OPEN ACCESS
ISSN: 2249-6645

International
Journal
Of Modern Engineering Research (IJMER)

Effective Information Retrieval System

Dr. V. Anandam
Professor, Department of Computer Science Engineering, Vardhaman College of Engineering, Hyderabad dr.anand@vardhaman.org

ABSTRACT:- *The continuing growth of information overflow has made it hard to obtain valuable information on the web. In this trend, the need for effective Information Retrieval (IR) technique has been increased. Although document data contain much more abundant information, users can retrieve necessary information only from the title and description in conventional web services. In order to meet the demands for fast and accurate retrieval of valuable information, we propose a fast and effective content-based document information retrieval system that retrieves the information from the actual content of a document. The proposed method is based on a topic model of Latent Dirichlet Allocation that is used to extract major keywords for a given document. The main contributions of our system are the increased flexibility, effectiveness, and fast retrieval of information. Our system can easily communicate with existing web service through the standard JSON format. In addition, we increase the speed of information retrieval by using NoSQL based database system with inverted indexing and B-tree based indexing. We validate the performance of our system on real data collected from the SlideShare service. The proposed system shows better retrieval performance over the existing IR system.*

Keywords:- *Information Retrieval; CBIR; Topic modeling; LDA; Inverted Indexing; B-tree Indexing*

I. INTRODUCTION

Information Retrieval (IR) [1, 2] has been used in many computer science fields, and plays a significant role on the web [3] owing to the large number of data to retrieve. The conventional web services are provided with IR. Users can upload documents or presentation files along with the title and description. Accordingly, users can search the needed information by searching through the title or description of the content entered with IR [4]. However, its usability and effectiveness are limited in that users have to enter the exact keyword because it is searched only with the given title or description. Furthermore, the title or description may not always contain enough information for the user to search the necessary content. In order to solve this problem, it is needed to provide the actual content information as well as the title and description in IR systems.

For this, in this paper, we propose a Fast and Effective Content based Document Information Retrieval system (CBDIR). The main advantages of our system are more flexible and more effective and faster retrieval of information. In perspective of flexibility, our system can easily communicate with widely used web platforms using the standard JSON format. We extract keywords from the actual content of a document using Latent Dirichlet Allocation (LDA) topic model. As a result, the search performance is improved compared to existing information retrieval systems. Our retrieval system is highly effective in that it uses actual contents as well as its title and description. Our system also provides users with required information in real-time at a faster retrieval speed by using inverted indexing and B-tree based indexing.

II. RELATED WORK

Content-based information retrieval has been studied in a variety of fields such as music [5], image s [6, 7] and documents [8]. In this study, we focus on the documents data. For effective content-based document retrieval, it is necessary to model the data with a topic model such as Probabilistic Latent Semantic Indexing (pLSI) [9] or Latent Dirichlet Allocation (LDA) [10]. The Latent Dirichlet Allocation, a generative topic model, has become very popular and has been used in various fields. There have been some studies using topic modeling in the information retrieval domain, for example, a topic model for ad-hoc information retrieval using LDA (IR) [11, 12]. Considering good performance of LDA model in previous work, we also select LDA as our topic.

For fast information retrieval, we need to construct the database schema efficiently. There are various indexing techniques for database schema. J. Zobel at al. [13] proposed inverted indexing schema for fast file access. E. Gabrilovich and S. Markovitch proposed explicit semantic analysis that represents texts from Wikipedia using inverted indexing [14]. Brian F. D Comer discusses the variations of B-tree [15]. C Von der Weth et al. proposed the indexing technique for NoSQL [16]. Zhu Wei-ping at al. tried to use NoSQL database in order to replace with RDBMS [17]. These approaches help us construct a fast retrieval system. Our system selects NoSQL rather than Relational Database Management System (RDBMS) and adapts indexing schema and B-tree based indexing from these studies.

III. SYSTEM OVERVIEW

The proposed Content Based Document Information Retrieval System (CBDIR) is an information retrieval system that is based on the actual document contents uploaded by users. Here, a document represents any file in Portable Document Format (PDF), DOC, or PPT format. PDF is a file format developed by Adobe Systems, and DOC and PPT are Microsoft MS Office file formats.

