Tag Cloud Control by Latent Semantic Analysis

submitted by
Angelina Velinska

supervised by
Prof. Dr. Ralf Möller
Dipl. Ing. Sylvia Melzer
Software Systems Institute (STS)
Technical University of Hamburg-Harburg

Dr. Michael Fritsch
CoreMedia AG
Hamburg

Declaration

I declare that:
this work has been prepared by myself,
all literal or content based quotations are clearly pointed out,
and no other sources or aids than the declared ones have been used.

Hamburg, December 2010
Angelina Velinska

Acknowledgements

I would like to thank Prof. Dr. Ralf Möller for his advice and guidance throughout the work on this thesis.
My special gratitude goes to Dr. Michael Fritsch in CoreMedia AG for his great patience, and constant support.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Information Retrieval systems become more important not only due to their use in Internet and digital libraries, but also because the majority of companies (business ?) organize their activities and depend on digital documents, and information. Finding the right information brings value to the business, and failing to do so often leads to losses.

The search applications nowadays, whether used in Internet or in companies internally, face constantly growing problem of information overload. Users searching for information usually submit queries composed of a few keywords only. The search application performs exact matching between the submitted keywords, and the document set it searches through, and often returns a long list of results. Users without domain expertise are not familiar with the appropriate terminology thus not submitting the right query terms with respect to relevance or specialization. Thus, users retrieve a large number of irrelevant pages.

Information Retrieval (IR) systems are facing challenges due to the peculiarities of human behavior, information overload, and information need.

- **Human behavior**. Users searching for information usually submit queries composed of a few keywords only, as shown in studies[1]. The search application performs exact matching between the submitted keywords, and the document set it searches through, and often returns a long list of results. When searching with short queries, the focus of a user is unclear, the missing information contributes to long result lists containing many unrelevant hits. Users without domain expertise are not familiar with the appropriate terminology thus not submitting the right query terms with respect to relevance or specialization. Another issue is the ambiguity of words, when words have more than one meaning. As a consequence, search results do not fit the information needs of users. When relevant documents contain words that are semantically relevant to the queries but not the same (synonyms), they will not be judged relevant.

- **Manual processing of results**. When a large number of matching documents is returned as a search result, only some of the documents found can be read, due to human limitations in information processing, and time constraints (users want to find information quickly). Human users need to narrow down the search iteratively by reformulating the query, since it is unclear in which context their queried words are used, and which words could be useful to focus the search. This reformulation is known as a frustrating and time consuming process.

- **Information need**. Search applications that implement the keyword search paradigm (or full-text search) are broadly accepted by users; however, the challenge for the next years is the development of search solutions that reflect users' context ("what the user meant" versus "what the user wrote"). In other words, solutions that are able to: *a*) organize search results better than in the form of long lists, *b*) adapt to a users personal skills and experience concerning the underlying document collection, and *c*) adapt to the retrieval task a user is concerned with, or to adapt to the users' information need.

The issues listed above have lead to the development of techniques to assist users effectively navigate and organize the available documents in the search results, with the goal to find the best matching their needs. *Documentclustering* is a technique that offers methods to structure the search results in categories, in order to aid users in finding the best results for their search needs. It solves the problem of finding a meaningful ordering in a large amount of returned results.

When a categorization according to topic is determined using an unsupervised approach (document clustering), it has to be presented to the users. In particular, the categories have to be labeled with characteristic terms for browsing. Which words from the cluster to choose as labels is a difficult problem to solve. This work offers an overview of the current algorithms for cluster labeling. It evaluates an interesting new method for cluster labeling, called Weighted Centroid Covering, proposed in [2] and [3]. It further makes a proposal for improving the quality of identified labels, by using external knowledge from a light-weight ontology. This work further presents the ontology, developed for this purpose.

Another important aspect of the problem is to find comprehensive and accurate descriptions (labels) for the clusters of semantically related documents.

