**Towards the Development of an Integrated Framework for Enhancing Enterprise Search Using Latent Semantic Indexing**

Obada Alhabashneh, Rahat Iqbal, Nazaraf Shah, Anne James, Saad Amin

Faculty of Engineering and Computing,

Coventry University, UK

{Alhabaso, r.iqbal, n.shah, a.james, s.amin}@coventry.ac.uk

**Abstract.** While we have seen a significant success in the web search, the enterprise search has not yet been widely investigated and as a result the benefits that can otherwise be brought to the enterprise are not fully materialized. In this paper, we present an integrated framework for enhancing the enterprise search. The framework is based on open source technologies which include Apache Hadoop, Tika, Solr and Lucene. Importantly, the framework also benefits from Latent Semantic Indexing (LSI) algorithm to improve the quality of the obtained search results. LSI is a mathematical model used to discover the semantic relationship patterns in a documents collection. We envisage that the proposed framework will benefit various enterprises to improve their productivity by meeting their information needs effectively.

**Keywords:** Enterprise Search (ES), Latent Semantic Indexing (LSI), Search Context, Document Ranking.

1 Introduction

Enterprises have a rich and diverse collection of various information resources. Such resources can be divided into two categories: structured which is encoded in the databases; and unstructured which is encoded into the enterprise documents.

The retrieval of structured information has been well investigated [1], and several search tools or products are available in the market, such as, the traditional database engines (e.g., Oracle, Microsoft SQL server, MySQL). However, retrieval of unstructured information is still a challenging task due to several problems associated with the search of unstructured information. For example, lack of anchor text (e.g., hyperlinks) and the heterogeneous formats of documents.

Enterprise Search Engines (ESEs) are still not advanced and matured enough to provide high quality results, satisfying its users’ needs. According to the International Data Corporation (IDC) report there are significant economic losses caused by poor quality of enterprise search. The report also noted that there is dissatisfaction by the enterprise executives about the performance and information quality of the available ESEs [2, 3].

Although a wide range of commercial enterprise search products are available from various vendors, such as, Google, Verity, IBM, Oracle, Microsoft and Panoptic but none of the existing enterprise search products provide an effective solution [2, 4, 5].

Enterprise search has relatively attracted less interest of the research community, particularly in the area o f unstructured information retrieval [4, 6]. A number of researchers have attempted to address enterprise search problems with a varying degree of success [6, 7, 1, 8, 9, 10, 11, 25]. For example, Dmitriev et al used the implicit and explicit annotation to substitute the lake of anchor texts in order to enhance the ranking of documents in the search result list [6]. Mangold et al proposed a framework to extract the search context and the logical structure of the information from the enterprise databases to enhance the search results quality [1]. Zhu et al attempted to disambiguate the user query using dictionaries to enrich the query with additional keywords [7].

In this paper, we present an integrated framework for enhancing the enterprise search. The framework is based on open source technologies which include Apache Solr, Lucene, Tika, and Hadoop. The selection of open source technologies is made in order to address the issues concerning scalability and enabling incorporation of LSI algorithm. The objective of using the LSI is to enhance search results obtained using available open source enterprise search technologies.

The rest of the paper is organized as follows. Section 2 discusses some of the problems related to enterprise search. Section 3 presents the proposed framework. Section 4 concluded the paper and outline future research direction.

2 Enterprise Search problems

2.1 Heterogeneous documents

Most of the enterprise documents are non-web documents. They are of heterogeneous nature having different types and structure. This type of document heterogeneity makes the webpage ranking techniques less efficient. For example the PowerPoint files consist of slides and each slide has a title and body; the title part, logically, should have a higher importance than the body part. On the other hand, the Excel sheets have a structure of columns and rows and always consist of numerical values with a limited text description apart from the columns’ titles. The challenge is how to apply the same ranking algorithm or technique on a different file types. The text based ranking methods are not effective in this case. [1, 2, 3, 4]