Fig. 1 shows the overview of our system. A client sends the title and short description as well as a document as attachment to the server. Note that a client can be regarded as a platform such as a web-browser, Android platform, a web-server like Apache Tomcat, whereas a server is CBDIR that we propose in this work.

CBDIR analyzes the content of information that a client sends to the server, and also constructs database schema for information retrieval. There are five main steps in CBDIR. First Connection part communicates with a client. Second Content Extraction part extracts the content of a document. Next Morpheme Analysis Part Extracts the set of morphemes in the content. Then Keyword Extraction part extracts the meaningful keywords among the set of morphemes. Finally Database Modeling part constructs the database schema using keywords.

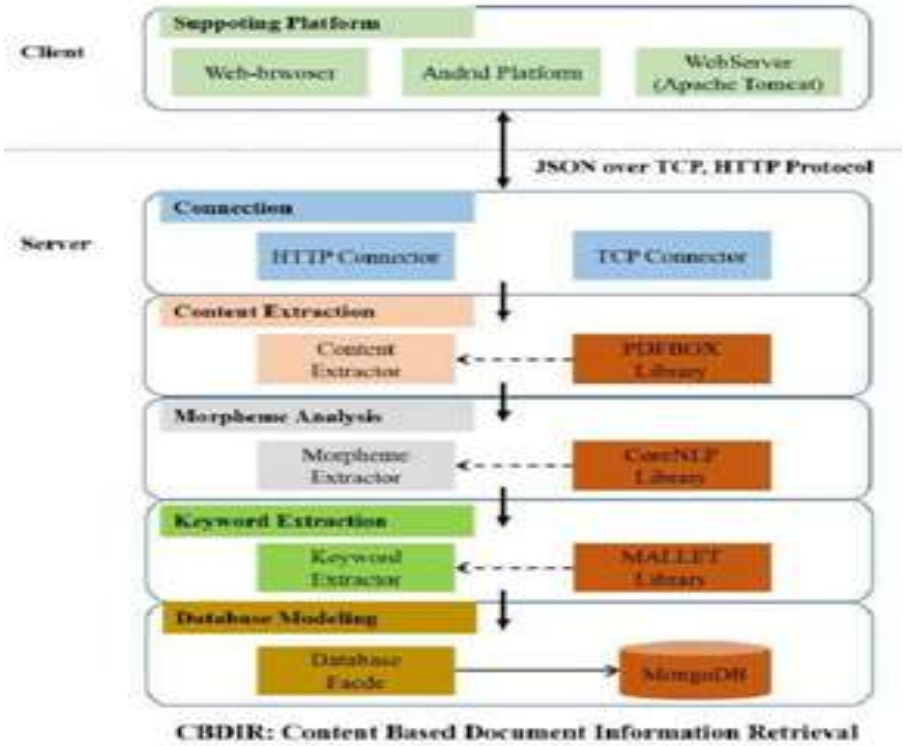

Fig. 1. The overview of CBDIR

IV. METHODOLOGY

A. Connection

In this stage, a client communicates with the CBDIR system. A client sends all the information to the system such as the title, description, and document uploaded by a user, and the system responds to it. The connection is established through a pre-defined protocol such as TCP or HTTP. The data is transmitted with

Java Script Object Notation (JSON) format [18]. JSON is an open standard format for transmitting data between a server and a client. It has a great advantage of being a data format that is independent of language and platform. Namely, it is available in many kinds of platforms and programming languages. We used the title, description, user ID, content ID, document content which is obtained automatically from the JSON format. It depends on the system of a client.

B. Content Extraction

The raw contents in the transmitted document are then extracted. In this paper, we focused on PDF-formatted files. To extract the content of PDF files, we used open source library PDFBOX which is provided by Apache [19].

C. Morpheme Analysis

The extracted raw text needs to be tokenized into morphemes. CBDIR tokenizes the input text into nouns and verbs. Other lexical categories are ignored as a noun and a verb contains the most representative meaning in the text.CBDIR uses publicly available CoreNLP library as a morphological analyzer [20]. For preprocessing, we removed e-mail addresses, URLs, and special characters first. Then, CoreNLP tokenizes the sentence into lexical categories, and also noun or verb is converted into lemma.

