Topological Tree for Web Organisation, Discovery and Exploration

Richard T. Freeman and Hujun Yin

Department of Electrical & Electronics Engineering
University of Manchester Institute of Science and Technology
Manchester, M60 1QD, UK
rics@swift.ee.umist.ac.uk, h.yin@umist.ac.uk
http://www.rfreeman.net

Abstract. In this paper we focus on the organisation of web contents, which allows efficient browsing, searching and discovery. We propose a method that dynamically creates such a structure called *Topological Tree*. The tree is generated using an algorithm called Automated Topological Tree Organiser, which uses a set of hierarchically organised self-organising growing chains. Each chain fully adapts to a specific topic, where its number of subtopics is determined using entropy-based validation and cluster tendency schemes. The Topological Tree adapts to the natural underlying structure at each level in the hierarchy. The topology in the chains also relates close topics together, thus can be exploited to reduce the time needed for search and navigation. This method can be used to generate a web portal or directory where browsing and user comprehension are improved.

1 Introduction

With the increasing demand to improve knowledge transfer, competitiveness and efficiency more information technology services and consultancies are offering enterprise portals, data mining and business intelligence solutions. These technologies are related to content and knowledge management, which are particulary valuable with the drastic and increasing amount of content available on the Internet and corporate Intranets. This may contain valuable employee best practices, knowledge and experiences, which are significant assets to a company especially with increased mobility of employees. There is a demand for efficient searching, exploring and knowledge discovery. The main underlying technologies used for textual content are *text mining* (e.g. [6]), *document classification* (e.g. [13]) and *document clustering* (e.g. [11]). Document clustering does not rely on human labelling, thus can provide an objective, automated hierarchy of topics for organisation, searching and browsing purposes. This is the focus of the paper.

Agglomerative [11] or divisive hierarchical methods (e.g. [14]) are commonly used document clustering algorithms that both typically generate a dendrogram. These can be useful for searching and retrieval [15] but are less useful for exploring and browsing a large number of topics. Other methods such as the self-organising maps (SOMs) have also been used for this purpose. For example, the

WEBSOM generates a fixed size 2-dimensional map where the topic clusters can be visualised and where the topology preservation properties ensure that similar topics are located close to one another [8]. Other SOM variants include the growing hierarchical SOM (GH-SOM) [10] or tree growing cell structures (TreeGCS) [7]. In the GH-SOM a set of 2-dimensional maps are grown dynamically until a fixed threshold is reached, then documents clustered to particular nodes are further clustered in child maps. In the TreeGCS growing cells structures are used to organise the documents. The deletion of nodes then allows the formation of sub-structures, in which the clustered patterns are placed in a dendrogram.

2 Topological Tree Generation

The proposed Automated Topological Tree Organiser (ATTO) uses self-organising chains as building blocks and dynamic hierarchy as structure. The preliminary idea of a 1-dimensional, tree type growing SOM was first explored by the authors for document organisation in [4] and later enhanced using contextual indexing terms [3]. The process of the tree type SOM is to let each chain grow until some terminating measure is reached. However this has to be decided before the training. If the criterion is set too high then the chain becomes too indiscriminative, or too low the chain becomes a flat partition. The early method uses the Euclidean distance, while the cosine correlation metric is arguably more appropriate in a sparse document space [11]. An enhancement was made using the correlation metric and the turning point in the average similarity to find the chain size [5]. Once a chain is trained each node is tested to determine if documents clustered to it should be further clustered in a child chain. This process is similar to divisive clustering except that the number of clusters is not limited or pre-defined.

In the proposed ATTO, the size of each chain is independently determined through a validation process using an entropy-based criterion. Once the size is determined, documents clustered to a node are tested using a cluster tendency method, to determine if they should form a leaf node or to be further expanded into a child chain. The successive addition of child chains effectively forms a taxonomy of topics possessing topology at each level. In addition, methods are used to deal with fast dot-product of sparse vectors, winner search method, and weight update function. Furthermore adaptive node insertion for each chain is performed using both interpolation and extrapolation. The nodes in the tree are also given insightful representative labels using a method taking into account of term frequencies and the node weights. Important relations between terms in the same cluster are also extracted using association rule mining. A general diagram showing the overall system and its features is shown in Fig. 1.

