Video Processing Using Hadoop

(UDP Project)

A PROJECT REPORT

Submitted by

110010107021– AAYUSHI J. MAHESHWARI

110010107022– DARSHIL D. PATEL

Internal Guide: Prof. SIDDHARTH SHAH

In fulfilment for the award of the degree of

Bachelor of Engineering

In

COMPUTER ENGINEERING

A. D. PATEL INSTITUTE OF TECHNOLOGY,
NEW VALLABH VIDYANAGAR

GUJARAT TECHNOLOGICAL UNIVERSITY, AHMEDABAD

May, 2015

TABLE OF CONTENTS

List of Figures

List of Tables

ACKNOWLEDGEMENT

The completion of any project would bring sense of satisfaction. However, it would not have been possible without the kind support and help of many individuals and organizations. It gives immense pleasure to extend our sincere thanks to all of them.

Our deepest gratitude is to Almighty God and our parents for holding our hands and guiding us throughout our life.

We are highly indebted to our guiding teacher **Mr. Siddharth Shah** for his guidance and constant supervision as well as for providing necessary information regarding the project.

We would also like to thank all the staff members and **H.O.D Dr. Ramji Makwana** of **Computer Engineering Department** for providing us with required facilities and support towards our project.

Our special thanks and appreciation also goes to our friends especially *Ms. Purva Thakkar* to figure out a way of drafting this Project Report and to all the people who have willingly helped us out with their abilities, especially when we were against the wall in the need of help.

AAYUSHI MAHESHWARI **DARSHIL PATEL**

A. D. PATEL INSTITUTE OF TECHNOLOGY, NEW VALLABH VIDYANAGAR, GUJARAT – INDIA

GUJARAT TECHNOLOGICAL UNIVERSITY

DEPARTMENT OF COMPUTER ENGINEERING

2014-2015

CERTIFICATE

Date:

This is to certify that the Project -**"Video Processing Using Hadoop"** has been carried out by **Aayushi Maheshwari (110010107021), Darshil Patel (110010107022)** under my guidance in fulfilment of the subject Project – II under the degree of Bachelor of Engineering in **COMPUTER ENGINEERING (8th Semester) of Gujarat Technological University, Ahmedabad** during the academic year 2014-15.

Internal Guide:
Prof. Siddharth Shah
Computer Engineering Dept.
A.D.I.T.

Head of Department:
Dr. Ramji M. Makwana
Computer Engineering Dept.
A.D.I.T.

Plagiarism Report For 'reportfinal.docx'

How does Viper work.....?

[+] Read more..

Location	Title	Words Matched	Match (%)	Unique Words Matched	Unique Match (%)
http://stackoverflow.com/questions/23669638/installing-opencv-on-ubuntu-12-04	Installing OpenCV on Ubuntu 12.04 - Stack Overflow	60	1	60	1
http://en.wikipedia.org/wiki/Mapreduce	MapReduce - Wikipedia, the free encyclopedia	151	2	151	2
http://www.michael-noll.com/tutorials/running-hadoop-on-ubuntu-linux-single-node-cluster/	Running Hadoop On Ubuntu Linux (Single-Node Cluster ...	481	6	481	6
http://www.sas.com/en_us/insights/big-data/hadoop.html	Hadoop and big data - SAS	214	3	214	3
http://www.knowbigdata.net/map-reduce-html/	MAP REDUCE	MAKE IN INDIA	151	2	0	<1
http://blog.pivotal.io/data-science-pivotal/products/using-hadoop-mapreduce-for-distributed-video-transcoding	Using Hadoop MapReduce for Distributed Video Transcoding ...	56	1	56	1
http://en.wikipedia.org/wiki/Eigenfaces	Eigenface - Wikipedia, the free encyclopedia	50	1	50	1
http://www.namhuy.net/1205/how-to-install-opencv-on-ubuntu.html	How to install OpenCV on Ubuntu - Nam Huy Linux	113	1	54	1
https://www.scribd.com/doc/131157168/hadoop-in-practice-pdf	hadoop_in_practice.pdf - Scribd	208	3	208	3
http://www.authorstream.com/Presentation/ikamalsharma-2333431-hadoop-practice/	Hadoop in Practice |authorSTREAM	211	3	3	<1
http://hadoop.apache.org/	Welcome to Apacheâ„¢ HadoopÂ®!	88	1	88	1
http://tutorialshadoop.com/setup-hadoop-ssh-configuration-linux-ubantu/	How To Setup Hadoop SSH Configuration In Linux Ubantu ...	118	2	0	<1
https://elementztechblog.wordpress.com/2014/07/28/installing-opencv-2-4-9-in-linux/comment-page-1/	Installing OpenCV 2.4.9 in Linux	Random Codes - Elementz ...	114	1	6	<1
http://answers.opencv.org/question/54955/not-able-to-install-opencv/	Not able to install OpenCV? - OpenCV Q&A Forum	98	1	0	<1
http://stackoverflow.com/questions/16936745/hadoop-home-is-deprecated-hadoop	java - $HADOOP_HOME is deprecated ,Hadoop - Stack Overflow	47	1	0	<1
https://hemprasad.wordpress.com/2014/06/23/opencv-installation-for-ubuntu-12-04-2/	OpenCV installation for Ubuntu 12.04 Â« Something More for ...	318	4	212	3
http://maexadata.in/entries/platforms/hadoop-map-reduce-2	Hadoop Map Reduce	maexadata - Welcome to the world of mega ...	161	2	10	<1
http://whoisusing.it/mapreduce	WhoIsUsing.it - Who is using MapReduce	118	2	0	<1
http://mysolvedproblem.blogspot.com/2012/05/installing-hadoop-on-ubuntu-linux-on.html	[Solved] Problem: Installing Hadoop on Ubuntu (Linux ...	55	1	0	<1
https://saphack.wordpress.com/2014/12/23/big-data-and-hadoop/	Big data and Hadoop	SAPHACK	171	2	0	<1
http://stackoverflow.com/questions/13211745/detect-face-then-autocrop-pictures	python - Detect face then autocrop pictures - Stack Overflow	112	1	112	1
http://stackoverflow.com/questions/19044225/what-is-the-exact-difference-between-hadoop-and-cloud-computing	What is the exact difference between Hadoop and Cloud ...	40	1	0	<1
http://www.oracle.com/technetwork/systems/hands-on-labs/hol-setup-hadoop-solaris-2041770.html	How to Set Up a Hadoop Cluster Using Oracle Solaris	59	1	0	<1

Documents found to be plagiarised

Matching Content: 23%

Master Document Text

Video Processing Using HadoopA PROJECT REPORTSubmitted by110010107021- AAYUSHI J. MAHESHWARI110010107022- DARSHIL D. PATELInternal Guide: Prof
SIDDHARTH SHAHIn fulfilment for the award of the degree ofBachelor of EngineeringInCOMPUTER ENGINEERINGD. PATEL INSTITUTE OF

TECHNOLOGY,NEW VALLABH

VIDYANAGARGUJARAT TECHNOLOGICAL UNIVERSITY, AHMEDABADMay, 2015 INDIAGUJARAT TECHNOLOGICAL UNIVERSITY DEPARTMENT OF COMPUTER ENGINEERING 2014-2015 CERTIFICATE Date: This is to

Processing Using Hadoop" has been carried out by Aayushi Maheshwari (110010107021), Darshil Patel (110010107022) under my guidance in the subject Project - II under the degree of Bachelor of Engineering in COMPUTER ENGINEERING (8th Semester) of

Ahmedabad during the academic year 2014-15. < Internal Guide: t> r w:rsidRPA.D.I.T. GTU Team ID: 3344Project Team MemberSr.NoEnrollment No.NameDisciplineEmailMobile No.1110010107021Maheshwari Aayushi

JagdishComputer Engineeringayushimaheshwari1412@gmail.com9429950749211001010702 2Patel Darshil DineshKumarComputer Engineeringdarshil_dpatel@yahoo.co

in8460584110PROJECT TITLEVideo Processing Using HadoopINDUSTRY DETAILSIndustry NameA D Patel Institute of TechnologyRepresentative's NameRepresentative's EmailRepresentative's Contact NumberPROJECT GUIDE DETAILSNameProf. Siddharth ShahEmailsiddharth7763@gmail.comContact Number

ACKNOWLEDGEMENTThe completion of any project would bring sense of satisfaction. However, it would not have been possible without the kind support and help of many individuals and organizations. It gives immense pleasure to extend our sincere thanks to all of them. Our deepest gratitude is to Almighty God and our parents for holding our hands and guiding us throughout our life. We are highly indebted to our guiding teacher Mr. Siddharth Shah for his guidance and constant supervision as well as for providing necessary information regarding the project. We would also like to thank all the staff members and H.O.D Dr. Ramji Makwana of Computer Engineering Department for providing us with required facilities and support towards our project.Our special thanks and appreciation also goes to our friends especially Ms. Purva Thakkar to figure out a way of drafting this Project Report and to all the people who have willingly helped us out with their abilities, especially when we were against the wall in the need of help

AAYUSHI MAHESHWARI DARSHIL PATEL time consuming process. Hadoop is an open-source software framework for storing and processing big data in a distributed fashion on large clusters of

commodity hardware. Essentially, it accomplishes two tasks: massive data storage and faster processing.Open-source software. Open source software differs from commercial

software due to the broad and open network of developers that create and manage the programs. Traditionally, it's free to download, use and contribute to, though more and more commercial versions of Hadoop are becoming available.Framework. In this case, it means everything you need to develop and run, your software applications is provided - programs, tool sets, connections, etc.Distributed. Data is divided and stored across multiple computers, and computations can be run in parallel across multiple connected machines.Massive storage. The Hadoop framework can store huge amounts of data by breaking the data into blocks and storing it on clusters of lower-cost commodity hardware.Faster processing. Hadoop processes large amounts of data in parallel across clusters of tightly connected low-cost computers for quick results.With the ability to economically store and process any kind of data (not just numerical or structured data), organizations of all sizes are taking cues from the corporate web giants that have used Hadoop to their advantage (Google, Yahoo, Etsy, eBay, Twitter, etc.).Moving forward to next phase of this project is to detect faces out from the video, which can be of great help reducing a lot of time scanning a large video. OpenCV is a very useful library which can be integrated along with the python programming language. OpenCV again is an open source library that is developed in C++. There are many situations where an organization has to scan the video footages of its various departments or there retail stores, so that they can make out how many and who all visited the respective department or that retail store. So that company can be ready in the situations like thefts or some
disasters. Hereby we are trying to create a system which can help us achieve the above mentioned goal. Thus amalgamation of Big Data and Python along with OpenCV can be of great help. We have taken a step towards the enlightenment of this domain by means of our final year project.TABLE OF

CONTENTSTitle pageiCertificate PageiiPhase IiiiPhase IIivAcknowledgementvAbstractviTable of ContentsviiList of FiguresixList of TablesxChapter

 Typically these surveillance video clips are stored as compressed video files corresponding to video for several hours.There is a need to process such huge video data sets to enable a quick summary of "interesting" events that happened during a specified time frame in a particular location.For instance one might be interested in anomalous trajectories of the objects within a scene.And thus in order to process such large scale data we need framework called hadoop.Video processing uses hardware, software, and combinations of the two for editing the images and sound recorded in video files. Extensive algorithms in the processing software and the peripheral equipment allow the user to perform editing functions using various filters. The desired effects can be produced by editing

frame by frame or in larger batches.A MapReduce program is composed of aÿMap()ÿprocedure that performs filtering and sorting (such as sorting students by first name into queues, one queue for each name) and aÿReduce ÿprocedure that performs a summary operation (such as counting the number of students in each queue, yielding name frequencies). The "MapReduce System" (also called "infrastructure" or "framework") orchestrates byÿmarshalling the distributed servers, running the various tasks in parallel

managing all communications and data transfers between the various parts of the system, and providing forÿredundancyÿandÿfault tolerance.The model is inspired by theÿmapÿandÿreduceÿfunctions commonly used inÿfunctional programmingÿalthough their purpose in the MapReduce framework is not the same as in their original forms.ÿThe key contributions of the MapReduce framework are not the actual map and reduce functions, but the scalability and fault tolerance achieved for a variety of applications by optimizing the execution engine once. As such, aÿsingle-threadedÿimplementation of MapReduce will usually not be faster than a traditional implementation. Only when the optimized distributed shuffle operation (which reduces network communication cost) and fault tolerance features of the MapReduce framework come into play, is the use of this model beneficial.CHAPTER-2 LITERATURE REVIEWFor our project, we are using the Apacher HADOOP and OpenCV.2.1 Apache HADOOPThe Apache Hadoop software library is a framework that

allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.HADOOP is installed to configure clusters. HADOOP also implements the HADOOP FileSystem(HDFS) in the clusters(i.e., the master node and the slave node). This is a distributed filesystem distributed over different nodes of the cluster.There are two components of the filesystem:Name NodeData Node.The Name Node is the node in the filesystem which keeps track of the data stored on the Data Nodes. The Name Node consists of names of all the Data Nodes and the files in those Data Nodes.The Data Nodes are the nodes which are submissive in nature and contains all the files and directories of the HDFS.HADOOP also has an interface JobTracker, which is used in clusters. In a typical HADOOP cluster, there is 1 master node and few slave nodes. When a client sends a request to the Master Node, it starts the JobTracker service. And the

is accomplished using Map/Reduce paradigm. The Map/Reduce paradigm is a programming paradigm using which, the master node fragments the program and each fragment or more are sent to individual slave nodes. The slave nodes process the fragments on its own and sends back the result to the Master node. And the master node has the responsibility to

merge those results and give the final result to the user. This is accomplished using the JobTracker service on the master node. Using this service, the jobs are assigned and scheduled to the slave nodes. And also result retrieval and collection is done by the JobTracker service.The Map/Reduce paradigm is inspired by Google's Map/Reduce paper.The following image depicts the

functional layers of a typical HADOOP cluster:Fig. 1: Layers of Services in a HADOOP clusterAs shown in Fig. 1, there are three nodes in the one namenode(analogous to Master node) which is marked as "hadoop-namenode" in the figure. And two datanodes(analogous to Slave nodes). Each node runs two functional layers, namely the "MapReduce Layer", and "HDFS Layer".Each of these layers have their functions to perform to run a MapReduce Job.The MapReduce Layer is responsible for running the Job, i.e., distributing and scheduling the tasks to the datanodes, retrieving the individual results from the datanodes and combining the results of every datanode to give the final result of the Job to the Namenode.A Job is the processing task requested by any client to the cluster. Whereas, a Task is assigned to the datanodes by dividing the job equally to the datanodes. So in simpler words, a "Job" is divided into "Tasks" and each datanode runs its own task locally, and all datanodes work simultaneously, thus decreasing the
computing speed. All this is done by the MapReduce Layer by exploiting the services of JobTracker on the namenode and TaskTracker on the datanode.The HDFS Layer is responsible for keeping track of the files on the HADOOP FileSystem. The DataNode is where the actual data or files of the HDFS is mounted and NameNode is where all the data is tracked on the whole cluster. It contains information about which files are stored where in the cluster wFig. 2: MapReduce Job execution FlowThe Fig.2 shows the execution flow of a MapReduce Job. First, a client request is accepted. Then, the Job divided into tasks using the Mapper function, which is called Mapping in HADOOP terminology. Then these tasks created by Mapper function is further processed by Reducer Function which performs the business logic. The corresponding outputs of each Reducer are compiled to form an output meant for the client.2.2 OpenCVOpenCV is an open source (see http://opensource.org) computer vision library available from http://SourceForge.net/projects opencvlibrary. The library is written in C and C++ and runs under Linux, Windows and Mac OS X. OpenCV is aimed at providing the basic tools needed to solve computer vision problems. In some cases, high-level functionalities in the library will be sufficient to solve the more complex problems in computer vision. Even when this is not the case, the basic components in the library are complete enough to enable creation of a complete solution of your own to almost any computer vision problem. In the latter case, there are several tried-and-true methods of using the library; all of them start with solving the problem using as many available library components as possible.We may use SimpleCV or OpenCV depending on circumstances which will be more suitable. Since this is not much of a concern for the time

being. We need to use CV libraries as ultimately our input image will be compared to the frames of the videos. Thus for the comparision we will use CV libraries.CHAPTER-3

RequirementsThe system will take video file as an input and divide it into respective frames.The frame would be consisting of some face images, i.e Image with faces in it, the system would divide the image into small parts that consists of only cropped faces.Finally user is left with only a folder in which he has faces of all the persons present in the video which makes task easier for the user.Now the user can check if the face that is detected is present in our database or not directly using the system.Non-Functional Requirements The system must be light weight and use minimum resources to reduce operational costs.Interactive GUI and real time notifications to the users regarding important system events.Following are the

tools used: Hadoop, OpenCV, and PIL.User should have flexibility of accessing the generated images.CHAPTER-4 used for developmentOS required : Ubuntu 14.04(linux system)Tool required to implement clustor: Apache Hadoop 1.2.1.SSH configuration

linuxProgramming langauge that can be used to code Maper, Reducer and driver classes: Python/Java.IDE used for programming: Ecllipse4.2 Configure PCsOur implementation of the project started with the configuration of the PCs allotted to us by our department. The configuration involved

installing Ubuntu 12.04 LTS into each PC and updating the operating system on each PC.4.3 Configuring HADOOPPre-Requisites of HADOOP1) JVM2) SSH

