Minimum Spanning Trees Displaying Semantic Similarity

Włodzisław Duch[1,2] and Paweł Matykiewicz[1,3]

[1] Department of Informatics, Nicholaus Copernicus University, Toruń, Poland
[2] School of Computer Engineering, Nanyang Technological University, Singapore
[3] Department of Biomedical Informatics, Children's Hospital Research Foundation, Cincinnati, Ohio, USA
Google: Duch; emailpawelm@phys.uni.torun.pl

Abstract. Similarity of semantic content of web pages is displayed using interactive graphs presenting fragments of minimum spanning trees. Homepages of people are analyzed, parsed into XML documents and visualized using TouchGraph LinkBrowser, displaying clusters of people that share common interest. The structure of these graphs is strongly affected by selection of information used to calculate similarity. Influence of simple selection and Latent Semantic Analysis (LSA) on structures of such graphs is analyzed. Homepages and lists of publications are converted to a word frequency vector, filtered, weighted and similarity matrix between normalized vectors is used to create separate minimum sub-trees showing clustering of people's interest. Results show that in this application simple selection of important keywords is as good as LSA but with much lower algorithmic complexity.

1 Introduction

Maps presenting connections, associations, clustering or similarity are a popular way to present information. Google knowledge management directory contains links to dozens projects on "mind maps", graphs that map associations between concepts and facilitate thought processes [4]. Mind maps are inspired by the associative nature of human memory and thinking. Projects in this area are focused on graphical representations and manual entry of data. Graphs presenting connectivity of web pages are the simplest to create. A commercial interface called "The Brain" [3] displays graphs similar to mind-maps helping to organize the web links, local files and user notes.

Automatic construction of mind maps containing semantic relations between text documents such as Web pages is more difficult. Various document clusterization and visualization methods may be used to display relations and similarity of text documents. For example, hierarchical SOM maps displaying document clusterization, called WebSOM [6], have been used in a number of applications. Presenting similarities between text objects (web pages or documents) on a graph is a useful technique allowing for a quick overview of a large amount of information.

In large organizations, such as universities or big companies, it is quite difficult to learn who works on similar projects. Analyzing people's homepages and their lists of publications is a good way to find groups and individuals who share common interest. Our work has been motivated by a practical need to bring people with similar interest in contact by showing them a map that links them to other people. We have performed a number of experiments collecting data from people's webpages and investigating the influence of keyword selection and dimensionality reduction via latent semantic analysis on the accuracy of clusterization and the structure of resulting graphs. In the next section algorithms used to derive similarity between people's interests are presented, and in the third section difficulties encountered in this type of application and results of computational experiments are described.

2 Algorithms

Conversion of HTML web pages to pure text requires removing all HTML tags, Javascript programs and comments. `HTML::Parser` [1] written in Perl is used for extracting text that is displayed by web browsers. After converting n documents the text is broken into terms and all terms from a stop-word list are removed. Then Porter's algorithm [12] is used for stemming, counting occurrence of every stemmed term in each document. Results are collected in a matrix $\mathbf{F}$ containing n vector columns, each containing frequency of m stemmed words in a given document. Finally term weighting is applied to create the final term-document matrix.

Let f_{ij} be the number of occurrences of a term j in a document i. Normalized term frequency tf_{ij} is obtained by dividing each column of the $\mathbf{F}$ matrix by the maximum element in that column:

$$tf_{ij} = \frac{f_{ij}}{\max_i(f_{ij})} \tag{1}$$

Inverse document frequency idf_j is a measure of uniqueness of term j. Let n be the number of all documents and d_j a number of documents in which term j occurs, then:

$$idf_j = \log_2\left(\frac{n}{d_j}\right) + 1 \tag{2}$$

The weight w_{ij} of a term i in a document j is defined by:

$$w_{ij} = tf_{ij} \cdot idf_j = \frac{f_{ij}}{\max_i(f_{ij})}\left(\log_2\left(\frac{n}{d_j}\right) + 1\right) \tag{3}$$

Relations between documents may be derived and visualized calculating similarity between columns of the $\mathbf{W} = (w_{ij})_{m\times n}$ matrix. This weight matrix is usually quite large, with many terms, therefore calculation of similarity between documents is influenced by many small contributions from accidental terms. Two dimensionality reduction techniques have been used to avoid it, simple term selection and latent semantic analysis.

