Topic Model based Approach for Improved Indexing in Content based Document Retrieval

Moon Soo Cha, So Yeon Kim, Jae Hee Ha, Min-June Lee, Young-June Choi, Kyung-Ah Sohn
Department of Information and Computer Engineering, Ajou University,
206, World cup-ro, Yeongtong-gu, Suwon-si, Gyeonggi-do,443-749, Republic of Korea
Email : {ckanstnzja, jebi771, lovesm135, tlsdmq12, choiyj, kasohn}@ajou.ac.kr

Abstract

Information Retrieval system plays an essential role in web services. However, the web services in which users can upload files as attachments typically do not support enough search conditions and often rely only on the title or the description that the users provide during upload. We present a topic-model based framework for fast and effective Content Based Document Information Retrieval that retrieves the information from the actual contents in the attachment. The proposed systems is analyzed and compared with conventional methods in various aspects. In particular, we propose an efficient keyword extraction method based on Latent Dirichlet Allocation which is compared with the Term Frequency Inverse Document Frequency approach typically used in conventional systems. Moreover, a per-category indexing structure is also proposed and compared with the existing total indexing scheme. Our experimental results validate the utility of the proposed system for web services that can upload document attachments.

Keywords: Information Retrieval; CBIR; TFIDF; LDA; Per-Category Indexing; Inverted Indexing;

1. Introduction

Information Retrieval (IR) [1, 2] plays an important role in World Wide Web services and has been developed in many computer science fields [3]. The web services provide users with necessary information through various retrieval systems such as Lucene, Solr and Elastic Search. Usually, the user can retrieve the information using search conditions such as a title, a description or an author based on the stored database [4].

However, the conventional information retrieval systems often suffer from certain limitation that arises from ignoring the actual contents of the documents. For example, when a user is searching a document related to a specific topic, he/she has to rely on the title, description and the author information provided by the one who uploads the document, and not directly on the full content of the document. According to the search conditions, the user has to enter the exact and correct keywords. Moreover, those limited search conditions may not provide enough information for the user to search the document. To resolve this issue, we propose a new approach for Content Based Document Information Retrieval (CBDIR) that uses a fast and effective indexing technique for improved information retrieval performance. It makes use of the embedded content in the document in addition to the title and description. To show the effectiveness of our system in terms of keyword extraction, we perform an extensive validation using various models for content analysis such as the vector space model, the probability model, Term Frequency Inverse Document Frequency (TFIDF) [5] and Latent Dirichlet Allocation (LDA) [6]. We also evaluated the result with the per-category indexing structure which is used in our system with a total indexing that is commonly used in a conventional system.

2. Related Work

Content-based information retrieval has been studied in various domains such as music [7, 8], images [9, 10] and documents [11]. In this study, we focus on the embedded content in a document. For effective content-based document retrieval, both the vector space model and the topic model such as Term Frequency Inverse Document Frequency [12], Probabilistic Latent Semantic Indexing (pLSI) [13] and Latent Dirichlet Allocation (LDA) have been used. The LDA, a generative topic model, has been widely used in many different kinds of area. For information retrieval, a topic model for ad-hoc information retrieval using LDA (IR) has been proposed [14, 15]. As the LDA model is proved to have better performance than other methods, we utilized the LDA model for topic modeling.

For fast information retrieval, the efficient database schema construction is required. There are various indexing techniques for the database schema construction. Kenta Funaki at al. [16] proposed a parallel indexing scheme for processing a large amount of data efficiently. J. Zobel at al. [17] proposed an inverted indexing scheme to speed up the file access time. E. Gabrilovich and S. Markovitch [18] proposed an explicit semantic analysis to represent the text from Wikipedia using inverted indexing. Brian F. D Comer [19] discussed the variations of B-tree based indexing scheme. C Von der Weth et al. [20] proposed the indexing technique for NoSQL. Zhu Wei-ping at al. [21] utilized NoSQL database in place of Relational Database Management System (RDBMS). Considering the pros and cons of the existing methods, we construct the database for this study with NoSQL rather than RDBMS and adapt the inverted indexing scheme with B-tree based indexing.

3. Content Based Document Information Retrieval (CBDIR)

We propose Content Based Document Information Retrieval (CBDIR) system that is an information retrieval system based on the embedded text in a document in a Portable Document Format (PDF), a Microsoft document file format (DOC) and a Power Point file (PPT) format. Fig. 1 shows the overview of our system that consists of both client and server. The client can vary such as a web browser, Android platform and a web-server. It sends the information such as a title, description and the content of a document.

