Optimised Image Storage and Retrieval on Hadoop

Abstract. With the exponential growth of data, it is difficult to efficiently store and retrieve data using traditional methods. There is a need to optimize the storage and to efficiently retrieve relevant data matching the user query. Traditional methods lack optimized storage and to effectively retrieve data. To overcome these limitations, in this project, we propose a distributed architecture framework to optimize memory usage and to effectively retrieve relevant data using Content-Based Image Retrieval (CBIR). The experimental results show that the proposed model enhances storage performance and retrieval time by 20%.

1 Introduction

Nearly all the software we use today is an extent of distributed system or involves Distributed Architecture (DA). A distributed system is a collection of independent and separate components called nodes that are linked together (Components could be both software and hardware components) by means of a network and work together in meaningful order by communicating and coordinating with each other. Google search engine, Amazon platforms, Netflix, Blockchain systems, and online transactions all use distributed systems for their functioning [9]. The nodes of a system could be unstructured or highly structured based on the system requirements. A bunch of independent nodes or computers cooperate and coordinate with each other to solve a common problem. A whole system consisting of all the nodes acts as a single computer to its users. Every node or computer in the distributed system has its own processor and harmony is achieved through synchronization and coordination, also each system has its own memory [8]. Processes are autonomous and execute tasks concurrently. The distributed file system is also dynamic in nature i.e it allows computers and nodes to join or leave the system at their will, this property has many advantages. It provides reliable interconnection between the nodes which helps in the easy sharing of the data. Scalability is also one of the main reasons why we use distributed file systems and also it has fault tolerance, which means the system and its services will still be operational and reliable even when part of the system 2 goes down [24].

***Corresponding Author: akhileshgadagkar@gmail.com**

It increases the overall performance of the system. Distributed systems help to achieve parallel processing and distributed data processing. Parallel processing involves the simultaneous execution of multiple sub-tasks that collectively comprise a larger task. The main aim is to reduce the execution time by dividing a single task into multiple smaller tasks. Similarly distributed data processing uses the same principle of divide and conquer and it is always achieved through separate machines networked together as a cluster [11]. Extra work of managing the organization, inefficient storage of data, Failure detection, and performance bottlenecks are some of the challenges in distributed file systems. There are a lot of modern data analytical tools which help the medical field in various ways like preventing epidemics, curing elusive diseases reducing costs, and improving quality of life [12]. The amount of medical data being collected on a daily basis is growing exponentially with the development of technology. There is a need for efficient high-performing techniques and algorithms for the processing of this kind of data this is where Big data analytics comes in, It covers the integration of heterogeneous data, analysis, modeling, data quality control, and validation [14]. In medicine and healthcare, Applications of big data such as Distributed file systems help in the analysis of large datasets from thousands of patients, identifying clusters as well as developing predictive models using data mining techniques.

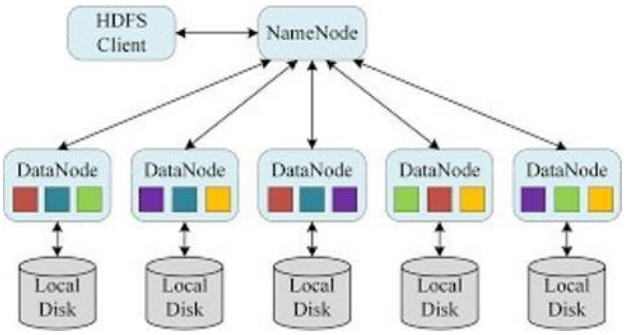

Fig. 1. Hadoop Architecture

Figure 1 describes Hadoop distributed file architecture. Hadoop is a popular open-source framework for large-scale data storage and data processing that is compatible with commodity hardware [13]. There is a need to improve its storage and retrieval system for better accuracy and more efficient use of the system. With respect to Medical image processing through Big data tools such as Hadoop. Existing studies have the following problems i) Focused on either optimized storing or retrieval, not on both ii) There is a need for a proper system for storage, indexing, and retrieval of the medical images also consisting of patient data iii) No proper utilization of the benefits of distributed file systems in the medical field[25]. So we propose – To optimize the storage of medical data in the Hadoop single-node cluster. Proper indexing of data along with the feature vectors of an image dataset. – CBIR for the query image given by the user. Section 2 is about the description of related works done in this field, section 3 describes our proposed methodology. Section 4 is about the implementation of our proposed system. Section 5 is the results and discussion and Section 6 is the conclusion.

