Large Scale Cross-media Data Retrieval based on Hadoop

Wenchen Cheng Jiang Qian
Beijing University of Posts and Telecommunications
Beijing, China
vcheng_dmt@163.com
jqian104@126.com

Zhicheng Zhao Fei Su
Beijing University of Posts and Telecommunications
Beijing, China
zhaozc@bupt.edu.cn
sufei@bupt.edu.cn

Abstract—With the rapid development of the Internet and speedy increase of the data size, there are more and more data intensive applications which often involve hundreds of megabytes of data. It is important and necessary to obtain the retrieval results from cross-media data quickly and accurately. Large scale cross-media data retrieval based on Hadoop is proposed to speed up the retrieval in this paper. We divide cross-media feature extraction and cross-media retrieval into paralleled pipeline, and implement with the combination of the HDFS, HBase and MapReduce framework. To verify the performance of the proposed method, comparisons with stand-alone mode on different sizes of the image dataset are conducted, and the experimental results demonstrate the good performances of proposed method, which sharply decreases time-consuming, and meanwhile keeps the same query precision.

Keywords：cross-media; image retrieval; Hadoop; MapReudce

I. INTRODUCTION

With the explosive growth of web data, there is urgent need for retrieving semantically relevant contents effectively and efficiently. It is of great challenge to analyze, manage and organize these huge multimodal data effectively.

In general, there are two kinds of image retrieval systems: text-based image retrieval (TBIR) and content-based image retrieval (CBIR). TBIR systems are designed based on the text-based image retrieval [1, 2, 12, 18]. They compare the similarities of the textual query and the surrounding texts of web images, and then use text-based database management systems to perform image retrieval. But they suffer from the mismatch between textual descriptions and web images. Some researchers solve this problem by using the relevant textural information. Authors in [3] and [5] proposed an approach using the scene text. In [4], the textual information was used to describe the annotated images in the document. A new approach was proposed in [6] which can handle noise in the loose labels of training images, to learn a robust classifier for text-based image retrieval. As we know, visual information is a main source of human to obtain information from the objective world. CBIR uses the information of the images themselves, that is, images are indexed by their own visual content, such as color, texture or shape [18].CBIR has become more and more important with the advance of computer technology. But there is also a gap between the human perception and the image features used to describe its content. A retrieval method using textual information retrieval techniques was proposed in [14], such as vector space model, and the authors in [15-17] proposed a retrieval method based on relevant feedback. Wen Li et al. [6] presented methods to query large on-line image databases using the images' content as the basis of queries. W. Niblack et al. [7] designed a system for web-based applications. Wu Yi et al. [8] integrated text-based and content-based image retrieval. Homgmei Tang et al.[9] described a method and a system that automatically constructs an initial face image database for a person using textual evidence obtained from the web, and then used this database to identify images of that person. A metric access method combining M-tree with the pivot-based approach was proposed in [11].

As we know, there are two phases of an image retrieval system: feature extraction and retrieval. Both of them are CPU-consuming and time-consuming, because the feature extraction and matching are complex. Furthermore, on large scale data size, the performance and adaptability of image-based retrieval system are still an open question.

To satisfy the requirements to obtain the retrieval results from cross-media data quickly and accurately, large scale image retrieval algorithm based on Hadoop platform is proposed in this paper. To speed up the retrieval, Hadoop distributed system is developed, and cross-media feature extraction and retrieval are divided into paralleled pipelines. Our algorithm is implemented with the combination of the HDFS, HBase and MapReduce framework. To verify the performance of the proposed method, the comparisons with stand-alone mode on different image dataset sizes are conducted. Experimental results demonstrate that the method sharply speed up the extraction and retrieval of the images, and meanwhile keeps the same query precision.

The remaining sections are organized as follows. Section 2 gives the brief introduction of Hadoop. Section 3 details the proposed method. Section 4 demonstrates the experimental results. The summary and conclusions are given in the last section.

