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

Contents

List of Figures

Chapter 1

Introduction

Identifying the main concepts in texts is the subject of many research studies in the field of information retrieval and data mining.

This work investigates the implementation of Latent Semantic Analysis (LSA) for discovering the main concepts in texts, in order to present an overview of the text content in the form of a tag cloud.

1. introductory words, why is this work being written
2. mention information retrieval, lsa, tag clouds - generally
3. mention cms ? document collections ? content ?
4. mention the work of david mugo

During the last decade there have been constant optimizations in information retrieval effectiveness, making web search the preferred source of finding information. A substantial part of information retrieval deals with providing access to unstructured information in various domains. Information Retrieval (IR) refers to finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers) [1]. Many people today use methods from the field of IR when they use a search engine online, or search through their emails. In this context "unstructured data" refers to data which does not have a clear structure.

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.

While IR deals with retrieval of documents, other systems manage content, such as documents. Content management includes a set of technologies and processes that support the creation, management and publication of content in any form or medium. Content may be documents, multi-media files, or any other file types that follow content lifecycle and require management. Content Management Systems (CMS) vary depending on their purpose and target environments - there are CMS for the web, for enterprise, for mobile devices, as well as CMS for managing collection of documents.

1.1 Motivation and objective

A drawback of the classical LSA implementation as an IR method is the low precision of the returned results. A previous work by David Mugo [2] has investigated the improvement of LSA precision performance by annotating the document collection and including the anotations used in LSA. In his work, Mugo constructs a concept-document matrix from the annotations used, and concatenates it with the word-document matrix normally generated in LSA process. The proposed solution, however, results in a slow speed of LSA, and has left Mugo's hypothesis open.

Taking into consideration the results from Mugo's work, the current project has several objectives to reach. It will investigate the implementation of LSA method for improving information retrieval in a domain-specific document management system with respect to context-based search. A further investigation will be made on improving the precision performance of LSA method by using semantic annotations, and on finding an adequate way to present the results of LSA as a tag cloud. And finally, it will be investigated how to use the tag cloud as a form of a relevance feedback to control LSA method.

In the context of the stated objectives, semantic annotations are meta data annotations used to add information to unstructured data, or to the document collection. Semantic annotations are based on an ontology in our case, specifically developed for the domain of interest CoreMedia CMS. Ontologies are used to capture some knowledge about a certain domain, by describing the concepts of the domain and the relationships between them. To further clarify the objectives, relevance feedback is an IR technique, used to influence the retrieved results based on the user's preference. It allows the user to modify the initial tag cloud by selecting the most relevant words. The tag cloud is then re-generated from LSA results with the relevance feedback posted as a query.

1.2 Outline

The reminder of this work is organized as follows. Chapter **??** describes in more detail what a document management system is, and provides an overview of the general structure of DocMachine 2.0, the CMS deployed at CoreMedia AG. Chapter 3 presents the basic concepts of ontologies and document annotations based on ontologies. In Chapter 2 an overview of latent semantic analysis method is given, as well as an approach for improving LSA's precision by including semantic annotations in the method. Chapter 4 presents the prototype implementation and makes an evaluation of the results achieved in this work. And finally, conclusions are drawn in Chapter 5, along with some limitations of the current study and outlook for a future research.

Chapter 2

Latent Semantic Analysis

Summary. *This chapter gives the theoretical foundations of LSA since it is used in this project as a method for defining the main concepts in text documents.*

2.1 Overview

LSA was developed at the end of 1980s to address certain deficiencies in Information Retrieval, caused by synonymy and polysemy. Synonymy is the case when several words describe the same concept. Polysemy is when words have more than one distinct meaning.

LSA method is applied in four main steps. First, a words-by-documents matrix, constructed using the documents in the text collection. This matrix is sparse, as not all words occur in all documents.

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

Each word (or term) is a row in the matrix, and each document - a column. The size of the matrix is **m x n**, where **m** is the number of documents in the text collection, and **n** is the number of terms. As a second step, the number of term occurences in each document is transformed into a weight using a weight function, such as entropy, term frequency, inverse document frequency. These weights are represented by the a_{ij} entries in the matrix in 2.1. After the initial pre-processing, and as a third step of LSA, the rank of the matrix is reduced by applying a method of matrix decomposition, called Singular Value Decomposition (SVD). The initial matrix A is large and sparse, such that decomposition reduces the number of rows and columns to a certain parameter **k**, defined empirically. The result from applying SVD to the terms-by-documents matrix A are tree matrices U, S and V, as in 2.2.

$$A = USV^T \tag{2.2}$$

U is a ...
V is ...
and S is ...
After the initial matrix A is decomposed by SVD, all but the highest k valued of S are set to 0. The resulting reduced matrix is the semantic space of the text collection.

