Alhoniemi, E. and Parhankangas, J. (2000), "SOM Toolbox for Matlab 5", Libella Oy, Finland. http://www.cis.hut.fi/projects/somtoolbox/package/papers/techrep.pdf

Corresponding author

Martti.Juhola@cs.uta.fi

Table 1. Effect of different weights: means of precision and recall values and number of documents in the nodes examined for 20 queries in the self-organising maps. Subscript 0 is for each best matching node and subscript 1 also includes its closest four neighbours.

Weights	$Precision_0$ %	$Recall_0$ %	$Documents_0$	$Precision_1$ %	$Recall_1$ %	$Documents_1$
Binary	26	17	18	22	26	35
Frequency	29	13	8	25	21	23
tf.idf	47	21	12	43	41	25

Table 2. Effect of document vector length: means of precision and recall values and number of documents in the nodes examined for 20 queries in the self-organising maps. Subscript 0 is for each best matching node and subscript 1 also includes its closest four neighbours.

Vector length	$Neighbourhood_0$			$Neighbourhood_1$		
	Precision %	Recall %	Number of documents	Precision %	Recall %	Number of documents
500	23	13	16	19	22	34
1000	26	17	18	22	26	35
2000	24	18	20	21	26	39
3000	21	12	18	20	22	34
4000	23	13	18	19	23	36
5000	20	15	19	19	24	32

Table 3. Effect of map size (number of nodes): means of precision and recall values and number of documents in the nodes examined for 20 queries in the self-organising maps. Subscript 0 is for each best matching node and subscript 1 also includes its closest four neighbours.

Number of nodes	$Neighbourhood_0$			$Neighbourhood_1$		
	Precision %	Recall %	Number of documents	Precision %	Recall %	Number of documents
9×9	20	19	24	19	28	48
11×11	26	17	18	22	26	35
13×13	35	18	14	32	29	28
15×15	46	14	9	37	24	18
17×17	45	11	7	41	24	17

Table 4. Effect of initialisation type: means of precision and recall values and number of documents in the nodes examined for 20 queries in the self-organising maps. Subscript 0 is for each best matching node and subscript 1 also includes its closest four neighbours.

Initiali-sation	$Neighbourhood_0$			$Neighbourhood_1$		
	Precision %	Recall %	Number of documents	Precision %	Recall %	Number of documents
random	26	17	18	22	26	35
linear	25	17	18	20	23	33

Table 5. Effect of neighbourhood computation type: means of precision and recall values and number of documents in the nodes examined for 20 queries in the self-organising maps. Subscript 0 is for each best matching node and subscript 1 also includes its closest four neighbours.

Neighbour-hood	$Neighbourhood_0$			$Neighbourhood_1$		
	Precision %	Recall %	Number of documents	Precision %	Recall %	Number of documents
bubble	26	17	18	22	26	35
Gaussian	15	19	37	15	27	57

Table 6. Effect of learning iteration ratio (numbers of ordering phase and tuning phase for maps): means of precision and recall values and number of documents in the nodes examined for 20 queries in the self-organising maps. Subscript 0 is for each best matching node and subscript 1 also includes its closest four neighbours.

Ratio	$Neighbourhood_0$			$Neighbourhood_1$		
	Precision %	Recall %	Number of documents	Precision %	Recall %	Number of documents
1/5	21	13	15	21	24	34
2/10	26	17	18	22	26	35
3/15	32	16	16	29	29	33
4/20	34	20	16	31	30	34

Table 7. Means of precision, recall and number of relevant documents for each topic from $Neighbourhood_1$ (the best matching and its four closest nodes) and $Neighbourhood_2$ (the best matching and its twelve closest nodes) of the 17×17 map.

Topic	$Neighbourhood_1$		$Neighbourhood_2$		Number of relevant documents
	Precision %	Recall %	Precision %	Recall %	
1	88	70	64	70	10
2	0	0	0	0	29
3	100	29	95	42	45
4	0	0	0	0	10
5	100	34	97	66	56
6	89	62	57	88	26
7	0	0	0	0	24
8	44	14	36	14	29
9	100	54	87	83	24
10	17	38	20	100	8
11	0	0	0	0	11
12	0	0	0	0	7
13	0	0	0	0	21
14	0	0	3	3	40
15	79	41	65	63	27
16	100	20	100	49	87
17	0	0	0	0	6
18	93	25	93	46	57
19	95	59	76	85	34
20	100	83	70	90	29
mean	50	26	43	40	29

Table 8. Effect of the size of document classes (topics): average precision, recall and relevant documents per topic after halving 20 topics into the small class S (<27 relevant documents per topic) and the large class L (≥27 relevant documents per topic) from the 17×17 map.