Since HADOOP's libraries are pre-dominantly written in Java, for obvious reasons, it will need JVM for HADOOP to run. Usually JVM is not a part of Ubuntu12.04 so we have to install it on our own. We can do this by doing the following commands:$sudo apt-get install sun-java7-jdk2.) Now, cluster is all about assigning jobs to the slave (NameNodes) and execute the jobs remotely on the slave nodes. This is accomplished using SSH. SSH is a remote logging client. But, Ubuntu 12.04 doesn't comes with SSH server pre-loaded in it. And we need it for Slave nodes to listen to the incoming requests. This can be done using the following command:$sudo apt-get install open-ssh-serverAdding User and Group in the computer:We will use a dedicated Hadoop user account for running Hadoop. While that's not required it is recommended because it helps to separate the Hadoop installation from other software applications and user accounts running on the same machine. Following are the commands to do this:$ sudo addgroup hadoop$ sudo adduser -ingroup hadoop hduserConfiguring SSH:Hadoop requires SSH access to manage its nodes, i.e. remote machines plus your local machine if you want to use Hadoop on it (which is what we want to do in this short tutorial). For our single node setup of Hadoop, we therefore need to configure SSH access to localhost for the hduser user we created.user@ubuntu:~$ su - hduserhduser@ubuntu:~$ ssh-keygen -t rsa -P ""Generating public/private rsa key

pair.Enter file in which to save the key (/home/hduser/.ssh/id_rsa):Created directory '/home/hduser/.ssh'.Your identification has been saved in /home/hduser/.ssh/id_rsa.Your public key has been saved in /home/hduser/.ssh/id_rsa.pub.The key fingerprint is: 9b:82:ea:58:b4:e0:35:d7:ff:19:66:a ef:ae:0e:d2hduser@ubuntu The key's randomart image is: [...snipp...]$ cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keysAfter the above steps the SSH is to be enabled using the generated key:The final step is to check whether SSH is working properly or not. This is done by connecting the hduser with the local machine. And to accomplish this, the command is as follows:hduser@ubuntu:~$ ssh localhostIf using the above command, if the terminal prompts for hduser's password, it indicates that the hduser was not successfully added to the "authorized_users" list. And one needs to take necessary steps to make it right, since SSH access without password is crucial for a master node to communicate and assign tasks to the slaves via

SSH. Thus, a Master has to be an authorized user of slaves and slaves should contain the public key generated by the Master node as "authorized_keys We downloaded the HADOOP package from the Apache Download Mirrors [http://www.apache.org/dyn/closer.cgi/hadoop/core] and extracted the content of the package to any desired location. For our project we used the "/usr/local/hadoop/" directory.$ cd /usr/local$ sudo tar xzf hadoop-1.0.3.tar.gz$ sudo mv hadoop-1.0.3 hadoop$ sudo chown -R hduser:hadoop hadoopUpdating $HOME/.bashrc file:The following lines were added to $HOME/bashrc file of the hduser account:# Set Hadoop-related environment variablesexport HADOOP_HOME=/usr/local/hadoop# Set JAVA_HOME (we will also configure JAVA_HOME directly for Hadoop later on)export JAVA_HOME=/usr/lib/jvm/java-7-sun# Some convenient aliases and functions for running Hadoop-related commandsunalias fs &> /dev/nullalias fs="hadoop fs"unalias hls &> /dev/nullalias hls="fs -ls"lzohead () {hadoop fs -cat $1 | lzop -dc | head

less}# Add Hadoop bin/ directory to PATHexport PATH=$PATH:$HADOOP_HOME/binPerforming the above steps will install HADOOP in our system and we are now

ready to configure it according to our needs.Our configuration for the time-being is more of a pseudo-distributed configuration, since we are using only one node. We can also derive the fully-distributed paradigm using the pseudo-distributed systems.CONFIGURATIONFor configuration of single-node cluster(or any other type of cluster, for that matters), following files are changed in the /usr/local/hadoop/conf/ direcory:The following table shows the different XML configuration files and their uses of how they are used to configure a HADOOP cluster:Table 1: XML configuration files and their descriptionsFILENAMEDESCRIPTIONHadoop-env.shEnvironment-specific settings go here. If a current JDK isn't in the system path you'll want to come here to configure your JAVA_HOME.You can also specify JVM options for various Hadoop components here.

Customizing directory locations such as the log directory and the locations of the master and slave files is also performed here, although by default you shouldn't have to do any of what was just described in aCDH setup.core-site.xmlContains system-level Hadoop configuration items, such as the HDFS URL, the Hadoop temporary directory and script locations for rackaware Hadoop clusters. Settings in this file override the settings in coredefault.xml.hdfs-site.xmlContains HDFS settings such as the default file replication count, the block size, and whether permissions are enforced.mapred-site.xmlHDFS settings such as the default number of reduce tasks, default min/max task memory sizes, and speculative execution are all set here.Hadoop-env.sh:Environment-specific settings go here. If a current JDK isn't in the system path you'll want to come here to configure your JAVA_HOME.You can also specify JVM options for various Hadoop components here. Customizing directory locations such as the log directory and the locations of the master and slave files is also

performed here, although by default you shouldn't have to do any of what was just described in a CDH.The file contains the default Java Home directory as follows:Fig. 3: hadoop-env.sh conf/*-site.xml:Here we first create the hadoop.tmp.dir which is used as the default directory for the

HDFS and Cluster configuration. We can assign any folder as hadoop.tmp.dir.$ sudo mkdir -p /app/hadoop/tmp$ sudo chown hduser:hadoop /app/hadoop tmpIn the file /conf/core-site.xml, add the following code snippet between the . . . .. tags:The following changes are

made in the core-site.xml file:Fig. 4: core-site.xmlIn the file conf/mapred-site.xml file:Fig. 5: mapred-site.xmlIn the file conf/hdfs-site.xml file Fig. 6: hdfs-site.xmlFORMATTING THE File System via the NameNode":The first step to starting up Hadoop installation is formatting the Hadoop filesystem which is implemented on top of the local filesystem of your "cluster". We need to do this for the first time only, while configuring the cluster.To format the partition, just type the following command in BASH:hduser@ubuntu:~$ /usr/local/hadoop/bin/hadoop namenode -formatThe output for the above command will look like as follows:hduser@ubuntu:/usr/local/hadoop$ bin/hadoop namenode -format10/05/08 16:59:56 INFO namenode.NameNode

STARTUP_MSG:/**STARTUP_MSG: Starting NameNodeSTARTUP_MSG: host = ubuntu/127.0.1.1STARTUP MSG: args = [-format]STARTUP_MSG: version = 0.20.2STARTUP_MSG: build = https://svn.apache.org/repos/asf/hadoop/common/branches/branch-0.20 -r compiled by 'chrisdo' on Fri Feb 19 08:07:34 UTC

FSNamesystem: fsOwner=hduser,hadoop10/05/08 16:59:56 INFO namenode.FSNamesystem: supergroup=supergroup10/05/08 16:59:56 INFO namenode.FSNamesystem

isPermissionEnabled=true10/05/08 16:59:56 INFO common.Storage: Image file of size 96 saved in 0 seconds.10/05/08 16:59:57 INFO
Storage directory .../hadoop-hduser/dfs/name hasbeen successfully formatted.10/05/08 16:59:57 INFO namenode.NameNode: SHUTDOWN_MSG

SHUTDOWN_MSG: Shutting down NameNode at ubuntu/127.0.1.1**********/hd user@ubuntu:/usr/local/hadoop Starting your single-node cluster:hduser@ubuntu:~$ /usr/local/hadoop/bin/start-all.shThis will start your Namenode, Datanaode, Jobtracker and

TaskTracker on our machine.A nifty tool for checking whether the expected Hadoop processes are running is jps(part of Sun's Java since v1.5.0).Fig

Output of jps command4.4 Running a sample MapReduce job (WordCount example)To run a MapReduce job, we first need to add some content to the HDFS for Mapper and reduce function to process something. Our example program is very simple. It's called the WordCount program. It counts the no. of times each word appears in a text file and generates an output files with each unique word with it's count separated by a tab.To copy local files to HDFS hduser@ubuntu:~$/usr/local/hadoop/bin/hadoop dfs -copyFromLocal $HOME/Downloads /User/DatasetThe above command will copy the contents from $HOME Downloads(Local Files) to /User/Dataset(HDFS directory). NOTE: To execute the command successfully we need to start all the services of HADOOP.hduser ubuntu:/usr/local/hadoop$ bin/hadoop jar hadoop-examples.jar wordcount /User/Dataset /User/outputRun MapReduce job:The above command will run the MapReduce job on the node taking the input as /User/Dataset as input, and /User/Output directory will store the output.Retrieve the Job result:hduser ubuntu:/usr/local/hadoop$ bin/hadoop dfs -cat /User/output/part-r-00000To inspect the file, you can copy it from HDFS to the local file system

Alternatively, you can use the commandThe above configuration process was for a single-node cluster. But to run a MapReduce Job on multiple nodes simultaneously, we have to add extra configurations: (Master node only) Following changes are made in $HADOOP_HOME/conf/masters and. /slaves files:Fig masters in $HADOOP_HOME/confFig. 9: slaves in $HADOOP_HOME/conf4.5 Output of wordcount program:Here we will display the output of simple word-count problem and how the map reduce task will display.Fig. 10(a): Output of simple mapreduce programFig. 10(b): Output of simple mapreduce program4.6 GUI of Apache HadoopHadoop comes with several interfaces like,http://localhost:50070/http://localhost:50070/ - NameNode daemonFig. 11: Namenode daemonhttp://localhost:50030/http://localhost:50030/ - JobTracker daemonFig. 12: JobTracker

daemonhttp://localhost:50060/http://localhost TaskTracker daemonFig. 13: TaskTracker daemon . UI of directories in hadoop filesystemFig. 14: Directories in Hadoop filesystem4.7 clusterHere we will compare the total execution time for completion of task. We will compare for the different size of datasets i.e 125,250,500 MB

You will see the difference between time taken by the text file to process.Here we are considering cluster as 5 nodes (1 master node ,

MB250 MB500 MBStand-alone64 sec95 sec173 secCluter(5 nodes)32 sec56 sec93 secLet's see the graph:Fig. 15: Graph4.8 OpenCV for pythonTo install opencv first make sure that everything in the system is updated and upgraded.ÿ1 sudoÿapt-get update2 sudoÿapt-get upgradeNow, you need to install many dependencies, such as support for reading and writing image files, drawing on the screen, some needed tools, other libraries, etc.1 sudoÿapt getÿinstallÿbuild-essential libgtk2.0-dev libjpeg-dev libtiff4-dev libjasper-dev libopenexr-dev cmake python-dev python-numpy python-tk libtbb-dev libeigen3-dev yasm libfaac-dev libopencore-amrnb-dev libopencore-amrwb-dev libtheora-dev libvorbis-dev libxvidcore-dev libx264-dev libqt4-dev libqt opengl-dev sphinx-common texlive-latex-extra libv4l-dev libdc1394-22-dev libavcodec-dev libavformat-dev libswscale-dev default-jdk ant libvtk5-qt devTime to get the OpenCV 2.4.9 source code:1 cdÿ~2 wgetÿhttp://sourceforge.net/projects/opencvlibrary/files/opencv-unix/2.4.9/opencv-2.4.9.ziph ttp sourceforge.net/projects/opencvlibrary/files/opencv-unix/2.4.9/opencv-2.4.9.zip3 unzip opencv-2.4.9.zip4 cdÿopencv-2.4.9Now we have to generate the

Makefile by using cmake. In here we can define which parts of OpenCV we want to compile. Since we want to use the viz module, Python, Java, TBB

OpenGL, Qt, work with videos, etc, here is where we need to set that. Just execute the following line at the terminal to create the appropriate Makefile. Note that there are two dots at the end of the line, it is an argument for the cmake program and it means the parent directory (because we are inside the build directory, and we want to refer to the OpenCV directory, which is its parent).1 mkdirÿbuild2 cdÿbuild3 cmake -D WITH_TBB=ON -D

BUILD_NEW_PYTHON_SUPPORT=ON -D WITH_V4L=ON -D INSTALL_C_EXAMPLES=ON -D INSTALL_PYTHON_EXAMPLES=ON -D BUILD_EXAMPLES=ON -D WITH_QT=ON -D WITH_OPENGL

ON -D WITH_VTK=ON ..To compile and install OpenCV 2.4.9:1 make2 sudoÿmakeÿinstallÿTo configure OpenCV. First, open the opencv.conf file with the following code:1 sudoÿgedit /etc/ld.so.conf.d/opencv.confAdd the following line at the end of the file(it may be an empty file, that is ok) and then save it:/usr/local/lib1 sudoÿldconfigNow you have to open another file:1 sudo gedit /etc/bash.bashrcAdd these two lines at the end of the file and save it:1 PKG_CONFIG_PATH=$PKG_CONFIG_PATH:/usr/local/lib/pkgconfig 2 export PKG_CONFIG_PATHClose the console and open a new one, restart the computer or logout and

then login again. OpenCV will not work correctly until you do this.4.9 Implementation for image processing TaskApache hadoop is basically for manipulating a text file. There are several input format available for the mapper and reducer class like text , typed

sequence file .In this task we need to take whole video as input but with Hadoop we cannot directly take video as input or even image as input.You can view video as collection of frames and this frame is sort of image, we can see image as binary file.Now if you divide video into frames then each frame will be of around 6 kb to 500 kb .it depends upon the quality of video.Hadoop is basically efficient in processing a large file rather collection of small file. Here we have collection of images as input and we can consider it as collection of small file.In Hadoop we cannot give this as input and it will not be efficient too, so we will covert collection of frames into on sequence file which will be one single large file. Now we can give sequence file as input as Hadoop support sequence file as input format.Fig. 16: Implementing Face Detection AlgorithmFig. 17(a): Output of

Face Detection AlgorithmFig. 17(b): Output of Face Detection AlgorithmNow we will see the steps need to be followed in order to complete our task Dividing video into frames:In our first step we need to divide the video into frames, this task can be done in several ways like ffmpeg it is tool to divide video into frames other way is writing a code in any language support image processing ,here I am using OpenCV which is Open source library of image processing and which is basically in c ++ but there is API available for python too.With the help of it we write code for dividing video into frame,and output will be bunch of images. 4.9.2 Convert into sequence file: Hadoop is efficient in processing a large file than collection of small file. Here frames are very small in size so we will convert the frames into sequence file for conversion of frames to sequence file we first need to convert into tar file with simple code in python .and using another code we will convert into sequence file. We have to emulate the stock "word-count of text-basedÿmap-reduce - but instead do a "colour-count" across all the pixels in the video.To achieve this, and allow scalability, by splitting up

the input video into frames, followed by each frame into a set of tiles. Then, we can combine all these tiles into aÿhttp://hadoop.apache.org/docs current/api/org/apache/hadoop/io/SequenceFile.htmlSequenceFileÿ- one of the handier container formats that Hadoop supports for processing lots of smaller files.There is a project that extends Hadoop for processing image data:ÿhttp://hipi.cs.virginia.edu/Hipi. This has developed some appealing concepts to support image data transfer and culling. We also desire more fine grained control over the image data - especially video data that offered by Hipi's ImageBundle file. Hipi is still Java based, and we would prefer to useÿpythonÿas our language of choice - mainly

for its conciseness, but also to exploit its support for numerical and computer vision package bindings (e.g. OpenCV).We can process the data following the paradigm ofÿhttp://hadoop.apache.org/docs/mapreduce/current/streaming.htmlHadoop Streaming. To achieve this, and enable effective invocation of OpenCV we use the "Dumbo" python module, effectively a wrapper around the job invocation to let us execute python code, but which also allows the python job to access the typed bytes that encode the image data to be processed.We need (with versions):Python - version 2.7.2Hadoop needs Java of course) - version 1.1.1FFmpeg - version 1.0.1ImageMagick - version 6.8.0Opencv - version ipython - version 0.13.1dumbo - version 0.21.35Fig. 18: Division of video into frames4.10 Identification of faces