II. HADOOP

A. HDFS

As a distributed file system for mass data and large files, HDFS is the main carrier to realize mass data storage, which can make the data block storage to each computing node [23]. HDFS stores large files across multiple machines. It achieves reliability by replicating the data across multiple hosts.

An HDFS cluster has two types of node operating in a

QSHINE 2015, August 19-20, Taipei, Taiwan

DOI 10.4108/eai.19-8-2015.2260108

master-worker pattern: a namenode (the master) and a number of datanodes (workers). The namenode manages the file system, whose namespace contains the file system tree and the metadata for all the files and dictionaries in the tree. This information is stored persistently on the local disk in the form of two files: the namespace image and the edit log. The namenode also knows the datanodes on which all the blocks for a given file are located. However, it does not store block locations permanently, since that information will be reconstructed from datanodes when the system starts.

In the operation of the system, the namenode stores the metadata in memory in order to reduce the time of searching the metadata, which leads the number of files stored in HDFS system will depend on the size of namenode memory.

Datanodes are the workers of the file system. They store and retrieve blocks when they are told to (by clients or the namenode), and they report back to the namenode periodically with lists of blocks that they store.

B. HBase

HBase is a distributed database, which is oriented to columns, developed from HDFS [22]. It is suitable for real-time reading of large scale data sets. It is constructed from top to bottom and can accomplish linear expansion by simply adding nodes. HBase is not a relational database which means it does not support SQL. But in the specific problem space, it can do what the RDBMS can't manage the large scale sparse table in the cluster formed in cheap hardware.

The physical architecture of HBase, corresponding to that of HDFS, is known as a kind of master and slave form, which the main node is called Hmaster and the child node Regionserver. The master node is mainly responsible for monitoring the entire state of HBase cluster, distributing list (HRegion) to a registered child nodes and handling with the possible failure of sub node. And the child nodes are mainly responsible for the operation and maintenance of HRegion and the reading and writing operation of the client.

C. MapReduce

MapReduce, a distributed computing framework issued publicly by Google Company[20], is a model running on a distributed cluster in processing large data sets and has been used in many scenarios. In Google, more than ten thousand different projects have adopted MapReduce to accomplish the algorithm, including large-scale graphics processing, word processing, data mining, machine learning and many other fields. The implementation of current software is to assign a Map(map) function, which is used to make a set of key-value pairs mapped into a new set of key-value pairs, and a Reduce(reduce)function, which is used to ensure that all mapping key-value pairs in each group share the same key.

The main steps in detail about the MapReduce program are as following:

(1) Partition of document un-processing. It will divide thefiles to be processed into specific input splits (Input Split), and generate some mapping functions depended on the different Input Splits. Each Mapper performs the specific mapping function for allocated Input Splits.

(2) The Map stage. The Mapper subroutine will execute the Map function for the allocated Input Splits and generate the key-value pairs as <KEY, Value> form. The Map function writes into memory in a buffer way and does the pre-ordering in consideration of the efficiency reasons. Once the buffer content reaches the threshold, a background thread begins to write the content on the disk. The Combiner will be executed before writing the output on the disk and merge the output of Map to reduce the time for transferring and the subsequent disk operation if the intermediate Combiner is set. Combiner can be regarded as a local Reducer, which runs in the Map stage. As long as there is a Map task completed, Reduce begins to copy its outputs and gets into the replication stage.

(3) The Reduce stage. Reducer executes after the completion of Map stage and receives all the intermediate results. The result introducing to Reduce will be associated in accordance with the value of key so that the inputs with the same key will be put together. The Reducer will be traverse the value list in the same value of key in accordance with the Reduce function specified by the user, and then generate the output as the design of the Reducer function.

III. THE PROPOSED METHOD

In recent years, the retrieval accuracy has been improved in the CBIR system, but there are still some disadvantages of the stand-alone mode. First of all, there are some repeated works in the CPU-based algorithm either in feature extraction or retrieval. As we know, we need to extract features of all images in the library one by one, and in the retrieval stage, the similarly measure of the two images is needed to compute repetitively. Secondly, on large scale data size, the pressure of the load may be huge if all the features are stored in the stand-alone version of the database. That is to say, if all the extracted features are centralized in local, the pressure of the CPU will be increased accompany with the computing allocation, which will slow down the retrieval speed dramatically.

