International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 4 Issue 5 ǁ May. 2016 ǁ PP. 23-28

Content Based Search Add-on API Implemented for Hadoop Ecosystem

Rahul Rao[1] Aditya Kamble[2] Siddhesh Palande[3] Kshama Jain[4] ShaileshHule[5]

Computer Department. Pimprichinchwad College Of Engineering Akurdi, Pune – India
Computer Department. Pimprichinchwad College Of Engineering Akurdi, Pune – India
Computer Department. Pimprichinchwad College Of Engineering Akurdi, Pune – India
Computer Department. Pimprichinchwad College Of Engineering Akurdi, Pune – India
Computer Department. Pimprichinchwad College Of Engineering Akurdi, Pune – India

Abstract: -Unstructured data like doc, pdf, epub is lengthy to search and filter for desired information. We need to go through every file manually for finding information. It is very time consuming and frustrating. It doesn't need to be done this way if we can use high computing power to achieve much faster content retrieval.
We can use features of big data management system like Hadoop to organize unstructured data dynamically and return desired information. Hadoop provides features like Map Reduce, HDFS, HBase to filter data as per user input. Finally we can develop Hadoop Add-on for content search and filtering on unstructured data. This add-on will be able to provide APIs for different search results and able to download full file, part of files which are actually related to that topic. It will also provide API for context aware search results like most visited documents and much more relevant documents placed first so user work get simplified.
This Add-on can be used by other industries and government authorities to use Hadoop for their data retrieval as per their requirement.
After this add-on, we are also planning to add more API features like content retrieval from scanned documents and image based documents.

Keywords: -Hadoop, Map Reduce, HBase, Content BasedRetrieval.

I. INTRODUCTION

Components In Hadoop Ecosystem:

1. **Map-Reduce**: Hadoop MapReduce (Hadoop Map/Reduce) is a software framework for distributed processing of large data sets on compute clusters of commodity hardware. It is a sub-project of the Apache Hadoop project. The framework takes care of scheduling tasks, monitoring them and re-executing any failed tasks.
2. **HDFS**: Hadoop File System was developed using distributed file system design. It is run on commodity hardware. Unlike other distributed systems, HDFS is highly fault tolerant and designed using low-cost hardware.
3. **HBase**: HBase is a data model that is similar to Google's big table designed to provide quick random access to huge amounts of structured data. This tutorial provides an introduction to HBase, the procedures to set up HBase on Hadoop File Systems, and ways to interact with HBase shell. It also describes how to connect to HBase using java, and how to perform basic operations on HBase using java.
4. **Hive**: Apache Hive is a data warehouse infrastructure built on top of Hadoop for providing data summarization, query, and analysis
5. **Sqoop**: Apache Sqoop(TM) is a tool designed for efficiently transferring bulk data between Apache Hadoop and structured datastores such as relational databases.
6. **Zookeeper**: ZooKeeper is a centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services. All of these kinds of services are used in some form or another by distributed applications.

Other components in hadoop ecosystem are Flume, Pig, Oozie, Mahout etc.

Missing Features:

Hadoop uses HDFS to store files and documents. HDFS is a hadoop distributed File system which provides features like location transparency, fault tolerance etc. But it does not provide content based search and retrieval on files present in the HDFS.

Tasks like assignments, taking notes from text books and reference books on particular topic, topics for presentation need deep reading and need to go through every document manually just to find relevant content on given topic. Currently present systems are only searching based on document title, author, size, and time but not on content. So to do content based search on big data documents and large text data Haddop framework can be used.

Manually filtering from any kind of unstructured data like PDF is tedious and time consuming, we are developing API for Hadoop finding relevant information from large sets and retrieving the same is main concern. So using Hadoop Big Data management framework consist of HDFS, MapReduce, and HBase, we are developing content based search on PDF documents to solve real life problem. So this is basic motivation for the project.

Our Solution to above problem:

As mentioned above, we are providing content based search and retrival on HDFS.
So using Hadoop Big Data management framework consist of HDFS, Map-Reduce, and HBase, we developed content based search on PDF documents to solve real life problem.

II. IMPLIMENTATION

Features provided:

- Metadata extraction of documents consist of file name, relevant keywords (Stopwords excluded).
- Filtering of files before searching for content.
- Page by page content search on filtered documents.