D. Keyword Extraction

To extract meaningful keywords from the extracted nouns and verbs, we apply Latent Dirichlet Allocation (LDA) that is the state-of-the-art topic modeling algorithm. We used Mallet open source library for LDA [21]. Our system should process one document promptly. LDA can extract topics from an individual document in real-time. As the title and description already contain user-defined keywords, only the document body content is used for LDA topic modeling.

1) LDA model: LDA is a generative probability model that assumes several topics allocated under Dirichelt distribution in a specific document. Fig. 2 shows the graphical model for LDA that extracts a number of topics and its word distribtuion from a given document. The words with high weights in the extracted topics are considered as highly relevant keywords. As shown in Fig. 2, there are several parameters in LDA model. M is the number of documents. N is the number of unique keywords in the document. The number of nouns and verbs from morphological analyzer are going to be N. W is the observed word and Z is the index to the allocated topic in W. Ö is the topic proportion in the document. α and β are rge Dirichlet priori weights of the topic and words in it respectively. Also we need to set the number of topics to K.

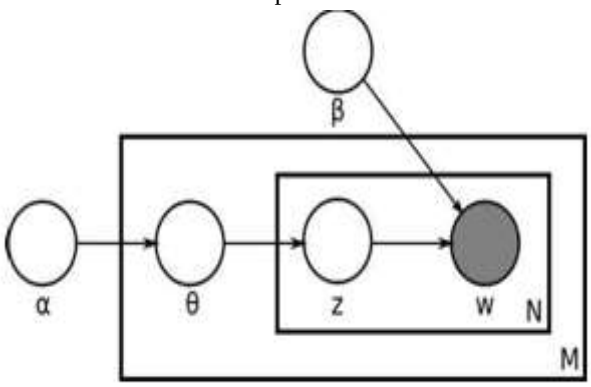

Fig. 2. The graphical model of LDA

2) LDA estimation: As a result of LDA estimation, N words are allocated to K topics. In order to consider distribution of both a topic and words in it, the weights of keywords are obtained in $TotalW$ in equation (1). Note that $TopicP$ is the proportion of topic *k* and W is the weight of word *n* in each topic. $TotalW$ is defined as

$$\text{TotalW}_n = \sum_{i=1}^{K} \text{TopicP}_i \times W_{in} \quad (1)$$

$$\text{where } TopicP_k = \frac{\alpha_k}{\sum_{i=1}^{K} \alpha_k}, \; W_{k_i} = \frac{\#\, of\, Word_{kn}}{\sum_{j=1}^{N} \#\, of\, Word_{ki}} \quad (2)$$

$$\text{where } \sum_{L_1}^{K} TopicP_i = 1, \; \sum_{i=1}^{N} W_{ki} = 1 \quad (3)$$

We used the top *P* keywords in ascending order of Total Wn. Then, we evaluated the performance of our system that is based on the extracted *P* keywords.

E. Database Modeling

We create database with extracted *P* keywords from LDA modeling. MongoDB based NoSQL is used as a database system. MongoDB has the virtue of flexibility, performance and scalability [22, 23]. MongoDB can store any type of data with key-document schema different from RDBMS. Thus, inverted indexing which is widely used in IR is adapted easily [24].

Inverted indexing takes words in content as a key and the content ID is stored in the key. Accordingly, the extracted keywords are stored in keys and document content is searched and stored by those keys. However, as inverted indexing yields too many keywords to process in real-time, we solved this issue by using B-tree based indexing that is provided by MongoDB. Therefore, our system can process continuously increasing data quickly with inverted indexing and B-tree based indexing.

V. EXPERIMENTAL RESULTS

A. Data

To evaluate the performance of CBDIR, we collected data using SlideShare Search API [25] that is composed of a title, description, and contents of each document. Users share PDF, PPT, DOC formatted documents with a title and description in Slide Share, and thus it is appropriately used in our system.