2.1 Document Parsing and Feature Selection

In the pre-processing stage the document files in the dataset are crawled, text enclosed in the title tags are recorded and the html parsed into plain text. The

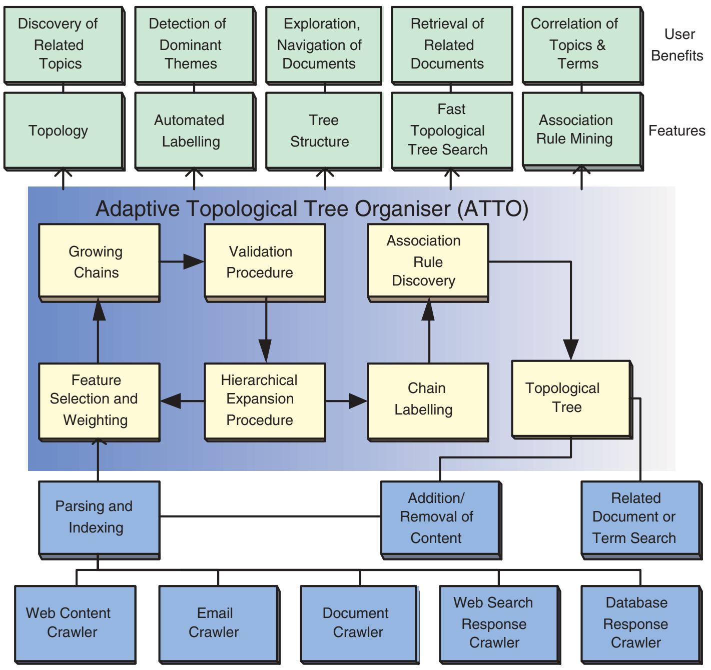

Discovery of Related Topics
Detection of Dominant Themes
Exploration, Navigation of Documents
Retrieval of Related Documents
Correlation of Topics & Terms
User Benefits
Topology
Automated Labelling
Tree Structure
Fast Topological Tree Search
Association Rule Mining
Features
Adaptive Topological Tree Organiser (ATTO)
Growing Chains
Validation Procedure
Association Rule Discovery
Feature Selection and Weighting
Hierarchical Expansion Procedure
Chain Labelling
Topological Tree
Parsing and Indexing
Addition/ Removal of Content
Related Document or Term Search
Web Content Crawler
Email Crawler
Document Crawler
Web Search Response Crawler
Database Response Crawler

Fig. 1. The overall Topological Tree generation

title and body text are then indexed using a sliding window of sizes p (e.g. $p = 4$, up to 4 successive words are used for each term) over sentences, ignoring common words in a stop-list. Since the number of terms can grow significantly, the least discriminative terms are discarded: those occurring in less than a few documents and those occurring in more than a large percentage of the total documents. In the vector space model [11] a document is stored as $[f_1, f_2, ..., f_N]$, where f_i is the frequency of term i and N is the total number of unique terms. Since a document vector is sparse, its term frequencies can be stored in a compressed representation as a hash table for efficient retrieval,

$$\mathbf{x} = [\{\gamma_1, \rho_1\}, \{\gamma_2, \rho_2\}, ..., \{\gamma_r, \rho_r\}], \gamma_k \in \{V | f_{\gamma_k} \neq 0\}, k = 1, 2, ...r \qquad (1)$$

where $\mathbf{x}$ is a document vector stored as term γ and weighted frequency ρ pairs, γ_k refers to the γ_k-*th* term in the vocabulary V, r is the total number of terms in the document. Term weighting ρ and feature selection are detailed in [5].