Information retrieval systems become more and more important, and have their main application areas in Internet, libraries, and in the business companies.

IR technologies find wide application - in search engines, for browsing or filtering document collections, for further processing a set of retrieved documents. Before retrieval the documents are indexed, otherwise at each search, they would have to be scanned through for each query. The index maps the words or terms back to the documents where they occur. A method for document indexing, which is applied in this work, is called Latent Semantic Analysis (LSA). It indexes the document collection by representing it as a reduced matrix of words and documents. LSA representation improves IR performance with respect to a basic problem of word-matching search - synonymy, or the case when more than one term describe the same concept.

1.2 Information Retrieval systems

In this work we investigate techniques from the field of IR. Therefore, what follows is an overview of the text analysis processes, common to most IR systems. Then, in the context of these processes, we shall summarize the contributions of the current work.

Most IR systems share common workflow, and follow common data processing stages. The main goal of text analysis is knowledge discovery. There are many challenges today in the fields where text analysis is applied, and the main one is that there exists a huge amount of information to process manually. Another challenge is the data ambiguity, synonyms, polysemy. We shall call text collections of documents a text corpus (pl. corpora). Data in such collections can be structured, e.g. data in database tables, semi-structured, e.g. in XML, HTML format, or unstructured - articles, reports, e-mail, etc.

Text analysis processes involve information retrieval, lexical analysis to study word frequency distributions, tagging/annotation, among the others.

Automated processing, modeling, and analysis of unstruc- tured text (news documents, web content, journal articles, etc.) is a key task in many data analysis and decision mak- ing applications. In many cases, documents are modeled as term or feature vectors and latent semantic analysis (LSA) [4] is used to model latent, or hidden, relationships between documents and terms appearing in those documents. LSA supplies conceptual organization and analysis of document collections by modeling high-dimension feature vectors in many fewer dimensions.