Eigenfacesÿis the name given to a set ofÿhttp://en.wikipedia.org/wiki/Eigenvectoreigenvectorsÿwhen they are used in theÿhttp://en.wikipedia.org/wiki Computer_visioncomputer visionÿproblem of humanÿhttp://en.wikipedia.org/wiki/Facial_recognition_systemface recognition.The eigenvectors are derived from theÿhttp://en.wikipedia.org/wiki/Covariance_matrixcovariance matrixÿof theÿhttp://en.wikipedia.org/wiki/Probability_distributionprobability distributionÿover the high-http://en.wikipedia.org/wiki/Dimensiondimensionalÿhttp://en.wikipedia.org/wiki/Vector_spacevector spaceÿof face images The eigenfaces themselves form a basis set of all images used to construct the covariance matrix. This produces dimension reduction by allowing the smaller set of basis images to represent the original training images. Classification can be achieved by comparing how faces are represented by the basis set.The code given below will perform the task of detecting the faces from an image.Fig. 19: Input to Face Recognition AlgorithmFig. 20: Output of Face Recognition Algorithm4.11 Cropping of faces from the imagesFig. 21: Multiple input images for cropping faceWe are giving two images in input as shown in the figure. The code given below shows how the number of faces are detected from the input images and stored in the location provided. After executing the code the detected faces are cropped and stored in the folder given as the path for storing the cropped faces.#Python Opencv 2.4.2#PIL 1.1.7import cv #Opencvimport Image #Image from PILimport globimport osdef DetectFace(image, faceCascade, returnImage=False wThis function takes a grey scale cv image and finds # whaar_scale = 1.1 cv.HaarDetectObjects(wIf faces are found y, w, h), n) in faces: # Convert bounding box to two pt1 = (t> returnImage: PIL image to a greyscale pil_im.convert('L') 1)

wreturn cv_imdef cv2pil(cv_im): # Convert the

imgCrop(image, cropBox, boxScale=1): # Crop a PIL image with the provided box [y(upper), w(width), h(height)] # Calculate scale factors w upper, right, lower] box=[cropBox[0]-xDelta, cropBox[1]-yDelta, cropBox[0]+cropBox[2]+xDelta, cropBox[1]+cropBox[3]+yDelta]

imagePattern,boxScale=1): = cv.Load('haarcascade_frontalface_alt.xml') print 'No Images Found' im,faceCascade) w:t> rprint 'No faces found:', imgdef test(abc): frontalface_alt.xml') all jpegs in a folder. Note: the code uses glob which follows unix shell rules.# Use the boxScale to scale the cropping area. 1=opencv box,

2=2x the width and heightfaceCrop('testPics/*.jpg',boxScale=1)Fig. 22: Execution of codeFig. 23: Cropped faces as outputComparing input image with images in the databaseThe following image is given as input and compared with the images in our database. If the match is found it gives the name of the suspect as output.Fig. 24: Image of the suspect as inputFig. 25: DatabaseFig. 26: Suspect detected as outputFig. 27: Output

AlgorithmCHAPTER-5 CONCLUSION AND FUTURE WORK 5.1 Objectives completed in this semester Edited the previous python code which now detects faces from a given image i.e. frames and then stores the new image which marks the face with a rectangle border.The detected faces through the previous code are now cropped by executing a new code. The code detects the faces, crops them and stores them in a given folder.An image containing the face of a suspect is given as input and compared with the images in our database to find a match. To detect faces throughout the video and store that images at one place i.e. file or folder in the system.Created a multi-node Hadoop cluster and deployed the whole video processing task on it.Our aim to justify our title and complete our project definition has been successfully achieved.Future WorkThe future scope of this project is to develop a system where a fast and accurate data processing is required. Where the user has the facility to search the media file from the tons of storage of which the user does not know the name of.CHAPTER-6 Tom White, "Hadoop - The Definitive Guide", O'Reilly Publications, May 2012]3. [S. Abiteboul, I. Manolescu, P. Rigaux, M. Rousset and P.

Web Data Management", Cambridge University Press, 2012].4. http://www.hadoop.apache.org/www.hadoop.apache.org/ releases.html#Download5. http://www wiki.apache.org/hadoopwww.wiki.apache.org/hadoop6. http://www.michael-noll.com/tutorials/www.michael-noll.com/tutorials/7. http://www.3pillarglobal com/blog/how-configure-apache-hadoop-standalone-modewww.3pillarglobal.com/blog/how-configure-apache-hadoop-standalone-mode8. http://blog.pivotal.io data-science-pivotal/products/using-hadoop-mapreduce-for-distributed-video-transcodinghttp://blog.pivotal.io/data-science-pivotal/products/using hadoop-mapreduce-for-distributed-video-transcoding9. http://www.academia.edu/6264430/The_Research_of_Smart_Surveillance_System_http://www.academia edu/6264430/The_Research_of_Smart_Surveillance_System_Using_Hadoop_Based_On_Crani

ofacial_Identification10. http://createdigitalmotion.com
processing-tutorials-getting-started-with-video-processing-via-opencv/11.
http://in.mathworks.com/help/vision/examples/face-detection-and-tracking
using-camshift.html12.
http://in.mathworks.com/help/vision/examples/object-detection-in-a-cluttered-scene-using-point-feature-matching.html
Periodic Progress Report (PPR) DetailsPeriodic Progess Report : First PPRProject Video Processing Using Hadoop:Status : Reviewed

Progress you have made in the Project ?We have edited the python code so that it now detects faces from a given image i.e. frames and then stores the new image which marks the face with a rectangle border. So we get all the faces in images, highlighted with rectangular borders around each one of them.What challenge you have faced ?We had to go through numerous image processing websites to attain accuracy with respect to our task.What support you need ?Assistance is required for implementing python programs onto Hadoop cluster.Which literature you have referred ?Face Recognition with OpenCV - opencv documentation How to do face recognition with python and opencv stackoverflow.com Face detection in Python using a webcam Python by realpython.com Comment by Internal Guide :OK5/15/2015Periodic Progress Report (PPR) DetailsPeriodic Progess Report : Second PPRProject Video Processing Using Hadoop:Status : Reviewed (Freeze)What Progress you have made in the Project ?The detected faces are now cropped by executing a new code. The code detects the faces, crops them and stores them in a given folder.What challenge you have faced ?We had to go through various face cropping algorithms to decide upon the suitable match for our requirement.What support you need ?For now we not need any support. Which literature you have referred ?stackoverflow.com-detect-face-then-autocrop-pictures chairnerd.seatgeek.com-opencv-face-detection-for-croppingfacesComment by Internal Guide :Good5/15/2015Periodic Progress Report (PPR) DetailsPeriodic Progess Report : Third PPRProject Video Processing Using Hadoop:Status Reviewed (Freeze)What Progress you have made in the Project ?We have given an image containing the face of a suspect as input and compared with the images in our database to find a match. After the successful match a folder with the data of that suspect is provided to the user as output.We have detected faces throughout the video and stored that images at one place i.e. file or folder in the system.What challenge you have faced ?We had network problems. So it took a lot of time to do a task which actually could have completed earlier.What support you need ?We need help in creating a multinode Hadoop cluster and guidance in how to run python code on Hadoop.Which literature you have referred ?Face comparison algorithms.Comment by Internal Guide :OK5/15/2015Periodic Progress Report (PPR) DetailsPeriodic Progess Report : Forth PPRProject Video Processing Using Hadoop:Status Reviewed (Freeze)What Progress you have made in the Project ?We have successfully implemented our video processing

tasks on multi-node Hadoop cluster so that the output obtained now takes much lesser time as compared to those implemented normally. Hence, we have completed our project definition successfully.What challenge you have faced ?We had to generate mapper and reducer code in python which is not available worldwide, and so we had challenges in creating the same.What support you need ?We are almost towards the completion of our project, still we would like some help regarding the challenge we have faced and content related to that if it can be provided.Which literature you have referred ?1. Automatic Face Detection Using Color Based Segmentation-Yogesh Tayal, Ruchika Lamba, Subhransu PadheeDepartment of Electrical and Instrumentation Engineering Thapar University, Patiala-147004, Punjab, India 2. NAPPROACH FOR FAST AND PARALLEL VIDEO PROCESSING ON APACHE HADOOP CLUSTERS Hanlin Tan,Lidong Chen

College of Information System and Management, National University of Defense Technology, China 3.The research of smart surveillance system using hadoop based on craniofacial identification Venkata lakshmi.k , Venkateswaran.sComment by Internal Guide :GOODPage

ABSTRACT

Recent research area projects usually include heavy processing of data which is a cumbersome, tedious and time consuming process. Hadoop is an open-source software framework for storing and processing big data in a distributed fashion on large clusters of commodity hardware. Essentially, it accomplishes two tasks: massive data storage and faster processing.

Open-source software. Open source software differs from commercial software due to the broad and open network of developers that create and manage the programs. Traditionally, it's free to download, use and contribute to, though more and more commercial versions of Hadoop are becoming available.

Framework. In this case, it means everything you need to develop and run, your software applications is provided – programs, tool sets, connections, etc.

Distributed. Data is divided and stored across multiple computers, and computations can be run in parallel across multiple connected machines.

Massive storage. The Hadoop framework can store huge amounts of data by breaking the data into blocks and storing it on clusters of lower-cost commodity hardware.

Faster processing. Hadoop processes large amounts of data in parallel across clusters of tightly connected low-cost computers for quick results.

With the ability to economically store and process any kind of data (not just numerical or structured data), organizations of all sizes are taking cues from the corporate web giants that have used Hadoop to their advantage (Google, Yahoo, Etsy, eBay, Twitter, etc.).

Moving forward to next phase of this project is to detect faces out from the video, which can be of great help reducing a lot of time scanning a large video. OpenCV is a very useful library which can be integrated along with the python programming language. OpenCV again is an open source library that is developed in C++. There are many situations where an organization has to scan the video footages of its various departments or there retail stores, so that they can make out how many and who all visited the respective department or that retail store. So that company can be ready in the situations like thefts or some disasters. Hereby we are trying to create a system which can help us achieve the above mentioned goal. Thus amalgamation of Big Data and Python along with OpenCV can be of great help. We have taken a step towards the enlightenment of this domain by means of our final year project.

CHAPTER-1 INTRODUCTION

1.1 Big Data And Parallel Processing

The world we live in today is dealing with huge amount of data every day (Terabytes of data). According to norms, anything greater than 10GB is Big Data. And there is an alarming urge to process these huge amounts of data for our use. Research is being done all over the world to develop techniques to process this data efficiently.

Parallel processing is the ability to carry out multiple operations or tasks simultaneously. The term is used in the context of both human cognition, particularly in the ability of the brain to simultaneously process incoming stimuli, and in parallel computing by machines. Parallel processing is the computing paradigm in which the files to be processed are distributed to many processors and files are processed parallel at each processors. And then when each processor is done with its processing, it sends back the result to the source.

For our project, we are using the Apache® HADOOP and OpenCV.

1.2 Apache HADOOP

The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.

HADOOP is installed to configure clusters. HADOOP also implements the HADOOP FileSystem(HDFS) in the clusters(i.e., the master node and the slave node). This is a distributed filesystem distributed over different nodes of the cluster.

There are two components of the filesystem:

1.) Name Node

2.) Data Node.

The Name Node is the node in the filesystem which keeps track of the data stored on the Data Nodes. The Name Node consists of names of all the Data Nodes and the files in those Data Nodes.

The Data Nodes are the nodes which are submissive in nature and contains all the files and directories of the HDFS.

HADOOP also has an interface JobTracker, which is used in clusters. In a typical HADOOP cluster, there is 1 master node and few slave nodes. When a client sends a request to the Master Node, it starts the JobTracker service. And the task is accomplished using Map/Reduce paradigm. The Map/Reduce paradigm is a programming paradigm using which, the master node fragments the program and each fragment or more are sent to individual slave nodes. The slave nodes process the fragments on its own and sends back the result to the Master node. And the master node has the responsibility to merge those results and give the final result to the user. This is accomplished using the JobTracker service on the master node. Using this service, the jobs are assigned and scheduled to the slave nodes. And also result retrieval and collection is done by the JobTracker service.

The Map/Reduce paradigm is inspired by Google's Map/Reduce paper.

The following image depicts the functional layers of a typical HADOOP cluster:

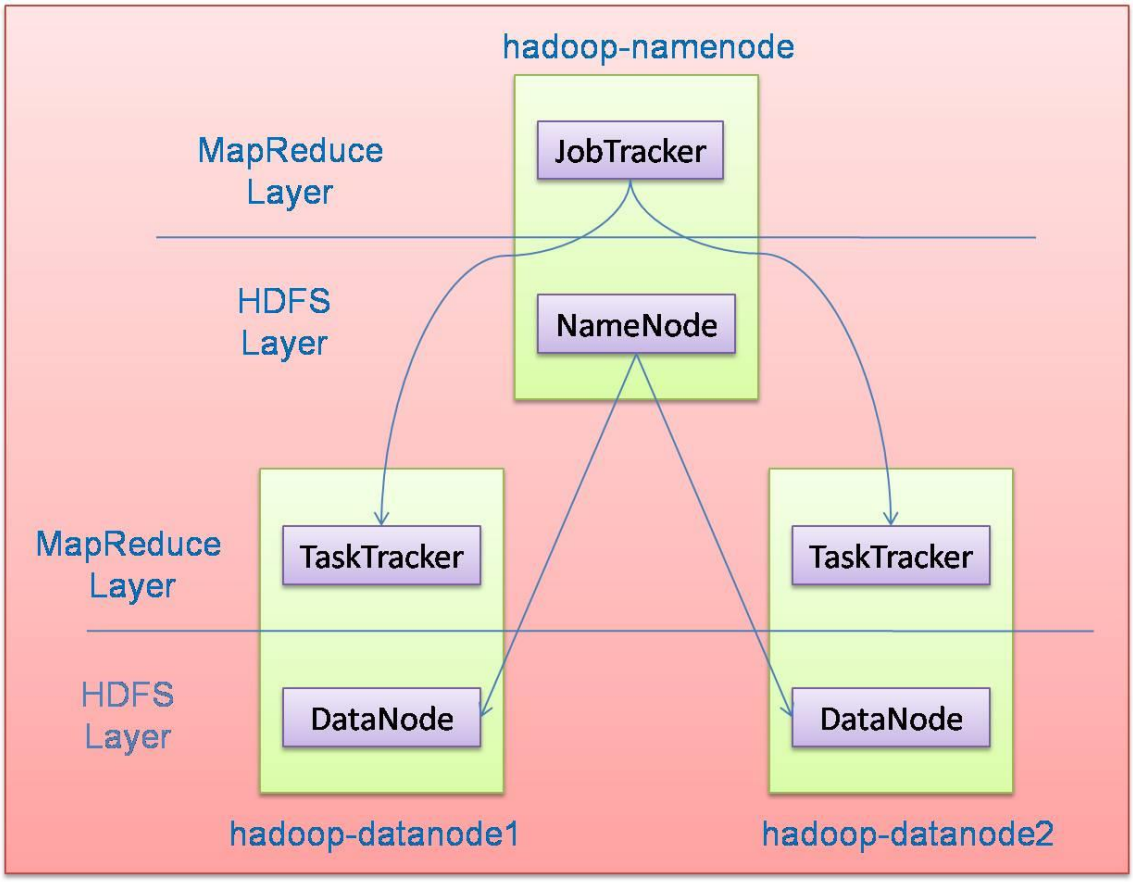

Fig. 1: Layers of Services in a HADOOP cluster

As shown in Fig. 1, there are three nodes in the cluster, one namenode(analogous to Master node) which is marked as "hadoop-namenode" in the figure. And two datanodes(analogous to Slave nodes). Each node runs two functional layers, namely the "MapReduce Layer", and "HDFS Layer".

Each of these layers have their functions to perform to run a MapReduce Job.

The MapReduce Layer is responsible for running the Job, i.e., distributing and scheduling the tasks to the datanodes, retrieving the individual results from the datanodes and combining the results of every datanode to give the final result of the Job to the Namenode.

A Job is the processing task requested by any client to the cluster. Whereas, a Task is assigned to the datanodes by dividing the job equally to the datanodes. So in simpler words, a *"Job"* is divided into *"Tasks"* and each datanode runs its own task locally, and all datanodes work simultaneously, thus decreasing the computing speed. All this is done by

the MapReduce Layer by exploiting the services of *JobTracker* on the namenode and *TaskTracker* on the datanode.

The HDFS Layer is responsible for keeping track of the files on the HADOOP FileSystem. The *DataNode* is where the actual data or files of the HDFS is mounted and *NameNode* is where all the data is tracked on the whole cluster. It contains information about which files are stored where in the cluster.