Prerequisite:

1. Hadoop
2. HDFS
3. Map-Reduce
4. HBase

Installation ofAPI:

1. Download ARKS.tar.gz
2. Extract ARKS.tar.gz to /usr/local
3. $sudo tar -xvfARKS.tar.gz -C /usr/local
4. Open /usr/local/arks/etc/conf.xml
5. Provide dataset_path, temp_path, wc_result_path, result_pdf_path, storage_mode, online_mode

System Architecture:

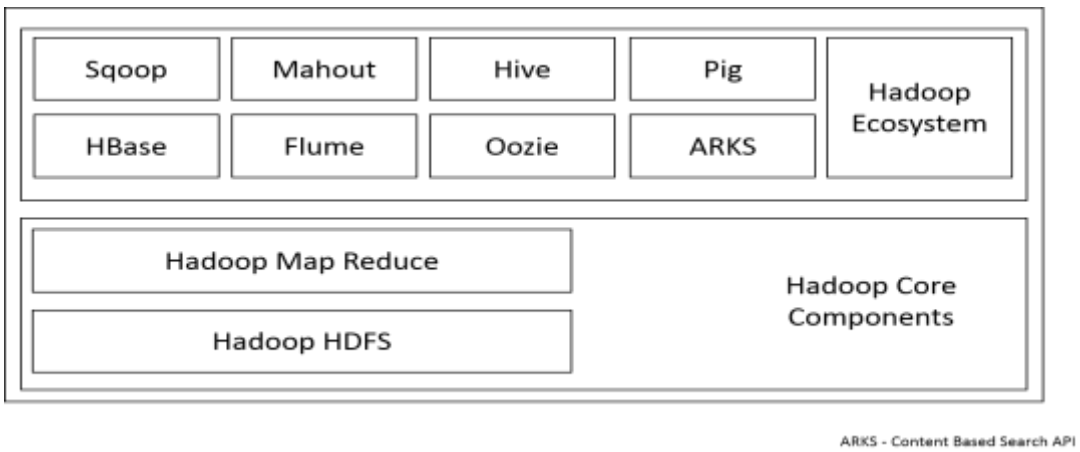

Figure 1 – System Architecture

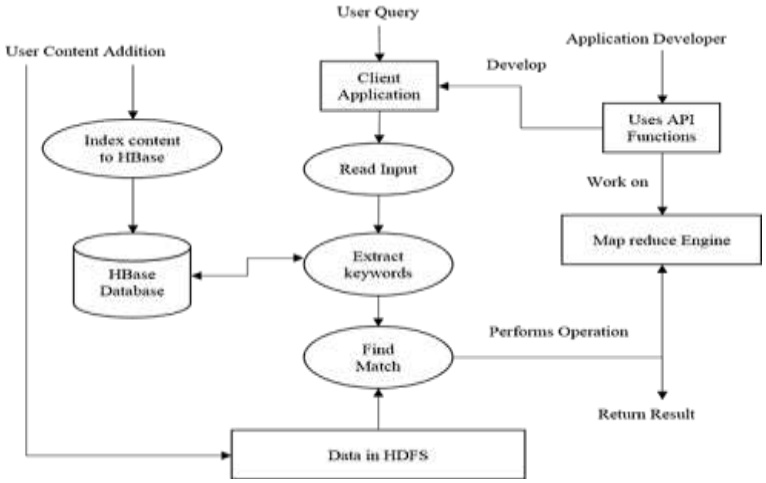

Figure 2 – Data Flow

Using API:

1. Add arks.jar dependency to any java project build path.
2. For maven projects add dependency to pom.xml.
3. Import ARKS packages and classes to java source code.

III. WORKING OF API

Overview:

The Hadoop Distributed Processing Framework and several components of its ecosystem generally does not provide a feature of extracting contents from a document using a distributed approach using MapReduce.This Framework fairly solves the problem of Content Based Search and Retrieval on any kind of documents but we are limiting our scope just for PDF files in this current release. The Content based search and retrieval here in this API has two very important components which are as follows:-

Metadata Extraction Job:

Extracts relevant keywords from each document parallely

Page By Page Search:

Finds the contents in the relevant pages and retrieves the output containing those relevant pages in the output File finally

Configuration:

Initially the ARKS before use must be configured for the various user defined parameters critical for the execution of API inside the conf.xml located in the etc directory in the root of the API.Foreg. The DATASET_PATH is the location used for storing all the documents in the file system.

Metadata Extraction:

Each and every document in the DATASET_PATH is traversed recursively for finding the list of PDF Documents. Also each and every document present in it provided to the Hadoop Mapreduce as a Job which distributively extracts the relevant keywords from the documents and stores them in a metadata storage database.This will be used for indexing each and every document in the HBase for quick retrieval of Metadata from the database.

Stopwords Removal:

StopWords are the words which are not so important redundant inside a document which are removed before the mapping phase so that they can be ignored during the mapping phase where the wordcount operation takes place.