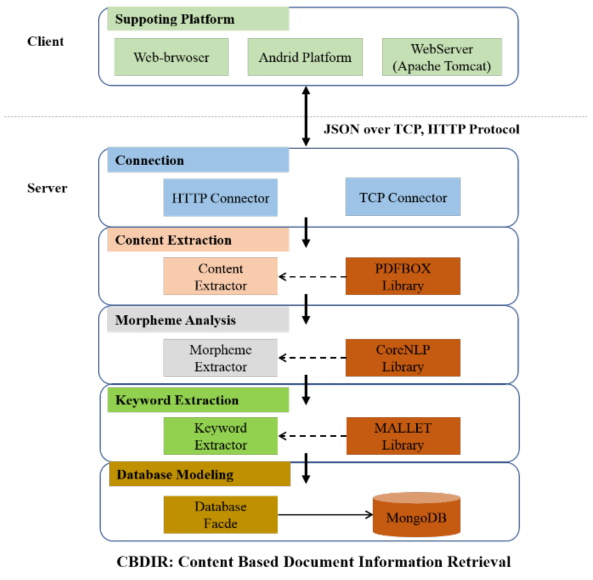

Fig. 1. The system structure of CBDIR

The server includes five main layers that are Connection, Content Extraction, Morpheme Analysis, Keyword Extraction and Database Modeling.

First layer is Connection that communicates with the client. The server receives the data with Java Script Object Notation (JSON) format [21] through protocol such as TCP or HTTP because JSON format has an advantage in that it is a platform independent and also a language independent. The Content Extraction layer extracts the raw text of the document using an open source PDFBOX library [22]. Morpheme Analysis layer tokenizes the text into lexical units using CoreNLP library [23]. In this stage, we only extracted nouns and verbs as they contain more representative meaning than the other lexical categories. Also, we removed email addresses, URLs and special characters for data pre-processing.

Keyword Extraction layer is the most important one because it extract meaningful keywords. For an effective keyword extraction, we utilized the LDA model that is a state-of-the-art topic modeling algorithm with the implementation in Mallet library [24]. In the final layer, we stored the extracted keywords in MongoDB of NoSQL database management system [25, 26]. For an efficient search, we used the inverted indexing structure and B-tree index [27]. The details of the Keyword Extraction layer and the Database Modeling layer will be provided in the next section.

4. Methodology

4.1. *Data Collection*

In a SlideShare Web service, the user can easily share their presentation files with others. To share their slides, PDF files are uploaded with the title and the description which summarize their slides. As it is the most widely used platform, we collected the data using a SlideShare Search API [29].

For data consistency, we extracted the data among six categories from Gartner's top-10 strategic technology trends for 2015. The selected categories are 'Computing Everywhere', 'Advanced Analytics', 'Internet of Things', '3D Printing', 'Software Defined' and 'Web Scale'. Each category name is used as a query for retrieving documents in the SlideShare Search API.

We manually selected relevant document written in English and removed the documents in other languages. We also excluded the documents that do not contain enough text to extract. Table 1 shows the number of documents in each category we used in our experiment.

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**

4.2. *Keyword Extraction Model*

To extract meaningful keywords for a given document, we used TFIDF for the vector space model and LDA for the probability model. These are commonly used in information retrieval and each model has its advantages and disadvantages. While LDA can extract topics from an individual document in real-time, TFIDF needs to extract the keywords by considering all the documents simultaneously. Additionally, while TFIDF considers the relation of words between documents, LDA does not reflect this relation.

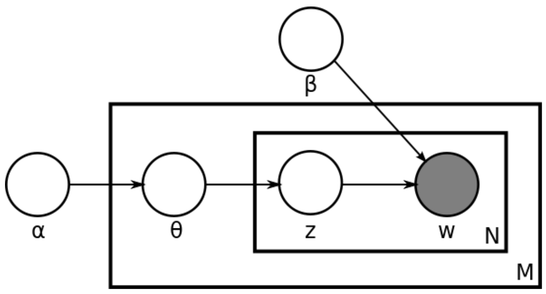

Fig. 2. LDA Graphical model

4.2.1 TFIDF Model

TFIDF is often used as a weight model in information retrieval framework such as Lucene, Solr and Elastic Search. We apply a standard TFIDF formula which is defined as:

$$\mathrm{TFIDF} = \mathrm{TF}(t,d) * \mathrm{IDF}(t,D) \quad (1)$$

$$\mathrm{TF}(t,d) = \frac{f_{t,d}}{\max f_{t,d}} \quad (2)$$

$$\mathrm{IDF}(t,D) = \log\left(\frac{D}{n_t} + 1\right) \quad (3)$$

where $f_{t,d}$ is the number of times that term t appears in a document d; $\max f_{t,d}$ is the total number of times that all terms appear in a document d; D is the total number of documents; n_t is the number of documents where the term t appears. Furthermore, we applied inverse frequency smooth as IDF that is the logarithmically scaled fraction of the documents. We note that the document represents the content of it uploaded by the user and we consider either nouns or verbs as terms.