2.2 Growing Chains

The ATTO uses a set of chains adaptively developed by a growing chains (GC) algorithm. Each chain begins with two nodes and grows until the termination process stops it. In the root chain the weights of the nodes are initialised to small random values, while child chains are initialised through interpolating its parent nodes. There are two important steps in the GC training: the search for the best matching unit and the update of the weights of the winner and its neighbourhood. At time t, an input document vector $\mathbf{x}(t)$ is mapped to a chain consisting of n nodes, where each node j has a weight vector $\mathbf{w}_j = [w_{j1}, w_{j2}, ..., w_{jN}]^T$ of N dimensions. The similarity metric can be calculated efficiently by exploiting the sparseness of $\mathbf{x}(t)$,

$$S_{dot}(\mathbf{x}(t), \mathbf{w}_j) = \sum_{k=1}^{r(t)} \rho_k \cdot w_{j\gamma_k}, \forall \{\gamma_k, \rho_k\} \in \mathbf{x}(t) \tag{2}$$

The best matching unit $c(\mathbf{x})$ is the node with maximum S_{dot} amongst all nodes n with respect to a document vector $\mathbf{x}(t)$,

$$c(\mathbf{x}) = \arg\max_j \{S_{dot}(\mathbf{x}(t), \mathbf{w}_j)\}, j = 1, 2, ..., n \tag{3}$$

The weights of the winner node and of its neighbourhood are updated using,

$$\mathbf{w}_j(t+1) = \frac{\mathbf{w}_j(t) + \alpha(t) h_{j,c(x)}(t)\mathbf{x}(t)}{\left\|\mathbf{w}_j(t) + \alpha(t) h_{j,c(x)}(t)\mathbf{x}(t)\right\|} \tag{4}$$

where α is the learning rate. To monitor the activity of each node a counter is used to record its winning times, which is set to zero at the creation of the chain and reinitialised after a new node is added. A new node is inserted once the training has converged, indicated by the stabilisation of a measure measure termed average similarity ($AvgSim$),

$$AvgSim = \frac{1}{n} \sum_{j=1}^{n} \frac{1}{m_j} \sum_{\forall \mathbf{x} \in C_j} (S_{dot}(\mathbf{x}, \mathbf{w}_j)), m_j \neq 0 \tag{5}$$

where m_j is the number of documents clustered in a particular node C_j by Eq. 3. Just before a new node is inserted an entropy-based validation measure is recorded. This measure is chosen over a weight-based measure, since this is already given by $AvgSim$ and optimised by the GCs. To discourage the tree from becoming a flat partition, the validation measure penalises larger number of clusters, as in the Schwarz's Bayesian information criterion [12]. The validation is also applied locally to a particular chain, taking into account its particular reduced and specialised vocabulary.

$$\Theta(n) = \frac{1}{m} \sum_{j=1}^{n} m_j \cdot \left(-\sum_i p(t_i|C_j) \log p(t_i|C_j) \right) + \frac{1}{2} n \log m \tag{6}$$

where $p(t_i|C_j)$ is the probability that a term t_i belongs to cluster C_j and m the total number of documents.

Once the validation measure is recorded, 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*. The new node's weight is initialised through interpolating or extrapolating the neighbouring nodes.

Once the map has reached a size of n_{max} the training is terminated and the optimum chain size τ is determined using,

$$\tau = \arg\min_{n} \left\{ \Theta(n) \right\}, n = 2, 3, .., n_{\max} \tag{7}$$

2.3 Creating the Topological Tree

After the root chain has been trained and optimum number of nodes τ identified, each node is tested to see whether documents clustered to it need a further sub-level division. 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. If the chain only contains few terms this indicates that the parent node is sufficiently specialised and the documents may already be very similar. 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 while further clustering the set [2]. If any of these tests fail then the parent node becomes a leaf node.

If a node spans a child chain, then all documents mapped to the node form a new subset of documents with reduced vocabulary. The weights of these nodes are initialised using the weights of the parent nodes to speed up the convergence.