2.1 Simple selection of terms

In the m-dimensional space of keywords a hypercube $\mathcal{H} = [0,1]^m$ may be defined. Documents correspond to vectors $\mathbf{W}_j = (w_{ij})_{m\times 1}$ pointing towards some vertices. The direction of this vector should be defined by the largest components of $\mathbf{W}$. Mapping of $\mathbf{W}_j \in \mathcal{R}^m$ vectors to $\mathbf{H}_j \in \mathcal{H}$ vectors that point to the vertex of that hypercube, e.g. $\mathbf{H}_j = (1,0,1,0,...)$, should preserve all important keywords. The average weight value for document j is given by the expectation $\mathbf{W}_j$ over non-zero weights:

$$\theta_j = E\left(\mathbf{W}_j | w_{ij} > 0\right) \tag{4}$$

Using the step function Θ with average as the bias:

$$h_{ij} = \Theta(w_{ij} - \theta_j), \tag{5}$$

a binary $\mathbf{H}_j = (h_{ij})_{m\times 1}$ matrix is produced; rows that contain only zeros are removed, reducing its dimensionality to m'. This matrix is used instead of the $\mathbf{W}$ matrix to calculate similarity between documents:

$$s^h_{ij} = \cos(\overrightarrow{H_j}, \overrightarrow{H_k}) = \frac{\sum_{i=1}^{m'} h_{ij} h_{ik}}{\sqrt{\sum_{i=1}^{m'} h_{ij}^2}\sqrt{\sum_{i=1}^{m'} h_{ik}^2}} \tag{6}$$

The set of all cosines defines a symmetrical similarity matrix between documents $\mathbf{S}^h = (s^h_{ij})_{n\times n}$, where every element s^h_{ij} reflects the degree of semantic similarity between documents i and j in the reduced space. Thresholds θ_j may also be treated as adaptive parameters and optimized using some cost function, but for our purpose simple selection is sufficient.

2.2 Latent Semantic Analysis

Classical LSA algorithm [8] is often used to improve document similarity estimations. Singular value decomposition (SVD) is a robust way of dimensionality reduction. In this algorithm matrix $\mathbf{W}$ is presented as:

$$\mathbf{W} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T \tag{7}$$

where $\mathbf{U}$ is a matrix with eigenvectors of a $\mathbf{W}\mathbf{W}^T$ matrix, representing the original row entries as vectors of derived orthogonal factor values, $\mathbf{V}$ is a matrix with eigenvectors of the transposed $\mathbf{W}^T\mathbf{W}$ matrix, representing the original column entries in the same way, and $\mathbf{\Lambda}$ is a matrix with diagonal elements equal to eigenvalues of the $\mathbf{W}^T\mathbf{W}$ matrix, acting as a scaling coefficients. Reduction of dimensionality is done by zeroing small eigenvalues, creating reduced $\overline{\mathbf{\Lambda}}$ matrix:

$$\overline{\mathbf{W}} = \mathbf{U}\overline{\mathbf{\Lambda}}\mathbf{V}^T \tag{8}$$

$\overline{\mathbf{W}}$ is the best reconstruction of the original matrix that may be obtained from the information in the reduced r-dimensional space, where r, called the rank of the matrix $\mathbf{W}$, is the number of non-zero elements in the $\overline{\mathbf{\Lambda}}$ matrix. Removing eigenvectors corresponding to small eigenvalues leads to reconstruction that is largely noise-free, capturing important regularities of the word distribution. The $\overline{\mathbf{W}}$ matrix is used to calculate similarity between documents corresponding to $\overline{\mathbf{W}}_j$ columns. Similarity matrix $\overline{\mathbf{S}}$ with elements $\overline{s}_{ij} = \overline{\mathbf{W}}_i \cdot \overline{\mathbf{W}}_j / \|\overline{\mathbf{W}}_i\| \|\overline{\mathbf{W}}_j\|$ captures semantical relationships between documents in the collection.