To solve the above problems, a parallel algorithm in the MapReduce frame is proposed in this paper. We focus on increasing the speed of image feature extraction and retrieval, while the storage strategy is concentrated on decreasing the pressure of the data concentration. Specially, the storage method used in HBase fits for the "Localization" of the MapReduce frame, which can decrease the transmission time dramatically. The structure of the proposed scheme is shown in Fig. 1.

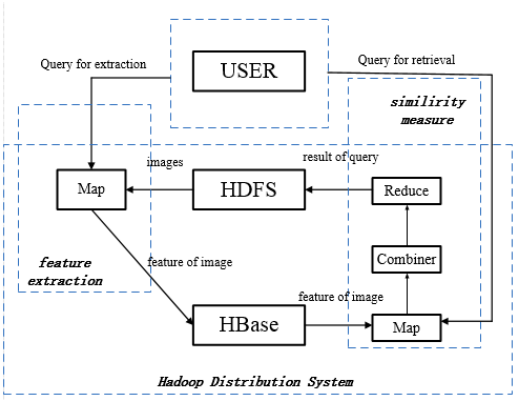

Figure 1. The structure of the proposed scheme

A. *Database Structure*

The database contains two parts in our proposed scheme corresponding to the storage of original image files and extracted features individually.

The former one takes use of the HDFS, while the latter stores those extracted features into HBase. The storage module for original images makes fully use of characteristics of HDFS. The storage and backup will follow the principle of HDFS, which can accord with the "Localization" for the Map phase. Storing the image features in Hbase simplifies the whole system, because image features will be scattered in various Region Server nodes, which can avoid the above centralization problem increasing the speed of the retrieval. The table of HBase is designed as follow: the line number is the image location, and the column family is the feature of image that stores the value of feature.

B. *Extraction of Image Feature Based On Hadoop*

（1） Image features and similarity measure

In this paper, the common used features relating to the color, texture and shape are adopted, including color histogram, Hu's invariant moments and Gray-level co-occurrence matrix of the image. The feature dimension is 87, including 64 dimensional color histogram, 7 dimensional Hu moments and 16 dimensional texture feature.

The similarity measure between different images can be carried out by calculating the correlation of the feature vectors. The greater the value, the more similar between images. The Reduce phase in Hadoop needs to calculate the similarity and output the result after sort.

In this paper, the cosine similarity is used as the similarity measure. Assume that the color histograms of image I and J are V(I) and V(J) respectively, the cosine similarity is computed using the Eq. (1).

$$sim(V(I),V(J)) = \cos\theta = \frac{V(I)\bullet V(J)}{\|V(I)\|\bullet\|V(J)\|} \quad (1)$$

where $\bullet$ represents the inner product between vectors, and $\|\cdot\|$ represents the L2-norm.

Suppose the similarity of the color histogram, Hu's invariant moments, and texture feature are *M*, *N* and *K* respectively. The corresponding weights of these three features are *u*, *v* and *w*. The final similarity measure is obtained using Eq. (2):

$$S = \frac{(M\bullet u + N\bullet v + K\bullet w)}{(u+v+w)} * 100\% \quad (2)$$

The weight parameters *u*, *v* and *w* can be set through cross-validation.

（2） Feature Extraction Based on Hadoop

In the module of MapReduce, transmission and allocation modules are time consuming. In this paper, a scheme of feature extraction based on Hadoop is proposed to decrease the transmission and allocation time from two aspects as shown in Fig.2.