4.2.1 LDA Model

While the conventional information retrieval framework does not support topic modeling as weighting schema, LDA can extract keywords considering the weights of them in a document. It is a generative probability model that allocates K topics under Dirichlet distribution in a document. The words with high weights in the extracted topics are considered as highly relevant keywords.

Fig. 2 shows the graphical model for LDA. There are several parameters in LDA model. M is the number of documents. N is the number of unique keywords in the document which is the number of nouns and verbs from morphological analyzer in our system. W is the observed word and Z is the index to the allocated topic in W. Θ is the topic proportion in the document. α and β is the Dirichlet priori weights of the topic and words in it, respectively. The number of topics is set to K.

As a result of LDA estimation, N words are allocated to K topics. We can obtain $TotalW_N$ that is calculated with both topic proportion and word weight and it follows the equation (4). Note that $TopicP_k$ is the proportion of topic k and W_{kn} is the weight of word n in each topic.

$$\mathrm{TotalW_n} = \sum_{i=1}^{K} \mathrm{TopicP_i} \times \mathrm{W_{in}} \qquad (4)$$

$$\text{where } TopicP_k = \frac{\alpha_k}{\sum_{i=1}^{K} \alpha_k}, W_{kn} = \frac{\#\,of\,Word_{kn}}{\sum_{j=1}^{N} \#\,of\,Word_{ki}} \qquad (5)$$

$$\sum_{i=1}^{K} TopicP_i = 1\,,\ \sum_{i=1}^{N} W_{ki} = 1 \qquad (6)$$

We can select the top P keywords in ascending order of $TotalW_N$

4.3. *Database Index Structure*

We constructed our database with the inverted indexing structure and B-tree index in MongoDB. The inverted indexing structure is widely used in IR and can be easily adapted. Because it is a key value schema design, keywords in document are stored as a key and document identifications are saved as a value. However, it has limitation that it yields too many number of keywords, thus makes it hard to retrieve them efficiently. Thus we used the B-tree indexing to resolve this issue.

In addition, we experiment with per-category indexing and total indexing. In total indexing, which has been used in conventional information retrieval systems, keyword is stored in one index database. On the other hand, we propose a per-category indexing structure. In per-category indexing, keyword is separately stored for each category in each index database.

