Content

List of Figures

1. INTRODUCTION

Big data analysis is one of the highly researched areas today, with an objective of taking advantage of various computational resources. In this project we have used Hadoop environment with the aim to speed up the matching process of biometric traits. We have used iris recognition, a biometric technique, as it is one of the strongest methods of authentication. Also Iris recognition is stable over time. We have used Hadoop, an open source cloud computing environment, to develop this model. Hadoop implements Map/Reduce framework in Java. Map/Reduce make easy to process large amount of data on.

There are various techniques to uniquely identify an individual based upon physical and behavioural traits in biometrics field. Biometric technologies differ but all works in a similar manner, gather unique physiological and behavioral characteristics of a person, and store it into a database or comparing to found already stored templates in database. Iris recognition is one of the strongest methods of biometric authentication. Iris recognition systems are gain-ing interest because it is stable over time. Iris recognition technology provides positive identification of an individual, at extremely high confidence levels. Iris scan has been developing an identification/verification system capable of positively identifying and verifying the identity of individuals. It uses the unique patterns of the human iris, shows promise of overcoming previous shortcomings. With the spread of cloud computing and network techniques and equipment in recent years, a large amount of data collected in a variety of social situations have been accumulated, and so the need for analysis techniques to take advantage of useful information that can be extracted from such data sets is increasing. This is also true for video data, in which image share sequential and the data includes the associated time and frame information of each frame.

Today, because video cameras are set up to perform surveillance of moving objects such as pedestrians and vehicles, a large amount of video data is generated, and stored in database form. To improve these video database systems, which hold a large amount of video and related data, parallel processing using CPUs and disks at multiple sites is an important area of research.

Also, when considering operations such as search and other types of analysis of video images recorded by video camera and stored in a database, there are limits on what can be done to improve the performance of single computers to make them able to process large-scale information. Therefore, the advantages distributed processing of a video database by using the computational resources of a cloud computing environment should be considered.

In addition, if computational resources can be secured easily and relatively inexpensively, then cloud computing is suitable for handling large video databases at low cost. Hadoop, as a mechanism for processing large numbers of databases by parallel and distributed computing has been recognized as promising. Nowadays, for reasons such as ease of programming, by using the function MapReduce on the Hadoop system, open-source cloud-based systems that can process data across multiple machines in a distributed environment have been studied for their application to various database operations. In fact, Hadoop is in use all over the world. Studies using Hadoop have been performed to treat one as a text data file or multiple files as a single file unit, such as for the analysis of large volumes of DNA sequence data, converting the data of a large number of still images to PDF format, and carrying out feature selection/extraction in astronomy. These examples demonstrate the usefulness of this system, which is due to its having the ability to run multiple processes in parallel for load balancing and task management.

1.1 Purpose of the System

Iris Recognition System must Collect the primary data (images or samples) Analyze the data (extract features) Use the data (searching/matching) Store the data (images and indexes) Exchange the data (sharing) each of these steps requires varying amounts of computation and storage re-sources. Building and testing large scale biometric algorithms involved a large group of computers, ad-hoc testing scripts, and a group of system administrators trying to maintain the whole assembly. The Basic Problem: too much data, too much time. Training biometric algorithms requires large training sets. Score results are quadratic in growth rate: N cases gives N*N results. Multiple matching algorithms can be used as input, so again, more data. Multiple feature extraction algorithms each produce their own feature data from input images. The more permutations are involved, the more CPU time and storage is needed.

In Iris Recognition, there is a lot of data to manage and handle reliably. Tests need to be fully check pointed, reliable, and redundant. A single hard disk or computer failure could doom an entire test, so the framework that runs the tests needs to be robust and automatically handle the problems With the advent in on-demand Hadoop based computing services and frameworks; we can now scale resources as needed, in a reliable, redundant and robust fashion. Thousands of CPU's can be allocated as needed for the test in order to meet throughput or latency requirements. Multi-Terabyte databases are no problem .Processing the statistics for trillions of match scores is easily done. An advanced scheduling system based on Apache Hadoop breaks up the calculations into small jobs that are scheduled throughout a loosely coupled network. A highly reliable file system replicates data in at least three places in the network, providing reliability in the face of a disk or network failure. Data replication not only helps with reliability, but increases data read throughput; important for feature matching.

1.2 Objectives

1. Implement Iris Recognition System for large number of users.
2. Improve Security by using probability based calculation of multiple images of single user.
3. Handling large number of Images using Hadoop.
4. Improving performance of image recognition using Hadoop Image Processing System.

1.3 Literature Survey

a. Brief review of Literature

There are various techniques to uniquely identify an individual based upon physical (finger print, face recognition, palm print, iris recognition) and behavioral (typing rhythm, gait, voice) traits in biometrics field. Biometric technologies differ but all works in a similar manner, gather unique physiological and behavioral characteristics of a person, and store it into a database or comparing to found already stored templates in database. Iris recognition is one of the strongest method of biometric authentication. Iris recognition systems are gaining interest because it is stable over time. Iris recognition technology provides positive identification of an individual, at extremely high confidence levels. Iris scan has been developing an identification/verification system capable of positively identifying and verifying the identity of individuals. It uses the unique patterns of the human iris, shows promise of overcoming previous shortcomings.

a. International status

Xuhui Liu et al. was reviewed, where Hadoop distributed file system (HDFS) is designed to manage large files and suffers performance penalty while managing a large amount of small files. As a consequence, many web applications, like WebGIS, may not take benefits from Hadoop. In this paper, we propose an approach to optimize I/O performance of small files on HDFS.

Milind Bhandarkar was designed will discuss design philosophy of Hadoop, describe how to design and develop Hadoop applications and higher-level application frameworks to crunch several terabytes of data, using anywhere from four to 4,000 computers.

Jiong Xie, Shu Yin, Xiaojun Ruan, was presented MapReduce has become an important distributed processing model for large-scale data-intensive applications like data mining and web indexing. Hadoop–an open-source implementation of MapReduce is widely used for short jobs requiring low response time. The current Hadoop implementation assumes that computing nodes in a cluster are homogeneous in nature. Data locality has not been taken into account for launching speculative map tasks, because it is assumed that most maps are data-local. Unfortunately, both the homogeneity and data locality assumptions are not satisfied in virtualized data centers.

Y onggang Wang Sheng Wang et al. was presented, where As the deep development of spatial information Sharing service, it brings forward high request to usability and expansibility of supporting system. Based on large-scale scalable server cluster, cloud computing brings hopes to resolves the existing difficult problems in the domain of geospatial information service. In this paper, we imported cloud computing technology Including MapReduce model and Hadoop platform into the domain of geographic information system (GIS). Those key technology problems in the application of GIS such as spatial data storage, spatial index and spatial operation were described and studied in detail.

1.4 Requirements

Software Requirements

1) Operating System : Ubuntu 12.04
2) Java 6
3) Hadoop 0.20.1
4) HIPI 1.0.1

Hardware Requirements

1)	Ram :	512 MB
2)	Hard Disk :	20 GB

2. WHAT IS HADOOP

2.1 Parallel and Distributed Processing on Hadoop

As the structure of the system, Hadoop consists of two components, the Hadoop Distributed File System (HDFS) and MapReduce, performing distributed pro- cessing by single-master and multiple-slave servers. There are two elements of MapReduce , namely JobTracker and TaskTracker, and two elements of HDFS, namely DataNode and NameNode. In Figure 1, the configuration of these elements of MapReduce and HDFS on Hadoop are indicated. There is also a mechanism that checks the metadata for NameNode.