2.3 Minimum Sub-Trees Clusterization

A graph algorithm is used to visualize document clusters. First the similarity matrices are replaced by dissimilarities $d_{ij} = 1 - s_{ij}$ for both $\mathbf{S}^h$ and $\overline{\mathbf{S}}$ matrices. Because $\mathbf{D}^h$ and $\overline{\mathbf{D}}$ matrices are symmetrical they may represent weights (arcs) in a fully connected graph. A modified Kruskal minimum spanning tree (MST) algorithm [7] is used for finding a collection of *minimum sub-trees* that represent document clusters. Such collection of trees is a decomposition of a *minimum spanning tree*. Connecting these minimum sub-trees to their nearest sub-trees via the shortest arc a minimum spanning tree is obtained.

The original Kruskal algorithm in application to our problem creates a single MST tree containing all documents, independent of the number of documents. Kruskal algorithm first sorts arcs, then marks the shortest arcs blue, avoiding cycles by marking other arcs as red. Blue arcs form the MST. The algorithm considers arcs in ascending order of their weights – if both endpoints of an arc are in *the same blue subtree*, then the arc becomes red, else it is blue.

In order to have a number of blue sub-trees depend on relations between documents a modification of the original Kruskal's algorithm is proposed. Marking of the blue arcs that have endpoints in different blue trees are preserved, obtaining separate blue trees. A minimum sub-tree clusterization algorithm considers arcs in the ascending order of their weights – if both endpoints of an arc are in *a blue tree*, color it red, else color it blue. This modification does not allow to form not only cycles but also to merge different minimum trees. The number of such trees p depends on the weight matrix and the number of documents.

Every minimum sub-tree can be considered as a $\tilde{C}_i$ cluster for a subset of documents. Such a cluster holds documents that have similar semantic representation and the number of documents cannot be lower then 2. Further the set of all clusters $\tilde{C} = \tilde{C}_1 \cup \tilde{C}_2 \cup \ldots \cup \tilde{C}_p$ containing all documents will be used for evaluation of the accuracy of visual representation obtained from algorithms used. It should be mentioned that minimum sub-tree clusterization algorithm has the same low computational cost that Kruskal algorithm that is $O(n \log n)$ for sorting and $O(n \log n)$ for processing the edges.

The accuracy of this algorithm in tests presented below was higher than the accuracy of threshold-based MST decomposition.

3 Testing accuracy on Reuter's data

To estimate the accuracy of clusterization provided by the sub-trees experiments with Reuters-21578 [9] datasets were conducted. The original format of these documents is SGML, therefore conversion and some preprocessing was performed. The title and the text body for documents containing more then 600 bytes and a single, unique label were used. These labels t_j were employed to group documents into topics T_j, that is sets containing documents with the same label. The minimum sub-tree cluster $\tilde{C}_i$ is assigned to a topic that the majority of documents in this cluster belong to. For every cluster $\tilde{C}_i \in \tilde{C}$ with $|\tilde{C}_i|$ elements, $n(\tilde{C}_i)$ documents belong to the topic that the cluster is assigned to. The number of clusters is typically larger than the number of topics, therefore the same topic may be assigned to several clusters. Accuracy is measured by summing $n(\tilde{C}_i)$ over all clusters and dividing by the number of documents, $A = \sum_i n(\tilde{C}_i)/n$.