2.4 Node Labelling and Descriptors

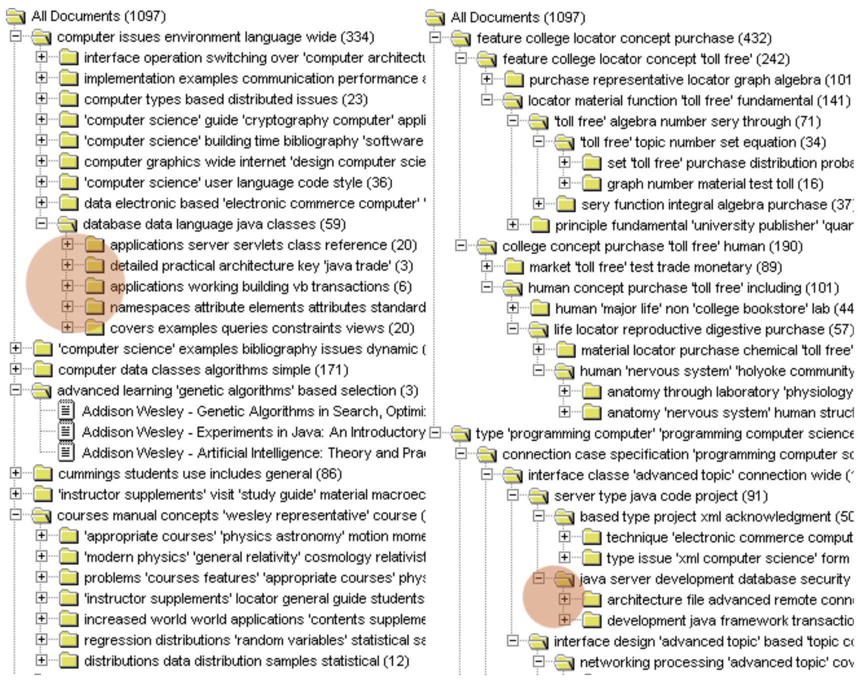
The browsing experience of the Topological Trees is enhanced through an automated selection of suitable labels for each node. They can be found by calculating the dot-product ts_i and frequency of occurrence f_i of each term i for all documents in a cluster. Simultaneously ranking f_i and ts_i (as primary key) provided understandable and high quality cluster labels.

Further details of a particular cluster can be displayed as well as the term associations for its documents. These are found using techniques from association rule mining, similar to those used in clustering frequent term sets [1]. When a node is selected, a ranked list of unidirectional co-occurring relations between two terms or more is displayed.

3 Experiments and Comparisons

The bisecting k-means method [14] was used as a basis for empirical comparison with the ATTO. A moderate dataset of 1097 randomly selected documents from the Addison Wesley website[1] was chosen to allow human verification of clustering results. To reduce excessive depth of the bisecting k-means dendrogram all excessive connections without a divisive split were removed.

[1] http://www.aw.com



```
All Documents (1097)
  computer issues environment language wide (334)
    interface operation switching over 'computer architectu
    implementation examples communication performance a
    computer types based distributed issues (23)
    'computer science' guide 'cryptography computer' appli
    'computer science' building time bibliography 'software
    computer graphics wide internet 'design computer scie
    'computer science' user language code style (36)
    data electronic based 'electronic commerce computer' '
    database data language java classes (59)
      applications server servlets class reference (20)
      detailed practical architecture key 'java trade' (3)
      applications working building vb transactions (6)
      namespaces attribute elements attributes standard
      covers examples queries constraints views (20)
  'computer science' examples bibliography issues dynamic (
  computer data classes algorithms simple (171)
  advanced learning 'genetic algorithms' based selection (3)
    Addison Wesley - Genetic Algorithms in Search, Optimi:
    Addison Wesley - Experiments in Java: An Introductory
    Addison Wesley - Artificial Intelligence: Theory and Pra
  cummings students use includes general (86)
  'instructor supplements' visit 'study guide' material macroec
  courses manual concepts 'wesley representative' course (
    'appropriate courses' 'physics astronomy' motion mome
    'modern physics' 'general relativity' cosmology relativist
    problems 'courses features' 'appropriate courses' phys
    'instructor supplements' locator general guide students
    increased world world applications 'contents suppleme
    regression distributions 'random variables' statistical se
    distributions data distribution samples statistical (12)
```

```
All Documents (1097)
  feature college locator concept purchase (432)
    feature college locator concept 'toll free' (242)
      purchase representative locator graph algebra (101
      locator material function 'toll free' fundamental (141)
        'toll free' algebra number sery through (71)
          'toll free' topic number set equation (34)
            set 'toll free' purchase distribution proba
            graph number material test toll (16)
          sery function integral algebra purchase (37
        principle fundamental 'university publisher' 'quar
    college concept purchase 'toll free' human (190)
      market 'toll free' test trade monetary (89)
      human concept purchase 'toll free' including (101)
        human 'major life' non 'college bookstore' lab (44
        life locator reproductive digestive purchase (57)
          material locator purchase chemical 'toll free'
          human 'nervous system' 'holyoke community
            anatomy through laboratory 'physiology
            anatomy 'nervous system' human struct
  type 'programming computer' 'programming computer science
    connection case specification 'programming computer sc
      interface classe 'advanced topic' connection wide ('
        server type java code project (91)
          based type project xml acknowledgment (5C
            technique 'electronic commerce comput
            type issue 'xml computer science' form
          java server development database security
            architecture file advanced remote conn
            development java framework transactio
        interface design 'advanced topic' based 'topic co
          networking processing 'advanced topic' cov
```