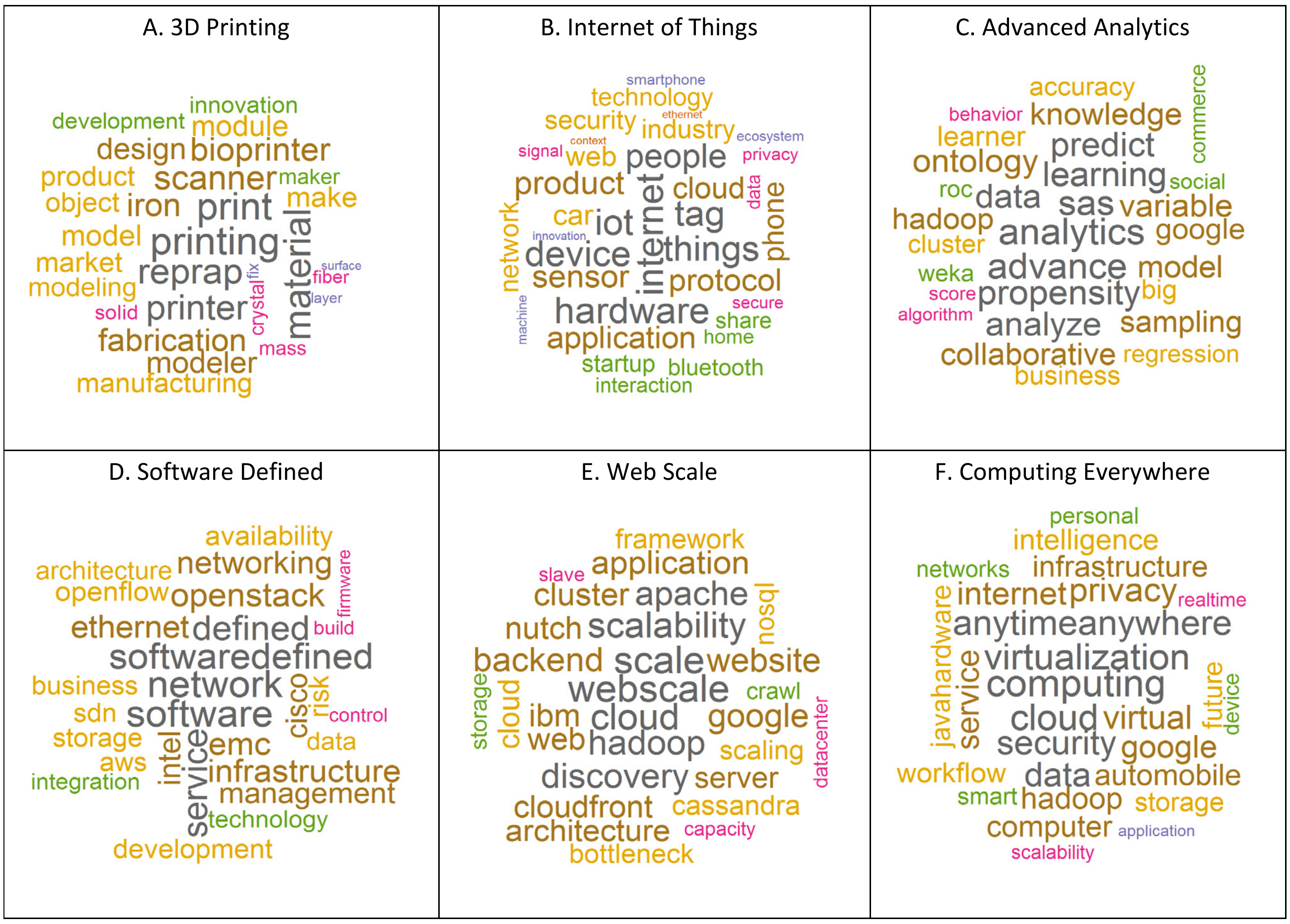

Fig. 3. The word cloud in each category extracted from CBDIR (A: 3D Printing, B: Internet of Things, C: Advanced Analytics, D: Software Defined, E: Web Scale, F: Computing Everywhere)

5. Experimental Results

5.1. *Extracted Keywords*

5.1.1 Ground Truth

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

To evaluate the performance, we manually selected relevant keywords in the title, description and content of the document for each category. In order to reduce the subjectivity, at least two computer science experts have chosen the relevance keywords for each category and then the overlapping keywords are used as ground-truth. Table 2 shows the number of relevant keywords for each category.

5.1.3 CBDIR

Our system needs to determine the optimal model parameters such as the number of topics K in LDA. The parameter setting for LDA is summarized in Table 3.

In Fig. 3, the extracted keywords from LDA estimation are shown in each category. The word size and color in the word-cloud varies depending on $TotalW_N$. We easily identify that 3D printing category in Fig. 3(A) contains many relevant words such as ‘print’, ‘material’, and ‘manufacturing’. Likewise, Fig. 3 shows that the other categories also contain many relevant words.

Table 3. The parameter setting in LDA.

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

5.2. *Evaluation*

We evaluated the retrieval performance of our system in terms of the precision, recall and F-measure defined as follows:

$$\text{Precision} = \frac{|\{\text{relevant documents}\}\cap\{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|} \quad (7)$$

$$\text{Recall} = \frac{|\{\text{relevant documents}\}\cap\{\text{retrieved documents}\}|}{|\{\mathit{relevant}\ \text{documents}\}|} \quad (8)$$

$$\text{F}-\text{measure} = \frac{2\times Precision \times Recall}{Precision+Recall} \quad (9)$$

Note that 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 mean of precision and recall.

In the equation (7), (8) and (9), 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 relevant as a ground truth. We note that if there is no document to be retrieved, the precision, recall and F-measure are all zero.

5.2.1 CBDIR Evaluation

The performance of CBDIR is evaluated for each keyword in each category. Table 4 describes the performance on 3D Printing category as on example. We calculated the average precision, recall and F-measure in each keyword.

Table 4. The performance on each keyword in 3D printing category.

