Available online at www.sciencedirect.com
SCIENCE DIRECT®
Neural Networks 17 (2004) 1255–1271
Neural Networks
www.elsevier.com/locate/neunet

2004 Special Issue

Adaptive topological tree structure for document organisation and visualisation

Richard T. Freeman, Hujun Yin*

Department of Electrical and Electronic Engineering, University of Manchester Institute of Science and Technology, P.O. Box 88, Manchester M60 1QD, UK

Received 15 January 2004; accepted 5 August 2004

Abstract

The self-organising map (SOM) is finding more and more applications in a wide range of fields, such as clustering, pattern recognition and visualisation. It has also been employed in knowledge management and information retrieval. We propose an alternative to existing 2-dimensional SOM based methods for document analysis. The method, termed Adaptive Topological Tree Structure (ATTS), generates a taxonomy of underlying topics from a set of unclassified, unstructured documents. The ATTS consists of a hierarchy of adaptive self-organising chains, each of which is validated independently using a proposed entropy-based Bayesian information criterion. A node meeting the expansion criterion spans a child chain, with reduced vocabulary and increased specialisation. The ATTS creates a topological tree of topics, which can be browsed like a content hierarchy and reflects the connections between related topics at each level. A review is also given on the existing neural network based methods for document clustering and organisation. Experimental results on real-world datasets using the proposed ATTS method are presented and compared with other approaches. The results demonstrate the advantages of the proposed validation criteria and the efficiency of the ATTS approach for document organisation, visualisation and search. It shows that the proposed methods not only improve the clustering results but also boost the retrieval.

Keywords: Self-organizing maps; Document clustering; Information retrieval; Growing network; Unsupervised learning; Text mining

1. Introduction

Rapid increases in electronic documents and contents on the Internet, corporate intranets and local servers are leading to an information overload. The number of returned pages from a search engine, given a general query, is getting larger and their relevance is becoming lower. Automatic organisation of documents into topic categories will assist in alleviating this problem. It can be achieved using a classification approach on a set of pre-labelled documents for categorising any new documents (the so-called document classification/categorisation) (Sebastiani, 2002). In this case, the training set has to be very large or complete. Any new, emerging topics would require time-consuming human categorisation and retraining of the classifier. Document organisation can also be achieved by grouping documents into topics solely based on discovering their similarities (the so-called document clustering) (Willett, 1988), or detecting and tracking important topics over time (Allan, Papka, & Lavrenko, 1998).

Typical document clustering algorithms include partition-based methods such as buckshot or fractionation algorithms (Cutting, Karger, Pederson, & Tukey, 1992). The main advantage is the linear time computation. While the limitations are that a fixed number of centroids has to be used; it is sensitive to initial conditions and difficult in visualising a large number of clusters. To overcome these problems hierarchical agglomerative methods can be employed (Willett, 1988). They are stable and produce consistent results, but are computational intensive for large document sets and generate a full scale dendrogram, for which a user has to make decisions on appropriate levels of detail. Divisive hierarchical methods are less computationally costly and have been reported giving better results than

* Corresponding author. Tel.: +44 161 2008714; fax: +44 161 2004784.
E-mail addresses: rics@swift.ee.umist.ac.uk (R.T. Freeman), h.yin@umist.ac.uk (H. Yin).

0893-6080/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2004.08.006

the agglomerative methods (Steinbach, Karypis, & Kumar, 2000). However, they also result in a full dendrogram representation (2-way division) that has to be manually selected. There have been attempts in divisive document clustering, to split n-way, but n is fixed at each level (Larsen & Aone, 1999). Fixing the number of clusters at each level imposes a subjective composition of clusters, rather than discovers the natural structures of the dataset. An approach to determine the number of clusters based on information theoretic methods has been proposed in document clustering and better results have been reported (Slonim, Friedman, & Tishby, 2002). However, it often relies on an assumed word distribution and needs several runs until a stable representation is found. Probabilistic methods such as the expectation–maximization have also been used to cluster documents (Liu, Gong, Xu, & Zhu, 2002). They can also determine the number of clusters and provide improved performances. The main drawback is that these methods often assume a word distribution or a model for each cluster.

In general most of these methods generate a dendrogram structure and are mainly used for search and retrieval, rather than exploration or visualisation. A dendrogram can have an enormous number of branches for large document sets, making it impossible to structure and visualise the documents effectively. Neural networks, biologically inspired connectionism information processing models, can be used efficiently for this task. Their abilities to use unsupervised learning to extract complex relations from data samples make them suited for document organisation. Unsupervised learning discovers the interrelations among the documents and forms groups of topics. The neural networks that have been applied to document analysis and clustering include the fuzzy adaptive resonance theory (ART) (Tan, 2002) and the self-organising maps (SOM) (Kohonen et al., 2000). The ART networks are suitable for document clustering as they adapt to accommodate new classes and can deal with non-stationary datasets. However they are generally sensitive to noise, outliers, over-fitting and the order of input presentation (He, Tan, & Tan, 2002). Furthermore, they do not generally define a topology between clusters, which is useful in visualising relationships between topics and browsing document content.

The SOM-based methods have two distinct properties over other document clustering methods, namely non-linear dimensionality reduction and cluster topology preservation. The non-linear projection property ensures that the input space is mapped onto a lower dimensional space with minimum information distortion. The topology preserving clustering enables that similar documents or topics are located closely on the map. The output is usually displayed in a 2-dimensional map on which various clusters can be distinguished so helping navigation, browsing and discovery of similar documents. Some early work on the SOM for document analysis, showed that few features are required to create a map (Lin, Soergel, & Marchionini, 1991), but it can also be scaled to organise millions of documents (Kohonen et al., 2000). A weakness, however, is that a complex interface is needed to provide different levels of abstraction and a long training time is required. In order to provide more efficient abstractions and different levels of details, the hierarchical variant of the SOM was introduced (Miikkulainen, 1990), which also benefits from a reduced computation. However, the sizes of the maps at all levels have to be fixed, so imposing predefined limit on the number of clusters for potentially unknown datasets. This could lead to either indiscriminate (small) maps or over representing (large) maps with underused or overused resources. The growing variants were introduced to address these issues, e.g. the growing SOM (GSOM) (Alahakoon, Halgamuge, & Srinivasan, 2000). However, if the maps grow too large then they may be difficult to visualise and suffer from a long training time as in the single level 2-dimensional SOMs. Hence, the growing maps have been combined in the hierarchical structures such as the growing hierarchical SOM (GH-SOM) (Rauber, Merkl, & Dittenbach, 2002) to provide good results by abstracting levels of details with dynamic map sizes. However, the growing depends on a sensitive parameter set a priori and only one map can be viewed at a time.

Tree representation of content is a natural way of rendering information, as complex data cannot be effectively organised in a flat, single level of details on a 2-dimensional map. A survey on users browsing the generated SOMs and Yahoo structure found that when browsing on 2-dimensional maps, many subjects have difficulties forming clear associations (Chen, Houston, Sewell, & Schatz, 1998). Instead, Yahoo type directory's alphabetic and hierarchical structures are better for browsing by untrained users. These limitations of 2-dimensional maps are also found in the GH-SOM which are regarded as not representing real-world document organisation and subsequent procedures have to be considered to adapt the 2-dimensional maps for obtaining a library representation (Rauber & Bina, 2000). In fact, books in a library are generally organised using the Dewey decimal classification system, where books are sorted into a 1-dimensional hierarchy of predefined topics. In a hierarchy various levels and degrees of details, as well as relationships of parents and sub-topics, can be represented effectively.

In this paper, these issues are addressed through the use of a tree representation and a clustering validation at each level. It has been found that browsing maps takes practice (Chen, Schuffels, & Orwig, 1996). Maps are not as natural as 1-dimensional hierarchical structures, e.g. tree structures. A hierarchical growing cell structure (Hodge & Austin, 2001) has been used to generate a dendrogram. It may be difficult to browse dendrograms, especially for large datasets, because the tree can be too deep and levels of abstractions are not always clear. This is the same to the typical hierarchical document clustering algorithms, which generate a 2-way split at each level (Willett, 1988). Approaches to dealing with these limitations have been to make larger splits like n-ways

but n is fixed at each level, e.g. $n=9$ (Larsen & Aone, 1999) or to allow the selection of most interesting clusters to be further clustered (Cutting et al., 1992). However, these approaches do not address the issue of maintaining a topology between clusters, a property that is particularly useful in browsing and exploring large document sets.

It is advantageous to maintain the topology preserving property inherited in the SOM while addressing the requirement of viewing many levels simultaneously and growing levels of detail dynamically. In addition, the issue of the inefficiency of dendrograms for browsing large datasets and dynamic n-way split are also addressed. This paper presents an approach termed the Adaptive Topological Tree Structure (ATTS) that generates a topic hierarchy automatically and possesses topology and natural tree structure. The tree representation is used to categorise and organise web documents efficiently and intuitively. The hierarchical tree structure is validated automatically at each level and trained from a set of unstructured documents. The topology allows similar themes to be located closely while different themes apart at each level of the hierarchy. Each node in the tree is automatically assigned meaningful labels, and no further cluster identification or other post-processing is required. The successful Dewey decimal classification system, computer file explorers, web portals or web directories are all in 1-dimensional, hierarchical tree structures (sometimes with cross-referencing of topics). These structures generally have a varying number of sub-topics, where branches have different depths, making them natural, asymmetric tree like.

The validation of the SOM is still an unexplored area and even more so for document organisation. Davies–Bouldin validation index has been used in the SOM for clustering a non-document dataset and compared with other clustering algorithms (Vesanto & Alhoniemi, 2000). A combination of existing validation methods has been proposed to validate graph clustering using the SOM (Gunter & Bunke, 2003). But there is no topology defined in this SOM variant as it deals with graphs and was not applied to a document set (Gunter & Bunke, 2002). Little work has been done to validate document maps themselves. If the SOM is used to perform clustering then the existing validation methods from cluster analysis can be used. However, these methods do not take into account of the topology preservation. The ART, agglomerative clustering, SOM and GH-SOM, were used to organise documents and the results compared empirically (Merkl, 1995; Rauber, Pampalk, & Paralic, 2000). Supervised measures of precision-recall have been used but require known class memberships of the documents prior to clustering. The comparisons were based on inverted index retrieval (Her, Jun, Choi, & Lee, 1999; Ye & Lo, 2001) or used to evaluate the quality of different document representations (Lagus, 2002). Various measures such as map usage, average intra-node similarity, fragmentation (number of isolated map regions) and purity (intra-cluster distance) have been used. However, the validations were made on the indexing terms but not the maps (Pullwitt, 2002). In this paper, an entropy-based validation scheme is proposed for finding the underlying number of topics in the document set.