Two tests are reported here. First 600 documents that passed through the pre-processing were used to create $\mathbf{W}$ matrix. The number of documents in 41 topics ranged from 176 to 1, so perfect clusterization is not possible (our clusters have at least 2 elements). Moreover, SVD revealed rank($\mathbf{W}$) = 595. The accuracy was evaluated without dimensionality reduction, with selection and with LSA with the number of eigenvectors equal to 0.8 and 0.6 times the rank of the original matrix (Table 1). The number of clusters for each case was similar, and the accuracy did not differ much, with simple selection achieving slightly better results at much lower computational cost comparing to LSA.

Table 1. Results of parsing first 600 documents, 41 topics.

Method	♯ topics	♯ clusters	accuracy
No dim red.	41	129	0.782
LSA dim red. 0.8 (476)	41	124	0.762
LSA dim red. 0.6 (357)	41	127	0.752
Simple Selection	41	130	0.785

An easier test, with the first 600 documents selected from 10 topics: *acq, coffee, crude, earn, GNP, interest, money-fx, ship, sugar, trade*, and 60 documents in each topic, is summarized in Table 2. Accuracy is slightly higher, with small differences in the number of clusters and simple selection again giving somewhat improved results.

LSA has slightly reduced the number of clusters in both experiments. Although more sophisticated clusterization methods may improve these results

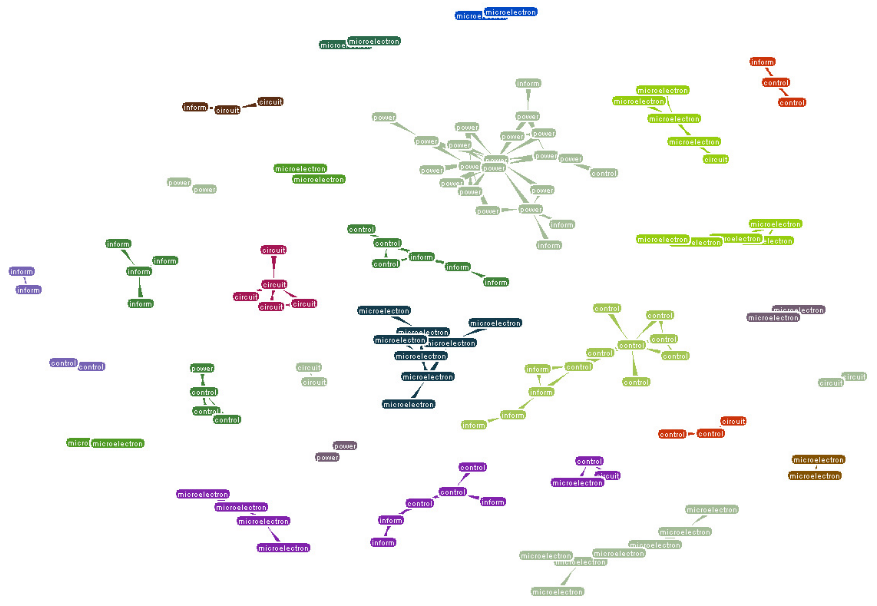

Fig. 1. TouchGraph LinkBrowser screenshot of 124 homepages from the EEE School of the Nanyang Technological University, Singapore, without dimensionality reductions.

for our purpose the simple selection approach, fully automated, without any adaptive parameters, seems to be sufficiently accurate.

Table 2. Results of parsing first 600 documents with selected topics.

Method	♯ topics	♯ clusters	accuracy
No dim red.	10	142	0.847
LSA dim red. 0.8 (467)	10	129	0.847
LSA dim red. 0.6 (350)	10	137	0.828
Simple Selection	10	145	0.855

4 Real application

Our main motivation was to discover groups of experts sharing common interest in large institutions, such as universities. 124 web pages of the School of Electrical and Electronic Engineering (EEE) of the Nanyang Technological University, Singapore, were used to create the weight matrix **W**. Before applying algorithms described above additional preprocessing was necessary. Only the pages that were at least 600 bytes long and contained a regular expression with the name of a *division, school* or an *institute*were used. These