For data consistency, we used six categories among the top ten strategic technology trends from Gartner 2015: Computing everywhewe, Advances Analytics, Internet of Things,3D Printing, Software Defined and Webscale.

We manually collected documents that are relevant to the category. The number of documents in each category is shown in Table I. To make it consistent, we only selected English documents. Also, if the document does not contain enough texts, we removed them. For example, as a presentation file is more likely to contain images rather than texts, it is removed.

TABLE 1: THE NUMBER OF DOCUMENTS IN EACH CATEGORY

Category	# of Document
Computing everywhere	20
Advanced Analytics	15
Internet of Things	18
3D printing	15
Software Defined	25
Web Scale	14
Total	**107**

B. Extracted Keywords

1. *Ground-truth:* We manually collected keywords based on a title, description, and the content of the document in six categories. In order to reduce subjectivity, at least two persons extracted keywords in each category and only the overlapping keywords are considered to be relevant as ground-truth. The number of keywords for each category is shown in Table II.

TABLE 2: THE NUMBER OF RELEVANT KEYWORDS IN EACH CATEGORY

Category	# of keyword
Computing everywhere	25
Advanced Analytics	14
Internet of Things	14
3D printing	12
Software Defined	24
Web Scale	17

2) LDA: Our system needs to determine the parameters in LDA such as the number of topis K. The parameter setting for LDA is summarized in Table III.

TABLE 3: THE PARAMETER SETTING IN LDA

Parameter	Value
α	1
β	0.1
K	5
# of Iteration	800

In Fig. 3, the extracted keywords from LDA estimation are shown in each category. The word size and color in the wordcloud follows Total Wn. We easily identify that 3D printing category in Fig. 3(A) contains many relevant words such as print, material and manufacturing. The other categories also contain many relevant words.

TABLE 4: THE JACCARD SIMILARITYOF CBDIRAND TITLE & DESCRIPTION IN EACH CATEGORY

Category	Similarity
Computing everywhere	0.52
Advanced Analytics	0.53
Internet of Things	0.57
3D printing	0.50
Software Defined	0.69
Web Scale	0.40

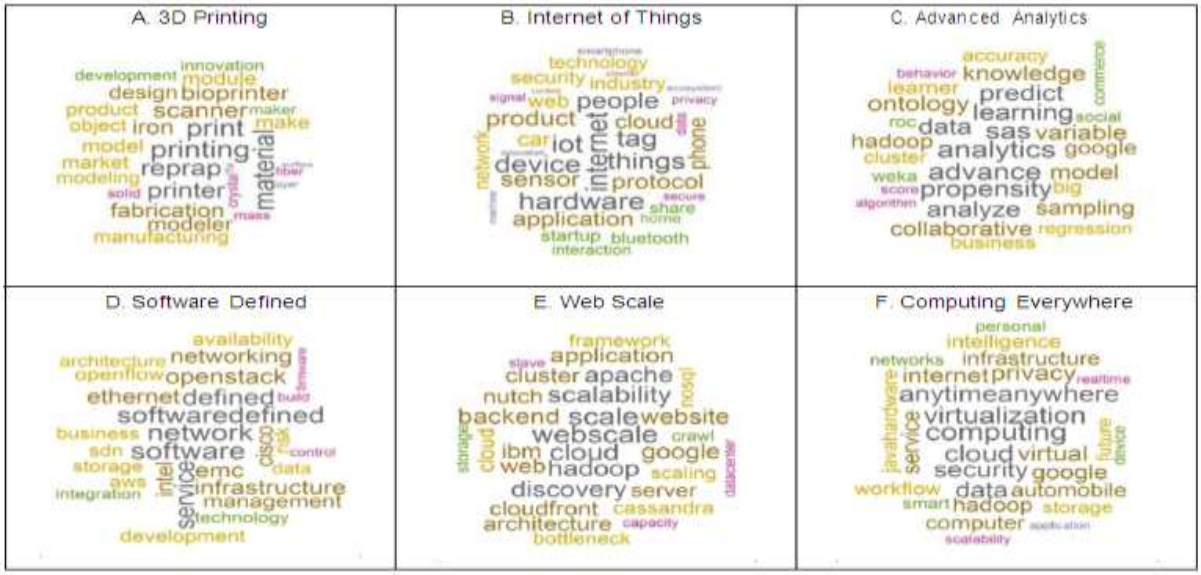

Fig. 3. The word clould in each category extracted from CBDIR (A: 3D Printing, B: Internet of Things, C: Advanced Analytics, D: Software Defined, E: Web Scale, F: Computing Everywhere)