2.2 Non-web document

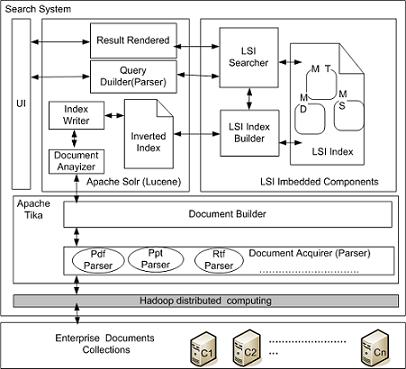
The structure analysis shows that the enterprise web is not following bow-tie structure of the WWW pages which makes the page rank algorithm less efficient in the enterprise search. Most of the enterprise documents have no anchor texts. The anchor texts are used by the traditional web search engines as a base to calculate the document importance in the document ranking algorithm. The lack of such texts makes these algorithms inefficient in enterprise search [1, 2, 3, 4, 12]

2.3 Search context

Search context is useful to disambiguate the short or ambiguous queries, since it adds more keywords to the user query which in turn makes it clearer to the search process and produce a properly ranked search result list [1, 2, 3, 4, 12]. Search context can be extracted from different resources such as user profile.

1. The proposed Framework

We propose an integrated framework in order to improve the quality of enterprise search. By quality we mean the efficiency, accuracy and user satisfaction of the obtained results. The proposed framework consists of four major components such as Hadoop, Tika, Solr and LSI as shown in Figure 1. These components are briefly described in the subsequent subsections and outlines in table 1.

**Figure 1.** The proposed Framework

**Table 1.** Description of the proposed framework components.

|  |  |  |
| --- | --- | --- |
| No | Component | Description |
| 1 | Apache Hadoop | It is an open source framework for distributed computing |
| 2 | Apache Tika | It is an open source toolkit that can parse and acquire different types of documents |
| 3 | Apache Solr | It is an open source enterprise search server |
| 4 | Latent Semantic Indexing (LSI) | It is a vector space based model that is used to retrieve the documents biased tier semantic relationships with user query. |

3.1 Apache Hadoop

The Apache Hadoop is an open source component designed for distributed computing paradigm. It includes several components to provide infrastructure for reliable and scalable distributed computing. Hadoop implements a computational paradigm named map/reduce, where the application is divided into many small fragments of work, each of which may be executed or re-executed on any node in the cluster [26, 28].

3.2 Apache Tika

Apache Solr and its underlying technology can parse or acquire limited types of documents. In order to address this issue, we use a specialised toolkit, called Apache Tika, to deal with a diversity of document types available in the enterprise document collection component.

Apache Tika is an open source document acquiring and building toolkit that is compatible with Solr since they are working under Apache umbrella. It is able to acquire different types of documents using its standard library. It can further offer new plugin services to acquire additional document types [25, 27, 30]. It has two sub components as briefly described below:

Acquiring Component.Itis responsible to extract the metadata as well as the body of the document in a form of text, and then pass them to the document building component.

Document Building Component:Itsfunction is to transform the document from the stream text to a field form such as; title, author, and date of creation, the body, the link and others. Solr’s analyzing component takes these forms as input and performs analysis on the documents to prepare them for indexing.

**3.3 Apache Solr**

Apache Solr is a java based open source enterprise search server available under Apache licence. Solr can communicate with other components and applications using standard language and protocols (XML, HTTP).

Apache Solr is widely used in many public websites such as CNet, Zappos, and Netflix, as well as intranet sites. In addition to the standard ability to return a list of search results for some query, it has various other features such as: result highlighting, faceted navigation which allows user to select various alternatives for their query, query spell correction, auto-suggest queries or more option for finding similar documents.

Apache Lucene is the core technology underlying Solr. Lucene is an open source, high-performance text search library [26, 29] whereas the Solr is a search server**.** Solr uses the following components:

**Document Analysis Component.**No search engine indexes text directly. Instead, the text is broken into a series of individual atomic elements called tokens. This is carried out in the ‘analyse document’ stage. Each token corresponds roughly to a ‘word’ in the language, and this step determines how the textual fields in the document are divided into a series of tokens preparing to add it to the index.

**Index Writer.** After the input has been converted into tokens it is then added to the index. Index writer stores the input in a data structure known as an inverted index. The index is built incrementally as building blocks called segments, and each segment consists of a number of documents. In general, the Solr (Lucene) index consists of the following main files: Segments file, Fields information file, Text information file, Frequency file, and Position file.