2 Related Work

In this research work[1], Xie Yaya and the team have discussed the advantages, feasibility, and problems that need to be addressed in the medical field for a large amount of medical image storage and retrieval. They have used HDFS clusters for large-scale image storage and retrieval to overcome the limitations of the data structure model. They have worked with the DICOM format of images in

their work. A comparative study of the HDFS system and existing system (PACS) is made in the work and got several interesting results[16]. HDFS storage and retrieval were 5% and 7% more efficient when compared to existing systems respectively. Optimization was achieved using algorithms such as map- reduce in HDFS.

Shunxing Bao and Bennet Landman in their research paper[23] have proposed an arithmetic enhancement to medical image processing using big data computing frameworks. They have developed a range of performance-enhancement algorithms. They have given some useful insights on the blending of the Apache Hadoop ecosystem with distributed machine learning algorithms to redesign and optimize the existing frameworks for medical images.

Kingsy Grace and Dr. R Manimegalai[3], in their research work, have tested the performance of Hadoop for content-based image retrieval (CBIR) with three current operative nodes. They proposed a grid architecture that makes use of Apache Hadoop to set up a grid. Images are retrieved based on the query image. The image which is very close to the given image is retrieved from the cluster. They have used the feature vector of the image which is generated on the visual patterns of the image. An algorithm that performs the similarity search is used for the comparison of the feature vector of two images. With the implementation using this technique, the accuracy of the retrieval increased drastically. The system implemented In this paper is also very easy to adapt in the cloud environment with, minimal cost and space.

In another study[4], authors have proposed a Hadoop-based medical image retrieval system for better efficiency in retrieval so that a large number of images can be processed at a time. HDFS and MapReduce were the key concepts used in their proposed methodology. They used distributed file systems to store the data and for parallel processing of the data, the MapReduce framework was used. They observed these frameworks helped to reduce retrieval time. They have discussed different image analysis methods namely Bag of visual words, Riesz Miniature, and Gabor Filter. They dedicated the master node of the Hadoop to take query images as input from the user and slave nodes to do image processing parallelly[6]. The main aim of the master node is to take input query images from the user and to create a job that will process the image and related images will be returned.

In this paper[5], they talked about the necessity of methods like content-based image retrieval to handle huge amounts of data using parallel processing techniques. They also described the disadvantages of older methods used in this like text-based image retrieval where images were retrieved on the basis of tags or keywords or the index of the image. The issue with this method is that we have to manually assign the tags or the indexing to the image which is difficult if the data is huge. They also discussed different properties of the images on the basis of which features of the images were extracted[7]. The most important feature is the color, they are usually defined in three-dimensional spaces. The next property considered is the texture and visual patterns associated with the image. Regularity, directionality, randomness, and degree of contrast are some of the texture features. Another property is the shape. The shape helps to determine the nature of the image. Direction, relative size, convexity, and circularity are some of the shape features.

The paper[6] discusses the use of the Hadoop framework for implementing a content-based image retrieval (CBIR) system for medical images. The CBIR system allows users to search for images based on their visual content, rather than just their metadata or file names. The authors of the paper propose a CBIR system that uses Hadoop to process and analyze the visual content of medical images, including features such as color, texture, and shape. The system includes a preprocessing stage to extract these features from the images, and a search stage that uses these features to retrieve relevant images. The authors also describe how they tested the system using a dataset of medical images and evaluated its performance using various measures, including precision and recall. They found that the system was able to effectively retrieve relevant images based on their visual content, with a high degree of accuracy. Overall, the paper suggests that Hadoop can be a useful tool for implementing a CBIR system for medical images, providing a scalable and cost-effective solution for searching and retrieving images.

3 Methodology

The methodology proposed for this paper is as per the flow chart in Fig [2]. It consists of Feature Extraction, storing the feature vectors in Hadoop Distributed File System (HDFS), Similarity Score Computation, and retrieving the most similar images on the basis of the similarity score from the database.

