**UNIVERSITY OF LOUISIANA at LAFAYETTE**

Implementing Vector Space Model in Hadoop

By,

Sai Akash Kuthuru-sxk3190

INTRODUCTION

Implementing any retrieval model on larger datasets leads to inefficient way of information retrieval of desired data from larger datasets. In normal way of implementing retrieval model it takes lot of time while extracting raw data and indexing, which leads to slow flow of execution. Additionally in order to process the query it takes more time. Due to the need to handle large datasets, implementation of information retrieval system results in having to cope with the challenge of execution of times. In order to decrease the execution time, we adopt MapReduce concept of Hadoop.

Method

Vector Space Model:

Representation of a set of documents and given query as vectors in a common vector space is known as the *vector space model.*

Term Frequency

* Term Frequency for term in a document is defined as how often a term(t) has occurred in that particular document(d).
* It is given by the formula Tft,d= (1+log(ft,d)) for ft,d >0

Inverse Document frequency

* Inverse document frequency for the term t is defined as term’s scarcity across all the documents.
* It is an inverse measure of the informativeness of t
* It is given by the formula Idft,d=(log10(N/nt)).

N:-Total collection of documents.

nt:-Documents containing term t.

Tf-Idf Weighting

* Term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.
* The tf-idf value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general.

Hadoop basically uses MapReduce concept which considers two significant methods namely Mapper and Reducers. MapReduce basically depends on the configurations of clusters which is a fully configured cluster, “running Hadoop” means running a set of daemons, or resident programs, on the different servers in your network.

Hadoop daemons are

NameNode

The NameNode is the master of HDFS that directs the slave DataNode daemons to perform the low-level I/O tasks. The NameNode is the bookkeeper of HDFS; it keeps track of how your files are broken down into file blocks, which nodes store those blocks, and the overall health of the distributed filesystem

DataNode

DataNode daemon to perform the grunt work of the distributed filesystem—reading and writing HDFS blocks to actual files on the local filesystem.

Secondary NameNode

The Secondary NameNode (SNN) is an assistant daemon for monitoring the state of the cluster HDFS. Like the NameNode, each cluster has one SNN, and it typically resides on its own machine as well.

Job Tracker

The JobTracker daemon is the liaison between your application and Hadoop. Once you submit your code to your cluster, the JobTracker determines the execution plan by determining which files to process, assigns nodes to different tasks, and monitors all tasks as they’re running.

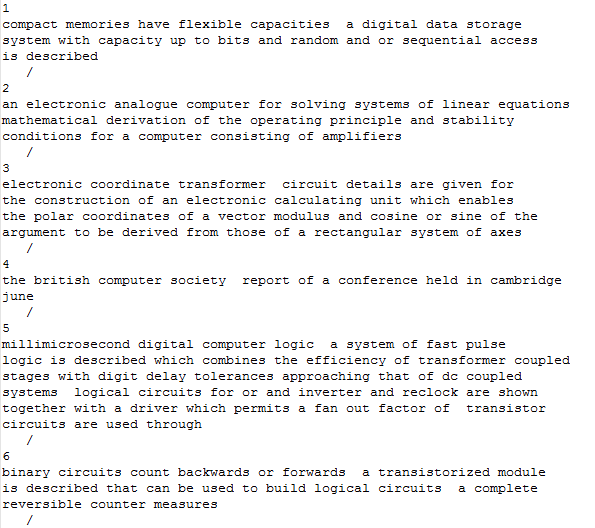
TaskTracker

TaskTracker is responsible for executing the individual tasks that the JobTracker assigns. Although there is a single TaskTracker per slave node, each TaskTracker can spawn multiple JVMs to handle many map or reduce tasks in parallel resulting in parallel programming.

Frame work: Hadoop using HDFS and MapReduce

Hadoop implementation: Single node Hadoop cluster(Hadoop 2.4.1)

Language: Map-Reducing programming, java SDK

Data Source: Corpus collection of 6000 documents is considered and is of type as below

Flow of execution

Step1: Set mapper and reducer classes

Step2: Initialize mapper where each file from the dataset is considered as a single map. In mapper class we generate key value pairs where key is the word and values if the form “1@filename” which is basically a count of the word which is appended with its filename in order to manage postings.

Step3: For each input mapper produces iterable datatype which consists of key value pairs which are sorted alphabetically with respect to keys.

Step4: Reducer class takes input from mapper class, for every key value pairs reducer method Is called. In reducer method.

Step5: HashTable is used in order to store term frequency of word and its posting list.

Step6: Calculating the idf values of the terms present in the given query.

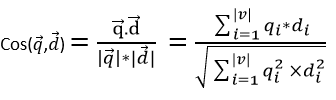
Step6: Calculating tf-idf weighting of the terms present in the query.

Step8: Converting tf-idf weights and documents as vectors.

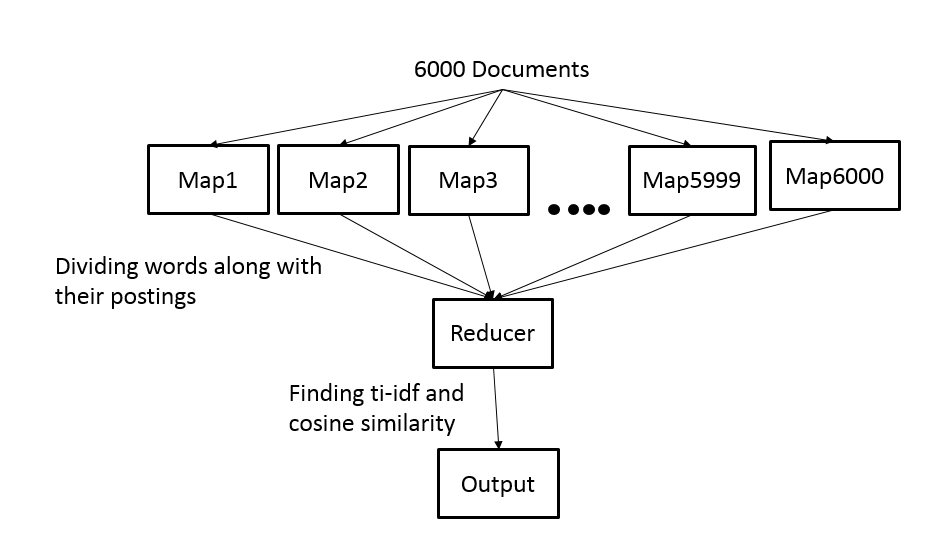
Step9:Calculating similarity between query and documents using cosine similarity.

Similarity: It is defined as the cosine angle between query vector and document vectors in which the terms in the query are present.

Formula for cosine similarity is given as

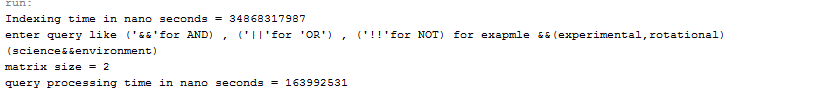


Step10: Calculating time taken to build Inverted Index and query processing time while using hadoop and comparing it with time consumed in assignment 1(Fuzzy Model).



Output for the project:

Time taken to construct inverted index and query processing time in assignment for corpus of 12000



Therefore total time taken for in seconds 35.0323104

Time consumed while implementing vector space model in Hadoop which is buiding inverted index and query processing time.