**Query Parser.**The query may have boolean operations, phrase queries (double quoted), wildcard terms and other expressions that might need specific processing to transform it into appropriate syntax required by search mechanism. At this stage, query parser transforms the user’s query into common search syntax, known as query object form. With respect to that, Solr provides a package to build the query called QueryParser.

**Result Renderer**This part is responsible for taking the search result as a plain list, and then puts it in the right order and finally passes it to the user interface. Result Renderer will be tailored to fit the LSI based search result.

3.4 Latent Semantic Indexing

Latent Semantic Indexing (LSI) is a vector space based technique used to create associations between conceptually related documents’ terms [16, 17, 18]. LSI is a novel information retrieval method used to retrieve relevant documents using a statistical algorithm. The algorithm is capable of finding relevant documents based on semantic relationship between the terms used in the user query and a collection of documents [13, 19, 20]. LSI has been proven to be more effective in searching and ranking relevant items as compared to other classical key word based methods [13, 21, 22, 23, 24].

The LSI uses a linear algebra to identify the conceptual correlations among a text collection. In general, the process involves the flowing steps;

* Building term-document matrix,this step builds the term-document matrix (A) which consists of the terms (rows) and documents (columns).
* Weighting the matrix (A)which applies a mathematical semantic weighting formula to give a numerical value for each cell in the matrix.
* Performing a Singular Value Decomposition (SVD) technique on the matrix (S) to identify patterns in the relationships between the terms and concepts contained in the text [14]. It builds the term-document vector spaces by transforming the single term-frequency matrix, *A*, into three other matrices: (T) the term-concept Victor matrix, (*S*) the singular values matrix, and (*D*) the concept-document vector matrix as shown below [14]:

|  |  |
| --- | --- |
| A≈ T .S.D |  |

After the term document vector space is built, the similarity of terms or documents can be represented as a factor of how close they are to each other, which could be simply computed as a function of the angle between the corresponding vectors.

* Querying and Augmenting LSI matrices, the same steps are used to locate the vectors representing the text of queries and new documents within the document space of an existing LSI index [15].

LSI Index Builder. The LSI index builder builds the LSI index. It takes the Solr inverted index as input and then creates the basic Term-Document matrix by applying the LSI semantic weighting function. Following that it applies Singular Value Decomposition to the matrix to extract the three sub matrices (T, D and S) which construct the LSI index.

LSI Searching and Ranking*.* We modify the Apache Solr searching and ranking component to be used with the LSI index in order take into account the semantic weights in the ranking of relevant documents.

1. Conclusions and Future Work

In this paper, we present an integrated framework for enhancing the enterprise search. The framework is based on open source technologies which include Apache Hadoop, Tika, Solr and Lucene. Importantly, the framework also benefits from Latent Semantic Indexing (LSI) algorithm to improve the quality of the obtained search results. LSI is a mathematical model used to discover the semantic relationship patterns in a documents collection. We envisage that the proposed framework will benefit various enterprises to improve their productivity by meeting their information needs effectively. Our future work will include further development of the proposed framework. We will conduct a series of experiment on test data (TREC 2007 Enterprise Document TEST Collection) to evaluate the effectiveness of the framework. Different evaluation metrics will be used to measure the accuracy of the obtained results, efficiency of the proposed implementation, and user satisfaction.