Topic	Neighbourhood$_1$		Neighbourhood$_2$		Number of relevant documents
	Precision %	Recall %	Precision %	Recall %	
S	29	22	23	34	15
L	71	31	64	46	43

Figure 1. Neighbourhood is defined as link distance from the best matching node in the centre. The closest neighbourhood contains four nodes, the next neighbourhood covers additional eight nodes and the third one still involves 12 nodes more.

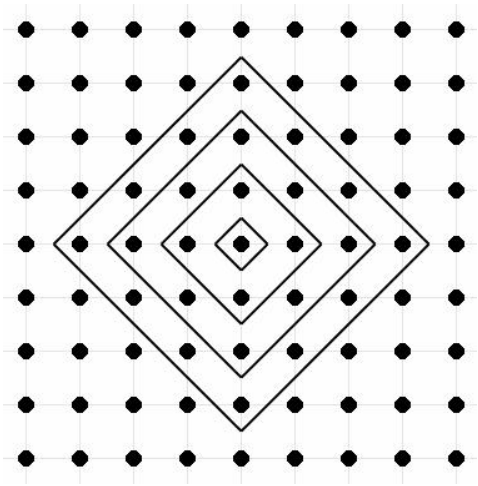

Figure 2. The self-organising map of 17×17 nodes. The darker the cluster, the more compact the document group focus.

