LATENT SEMANTIC ANALYSIS IN SEARCH ENGINE.

N. URMANOV[*] *Computer Science and Software Engineering, International University of Informational Technology, 050040, Almaty, Kazakhstan*

BEKTEMYSSOVA G.U.[*] *Computer Science and Software Engineering, International University of Informational Technology, 050040,Almaty, Kazakhstan*

ABSTRACT

Latent Semantic Analysis(LSA) method and theory for automatic indexing and implicit higher-order structure of connections term-by-document using singular-value decomposition technique for finding relevant documents on basis of terms found in queries. The main idea is that document is set of words, which have only one idea. The order of words is ignored, only take a note how many times particular word is represented in one document. Through words, which is terms LSA find other documents which have same words – same meaning.

KEYWORDS:*Latent Semantic Analysis, term-by-document matrix, Singular Value Decomposition, Stemming, Dimension reduction, machine learning, document classification, unsupervised machine learning.*

1.INTRODUCTION

Nowadays we cannot imagine world without search engines like Google, Yandex and other big IT-companies. Cause every person who have access to the internet and can use devices search needed question thought Google, where their write their queries.

The popularity of the Internet has caused an exponential increase in the amount of on-line text and in the number of people who create and use this text. As the amount of documents and number of users rise, automatic document categorization becomes an increasingly important tool for helping people organize this vast amount of data.

However, what happen if person does not know what exactly he need or he need more information? One of the answer is to find themes or documents, whichdoes not have key values, which entered by person, but still have same meaning with themes or documents, which found by queries. It increase search range of theme and maybe will have needed information in documents, which have same meaning.

In this article we cluster words into groups specifically for the benefit of document classification. Word clustering methods create new, reduced-size event spaces by joining similar words into groups.

1. Latent Semantic Analysis – theory

Latent Semantic Analysis(LSA) method and technique in natural language processing, in a very high(e.g. 50-1500) dimensional semantic space, of analyzing connections between set of documents and words which they contain by creating term-by-document matrix. This matrix in which each word is one row and it's unique, each document is column. Matrix show which word is represent in particular document and how many times it was used in this document.

LSA is based on singular value decomposition, a mathematical matrix decomposition technique.

LSA, as a text data mining and natural language processing (NLP) technique allows a systematic, computational and comprehensive analysis of a large corpus of literature for matching between query and document at topic level. In addition to the application areas of text data mining and NLP techniques across various disciplines, recent studies also demonstrate the viability of these techniques for the architecture, engineering and construction (AEC) industry. Similar text-based data mining and NLP techniques have been shown to be useful in the AEC industry for various aspects such as information retrieval, increasing the efficiency in decision making, predicting cost over-runs, reducing the human efforts, reducing errors in recognitions, and dispute resolution in construction accidents.[3]

1.1. Singular Value Decomposition

SVD is expansion of a rectangular material or complex matrix having broad application owing to the evident geometrical interpretation, in case of the decision of many application-oriented tasks. The idea of using SVD for singular expansion consists that it

selects key component matrixes, allowing to ignore noise and find terms in text. In SVD, a rectangular matrix is decomposed into the product of three other matrices. One component matrix describes the original row entities as vectors of derived orthogonal factor values, another describes the original column entities in the same way, and the third is a diagonal

matrix containing scaling values such that when the three components are matrix-multiplied, the original matrix is reconstructed. There is a mathematical proof that any matrix can be so decomposed perfectly, using no more factors than the smallest dimension of the original matrix. When fewer than the necessary number of factors are used, the reconstructed matrix is a least-squares best fit. One can reduce the dimensionality of the solution simply by deleting coefficients in the diagonal matrix, ordinarily starting with the smallest. (Figure 1.1) [2]