C. Evaluation

We evaluated the retrieval performance of our system in terms of the precision, recall and F-measure.

$$\text{Precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|} \quad (4)$$

$$\text{Recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|} \quad (5)$$

$$\text{F} - \text{measure} = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (6)$$

Here, the retrieved documents are the number of documents that are retrieved from the extracted keywords in CBDIR, and the relevant documents are the number of documents which are truly relevant as ground truth. Note that if there is no retrieved document, all the precision, recall and F-measure are 0.

Precision means the proportion of relevant documents among all retrieved documents. Recall means the proportion of retrieved documents among all relevant documents. F-measure is the harmonic meaning of precision and recall. In Table V, the result of 3D printing category that shows the best performance is summarized for each keyword.

We show the performance of our system as a function of the number of used keywords (P) in terms of precision, recall, and F-measure in Fig. 4, 5 and Table VI, respectively. We compared the performance of CBDIR that contains the title, description and the document content with the one using only the title and description as a baseline to see how much improvement can be achieved by adding content information.

TABLE 5: THE PERFORMANCE OF EACH KEYWORDS IN 3D PRINTING

Keyword	Precision	Recall	F-measure
3D	1	0.86	0.92
Printer	0.23	0.75	0.35
Printing	0.75	0.9	0.82
Bio-printing	1	1	1
Reprap	0.25	1	0.4
Model	0.25	1	0.4
Scanning	1	1	1
Modeler	1	1	1
Market	1	1	1
Remote control	0.5	1	0.67
chemistry	1	1	1
Chemical structure	0.34	1	0.5
Average	0.69	0.96	0.78

In this experiment, we found that the performance on 3D printing category is about twice better than those from other categories. This is because other categories such as ▪computing everywhere▪ or ▪web-scale▪ are quite broad and they contain keywords from various kinds of areas. But 3D printing contains more specific keywords than other categories. Despite of these limitations, CBDIR shows better performance overall than the baseline.

In Fig. 4, the average precision of CBDIR improves the performance up to 20% compared to the baseline. With the increase ofthe number of keywords, the performance is likely to converge at around 25 keywords. However, it gets lower in category ▪Internet of Things▪ and ▪Web scale▪ because irrelevant keywords can also be extracted in those categories.

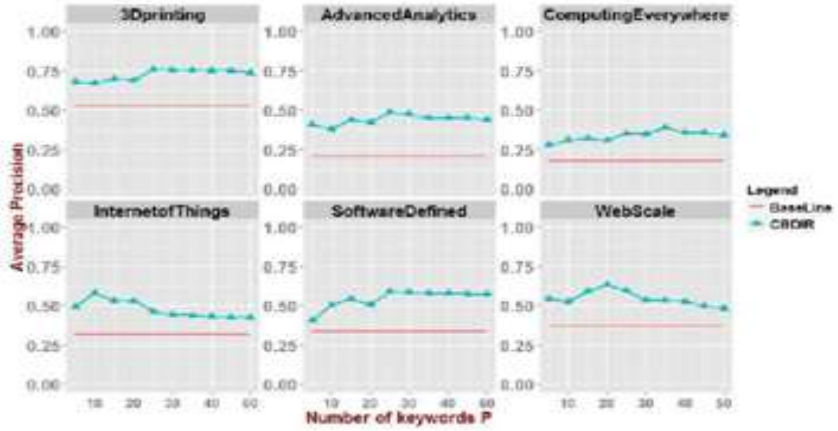

Fig. 4. Average precision in each category across the number of keywords (P) (blue line: CBDIR, red line: baseline (title description))

In Fig. 5, it is seen that CBDIR improves the average recall by up to 30%. Different from the precision, the recall goes up as the number of keywords increases. The reasoning is that if we use more keywords, the number of retrieved documents increases accordingly while the number of relevant documents remains consistent; thus only the numerator of recall increases.