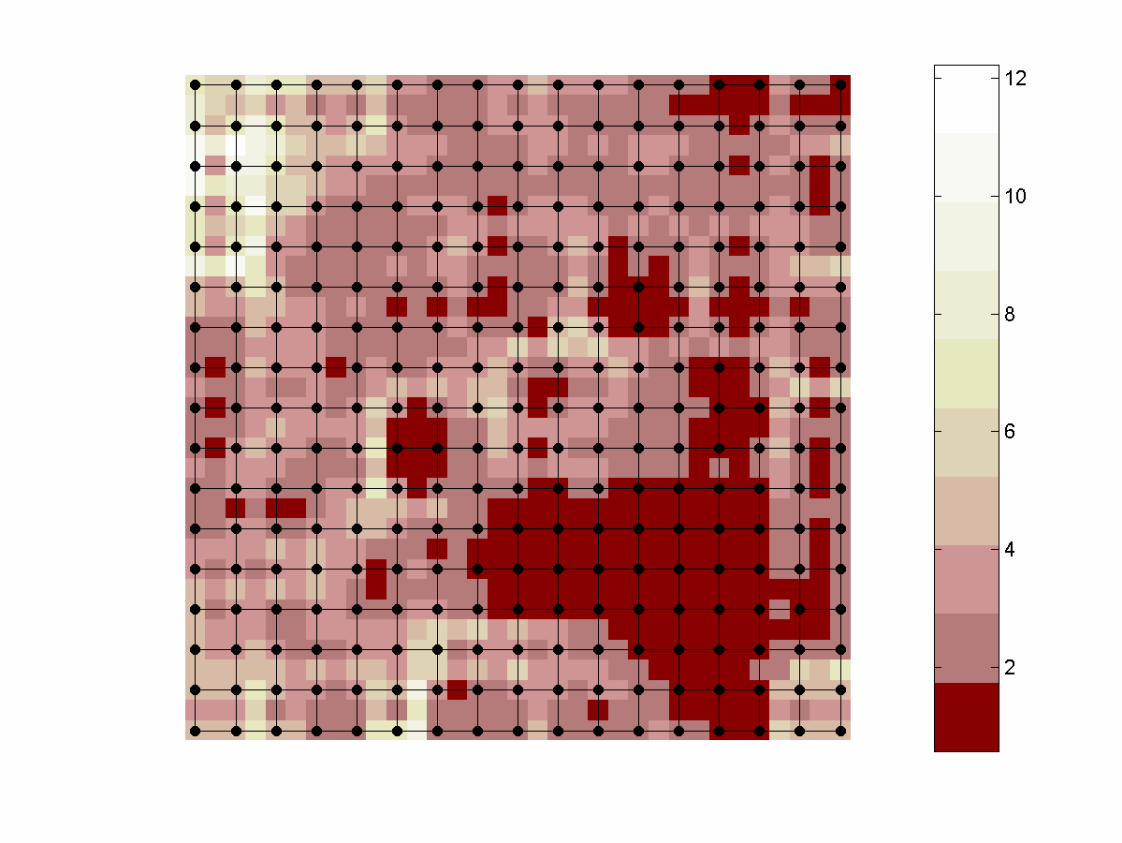

Figure 3. The topic of ‘oil accidents and birds’ as marked nodes with occurrence numbers (Topic 2).

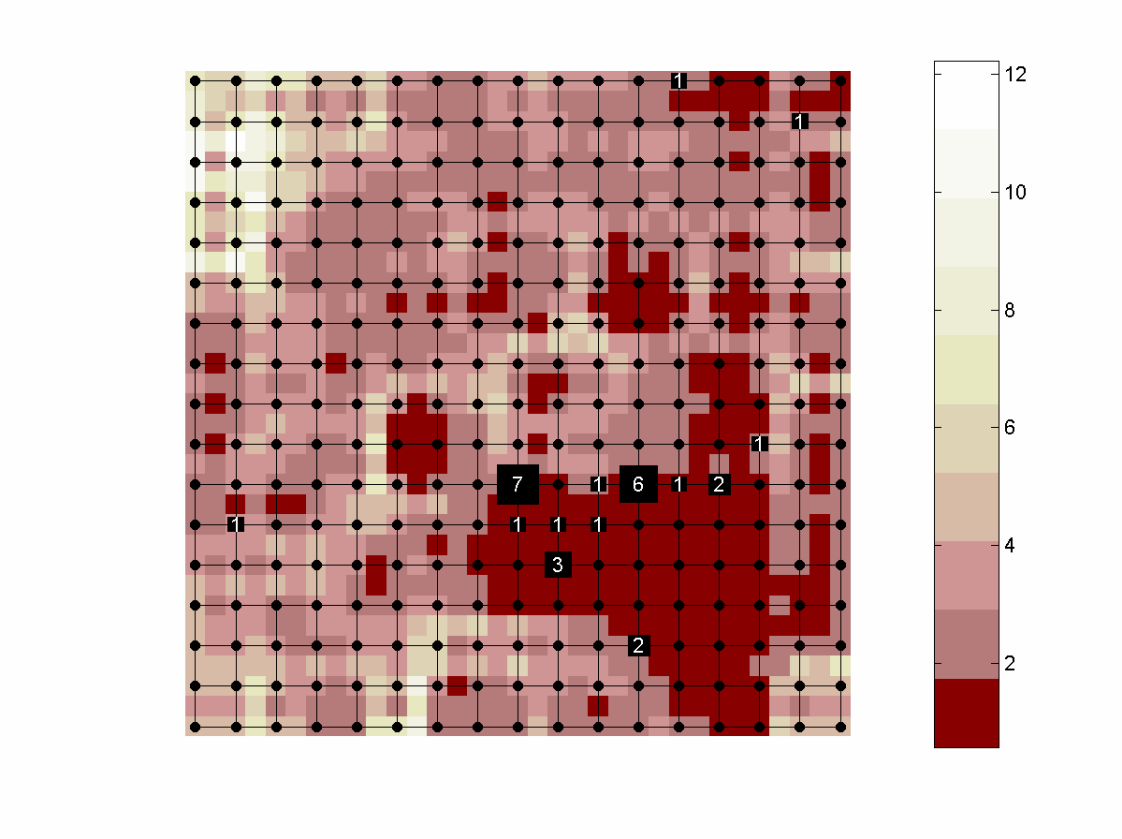

Figure 4. The topic of ‘Olympic games and peace’ as marked nodes with occurrence numbers (Topic 6).