The SVD followed by the selection of the most relevant singular values is named Truncated Singular Value Decomposition (TSVD). LSA finds a low-rank approximation of the original term-document matrix. Both the original matrix and its approximation can be seen as vectors whose dimensionality is equal to the number of elements of the matrices (which is the same for both). [4]

$$\{X\} = \{W\}\{S\}\{P\}'$$

$$\{W\} =$$

0.22	-0.11	0.29	-0.41	-0.11	-0.34	0.52	-0.06	-0.41
0.20	-0.07	0.14	-0.55	0.28	0.50	-0.07	-0.01	-0.11
0.24	0.04	-0.16	-0.59	-0.11	-0.25	-0.30	0.06	0.49
0.40	0.06	-0.34	0.10	0.33	0.38	0.00	0.00	0.01
0.64	-0.17	0.36	0.33	-0.16	-0.21	-0.17	0.03	0.27
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.30	-0.14	0.33	0.19	0.11	0.27	0.03	-0.02	-0.17
0.21	0.27	-0.18	-0.03	-0.54	0.08	-0.47	-0.04	-0.58
0.01	0.49	0.23	0.03	0.59	-0.39	-0.29	0.25	-0.23
0.04	0.62	0.22	0.00	-0.07	0.11	0.16	-0.68	0.23
0.03	0.45	0.14	-0.01	-0.30	0.28	0.34	0.68	0.18

Figure 1.1 SVD of term-by-document matrix

1.2. Term-Passage Matrix

A large collection of text statistically representative of human language experience is first divided into passages with coherent meanings, typically paragraphs or documents. The collection is then represented as a term-passage matrix. Rows stand for individual terms and

columns stand for passages or documents (or other units of analysis of interest.) Individual cell entries contain thefrequency with which each term occurs in a document. [2]

1.3. Term frequency – inverse document frequency (TF-IDF)

TF-IDF is the most widely used term weight algorithm nowadays. However, it has the following drawbacks as well.

TF-IDF is one of the most commonly used term weighting algorithms in today's information retrieval systems. Two parts of the weighting were proposed by Gerard Salton and Karen Spärck Jones. TF, the term frequency, also called Local Term Weight, is defined as the number of times a term in question occurs in a document. Obviously, it is "single document wide" measurement.[6]

TF-IDF method determines the relative frequency of words in a specific document through an inverse proportion of the word over the entire document corpus. In determining the value, the method uses two elements: TF - term frequency of term i in document j and IDF - inverse document frequency of term i. In our research and testing the algorithm of framework this method showed good results. TFIDF can be calculated as

$$a_{ij} = tf_{ij}\, idf_i = tf_{ij} * \log_2 \frac{N}{df_i}$$

where a_{ij} is the weight of term i in document j, N is the number of documents in the collection, tf_{ij} is the term frequency of term i in document j and df_i is the document frequency of term i in the collection. This formula implemented in the framework, during the testing has shown good results when we used the documents with equal length.[5]

1.4. Transformed Term-Passage Matrix

The entries in the term-document matrix are often transformed to weight them by their estimated importance in order to better mimic the human comprehension process. For language simulation, the best performance is observed when frequencies are cumulated in a sublinear fashion within cells (typically *log*(*freqij*+1), where *freqij*is the frequency of term *i*in

document *j*), and inversely with the overall occurrence of the term in thecollection (typically using inverse document frequency or entropy measures).(Figure 1.2) [2]

	Docs																			
Terms	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
data	1	1	0	0	2	0	0	0	0	0	1	2	1	1	1	0	1	0	0	0
examples	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
introduction	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
mining	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0
network	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1
package	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Figure 1.2 Term-by-document matrix.

1.5. Stop-listing and stemming

These are very rarely used. In keeping with the underlying theory and model, neither stemming nor stop-listing is appropriate or usually effective. As in natural language, the meaning of passages cannot be accurately reconstructed or understood without all of its words. However, when LSA is used to compare word strings shorter than normal text paragraphs, e.g. short sentences, zero weighting of function words is often pragmatically useful. [2]

In stemming, conversion of morphological forms of a word to its stem is done assuming each one is semantically related. The stem need not be an existing word in the dictionary but all its variants should map to this form after the stemming has been completed. There are two points to be considered while using a stemmer:

- Morphological forms of a word are assumed to have the same base meaning and hence should be mapped to the same stem
- Words that do not have the same meaning should be kept separate

These two rules are good enough as long as the resultant stems are useful for our text mining or language processing applications. Stemming is generally considered as a recall-enhancing device. For languages with relatively simple morphology, the influence of stemming is less than

for those with a more complex morphology. Most of the stemming experiments done so far are for English and other west European languages.[7]

1.6. Dimension reduction

A reduced-rank singular value decomposition (SVD) is performed on the matrix, in which the *k* largest singular values are retained, and the remainder set to 0. The resulting representation is the best *k*-dimensional approximation to the original matrix in the least-squares sense. Each passage and term is now represented as a *k*-dimensional vector in the space derived by the SVD. In most applications *k* the dimensionality is much smaller than the number of terms in the term-passage matrix. For most language simulation *k* ,>50 and <1,000 dimensions are optimal, with 300 +/- 50 most often best, although there is neither theory nor method to predict the optimum. It has been conjectured that in many cases, such as language simulation, that the optimal dimensionality is intrinsic to the domain being simulated and thus must be empirically determined. The dimension reduction step performs a powerful induction: a different value for the similarity of every word to every other whether or not they have ever occurred in a common context. [2]

1.7. The mathematics

Consider a rectangular *t,* x, *p* matrix of terms and passages, *X* .Any rectangular matrix can be decomposed into the product of three other matrices using the singular value decomposition. Thus

$$X=T*S*PT \quad (1)$$

is the SVD of a matrix *X* where *T* is a *t,*x,*r*matrix with orthonormal columns, *P* is a *p* x *r* matrix with orthonormal columns, and *S* is an *r* x *r* diagonal matrix with the entries sorted in decreasing order. The entries of the *S* matrix are the singular values (eigenvalue.5), and the *T* and *P* matrices are the left and right singular vectors, corresponding to term and passage vectors. This is simply a re-representation of the *X* matrix using orthogonal indexing

dimensions. LSA uses a truncated SVD, keeping only the *k* largest singular values and their associated vectors, so [2]

$$X=Tk*Sk*PTk \quad (2)$$

1.8. Similarity in the reduced space

Since both passages and terms are represented as vectors, it is straightforward to compute the similarity between passage-passage, term-term, and term-passage. In addition, terms and/or passages can be combined to create new vectors in the space. The process by which new vectors can be added to an existing LSA space is called folding-in. The cosine distance between vectors is used as the measure of their similarity for many applications because of its relation to the dot-product criterion and has been found effective in practice, although other metrics, such as Euclidean or city-block distances are sometimes used. To accurately update the SVD and thereby take into account new term frequencies and/or new terms requires considerable computation; minor perturbations to the original term-by-document matrix *X* can produce different term and passage vectors (and therefore affect precision and recall for query matching).

2. Model of system architecture

Web portal will connect to 2 different database, first is OrientDB, where get access to metadata which produced and write by LSA, second is PostgreSQL database for users data.(Figure 1.3)

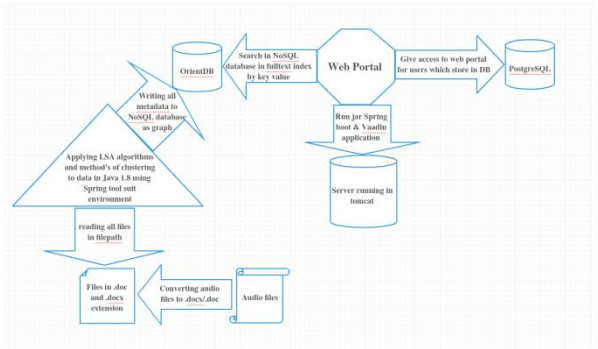

Figure 1.3 Model of collection, transfer, processing and delivery of information

LSA and NLP technique applied a large text corpus. Final metadata written to OrientDB as connection of words anddocuments (Figure 1.4).