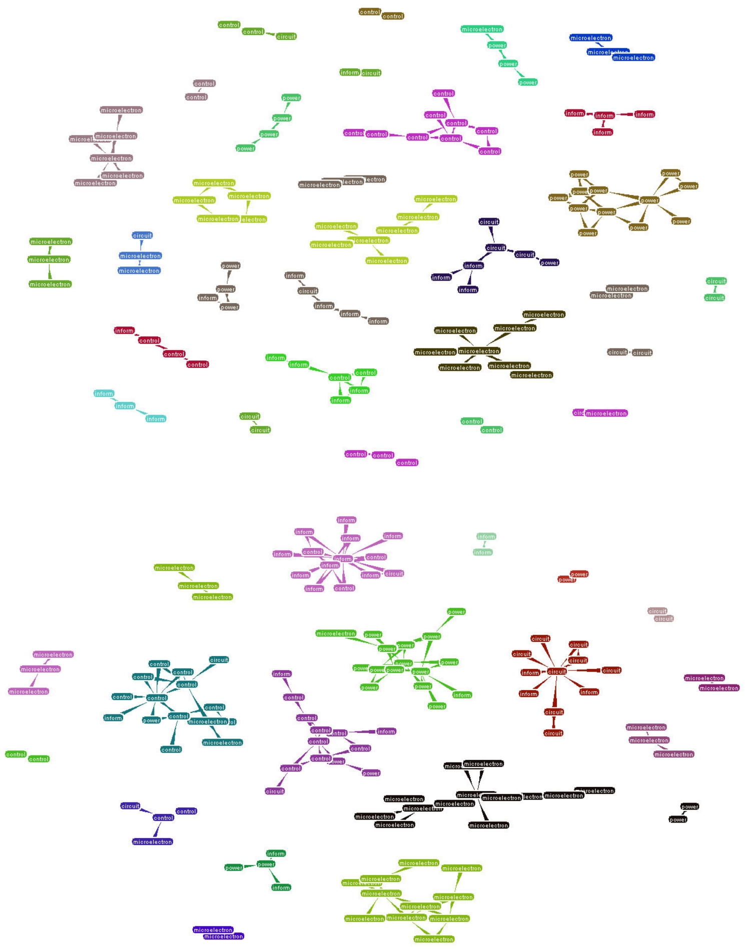

Fig. 2. Same webpages after selection (left) and LSA (right) with reduction factor 0.8.

names were used as topic labels for the Web pages. 5 topics were found, *microelectronics, information, circuit, power, control*, with the number of pages ranging from 14 to 41. The terms were cleaned using a stop-word list consisting of standard words (*and, the, for* ...) extended by words that occurred only once or were present on all pages.

Figures 1 and 2 show screenshots of the TouchGraph LinkBrowser [2] applet with graphical representation of the similarity information (stored in the XML format). Every cluster (minimum sub-tree) was colored for better visualization. Moreover, only first one or two shortest edges inside every cluster are presented. Figure 1 shows similarity between homepages with dimensionality reduction. Clustering of experts into different research groups is clearly visible. Clicking on an individual label centers the graph on a given person showing links to people with highly overlapping interest [10]. Graphs show names and divisions of people, and clicking on them leads to their web pages.

There is no guarantee that someone working in the division of microelectronics does not belong to a cluster dominated by control people, therefore accuracy in Table 3 is only approximate. Figures 1, 2 suggest that simple selection is a good approach for that kind of clusterization.

Table 3. Accuracy of clustering personal home shown in Fig. 1

Method	♯ topics	♯ clusters	*accuracy*
no dim red.	5	28	0.831
LSA dim red. 0.8 (98)	5	19	0.815
LSA dim red. 0.6 (74)	5	22	0.766
Simple Selection	5	30	0.887

5 Discussion and conclusions

This paper has been driven by a practical application, visualization of shared interest based on similarity analysis of homepages and publication records. A modified minimum spanning sub-trees were used to present relations among different interest groups. It has been found that in this case simple selection of important keywords improves accuracy of clustering and thus the structure of semantic connectivity graphs more than latent semantic analysis LSA. There are many knowledge-based clustering methods [11] that could also be used but problems encountered in practical applications cannot be solved by modification of clustering methods.