2.1.1 Job Tracker

Job Tracker manages cluster resources and job scheduling to and monitoring on separate components.

2.1.2 Task Tracker

Task Tracker is a slave node daemon in the cluster that accepts tasks and returns the results after executing tasks received by Job Tracker.

2.1.3 Name Node

An HDFS cluster consists of a single Name Node, a master server that manages the file system namespace and regulates access to files by clients. Name Node executes file system name space operations, such as opening, closing, and renaming files and directories. It also determines the mapping of blocks to Data Nodes.

2.1.4 Data Node

The cluster also has a number of Data Nodes, usually one per node in the cluster .Data Nodes manage the storage that is attached to the nodes on which they run .Data Nodes also perform block creation, deletion, and replication in response to direction from Name Node.

2.1.5 Secondary Name Node

Secondary Name Node is a helper to the primary Name Node. Secondary is responsible for supporting periodic checkpoints of the HDFS metadata.

2.2 Hadoop Distributed File System (HDFS)

HDFS is designed to reliably store very large _les across machines in a large cluster. It is inspired by the Google File System. HDFS is composed of Name Node and Data Node. HDFS stores each _le as a sequence of blocks (currently 64 MB by default) with all blocks in a _le the same size except for the last block .Blocks belonging to a _le are replicated for fault tolerance. The block size and replication factor are configurable per file. Files in HDFS are write once and can have only one writer at any given time.

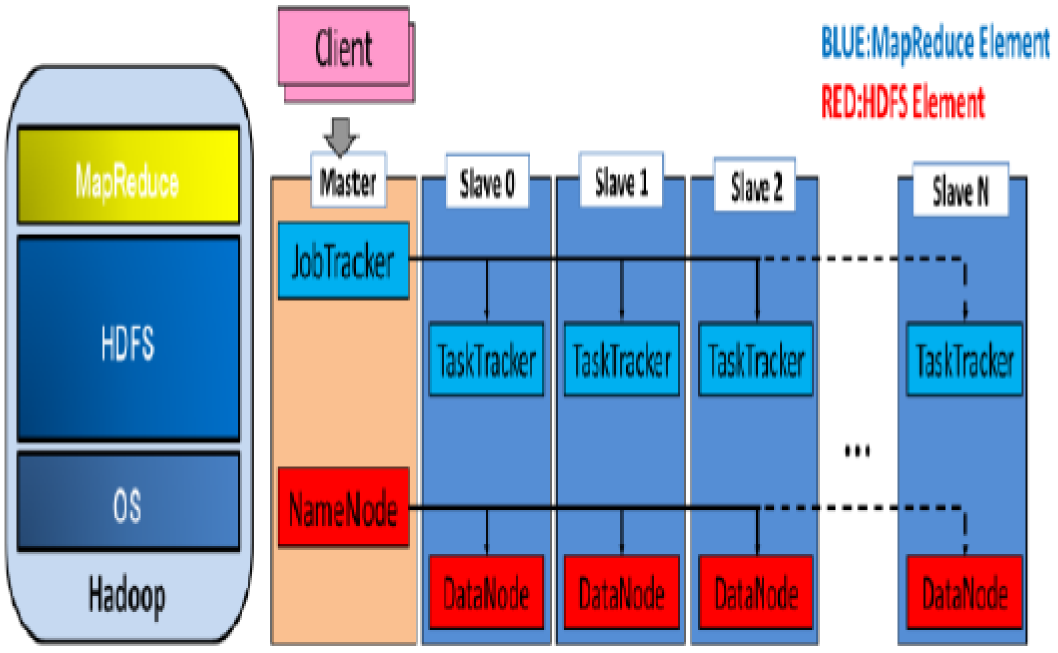


Figure 1: Structure with elements of MapReduce and HDFS

2.3 MapReduce

MapReduce (implemented on Hadoop) is a framework for parallel distributed processing large volumes of data. In programming using MapReduce, it is possible to perform parallel distributed processing by writing programs involving the following three steps: Map, Shuffle, and Reduce. Figure 2 shows an example of the flow when Map and Reduce processes are performed. Because Map Reduce automatically performs inter-process communications between

Map and Reduce processes, and maintain load balancing of the processes.

2.3.1 Map concept of data processing

The Map function takes a key-value pair <K, V> as the input and generates one or multiple pairs <K', V'> as the intermediate output.

2.3.2 Shuffle concept of data processing

After the Map phase produces the intermediate key-value pair or key-value pairs, they are efficiently and automatically grouped by key by the Hadoop system in preparation for the Reduce phase.

2.3.3 Reduce concept of data processing

The Reduce function takes as the input a <K', LIST V'> pair, where _LIST V' _ is a list of all V' values that are associated with a given key K'. The Reduce function produces an additional key-value pair as the output.

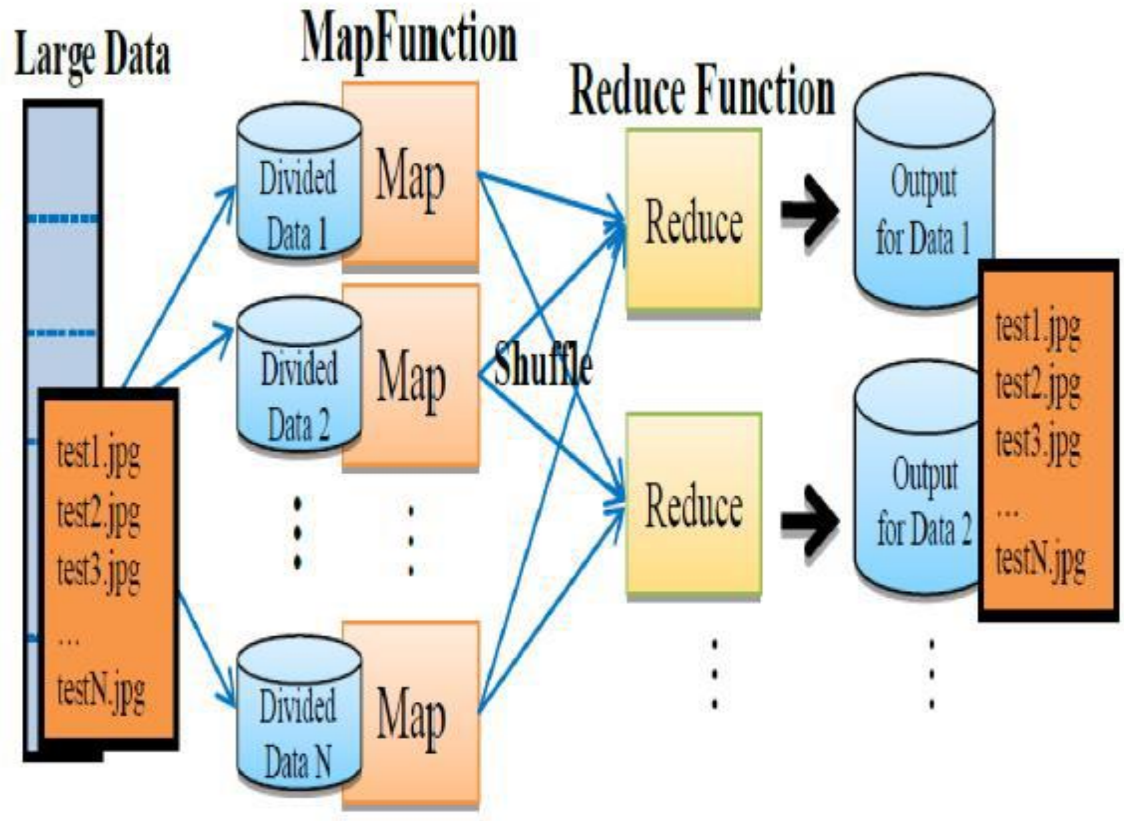


Figure 2: Processes performing the map and reduce phases

3. Iris Recognition

A biometric system provides automatic recognition of an individual based on some sort of unique feature or characteristic possessed by the individual. Biometric systems have been developed based on fingerprints, facial features, voice, hand geometry, handwriting, the retina and the iris.