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
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**

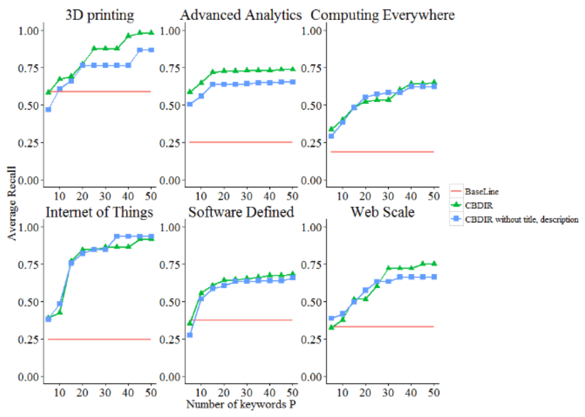

Fig. 4. Performance Evaluation - Average Recall in each category across the number of keywords (red: baseline, green: CBDIR, blue line: CBDIR without the title and description information)

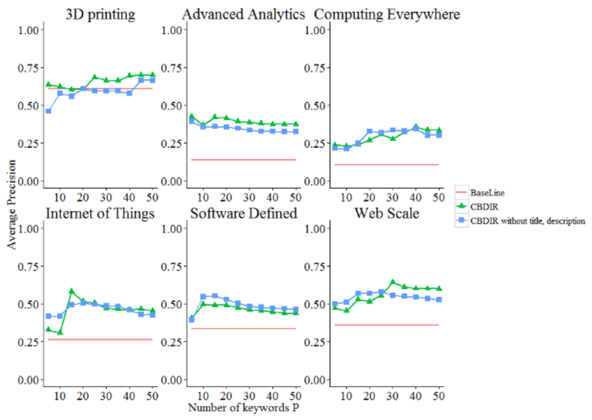

Fig. 5. Performance Evaluation - Average Precision in each category across the number of keywords (red: baseline, green: CBDIR, blue line: CBDIR without the title and description information)

Then we examine the performance along with a varying number of used keywords P in six categories including 3D Printing, Advanced Analytics, Computing Everywhere, Internet of Things, Software Defined and Web Scale. The performance is evaluated from 5 to 50 extracted keywords using LDA weight schema. The comparison is done with the conventional system using only the title and the description without using indexing scheme or topic modeling as a baseline. Note that the baseline performance does not change regardless of the number of extracted keywords. To compare the contribution of actual contents on CBDIR performance, we also consider the performance of CBDIR that does

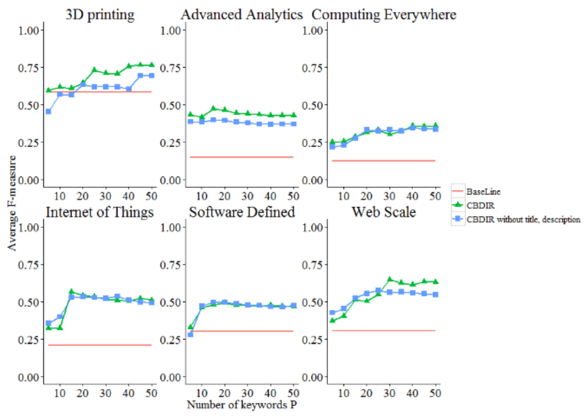

Fig. 6. Performance Evaluation - Average F-measure in each category across the number of keywords (red: baseline, green: CBDIR, blue line: CBDIR without the title and description information)

not use the title and description, but relies only on the content. The overall result shows the competitive improvement by adding the content information in a document.

In Fig. 4, the average recall can reach 97% in CBDIR whereas that of the baseline is 58% in the 3D Printing category. With the increase of the number of keywords, there was a steady rise in average recall from 5 to 50 keywords. While there is little difference between CBDIR and CBDIR without the title and description information, there is slight improvement in 3D Printing, Advanced Analytics and Web Scale. In this result, we show that CBDIR can also extract the important keywords without title and description. Also, we demonstrate that the content of the document has a high impact on the improvement.

Similarly as in the average recall, there was substantial improvement in average precision by 20% from CBDIR to the baseline as shown in Fig. 5. Likewise, the average F-measure of CBDIR is also increased in CBDIR as shown in Fig. 6.

In CBDIR evaluation, we found that the performance on 3D Printing category is more than twice Computing Everywhere categories by 40% from about 23 % to about 63 % in each measure and other categories also is similar. This is because other categories such as ‘computing everywhere’ or ‘web-scale’ cover a wider range and they contain various kinds of keywords in different areas. Meanwhile, 3D printing contains more specific keywords than other categories.