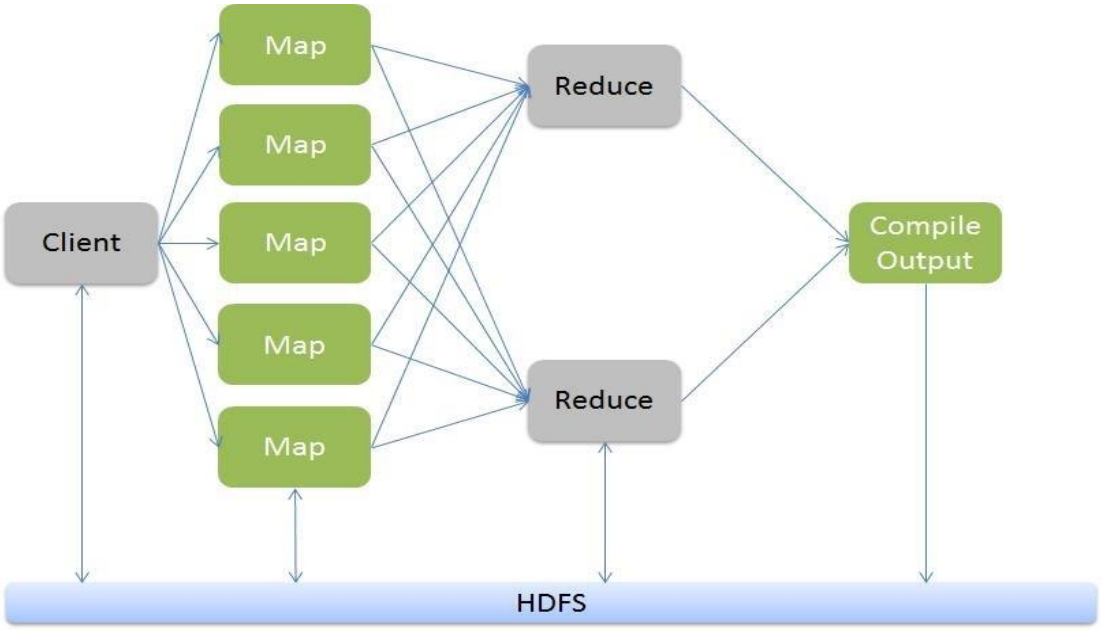

Fig. 2: MapReduce Job execution Flow

The Fig.2 shows the execution flow of a MapReduce Job. First, a client request is accepted. Then, the Job is divided into tasks using the Mapper function, which is called Mapping in HADOOP terminology. Then these tasks created by Mapper function is further processed by Reducer Function which performs the business logic. The corresponding outputs of each Reducer are compiled to form an output meant for the client.

1.3 OpenCV

OpenCV is an open source (see http://opensource.org) computer vision library available from http://SourceForge.net/projects/opencvlibrary. The library is written in C and C++ and runs under Linux, Windows and Mac OS X. OpenCV is aimed at providing the basic tools needed to solve computer vision problems. In some cases, high-level functionalities in the library will be sufficient to solve the more complex problems in computer vision.

Even when this is not the case, the basic components in the library are complete enough to enable creation of a complete solution of your own to almost any computer vision problem. In the latter case, there are several tried-and-true methods of using the library; all of them start with solving the problem using as many available library components as possible.

We may use SimpleCV or OpenCV depending on circumstances which will be more suitable. Since this is not much of a concern for the time being.

We need to use CV libraries as ultimately our input image will be compared to the frames of the videos. Thus for the comparision we will use CV libraries.

1.4 Theme of the project

Our project revolves around the concepts of Big Data and Parallel Processing paradigm. It gives us the essence of how brilliantly, easily and efficiently can the Big Data be processed using the parallel processing concepts.

1.5 Scope

- This system actually automates the procedure of scanning through the video eliminating human effort and thus be useful for surveillance.
- This system can be used in large enterprise and data centres where large data is to be manipulated.
- This framework can be implemented where the execution time of the service is crucial.

1.6 Background

With this world generating terabytes of data every day and the arising concern of real-time processing on this data, we were inspired to contribute in this field by taking up this project and strive for successful implementation of the same. As the project pertains to big data and parallel processing paradigm, we will use the latest techniques like HDFS(HADOOP Filesystem) to store the big data efficiently and MapReduce paradigm which is a parallel processing approach and applies very strong and efficient techniques to accomplish parallel processing tasks.

1.7 Objective

- The objective of our project is to give an image as an input and use it as a parameter for our large video repository and compare that image with the videos so that the video containing frame closest to the image is retrieved.
- Our project can come to a prospect use in surveillance systems, both on small and large scale surveillance systems. For example, if we have a photo of a suspect, our system can be used to compare the image of the suspect with the surveillance videos to flag if he could be seen in the surveillance videos. Rather watching videos manually, our system will process the videos to check it. And since these videos will be very large in size there is a good prospect of using HADOOP clusters to reduce the processing speed substantially. A similar project is already implemented in the city of Shanghai. The project implements surveillance cameras all over the city and implements real-time processing on the surveillance videos. The project processes these surveillance videos using HADOOP at the back-end.
- This model is to reduce the execution time of any service which requires a large configuration of hardware, dynamically by distributing the work among several systems.

1.8 Expected Outcome

- With the completion of the project, we expect a text file which will hold the data about the frame which is closest to the input image in the video repositories

1.9 Definition

- Surveillance cameras installed in enterprise facilities and public produce lots of video data everyday. Typically these surveillance video clips are stored as compressed video files corresponding to video for several hours.
- There is a need to process such huge video data sets to enable a quick summary of "interesting" events that happened during a specified time frame in a particular location.
- For instance one might be interested in anomalous trajectories of the objects within a scene.
- And thus in order to process such large scale data we need framework called hadoop.

Video processing uses hardware, software, and combinations of the two for editing the images and sound recorded in video files. Extensive algorithms in the processing software

and the peripheral equipment allow the user to perform editing functions using various filters. The desired effects can be produced by editing frame by frame or in larger batches.

A **MapReduce** program is composed of a **Map()** procedure that performs filtering and sorting (such as sorting students by first name into queues, one queue for each name) and a **Reduce()** procedure that performs a summary operation (such as counting the number of students in each queue, yielding name frequencies).

The "MapReduce System" (also called "infrastructure" or "framework") orchestrates by marshalling the distributed servers, running the various tasks in parallel, managing all communications and data transfers between the various parts of the system, and providing for redundancy and fault tolerance.

The model is inspired by the map and reduce functions commonly used in functional programming although their purpose in the MapReduce framework is not the same as in their original forms. The key contributions of the MapReduce framework are not the actual map and reduce functions, but the scalability and fault-tolerance achieved for a variety of applications by optimizing the execution engine once. As such, a single-threaded implementation of MapReduce will usually not be faster than a traditional implementation. Only when the optimized distributed shuffle operation (which reduces network communication cost) and fault tolerance features of the MapReduce framework come into play, is the use of this model beneficial.

1.10 Literature review and Prior Art Search (PAS)

1) **Title of Invention: Video big data distributed decoding method based on Hadoop**
   Patent No. CN 103279521 A
   Summary of Invention:
   Massive video files will be uploaded directly to the Hadoop Distributed File System HDFS for storage;
   HDFS with fuse_dfs will mount to the local file system to use a unified approach to access the HDFS file; modify data in Hadoop MapReduce computing framework segmentation strategy to image segmentation border towel Jung as to solve the frame problem of division and cannot decode byte division lead; read the public information required to decode mount HDFS from the local file system, and then use MapReduce

computing framework and FFmpeg video decoding library to complete the massive distributed decoding;

As a result of the decoding of MapReduce Map input for subsequent intelligent video analysis.

2) **Title of Invention: Method for retrieving massive face images based on Hadoop**
Patent No. CN 103207889 A

Summary of Invention:

In response to these problems and shortcomings of the prior art, the present invention is to provide a massive Hadoop retrieval method based on facial image, solve massive facial image recognition problems of slow to realize Massive completed quickly retrieve face images in low-configurable hardware environment.

3) **Title of Invention: Hadoop-platform-based method for improving video transcoding efficiency**
Patent No. : CN 103297807 A

Summary of Invention:

The present invention solves the technical problem are: to construct a video adaptive techniques to improve the efficiency of a method to overcome high concurrency existing video transcoding, the video segment sync, video merge, network bandwidth consuming large traffic load imbalance problem. Video job configuration module handles video processing configuration information entered by the user, the video processing tasks will be packaged into a job object is sent to the job queue management module, and then the video job configuration module notifies the video segmentation module divides the original video. After a good video segmentation video segmentation module, the segmentation results to the video transmission module; [0011] Step 300: a good video transmission module to upload files to split Distributed File System HDFS. The video is divided by the total number of files and split the clip frames to estimate the approximate location of the file truncation, then traverse time stamp files, locate the exact position of the cut-off point.

4) **Title of Invention: Image Processing System and Method of Improving Human Face Recognition**

Patent No. : US2014348399 (A1)

Summary of Invention:

The image processing system includes a camera device, a face detection unit, an exposure value adjusting unit, an image analyzing unit, an image processing unit and an image outputting unit. The camera device captures an image including a human face. The face detection unit executes a face detection in a predetermined area of the image .When the face detection does not recognize the human face, the exposure value adjusting unit adjusts exposure value of the predetermined area to an expected value. When the exposure value of the predetermined area reaches the expected value, the image analyzing unit executes the face detection and analyzes an image information in a face area of the image. The image processing unit selects a model parameter and an image adjusting parameter corresponding to the model parameter according to the image information and processes the face area of the image .The image outputting unit outputs a processed image to the human face recognition system.

5) **Title of Invention: System and method for executing map-reduce tasks in a storage device**

Patent No. : US 8819335 B1

Summary of Invention:

Aspects of embodiments of the present disclosure are directed toward a system and method of providing enhanced data processing and analysis in a cluster of compute nodes executing Map -Reduce tasks in a HadoopTM framework. HadoopTM framework divides a data query (Map-Reduce job) into a large number of small fragments of work, each of which may be performed on one of a large number of compute nodes. The work may involve a map task and a reduce task which may be used to categorize and analyze large amounts of data in distributed systems. A HadoopTM cluster contains a master node and a plurality of slave nodes. The slave nodes include intelligent solid -state drives capable of executing Map-Reduce tasks. The use of intelligent solid-state drives reduces the need to exchange data with a CPUin a server.

6) **Title of Invention: ATM alarming system based on Adaboost face detection and method**

Patent No. : CN 103198567 A

Summary of Invention:

The main object of the present invention is to provide an alarm system based on ATM machines and methods

Adaboost face detection, which is to block suspicious figures for recording and prompts the other to remove the obstruction by Adaboost face detection technology, and have suspicious persons ATM machine number and the location sent by MMS way into the hands of ATM management staff by the ATM machine management personnel to determine whether to take further action.

7) **Title of Invention: Face recognition from video images**

Patent No. : EP 1072014 B1

Summary of Invention:

The invention relates to a process for recognizing objects in an image frame, and to an apparatus for recognizing objects in an image frame. The present invention is embodied in an apparatus, and related method, for detecting and recognizing an object in an image frame. The object detection process uses robust and computationally efficient techniques. The object identification and recognition process uses an image processing technique based on model graphs and bunch graphs that efficiently represent image features as jets. The system of the invention is particularly advantageous for recognizing a person over a wide variety of pose angles. The object is detected and a portion of the image frame associated with the object is bounded by a bounding box. The bound portion of the image frame is transformed using a wavelet transformation to generate a transformed image. Nodes associated with distinguishing features of the object defined by wavelet jets of a bunch graph generated from a plurality of representative object images are located on the transformed image. The object is identified based on a similarity between wavelet jets associated with an object image in a gallery of object images and wavelet jets at the nodes on the transformed image.

8) **Title of Invention: Enhanced face recognition in video**

Patent No. : US 20120314914 A1

Summary of Invention:

The computational resources needed to perform processes such as image recognition can be reduced by determining appropriate frames of image information to use for the processing. In some embodiments, infrared imaging can be used to determine when a person is looking substantially towards a device, such that an image frame captured at that time will likely be adequate for facial recognition. In other embodiments, sound triangulation or motion sensing can be used to assist in determining which captured image frames to discard and which to select for processing based on any of a number of factors indicative of a proper frame for processing.

9) **Title of Invention: Video retrieval system for human face content**

Patent No. : US20080080743

Summary of Invention:

none embodiment, the present disclosure contemplates a method for processing video data. The method comprises detecting human faces in a plurality of video frames in the video data, and, for at least one detected human face, identifying all frames in which this face is present irrespective of whether the detected human face is present in these â€œface-specificâ€• set of video frames in a substantially temporally continuous manner. The method also comprises enabling a user to view face-specific video segments in the video data based on the face -specific set of video frames identified.

In another embodiment, the present disclosure contemplates another method for processing video data. The method comprises detecting human faces in a plurality of video frames in the video data and indicating one or more unmatched human faces in the detected human faces based on a comparison of the detected human faces against a plurality of human face images stored in a database. The method further includes tracking of at least one unmatched human face across the video data by locating a face -specific set of video frames therefor irrespective of whether the unmatched human face is present in the face-specific set of video frames in a substantially temporally continuous manner.

10) **Title of Invention: Video surveillance system with object tracking and retrieval**
Patent No. : US 20110096149 A1

Summary of Invention:

A system for capturing and retrieving a collection of video image data captures video image data from a live scene with still cameras and PTZ cameras, and automatically detects an object of interest entering or moving in the live scene. The system automatically controls the PTZ camera to enable close -up real time video capture of the object of interest. The system automatically tracks the object of interest in the captured video image data and analyses features of the object of interest.

1.11 Tools used for development

- OS required : Ubuntu 14.04(linux system)
- Tool required to implement cluster: Apache Hadoop 1.2.1.
- SSH configuration onto linux
- Programming language that can be used to code Maper, Reducer and driver classes: Python/Java.

IDE used for programming: Eclipse

CHAPTER-2 Analysis, Design Methodology and Implementation Strategy

2.1 Observation Matrix

1. Observations:
   a) Increase in crimes
   b) Increase in population
   c) Loop holes in current security
   d) Need of cluster computing
   e) Increase in quality of data
   f) Need of computerizing the security

2. Scouted Challenges:
   a) High speed data access technique
   b) Time required to maintain security
   c) High cost of database
   d) Detecting face if face is covered
   e) Low light Images
   f) Cluster Connections with python
   g) Face Recognization
   h) Low quality of videos

3. Top 5 Problems:-
   a) Lack of Infrastructure
   b) Lack of time
   c) Lack of expertise in technicians
   d) High cost of traditional hardware
   e) Less use of technology

4. Final Problem:-
a) Time consuming traditional process

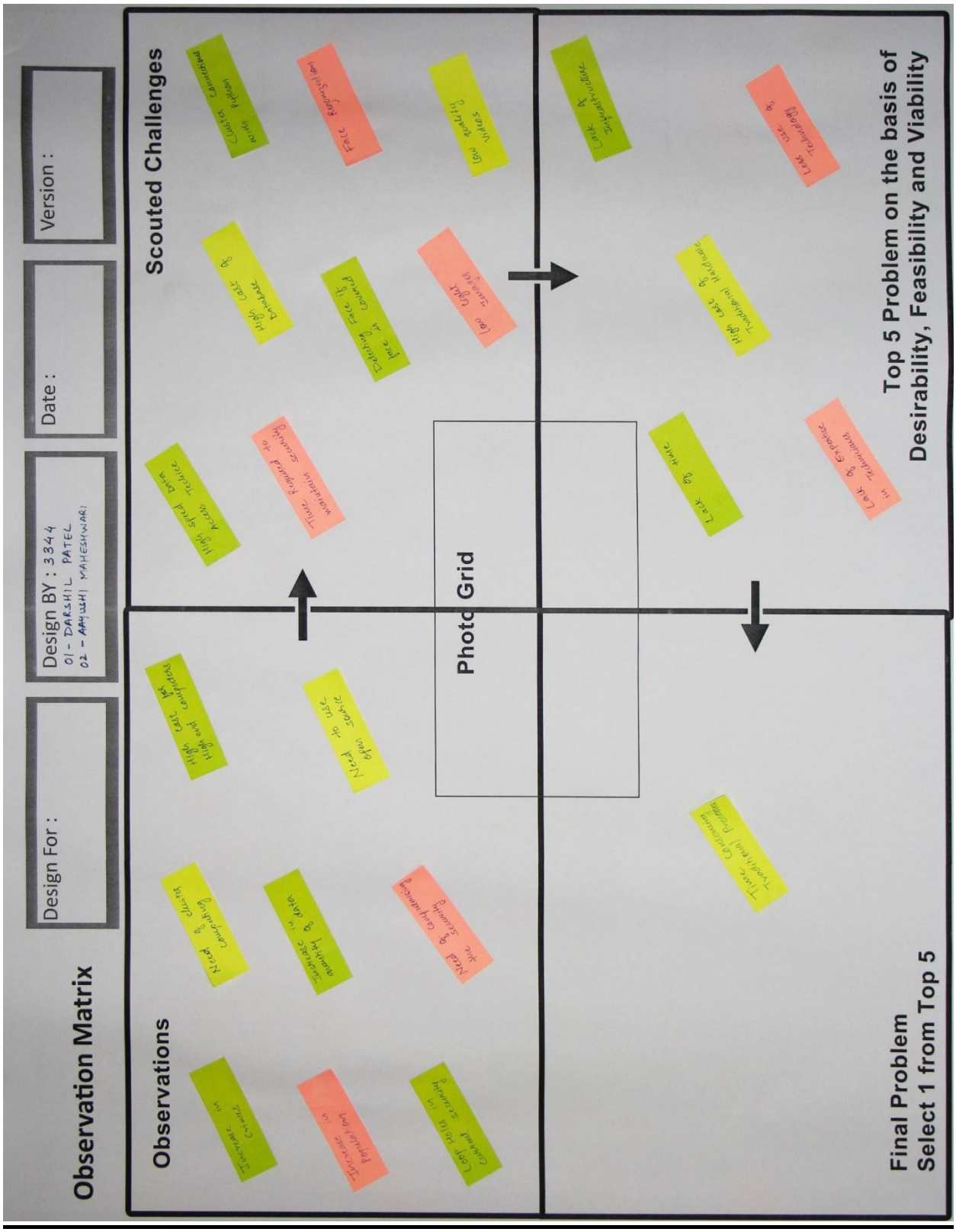

Fig. 3: Observation Matrix

2.2 Ideation Canvas

1. People:-

a) Security Agencies
b) Bank Consultants
c) Police Departments
d) Entertainment Industry
e) Military Services
f) News Reporters
g) Hotel Manager
h) ATM supervisors

2. Activities:-

a) Keep Check
b) Secret Services
c) Secure Province
d) Anti-theft
e) Catch Culprits
f) Provide Security
g) Identify the criminals
h) Surveillance

3. Key Resources:-

a) High-end computer systems: The most fundamental part of our project is high end computers which help us in making the distributed system.
b) Well directioned cameras: It is very important to align and attach the cameras such that very clear and proper video can be captured.
c) Secure Transmission lines: We have to be very careful about the security of the transmission that takes place between the webcams and the projected system.