Biometric systems work by rest capturing a sample of the feature, such as recording a digital sound signal for voice recognition, or taking a digital color image for face recognition. The sample is then transformed using some sort of mathematical function into a biometric template. The biometric template will provide a normalized, efficient and highly discriminating representation of the feature, which can then be objectively compared with other templates in order to determine identity. Most biometric systems allow two modes of operation .An enrollment mode for adding templates to a database, and an identification mode, where a template is created for an individual and then a match is searched for in the database of preenrolled templates.

A good biometric is characterized by use of a feature that is; highly unique so that the chance of any two people having the same characteristic will be minimal, stable so that the feature does not change over time, and be easily captured in order to provide convenience to the user, and prevent misrepresentation of the feature.

3.1 The Human Iris

The iris is a thin circular diaphragm, which lies between the cornea and the lens of the human eye. A front-on view of the iris is shown in Figure 3.1. The iris is perforated close to its center by a circular aperture known as the pupil. The function of the iris is to control the amount of light entering through the pupil, and this is done by the sphincter and the dilator muscles, which adjust the size of the pupil. The average diameter of the iris is 12 mm, and the pupil size can vary from 10% to 80% of the iris diameter.

The iris consists of a number of layers; the lowest is the epithelium layer, which contains dense pigmentation cells. The stromal layer lies above the layer, and contains blood vessels, pigment cells and the two iris muscles. The density of stromal pigmentation determines the color of the iris. The exrnally visible surface of the multilayered iris contains two zones, which often differ in colour. An outer ciliary zone and an inner pupillary zone, and these two zones are divided by the collaret which appears as a zigzag pattern.

3.2 Iris Recognition

Formation of the iris begins during the third month of embryonic life. The unique pattern on the surface of the iris is formed during the year of life, and pigmentation of the stroma takes place for the few years. Formation of the unique patterns of the iris is random and not related to any genetic factors. The only characteristic that is dependent on genetics is the pigmentation of the iris, which determines its color. Due to the epigenetic nature of iris patterns, the two eyes of an individual contain completely independent iris patterns, and identical twins possess uncorrelated iris patterns. For further details on the anatomy of the human eye consult the book by Wol.

The iris is an externally visible, yet protected organ whose unique pattern re- mains stable throughout adult life. These characteristics make it very attractive for use as a biometric for identifying individuals. Image processing techniques can be employed to extract the unique iris pattern from a digitized image of the eye, and encode it into a biometric template, which can be stored in a database. This biometric template contains an objective mathematical representation of the unique information stored in the iris, and allows comparisons to be made between templates. When a subject wishes to be identified by iris recognition system, their eye is photographed, and then a template created for their iris region. This template is then compared with the other templates stored in a database until either a matching template is found and the subject is identified, or no match is found and the subject remains unidentified.

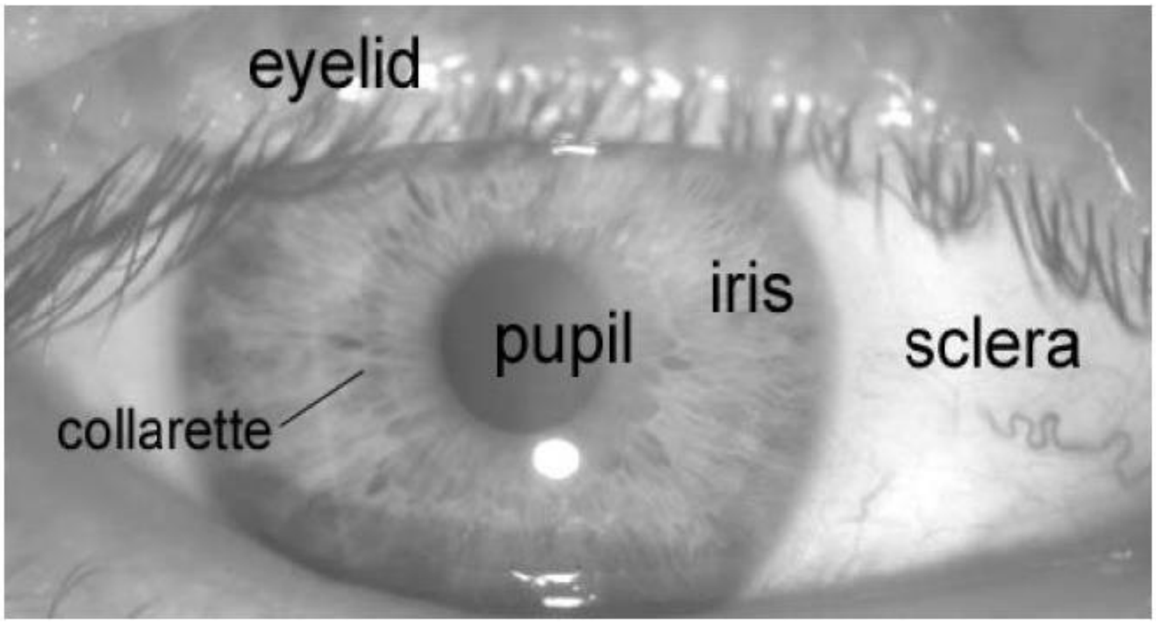


Figure 3: Iris image

These stages are Image acquisition deals with capturing sequence of iris images from the subject using cameras and sensors, Preprocessing involves various steps such as iris liveness detection, pupil and iris boundary detection, eyelid detection and removal and normalization, Feature extraction creating a template containing only the most discriminating features of the iris Feature Matching (Recognition) achieves result by comparison of features with stored patterns. The input to the system will be an eye image, and the output will be an iris template, which will provide a mathematical representation of the iris region.

3.2.1 Image Acquisition

Image acquisition deals with capturing sequence of iris images from the subject using cameras and sensors. An image acquisition consists of illumination, position and physical capture system. The occlusion, lighting, number of pixels on the iris are factors that affect the image quality. Many iris recognition systems require stern cooperation of the user for image acquisition. Ketchantang proposed a method in which the entire sequence of images is acquired dur ing enrollment and the best feasible images are selected, to increase flexibility. Enrollment aids to provide strong identity management.

3.2.2 Preprocessing

The purpose of preprocessing stage of iris recognition is to isolate the actual iris region in a digital eye image. The iris region, can be approximated by two circles, one for the iris/sclera boundary and another, interior to the, for the iris/pupil boundary. The eyelids and eyelashes normally occlude the upper and lower parts of the iris region. Also, specular reflections can occur within the iris region corrupting the iris pattern. A technique is required to isolate and exclude these arte facts as well as locating the circular iris region.