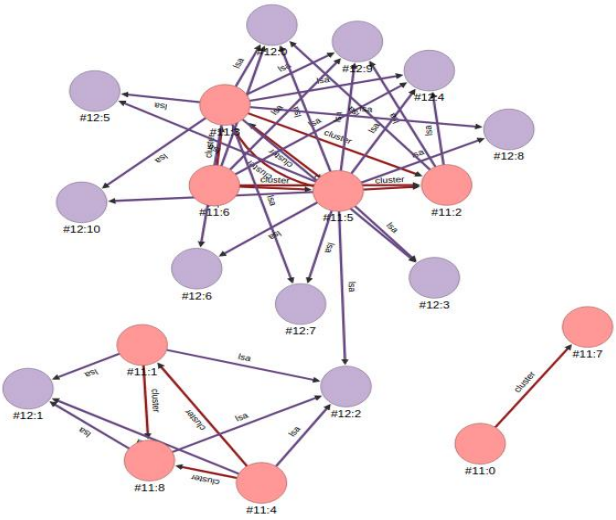

Figure 1.4 Graphical view of metadata.

The system architecture will be based on users which will use search engine. WebUI should have good design and usability. API for writing metadata to database, search data from database and other logical operations. Also all system should controlled by administrator.

2.1. Model of information streams in information (telecommunication) system and its description

In system 2 heterogeneous sources of data : Audio file and documents with .doc or .docx extensions. Audio file will be converted to .txt format and then to .doc or .docx format. Documents with .doc and .docs format will be read in java environment and processed by LSA algorithm and methods of clustering to make metadata of documents. All this metadata will be write to OrientDB(NoSQL) database. User will use visual search portal which will search all needed document from this OrientDB database.

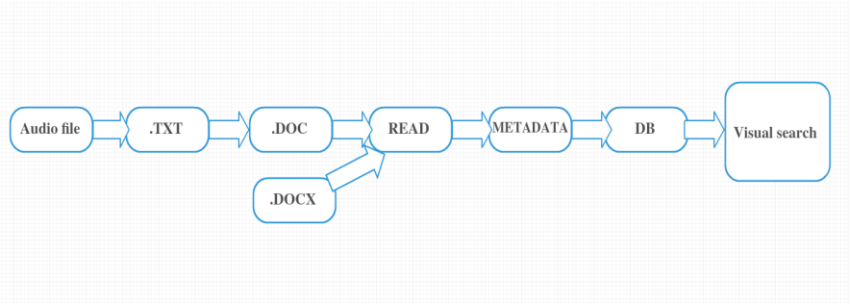

Figure 1.5 Model of streaming and converting data for search engine.

3. EXPERIMENTAL PART

First step is to read all needed data. We read documents only with extensions .doc and .docx in specific folder.

Apply LSA algorithm to read documents. LSA algorithm consist next algorithms:

1. Document Space Reduction. This algorithm uses for creating 2d or more space, which depend to number of dimension reduction, where we store words and documents. (Figure 1.4)

2. Number of dimension to retain in the reduced space. This number is equal to 2. In some researches it equal to 3, but applied for more than 500 documents. In our case we use only more than 200 documents and 2 is more acceptable number for performing the SVD. If number of documents will raise through the time, number of dimensions also raise. For example if number of document will be more than millions, dimension should be between 300 and 1000

3. Transform value, An interface for Matrix transformations. These tranformations should support both a Matrix and a serialized Matrix stored in a File, in one of the supported matrix formats.

Once a matrix has been transformed, use the existing state of this transform to apply the same transformate to this new column vector, as if it had been a part of the input matrix, but without changing this transform's state.In our case we use TF-IDF transform matrix.

4. SingularValueDecomposition reducer, this value means the SingularValueDecomposition algorithm to apply to reduce the transformed term document matrix.

SVD - A refinement of the Matrix Factorization interface for algorithms that compute the Singular Value Decomposition. This interface exposes the three matrices that are generated as a result of the decomposition.

We set Singular Value Decomposition LibC() - this class is implementation of SVD in java using LibC library, which able only in Linux operations systems.