An important part of visualisation and exploration of the SOM-based methods is to automatically identify and label regions of interest. Multilayered SOMs have been used to organise documents relating to entertainment and brainstorming (e.g. the ET-MAP (Chen et al., 1996)), where each node is labelled using the best matching terms and nodes with similar terms are merged together to form regions on the maps. This is similar to the approach adopted in the LABELSOM (Rauber et al., 2002), which uses the quantisation error to determine the best terms to label the nodes without merging nodes into regions. The WEBSOM uses the term frequency-based measure to determine the labels (Lagus & Kaski, 1999). The method looks at the frequencies of the most discriminating terms occurring in a cluster. The WEBSOM uses more nodes than the other methods, so not every node is labelled, but rather there is a fixed radius of minimum distance between labels. On the map, the cluster densities are coloured and smoothed, making them more visually appealing than the ET-MAP and GH-SOM. The WEBSOM also reduces the term space using the random projection to create the document vectors. A recent approach using the WEBSOM consist of finding significant words within the reduced term space by random projection (Azcarraga & Yap, 2001).

Section 2 reviews the ART- and SOM-based methods for document organisation. In Section 3, the proposed ATTS method is presented, together with its validation criteria. Section 4, discusses and compares the results of the ATTS with the existing SOM-variants and a hierarchical divisive method. A conclusion is drawn in Section 5.

2. Document organisation using neural networks

2.1. Adaptive resonance theory-based networks

The adaptive resonance theory (ART) (Carpenter & Grossberg, 1988) can learn in either supervised or unsupervised manner. The ART addresses the stability–plasticity dilemma by maintaining stable network when presented with noisy data while still adapting to dynamic changes of the data. To do this, a vigilance parameter determines if the network represents a chosen sample sufficiently well. If this is the case then the weights *resonate* and are updated (typically winner-takes-all). Otherwise, a new node is added, until a threshold is reached. This plastic versus stable learning scheme makes it suitable for organising dynamic datasets. There are many ART-variants, which have been used in unsupervised document organisation including the basic ART1, ART2 and fuzzy-ART variant.

Some early work on document organisation using ART-like network and binary document vectors, was undertaken

(MacLeod & Robertson, 1991). Superior performance in terms of recall compared to hierarchical agglomerative methods has been reported. Recent work on evaluating the baseline quality of binary ART1 in text clustering has found that ART1 can recover about a half of the expected solutions from a text classification dataset (Massey, 2003).

Often topics overlap or documents have a degree of membership in many topics, some work has been done using soft or fuzzy-ART variants. Hybrid fuzzy-ART supervised/unsupervised method called adaptive resonance associative map has been used to cluster documents while allowing the user to influence the process by weighting important features or specifying the model hierarchy (Ong, Tan, Ng, Pan, & Li, 2001; Tan, 2002). Work on topic detection and tracking, and trend analysis was also carried out using the fuzzy-ART network (Rajaraman & Tan, 2001). A soft-version of a modified fuzzy-ART called Kondadadi–Kozma modified ART updates all the weights that resonate rather than the most similar ones to the input pattern (Kondadadi & Kozma, 2002). Better results have been reported compared to the *k*-means and fuzzy-ART.

Some hierarchical type methods have also been proposed such as the keyword generation using a set of fuzzy-ARTs (Munoz, 1997). This approach uses a set of fuzzy-ART networks to specialise in different asymmetric co-occurring terms. Relations between the terms local to each ART are then analysed to extract important associations. This method allows the creation of a dynamic thesaurus based on the document set, rather than using manually generated one that is time consuming and may not include the specific vocabulary. Another variant is the ART2 which uses the competitive Hebbian learning and has been used to organise a small set of web pages into a dendrogram (Vlajic & Card, 1999).

While the ART networks can be used for clustering non-stationary data and adapting to accommodate new classes, some disadvantages have been discussed such as being sensitive to noise, outliers, over-fitting and input order presentation (He et al., 2002; Vlajic & Card, 1998). Generally, the ART is not used to provide spatial relations amongst clusters even if some attempts have been made (Vlajic & Card, 1999). The ART optimises the inter-cluster separation while the SOM provides better intra-cluster compactness, e.g. in the quantitative comparison (He et al., 2002). The ART is dependent on many parameter settings (Merkl, 1995, 1998).

2.2. 2-dimensional SOM

Some of the earlier work on the application of the SOM in document organisation demonstrated the advantages of using topological maps in discovering similarities between documents and clusters. The WEBSOM was one of the major projects, which aimed at showing the importance of this type of representation and its scalability (Kohonen et al., 2000). The representation allows visualisation of document clusters and their relationships on 2-dimensional maps. In addition, clusters are labelled in favouring important words in a particular cluster while disfavouring the words if they are common and appear in other clusters (Lagus & Kaski, 1999). Other labelling techniques are possible, for example by extracting important terms in the reduced term space created by random projection (Azcarraga & Yap, 2001). The scalability requirements have prompted several additional optimisations strategies such as efficient feature reduction, distance calculations, winner selection and the batch updating (Kohonen et al., 2000). Fast Euclidean distance computation is performed by considering only non-zero terms. Fast winner selection uses a pointer to previous winner, and estimates a larger map using a smaller one trained with a subset of documents. The batch training, analogous to the *k*-means, has also been introduced to reduce the computation. Random projection has been used for feature reduction (Kaski, 1998; Kaski, Honkela, Lagus, & Kohonen, 1998; Kohonen et al., 2000), instead of latent semantic indexing (Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990), and produces similar results at a significantly reduced computational cost. These optimisations allow larger scale implementation. For example, the WEBSOM has been trained to organise up to about 7 million patents into a map used for interactive exploration and discovery.

The main disadvantage of using a single level 2-dimensional map is that the map size has to be defined before training and remains constant during training. It results in a single level of abstraction, thus navigation and visualisation can become cumbersome for large maps; and extra interface, e.g. panning, zoom-in and out options, has to be complemented. Another drawback is that the computational complexity is almost quadratic in the number of nodes. Whilst hierarchical structures can reduce the single level map to smaller ones at each level, so reducing the computation significantly. For example, for a 100×100 SOM, the computational complexity is roughly in order of $10{,}000^2$, while a two-level hierarchical SOM of an equivalent resolution (both levels are of 10×10 size) will require only 2×100^2 computation. Hierarchical extensions can efficiently deal with computational issues using small numbers of nodes at each level and can also effectively abstract at various degrees of details.

2.3. Hierarchical SOM variants

The general idea of using a hierarchy is to have the most general map at the top and the specific and detailed ones at the bottom. This helps navigation and visualisation, as each map deals with organising a specialised topic. Some early work on hierarchical SOMs, called hierarchical feature maps, were used to organise script-based stories (Miikkulainen, 1990). Hierarchical SOMs have also been used to organise a set of documents relating to C++ manuals, which possesses a natural hierarchical organisation

(Merkl, 1997). The hierarchical SOM begins with a single map. Once this root map reaches a stable state, documents clustered to a particular node are subsequently used to train a child map provided they meet the expansion criterion. Each child map in the second level is independently trained until they are stable and may also subsequently span another level of child maps. The main advantage of this approach is that the number of terms is reduced in child maps, which become more specialised—a property that a single level SOM does not have as the vocabulary is generally constant.

The Illinois Digital Library Initiative, or the ET-Map, was proposed for organising documents on entertainment using a multilayered SOM (Chen et al., 1996). Each node is labelled with the best matching terms. Nodes labelled with similar terms are also merged to form regions of similar topics. The multilayered SOM can reveal different levels of details and address scalability limitations of the single level 2-dimensional SOM. The Scalable SOM (SSOM) (Roussinov & Chen, 1998) was later proposed to enhance the winner search and weight updates by exploiting Boolean and sparse document features. These enhanced maps are able to scale better than the traditional SOMs. The SSOM has also been used to organise the documents retrieved from a search engine given a user query (Roussinov & Chen, 2001).

A hierarchical variant termed nested software SOM (NSSOM) has also been proposed based on the earlier software SOM for organising and searching UNIX documentation (Ye & Lo, 2000, 2001). In this method, the most important features are selected using the feature competitive algorithm, where initially a SOM is trained until a feature reduction ratio is reached. The Tanimoto similarity is adopted to compare weights and documents more accurately, taking into account of only common words. The final representation is automatically labelled for browsing, visualisation and optimising search results. An increased recall-precision over other text-based search engine has been reported (Ye & Lo, 2001). The main limitation is that many parameters have to be specified prior to the training, such as numbers of clusters at each level, minimum distances and the depth of the hierarchy.

Generally, in the hierarchical approaches the size of the maps and depth of the hierarchy are fixed. These limitations can be overcome through the use of growing SOM-variants, where the sizes of the maps are adjusted dynamically during training.

2.4. Growing variants of the SOM

There are two distinct types of growing variants. The first grows on a fixed grid, while the second expands its topology freely without such a constraint. The methods that preserve a fixed grid/topology on a lower dimensional space include the growing grid (GG) (Fritzke, 1995a), incremental growing grid (IGG) (Blackmore & Miikkulainen, 1993) and growing SOM (GSOM) (Alahakoon et al., 2000). The GG maintains a rectangular grid by inserting rows or columns between the most active node and its neighbouring node with the most different weights. The disadvantage is that the map may become very large after a few insertions (since no heuristic methods are used for node deletion) and some resources may be underused. The IGG does not maintain such a full rectangular grid but inserts nodes in unoccupied slots next to existing boundary nodes, on a fixed 2-dimensional topology. The IGG also adds or removes connections between nodes (Blackmore & Miikkulainen, 1993). It also allows insertion of nodes next to boundary nodes and between existing nodes. The GSOM has been applied to knowledge discovery and compared to IGG. It reduces the possibility of twisted maps. Different topologies or grid structures can also be used to visualise changes in a document collection (Nurnberger & Detyniecki, 2002). This method inserts new nodes in available slots next to boundary nodes. The hyperbolic SOM has also been used for text exploration and categorization (Ontrup & Ritter, 2001). Hyperbolic, instead of Euclidean, space is used to represent a spherical output suitable for ‘fish-eye’ browsing.