3.2.3 Hough Transform

The Hough transform is a standard computer vision algorithm that can be used to determine the parameters of simple geometric objects, such as lines and circles, present in an image. The circular Hough transform can be employed to deduce the radius and centre coordinates of the pupil and iris regions. An automatic segmentation algorithm based on the circular Hough transform is employed by Wildes et al. Firstly, an edge map is generated by calculating the derivatives of intensity values in an eye image and then thresholding the result. From the edge map, votes are cast in Hough space for the parameters of circles passing through each edge point. These parameters are the centre coordinates xc and yc, and the radius r, which are able to define any circle according to the equation

$$xc2 + y\ c2 - r\ 2 = 0 \quad (3.1)$$

A maximum point in the Hough space will correspond to the radius and center coordinates of the circle best defined by the edge points. The eyelids, in performing the preceding edge detection step, Wildes et al. bias the derivatives in the horizontal direction for detecting the eyelids, and in the vertical direction for detecting the outer circular boundary of the iris. boundary edge map if using all gradient data. Taking only the vertical gradients for locating the iris boundary will reduce influence of the eyelids when performing circular Hough transform, and not all of the edge pixels defining the circle are required for successful localization. Not only does this make circle localization more accurate, it also makes it more efficient, since there are less edge points to cast votes in the Hough space.

There are a number of problems with the Hough transform method. First of all, it requires threshold values to be chosen for edge detection, and this may result in critical edge points being removed, resulting in failure to detect circles/arcs. Secondly, the Hough transform is computationally intensive due to its brute-force' approach, and thus may not be suitable for real time applications. It was decided to use circular Hough transform for detecting the iris and pupil boundaries. This involves employing canny edge detection to generate an edge map. Gradients were biased in the vertical direction for the outer iris/sclera boundary, as suggested by Wildes et al. Vertical and horizontal gradients were weighted equally for the inner iris/pupil boundary.

3.2.4 Feature Extraction

In order to provide accurate recognition of individuals, the most discriminating information present in an iris pattern must be extracted. Only the significant features of the iris must be encoded so that comparisons between templates can be made. Most iris recognition systems make use of a band pass decomposition of the iris image to create a biometric template.
Feature extraction identifies the most prominent features for classification .Some of the features are x-y coordinates, radius, shape and size of the pupil, intensity values, orientation of the pupil ellipse and ratio between average intensity of two pupils. The features are encoded to a format suitable for recognition.

3.2.5 Feature Matching

The template that is generated in the feature encoding process will also need a corresponding matching metric, which gives a measure of similarity between two iris templates. This metric should give one range of values when comparing templates generated from the same eye, known as intra-class comparisons, and another range of values when comparing templates created from different irises, known as inter-class comparisons. These two cases should give distinct and separate values, so that a decision can be made with high confidence as to whether two templates are from the same iris, or from two different irises.

4. HIPI: A Hadoop Image Processing Interface

The amount of images being uploaded to the internet is rapidly increasing, with Facebook users loading over 2.5 billion new photos every month, however, applications that make use of this data are severely lacking. Cur- rent computer vision applications use a small number of input images because of the difficulty is in acquiring computational resources and storage options for large amounts of data. As such, development of vision applications that use a large set of images has been limited. The Hadoop MapReduce platform provides a system for large and computationally intensive distributed processing, though use of Hadoops system is severely limited by the technical complexities of developing useful applications. To immediately address this, we propose an open source Hadoop Image processing interface (HIPI) that aims to create an interface for computer vision with MapReduce technology. HIPI abstracts the highly technical details of Hadoop's system and is flexible enough to implement many techniques in current computer vision literature.

Many image processing and computer vision algorithms are applicable to large-scale data tasks. It is often desirable to run these algorithms on large data sets (e.g. larger than 1 TB) that are currently limited by the computational power of one computer. These tasks are typically performed on a distributed system by dividing the task across one or more of the following features: algorithm parameters, images, or pixels. Performing tasks across a particular parameter is incredibly parallel and can often be perfectly parallel. Face detection and landmark classification are examples of such algorithms. The ability to parallelize such tasks allows for scalable, efficient execution of resource-intensive applications. The MapReduce framework provides a platform for such applications.

Basic vision applications that utilize Hadoop’s MapReduce framework require a staggering learning curve and overwhelming complexity. The overhead required to implement such applications severely cripples the progress of researchers. HIPI removes the highly technical details of Hadoops system and provides users with the familiar feel of an image library with the access to the advanced resources of a distributed system. Our platform is focused around giving users unprecedented access to image-based data structures with a pipeline that is intuitive to the MapReduce system, allowing for easy and flexible use for vision applications. Because of the similar goals in our frameworks, we take particular inspiration from the project as a model for designing an open API to provide access to computing resources

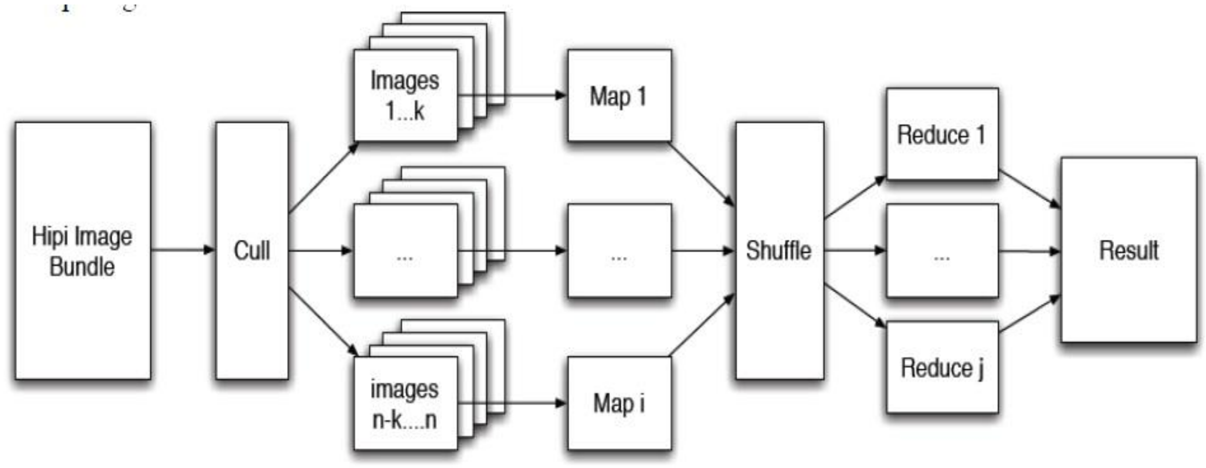


Figure 4: A typical MapReduce pipeline using our Hadoop Image Processing Interface with n images, i map nodes, and j reduces nodes.

4.1 The HIPI Framework Software Requirements

HIPI was created to empower researchers and present them with a capable tool that would enable research involving image processing and vision to be performed extremely easily. With the knowledge that HIPI would be used for researchers and as an educational tool, we designed HIPI with the following goals in mind.

1. Provide an open, extendible library for image processing and computer vision applications in a MapReduce framework.

2. Store images e_ciently for use in MapReduce applications.

3. Allow for simple filtering of a set of images.

4. Present users with an intuitive interface for image-based operations and hide the details of the MapReduce framework.

5. HIPI will set up applications so that they are highly parallelized and balanced so that users do not have to worry about such details.

4.1.1 Data Storage

Hadoop uses a distributed file system to store files on various machines throughout the cluster. Hadoop allows files be accessed, however, without knowledge of where it is stored in the cluster, so that users can reference files the same way they would on a local machine and Hadoop will present the file accordingly.

When performing MapReduce jobs, Hadoop attempts to run Map and Reduce tasks at the machines were the data being processed is located so that data does not have to be copied between machines. As such, MapReduce tasks run more efficiently when the input is one large file as proposed to many small files. Large files are significantly more likely to be stored on one machine whereas many small files will likely be spread out among many diferent machines, which require significant overhead to copy all the data to the machine where the Map task is running. This overhead can slow the runtime ten to one hundred times. Simply put, the MapReduce framework operates more efficiently when the data being processed is local to the machines performing the processing.