First, an index file is generated whose content corresponds to the path of images before feature extraction. Second, we redesign the InputFormat class by overriding the split method and omitting the Reduce phrase as well. In detail, the module first calls the image feature extraction procedure, and then JobTracker initializes the job and schedules as assigning the Map task to each TaskTracker to perform. The program is processed as follows:

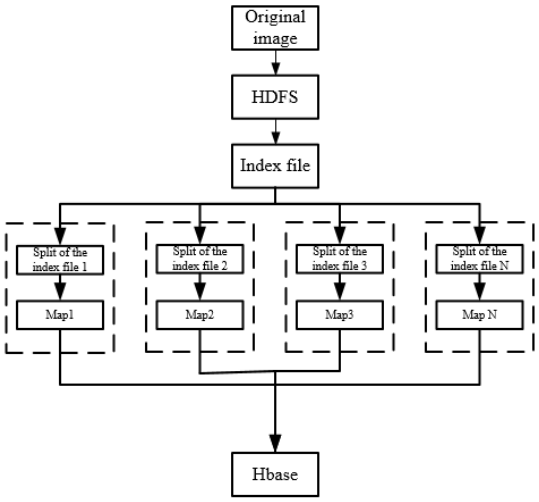

Figure 2. Feature Extraction based on Hadoop

C. *Image Retrieval Module*

The input of image retrieval module is from HBase. To save the data transmission time to Map task, in our scheme, the datanode and Hbase are set to the same machine. Furthermore, the image retrieval can be realized as a TOP-N problem. It will generate N results in sort and stores the retrieval result in the HDFS. The image retrieval module based on Hadoop is shown in Fig.3.

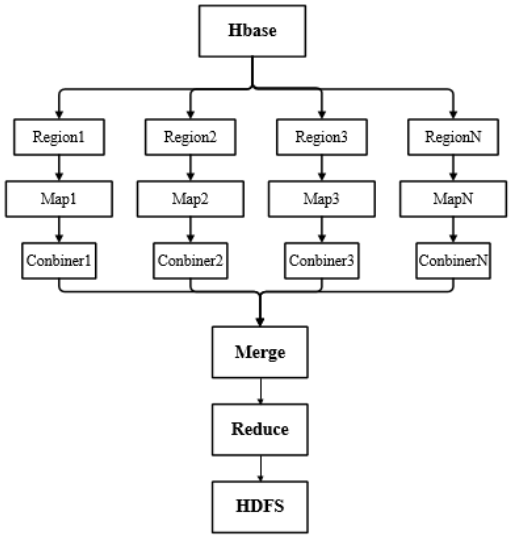

Figure 3. Image Retrieval based on Hadoop

IV. EXPERIMENTAL RESULTS AND ANALYSIS

To verify the performance of the proposed method, we experimentally evaluate it in the database including 400, 000 images. All the images in corel10000 [http://wang.ist.psu.edu/docs/related.shtml] are included, and others are downloaded from the Internet.

In our scheme, we extract the image feature by the distributed parallel computing, build the library of the original image on the HDFS and store the extracted feature on the HBase database. Meanwhile, we retrieve the image through the MapReduce framework. Comparisons with stand-alone mode on different sizes of Hadoop cluster and different scale of image library are conducted. In our experiments, the environment of Hadoop includes 6 slave nodes. The specific configuration of Namenode and Datanode is shown in the Table 1.

Table 1 Configuration of clusters

Configuration	Value
Machine Model	Dell PowerEdge R410
Operating System	Ubuntu 12.04.2
Hadoop Version	Hadoop-0.20.2
Java Version	JDK-1.6

A. *Comparison of Image Feature Extraction Time*

To verify the effect of the image feature extraction based on Hadoop, the time-consuming comparisons of feature extraction in different image library scales are shown in Fig.4. Here, the number of datanodes used in the Hadoop is 6.