4. Situation/Context/Location:-

a) Bank Theft

b) Border Trespass

c) Jewellery Store

d) Crime Scene Evidence

e) Court Rooms

f) ATM Theft

g) Road Accidents

h) Museums & Art galleries

5. Prons/Possible Solutions:-

a) Face Detection algorithms

b) HD Quality Webcams

c) High speed computing networks

d) Hadoop+OpenCV frameworks

e) Reliable Image Processing

f) Cluster Setup

g) Systems that can process large data

Fig. 4: Ideation Canvas

2.3 Product Development Canvas

1. Purpose:-
   a) To save time & resources
   b) Surveillance & Identify criminals

2. People:-
   a) Police and crime branch
   b) ATM Manager

3. Product Experience:-
   a) Feeling of reliability
   b) Ease of usage
   c) No Bottlenecks

4. Product Functions:-
   a) Division of video into frames
   b) Making & managing database
   c) Recognizing the culprit

5. Product Features:-
   a) Time saving procedures
   b) Multi-processing
   c) Division of large task
   d) Simple upload/download facility

6. Components:-
   a) High speed Clusters
   b) Configurable GUI
   c) HD Web cameras

7. Customer Revalidation:-

a) Simplest Implementaion

b) Very easy to use

c) Difficult to understand

d) Needs proper user interface

e) Best time utilization

f) Better multitasking possible

8. Reject,Redesign,Retain:-

a) Elongated user Interface

b) Intermediate steps

c) Manual Configuration

d) Better User Interface

e) Direct jumping to output

f) Semi-automated configuration

g) Small & Simple UI

h) One click processing

i) Fully automated configuration

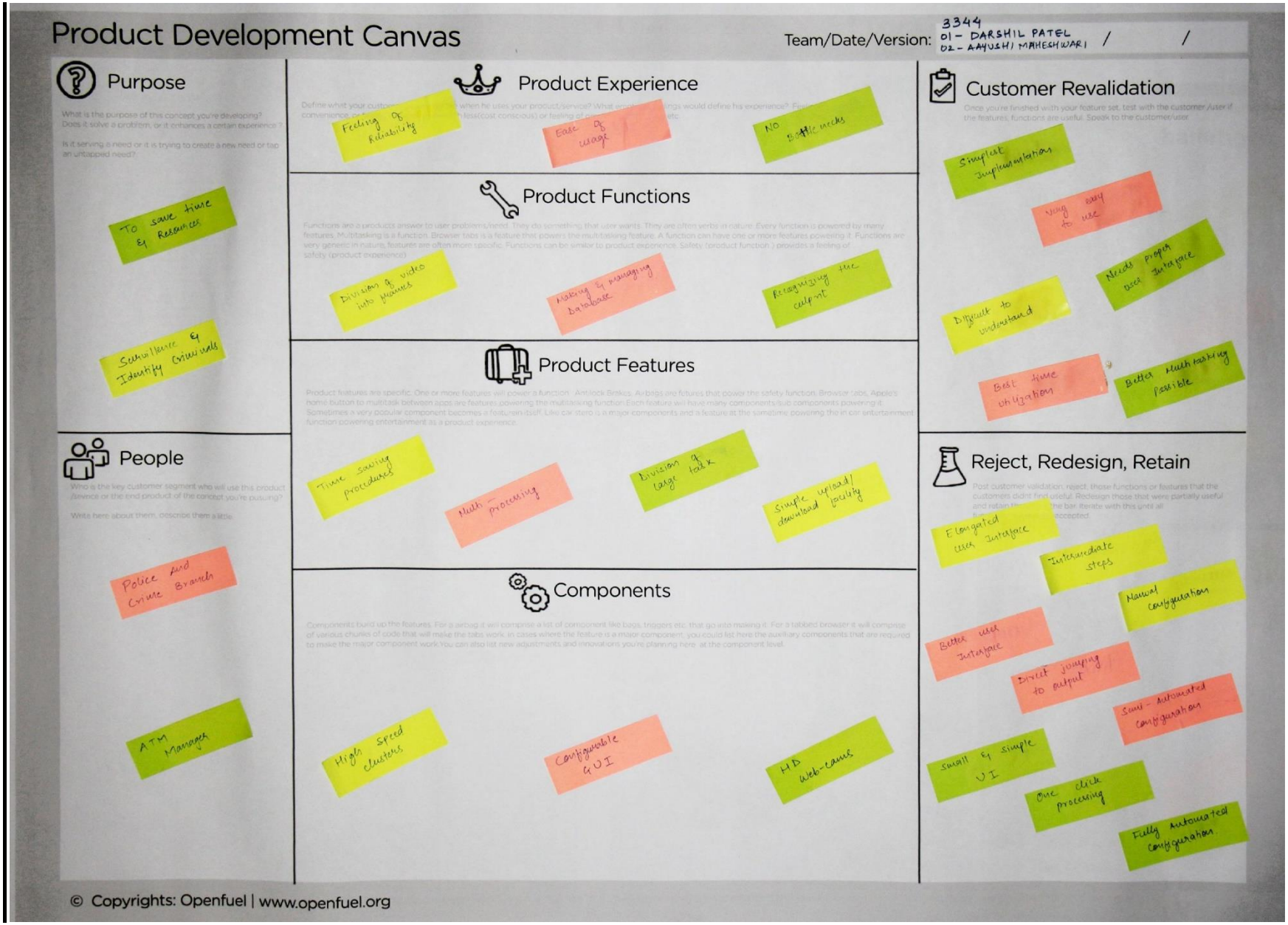

Fig. 5: Product Development Canvas

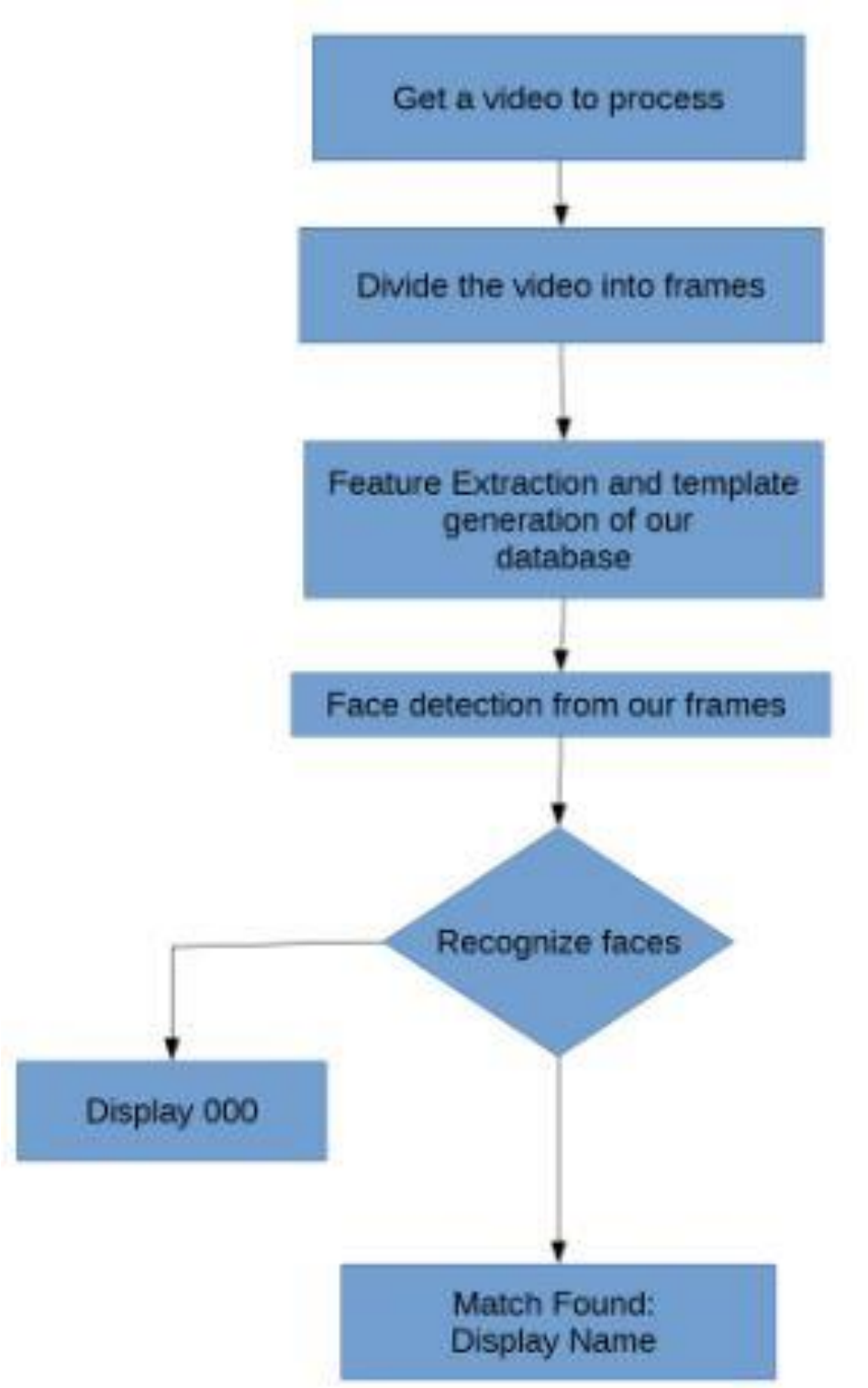

Fig. 6: Video Processing Flowchart

2.4 Functional and Non-Functional Requirements

Functional Requirements

- The system will take video file as an input and divide it into respective frames.
- The frame would be consisting of some face images, i.e. Image with faces in it, the system would divide the image into small parts that consists of only cropped faces.
- Finally user is left with only a folder in which he has faces of all the persons present in the video which makes task easier for the user.
- Now the user can check if the face that is detected is present in our database or not directly using the system.

Non-Functional Requirements

- The system must be light weight and use minimum resources to reduce operational costs.
- Interactive GUI and real time notifications to the users regarding important system events.
- Following are the tools used: Hadoop, OpenCV, and PIL.
- User should have flexibility of accessing the generated images.

CHAPTER-3 IMPLEMENTATION AND RESULTS

3.1 Configure PCs

Our implementation of the project started with the configuration of the PCs allotted to us by our department. The configuration involved installing Ubuntu 12.04 LTS into each PC and updating the operating system on each PC.

3.2 Configuring HADOOP

Pre-Requisites of HADOOP

1) JVM

2) SSH

1.) Since HADOOP's libraries are pre-dominantly written in Java, for obvious reasons, it will need JVM for HADOOP to run. Usually JVM is not a part of Ubuntu12.04 so we have to install it on our own. We can do this by doing the following commands:

```
$sudo apt-get install sun-java7-jdk
```

2.) Now, cluster is all about assigning jobs to the slave (NameNodes) and execute the jobs remotely on the slave nodes. This is accomplished using SSH. SSH is a remote logging client. But, Ubuntu 12.04 doesn't comes with SSH server pre-loaded in it. And we need it for Slave nodes to listen to the incoming requests. This can be done using the following command:

```
$sudo apt-get install open-ssh-server
```

Adding User and Group in the computer:

We will use a dedicated Hadoop user account for running Hadoop. While that's not required it is recommended because it helps to separate the Hadoop installation from other software applications and user accounts running on the same machine. Following are the commands to do this:

```
$ sudo addgroup hadoop

$ sudo adduser –ingroup hadoop hduser
```

Configuring SSH:

Hadoop requires SSH access to manage its nodes, i.e. remote machines plus your local machine if you want to use Hadoop on it (which is what we want to do in this short tutorial). For our single node setup of Hadoop, we therefore need to configure SSH access to localhost for the hduser user we created.

```
user@ubuntu:~$ su - hduser

hduser@ubuntu:~$ ssh-keygen -t rsa -P ""

Generating public/private rsa key pair.

Enter file in which to save the key (/home/hduser/.ssh/id_rsa):

Created directory '/home/hduser/.ssh'.

Your identification has been saved in /home/hduser/.ssh/id_rsa.

Your public key has been saved in /home/hduser/.ssh/id_rsa.pub.

The key fingerprint is: 9b:82:ea:58:b4:e0:35:d7:ff:19:66:a6:ef:ae:0e:d2

hduser@ubuntu The key's randomart image is: [...snipp...]
```

After the above steps, the SSH is to be enabled using the generated key:

```
$ cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys
```

The final step is to check whether SSH is working properly or not. This is done by connecting the hduser with the local machine. And to accomplish this, the command is as follows:

```
hduser@ubuntu:~$ ssh localhost
```

If using the above command, if the terminal prompts for hduser's password, it indicates that the hduser was not successfully added to the "authorized_users" list. And one needs to take necessary steps to make it right, since SSH access without password is crucial for a master node to communicate and assign tasks to the slaves via SSH. Thus, a Master has to be an authorized user of slaves and slaves should contain the public key generated by the Master node as "authorized_keys".

We downloaded the HADOOP package from the Apache Download Mirrors [http://www.apache.org/dyn/closer.cgi/hadoop/core] and extracted the content of the package to any desired location. For our project we used the "/usr/local/hadoop/" directory.

```
$ cd /usr/local

$ sudo tar xzf hadoop-1.0.3.tar.gz

$ sudo mv hadoop-1.0.3 hadoop

$ sudo chown -R hduser:hadoop hadoop
```

Updating $HOME/.bashrc file:

The following lines were added to $HOME/bashrc file of the hduser account:

```
# Set Hadoop-related environment variables

export HADOOP_HOME=/usr/local/hadoop

# Set JAVA_HOME (we will also configure JAVA_HOME directly
for Hadoop later on)

export JAVA_HOME=/usr/lib/jvm/java-7-sun

# Some convenient aliases and functions for running Hadoop-
related commands

unalias fs &> /dev/null

alias fs="hadoop fs"

unalias hls &> /dev/null

alias hls="fs -ls"

lzohead () {

hadoop fs -cat $1 | lzop -dc | head -1000 | less

}

# Add Hadoop bin/ directory to PATH

export PATH=$PATH:$HADOOP_HOME/bin
```

Performing the above steps will install HADOOP in our system and we are now ready to configure it according to our needs.

Our configuration for the time-being is more of a pseudo-distributed configuration, since we are using only one node. We can also derive the fully-distributed paradigm using the pseudo-distributed systems.

CONFIGURATION

For configuration of single-node cluster(or any other type of cluster, for that matters), following files are changed in the /usr/local/hadoop/conf/ direcory:

The following table shows the different XML configuration files and their uses of how they are used to configure a HADOOP cluster:

Table 1: XML configuration files and their descriptions

FILENAME	DESCRIPTION
Hadoop-env.sh	Environment-specific settings go here. If a current JDK isn't in the system path you'll want to come here to configure your JAVA_HOME.You can also specify JVM options for various Hadoop components here. Customizing directory locations such as the log directory and the locations of the master and slave files is also performed here, although by default you shouldn't have to do any of what was just described in aCDH setup.
core-site.xml	Contains system-level Hadoop configuration items, such as the HDFS URL, the Hadoop temporary directory, and script locations for rackaware Hadoop clusters. Settings in this file override the settings in coredefault.xml.
hdfs-site.xml	Contains HDFS settings such as the default file replication count, the block size, and whether permissions are enforced.
mapred-site.xml	HDFS settings such as the default number of reduce tasks, default min/max task memory sizes, and speculative execution are all set here.