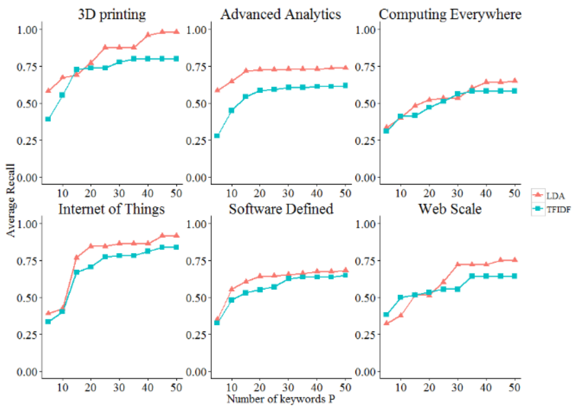

Fig. 7. Comparison of Keyword Extraction Model - Average Recall in each category across the number of keywords (red: LDA, blue: TFIDF)

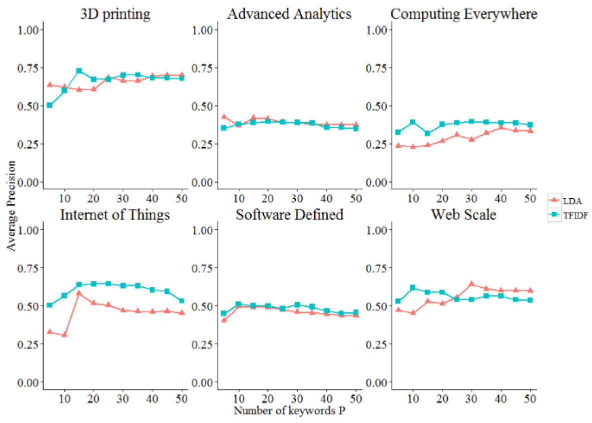

Fig. 8. Comparison of Keyword Extraction Model - Average Precision in each category across the number of keywords (red: LDA, blue: TFIDF)

The overall result shows that the performance is likely to converge when about 25 keywords are used as the number of keywords increases. 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.

5.2.2 Keyword Extraction Model Comparison

We then compare the keyword extraction model based on LDA in CBDIR with the one using TFIDF weighting scheme. Note that most of the information retrieval system has been adapted TFIDF weight schema.

Fig. 7 compares the average recall between LDA and TFIDF models. There was a gradual increase in average

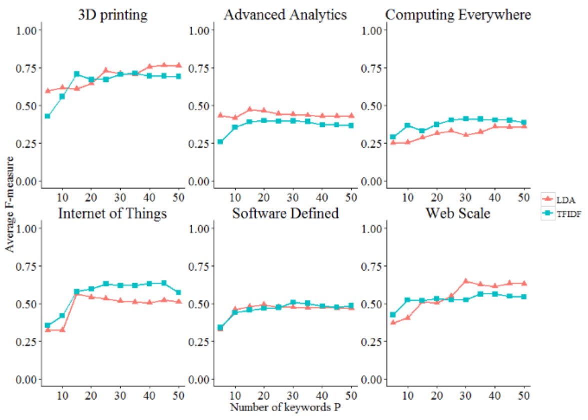

Fig. 9. Extracting Keyword Model Evaluation - Average F-measure in each category across the number of keywords (P) (red line: LDA, blue line: TFIDF)

recall from 5 to 50 keywords in both LDA and TFIDF. The average recall in LDA was generally higher than that of TFIDF in four categories excluding ‘Computing Everywhere’ and ‘Web Scale’. Likewise, Fig. 8 shows the average precision between LDA and TFIDF. There is no big difference in both LDA and TFIDF. The average precision in TFIDF overall is greater than that of LDA.

In Fig. 9, because F-measure is the harmonic mean of recall and precision, there is not big difference between LDA and TFIDF. In Advanced Analytics, LDA has better performance than TFIDF. On the other hand, TFIDF is higher performance than LDA in Computing Everywhere, Internet of Things. Also, performance in other categories are similar.

Precision may play an important role in the retrieval system because users are easy to find relevance document in retrieved documents. Although TFIDF is slightly better than LDA in terms of Precision, the difference is marginal and considering other aspects as well, LDA can also be adapted as weighting scheme in information retrieval. In particular, TFIDF needs to re-compute the term weights using all the documents whenever a new document comes, while LDA can extract the keywords considering the new specific document only.

5.2.3 Indexing Structure Evaluation

As most of the information retrieval system has commonly used total indexing structure, we compared the per-category indexing scheme we proposed with the total indexing to evaluate its performance. The performance comparison measured as the average F-measure in these two indexing structures is shown in Fig. 10 and 11 with LDA and TFIDF, respectively.