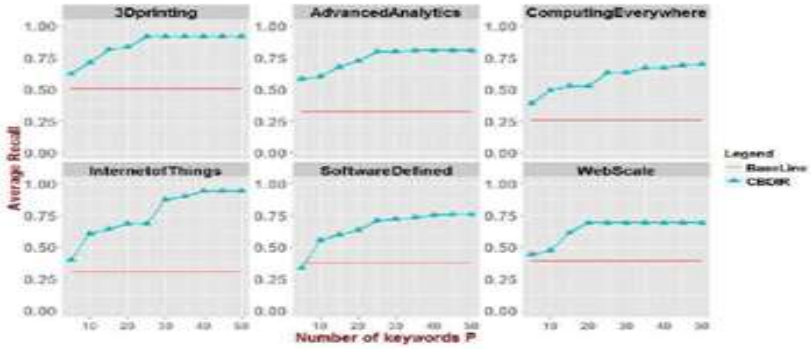

Fig. 5. Averaged recall in each category across the number of keywords (P) (red line: CBDIR, blue line: baseline (title+description))

The overall result shows that the performance using around 25 keywords are almost the same as the one using 50 keywords. As the size of data highly affects the computation speed and maintenance cost which are very important in IR, our system can be applied in real world effectively.

We also calculated F-measure in each category when 25 keywords are used as shown in Table VI. As expected, all of the categories show improvements over the baseline. Though Advanced Analytics and Computing Everywhere showed similar performance as the baseline, Advanced Analytics showed the biggest improvements and Computing Everywhere showed the lowest in our system. It demonstrates that the performance gain of our system is more dominant in a specific area than a broad area.

Table VI. Average f-measure in each catgory in cbdir with 25 keywords and baseline

Category	Baseline	CBDIR (25 keywords)
Computing everywhere	0.20	0.38
Advanced Analytics	0.22	0.58
Internet of Things	0.27	0.50
3D printing	0.50	0.78
Software Defined	0.30	0.57
Web Scale	0.33	0.50

The computation time is a significant factor in IR. We compared the number of keyword scans in each database as shown in Table VII. Note that the number of scanning is the number of the key which is scanned by MongoDB. To demonstrate the retrieval efficiency of our system, we compared it with the baseline and the CBDIR without B-tree based indexing. The baseline or the CBDIR without B-tree based indexing needs scanning of all the keywords in the database. But CBDIR with B-tree based indexing only retrieves the keyword from B-tree indices instantly. Therefore, the proposed system using inverted indexing and B-tree based indexing can be used efficiently for IR systems which should be computed in realtime.

TABLE VII. THE NUMBER OF SCANNING THROUGH DATABASE

	Baseline	CBDIR without the B-tree index	CBDIR
number of scanned	1256	2014	1

VI. CONCLUSION

In this paper, we proposed a fast and effective Content Based Document Information Retrieval (CBDIR) system. We evaluated the CBDIR system with real data and investigated its performance improvement over the baseline. Based on the experimental results, we showed that our system has three main advantages. First, our system is easily adaptable to existing systems. It can easily communicate with any type of clients using JSON format. To support various types of data, we used MongoDB based NoSQL. Second, we extracted keywords in document content effectively using LDA topic model and showed improvements in overall performance. As there is not enough information in a title and description, our system also retrieves the information of content from documents. Finally, by optimizing the number of keywords P that are extracted from document content, we successfully improved the retrieval speed.

Also we validated the efficiency of our system by comparing it with the one without B-tree based indexing and the baseline that does not use any indexing schema. The result shows that our system is significantly better than the existing systems with respect to both the overall performances and the retrieval speed.Although our system shows much better performance than the baseline, there are rooms for further validation and improvement. First, larger sized data would be needed for evaluation to induce more reliable results. Second, the computation time of LDA estimation could be reduced to construct the database more quickly. Third, our system needs more storage because we use more information from the document content. In our future work, other topic modeling techniques such as Hierarchical Dirichlet Processes (HDP) [26] and explicit semantic analysis [27] will be investigated for further improvement. We will also work on the improvement of LDA estimation speed for faster database construction.