Hadoop-env.sh:

Environment-specific settings go here. If a current JDK isn't in the system path you'll want to come here to configure your JAVA_HOME.You can also specify JVM options for various Hadoop components here. Customizing directory locations such as the log directory

and the locations of the master and slave files is also performed here, although by default you shouldn't have to do any of what was just described in a CDH.The file contains the default Java Home directory as follows:

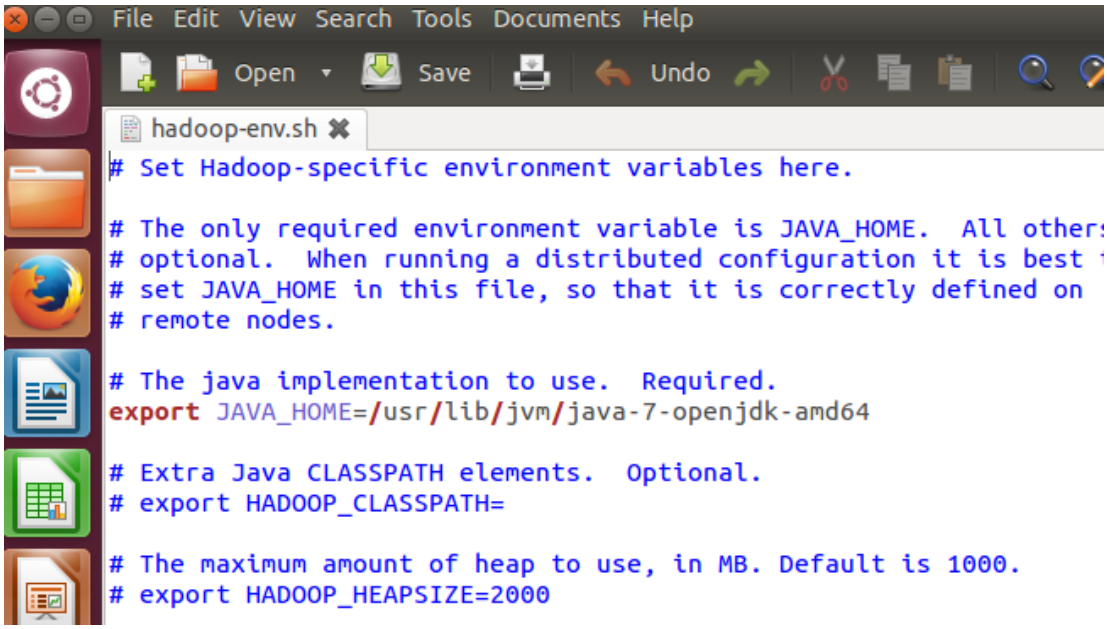

Fig. 7: hadoop-env.sh

conf/*-site.xml:

Here we first create the *hadoop.tmp.dir* which is used as the default directory for the HDFS and Cluster configuration. We can assign any folder as *hadoop.tmp.dir*.

```
$ sudo mkdir -p /app/hadoop/tmp

$ sudo chown hduser:hadoop /app/hadoop/tmp
```

In the file /conf/core-site.xml, add the following code snippet between the <configuration>. . . ..</configuration> tags:

The following changes are made in the core-site.xml file:

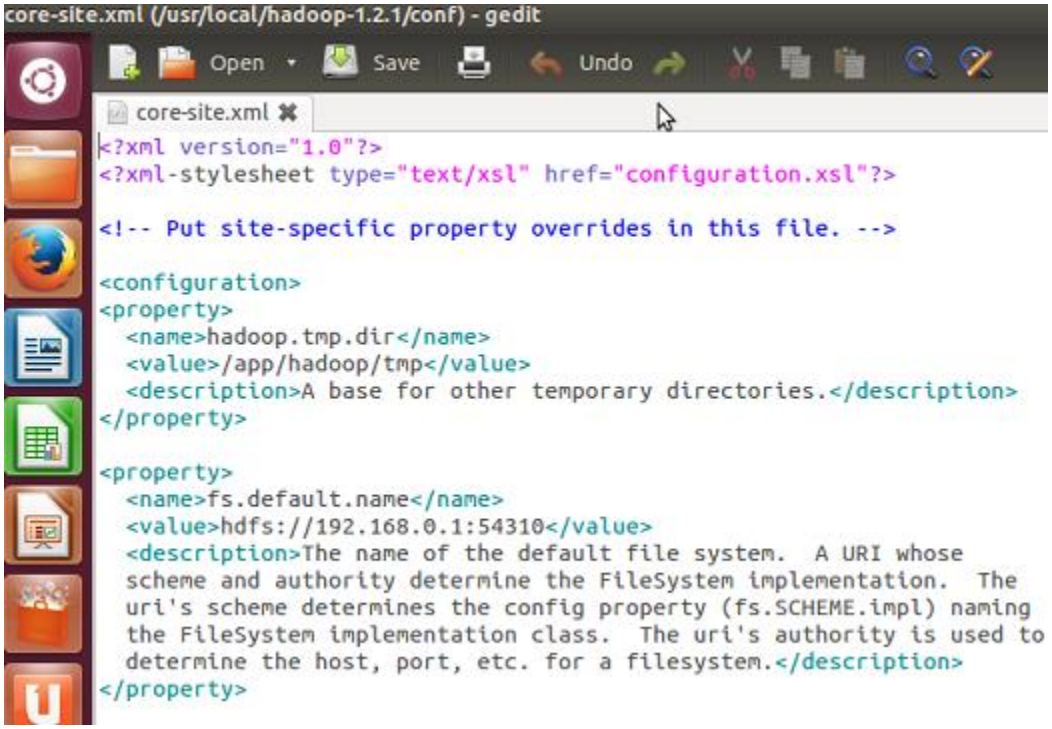

Fig. 8: core-site.xml

In the file *conf/mapred-site.xml* file*:*

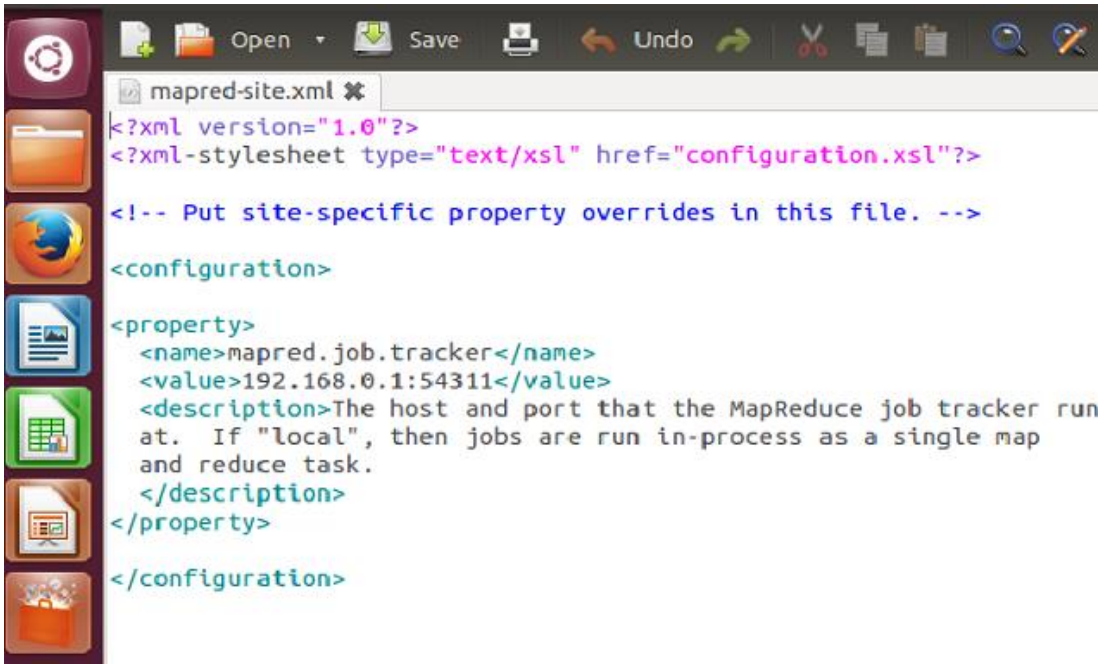

Fig. 9: mapred-site.xml

In the file *conf/hdfs-site.xml* file:

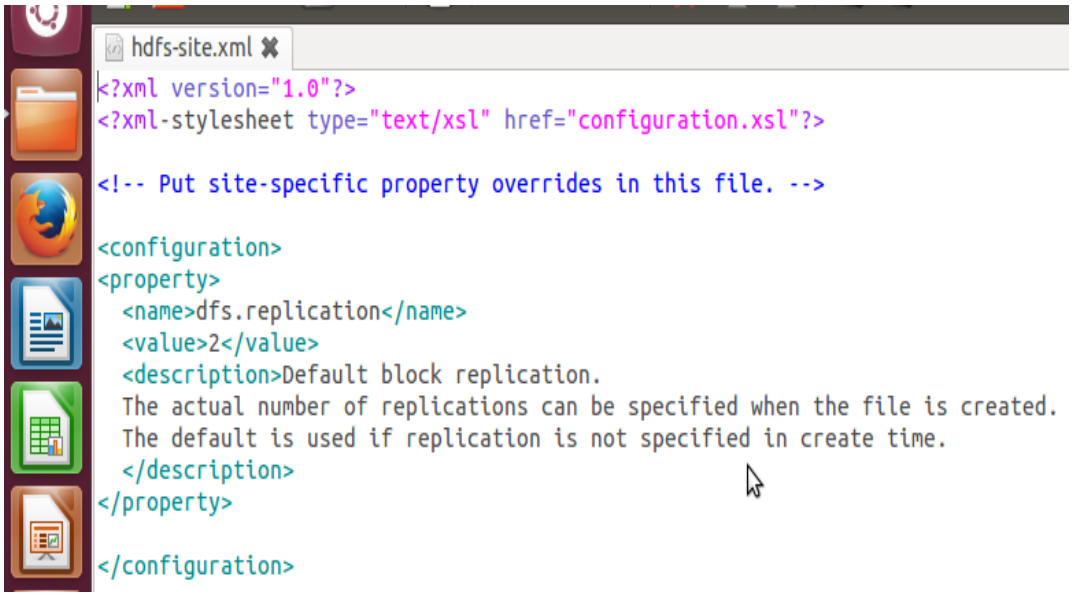

Fig. 10: hdfs-site.xml

FORMATTING THE File System via the NameNode":

The first step to starting up Hadoop installation is formatting the Hadoop filesystem which is implemented on top of the local filesystem of your "cluster". We need to do this for the first time only, while configuring the cluster.

To format the partition, just type the following command in BASH:

```
hduser@ubuntu:~$ /usr/local/hadoop/bin/hadoop namenode -format
```

The output for the above command will look like as follows:

```
hduser@ubuntu:/usr/local/hadoop$ bin/hadoop namenode -format

10/05/08 16:59:56 INFO namenode.NameNode: STARTUP_MSG:

/************************************************************

STARTUP_MSG: Starting NameNode

STARTUP_MSG: host = ubuntu/127.0.1.1

STARTUP_MSG: args = [-format]

STARTUP_MSG: version = 0.20.2

STARTUP_MSG: build =
https://svn.apache.org/repos/asf/hadoop/common/branches/branch-

0.20 -r 911707; compiled by 'chrisdo' on Fri Feb 19 08:07:34 UTC 2010

************************************************************/

10/05/08 16:59:56 INFO namenode.FSNamesystem: fsOwner=hduser,hadoop

10/05/08 16:59:56 INFO namenode.FSNamesystem: supergroup=supergroup

10/05/08 16:59:56 INFO namenode.FSNamesystem: isPermissionEnabled=true

10/05/08 16:59:56 INFO common.Storage: Image file of size 96 saved in 0 seconds.

10/05/08 16:59:57 INFO common.Storage: Storage directory .../hadoop-hduser/dfs/name
has

been successfully formatted.

10/05/08 16:59:57 INFO namenode.NameNode: SHUTDOWN_MSG: /

************************************************************

SHUTDOWN_MSG: Shutting down NameNode at ubuntu/127.0.1.1

************************************************************/

hduser@ubuntu:/usr/local/hadoop$
```

Starting your single-node cluster:

```
hduser@ubuntu:~$ /usr/local/hadoop/bin/start-all.sh
```

This will start your Namenode, Datanaode, Jobtracker and TaskTracker on our machine.

A nifty tool for checking whether the expected Hadoop processes are running is jps(part of Sun's Java since v1.5.0).

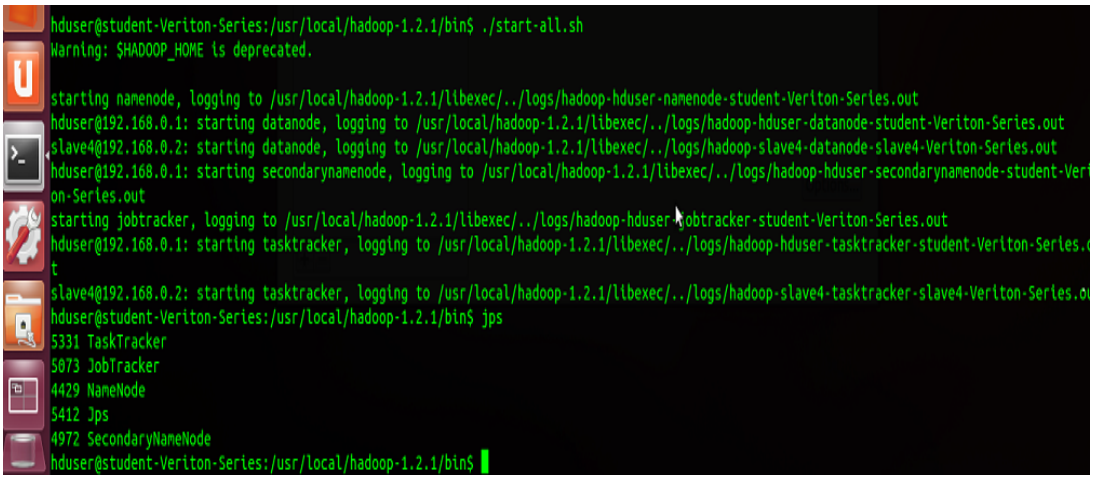

Fig. 11: Output of jps command

3.3 Running a sample MapReduce job (WordCount example)

To run a MapReduce job, we first need to add some content to the HDFS for Mapper and reduce function to process something. Our example program is very simple. It's called the WordCount program. It counts the no. of times each word appears in a text file and generates an output files with each unique word with it's count separated by a tab.

To copy local files to HDFS:

```
$hduser@ubuntu:~$/usr/local/hadoop/bin/hadoop dfs -copyFromLocal
$HOME/Downloads /User/Dataset
```

The above command will copy the contents from $HOME/Downloads(Local Files) to /User/Dataset(HDFS directory).

NOTE: To execute the command successfully we need to start all the services of HADOOP.

Run MapReduce job:

```
hduser@ubuntu:/usr/local/hadoop$ bin/hadoop jar hadoop-examples.jar
wordcount /User/Dataset /User/output
```

The above command will run the MapReduce job on the node taking the input as /User/Dataset as input, and /User/Output directory will store the output.

Retrieve the Job result:

To inspect the file, you can copy it from HDFS to the local file system. Alternatively, you can use the command

```
hduser@ubuntu:/usr/local/hadoop$ bin/hadoop dfs -cat /User/output/part-r-00000
```

The above configuration process was for a single-node cluster. But to run a MapReduce Job on multiple nodes simultaneously, we have to add extra configurations: (Master node only)

Following changes are made in $HADOOP_HOME/conf/masters and ./slaves files:

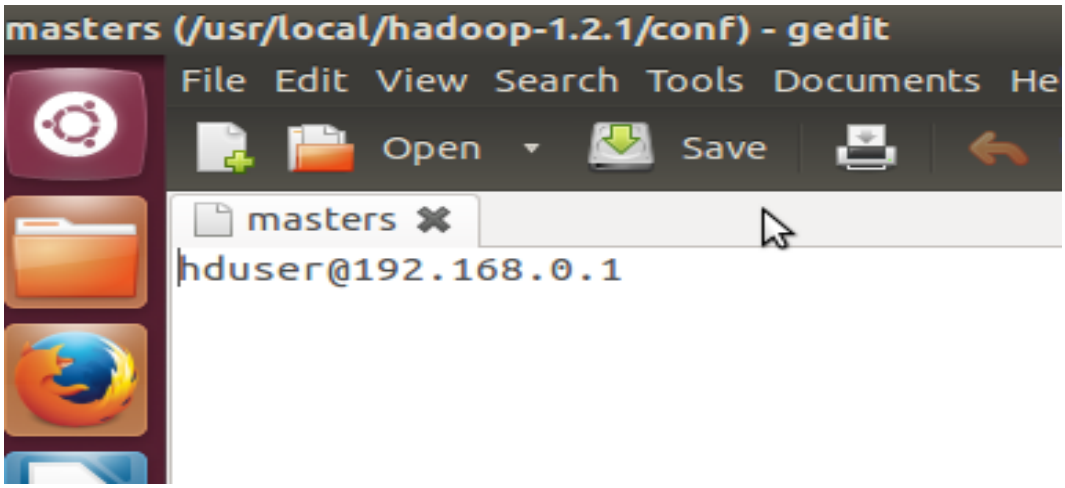

Fig. 12: masters in $HADOOP_HOME/conf

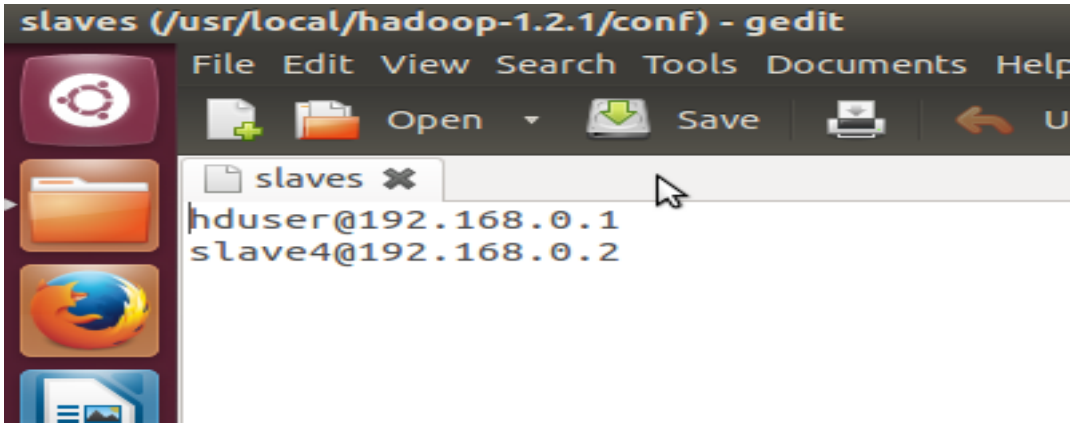

Fig. 13: slaves in $HADOOP_HOME/conf

3.4 Output of wordcount program:

Here we will display the output of simple word-count problem and how the map reduce task will display.