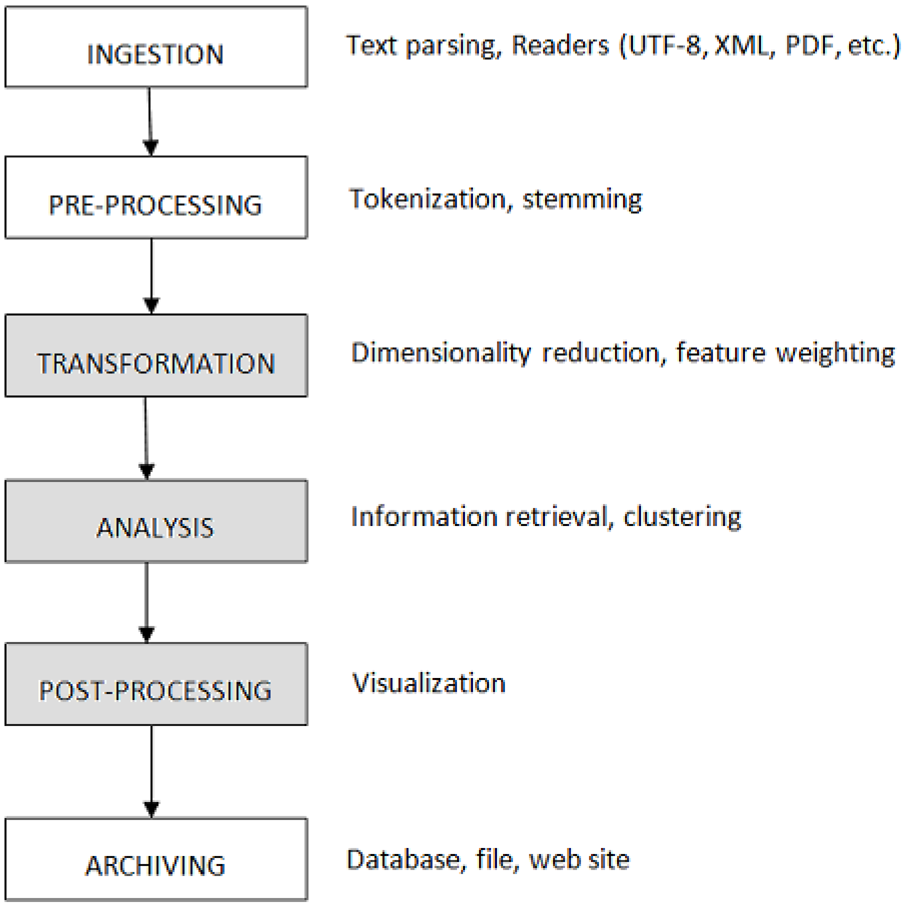

Figure 1.1: [Workflow in IR systems

The current work contributes in the transformation, analysis and post-processing phases of IR workflow. The research focus is automatic identification of the main topics in a set of documents, returned as results to user queries. Research is done in analysis and visualization phases of the text analysis sequence diagram1.1.

This thesis contributes right here; techniques are developed which address the design and the operationalization of IR processes respecting the above points. The research focus is automatic document categorization, a technique which has been shown to improve retrieval performance.
As a contribution, this work offers an overview of the current cluster labeling algorithms.

1.3 Goal and scope of work

This works contributes with the following:

1. It makes an evaluation of Weighted Centroid Covering (WCC) algorithm, proposed for unsupervised topic labeling of clusters in [2] and [3].

2. It proposes an improvement in WCC algorithm, performing topic identification based on external knowledge. A light-weight ontology has been developed for this purpose, in order to be used as a reference for external semantic knowledge during cluster labeling.

3. A software application for executing an IR process has been developed. It implements LSA for information retrieval, and WCC for visualization of the main concepts contained in a document set in the form of a tag cloud.

1.4 Outline

This chapter motivates the presented research work, and summarizes its contributions. It also offers a general overview to the processes involved in text analysis, common to most IR systems. Chapter 2 gives the theoretical foundations for preprocessing and transformation phases of text analysis, and offers a review of a specific technique for information retrieval, called Latent Semantic Analysis. Chapter 3 contains the contribution related to evaluation of WCC algorithm, and a proposal for its improvement, by using a light-weight ontology developed for this purpose. It represents the analytical phase of text analysis problem. Chapter 4 refers to the post-processing phase of text analysis, giving the visualization means to present the concepts retrieved from a document set. The software contribution, developed as a part of this work, is given in Chapter 5. Finally, in Chapter 6 the thesis concludes with evaluation of results and outlook.

Chapter 2

Latent Semantic Analysis

Summary. *The chapter gives a theoretical overview of LSA in the context of its use in this work.*

2.1 Overview

LSA was first introduced in [4] and [5] as a technique for improving information retrieval. Most search engines work by matching words in a user's query with words in documents. Such information retrieval systems that depend on lexical matching have to deal with two problems: synonymy and polysemy. Due to the many meanings which the same word can have, also called polysemy, irrelevant information is retrieved when searching. And as there are different ways to describe the same concept, or synonymy, important information can be missed. LSA has been proposed to address these fundamental retrieval problems, having as a key idea dimension reduction technique, which maps documents and terms into a lower dimensional semantic space. LSA models the relationships among documents based on their constituent words, and the relationships between words based on their occurrence in documents. By using fewer dimensions that there are unique words, LSA induces similarities among words including ones that have never occurred together [6]. There are three basic steps to using LSA: text pre-processing, computing Singular Value Decomposition (SVD) and dimensionality reduction, and querying the constructed semantic space.

2.2 Text pre-processing

If we have a document collection or a text corpus, on which we want to apply LSA, the initial step is to pre-process the texts into a suitable form for running LSA. Pre-processing can include a number of techniques, depending on the application requirements. The process of parsing, also called tokenization, is breaking the input text stream into useable tokens. During tokenization, filtering can be applied, i.e. removing HTML tags or other markup, as well as stop-wording, and removing punctuation marks. Stop words don't convey information specific to the text corpus, but occur frequently, such as: $a, an, and, any, some, that, this, to$.