Fig. 2. ATTO Topological Tree (left) and bisecting k-means binary tree (right)

The left of Fig. 2 shows the Topological Tree generated by the ATTO. From top to bottom the following themes can be observed: computer languages, computer development, computer algorithms, life sciences, economics and mathematics. The right of Fig. 2 shows the binary tree generated by the bisecting k-means. The upper levels deals with mathematics/life sciences and lower section with computer science. The two highlighted regions compare both approaches on the topic of databases and it can be observed that the Topological Tree spreads the topic further at a shallower stage than the bisecting k-means. The Topological Tree is clearly less complex, so avoiding deeper divisions aiding visualisation, exploration and understanding. The topology also ensures that similar topics can easily be found in neighbouring nodes. This further allows quick retrieval of documents given a query by allowing expansion of the retrieved documents or search for related topics.

4 Discussions and Conclusions

We have presented a self-organising method for generating a Topological Tree, where the size of each level is determined through validation. Compared to the existing SOM-based methods the proposed Topological Tree is adaptive, dynamically validated and is natural for visualisation and exploring as in many file managers. The topology provides a unique feature that can be used for finding related topics during browsing and extending the search space, a clear distinctive advantage over other clustering algorithms. The topology is obtained without any further processing like reordering the leaf nodes [9] or making use of more advanced representations like graph structures. Topological trees can be defined as a hybrid graph-tree, where the hierarchical relations are represented between clusters.

References

1. F. Beil, M. Ester, and X. Xu. Frequent term-based text clustering. In *Proc. SIGKDD'02*, pages 436–442, Edmonton, Canada, 2002.
2. A. El-Hamdouchi and P. Willett. Techniques for the measurement of clustering tendency in document retrieval systems. *Journal of Information Science*, 13(6):361–365, 1987.
3. R. Freeman and H. Yin. Self-organising maps for hierarchical tree view document clustering using contextual information. In H. Yin, N. Allinson, R. Freeman, J. Keane, and S. Hubbard, editors, *LNCS 2412, Intelligent Data Engineering and Automated Learning*, pages 123–128, Manchester, UK, 2002.
4. R. Freeman, H. Yin, and N. M. Allinson. Self-organising maps for tree view based hierarchical document clustering. In *Proc. IJCNN'02*, volume 2, pages 1906–1911, Honolulu, Hawaii, 2002. IEEE.
5. R. T. Freeman and H. Yin. Tree view self-organisation of web content. *Neurocomputing*, in press 2004.
6. M. A. Hearst. Untangling text data mining. In *Proc. ACL'99*, 1999.
7. V. J. Hodge and J. Austin. Hierarchical growing cell structures: Treegcs. *IEEE Trans. Knowledge & Data Engineering*, 13(2):207–18, 2001.
8. T. Kohonen, S. Kaski, K. Lagus, J. Salojarvi, J. Honkela, V. Paatero, and A. Saarela. Self organization of a massive document collection. *IEEE Trans. Neural Networks*, 11(3):574–85, 2000.
9. S. A. Morris, B. Asnake, and G. G. Yen. Dendrogram seriation using simulated annealing. *Information Visualization*, 2(2):95–104, 2003.
10. A. Rauber, D. Merkl, and M. Dittenbach. The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data. *IEEE Trans. Neural Networks*, 13(6):1331–1341, 2002.
11. G. Salton. *Automatic text processing - the transformation, analysis, and retrieval of information by computer.* Addison-Wesley, 1989.
12. G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.
13. F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.
14. M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *Proc. KDD'2000, Boston, USA*, 2000.
15. C. J. Van Rijsbergen. *Information Retrieval.* Butterworth, 2 edition, 1979.