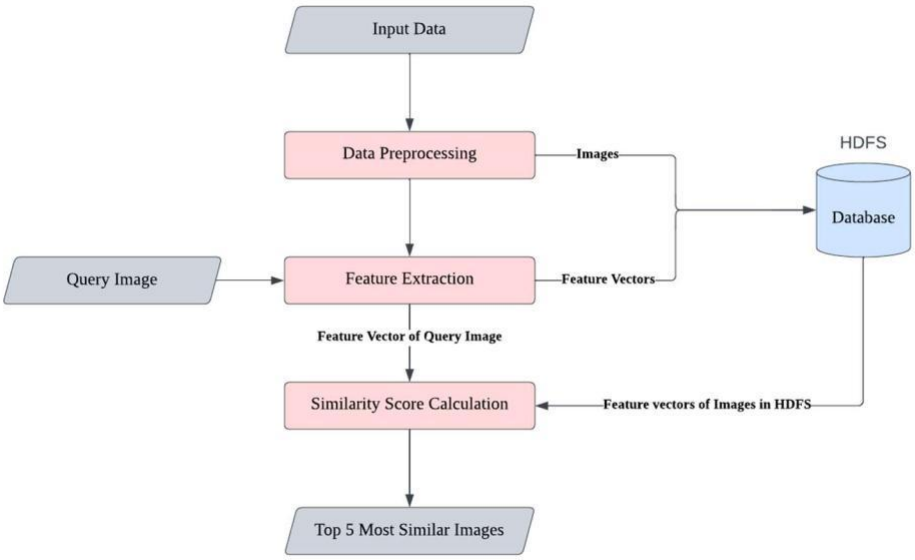

Fig. 2. Flow Chart

3.1 Data Preprocessing

Data Pre-processing is a data mining technique which is used to transform the raw data in a useful and efficient format. Data pre-processing includes various steps such as Data Cleaning, Handling Noisy Data, and Data Transformation. In this study as a part of data pre-processing first, we performed a detailed study on the dataset and analyzed it. There were in total 1, 12,120 patients registered in the dataset out of which 30,805 were unique having 14 different types of the diseases. For the data cleaning part we have dropped the NULL values as they could have created problem while training the model. The dataset has X-ray Images and a CSV file which contains the details of the disease diagnosed in X-ray. The X-ray is taken in two positions named Posterior Anterior position and Anterior Posterior Position. After analysis, we can say that most of the X-rays are in Anterior-Posterior Position. 60% of the X-rays are Anterior-Posterior type whereas 40% of the X-rays are Posterior-Anterior type. When the patient is in very bad condition and he/she is unable to stand then the X-ray is taken in the Posterior-Anterior position.

3.2 Feature Vector Extraction

Feature Extraction is the technique of turning raw data into numerical features that can be handled while keeping the information in the original dataset. The outcome of feature extraction is called feature vector. The CNN model that we used for feature extraction is VGG16. It is a 16 layers deep convolution neural network. It is a pre trained model trained on Image Net dataset. It converts the features of the

image into numerical vectors[17]. The features considered are decided on the basis of type of the image. If it is a coloured image then features. will include colours, depth of colours, and if the image is a black and white image then the features that are considered are depth of pixels, and depth of white an black colour in the image[19].

3.3 Comparison of similarity scores of feature vectors

Content-Based Image Retrieval (CBIR) is a method of retrieving images from the database. In content-Based image retrieval system a user specifies a query image and then the images which are similar to it are retrieved from the database[20]. Content-Based image retrieval system works in two parts: i. Feature Matching ii. Calculating similarity index on the bases of features matched When the user specifies a query image it passes through the phase of feature extraction in which the features of that image are stored and then they are compared with the feature vectors stored in the database using the concept of Cosine Similarity. Cosine Similarity is the technique that quantifies the similarity between two or more vectors. The cosine similarity is the cosine of the angle between vectors. The formula to calculate the cosine similarity index is: Cos (x, y) = x.y/x + y = 1 In order to calculate the distance rather than the cosine similarity, we can use the spatial.cosine.distance() function from the SciPy package. After calculating the similarity index we retrieve the top 5 images that are most similar to the query image from the database as shown in fig [3].

```
Top5
[0.9278912544250488,
 0.9194123148918152,
 0.9108192920684814,
 0.8882317543029785,
 0.8839909434318542]
```