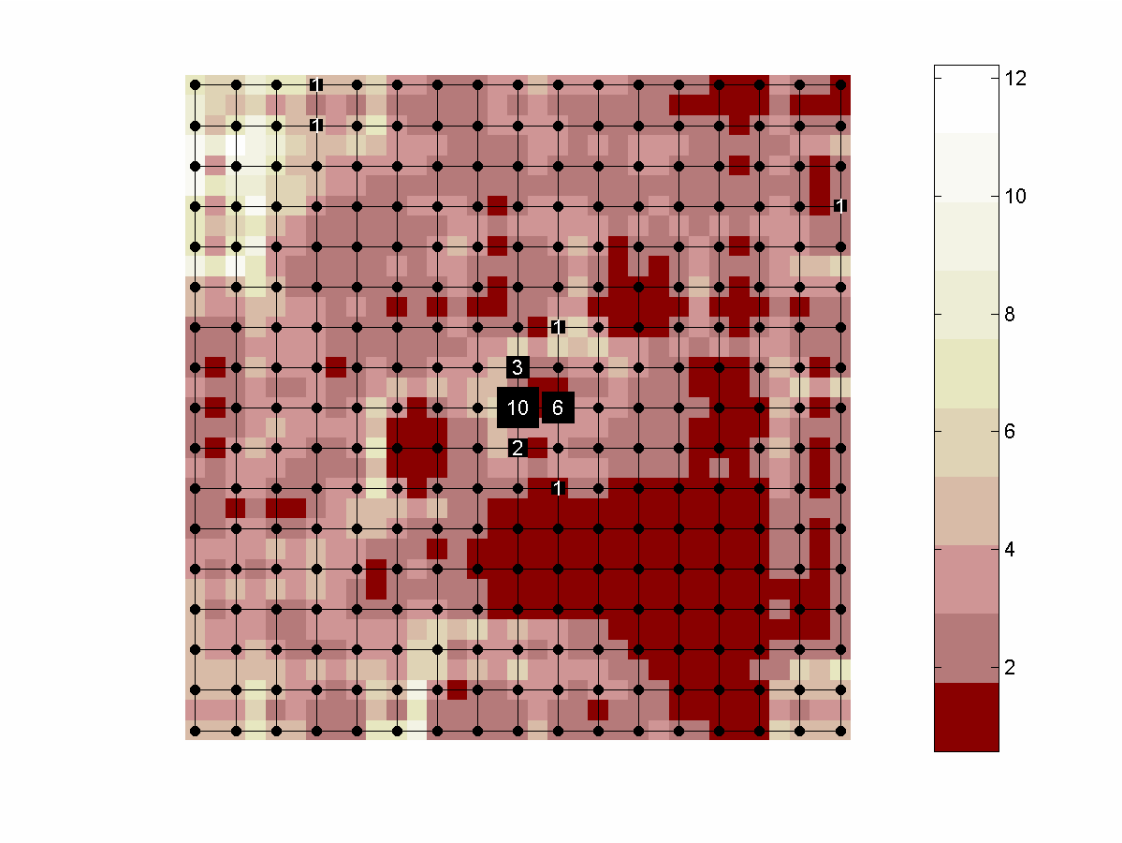

Figure 5. The topic of ‘ice hockey final in Lillehammer’ as marked nodes with occurrence numbers (Topic 10).