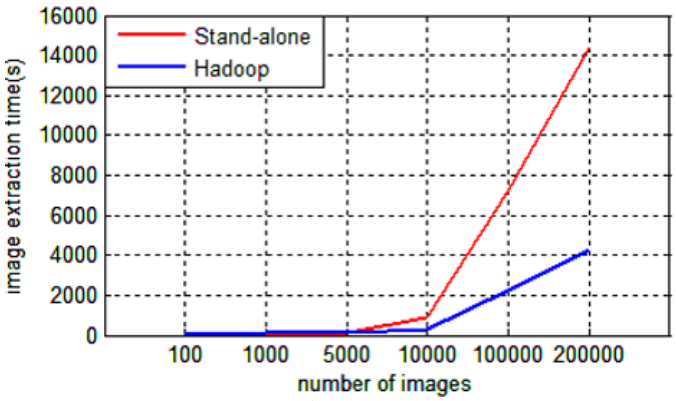

Figure 4. Comparison of Image Feature Extraction time

Since the proposed scheme uses multiple datanodes for parallel computing, it is obviously that the image feature extraction based on Hadoop has remarkable advantages compared to the stand-alone module, especially in the large scale image library. As can be seen, with the increase of the amount of images, the time-consuming of the two module shows a linear growth, while the Hadoop one maintains in 1/4 of the stand-alone.

B. *Comparisions of the different number of datanodes used in Hadoop*

In this section, the comparisons of the different number of datanodes used in Hadoop are given. Experimental results are shown in Fig.5. Here, the image library size is fixed to 50000. As we think, the time of feature extraction decreases in case the number of datanodes used in Hadoop increasing. Furthermore, we find that the performance of the system doesn't improve in accordance with the number of nodes in a linear growth. That is because there is also some time cost in switching on the MapReduce program, scheduling of the task and writing into HBase.

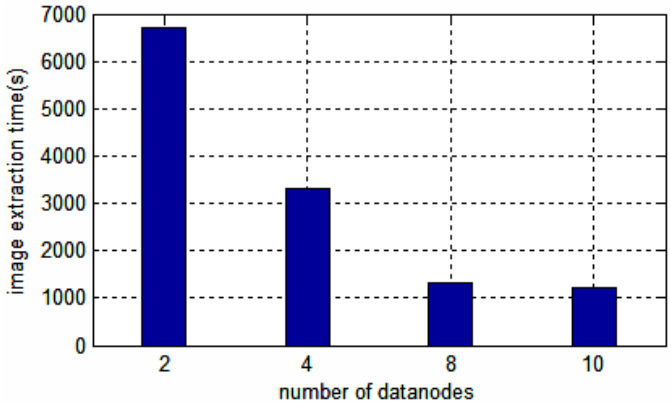

Figure 5. Comparision of the different number of datanodes used in Hadoop

C. *Comparision of the Image RetrievalTime*

We also verify the performance of the image retrieval time. We implemented the image retrieval algorithm on the

Hadoop environment with 4, 6, 8 slave nodes individually, and compared all the results with that of the stand-alone under different image library scales from 100,000 to 400,000 images. The comparison results are show in Fig. 6.

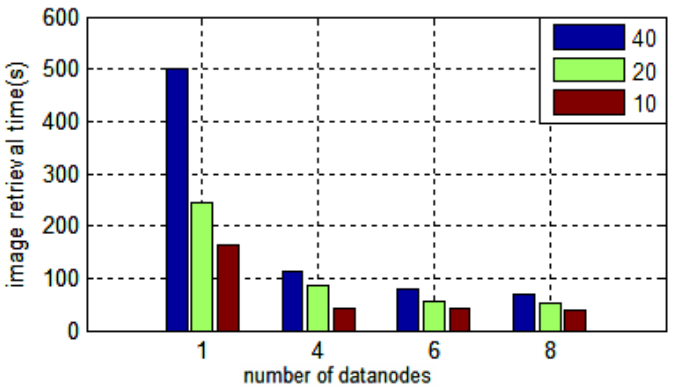

Figure 6. Comparison of the Image Retrieval Time

The size of data is from 100,000 to 400,000. It is obviously that the performance of any configuration of Hadoop platform exceeds that of the stand-alone. Furthermore, the more datanodes used in Hadoop, the more advantages would be obtained, especially in large scale image library size.