Hadoop Mapreduce:

The Hadoop by-default does not provide any mechanism for creating key value pairs on the files other than .txt files. Therefore it is very much necessary to create a customized InputFormat which accepts PDF

documented files as input. The InputFormat describes the methods to read data from a file into the Mapper instances of Hadoop.

A split length is the size of the split data (in bytes), while locations is the list of node names where the data for the split would be local. Split locations are a way for a scheduler to decide on which particular machine to execute this split.

The Record Reader associated with PDFInputFormat is there to handle the splits that do not necessarily correspond neatly to line-ending boundaries. In fact, the RecordReader will read past the theoretical end of a split to the end of a line in one record. The reader associated with the next split in the file will scan for the first full line in the split to begin processing that fragment. All RecordReader implementations must use some similar logic to ensure that they do not miss records that span InputSplit boundaries.

HBase:

Here there are two column stores maintained

File_info:

The file_info stores the location of the file and other general metadata of the file like filename, filepath, totalpages, domain, subdomain etc about each and every document found in the DATASET_PATH directory.

File_keywords:

The file_keywords stores the relevant keywords after applying the StopWords filter to remove the stopwords during the mapping phase and its frequency count of it.It is stored in the fileindex_id format where the file index is which document the relevant keyword belongs to and id is the indexing of the relevant keyword.

Apache PDFBox:

Here the PdfBOX is the Java Api provided by the Apache Software Foundation used to parse and process the contents in a Pdf File.Here the PDFBox plays an important role not only in metadata extraction but also in generating the output for the retrieval of result after the content based search takes place.

In the metadata extraction each and every line from every document is read from the document and passed to the Mapper where the tokenization process takes place and the InputFileFormat which reads the Pdf calls the RecordReader which in turn generates the GenericSplits to instantiate the mapper to read line by line distributively to generate the relevant keywords and the frequency count as key-value pairs to be stored in Hbase.

In the Page By Page search each and every page is parsed and scanned by the Java Api provided by PdfBox to find the number of pages as well as the page number to check for the presence of the required content as passed in the query of searchForPages("user_query") method of ContentBasedSearch class.

The resultant of the page number list is split and combined to generate the output file containing the relevant content as the output required by the User.

IV. APPLICATIONS

Content based search and retrieval on

1. Digital library books
2. Conference Papers
3. Private Authorities
4. Government Authorities
5. Unstructured text files

This API can be used to develop above functionality on platforms like

1. Web Application
2. Android Application
3. Standalone Application
4. Command Line Interface Application

V. API TESTING

This API had been tested for Technical e-books using page by page search algorithm and obtained relevant pages in single pdf file.

In above test case there were 10 different technical e-books was taken as dataset. Initially the metadata database had been maintained inHBase consist of various relevant keywords in all the documents. These

keywords used to filter the documents present in HDFS. In this case, 10 files were filtered to 2 files which was actually related to the test input.

After that content based search and retrieval performed on filtered documents using page by page search algorithm. Results were fairly accurate.

VI. RESULTS

The dataset of pdf documents related to android, database, and design pattern technical books.

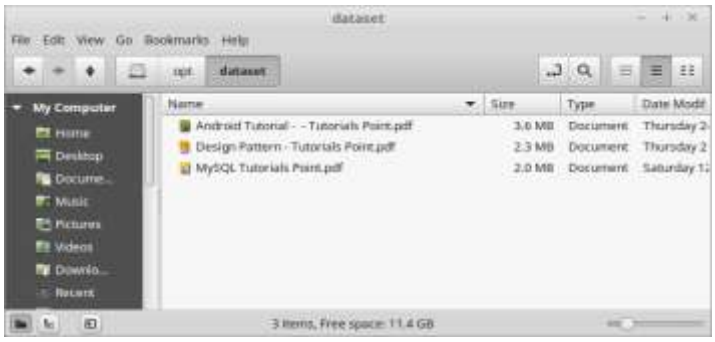

Figure 3 - Dataset

For Example: Input query is "Checkbox"

Then API will initially filter relevant documents from above dataset and then run content based search on those documents.

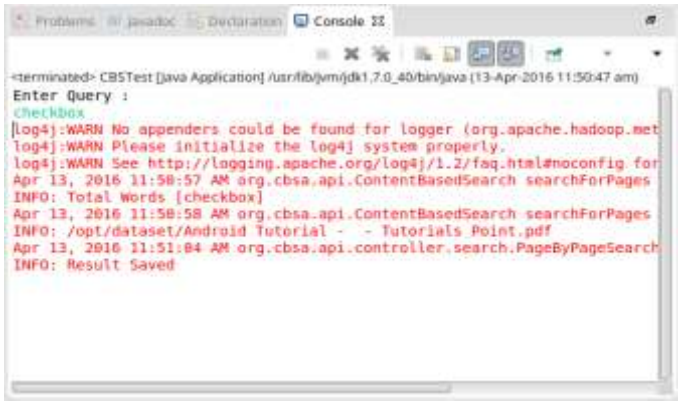

Figure 4 - Input Query

Here one document is filtered which is actually related to input query. The original size of document is large and has around 200 pages.