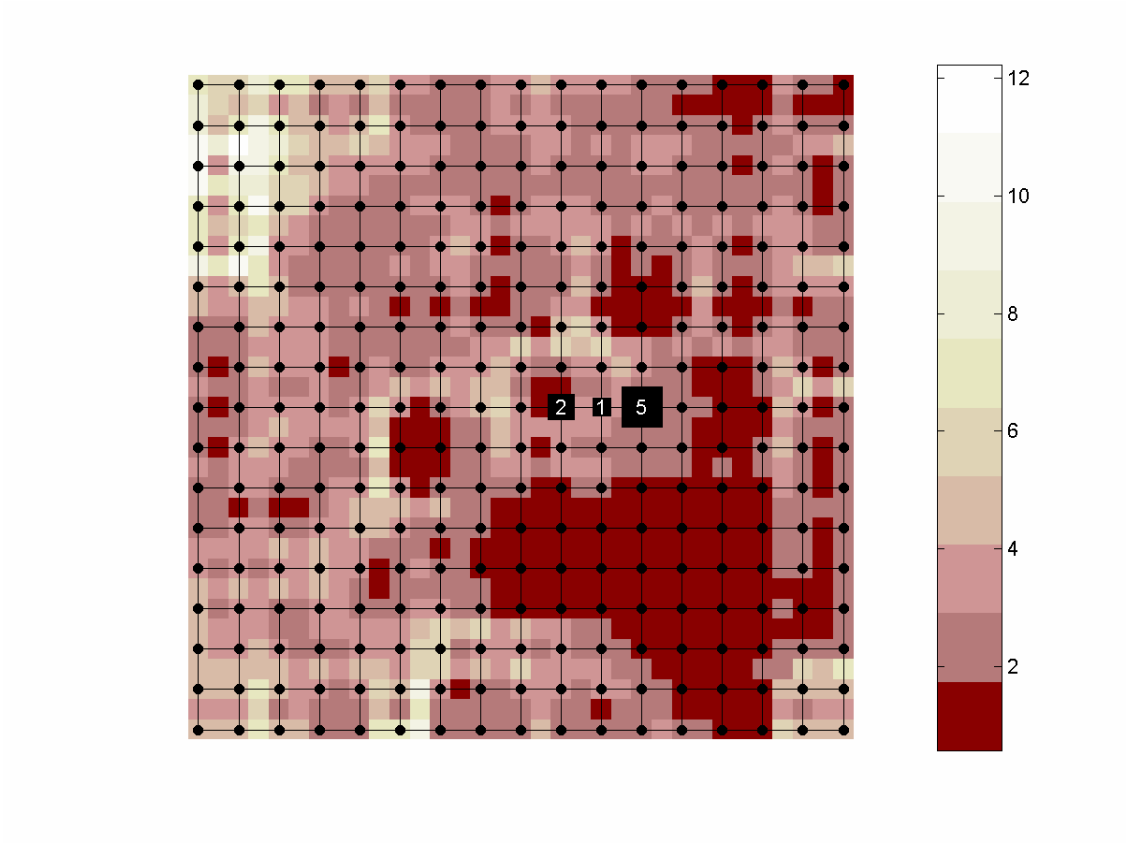

Figure 6. The topic of ‘the EU and Baltic countries’ as marked nodes with occurrence numbers (Topic 19).