The time of retrieval doesn't decrease linearly with the number of datanodes or images increase. That is caused by the time cost on initializing the MapReduce module, scheduling the task and clearing the output. Meanwhile, intermediate results generated by Map task also need some time to transfer to the Result task. With the increase of image scale, time-consuming of the stand-alone mode has linear growth, while the time-consuming of the Hadoop model is slow growth.

D. Influence of the number of HBase Region blocks

Furthermore, to verify the influence of different number of HBase Region blocks, experiments on different number of Region blocks in Hadoop platform are conducted, as shown in Fig.7.

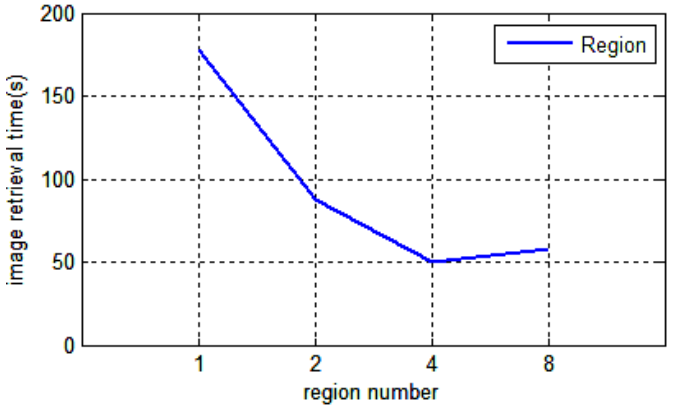

Figure 7. Comparisons on different number of Region blocks

In this experiment, the image library includes 200,000 images, and the datanodes are set to 6. From the experimental results, we can see that the number of region blocks affects the performance. The retrieval time decreases until the region number is 4, and then it takes a little more time in retrieval. That is because the data processed by Mapper is from one region. A Mapper task will process less data while the number of region is increased. And time cost in switching on the MapReduce program, scheduling of the task will increase. In the real application, it is better to add the region number selection process.

E. Comparison of accuracy

Though the Hadoop platform can speed up the time-consuming in feature extraction and retrieval, we also need to compare the accuracy between the Hadoop platform and stand-alone module. Table 2 gives the comparison results on different scenarios.

Precision (number of relevant images retrieved in stand-alone module/number of relevant images retrieved from Hadoop) is used as criteria for the accuracy assessment. We select Top-K as the result.

Table 2 Comparison of accuracy between Hadoop and stand-alone

Evaluation / Images	K	Precision
100,000	10	100%
200,000	50	100%
400,000	100	100%

From the experimental results, we can see that the proposed scheme keeps the same query result as that in stand-alone version.

V. CONCLUSION

This paper propose a scheme for Large scale cross-media data retrieval based on Hadoop to solve the problem introduced by the huge increase of the data size. Cross-media feature extraction and retrieval are divided into paralleled pipelines, and implemented with the combination of the HDFS, HBase and MapReduce framework. Experimental results demonstrate the good performances of proposed method, which sharply decreases time-consuming, and meanwhile keeps the same query result.

Our future work will focus on how to reduce the data of transmission under the MapReduce framework.

ACKNOWLEDGMENT

This work is supported by Chinese National Natural Science Foundation (61101212, 61372169), National High Technology R&D Program of China (863 Program) (No.2012AA012505).

REFERENCES

[1] L. Chen, D. Xu, I. W. Tsang, and J. Luo, "Tag-based web photo retrieval improved by batch mode re-tagging", in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recogn., 2010, pp. 3440–3446.

[2] Y. Liu, D. Xu, I. W. Tsang, and J. Luo, "Textual query of personal photos facilitated by large-scale web data", IEEE Trans. Pattern Anal.Mach. Intell vol. 33, no. 5, pp. 1022–1036, May 2011.