Fig. 3. Similarity Score of top5 most similar images

4 Implementation

4.1 Experimental Setup

Setting up the Hadoop multi-node is the first step of our proposed methodology. Java is the main prerequisite as Hadoop runs on java so we have to install the latest JDK version. The next step is to create a USER account on both the master and slave systems. We have to setup SSH keys in every node so that they can communicate with one another. The operating system that we used for setting up the Hadoop system is Linux with a memory space of 50-60 GB. The system on which the Hadoop Multi-node was set up had a RAM of 8Gb and a 1000 Gb Hard Disk. The system was working on a core i5 processor. We have used the latest version of Hadoop which is 3.2.2. We also required software to execute the queries in Hadoop for which we installed Java JDK. To differentiate the root from the Hadoop user we have created a new user for the Hadoop setup named HD user in the Hadoop user configuration. To access setup from other devices we have generated a Secure Shell (SSH) key. In Namenode and Data node configuration part, all the Hadoop configurations on files are updated. All the XML files are also updated. For the formatting part of the Namenode and Datanode, all the different nodes of the Hadoop like Namenode, Datanode, Resource Manager, and Jps are started for query

processing. After setting up everything, Namenode is started on Localhost:9870, and Resource Manager is started on Localhost:8088.

Setting the core-site.xml file in the following manner will provide us an user interface at localhost:8970.

```
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://localhost:9000</value>
</property>
</configuration>
```

Fig. 4. Configuring core-site.xml file

```
hduser@akhi-VirtualBox:/usr/local/hadoop/sbin$ jps
5008 Jps
3333 DataNode
4278 NodeManager
3117 NameNode
3647 SecondaryNameNode
4095 ResourceManager
hduser@akhi-VirtualBox:/usr/local/hadoop/sbin$
```

Fig. 5. Hadoop set up

4.2 Feature Vector Generation

Feature extraction leads to the generation of feature vectors which is done in two parts (as shown in Fig[5]): 1. Define the function to extract features: The model used to extract the feature vectors is VGG16. We load only 15 layers of feature vectors because 16th layers is for object detection which is not required in feature extraction. Store the feature vectors in a dictionary and map them according to the index of the images in the dictionary. 2. Extract the feature vectors for the dataset.

```
1. Define Function extract_features
        Load model = VGG16
        Pop last layer
        Declare a dictionary(features) to store the features
        for each image in dataset
            filename = directory + name
            image = load_image (filename, resize(244x244)
            image = image_to_array(image)
            features = model.predict(image)
            map the feature according to image id
        return features

2. Load the dataset
   features = extract_features(dataset)

   dump features into features.txt
```

Fig.6. PseudoCode to Generate Feature Vectors

4.3 Comparing Feature Vectors

Comparison of the feature vectors is done using the concept Cosine Distance. The features of the images stores in the database are stored in list1, whereas the feature vectors of the query image are stores in list2. Similarity scores are sorted in descending order and on the basis of that top5 similarity scores are stored in function top5(as shown in Fig[6]).

```
declare list1[], list2[], list3
list1.append(dataset_features)
list2.appned(query_features)

declare result[]

for i to length(list1)
        result.append(1-cosine_distance(list1[i], list2)

sort the result in ascending order
declare top5[]
for i =0 to i = 5
        top5[i] = result[i]
```

Fig. 7. Pseudo Code to Compare Feature Vectors

4.4 Dataset Description

It consists of 1,12,120 images with labelled diseases of 30,805 unique patients. Diseases are labelled with the help of Natural Language Processing(NLP) and have an accuracy of 90%. All the images are

of size 1024*1024. It also consists of 1 CSV file which contains an image and patient details, attributes of the CSV file are image index, disease type follow-up, patient age, patient gender, X-ray orientation, Original Image Width, Original Image Height Original Image Pixel Spacing x Original Image Pixel Spacing y and also in the disease class there are 15 predefined diseases which are already classified with the help of natural language processing they are Atelectasis, Consolidation, Infiltration, Pneumothorax, Edema, Emphysema, Fibrosis, Effusion, Pneumonia, Pleural thickening, Cardiomegaly, Nodule Mass, Hernia and images which does not belong to any of the above-mentioned categories are stored under no findings category. The chosen dataset was suitable for the proposed system and had the requirements of efficient storage and retrieval as the size of the data is huge and difficult to process.

5 Results

After the setup of Hadoop nodes and feature vector extraction of images, they are stored in the Hadoop nodes. The query image is taken from the user and the feature vector of the query image is extracted and it is compared with the feature vector of all the existing images. Top-5 matching images are extracted with respect to the query image.