A distinction has to be made between words or terms, and tokens. A term is the class which is used as a unit during parsing, and a token is each occurence of this class. For example, in the sentence:

> *CoreMedia CMS is shipped with an installation program for interactive graphical installation and configuration of the software.*

the term *installation* is represented by two tokens.

There is no universal way in which to parse a text, and the parsing decisions to address depend on the application in which the text collection will be used. Text parsing will influence all posterior processing in the following stages of LSA.

After tokenization, one has to construct a term-document matrix (2.1). Having as rows the terms, and as columns the documents, its elements are the occurrences of each term in a particular document, where a_{ij} denotes the frequency with which term i occurs in document j. The size of the matrix is **m x n**, where **m** is the number of terms, and **n** is the number of documents in the text collection. Since every term doesn't appear in each document, the matrix is usually sparse.

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \tag{2.1}$$

Local and global weightings are applied to increase or decrease the importance of terms within documents. We can write

$$a_{ij} = L(i,j) \times G(i), \tag{2.2}$$

where $L(i, j)$ is the local weighting of the term i in document j, and $G(i)$ is the global weighting for term i. The choice of a weight function has impact on LSA performance, therefore in Section 2.5 we give an overview of the most common weight functions.

2.3 Singular Value Decomposition

After the initial pre-processing, the term-document matrix is decomposed into three matrices (2.3) by applying Singular Value Decomposition (SVD). It is a unique decomposition of a matrix into the product of three matrices - U and V are ortonormal matrices, and Σ is a diagonal matrix having singular values on its diagonal.

$$A = U\Sigma V^T \tag{2.3}$$

After the initial matrix A is decomposed, all but the highest k valued of S are set to 0. The resulting reduced matrix is the semantic space of the text collection. A classical example presenting the truncated SVD [4] can be used for displaying dimensionality reduction, and how it affects all three matrices.

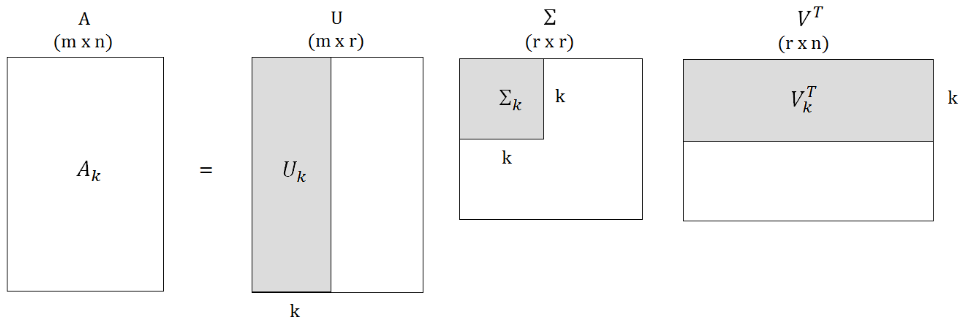

Figure 2.1: Diagram of truncated SVD

A_k - best rank-k approximation of A	m - number of terms
U - term vectors	n - number of documents
Σ - singular values	k - number of factors
V^T - document vectors	r - rank of A

Figure 2.1 is a visual representation of SVD as defined in equation (2.3). U and V are considered as containing the term and document vectors

respectively, and Σ is constructed by the singular values of A. An impoprant property of SVD is that the singular values placed on the diagonal of Σ are in decreasing order. Hence, if all but the first k singular values are set to 0, the semantic meaning in the resulting space is preserved to some approximation k, while noise or variability in word usage, is filtered out. Noise in this case are the terms with lowest weights which carry little meaning. By using fewer dimensions k, LSA induces similarities amont terms including ones that have never occured together. Terms which occur in similar documents, for example, will be near each other in the k-dimensional space even if they never co-occur in the same document. This means that some documents which do not share any words with a users query may be near it in k-space.