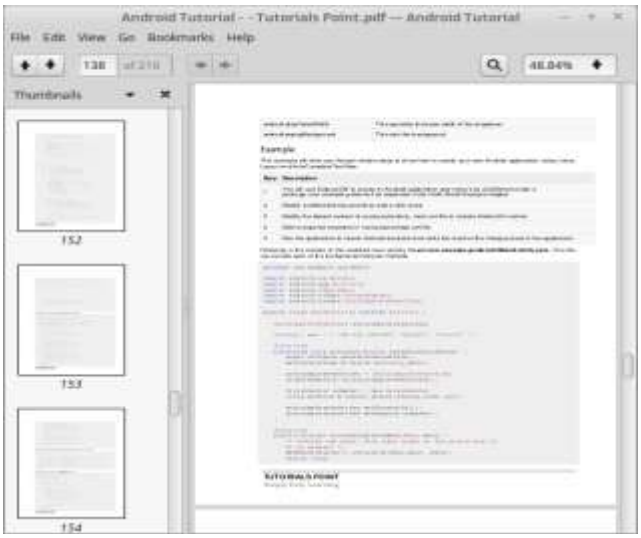

Figure 5 - Document before search

But after applying content based search on this filtered document output is generated in single pdf of relevant pages only. Result pdf is around 8 pages.

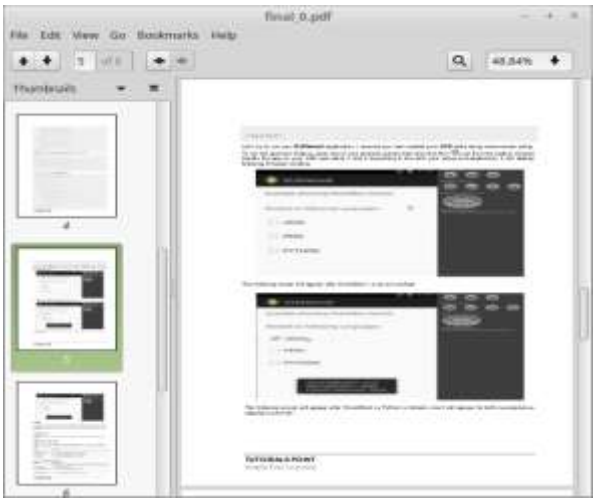

Figure 6 - Result PDF

VII. FUTURE SCOPE

Extending our API further to read scanned text copies and retrieve the data from them would solve further advanced issues. This API again can be extended further to work with Image processing so that it may find a relevant information from the digital images for the end- users.

We can also extend the API of ARKS to work on research paper by implementing a research paper parser using the Apache PDFBox so that the content can be extended for Research paper search and retrieval which typically has a two column format of PDF files.

We can also extend the API to work on multiple file formats like ePUB3,mobi,.odt,.docx by implementing the required RecordReader for the Hadoop Mappers to support such formats.

This API can be extended for content based searching and retrieving from paragraph.

VIII. CONCLUSION

This API is favorable for any system using Hadoop distributed environment to manage their documents and raw data. Thus we have contributed to solve the problem for content based search and retrieval on files in HDFS and Hadoop environment.

REFERENCES

[1]. Parul Gupta, Dr. A.K. Sharma, "Content based Indexing in Search Engines using Ontology ",2010
[2]. Lars George, "HBase: The Definitive Guide", 1st edition, O'Reilly Media, September 2011, ISBN 9781449396107
[3]. Tom White, "Hadoop: The Definitive Guide", 1st edition, O'Reilly Media, June 2009, ISBN 9780596521974
[4]. Apache Hadoop HDFS homepage http://hadoop.apache.org/hdfs/
[5]. MehulNalinVora ,"Hadoop-HBase for Large-Scale Data",InnovationLabs,PERC,ICCNT Conference
[6]. YijunBei,ZhenLin,ChenZhao,Xiaojun Zhu ,"HBase System-based Distributed Framework for Searching Large Graph Databases",ICCNT Conference
[7]. SeemaMaitrey,C.K."Handling Big Data Efficiently by using Map Reduce Technique",ICCICT
[8]. Maitrey S, Jha. An Integrated Approach for CURE Clustering using Map-Reduce Technique. In Proceedings of Elsevier,ISBN 978-81- 910691-6-3,2 nd August 2013