5. BasisMapping<String, String> map variable. An interface for specifying how a set of features can be mapped to a vector basis. In the naive case, each feature is mapped it its own dimension. However, many approaches may use information about the feature to represent several different features using a single dimension. For example, each word may correspond to a unique dimension regardless of how it is grammatically related.

This interface also provides support for describing dimensions. The named of the description is left open to the implementation. For example, an implementation may chose to return a String with a human-readable description. Another implementation may return a Set of features that are represented by the dimension in order to facilitate further processing. Create new StringBasisMapping, A string based. Keys must be strings and each dimensions described by the associated key.

6. Create a mapping for each term that is seen in the document to the number of times it has been seen. This mapping would more elegantly be a SparseArray<Integer> however, the length of the sparse array isn't known ahead of time, which prevents it being used by the MatrixBuilder. Note that the SparseArray implementation would also incur an additional performance hit since each word would have to be converted to its index form for each occurrence, which results in a double Map look-up.

7. Stem each word. Stemming is the process of reducing inflected (or sometimes derived) words to their word stem, base or root form—generally a written word form. Termination is a set of array which is represented by UTF-8 codes.

8. Once all the documents have been processed, performs any post-processing steps on the data. An algorithm should treat this as a no-op if no post-processing is required. Callers may specify the values for any exposed parameters using the properties argument.

9. The Latent Semantic Analysis algorithm is end, and we find all terms of document, location of terms and documents in semantic space. We have their x,y and magnitude value. After LSA, now we will write all needed data to OrientDB(NoSql) database

4. CONCLUSIONS

The work presented in this paper give information about unsupervised machine learning technique – Latent Semantic Analysis, which analyze connections between set of documents and words and cluster them. Clustered metadata recorded in NoSQL OrientDB database for further operation to use in front-end.

ACKNOWLEDGMENTS

I would like to thank all my teachers from International University of Information Technology for giving a lot of experienceand knowledge in master degree courses. Additional many thanks to KourosBasiri, for his courses and motivations for writing this article.

REFERENCES

[1] Landauer T. K., Foltz P. W., &Laham D. (1998), «Introduction to Latent Semantic Analysis.», Discourse Processes 25, pp. 259-284.

[2] Scholar pedia, Latent Semantic Analysis, Available: http://www.scholarpedia.org/article/Latent_semantic_analysis [Accessed: 4 March. 2017]

[3] Mehmet Yalcinkaya , Vishal Singh. (2015), «Patterns and trends in Building Information Modeling (BIM) research: A Latent Semantic Analysis.», Automation in Construction 59 (2015) , pp. 68–80.

[4] GIOVANNI PILATO AND GIORGIO VASSALLO. (2015), « TSVD as a Statistical Estimator in the Latent Semantic Analysis Paradigm.», EMERGING TOPICS IN COMPUTING v3(2015) , pp. 185–192.

[5] Bruno Trstenjaka ,SasaMikacb , DzenanaDonko. (2013), «KNN with TF-IDF Based Framework for Text Categorization.», 24th DAAAM International Symposium on Intelligent Manufacturing and Automation (2013), pp. 1356-1364

[6] Tian Xia, Yanmei Chai (2011), «An Improvement to TF-IDF: Term Distribution based Term Weight Algorithm.», JOURNAL OF SOFTWARE, VOL. 6, NO. 3 (2011), pp. 413-420

[7] Ms. Anjali Ganesh Jivani (2011), «A Comparative Study of Stemming Algorithms.», IJCTA (2011), pp. 1930-1938

Authors	
	Nurbolat Urmanov, 16.09.1995 **Current position, grades:** Master **University studies:** International University of Informational Technology **Scientific interest:** Machine learning, Java programming
	BektemyssovaGulnuraUmitkulovna **Current position, grades:**Candidate of technical sciences, associate professor of the department of Computer Engineering and Telecommunication, **University :**International University of Informational Technology **Scientific interest:**Big data management, Processing Big Data, Machine Learning