[3] Thuy Ho, Ngoc Ly, "A Scene Text-Based Image Retrieval System", Signal Processing and Information Technology (ISSPIT), 2012 IEEE International Symposium

[4] Pipit Dewi Arnesia, Sarifuddin Madenda, "Matching Images With Textual Document Using TFIDF Method", 2012 5th International Congress on Image and Signal Processing

[5] G. Schroth, S. Hilsenbeck, R. Huitl, F. Schweiger, E. Steinbach "Exploiting text-related features for content-based image retrieval", 2011 IEEE International Symposium on Multimedia

[6] Wen Li, Lixin Duan, Dong Xu, Tsang, "Text-Based Image Retrieval using Progressive Multi-Instance Learning", Computer Vision (ICCV), 2011 IEEE International Conference

[7] W. Niblack, R.Barber, W.Equitz, M.Flickner, E.Glasman, D.Petkovic, P.Yanker, C.Faloutsos, G.Taubin, "The QBIC Project: Querying Images By Content Using Color, Texture, and Shape", IBM Research Division, Almaden Research Center

[8] Wu Yi, Zhuang Yue-Ting, Pan Yun-He, "Image Retrieval System for Web: Webscope-CBIR", Database and Expert Systems Applications, 2000. Proceedings. 11th International Workshop on DOI: 10.1109/DEXA.2000.875089

[9] Hongmei Tang,Ming Yu,Zhitao Xiao,Yingchun Guo, "A Content-based Image Retrieval System on the Mode of Network", Circuits and Systems, 2000. IEEE APCCAS 2000. The 2000 IEEE Asia-Pacific Conference

[10] Z Alp Aslandogan, Clement Z Yu, "Automatic feedback for content based image retrieval on the web", Multimedia and Expo, 2002. ICME '02. Proceedings. 2002 IEEE International Conference

[11] Paolo Ciaccia, Marco Patella, Pavel Zezula, M-tree: An Efficient Access Method for Similarity Search in Metric Spaces, PROCEEDINGS OF THE 23RD VLDB

[12] Thuy Ho, Ngoc Ly, "A Scene Text-Based Image Retrieval System", Signal Processing and Information Technology (ISSPIT), 2012 IEEE International Symposium on Date of Conference:12-15 Dec. 2012, Page(s):000079 - 000084

[13] YuXin Chen, Luo Bo, "iLike: Bridging the Semantic Gap in Vertical Image Search by Integrating Text and Visual Features", IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 25, NO. 10, OCTOBER 2013

[14] Yu Suzuki, Masahiro Mitsukawa, Kyoji Kawagoe, "A IMAGE RETRIEVAL METHOD USING TFIDF BASED WEIGHTING SCHEME", ISBN: 978-0-7695-3299-8, Page(s):112-116

[15] Monir, S. M. Ghazanfar, Hasnain, S.K, "A Framework for Interactive Content-Based Image Retrieval[C]". 2005 Pakistan Section Multitopic Conference, INMIC, 2005.

[16] Rui Y, Huang T S, Mehrotra S, "Relevance feedback: a powerful tool in interactive content-based image retrieval [J]". IEEE Transactions on Circuits Systems for Video Technology, 1998,8(5):644 – 655

[17] Heng Chen, Zhicheng Zhao, Anni Cai,et al. "An Effective Relevance Feedback Algorithm for Image Retrieval". Proceedings - 2010 2nd IEEE International Conference on Network Infrastructure and Digital Content, IC-NIDC 2010, p251-255, 2010.

[18] Mohammad F.A. Fauzi,Paul H. Lewis, "Texture-based Image Retrieval Using Multi Scale Sub-image Matching", Image and Video Communications and Processing 2003

[19] Tom White, Hadoop: The Definitive.Guide.,3rd Edition, O'REILLY, 2012

[20] Sanjay Ghemawat,Howard Gobioff, and Shun-Tak Leung. The Google File Syste. SOSP'03, October, 2003

[21] Apache Hadoop, http://hadoop.apache.org/. 2013

[22] wiki.apache.org/hadoop/HDFS

[23] wiki.apache.org/hadoop/MapReduce

[24] wiki.apache.org/hadoop/Hbase