The second type of growing SOMs adapts its structure to the input space without using a grid and time decaying function to converge (Fritzke, 1993, 1995b). Examples are the growing cell structures (GCS) (Fritzke, 1991) and the growing neural gas (GNG) of totally unconstrained topology (Fritzke, 1995b). The GCS adds new cells between the cells with the largest quantisation errors or the most active cells, and their furthest neighbours. The newly inserted cells are then connected to their surrounding cells. Cells in low-density areas are removed. This can result in disconnected structures or sub-structures. The GCS has been applied to document clustering along with some semantic information extracted from dictionaries and has been reported to provide a good accuracy on synthetic data (Deng & Wu, 2001). One of the main drawbacks of these methods is that it is difficult to visualise high dimensional data and levels of clustering. Although the intra-cluster similarities can be represented it is difficult to measure inter-cluster similarities and memberships of documents belonging to different classes (Merkl, 1998). Generally, these methods may continue growing, resulting in a very large structure and long training times. It is difficult to navigate the map. To solve these issues the hierarchical models have been combined with the growing variants.

2.5. Growing hierarchical variants

There are two types of growing hierarchical variants, those maintaining a rectangular grid and those derived from GCS. In the former, the growing hierarchical SOM (GH-SOM) (Merkl & Rauber, 2000; Rauber et al., 2002) uses the hierarchical association of maps (Merkl, 1998; Miikkulainen, 1990) with a growing algorithm similar to the GG for map growth (Fritzke, 1995a). The GH-SOM has been applied to several document sets such as the documents

returned from a search engine (Rauber & Bina, 2000), the CIA world fact book (Merkl & Rauber, 2000) or news articles (Rauber et al., 2002). The LABELSOM (Rauber, 1999) is used to assign meaningful labels to each node. Initialisations of sub-maps are done using parent maps (Dittenbach, Rauber, & Merkl, 2001). This method is autonomous and efficient in generating a hierarchy of maps, but is sensitive to parameters such as maximum growth and depth.

The second growing hierarchical type is called the TreeGCS (Hodge & Austin, 2001, 2002), which uses the GCS structure to add or remove cells. To improve stability, cell deletions are only performed after 90% of the maximum number of cells is reached. The deletion of cells allows the formation of sub-structures, of which the clustered patterns are placed in a dendrogram. To further improve the stability of the growth and dendrogram, the documents are presented in a cyclic order (rather than random) and appropriate parameters have been suggested (Hodge & Austin, 2001). These factors ensure that the cell structures have reached a stable state before the splitting process takes place. The TreeGCS has been used for the organising the CIA world fact books and for automatic thesaurus generation (Hodge & Austin, 2001, 2002). The results are presented in the form of a dendrogram that can be used to explore the hierarchical relationships of the clusters. The method is generally computationally intensive (in the order of $O(n^3)$) (Hodge & Austin, 2001) and in a dendrogram representation most of the topological relations obtained by the GCS can be lost.

3. Adaptive topological tree structure (ATTS)

3.1. Overview

The ATTS uses a set of hierarchically organised and independently spanned 1-dimensional growing SOMs, or growing chains. Preliminary work on document analysis using a tree type SOM has been first explored (Freeman, Yin, & Allinson, 2002) and later enhanced with short textual context (Freeman & Yin, 2002). The process of the tree type SOM is to let each chain grow until some terminating quantisation error is reached. Once a chain is trained each node is tested to determine if documents clustered to it should be further clustered into a child chain. This process is similar to divisive clustering except where the number of clusters is pre-defined. Generally, a stopping condition has to be decided before the training. If the criterion is set too high then map becomes too indiscriminative, if too low the chain becomes a flat partition. Another point is that it uses the Euclidean distance in sparse document space, where the cosine correlation is more appropriate (Salton, 1989).

In the proposed ATTS, the size of each chain is independently determined through a validation process using an entropy-based Bayesian information criterion (BIC). Once the size is determined, documents clustered to a node are tested to determine if they should become a leaf node or should be further expanded into a child chain. The child chains have smaller vocabularies compared to their parents—this allows more specialised topic clusters while significantly reducing computing complexity. The successive addition of child chains effectively forms a taxonomy of topics possessing a topology at each level. In addition, the ATTS incorporates the methods for dealing with sparse vectors such as fast dot product, winner search method, and weight update function. Furthermore, adaptive node insertion for each chain is performed using both interpolation and extrapolation. In the child chains weights are initialised based on their parent nodes to improve convergence. All the nodes in the tree are given representatives labels taking into account of term frequencies and node weights. Important relations between terms in the same cluster are extracted using a fast association rule. These detected association terms complement the labels to enhance browsing and exploration of the dataset. A general diagram of the overall system and its features is shown in Fig. 1.

3.2. Indexing and feature selection

In the pre-processing stage, the document files (html or other formats such as plain text, Word and PDF) in the dataset are crawled, text enclosed in the title tags are recorded and the html is parsed to turn into plain text. The title and body text are then indexed using a sliding window of sizes p (e.g. for $p=4$, up to 4 successive words are used for each term) over sentences, while ignoring common words in a stop-list. Since the number of terms is significant, the least discriminative terms can be discarded: those occurring in less than a few documents and those that occur in more than a percentage of the total documents. In the vector space model (Salton, 1989) a document is stored as $[f_1,\ldots,f_N]$, where f_i is the frequency of term i and N is the total number of unique terms.

Since a document vector $\mathbf{x}$ is sparse, each of its term frequencies can be stored in a compressed representation as a hash table for efficient individual retrieval

$$\mathbf{x} = [\{\gamma_1, \rho_1\}, \{\gamma_2, \rho_2\}, \ldots, \{\gamma_r, \rho_r\}], \quad \gamma_k \in \{V | f_{\gamma_k} \neq 0\}, \quad k = 1, 2, \ldots r \tag{1}$$

where document $\mathbf{x}$ is stored as pairs: γ_k and ρ_k. γ_k refers to the γ_kth term in the vocabulary V, ρ_k is its weighting-defined by Eq. (2). r is the total number of terms in this document. The typical information retrieval weighting scheme for terms (term frequency multiplied by the inverse document frequency (Salton, 1989)) is used. This scheme is further combined with a function, which weights more on the terms containing multiple words, and a normalisation to take into

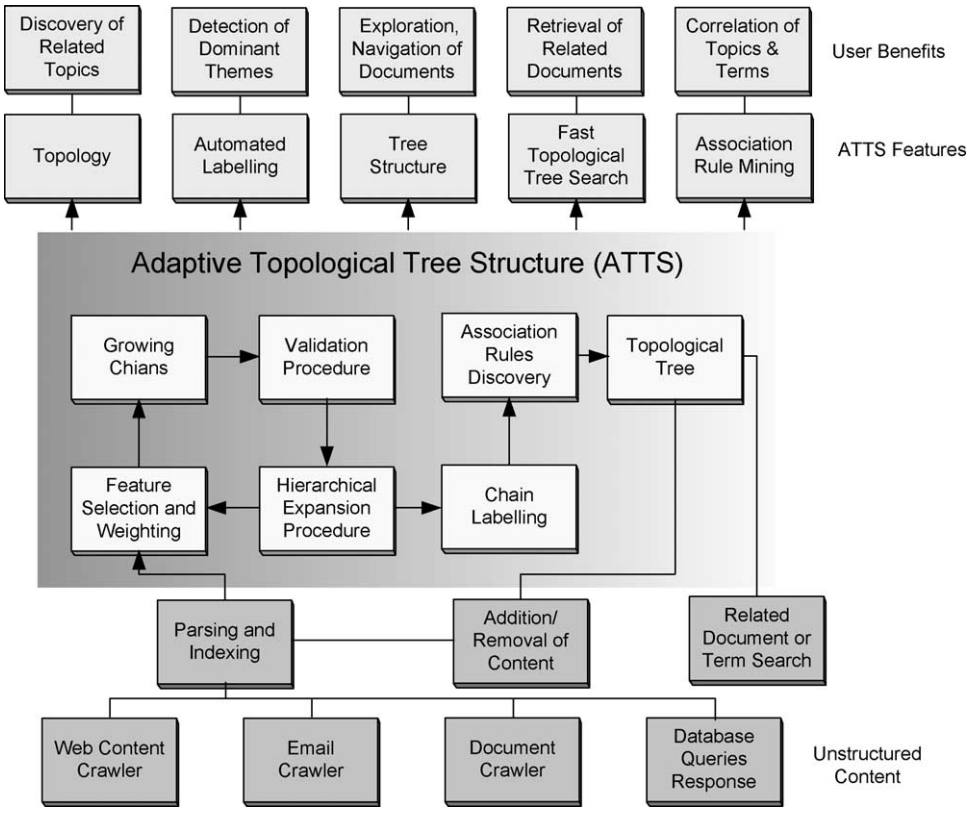

Fig. 1. The block diagram and features of the Adaptive Topological Tree Structure (ATTS).

account of different document length:

$$\rho_{ij} = \left(\frac{\xi \cdot f_{ij}}{s}\log\frac{m}{D_j}\right) \Big/ \left\|\frac{\xi \cdot f_{ij}}{s}\log\frac{m}{D_j}\right\| \tag{2}$$

where ρ_{ij} denotes the final weighted frequency of term $\gamma_j$in document i. ξ is the importance factor of a term, based on how many words are in the term (Freeman & Yin, 2002). $s = \xi(\text{single-word}) + \cdots + \xi(p\text{-word})$. f_{ij} is the frequency of term j in document i, m is the total number of documents in collection, and D_j is total number of document containing term j.

3.3. Self-organising chains

The ATTS uses a set of chains adaptively developed by a growing chain (GC) algorithm. Each chain begins with two nodes and grows until a termination process stops it (see Section 3.4). In the root chain, the weights of the nodes are initialised to small random values, while child chains are initialised using the parent nodes as described in Section 3.5. There are two major steps in the GC training: search for the best matching unit and update of the winner and its neighbourhood. At time t, an input document vector $\mathbf{x}$ is mapped to a chain consisting of n nodes. The weight vector of node j, $\mathbf{w}_j = [w_{j1}, w_{j2}, \ldots w_{jN}]^T$, is of N dimensions (where N is the number of unique terms in the collection). In order to compare the similarity between the node weights $\mathbf{w}$ and document vector $\mathbf{x}$, a modified dot product is used as the input is expressed as a hash table:

$$S_{\text{dot}}(\mathbf{x}(t), \mathbf{w}_j) = \sum_{k=1}^{r(t)} \rho_k \cdot w_{j\gamma_k}, \quad \forall \{\gamma_k, \rho_k\} \in \mathbf{x}(t) \tag{3}$$

The dot product between two vectors can be computed efficiently as $r(t) \ll N$ in almost all cases, The best matching unit $c(\mathbf{x})$ is the node with maximum S_{dot} amongst all the nodes with respect to the document $\mathbf{x}(t)$

$$c(\mathbf{x}) = \arg\max_j \{S_{\text{dot}}(\mathbf{x}(t), \mathbf{w}_j)\}, \quad j = 1, 2, \ldots, n \tag{4}$$

where n is the current number of nodes in the chain. Once the winner node $c(\mathbf{x})$ is found its weight and the weights of its neighbourhood are updated using,

$$\mathbf{w}_j(t+1) = \frac{\mathbf{w}_j(t) + \alpha(t) h_{j,c(x)}(t)\mathbf{x}(t)}{\|\mathbf{w}_j(t) + \alpha(t) h_{j,c(x)}(t)\mathbf{x}(t)\|} \tag{5}$$

where $\alpha(t)$ is a monotonically decreasing learning rate and $h_{j,c(x)}(t)$ the neighbourhood function, typically a Gaussian kernel. In order to reduce the noise from the sparse vectors, the weights smaller than a threshold (e.g. 0.00001) are set to zero. Table 1 outlines the GC algorithm in pseudo-code.