REFERENCES

[1]. E. Greengrass, "Information retrieval: A survey," 2000.

[2]. W. B. Croft, D. Metzler, and T. Strohman, Search engines: Information retrieval in practice: Addison-Wesley Reading, 2010.

[3]. C. V. Forecast, "Cisco Visual Networking Index: Global Mobile data Traffic Forecast Update 2009-2014," Cisco Public Information, February, vol. 9, 2010.

[4]. C. Zhai and J. Lafferty, "Two-stage language models for information retrieval," in Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, 2002, pp. 49-56.

[5]. M. A. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney, "Content-based music information retrieval: Current directions and future challenges," Proceedings of the IEEE, vol. 96, pp. 668-696, 2008.

[6]. V. N. Gudivada and V. V. Raghavan, "Content based image retrieval systems," Computer, vol. 28, pp. 18-22, 1995.

[7]. M. S. Lew, N. Sebe, C. Djeraba, and R. Jain, "Content-based multimedia information retrieval: State of the art and challenges," ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP), vol. 2, pp. 1-19, 2006.

[8]. G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," Information processing & management, vol. 24, pp. 513-523, 1988.

[9]. T. Hofmann, "Probabilistic latent semantic indexing," in Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, 1999, pp. 50-57.

[10]. D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," the Journal of machine Learning research, vol. 3, pp. 993-1022, 2003.

[11]. L. Azzopardi, M. Girolami, and C. Van Rijsbergen, "Topic based language models for ad hoc information retrieval," in Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on, 2004, pp. 3281-3286.

[12]. X. Wei and W. B. Croft, "LDA -based document models for ad-hoc retrieval," in Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, 2006,pp. 178-185.
[13]. J. Zobel, A. Moffat, and R. Sacks-Davis, "An efficient indexing technique for full-text database systems," in PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES, 1992, pp. 352-352.
[14]. E. Gabrilovich and S. Markovitch, "Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis," in IJCAI, 2007, pp. 1606-1611.
[15]. D. Comer, "Ubiquitous B-tree," ACM Computing Surveys (CSUR), vol. 11, pp. 121-137, 1979.
[16]. C. Von der Weth and A. Datta, "Multiterm keyword search in NoSQL systems," Internet Computing, IEEE, vol. 16, pp. 34-42, 2012.
[17]. Z. Wei-ping, L. Ming-Xin, and C. Huan, "Using MongoDB to implement textbook management system instead of MySQL," in Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on, 2011, pp. 303-305.
[18]. D. Crockford, "The application/json media type for javascript object notation (json)," 2006.
[19]. J. P. PDFBox, "processing Library," Available: http://www. pdfbox. org.
[20]. K. Toutanova, D. Klein, and C. Manning, "Stanford Core NLP," ed: The Stanford Natural Language Processing Group. Available: http://nlp. stanford. edu/software/corenlp. shtml. Accessed, 2013.
[21]. McCallum, Andrew Kachites, "MALLET: A Machine Learning for Language Toolkit," Available: http://mallet.cs.umass.edu, 2002.
[22]. K. Banker, MongoDB in action: Manning Publications Co., 2011.
[23]. B. G. Tudorica and C. Bucur, "A comparison between several NoSQL databases with comments and notes," in Roedunet International Conference (RoEduNet), 2011 10th, 2011, pp. 1-5.
[24]. K.-P. Lee, H.-G. Kim, and H.-J. Kim, "A social inverted index for social-tagging-based information retrieval," Journal of Information Science, vol. 38, pp. 313-332, 2012.
[25]. SlideShare Corp. (2006), "SlideShare developer search api," Available: http://www.slideshare.net/developers.
[26]. Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei, "Hierarchical dirichlet processes," Journal of the american statistical association, vol. 101, 2006.
[27]. O. Egozi, S. Markovitch, and E. Gabrilovich, "Concept-based information retrieval using explicit semantic analysis," ACM Transactions on Information Systems (TOIS), vol. 29, p. 8, 2011