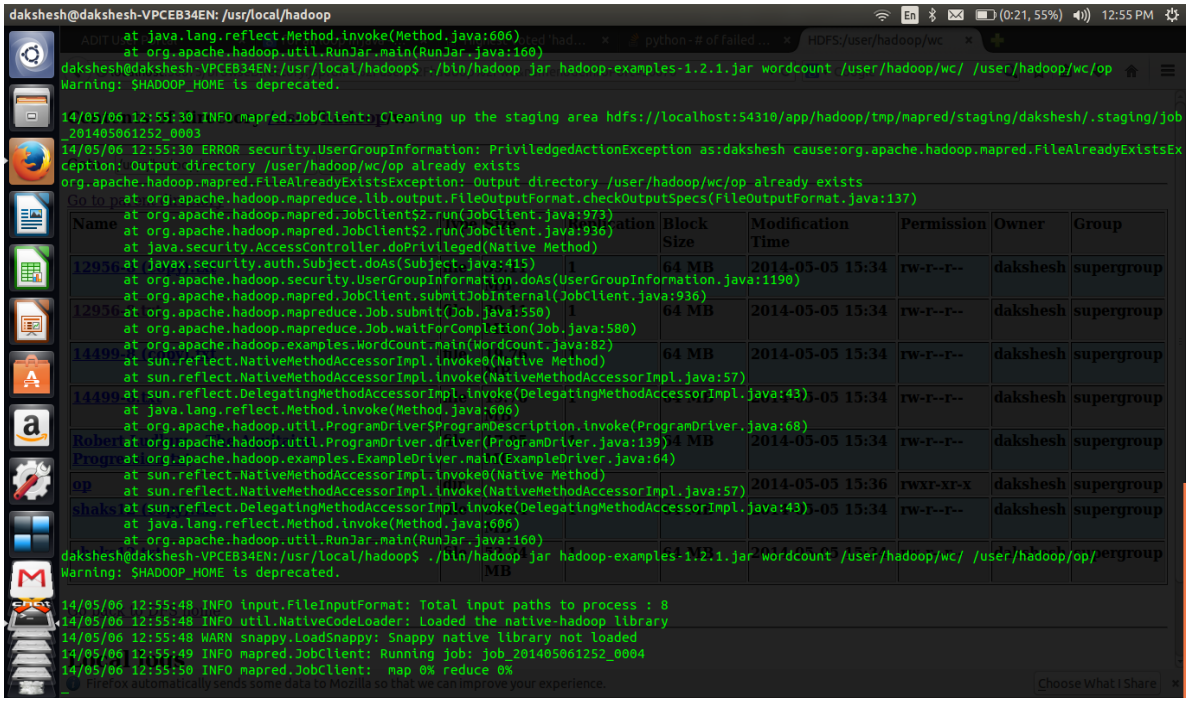

Fig. 14(a): Output of simple mapreduce program

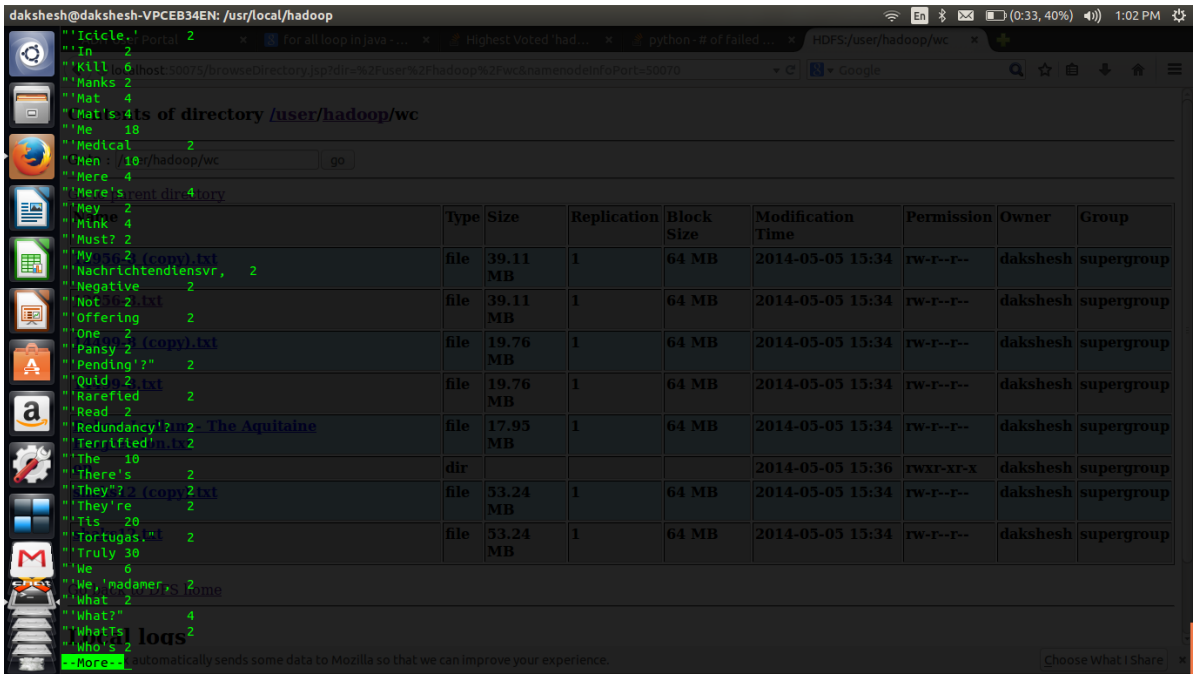

Fig. 14(b): Output of simple mapreduce program

3.5 GUI of Apache Hadoop

Hadoop comes with several interfaces like,

- http://localhost:50070/ - NameNode daemon

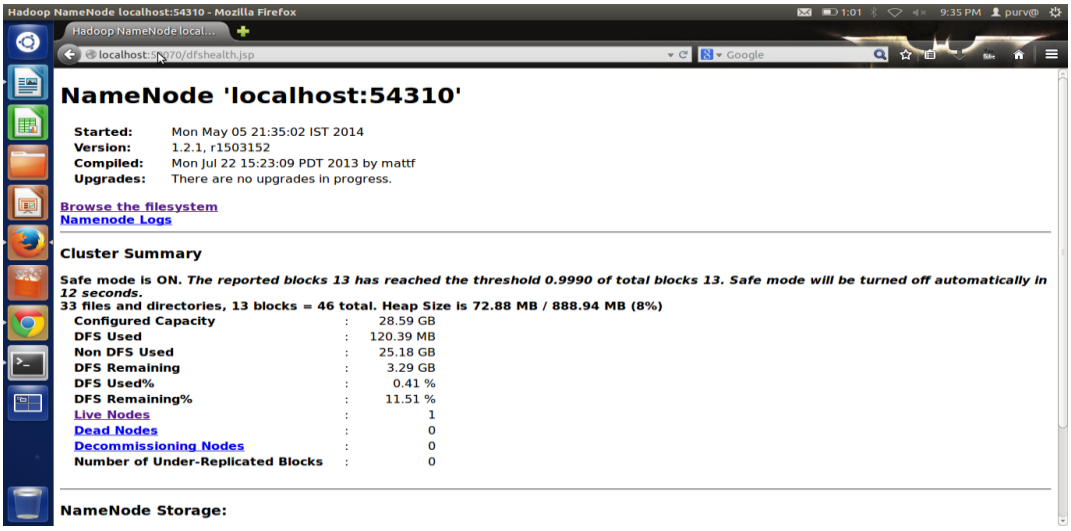

Fig. 15: Namenode daemon

- http://localhost:50030/ - JobTracker daemon

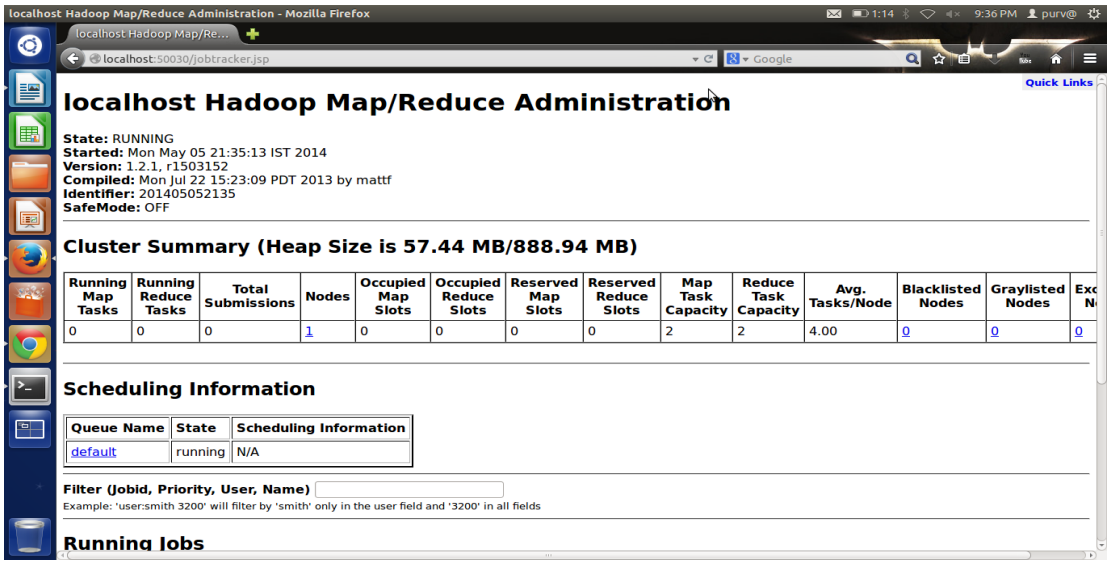

Fig. 16: JobTracker daemon

- http://localhost:50060/ - TaskTracker daemon

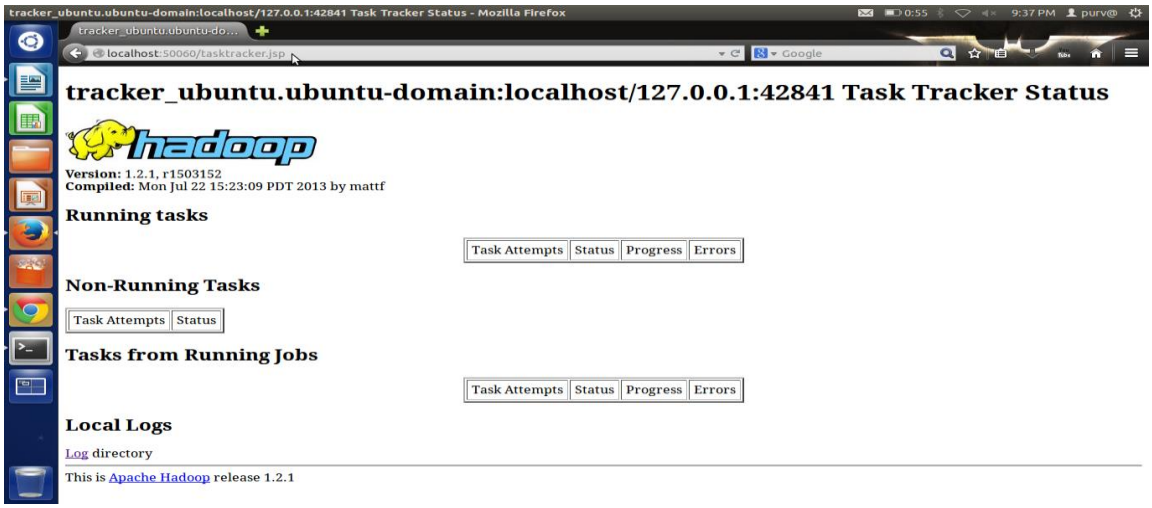

Fig. 17: TaskTracker daemon

.

- UI of directories in hadoop filesystem

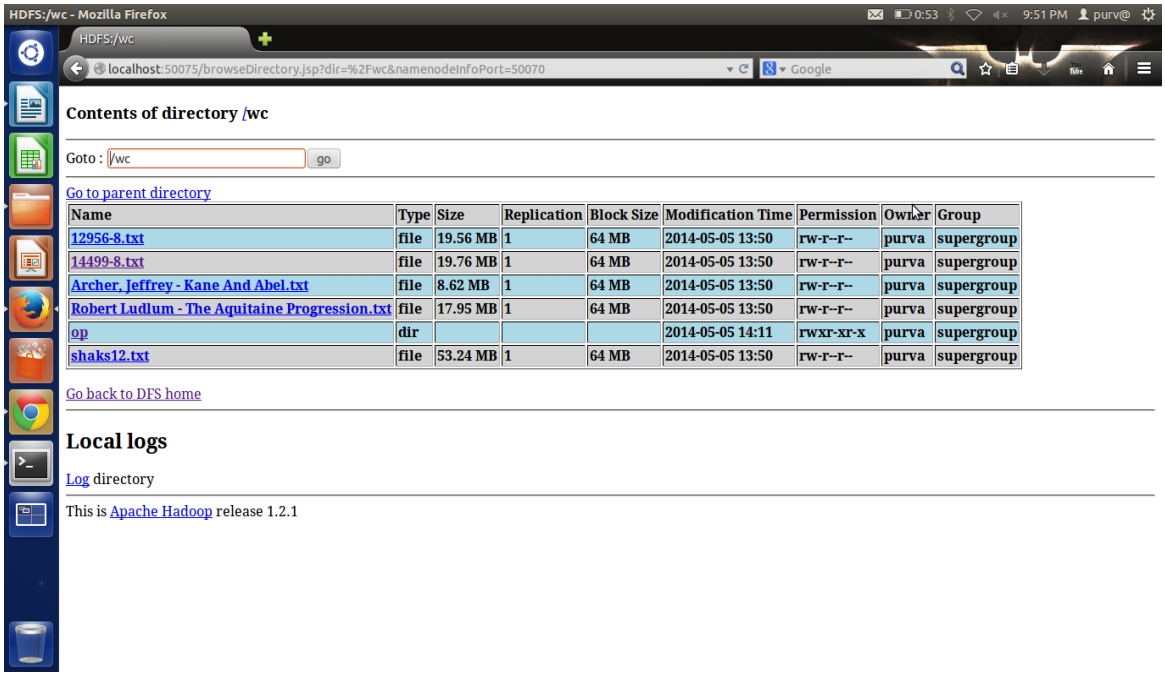

Fig. 18: Directories in Hadoop filesystem

3.6 Stand-alone vs cluster

Here we will compare the total execution time for completion of task. We will compare for the different size of datasets i.e 125,250,500 MB. You will see the difference between time taken by the text file to process.

Here we are considering cluster as 5 nodes (1 master node , 4 slave nodes).