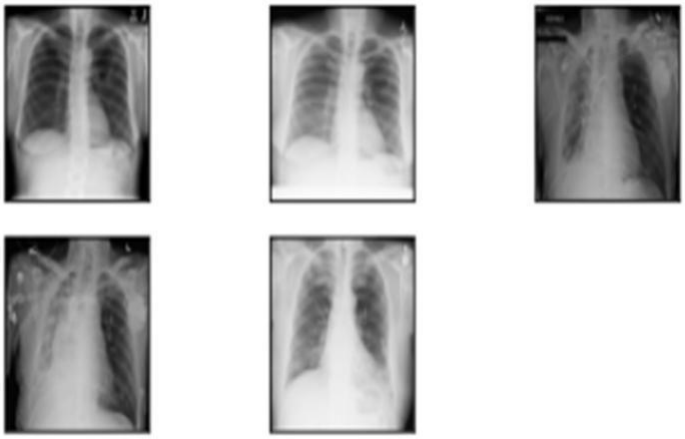

Fig. 8. Images most similar to the given query image

The top 5 images matching the given query image from one of the test cases are shown in Figure 8. Table 1 describes storage time in milliseconds based on number of images stored from the dataset in the HDFS. Table 2 depicts the comparison of image Retrieval Time (RT) on traditional HDFS methods and the proposed HDFS storage method. Figure 9 describes the comparison of the retrieval time of traditional HDFS and the proposed system. The result shows that for the small size of images, the retrieval time is almost same. As the size increases, the proposed model performs very well.

Table 1.Storage time for uploading images in HDFS

Sl No	Size of images(mb)	Storage time(ms)
1	10	10.76
2	50	47.81
3	100	75.35
4	150	100.30
5	200	121.19
6	250	132.18
7	300	147.23
8	350	151.92

Table 2. Comparison of image retrieval between HDFS and local system

Sl No	Images per folder	RT traditional HDFS	RT proposed HDFS
1	10	29	27
2	50	47	42
3	100	82	61
4	500	356	271

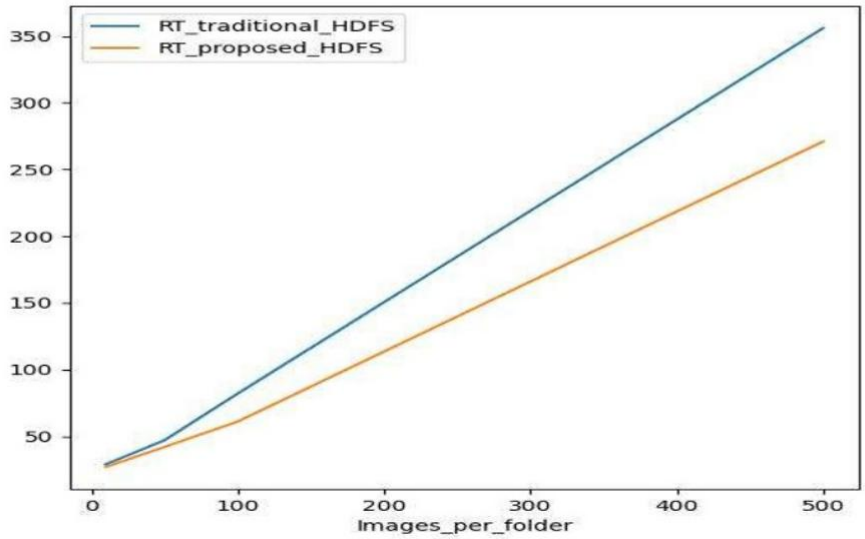

Figure 9. Graph showing a comparison of the proposed method and state-of-the-art methods in terms of retrieval time.

6 Conclusion

In this project, we have used the Hadoop framework for storing and CBIR systems. The established Hadoop-CBIR in this work has enormous potential to be applied in numerous other fields. The use of CBIR of medical images on Hadoop has been shown to be a powerful tool for medical image analysis. It can provide a cost-effective and efficient way to process large amounts of data, while providing accurate results. The use of Hadoop also allows for scalability and flexibility, allowing for the integration of new technologies and algorithms as they become available. With the increasing

availability of medical imaging data, CBIR on Hadoop is becoming an increasingly important tool in the field of medical image analysis. The application created using the suggested methodology is quick and effective in retrieving medical photos. It also makes it easier to accurately retrieve photographs that match the image you've asked for. The created application uses a content-based image retrieval technique that reliably and quickly yields accurate results. In the future it is very easy to adapt to a cloud-based environment with minimal overhead as we have already implemented this in Hadoop.