3.4. Node insertion and validation

To monitor the activity of each node in a chain a counter is used to record the node's winning times. The counter is set to zero at the creation of the chain and reinitialised after

Table 1
Pseudo-code for the growing chain

```
If root_chain
  Initialise 2 nodes randomly
Else
  Initialise 2 nodes using parent nodes
For n=2:n_max
{
  While (AvgSim not converged)
  {
    Select document x(t)
    Find winning node via Eq. (4)
    Update chains weights using Eq. (5)
    Calculate AvgSim by Eq. (6) //can be calculated
less frequently
  }
  Calculate external validation measure Θ(n) Eq. (10)
  Record current validation Θ(n) and weights
  Insert new node as described in Section 3.4
}
Determine best number of nodes τ via Eq. (11)
Restore chain with τ nodes
```

each node addition. A measure termed average similarity (AvgSim) is recorded during the training

$$\text{AvgSim} = \frac{1}{n}\sum_{j=1}^{n}\frac{1}{m_j}\sum_{i=1}^{m_j}(S_{\text{dot}}(x_i, w_j)), \quad m_j \neq 0 \tag{6}$$

where n is the current number of nodes in the chain for $m_j \neq 0$ where $2 \leq n \leq n_{\max}$ and $n_{\max}$ is the maximum number of nodes used for the validation. m_j is the number of documents clustered in a particular node j and $\mathbf{x}_i$ are the documents i such that $c(\mathbf{x}_i)=j$. Each cluster is normalised irrespective of its size as given by m_j, to give the documents of different lengths equal importance.

Once a chain is judged to have sufficiently converged via the stabilisation of AvgSim, a new node is inserted. The new node is inserted in the proximity of the node with the largest activity, to its left or right, depending on which leads to a higher AvgSim. Its weights are initialised using interpolation or extrapolation, depending on which increases the AvgSim more. If the nodes are inserted at boundaries, their weights are initialised through (linear) extrapolation. Since, no term frequencies in the document vectors are negative, if this difference is negative the weights are set to zero.

During the chain growth and just before a node is inserted, the values of all the weights in the chain are stored along with the validation measure for the current number of nodes n. The chain is allowed to grow to a specified maximum number of nodes, e.g. $n_{\max}$. The validation procedure then chooses the optimum number of nodes to represent the topics in the document set. Choosing the most suitable size in this way is less costly than full growth (possibly up to one cluster for each document) followed by pruning process. Likewise other methods, which simultaneously add and remove of nodes until a threshold or maximum number of iterations is reached (like in ART or GCS), are also costly and provide little gain especially if nodes get added and removed continuously at the same location.

The purpose of using a validation is to independently discover the number of underlying topic clusters. Typical measures aim to maximise the intra-cluster (within the cluster or compactness) and minimise the inter-cluster (between clusters) similarities. In real-world applications, it is usually impractical to optimise both values since many outliers, noise or overlapping clusters are present in the data. The validation is applied locally to each chain, taking into account of its reduced and specialised vocabulary. The Bayesian information criterion (BIC) (Schwarz, 1978) has been proposed to validate clustering of a model based approach and has found wide applications. The BIC infers the correct number of clusters by maximising the likelihood of term distributions under the model, with a penalty term to discourage the tree to become a flat partition. A probability distribution model, such as Gaussian, is needed for the BIC and the results can be influenced by model parameters.

Instead of using (often continuous) models, the (discrete) term distribution of a term in a cluster can be estimated by the probability of that term in that cluster

$$p(t_i|C_j) = \frac{c(t_i|C_j)}{\sum_{t_i' \in C_j} c(t_i'|C_j)} \tag{7}$$

where $c(t_i|C_j)$ is the frequency or occurrence of term t_i in cluster C_j.

The use of likelihood also implies equal prior probabilities for all the terms, which is not often the case in practice. As the term probabilities are available now, one can take the expectation of the (log) likelihood, resulting in an entropy H of cluster C_j as:

$$H(C_j) = -\sum_i p(t_i|C_j)\log p(t_i|C_j) \tag{8}$$

The total entropy is the normalised and weighted sum of entropies for all clusters

$$H_t(n) = \frac{1}{m}\sum_{j=1}^{n} m_j \cdot H(C_j) \tag{9}$$

where m is the total number of documents, and m_j is the size of the cluster. Choosing n with the lowest H_t favours large numbers of clusters as entropy decreases with an increase in the number of clusters. Hence, the entropy-based BIC validation measure can be defined as

$$\Theta(n) = H_t(n) + \frac{1}{2} n \log m \tag{10}$$

The most suitable number of clusters τ, is found by:

$$\tau = \arg\min_n\{\Theta(n)\}, \quad n = 2, 3, \ldots, n_{\max} \tag{11}$$

Then the chain with τ nodes is restored and added to the hierarchy, and its nodes can be used for further clustering

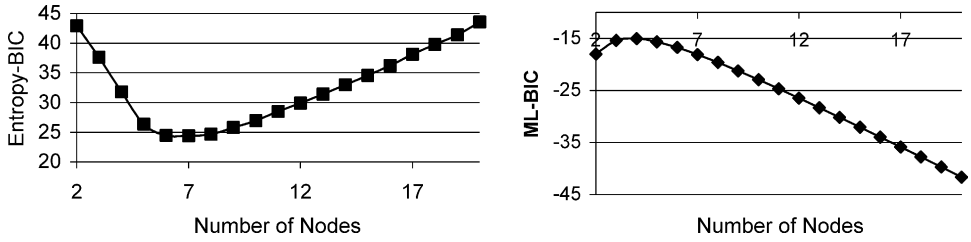

Fig. 2. (a) The proposed entropy-based BIC validation and (b) likelihood based BIC using a Gaussian model of the growing process.

(of non-leaf nodes) or final presentation (leaf nodes). An example on a known dataset of seven clusters using the average of 10 runs of the proposed entropy-based BIC validation curve and the standard BIC is given in Fig. 2. In this case, both weight initialisation and document selection were random, however, stable results were produced. It clearly demonstrates that the proposed validation method has chosen the correct number. The proposed entropy-based criterion has shown consistent better performance over the original BIC metric in various experiments.

3.5. Adaptive tree structure

After the root chain has been trained and its optimum number of nodes identified, each node is tested to see whether documents clustered to it need a further sub-level division, i.e. to span a child chain. One heuristic approach is to allow a minimum number of documents in each child chain. Another test then checks the number of terms in the chain vocabulary as explained in Section 3.2. If the chain only contains few terms it indicates that the parent node is sufficiently specialised, the documents may already be very similar, and there are insufficient terms for further classification. The final test is to use a cluster tendency method called term density test, which relates number of average frequency of terms to number of terms in dictionary, to determine if it is worth further clustering the set (El-Hamdouchi & Willett, 1987). It is to ensure that there are a sufficient number of terms shared by the documents, to allow discrimination between them. If any of these tests fail then the parent node becomes a leaf node. Fig. 3 shows an example of self-generating topological tree tree structure.

If a node spans a child chain, then all documents mapped to the node form a new subset of documents with a reduced vocabulary. The term weightings are adjusted using Eq. (2) based on the original term frequency. The feature selection and weighting are important because the parent node has already used specific terms to cluster the documents some of which may no longer be discriminative (e.g. terms originally

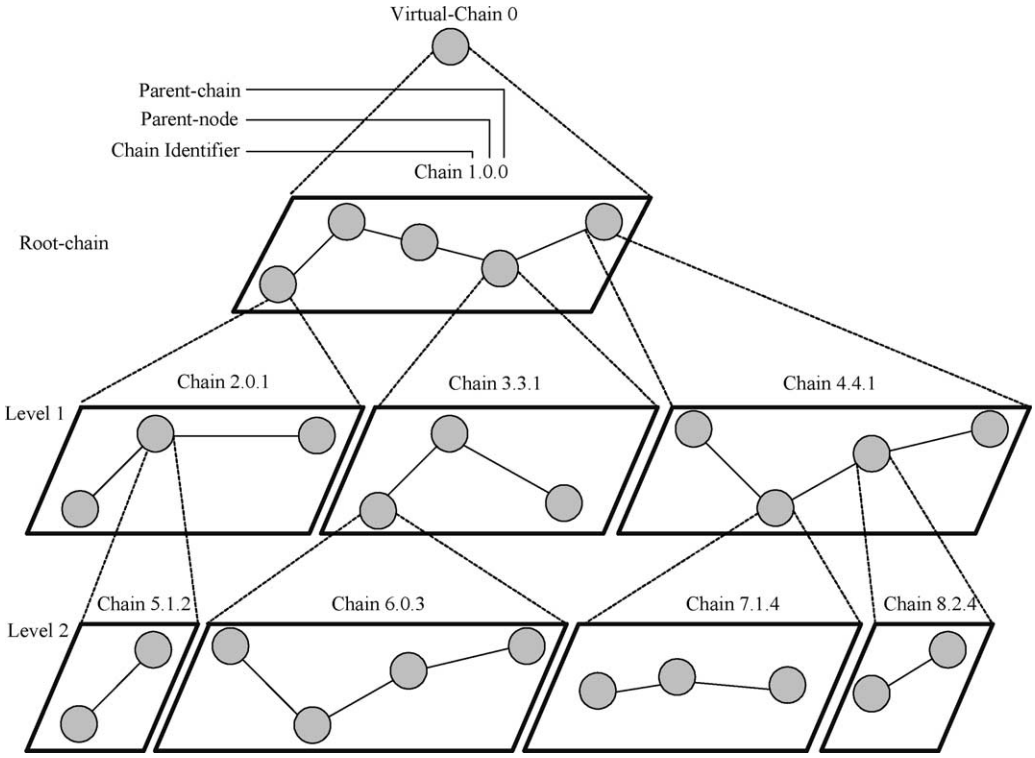

Fig. 3. Example of a generated topological tree, with chains spanned from parent nodes.