1. References
2. Mangold, C., Schwarz, H., Mitschang, B.: u38: A Framework for Database-Supported Enterprise Document-Retrieval, In 10th International Database Engineering and Applications Symposium (IDEAS'06) , IEEE 0-7695-2577-6/06 (2006)
3. Hawking, D.: Challanges in Entrerprise Search. In 5th Australasian Database Conference (ADC2004), Dunedin, NZ. Conferences in Research and Practice in Information Technology, Vol. 27. (2004).
4. Feldman, S., Sherman. C.:The cost of not finding Information. IDC. (2003)
5. Dmitriev,P., Serdyukov, P., Chernov, S.: Enterprise and desktop search. WWW 2010: 1345-1346 (2010)
6. Owens, L.: The Forrester Wave™:Enterprise Search, Q2 . (2008)
7. Dmitriev,P., Eiron, N., Fontoura, M., Shekita, E.:Using Annotations in Enterprise Search. In WWW2006. ACM 1595933239/06/0005. Edinburgh (2006)
8. Zhu, H., Raghavan, S., Vaithyanathan, S., L¨oser.Navigating A.: The intranet with high precision. In 16th international conference on World Wide Web, pp.491–500 (2007)
9. Li, H., Cao, Y., Xu, J., Hu, Y., Li, S., Meyerzon, D.:A new approach to intranet search based on information extraction. In 14th ACMinternational conference on Information and knowledge management, pp. 460–468 (2005).
10. Xue, G., Zeng, H., Chen, Z., Zhang, H., Lu, C.: Implicit link analysis for small web search In 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, pp. 56–63 (2003).
11. Fisher, M., Sheth, A.: Semantic Enterprise Content Management. Practical Handbook of Internet Computing,( 2004).
12. Demartini, G.: Leveraging Semantic echnologies for Enterprise Search, In PIKM’07, ACM 978-1-59593-832-9/07/001, Lisboa(2007).
13. Mukherjee R., Mao. J..: Enterprise search: tough stuff .Qeue 2 (2004)
14. Telcordia Technologies, <http://lsi.research.telcordia.com>
15. Berry, W., Dumais, T., Brien, W.: Using Linear Algebra for Intelligent Information Retrieval. SIAM Review 37:4 (1995), pp. 573–595 (1994)

Brand, M.: Fast Low-Rank Modifications of the Thin Singular Value Decomposition. Linear Algebra and Its Applications **415**: 20–30. (2006)

1. Deerwester, S., Dumais, S., Landauer, T., Furnas, G., Harshman R.: Indexing by latent semantic analysis. J.of the Society for Information Science, 41(6) (1990).
2. Chen, C., Stoffel, N., Post, M., Basu, C., Bassu, D., Behrens, C.:Telcordia LSI Engine: Implementation and Scalability Issues. In 11th Int. Workshop on Research Issues in Data Engineering (RIDE 2001): Document Management for Data Intensive Business and Scientific Applications,Heidelberg (2001)
3. Deerwester, S., Dumais, S., Landauer, T., Furnas , G., Harshman, R.: Indexing by latent semantic analysis. J. of the Society for Information Science, 41(6), (1990)
4. Landauer, T.: Learning Human-like Knowledge by Singular Value Decomposition: A Progress Report. MIT Press, pp. 45–51(1998).
5. Dumais, S., Platt J., Heckerman D., Sahami M.: Inductive Learning Algorithms and Representations For Text Categorization, in ACM-CIKM’98, Maryland (1998).
6. Zukas, A., Price, Robert J.: Document Categorization Using Latent Semantic Indexing. White Paper, Content Analyst Company, LLC (2003)
7. Homayouni, R,. Heinrich, K., Wei, L., Berry, W.: Gene Clustering by Latent Semantic Indexing of MEDLINE Abstracts. Bioinformatics 21, pp.104–115 (2004)
8. Ding, C.: A Similarity-based Probability Model for Latent Semantic Indexing. In 22nd International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 59–65 California (1999).
9. Bartell, B., Cottrell, G., and Belew, R.: Latent Semantic Indexing is an Optimal Special Case of Multidimensional Scaling, Proceedings, ACM SIGIR Conference on Research and Development in Information Retrieval. , pp. 161–167(1992)
10. Fagin, R., Kumar, Ravi., McCurley, K., Novak, J., Sivakumar, D., Tomlin , J., Williamson, D.: Searching the workplace web. In 12th World Wide Web Conference, 1581136803/03/0005, Budapest (2003).
11. McCandless, M., Hatcher, E., Mccandless, M.: Lucene in Action, Manning Publications (2009)
12. Smiley, D., Pugh, Eric.: Solr 1.4 Enterprise Search Server, Packt Publishing (2009)
13. Apache Hadoop <http://hadoop.apache.org/>
14. Apache Lucene <http://lucene.apache.org/solr/>
15. Apache Tika <http://tika.apache.org/>