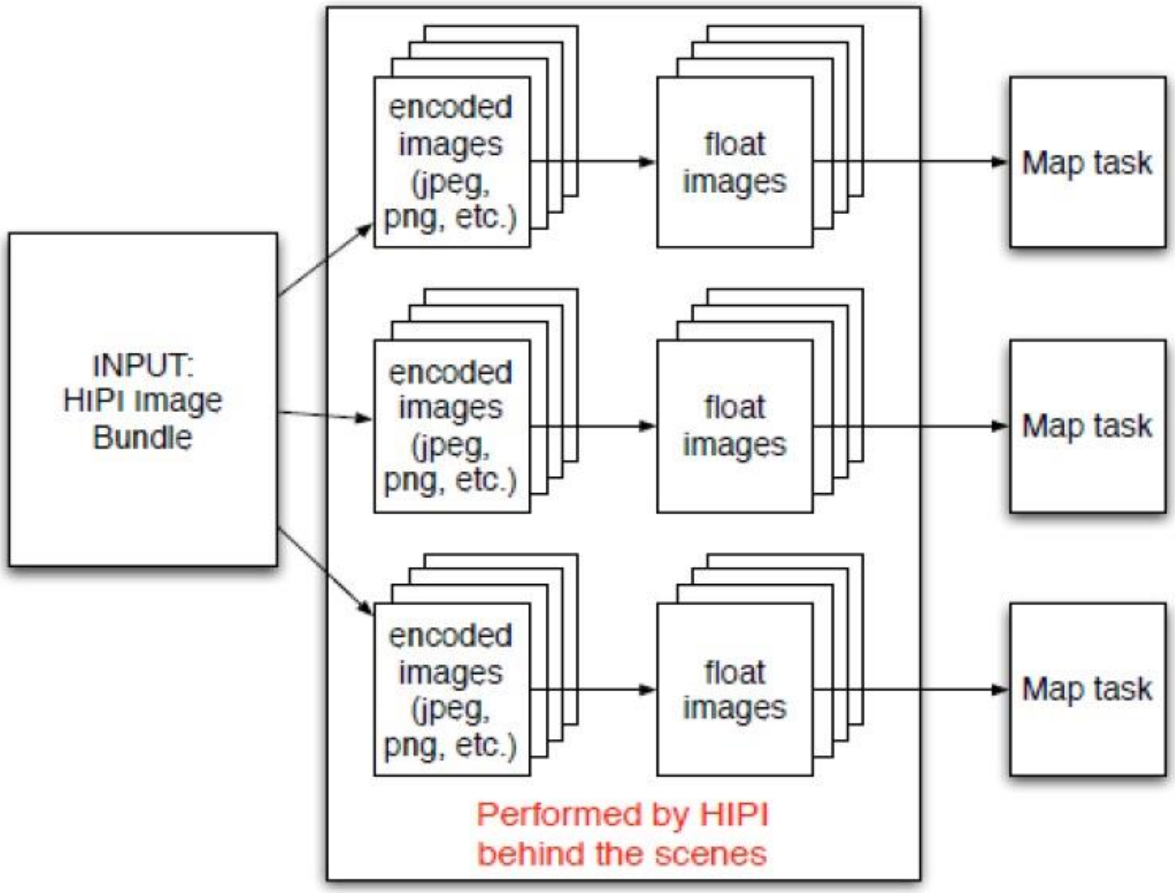

INPUT: HIPI Image Bundle
encoded images (jpeg, png, etc.)
float images
Map task
encoded images (jpeg, png, etc.)
float images
Map task
encoded images (jpeg, png, etc.)
float images
Map task
Performed by HIPI behind the scenes

Figure 5: HIPI IMAGE BUNDLE

With this in mind, we created a HIPI Image Bundle data type that stores many images in one large file so that MapReduce jobs can be performed more efficiently. A HIPI Image Bundle consists of two files: a data file containing concatenated images and an index file containing information about the offsets of images in the data file as shown. This setup allows us to easily access images across the entire bundle without having to read in every image.

We observed several benefits of the HIPI Image Bundle in tests against Hadoop's Sequence file and Hadoop Archive (HAR) formats. HARs are only useful for archiving files (as backups), and may actually perform slower than reading in files the standard way. Sequence files perform better than standard applications for small files, but must be read serially and take a very long time to generate. HIPI Image Bundle have similar speeds to Sequence files, do not have to be read serially, and can be generated with a MapReduce program .Additionally, HIPI Image Bundles are more customizable and are mutable, unlike Sequence and HAR files. For instance, we have implemented the ability to only read the header of an image file using HIPI Image Bundles, which would be considerably more difficult with other file types. Further features of the HIPI Image Bundles are highlighted

4.1.2 Image-based MapReduce

Standard Hadoop MapReduce programs handle input and output of data very effectively, but struggle in representing images in a format that is useful for researchers. Current methods involve significant overhead to obtain standard float image representation. For instance, to distribute a set of images to a set of Map nodes would require a user to pass the images as a String, and then decode each image in each map task before being able to do access pixel information. This is not only inefficient, but inconvenient. These tasks can create headaches for users and make the code look cluttered and difficult to interpret the intent of the code. As such, sharing code is less efficient because the code is harder to read and harder to debug. Our library focuses on bringing familiar image-based data types directly to the user for easy use in MapReduce applications. Using the HIPI Image Bundle data type as inputs, we have created an input specification that will distribute images in the HIPI Image Bundle across all map nodes. We distribute images such that we attempt maximize locality between the mapper machines and the machine where the image resides. Typically, a user would have to create Input Format and Record Reader classes that specify how the MapReduce job will distribute the input, and what information gets sent to each machine. This task is nontrivial and often becomes a large point of headaches for users. We have included Input Format and Record Readers that take care of this for the user. Our specification works on HIPI Image Bundles for various image types, sizes, and varying amounts of header and ex if information. We handle all of these different image permutations behind the scenes to bring images straight to the user as float images. No work is needed to be done by the user, and float images are brought directly to the Map tasks in a highly parallelized fashion.

During the distribution of inputs but before the map tasks start we introduce a culling stage to the Reduce pipeline. The culling stage allows for images to be filtered based on image properties. The user specifies a culling class that describes how the images will be filtered (e.g. pictures smaller than 10 megapixels, pictures with GIS location header data). Only images that pass the culling stage will be distributed to the map tasks, preventing unnecessary copying of data. This process is often very efficient because culling often occurs based on image header information, so it is not required to read the entire image.