A factor to be considered when computing SVD is the run-time complexity of the algorithm. For decomposition of very large matrices, it is $O(n^2k^3)$, where n is the number of terms in the text corpus, and k is the number of dimensions in semantic space after dimensionality reduction. Note that k is typically a small number between 50 and 350.

A more detailed description of SVD can be found in [7] and [8].

2.4 Querying the semantic space

In this work we are using LSA for IR purpose. Therefore, the final step of applying the technique is to pose queries on the constructed semantic space. A query q is a set of words which must be represented as a document in the k-dimensional space, in order to be compared to other documents. The user's query can be represented by

$$q = q^T U_k \Sigma_k^{-1} \tag{2.4}$$

where q is the set of words in the query, multiplied by the reduced term and singular values matrices. Using the transformation in (2.4), the query is "mapped" onto the reduced k-space. After the mapping, the resulting query vector can be compared to the documents in the k-space, and the results ranked by their similarity or nearness to the query. A common similarity measure is the cosine between the query and the document vector. From the resulting document set, the documents closest to the query above certain treshold are returned.

2.5 Factors influencing LSA performance

The effective usage of LSA is a process of a sophisticated tuning. Several factors can influence the performance of the technique. These factors are pre-processing of texts (removal of stop-words, filtering, stemming), frequency matrix transformations, choice of dimensionality k, choice of similarity measure.

Dumais et al. [9] and Nakov et al. [10] have carried research on LSA performance depending on the choice of factors such as frequency matrix transformations, similarity measures, and choice of dimension reduction parameter k. They conclude that performance based on the choice of these factors depends on the particular text corpus, as well as on the purpose of LSA application. However, in the case of matrix transform, log-entropy performs better as compared to other matrix transform function combinations, including the popular term frequency - inverse document frequency ($tf \times idf$). Therefore, we implement the former in this work.

Local function: logarithm	$L(i,j) = \log(tf(i,j)+1)$
Global function: entropy	$G(i) = 1 + \frac{\Sigma_j p(i,j)}{\log n}$

where n is the number of documents in the collection.

Further, it has been stated ([9],[11]) that with respect to similarity measures used, LSA performs optimal when cosine similarity measure is implemented to calculate the distance between vectors in the semantic space. We have therefore used it to measure the relevance between queries and documents. The cosine measure between two vectors d_1 and d_2 is given by:

$$sim(d1,d2) = \frac{\overrightarrow{V}(d_1).\overrightarrow{V}(d_2)}{\left|\overrightarrow{V}(d_1)\right|.\left|\overrightarrow{V}(d_2)\right|} \tag{2.5}$$

Dimensionality reduction parameter k is defined empirically based on the experimentation results presented in Chapter 5.

Chapter 3

Topic identification in clusters

A major problem in text analysis is to determine the topics in a text collections and identify the most important, novel or significant relationships between topics. Clustering and visualizations (tag clouds) are key analysis methods in order to solve this problem.

Clustering is widely used for recommendation, and for categorizing search. An example of a recommendation is ”X” like these. The search engine will look for similar results as the ones presented.

disadvantages of clustering:
objects can be assigned to one cluster only
in social networks, clustering can be used to recognize communities in large groups of people.
clustering is also used in partitioning web documents into groups, a.k.a. genres (data mining)
search engines - categorization of search results or grouping (Yippy search engine)
recommender systems - recommend new items based on user's taste

When performing classification by clustering, the cluster labels are usually manually created by human beings. However, this is a very expensive approach. It is sensible to find and algorithm for automatically identifying topic labels or cluster labels. Therefore, we have investigated the performance of Topic identification algorithm by Stein and zu Eissen[2].