Homepages frequently contain meaningless titles, and may contain a lot of irrelevant information. Bibliographies are very useful source of important keywords but on some pages they are generated on demand and their retrieval requires sophisticated agents gathering information. They also contain many names, acronyms and abbreviations that may not be easy to map uniquely to full journal titles. Publishing in the same journals is rather weak indication of similarity of interest. To determine how relevant a given word found in a web page is *a priori* inverse document frequencies may be calculated using some large corpus. Most words in the list of a top few thousands highest frequency words are not useful to determine similarity of expert's interest. Instead of a stop-list, a go-list could be compiled in some applications, including only specialized keywords that are found in medical subject headings, science abstracts, or hierarchical classification schemes used by libraries identify different fields of science. Using a specialized corpus of texts that belong to a general category would be helpful in determination of *a priori* weights. Selection of relevant information for clustering is a real challenge.

Statistical approach to similarity should benefit from knowledge-based analysis based on an attempt to understand some information contained in personal pages. Such knowledge based clustering should include synonyms, noun phrases, and similarity between concepts. This may be realized by using concept vectors $\mathbf{V}_i$ instead of terms i, and modifying frequencies $f_{ij} = \sum_k S(\mathbf{V}_i, \mathbf{V}_k)$ by summing over all concepts k in the document j, where $S(\mathbf{V}_i, \mathbf{V}_k)$ will be 1 for concepts that are identical or synonymous, and 0 for concepts that are significantly different. Some users may be interested in finding a very specific interest group, not just a group that shares their general interest. For example, the fact that people are members of the same organization, have obtained their degrees from the same university, have links to the same Web pages, may all be of interest to the user. This will require addition of inference mechanisms.

Minimum spanning trees may not be the best way to display similarity information and we have also experimented with multidimensional scaling algorithms [5]. Depending on the actual objective functions global or local structure may be preserved in a better way in MDS maps. While MST graphs may show several unrelated clusters MDS may reveal that some members of a cluster are in fact similar to those of other clusters.

Software for creation of graphs displaying homepages of people sharing common interest may have numerous applications. Obviously it should be coupled with a web crawler that visits sites within some domain, checks if they have characteristics of homepages, follows links to separate pages containing bibliographies and retrieves them. Graphs may be created from different perspectives, displaying homepages of people that share particular interest. For example, asking for all people interested in “neural networks for control” a virtual group of experts that work on this topic in some region or large institution may be created. Trying to derive the same information from search engines proved to be quite difficult. We have identified the main problems and made the first steps towards practical application but the algorithms presented here may still be improved in many ways.

References

1. G. Aas. Html-parser. http://search.cpan.org/~gaas/HTML-Parser, 2004.
2. G. Aas. Html-parser. http://www.touchgraph.com, 2004.
3. The Brain. The brain. http://www.thebrian.com, 2004.
4. T. Buzan. Mind maps. http://www.mind-map.com, 2004.
5. P. Groenen I. Borg. *Modern Multidimensional Scaling. Theory and Applications.* Springer Series in Statistics, Heidelberg, 1996.
6. T. Kohonen. Websom. http://websom.hut.fi, 1999.
7. J. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. In *Proceedings of the American Mathematical Society*, volume 7, pages 48–50, 1956.

8. T. K. Landauer, P. W. Foltz, and D. Laham. Introduction to latent semantic analysis. *Discourse Processes*, 25:259–284, 1998.
9. D.D. Lewis. Reuters-21578 text categorization test collection. http://www.daviddlewis.com/resources/testcollections/reuters21578/, 1997.
10. P. Matykiewicz. Demonstration applet. http://www.neuron.m4u.pl/search, 2004.
11. W. Pedrycz. *Knowledge-Based Clustering: From Data to Information Granules.* John Wiley and Sons, Chichester, 2005.
12. M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):48–50, 1980.