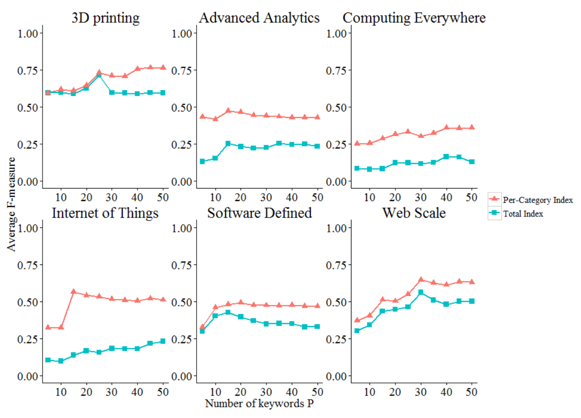

Fig. 10. Indexing Structure Evaluation In LDA - Average F-measure in each category across the number of keywords (P) (red line: Per-Category Index, blue line: Total Index)

Fig. 10 shows that the performance from per-category indexing is substantially better than that from total indexing in all the categories. With the increase of the number of keywords, the performance is likely to be improved in all the categories except for 3D Printing. 3D printing category has almost the same average F-measure after more than 25 keywords are extracted.

As shown in Fig. 11, average F-measure in TFIDF shows very similar result as in LDA. We find that if the data can be included in a specific category, per-category indexing structure is better than total indexing. For per-category indexing structure, category keyword takes priority over other keywords. Also, there are similar keyword of stored keywords than total indexing structure.

5.2.4 Computation Power

The computation power is a significant factor in IR. These factor is related to the number of keywords stored in a database. We analyzed the number of keywords in the database with total indexing and per-category indexing.

Table 5. The number of the stored keywords in total index structure

Method	Number of the stored keywords
Baseline	1256
CBDIR with LDA (25 keywords)	1706
CBDIR with TFIDF (25 keywords)	2042
CBDIR with LDA (50 keywords)	2663
CBDIR with TFIDF (50 keywords)	3504

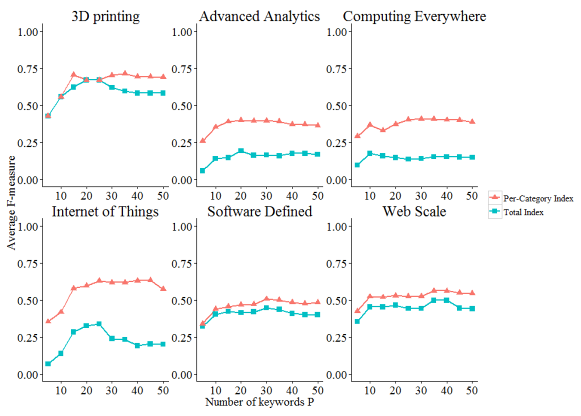

Fig. 11. Indexing Structure Evaluation In TFIDF - Average F-measure in each category across the number of keywords (P) (red line: Per-Category Index, blue line: Total Index)

Table 5 shows the number of the stored keywords in the total index structure with various methods including baseline, CBDIR with LDA (25 keywords and 50 keywords) and CBDIR with TFIDF (25 keywords and 50 keywords). Baseline method stored the smallest number of keywords, because it only uses the title and description. Also, CBDIR with TFIDF has more number of stored keywords than that of LDA. It shows that LDA is more efficient than TFIDF. In addition, if we only used 25 keywords which performs similar as 50 keywords, it is more efficient because it uses less stored keywords than 50 keywords.

Table 6. The number of the stored keywords in per-category index structure

Category	LDA (Keywords P)		TFIDF (Keywords P)	
	25 P	50 P	25 P	50 P
3D Printing	429	701	403	719
Advanced Analytics	360	564	339	595
Computing Everywhere	451	717	459	806
Internet of Things	395	668	412	722
Software Defined	466	765	524	448
Web Scale	332	559	355	646
Total	2433	3974	2492	3936

Table 6 shows the number of the stored keywords in per-category indexing structure. LDA leads to a less number of stored keywords than TFIDF in three categories such as Internet of Things, Software Defined and Web Scale, while TFIDF has a less number of the keywords than LDA in the other categories.

When we compare the category indexing and total indexing, per-category indexing structure stores much less number of keywords. The average F-measure of category indexing is also much higher than that of total indexing as shown in Fig. 10 and 11. It is because per-category indexing structure actually retrieves the query in each category index, thus the number of retrieved keywords is much less than that from total indexing.