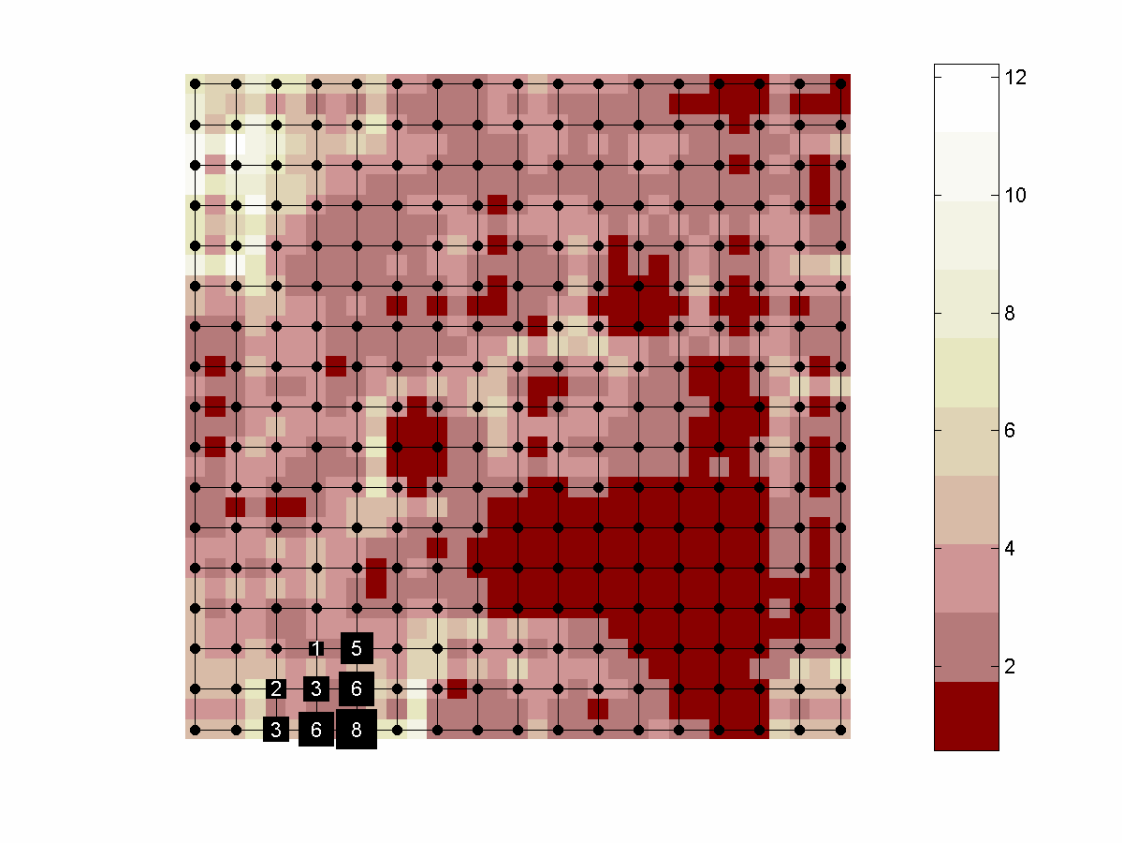

Figure 7. Comparison results of the self-organising maps, *k*-means and Ward's clustering with Neighbourhood_0, which corresponds to the best match node. Note that the results of the two latter were computed with 20, 30, 40, …, 290, 300 clusters and from them the closest numbers to 16, 25, 36, …, 289 nodes as with the self-organising maps were chosen for the current results. (This is similarly in Figs. 8 and 9.) The dashed curves depict how mean sizes of result sets i.e. relevant and non-relevant documents obtained decrease along with the increasing numbers of nodes or clusters.