3.1 External topic identification

The best scenario is that cluster labels should present a conceptualization of the documents in text corpus. This is not achieved by the algorithm presented. Technically, a hierarchical clustering algorithm can construct from each Document set D a category tree. However, the labeling based on this hierarchical clustering will be far from a semantical taxonomy. This weakness of the algorithm presented can be corrected by using an external classification knowledge, e.g. an upper-level ontology.

We believe that the weaknesses of topic identification algorithms in categorizing search engines could be overcome if external classification knowledge were brought in. We now outline the ideas of such an approach where both topic descriptors and hierarchy information from an upper ontology are utilized.

Then, topic identification is based on the following paradigms:
1. Initially, no hierarchy (refines-relation) is presumed among the C 2 C. This is in accordance with the observations made in [Ertz et al. 2001].
2. Each category C 2 C is associated to its most similar set O 2 O. If the association is unique, $To(O)$ is selected as category label for C.
3. Categories which cannot be associated uniquely within O are treated by a polythetic, equivalence-presuming labeling strategy in a standard way. In essence, finding a labeling for a categorization C using an ontology O means to construct a hierarchical classifier, since one has to map the centroid vectors of the clusters C 2 C onto the best-matching O 2 O. Note that a variety of machine learning techniques has successfully been applied to this problem; they include Bayesian classifiers, SVMs, decision trees, neural networks, regression techniques, and nearest neighbor classifiers.

Chapter 4

Tag Clouds

We have found so far no other application implementing LSA in order to present an overview of the main topics found in a collection of unstructured texts, based on user queries. The TagCloud Summarizer is in this sense new.

There exist, however, search engines, which utilize categorizing of search results, such as Yippy(former Clusty, Vivisimo). It utilizes search, classification, and Social web (Web 2.0).

4.1 existing implementations

http://cloud.yippy.com/ - visualizes topics based on search queries. Created at Vivisimo company, also creator of one of the most successful meta search engines, offering classification of search results, Clusty (Vivisimo).

SenseBot Summarizer summarizes search results in the form of a tag cloud.

Google on the other hand offers "Wonder wheel" option, in order to display search results

TagCloud Summarizer is a tool that users can use to instantly visualize a topic using the familiar tag cloud display. Users can create a cloud based on a query.

TagCloud Summarizer generates a cloud using the user's search results for the topic they enter. Using the Summarizer to generate the cloud also ensures that it is always up-to-date because topics/main concepts are generated in real-time, based on the user's query.

4.2 use

Use the TagCloud Summarizer for online web-pages, search systems, or personal web-sites.

Summary. *This chapter presents an overview of tag-clouds used as a method for representing text content.*

Tag Clouds are popular applications used for vaious purposes: as a navigation mechanism, as indicators of activity within social media experiences, for visualization in texts and textual data, for annotation of documents 4.1. The importance or weight of words in the tag cloud are shown with size of font and/or color. The tag clouds are hyperlinks leading to a collection of items associated with the tag.
A version of tag cloud is called text cloud. It is used as a visual display that conveys the broad themes that emerge from textual analysis. There are three types of tag clouds depeding on their purpose and use. The first type contains a tag represeting the frequency of each term. The second type is a global tag cloud whose tags has frequencies aggreggated over all items and users. The third type of tag cloud contains categories, and its tags' size indicates the number of subcategories.