5. SYSTEM DESIGN

5.1

a) Data Flow Diagram

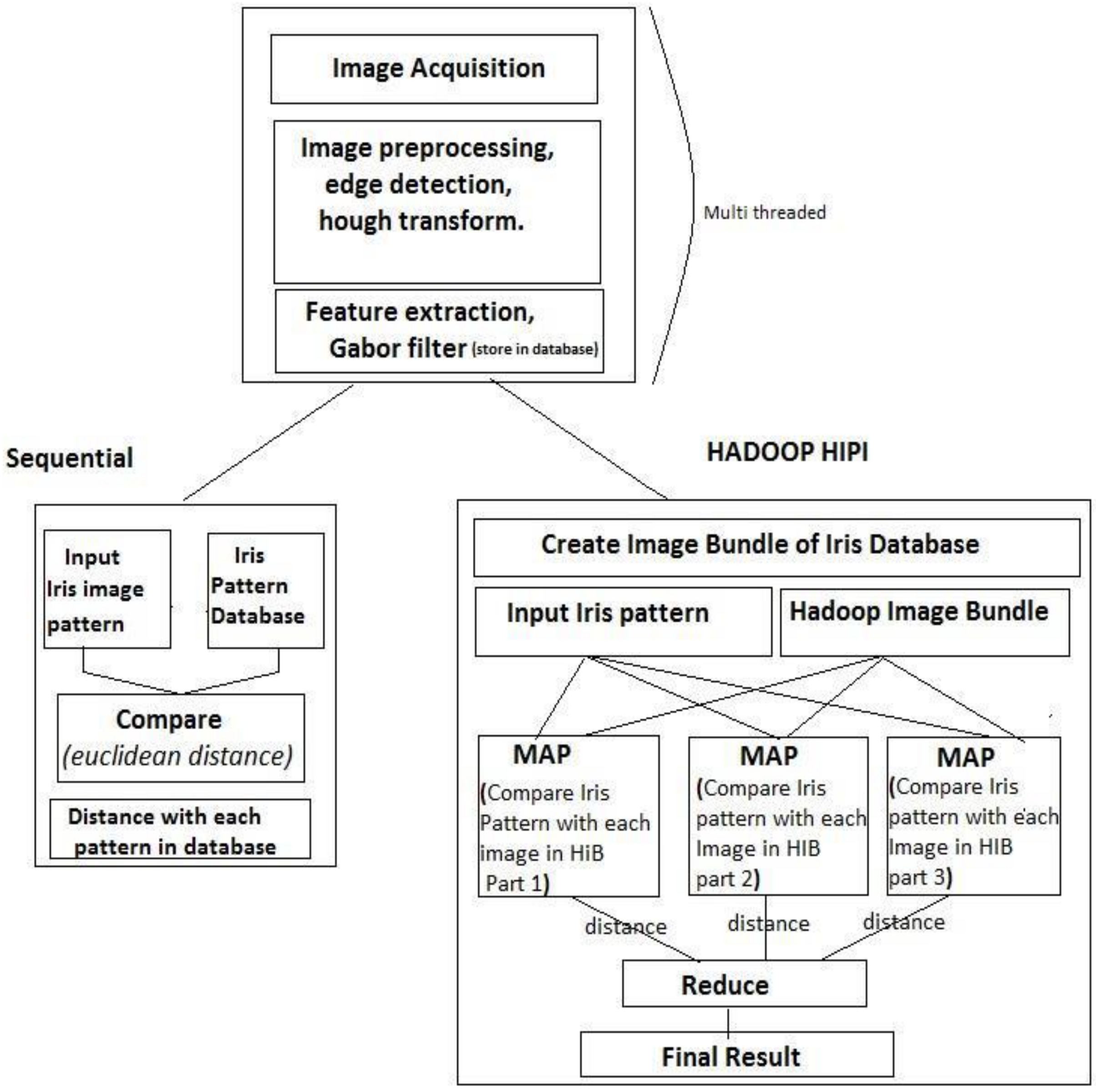
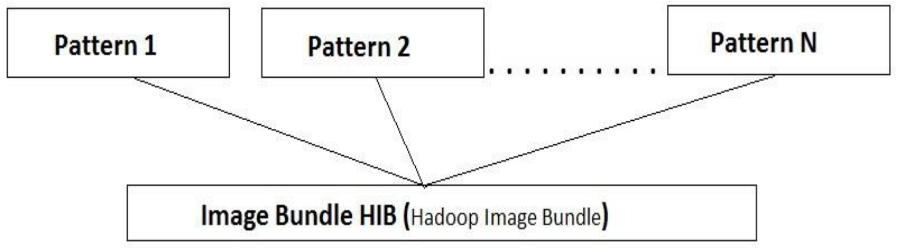
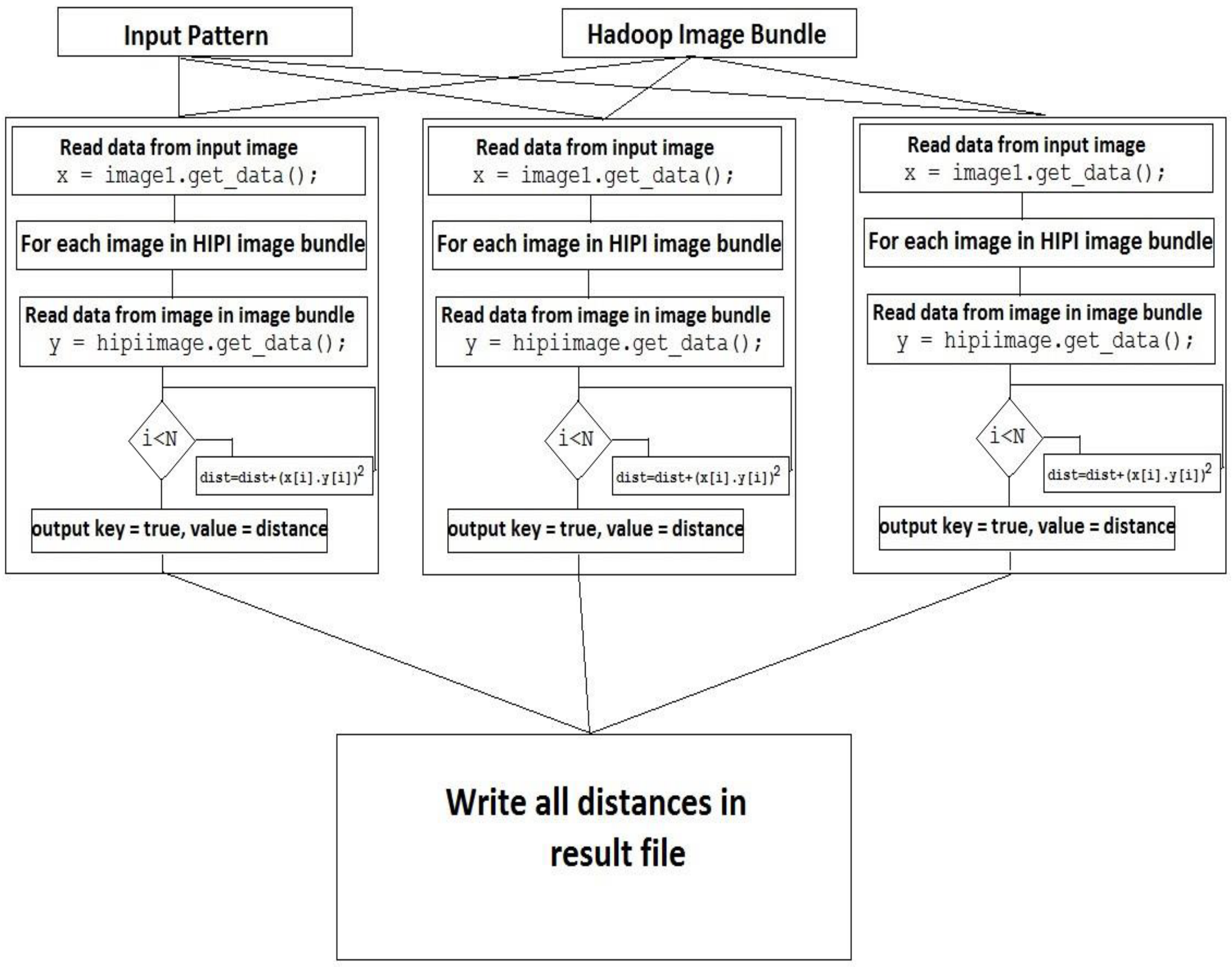