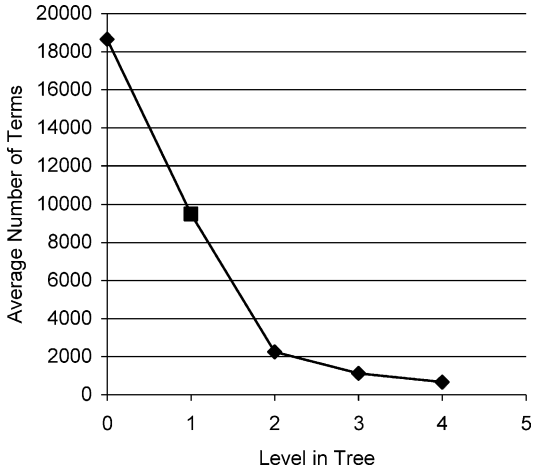

Fig. 4. The average number of terms or vocabularies at each level in the topological tree on dataset B.

occurring in 5% of the document in the parent set may now occur in 95% of documents the child set). In this way, each addition of a new chain to the tree contributes to the specialisation of the topics and vocabulary in child levels. An example of the average reduction in number of non-zero terms between levels is shown in Fig. 4.

The two starting nodes in a child chain are linearly initialised based on its parent node and its immediate neighbours. If the parent node is on a boundary, then the two child nodes are initialised based on the parent nodes and its one-sided neighbour. This is to ensure that the child chain has the same orientation as the parent chain, especially around its parent node. The process of initialising weights is analogous to linear initialisation using the largest eigenvectors (Kohonen, 1997) or an efficient computation using smaller maps to estimate larger maps (Kohonen et al., 2000). This allows the child chains to proceed more quickly to convergence phase.

3.6. Visualisation and implementation

A tree structure is a natural and intuitive way to present contents and their structures as described in Section 1, and it is more helpful and advantageous for understanding and navigating materials than other presentations such as 2-dimensional grids, flat partitions or dendrograms (typically generated from hierarchical clustering). The proposed ATTS method uses the GCs as building blocks to represent the contents in a topological tree structure. It does not require any manual intervention. To obtain a meaningful topic hierarchy and ontology, a number of most descriptive key words have to be assigned to each cluster. All the terms judged statistically irrelevant are first removed from the possible terms used for labelling. The term score ts_{ji} is calculated for the remaining terms i in node j for document k

$$ts_{ji} = \sum_{k=1}^{m_j} \rho_k(i) w_{j\gamma(i)}, \quad i = 1, 2, \ldots, r_k \tag{12}$$

where i corresponds to the sparse index of the document as described in Eq. (1). Simultaneously, ranking the terms by frequency of occurrence and ts_i (as primary key) provides understandable and quality and cluster labels. The first few terms (e.g. 5) with the highest ranking values are selected to label the cluster. The set of labels summarises the content of the clusters and allows the user to quickly identify and grasp the underlying topics at a particular level and their most related areas though the topology of the chains. This labelling scheme includes the same terms used for clustering, which reflect the importance of key terms in the nodes weights. Terms constructed with multiple words are marked with single quotes to distinguish them from single words. To avoid duplication of terms a bottom-up heuristic method is used to allow more concise labelling, as only limited text space is available for the labels.

In addition to cluster labels, a list of frequently co-occurring terms is provided for each cluster. The search for frequent associations originates from the association rule mining of transaction databases (Hipp, Guntzer, & Nakhaeizadeh, 2000). More recently, there has been a focus on extracting frequent itemsets, which are the associations between items that meet a threshold of minimum support. Using these itemsets as termsets in document clustering is an efficient method to perform feature selection (Beil, Ester, & Xu, 2002). The Apriori algorithm (Agrawal & Srikant, 1994) has been used to find the termsets, which considers the association of multiple terms. In the ATTS, the number of associations is limited to two terms since multiple-word terms are already taken into account at indexing. The objective is to extract the strong associations within the same cluster as $X \Rightarrow Y$, e.g. economy ⇒market (support, 6%; confidence, 85%). There are two important measures in association detection, support and confidence. Support (supp) is defined as the count of dual term associations divided by the total count of term associations. The confidence is the measure defined as: if X occurs then Y will follow

$$\text{conf}(X \Rightarrow Y) = \text{supp}(X \cup Y)/\text{supp}(X) \tag{13}$$

A hash table is used to record all associations and their frequencies in one pass over the document vectors belonging to a particular cluster. Associations not meeting the minimum levels of support and confidence are discarded. Finally, the associations are ranked by confidence (as primary key) and support. This provides complementary descriptions of the clusters in addition to the cluster labels.

In order to search for related documents in a topological tree, the leaf node that best matches the query needs to be identified. The search through the tree begins at the root chain, where the winning node is identified. The child chain generated from that root node is then searched for the winning node and this process continues for the descendent chains until the winning leaf node is identified. This path

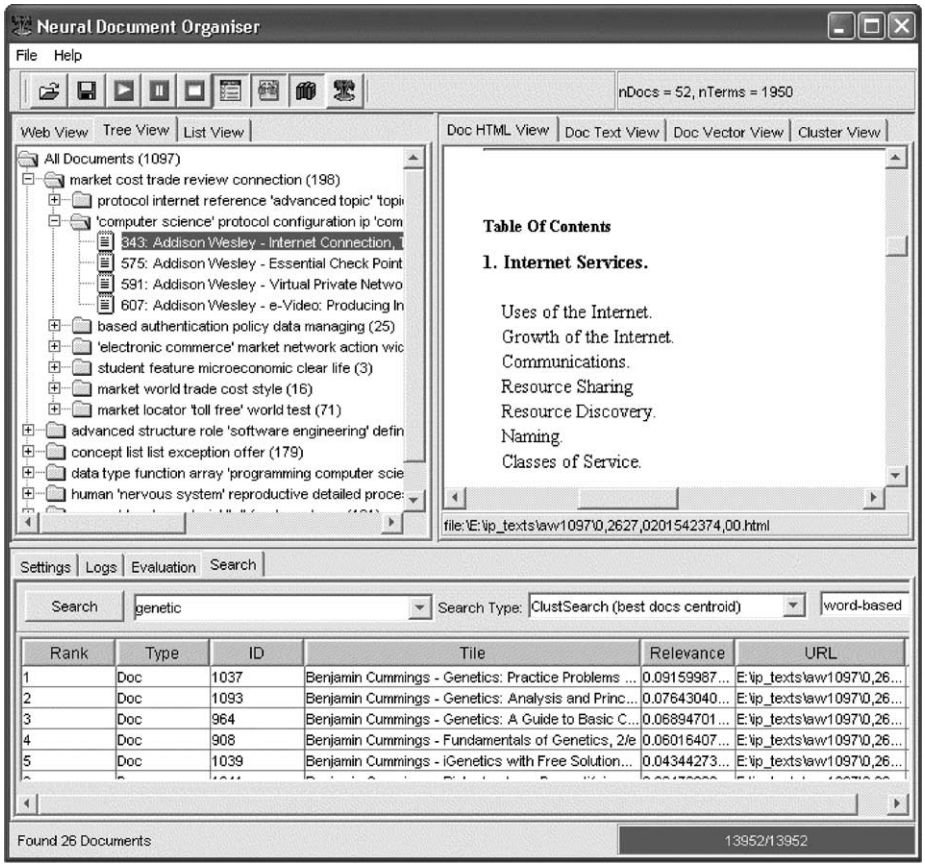

Fig. 5. Screen shot of the implemented application, with the left showing the topological tree organisation of the dataset, right the document/cluster details, and lower pane the settings and the query, as well as its respective ranked retrieved documents.

through the tree can be referred to as the winning path. Rather than searching all the nodes for the winner, the topology property has been used in this search mechanism. The topology ensures that similar documents are located close to the winning leaf node. Hence, these nodes can also be used to return relevant documents, as well as those on the winning path. This is a key property of using the topological trees.

A full application package featured with a graphical user interface has been constructed to allow efficient testing on real-world document sets. This application is implemented in Java for object-orientation and cross-platform support. It includes a crawler, full-text parser and indexer, pre-processor, various clustering and organisation methods, search engine, and visualisation support. The program is given a set of documents and user settings as input, and outputs a topological tree view. A typical result of the application package is shown in Fig. 5 with the left window representing the (mouse sensitive) tree, the right displaying the document or cluster viewer, and the lower pane providing the general settings and search options. Selecting a tree node will reveal information about the cluster (terms, weight, *AvgSim*, validation measures, term associations, etc.). Other output formats are also possible, such as a JavaScript tree or XML representations. The SOM, GH-SOM and bisecting *k*-means methods have also been implemented in the package for comparison. The GH-SOM expresses its hierarchical relations through the use of hyperlinks.

4. Results and discussions

In this section, results of the SOM, GH-SOM and ATTS are compared. All three methods have been applied to the same datasets with the same pre-processing techniques described in Section 3.2. All three methods used the same dot product and speedups as described in Section 3.3. The datasets used, listed in Table 2, are moderate to allow manual verification of clusters. The hierarchical methods ATTS and GH-SOM do not produce very deep or brief structures. While the ATTS has the minimum computation time amongst these three methods.