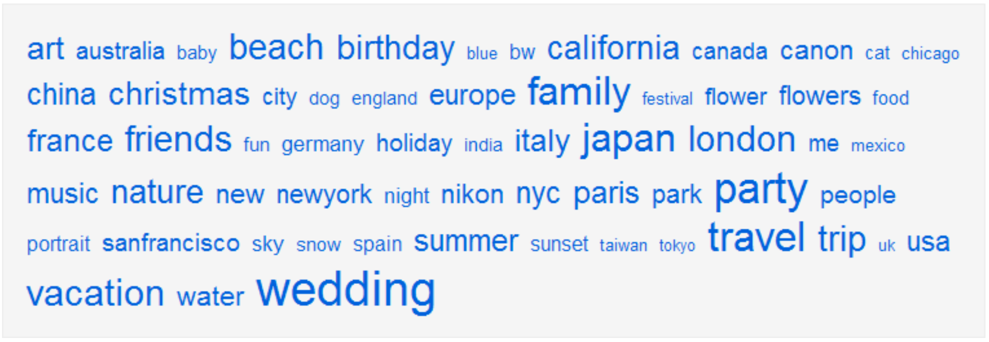
art australia baby beach birthday blue bw california canada canon cat chicago
china christmas city dog england europe family festival flower flowers food
france friends fun germany holiday india italy japan london me mexico
music nature new newyork night nikon nyc paris park party people
portrait sanfrancisco sky snow spain summer sunset taiwan tokyo travel trip uk usa
vacation water wedding

Figure 4.1: Tag Cloud

4.3 1

Related work
Opinion Crawl[1] - web sentiment analysis application. It generates a concept cloud from daily scanned blogs, web site articles.

[1]`http://www.opinioncrawl.com/`

SenseBot Search Results Summarizer is a plugin for Mozilla Firefox browser that generates a tag cloud of the main concepts returned as search results from Google.

Search Cloudlet[2] is another Firefox Addon that inserts a related tagcloud into Google interface. Working behind the scenes, Search Cloudlet injects a tag cloud of related words in to both Google and Yahoo search results pages. Then you can use the tag links to quickly and easily filter and refine your searches.
LinkSensor SenseBotSummarizer All three are based on SenseBot - a semantic search engine. Made available from Semantic Firefox Extensions [3]

4.4 Brainstorming

What is a tag cloud? Graphical representation of a collection of tags. Tag clouds visualize word frequency in a given text.

Tag clouds may be used as a topic summary.

There are three main types of tag cloud applications used in social software.

1. frequency of items / tags
2. number of items to which a tag has been applied
3. tags are categorization method for content items

The following tag clouds were evaluated in order to select the solution that is most applicable for Tag Cloud Summarizer project.

- TagsTreeMaps[4]
- OpenCloud[5]

[2]`https://addons.mozilla.org/en-US/firefox/addon/9943/`
[3]`http://www.semanticengines.com/plugins.htm`
[4]`http://tagstreemaps.sourceforge.net/TagsTreeMaps.html`
[5]`http://opencloud.sourceforge.net/`

Chapter 5

Implementation and evaluation

Summary. *This chapter reports the implemented solution for the given thesis problem, gives discusses its advantages and disadvantages.*

It is a challenge by itself to come up with a sensible evaluation set for an IR implementation. ... Define here precision, recall, measures , how to measure LSA performance with diff. k, clusters, cluster labelling.

The document collection consists of guides and manuals about CoreMedia Content Management Systems (CMS) 5.2.

5.1 LSA implementation

LSA was applied to a collection of 11818 words in 4000 documents, all of which describe CoreMedia CMS 5.2. The algorithm took 63552 ms for preprocessing and indexing of the whole document collection.

For the implementation of LSA this work uses the open LSA library which is part of Semantic Spaces Project[12]. It is developed at the Natural Language Processing Group at the University of California at Berkley (UCLA)[1].

The real difficulty of LSA is to find out how many dimensions to remove - the problem of dimensionality.

[1]`http://code.google.com/p/airhead-research/`

TODO:test if inluding only terms that occur in more than one document improved LSA performance with respect to generating precise tag clouds.

5.2 Tag Cloud implementation

The implemented open source library used for tag cloud generation is called Opencloud[2], and is provided by Marco Cavallo.

5.3 Tools used

Airhead Research[3] project was used as a semantic spaces Package which provides a java-based implementation of LSA.

Apache Lucene[4] was used as an indexing and search library.