Image Acquisition
Image preprocessing, edge detection, hough transform.
Feature extraction, Gabor filter (store in database)
Multi threaded
Sequential
HADOOP HIPI
Input Iris image pattern
Iris Pattern Database
Compare (euclidean distance)
Distance with each pattern in database
Create Image Bundle of Iris Database
Input Iris pattern
Hadoop Image Bundle
MAP (Compare Iris Pattern with each image in HiB Part 1)
MAP (Compare Iris pattern with each Image in HIB part 2)
MAP (Compare Iris pattern with each Image in HIB part 3)
distance
distance
distance
Reduce
Final Result

b) Iris Recognition Process




Input Pattern
Hadoop Image Bundle
Read data from input image
x = image1.get_data();
For each image in HIPI image bundle
Read data from image in image bundle
y = hipiimage.get_data();
i<N
dist=dist+(x[i].y[i])^2
output key = true, value = distance
Write all distances in result file

5.2 Data Flow View

A)

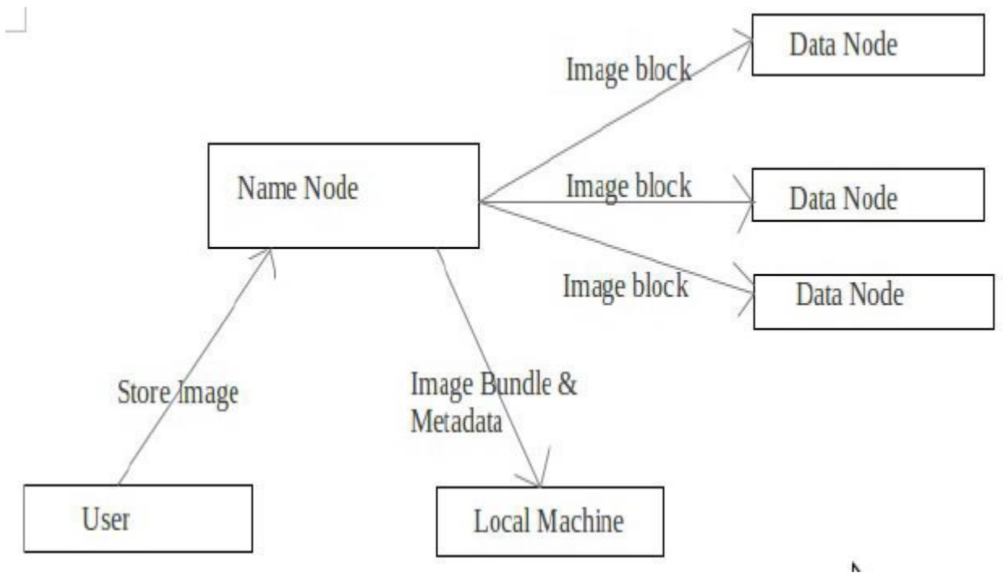


Figure 7: (a) Data flow diagram for storing images

B)