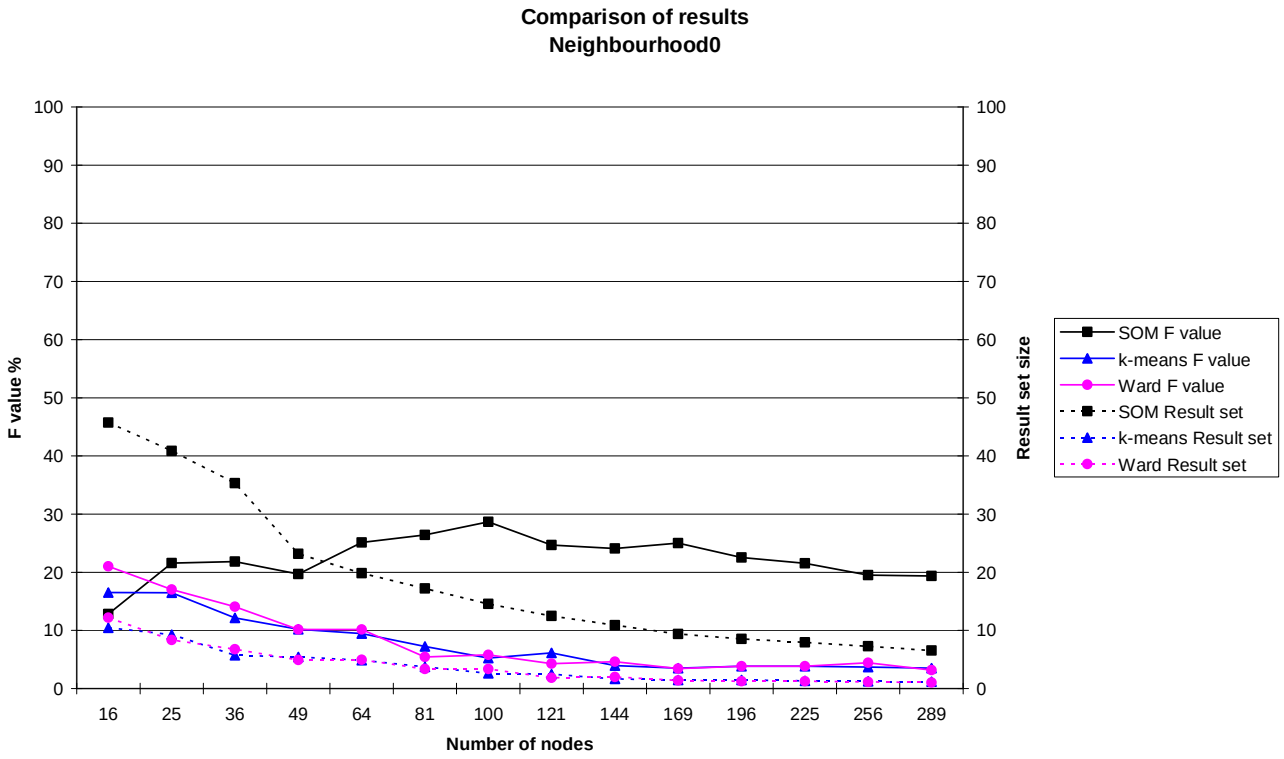

Figure 8. Comparison results of the self-organising maps, *k*-means and Ward's clustering with Neighbourhood$_1$, which corresponds to the best match node and its four closest nodes. The dashed curves present the mean sizes of document sets retrieved.

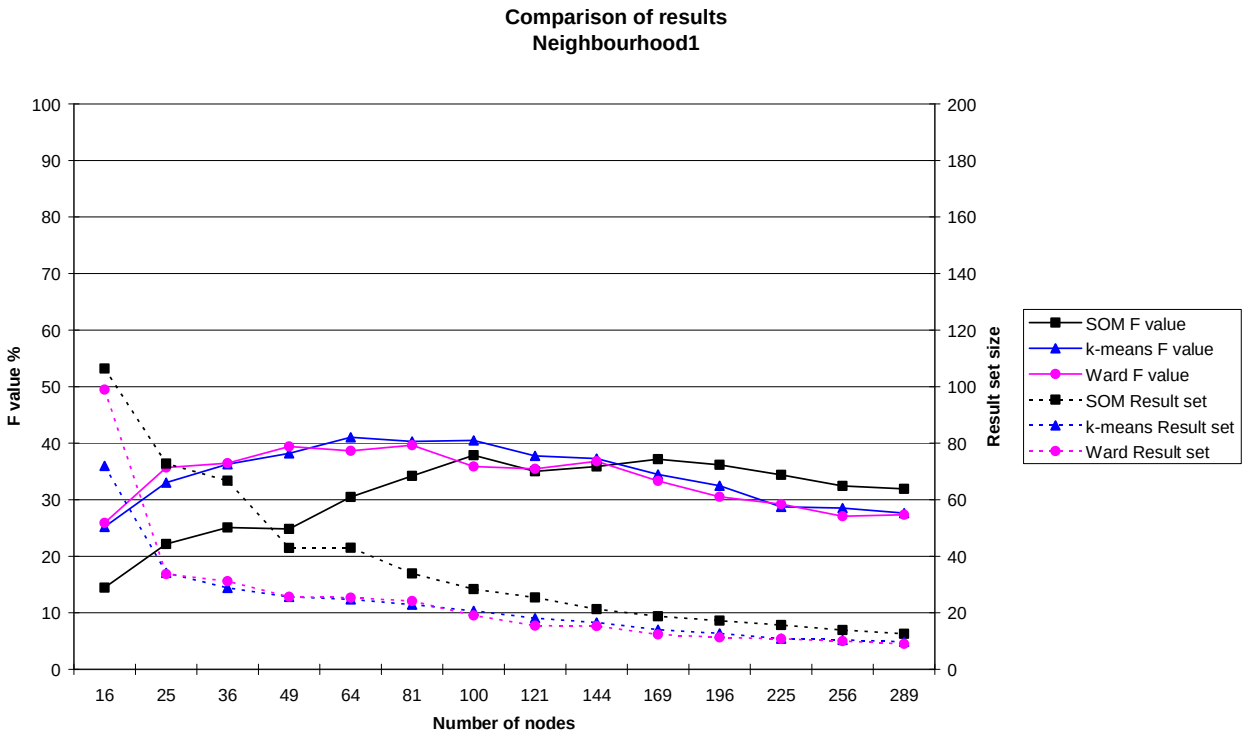

Figure 9. Comparison results of the self-organising maps, *k*-means and Ward's clustering with Neighbourhood$_2$, which corresponds to the best match node and its 12 closest nodes. The dashed curves show the mean sizes of document sets obtained.