Table 2
The datasets used in the experiments

Dataset	Number of documents	Number of topics	Source
A	1097	*n/a*	http://www.aw.com
B	2001	*n/a*	http://news.bbc.co.uk
C	402	8	Reuters21578
D	899	14	Reuters21578

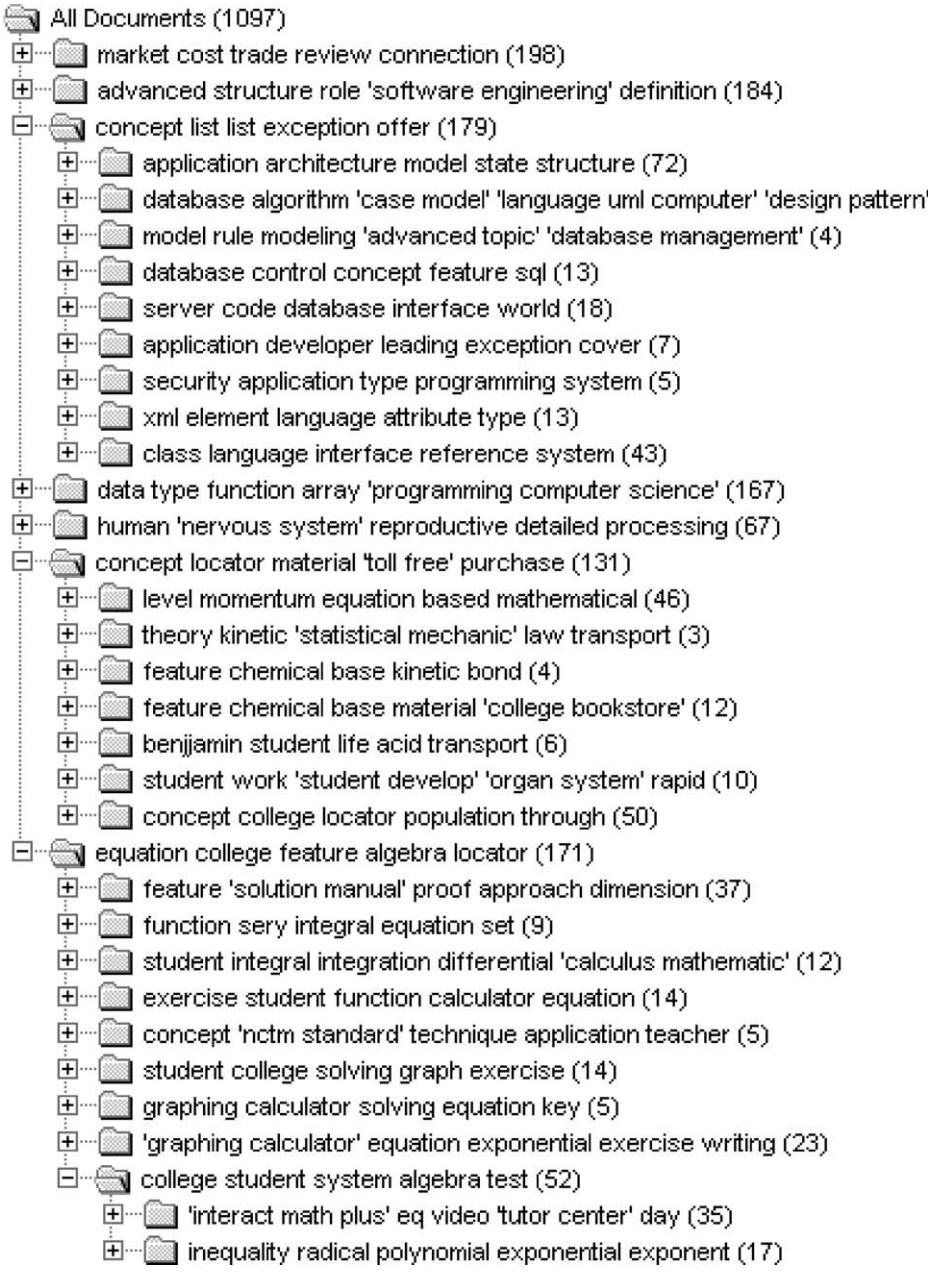

Fig. 6. A topological content tree generated using the ATTS method on dataset A.

Fig. 6 displays the result of the ATTS method applied on dataset A. The topological orders of the chains are logical and meaningful. From top to bottom the following topics can be observed: finance/stock market, software engineering, databases/programming languages, computer science, medical sciences/genetics, chemistry/physics and mathematics. These topics are evident from the labels and connections between topics are naturally understandable. Generally, the labelling scheme produces meaningful summaries for clusters. On occasions some labels may not be as specific as handcrafted ones. Using human indexers would rely on their knowledge and experience, sometimes is also prone to errors and can become impractical for categorising millions of unlabelled and unstructured documents. The proposed method provides consistent and systematic labels. An enhancement could be made by the use of external knowledge to infer more general and descriptive words not necessarily in the corpus.

The same dataset A, was organised by the 2-dimensional SOM. The resulting map, shown in Fig. 7, was created by further manually merging the nodes to form regions so that, for display clarity, some of the subtopics were joined (e.g. databases and software engineering into computer science class). The regions have been labelled as computer science, medical sciences, physics, finance, chemistry, and mathematics. Even with a manual segmentation of the map into

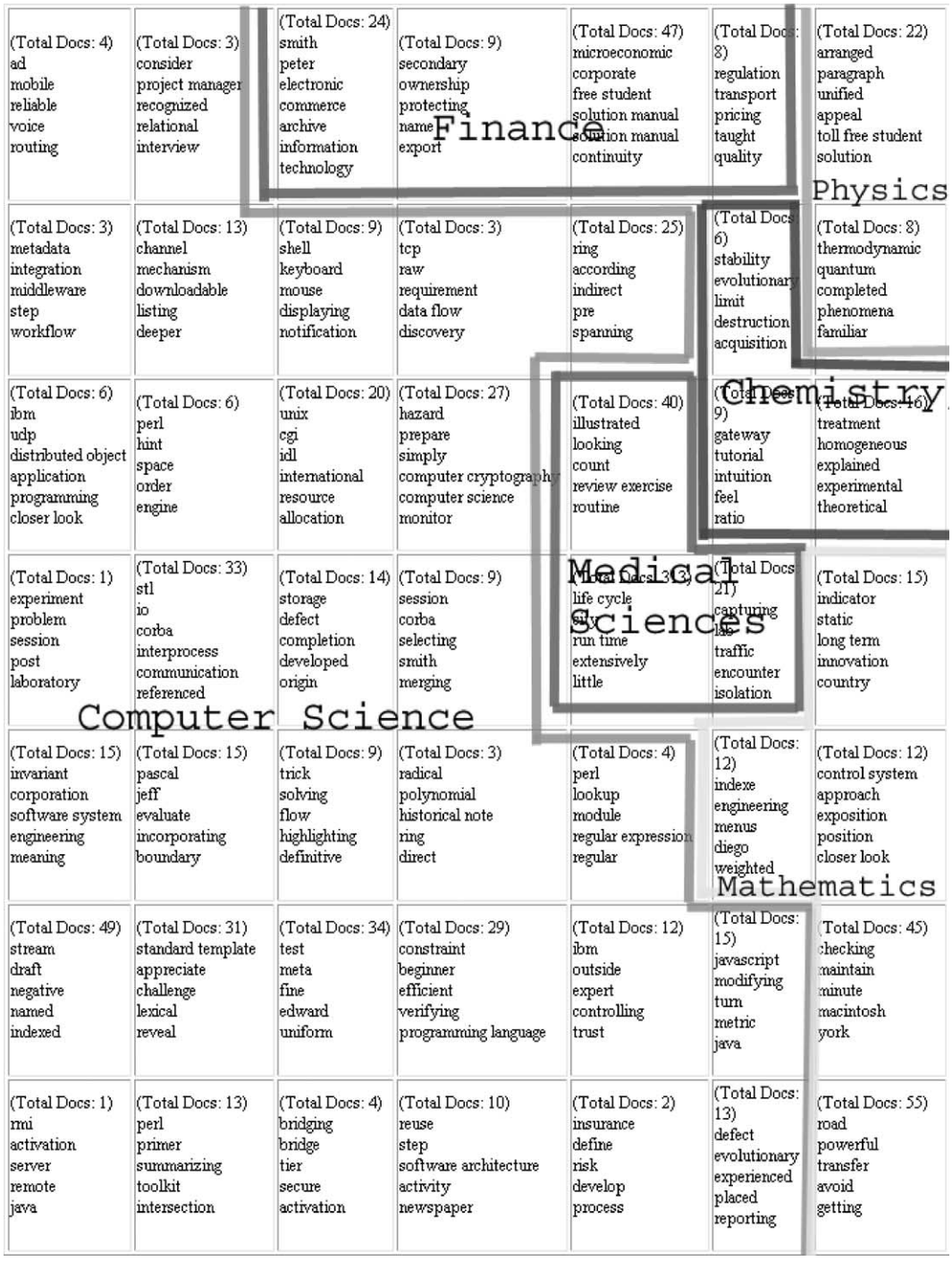

Fig. 7. 7×7 SOM on dataset A with manual segmentations.

clusters it is still difficult to apprehend the sub-topics especially when the numbers of documents in the clusters are large. The SOM algorithm produced a similar clustering result to the ATTS at the first level. Another observation is that the largest group computer science in the map are also given the largest numbers of nodes in the topological tree; this is due to the fact that the chains and maps all approximate the document probability density.

The GH-SOM was also used for generating a hierarchy in the comparison. The results are shown in Fig. 8. The GH-SOM grows maps up to a certain number of nodes at each level. Additionally, it generates a hierarchy of maps that is easier to browse (without recurring to zooming) compared to the single level SOM. For example, Fig. 8 shows one branch from the GH-SOM generated hierarchy. In the root map, the top two rows deal with computer science, middle

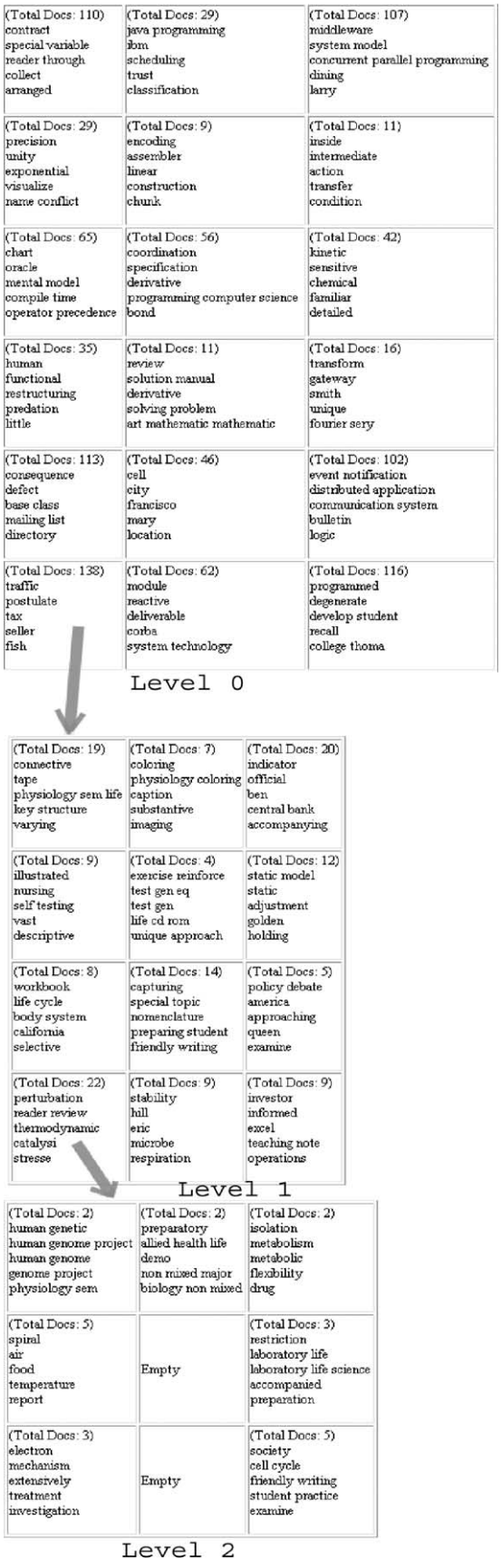

Fig. 8. The top level and a branch in a hierarchy generated by the GH-SOM for dataset A.

right physics/mathematics, bottom left finance/chemistry and bottom right mathematics. The branch expanding into a child map level-1 deals with anatomy (upper section), finance (the right column) and chemistry (lower section). The child map in level-2 is expanding on the topic of organic chemistry to the left and bio/molecular-chemistry to the right.