And the final step of applying LSA is to the compute similarities between entities in the semantic space. This includes computing similarities between queries posted on the document collection, and the documents in the semantic space. In this case the query q has to be "translated" as a document from the semantic space, by using 2.3.

$$q = q^T U_k S_k^{-1} \tag{2.3}$$

2.2 Singular Value Decomposition

SVD is an unique decomposition of

LSA constructs a term-by-document matrix based on term occurrence, and uses a given similarity measure to find out the distance between vectors (documents) in the semantic space it generates.

There are three main factors that can influence the performance of LSA[3][4]:

- Frequency matrix transoformations (choice of weighting function)
- Choice of dimensionality

- Text preprocessing prior to SVD, choice of similarity measure (???)

Further, the choice of dimensionality is dependent upon the matrix transformations performed, as pointed out by Nakov[4].

2.3 Latest development in the field of LSA

LSAView is a tool for visual exploration of latent semantic modelling, developed at Sandia National Laboratories [5].

at the end- improvements of lsa with the basics explained.
why am i using lsa instead of lda for example?

2.4 plan

1. text processing and peculiarities; stemming, lemmatization, stopwording
2. lsa and basics
3. weighting functions and their effect ot LSA results [3].
4. lsa used for information retrieval; lsa used for defining the main concepts in texts. precision vs. recall.
(first explain the basics of LSA, then explain how factors can influence lsa)
Several factors influence the quality of results which LSA delivers. These factors are pre-processing (removal of stop-words, stemming, lemmatization), frequency matrix transformations, choice of dimensionality, choice of similarity measure.
A study by Nakov, Popova, Mateev[3] has summarized the influence of those factors on LSA, and has concluded that...

2.5 storage

Text Processing and LSA

Text processing:
- retrieve documents from DB
- tokenize texts
- stem/lemmatize texts - this drops off as we will use the terms as a part of a tag cloud

- stop wording
- build SVD
- post queries on the matrix

The document collection consists of guides and manuals about CoreMedia Content Mangement System 5.2.

Improving performance of LSA information retrieval method includes tf*idf weighting scheme, relevance feedback by implementing Tag Cloud, and choosing the number of dimensions for the reduced spacing. Stemming as a method for LSA improvement is not applied, as investigations showed at most modest improvements with this method.

Library/implementation used for LSA is S-Spaces from Airhead Research project of UCLA (University of California at Los Angeles). The implemented algorithm for SVD is Lanczos, ported from SVDLIBC implementation by Doug Rohde from Tennessee University.

Use the paper "Weight functions impact on LSA performance" by Preslav Nakov, Antonia Popova, Plamen Mateev - very nice concise description of LSA + analysis.

IMPORTANT
I should test entropy and idf , as sometimes entropy global weighting function has a better performance.

For text processing, Snowball project is used, from the laboratory of Martin Potter, the author of the infamous Porter Stemming algorithm. !!! No stemming or lemmatization should be done on the input document collection, as the resulting terms/tags from LSA will be used in a TagCLoud!

11818 words in word space
63552ms to run LSA on 4000 documents
and IDEA blocks

Due to the problem above, the process of SVD calculation has to be performed in a multi-threaded way, and the project has to be optimized with respect to performance, in order to be able to successfully run.
Keep only wht words common to at least 2 documents???

2.6 Alternative approaches for LSA

1. PLSA - characteristics, advantages, disadvantages

2. LDA - characteristics, advantages, disadvantages

Chapter 3

Tag Clouds

Summary. *This chapter presents an overview of tagclouds 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 3.1. 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.

3.1 1

Related work 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.

LinkSensor SenseBotSummarizer All three are based on SenseBot - a semantic search engine. Made available from Semantic Firefox

Figure 3.1: Tag Cloud

Extensions [1]

3.2 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.

(a) frequency of items / tags

(b) number of items to which a tag has been applied

(c) 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[2]
- OpenCloud[3]

[1] http://www.semanticengines.com/plugins.htm
[2] http://tagstreemaps.sourceforge.net/TagsTreeMaps.html
[3] http://opencloud.sourceforge.net/

Chapter 4

Implementation and evaluation of results

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

4.1 LSA implementation

For the implementation of LSA this work uses the open LSA library which is part of Semantic Spaces Project[6]. 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.

4.2 Tag Cloud implementation

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

[1] `http://code.google.com/p/airhead-research/`
[2] `http://opencloud.mcavallo.org/`

Chapter 5

Conclusion and outlook

Summary. *summarize me*

5.1 1

5.2 Future Work

(a) Improve TagCloudSummarizer to work also with German texts (company has a website that support German, Russian, French..)

(b) Make the process run in parallel.

Acronyms

LSA Latent Semantic Analysis.

LSI Latent Semantic Indexing.

Appendix A

Appendix

You should not print the full source code :-). But note that the chapters are now called "appendix" and numbered with letters.

Bibliography

[1] C. D. Manning, P. Raghavan, and H. Schtze, *Introduction to Information Retrieval.* New York, NY, USA: Cambridge University Press, 2008.

[2] D. M. Mugo, "Connecting people using Latent Semantic Analysis for knowledge sharing," Master's thesis, Hamburg University of Technology, Jan. 2010.

[3] 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.

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

[5] 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.

[6] 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.