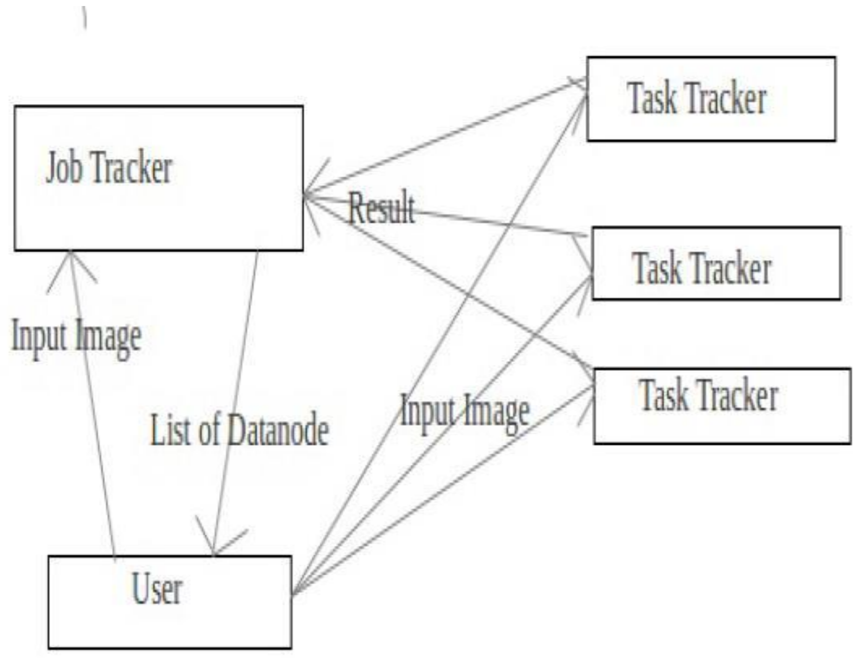


Figure 8: (b) Data flow diagram for user image recognition

5.3 Sequence Diagram

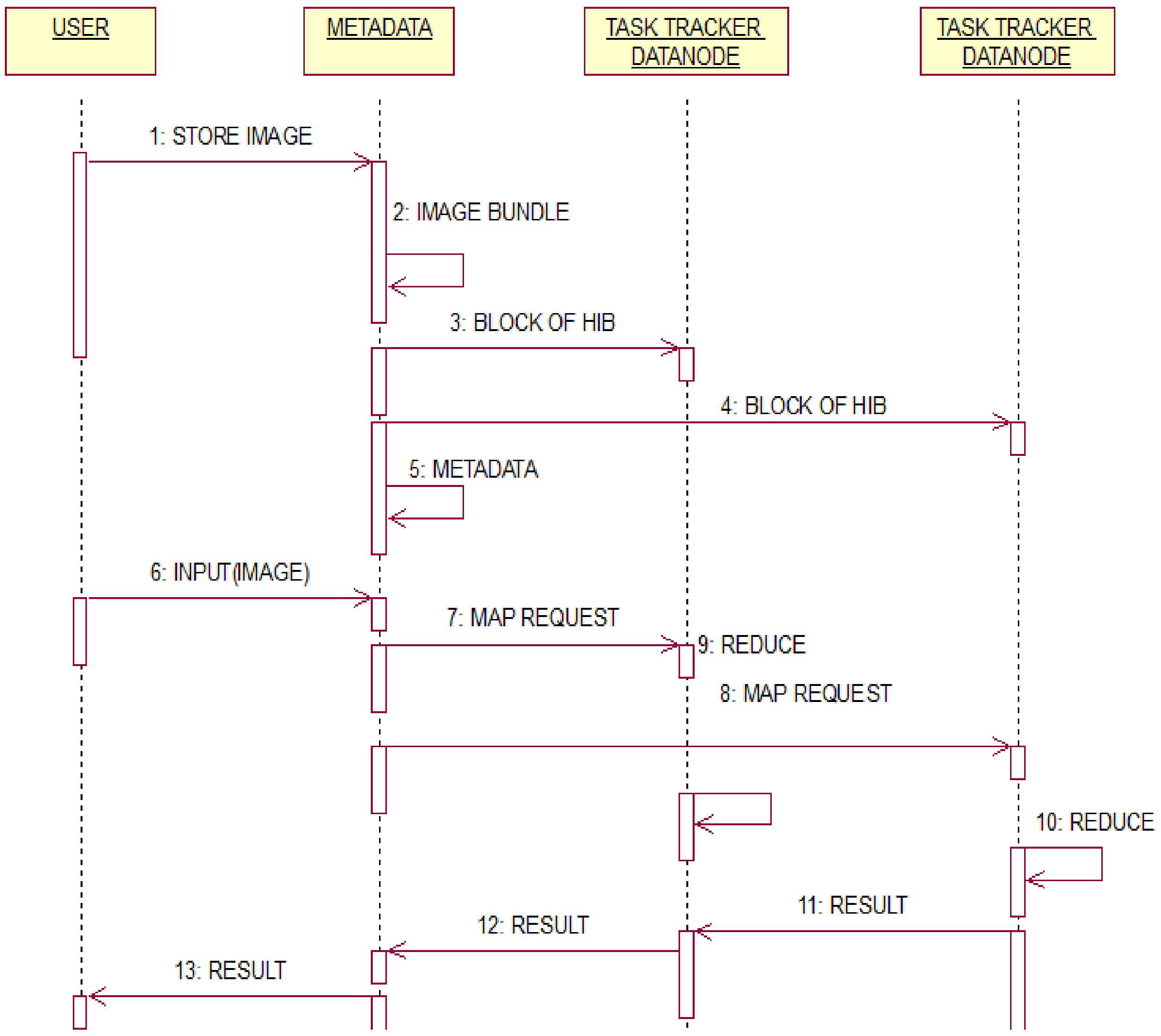


Figure 9: Sequence Diagram

5.4 Use case Diagram

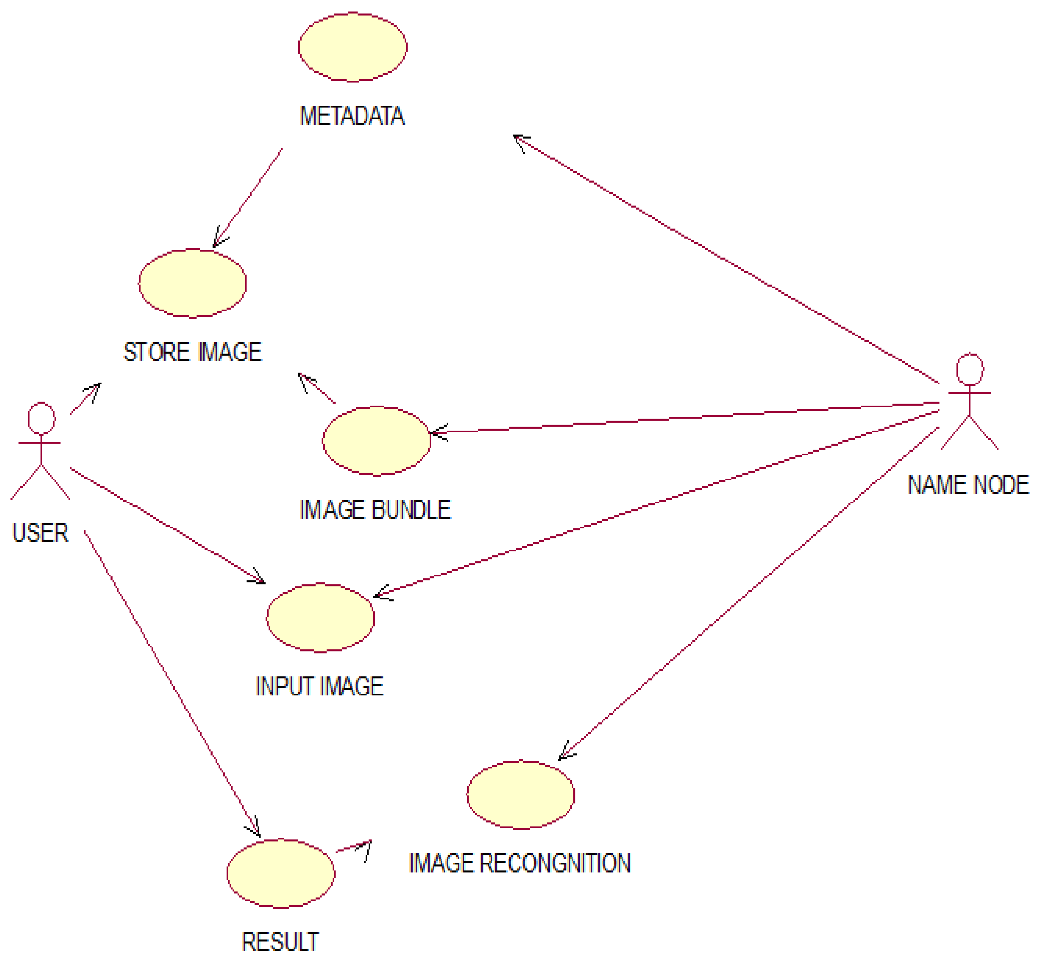


Figure 9: Use Case Diagram.

6. CONCLUSION

This project has described our library for image processing and vision applications on a MapReduce framework - Hadoop Image Processing Interface (HIPI). This library was carefully designed to hide the complex details of Hadoop’s powerful MapReduce framework and bring to the forefront what users care about most: images. Our system has been created with the intent to operate on large sets of images. We provide a format for storing images for efficient access within the MapReduce pipeline, and simple methods for creating such files. By providing a culling stage before the mapping stage, we give the user a simple way to filter image sets and control the types of images being used in their MapReduce tasks.

Finally, we provide image encoders and decoders that run behind the scenes and work to present the user with float image types which are most useful for image processing and vision applications. Through these features, our interface brings about a new level of simplicity for creating large-scale vision applications with the aim of empowering researchers and teachers with a tool for efficiently creating MapReduce applications focused around images. This paper describes two example applications built with HIPI that demon strate the power it presents users with. We hope that by bring the resources and power of MapReduce to the vision community that we will enhance the ability to create new vision projects that will enable users to push the field of computer vision.

7. REFERENCES

[1] http://www.technicalsymposium.com/Iris_Recognition.html

[2] Wikipedia web site: [online available] http://en.wikipedia.org/wiki/Iris_recognition

[3] 10 reasons Why Cloud Computing is the Wave of the Future, April 29, 2009, by Gilberto J. Perera

URL: http://laptoplogic.com/resources/10-reasons-why-cloud-computing-is-the-wave-of-the-future

[4] SLIC: Short-Length Iris Codes, James E. Gentile, Nalini Ratha and Jonathan Connell IBM T.J.

Watson Research Center

[5] Wikipedia on cloud computing [online]

URL: http://en.wikipedia.org/wiki/Cloud_computing

[6] http://www.ibm.com/developerworks/aix/library/au-cloud_apache/

[7] T. White, Hadoop: The Definitive Guide. O'Reilly Media, [online]:

URL: http://oreilly.com/catalog/0636920010388/preview

[8] Apache Hadoop HDFS Architecture Guide. [Online]

URL: http://hadoop.apache.org/hdfs/docs/current/hdfs_design.html

[9] A Method for Iris Recognition Based on 1D Coiflet Wavelet Agus Harjoko, Sri Hartati, and Henry Dwiyasa

[10] Online available: http://scgwww.epfl.ch/courses/Biometrics-Lectures-2010-2011-pdf

[11] “Don’t Blink: Iris Recognition for Biometric Identification ”, Mary Dunker , SANS Security Essentials, July 2003 , GSEC Certification Practical, Version 1.4b

[12] S. Sanderson, J. Erbetta. “Authentication for secure environments based on iris scanning technology”. IEE Colloquium on Visual Biometrics, 2000