Although the GH-SOM generates a set of maps showing the similar nodes/clusters close to one another, in some cases the results are not directly useful to a user and are difficult to navigate. For example, when there are many empty nodes, or topics spanning two or more nodes are represented in two or more child maps. A better representation can be created by merging similar clusters to form coloured regions and only use documents of each region in child maps as in the multilayer SOM in the ET-MAP (the limitation being the fixed map sizes). This extra processing step depends on threshold settings or manual segmentation. The ATTS, however, does not require such a step as each node is considered as a cluster and for which internal validation has already been applied. If a large topic happens to be split into two or more nodes, expanding into a child chain, then one can still view all chains through the tree. The topological tree represents the most important and strongest relations amongst topics and these connections are easier to justify and understand, since each node has at most two connections/links, much simpler than the rectangular grid of the GH-SOM. Another drawback of the GH-SOM is its dependence on quantisation error for growth. The wrong choice of its value may result in a flat or deep hierarchy. In the ATTS a validation is used to determine the optimum number of nodes in each chain. This process adds little overhead, since the validation measure is only calculated after each node insertion, rather than at frequent fixed intervals as for the calculation of the quantisation error in the GH-SOM. The topological tree is easy to navigate at various levels simultaneously, not possible with 2-dimensional representations such as the 2-dimensional SOM, GCS or GH-SOM.

The topological tree can be advantageous in retrieving documents given a query in terms of *recall* and *F-measure* compared to the bisecting *k*-means and the 2-dimensional SOMs. The bisecting *k*-means was chosen since it was shown to provide better results than typical hierarchical agglomerative methods (Steinbach et al., 2000). The SOM was compared with the ATTS as it also has a topology and has previously been used for document organisation. Both resulting structures have also been used for retrieval purposes (Georgakis, Kotropoulos, Xafopoulos, & Pitas, 2004; Lagus, 2002; Willett, 1988).

The datasets C and D are subsets of the Reuters-21578[1] newswire articles, typically used in text classification. Documents without labels were removed and documents

[1] http://www.daviddlewis.com/resources/testcollections/reuters21578/.

with more than one topic were assigned to the first topic. The training datasets were then used to train the SOM, ATTS and bisecting *k*-means methods. Then each document in the test datasets were used as a query to search for related documents of the same topic. The *F-measure*, typically used in information retrieval (Baeza-Yates & Ribeiro-Neto, 1999), was adopted to evaluate different methods,

$$F_{\mathrm{m}} = \frac{1+\beta^2}{\beta^2 \frac{1}{R} + \frac{1}{P}} \tag{14}$$

where $P = R_a/B$ is the *precision* and $R = R_a/A$ is the *recall*. R_a is the number of relevant retrieved documents, A is the total number of documents relevant documents in the set and B the total number of documents retrieved from the search. β marks the importance of P and R. Higher importance is generally given to R than P as it is sometimes desirable to traverse a larger relevant document set. Setting β to 1 gives equal importance to both P and R.

A top-down search (Willett, 1988) was used to seek for the related documents. Other strategies are also possible such as bottom-up which begins with examining the leaf nodes (Willett, 1988). For the SOM each document in the test dataset is matched with the most similar nodes. For the ATTS and bisecting *k*-means the search begins with the winning node in the root chain and then proceeds down the hierarchy. The topology of the ATTS helps improve the search, as the selection of relevant clusters will be topologically close to the path from the winning root node to the winning leaf node.

The search results are based on selecting the best matching leaf clusters in each structure. For the SOM and ATTS the best number of leaf clusters is tested from 1 node to 8 nodes. The most similar clusters to query documents are found and all the documents in these cluster are ranked by their relevance. Those falling below a cut-off similarity to query threshold were ignored (e.g. 0.025), as it has been shown to be more efficient than retrieving all the documents from a cluster (Voorhees 85).

For all the experiments the average *F-measure* over all queries is calculated using Eq. (14) with $\beta = 1$. The *F-measure* was chosen as a comparative benchmark as it does not require a strict ranking of the documents. The SOM map size was varied from 2×2 to 5×5 and the maximum average *F-measure* was recorded. For the bisecting *k*-means different dendrogram depths were chosen (5–35) and the maximum average *F-measure* was used. For the ATTS the maximum average *F-measure* was picked.

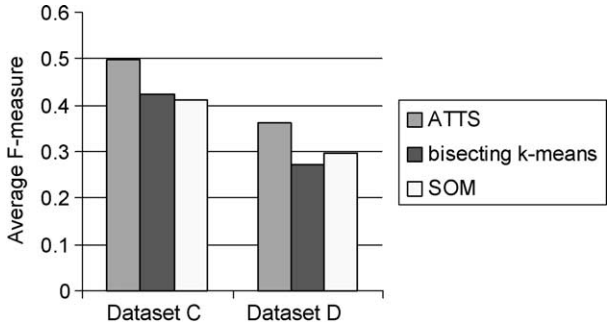

Fig. 9. The average *F-measure* for retrieving most relevant documents from clusters for the ATTS, bisecting *k*-means and SOM.

The evaluation of the search is shown in Fig. 9. For both datasets the ATTS outperforms the bisecting *k*-means and the SOM. For dataset C about half of all the relevant documents are found within 1 or 2 leaf nodes. It has also been observed that, in most cases, documents in the neighbouring nodes in the SOM and ATTS are the most relevant. Further work needs to be done to investigate the relation between topology and relevance. The ATTS estimates the numbers of nodes/clusters using the entropy-BIC, which does not require a probability model and improves the clustering recall thus the *F-measure*. Although there are still arguments on whether to use 1- or 2-dimensional, single level or hierarchical structure, of the SOM, it shows that the hierarchical topological tree structure performs better and is consistent with many existing computer file systems. More research or user feedbacks will be required to make a fuller justification.

5. Conclusions

A topological tree structure has been used to self-organise and present documents and information contents. The results and comparisons show the benefits of the proposed ATTS method over the SOM, GH-SOM and bisecting *k*-means methods for document analysis, organisation and search. The topological tree is advantageous for categorising, summarising, navigating and exploring unstructured documents textual content. It provides abstractions and continuity of the content at various levels and better scalability than the grid based methods. Other benefits are that the multiple levels/-branches can be viewed simultaneously, and topic relations can be easily visualised in a way similar to most file manager systems. The underlying number of topics at each level is automatically chosen and verified by an entropy-based BIC validation procedure. The topological tree structure also improves document browsing and retrieval.

Acknowledgements

The authors are grateful to the reviewers for their helpful comments. The first author is supported by a UK Engineering Physical Sciences Research Council (EPSRC) PhD studentship.

References

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. *Proceedings of the 20th international conference very large data bases*. Los Altos, CA: Morgan Kaufmann pp. 487–499.

Alahakoon, D., Halgamuge, S. K., & Srinivasan, B. (2000). Dynamic self-organizing maps with controlled growth for knowledge discovery. *IEEE Transactions on Neural Networks*, *11*(3), 601–614.

Allan, J., Papka, R., & Lavrenko, V. (1998). On-line new event detection and tracking. *Proceedings of the 21st international conference on research and development in information retrieval (SIGIR'98)*. New York: ACM pp. 37–45.

Azcarraga, A. P., & Yap, T. N. (2001). Extracting meaningful labels for WEBSOM text archives. *Proceedings of the 10th conference on information and knowledge management (CIKM)* pp. 41–48.

Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval*. New York/Reading, MA: ACM Press/Addison-Wesley.

Beil, F., Ester, M., & Xu, X. (2002). Frequent term-based text clustering. *Proceedings of the eighth international conference on knowledge discovery and data mining*. New York: ACM pp. 436–442.

Blackmore, J., & Miikkulainen, R. (1993). Incremental grid growing: encoding high-dimensional structure into a two-dimensional feature map. *Proceedings of the international conference on neural networks (ICNN'93)*, Vol. 1. New York: IEEE pp. 450–455.

Carpenter, G. A., & Grossberg, S. (1988). The ART of adaptive pattern recognition by a self-organizing neural network. *Computer*, *21*(3), 77–88.

Chen, H., Houston, A. L., Sewell, R. R., & Schatz, B. R. (1998). Internet browsing and searching: user evaluations of category map and concept space techniques. *Journal of the American Society for Information Science*, *49*(7), 582–603.

Chen, H., Schuffels, C., & Orwig, R. (1996). Internet categorization and search: a self-organizing approach. *Journal of Visual Communication and Image Representation*, *7*(1), 88–102.

Cutting, D. R., Karger, D. R., Pederson, J. O., & Tukey, J. W. (1992). Scatter/gather: A cluster-based approach to browsing large document collections. *Proceedings of the 15th international conference on research and development in information retrieval*. New York: ACM pp. 318–329.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, *41*(6), 391–407.

Deng, W., & Wu, W. (2001). Document categorization and retrieval using semantic microfeatures and growing cell structures. *Proceedings of the 12th international workshop on database and expert systems applications (DEXA 2001)*. Silver Spring, MD: IEEE Computer Society pp. 270–274.

Dittenbach, M., Rauber, A., & Merkl, D. (2001). Recent advances with the growing hierarchical self-organising map. *Proceedings of the advances in self-organising maps*. Berlin: Springer pp. 140–145.

El-Hamdouchi, A., & Willett, P. (1987). Techniques for the measurement of clustering tendency in document retrieval systems. *Journal of Information Science Principles and Practice*, *13*(6), 361–365.