5.4 Advantages and drawbacks

why am i using lsa instead of lda for example?

1. PLSA - characteristics, advantages, disadvantages
2. LDA - characteristics, advantages, disadvantages

5.5 Latest development in the field of LSA

LSAView is a tool for visual exploration of latent semantic modelling, developed at Sandia National Laboratories [13].

at the end- improvements of lsa with the basics explained.

5.6 Improvements

only terms occuring in more than one documents have been included
stop words

[2]`http://opencloud.mcavallo.org/`
[3]`http://code.google.com/p/airhead-research/`
[4]`http://lucene.apache.org/java/3_0_2/`

compound words list
Lanczos algorithm for computation of SVD of large sparse matrices
Multi-threading computation in the project

Chapter 6

Conclusion and outlook

Summary. *summarize me*

6.1 1

6.2 Future Work

1. implement LSA based IR in full-text search results, or as recommendation.
2. link tags from tag cloud to the documents where they occur (using lucene indexing ???)
3. investigate also the case of updating the document collection - how to handle this case
4. Improve TagCloudSummarizer to work also with German texts (company has a website that support German, Russian, French..)
5. Make the process run in parallel.

Acronyms

CMS	Content Management Systems.
IR	Information Retrieval.
LSA	Latent Semantic Analysis.
SVD	Singular Value Decomposition.
WCC	Weighted Centroid Covering.

Appendix A

Appendix

TODO: insert important source code parts here.

Bibliography

[1] A. Spink, J. Bateman, and B. J. Jansen, “Users’ searching behavior on the excite web search engine,” in *WebNet*, 1998.

[2] B. Stein and S. M. Z. Eissen, “Topic identification: Framework and application,” in *Proc of International Conference on Knowledge Management (I-KNOW)*, 2004.

[3] B. Stein and S. M. zu Eissen, “Topic identification,” *KI*, vol. 21, no. 3, pp. 16–22, 2007.

[4] S. T. Dumais, G. W. Furnas, T. K. Landauer, S. Deerwester, and R. Harshman, “Using Latent Semantic Analysis to improve access to textual information,” in *Sigchi Conference on Human Factors in Computing Systems*, pp. 281–285, ACM, 1988.

[5] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, “Indexing by Latent Semantic Analysis,” *Journal of the American Society for Information Science*, vol. 41, pp. 391–407, 1990.

[6] S. Dumais, “LSA and Information Retrieval: Getting Back to Basics,” pp. 293–321, 2007.

[7] M. W. Berry, S. Dumais, G. O’Brien, M. W. Berry, S. T. Dumais, and Gavin, “Using linear algebra for intelligent information retrieval,” *SIAM Review*, vol. 37, pp. 573–595, 1995.

[8] G. H. Golub and C. F. Van Loan, *Matrix computations (3rd ed.)*. Baltimore, MD, USA: Johns Hopkins University Press, 1996.

[9] S. T. Dumais, “Improving the retrieval of information from external sources,” *Behavior Research Methods, Instruments, & Computers*, vol. 23, pp. 229–236, 1991.

[10] P. Nakov, A. Popova, and P. Mateev, “Weight functions impact on LSA performance,” in *EuroConference RANLP’2001 (Recent Advances in NLP*, pp. 187–193, 2001.

[11] P. Nakov, "Getting better results with Latent Semantic Indexing," in *In Proceedings of the Students Prenetations at ESSLLI-2000*, pp. 156–166, 2000.

[12] D. Jurgens and K. Stevens, "The S-Space package: an open source package for word space models," in *ACL '10: Proceedings of the ACL 2010 System Demonstrations*, (Morristown, NJ, USA), pp. 30–35, Association for Computational Linguistics, 2010.

[13] P. Crossno, D. Dunlavy, and T. Shead, "Lsaview: A tool for visual exploration of Latent Semantic Modeling," in *IEEE Symposium on Visual Analytics Science and Technology*, 2009.