6. Discussion

In this work, we analyzed CBDIR in various aspects. Especially, we found the following observations. First of all, both CBDIR and CBDIR without using the title and description show superior performance than that of the baseline. Also, CBDIR has better performance than CBDIR not using the title and description information. Second, the performance from TFIDF and LDA weighting scheme for keyword extraction is very similar. As we mentioned earlier, TFIDF weight schema considers the document relation, but it needs to go through the entire documents to extract the keywords when a new document is uploaded. In contrast, LDA extracts the keyword independently in each document, so it does not need to re-do the calculations in the same situation. Therefore, the usage of LDA can greatly enhance the efficiency of the system. Third, per-category indexing structure improves the performance over the total indexing scheme. Also it has a good performance in terms of search speed.

As a result, the final CBDIR is based on LDA as weighting schema on all the title, description, and the contents information together along with the per-category indexing structure. Also, our system should be maintained in situations with continuous content uploads. However, the selection depends on data characteristic and system development environment.

7. Conclusion

In this paper, we analyzed and showed the effectiveness of the proposed CBDIR system with real data and various perspectives. The proposed system can be easily applied in web service that supports data uploading. We shows that the performance of CBDIR without title and description is similar with that of CBDIR. It shows that the content of the document is the most important factor for improving the retrieval performance. The performance between TFIDF and LDA is also very similar. Even though most of the IR system showed the TFIDF weighting scheme, we select LDA, because it is useful for real-time in information retrieval. We also demonstrated that per-category indexing structure showed better performance than total indexing when the data is categorized.

Moreover, there still remain possibilities for further improvement. First, larger number of data would be needed for evaluation to show more reliable results. Second, the computation time of LDA estimation can be reduced to construct the database more quickly. Third, our system needs more storage because there is a lot of information in the document.

In our future work, other topic modeling techniques such as Hierarchical Dirichlet Processes (HDP) [30] and explicit semantic analysis [31] will be investigated for further improvement. We can also speed up the LDA estimation time for faster database construction.

Acknowledgements

This research was supported by the Basic Science Research Program through the National Research Foundation (NRF) of Korea funded by the Ministry of Science, ICT, and Future Planning (MSIP) (2014R1A1A3051169 and 2010-0028631).

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. Joachims, Thorsten, A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization, CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF COMPUTER SCIENCE, No. CMU-CS-96-118, 1996.
6. D. M. Blei, A. Y. Ng, and M. I. Jordan, Latent dirichlet allocation, the Journal of machine Learning research, vol. 3, 2003, pp. 993-1022.
7. 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, 2008, pp. 668-696.
8. Zhang D. Jangam, A. Zhou L., & Yakut I, Context-Aware Multimedia Content Adaptation for Mobile Web. International Journal of Networked and Distributed Computing, 3(1), 2014, pp. 1-10
9. V. N. Gudivada and V. V. Raghavan, Content based image retrieval systems, Computer, vol. 28, 1995, pp. 18-22

10. 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, 2006, pp. 1-19.
11. G. Salton and C. Buckley, Term-weighting approaches in automatic text retrieval, Information processing & management, vol. 24, 1988, pp. 513-523.
12. RUI Yong, HUANG Thomas S, MEHROTRA Sharad, Content-based image retrieval with relevance feedback in MARS, In: Image Processing, 1997. Proceedings., International Conference on. IEEE, 1997, pp. 815-818.
13. 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.
14. 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.
15. 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.
16. Kenta Funaki, Teruhisa Hochin, Hiroki Nomiya, Hideya Nakanishi, Parallel Indexing Scheme for Data Intensive Applications, International Journal of Networked and Distributed Computing, 3(2), 2015, pp. 89-98
17. 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.
18. E. Gabrilovich and S. Markovitch, Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis, in IJCAI, 2007, pp. 1606-1611.
19. D. Comer, Ubiquitous B-tree, ACM Computing Surveys (CSUR), vol. 11, 1979, pp. 121-137.
20. C. Von der Weth and A. Datta, Multiterm keyword search in NoSQL systems, Internet Computing, IEEE, vol. 16, 2012, pp. 34-42.
21. 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.
22. D. Crockford, The application/json media type for javascript object notation (json), 2006.
23. J. P. PDFBox, processing Library, Available: http://www. pdfbox. org.
24. 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.
25. McCallum, Andrew Kachites, MALLET: A Machine Learning for Language Toolkit, Available: http://mallet.cs.umass.edu, 2002.
26. K. Banker, MongoDB in action: Manning Publications Co., 2011.
27. 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.
28. 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, 2012, pp. 313-332.
29. SlideShare Corp. (2006), SlideShare developer search api, Available: http://www.slideshare.net/developers.
30. 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.
31. Egozi, S. Markovitch, and E. Gabrilovich, Concept-based information retrieval using explicit semantic analysis, ACM Transactions on Information Systems (TOIS), vol. 29, pp. 8-20