Freeman, R., & Yin, H. (2002). Self-organising maps for hierarchical tree view document clustering using contextual information. *Proceedings of the intelligent data engineering and automated learning (IDEAL'02)*. Lecture notes in computer science (Vol. 2412) (pp. 123–128). Berlin: Springer.

Freeman, R., Yin, H., & Allinson, N. M. (2002). Self-organising maps for tree view based hierarchical document clustering. *Proceedings of the international joint conference on neural networks (IJCNN'02)*, Vol. 2. New York: IEEE pp. 1906–1911.

Fritzke, B. (1991). *Unsupervised clustering with growing cell structures Proceedings of the international joint conference on neural networks (IJCNN'91)*, Vol. 2. New York: IEEE pp. 531–536.

Fritzke, B. (1993). Kohonen feature maps and growing cell structures—A performance comparison. *Advances in Neural Information Processing Systems*, *5*, 115–122.

Fritzke, B. (1995a). Growing grid-a self-organizing network with constant neighbourhood range and adaptation strength. *Neural Processing Letters*, *2*(5), 9–13.

Fritzke, B. (1995b). A growing neural gas network learns topologies. *Advances in Neural Information Processing Systems*, *7*, 625–632.

Georgakis, A., Kotropoulos, C., Xafopoulos, A., & Pitas, I. (2004). Marginal median SOM for document organization and retrieval. *Neural Networks*, *17*(3), 365–377.

Gunter, S., & Bunke, H. (2002). Self-organizing map for clustering in the graph domain. *Pattern Recognition Letters*, *23*(4), 405–417.

Gunter, S., & Bunke, H. (2003). Validation indices for graph clustering. *Pattern Recognition Letters*, *24*(8), 1107–1113.

He, J., Tan, A.-H., & Tan, C.-L. (2002). ART-C: A neural architecture for self-organization under constraints. *Proceedings of the international joint conference on neural networks (IJCNN'02)*, Vol. 3. New York: IEEE pp. 2550–2555.

Her, J.-H., Jun, S.-H., Choi, J.-H., & Lee, J.-H. (1999). A Bayesian neural network model for dynamic web document clustering. *Proceedings of the IEEE region 10 conference, TENCON 99*, Vol. 2 pp. 1415–1418.

Hipp, J., Guntzer, U., & Nakhaeizadeh, G. (2000). Algorithms for association rule mining—A general survey and comparison. *SIGKDD Explorations*, *2*(1), 58–64.

Hodge, V. J., & Austin, J. (2001). Hierarchical growing cell structures: TreeGCS. *IEEE Transactions on Knowledge and Data Engineering*, *13*(2), 207–218.

Hodge, V. J., & Austin, J. (2002). Hierarchical word clustering—Automatic thesaurus generation. *Neurocomputing*, *48*(1–4), 819–846.

Kaski, S. (1998). Dimensionality reduction by random mapping: fast similarity computation for clustering"." *Proceedings of the international joint conference on neural networks (IJCNN'98)*, Vol. 1. New York: IEEE pp. 413–418.

Kaski, S., Honkela, T., Lagus, K., & Kohonen, T. (1998). WEBSOM-self-organizing maps of document collections. *Neurocomputing*, *21*(1–3), 101–117.

Kohonen, T. (1997). *Self organizing maps* (2nd ed.). Berlin: Springer.

Kohonen, T., Kaski, S., Lagus, K., Salojarvi, J., Honkela, J., Paatero, V., & Saarela, A. (2000). Self organization of a massive document collection. *IEEE Transactions on Neural Networks*, *11*(3), 574–585.

Kondadadi, R., & Kozma, R. (2002). A modified fuzzy ART for soft document clustering. *Proceedings of the international joint conference on neural networks (IJCNN'02)*, Vol. 3. New York: IEEE pp. 2542–2549.

Lagus, K. (2002). Text retrieval using self-organized document maps. *Neural Processing Letters*, *15*(1), 21–29.

Lagus, K., & Kaski, S. (1999). Keyword selection method for characterizing text document maps. *Proceedings of the ninth international conference on artificial neural networks (ICANN99)*, Vol. 1. London: IEE pp. 371–376.

Larsen, B., & Aone, C. (1999). Fast and effective text mining using linear-time document clustering. *Proceedings of the international conference on knowledge discovery and data mining*. New York: ACM pp. 16–22.

Lin, X., Soergel, D., & Marchionini, G. (1991). A self-organizing semantic map for information retrieval. *Proceedings of the 14th forum, special interest group on information retrieval*. New York: ACM pp. 262–269.

Liu, X., Gong, Y., Xu, W., & Zhu, S. (2002). Document clustering with cluster refinement and model selection capabilities. *Proceedings of the 25th international conference on research and development in information retrieval*. New York: ACM pp. 191–198.

MacLeod, K. J., & Robertson, W. (1991). A neural algorithm for document clustering. *Information Processing and Management*, *27*(4), 337–346.

Massey, L. (2003). On the quality of ART1 text clustering. *Neural Networks*, *16*(5/6), 771–778.

Merkl, D. (1995). Content-based software classification by self-organization. *Proceedings of the international conference on neural networks*, Vol. 2. New York: IEEE pp. 1086–1091.

Merkl, D. (1997). Exploration of text collections with hierarchical feature maps. *Proceedings of the 20th forum, special interest group on information retrieval*. New York: ACM pp. 186–195.

Merkl, D. (1998). Text classification with self-organizing maps: Some lessons learned. *Neurocomputing*, *21*(1–3), 61–77.

Merkl, D., & Rauber, A. (2000). Document classification with unsupervised artificial neural networks. In F. Crestani, & G. Pasi (Eds.), *Soft computing in information retrieval* (pp. 102–121). Wurzburg (Wien): Physica-Verlag.

Miikkulainen, R. (1990). Script recognition with hierarchical feature maps. *Connection Science: Journal of Neural Computing, Artificial Intelligence and Cognitive Research*, *2*(1/2), 83–101.

Munoz, A. (1997). Compound key word generation from document databases using a hierarchical clustering ART model. *Intelligent Data Analysis*, *1*(1), 25–48.

Nurnberger, A., & Detyniecki, M. (2002). Visualizing changes in data collections using growing self-organizing maps. *Proceedings of the international joint conference on neural networks (IJCNN'02)*, Vol. 2 pp. 1912–1917.

Ong, H.-L., Tan, A.-H., Ng, J., Pan, H., & Li, Q.-X. (2001). *FOCI:* Flexible organizer for competitive intelligence. *Proceedings of the 10th international conference on information and knowledge management*. New York: ACM pp. 523–525.

Ontrup, J., & Ritter, H. (2001). Text categorization and semantic browsing with self-organizing maps on non-Euclidean spaces. *Proceedings of the fifth European conference on principles of data mining and knowledge discovery (PKDD 2001)*. Lecture Notes in Computer Science (Vol. 2168) (pp. 338–349). Berlin: Springer.

Pullwitt, D. (2002). Integrating contextual information to enhance SOM-based text document clustering. *Neural Networks*, *15*(8–9), 1099–1106.

Rajaraman, K., & Tan, A.-H. (2001). Topic detection, tracking, and trend analysis using self-organizing neural networks. *Proceedings of the fifth Pacific-Asia conference on advances in knowledge discovery and data mining (PAKDD 2001)*, Vol. 2035. Berlin: Springer pp. 102–107.

Rauber, A. (1999). LabelSOM: On the labeling of self-organizing maps. *Proceedings of the international joint conference on neural networks (IJCNN'99)*, Vol. 2. New York: IEEE pp. 3524–3532.

Rauber, A., & Bina, H. (2000). 'Andreas, Rauber? Conference pages are over there, German documents on the lower left…': an 'old-fashioned' approach to Web search results visualization. *Proceedings of the 11th international workshop on database and expert systems applications*. Silver Spring, MD: IEEE Computer Society pp. 615–619.

Rauber, A., Merkl, D., & Dittenbach, M. (2002). The growing hierarchical self-organizing map: Exploratory analysis of high-dimensional data. *IEEE Transactions on Neural Networks*, *13*(6), 1331–1341.

Rauber, A., Pampalk, E., & Paralic, J. (2000). Empirical evaluation of clustering algorithms. *Zbornik Radova, Journal of Information and Organizational Sciences*, *24*(2), 195–209.

Roussinov, D. G., & Chen, H. (1998). A scalable self-organizing map algorithm for textual classification: a neural network approach to thesaurus generation. *Cc-Ai, the Journal for the Integrated Study of Artificial Intelligence, Cognitive Science and Applied Epistemology*, *15*(1/2), 81–111.

Roussinov, D. G., & Chen, H. (2001). Information navigation on the Web by clustering and summarizing query results. *Information Processing and Management*, *37*(6), 789–816.

Salton, G. (1989). *Automatic text processing—The transformation, analysis, and retrieval of information by computer*. Reading, MA: Addison-Wesley.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, *6*(2), 461–464.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, *34*(1), 1–47.

Slonim, N., Friedman, N., & Tishby, N. (2002). Unsupervised document classification using sequential information maximization. *Proceedings of the 25th international conference on research and development in information retrieval (SIGIR '02)*. New York: ACM pp. 129–136.

Steinbach, M., Karypis, G., & Kumar, V. (2000). A comparison of document clustering techniques. *Proceedings of the world text mining conference (KDD2000)*.

Tan, A.-H. (2002). Personalized information management for web intelligence. *Proceedings of the international conference on fuzzy systems (FUZZ'02)*, Vol. 2. New York: IEEE pp. 1045–1050.

Vesanto, J., & Alhoniemi, E. (2000). Clustering of the self-organizing map. *IEEE Transactions on Neural Networks*, *11*(3), 586–600.

Vlajic, N., & Card, H. C. (1998). Categorizing Web pages using modified ART. *Proceedings of the Canadian conference on electrical and computer engineering*, Vol. 1. New York: IEEE pp. 586–600.

Vlajic, N., & Card, H. C. (1999). An adaptive neural network approach to hypertext clustering. *Proceedings of the international joint conference on neural networks (IJCNN'99)*, Vol. 6. New York: IEEE pp. 3722–3726.

Willett, P. (1988). Recent trends in hierarchic document clustering: A critical review. *Information Processing and Management*, *24*(5), 577–597.

Ye, H., & Lo, B. W. N. (2000). A visualised software library: nested self-organising maps for retrieving and browsing reusable software assets. *Neural Computing and Applications*, *9*(4), 266–279.

Ye, H., & Lo, B. W. N. (2001). Towards a self-structuring software library. *IEE Proceedings-Software*, *148*(2), 45–55.