	125 MB	**250 MB**	**500 MB**
Stand-alone	64 sec	95 sec	173 sec
Cluter(5 nodes)	32 sec	56 sec	93 sec

Let's see the graph:

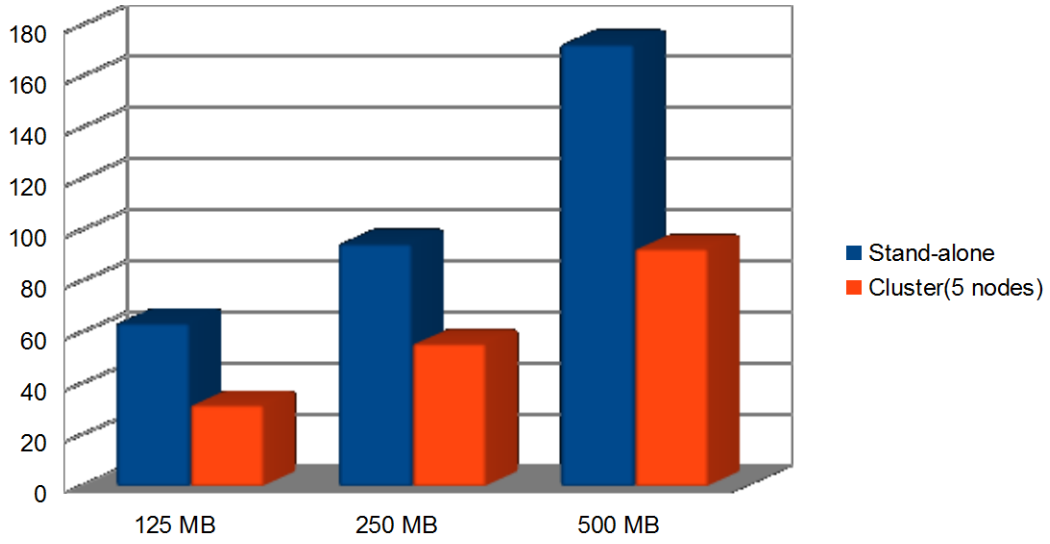

Fig. 19: Graph

3.7 OpenCV for python

To install opencv, first make sure that everything in the system is updated and upgraded.

1 sudo apt-get update

2 sudo apt-get upgrade

Now, you need to install many dependencies, such as support for reading and writing image files, drawing on the screen, some needed tools, other libraries, etc…

```
1 sudo apt-get install build-essential libgtk2.0-dev libjpeg-dev libtiff4-dev libjasper-dev libopenexr-dev cmake python-dev python-numpy python-tk libtbb-dev libeigen3-dev yasm libfaac-dev libopencore-amrnb-dev libopencore-amrwb-dev libtheora-dev libvorbis-dev libxvidcore-dev libx264-dev libqt4-dev libqt4-opengl-dev sphinx-common texlive-latex-extra libv4l-dev libdc1394-22-dev libavcodec-dev libavformat-dev libswscale-dev default-jdk ant libvtk5-qt4-dev
```

Time to get the OpenCV 2.4.9 source code:

```
1 cd ~

2 wget http://sourceforge.net/projects/opencvlibrary/files/opencv-unix/2.4.9/opencv-2.4.9.zip

3 unzip opencv-2.4.9.zip

4 cd opencv-2.4.9
```

Now we have to generate the Makefile by using cmake. In here we can define which parts of OpenCV we want to compile. Since we want to use the viz module, Python, Java, TBB, OpenGL, Qt, work with videos, etc, here is where we need to set that. Just execute the following line at the terminal to create the appropriate Makefile. Note that there are two dots at the end of the line, it is an argument for the cmake program and it means the parent directory (because we are inside the build directory, and we want to refer to the OpenCV directory, which is its parent).

```
1 mkdir build

2 cd build

3 cmake -D WITH_TBB=ON -D BUILD_NEW_PYTHON_SUPPORT=ON -D WITH_V4L=ON -D INSTALL_C_EXAMPLES=ON -D INSTALL_PYTHON_EXAMPLES=ON -D BUILD_EXAMPLES=ON -D WITH_QT=ON -D WITH_OPENGL=ON -D WITH_VTK=ON ..
```

To compile and install OpenCV 2.4.9:

```
1 make

2 sudo make install
```

To configure OpenCV. First, open the opencv.conf file with the following code:

```
1 sudo gedit /etc/ld.so.conf.d/opencv.conf
```

Add the following line at the end of the file(it may be an empty file, that is ok) and then save it:

/usr/local/lib

```
1 sudo ldconfig
```

Now you have to open another file:

```
1 sudo gedit /etc/bash.bashrc
```

Add these two lines at the end of the file and save it:

```
1 PKG_CONFIG_PATH=$PKG_CONFIG_PATH:/usr/local/lib/pkgconfig

2 export PKG_CONFIG_PATH
```

Close the console and open a new one, restart the computer or logout and then login again. OpenCV will not work correctly until you do this.

3.8 Implementation for image processing Task

Apache hadoop is basically for manipulating a text file. There are several input format available for the mapper and reducer class like text , typed bytes ,sequence file .

In this task we need to take whole video as input but with Hadoop we cannot directly take video as input or even image as input.

You can view video as collection of frames and this frame is sort of image, we can see image as binary file.

Now if you divide video into frames then each frame will be of around 6 kb to 500 kb .it depends upon the quality of video.

Hadoop is basically efficient in processing a large file rather than collection of small file. Here we have collection of images as input and we can consider it as collection of small file.

In Hadoop we cannot give this as input and it will not be efficient too, so we will covert collection of frames into on sequence file which will be one single large file. Now we can give sequence file as input as Hadoop support sequence file as input format.

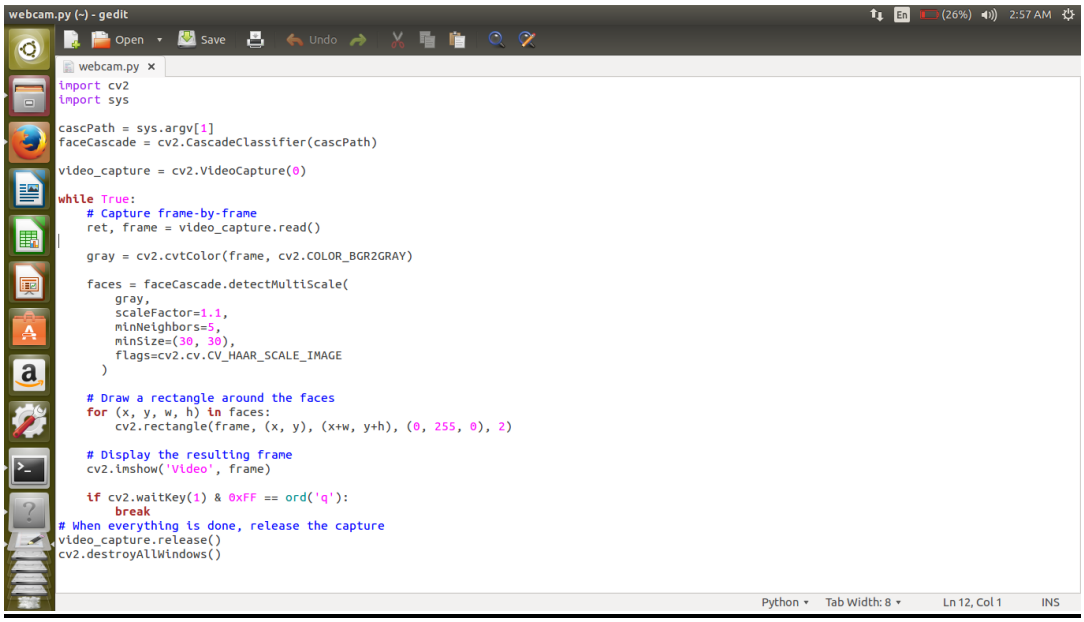

Fig. 20: Implementing Face Detection Algorithm

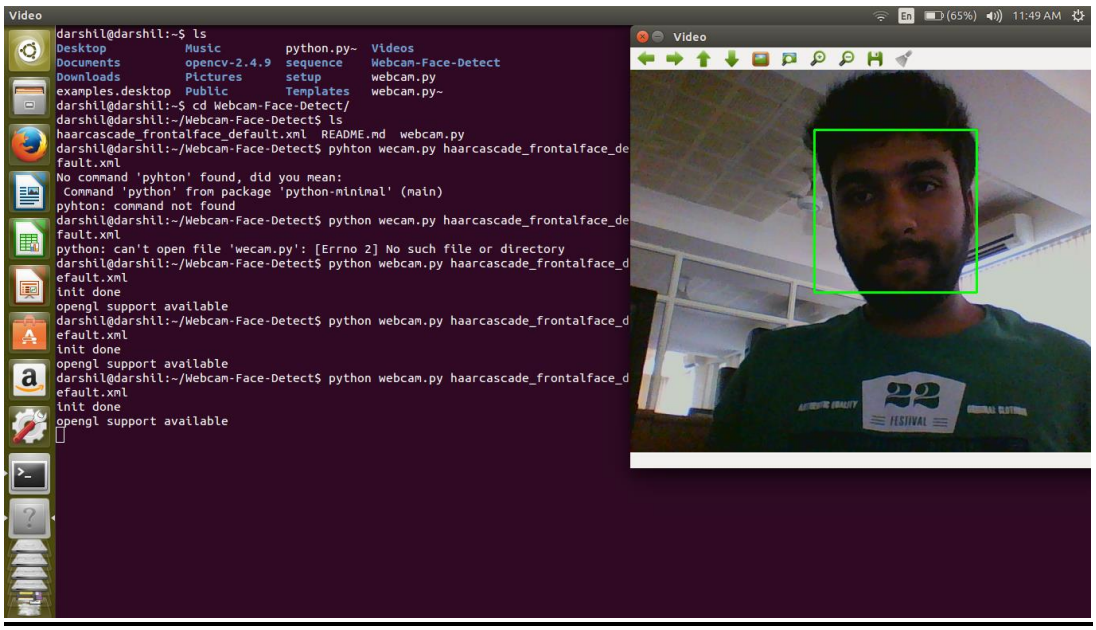

Fig. 21(a): Output of Face Detection Algorithm

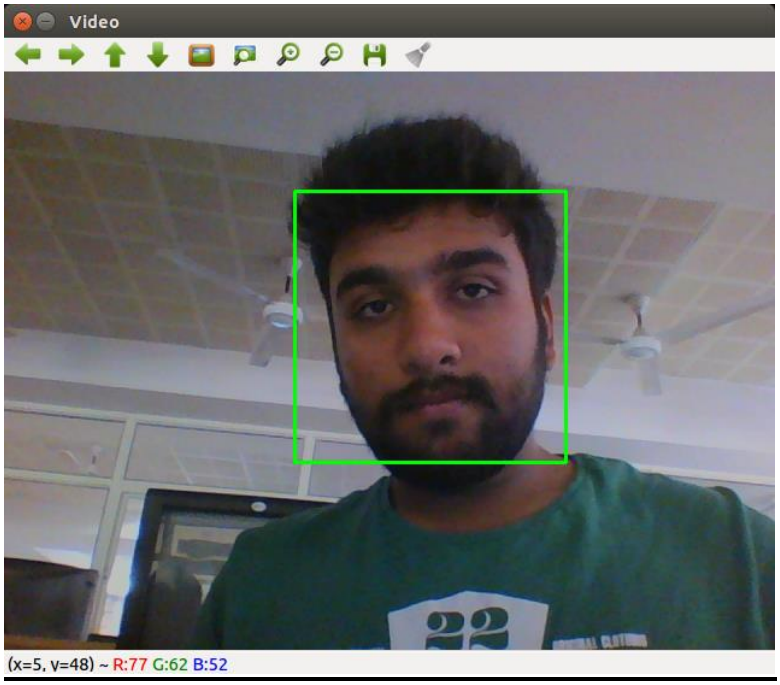

Fig. 21(b): Output of Face Detection Algorithm

Now we will see the steps need to be followed in order to complete our task:

3.8.1 Dividing video into frames:

In our first step we need to divide the video into frames, this task can be done in several ways like ffmpeg it is tool to divide video into frames other way is writing a code in any language support image processing ,here I am using OpenCV which is Open source library of image processing and which is basically in c ++ but there is API available for python too.

With the help of it we write code for dividing video into frame,and output will be bunch of images.

3.8.2 Convert into sequence file:

Hadoop is efficient in processing a large file than collection of small file. Here frames are very small in size so we will convert the frames into sequence file for conversion of frames to sequence file we first need to convert into tar file with simple code in python .and using another code we will convert into sequence file.

We have to emulate the stock "word-count" of *text-based* map-reduce - but instead do a "colour-count" across all the pixels in the video.

To achieve this, and allow scalability, by splitting up the input video into frames, followed by each frame into a set of tiles. Then, we can combine all these tiles into a SequenceFile - one of the handier container formats that Hadoop supports for processing lots of smaller files.

There is a project that extends Hadoop for processing image data: Hipi. This has developed some appealing concepts to support image data transfer and culling. We also desire more fine grained control over the image data - especially video data - beyond that offered by Hipi's ImageBundle file. Hipi is still Java based, and we would prefer to use *python* as our language of choice - mainly for its conciseness, but also to exploit its support for numerical and computer vision package bindings (e.g. OpenCV).

We can process the data following the paradigm of Hadoop Streaming. To achieve this, and enable effective invocation of OpenCV we use the "Dumbo" python module, effectively a wrapper around the job invocation to let us execute python code, but which also - crucially - allows the python job to access the typed bytes that encode the image data to be processed.

We need (with versions):

- Python - version 2.7.2
- Hadoop (needs Java of course) - version 1.1.1
- FFmpeg - version 1.0.1
- ImageMagick - version 6.8.0
- Opencv - version 2.4.2
- Numpy - version 1.6.2
- matplotlib - version 1.2.0
- ipython - version 0.13.1
- dumbo - version 0.21.35

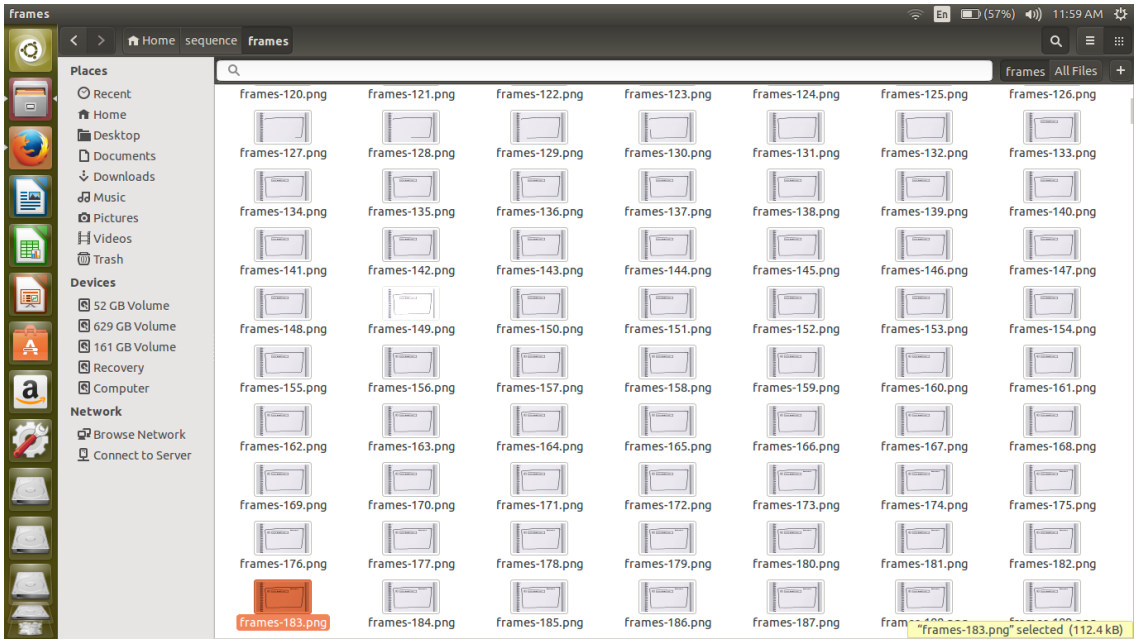

Fig. 22: Division of video into frames

3.9 Identification of faces from stored images

Eigenface: Eigenfaces is the name given to a set of eigenvectors when they are used in the computer vision problem of human face recognition.

The eigenvectors are derived from the covariance matrix of the probability distribution over the high-dimensional vector space of face images. The eigenfaces themselves form a basis set of all images used to construct the covariance matrix. This produces dimension reduction by allowing the smaller set of basis images to represent the original training images. Classification can be achieved by comparing how faces are represented by the basis set.

The code given below will perform the task of detecting the faces from an image.

```
import cv2
import sys

# Get user supplied values
imagePath = sys.argv[1]
cascPath = "/home/quinn/workspace/facedetect/haarcascade_frontalface_default.xml"

# Create the haar cascade
faceCascade = cv2.CascadeClassifier(cascPath)

# Read the image
image = cv2.imread(imagePath)
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Detect faces in the image
faces = faceCascade.detectMultiScale(
    gray,
    scaleFactor=1.1,
    minNeighbors=5,
    minSize=(30, 30),
    flags = cv2.cv.CV_HAAR_SCALE_IMAGE
)

print "Found {0} faces!".format(len(faces))

# Draw a rectangle around the faces
for (x, y, w, h) in faces:
    cv2.rectangle(image, (x, y), (x+w, y+h), (0, 255, 0), 2)

cv2.imshow("Faces found", image)
cv2.imwrite(sys.argv[1]+"face_detected.png", image)
cv2.waitKey(0)
```

Fig. 23: Input to Face Recognition Algorithm

Fig. 24: Output of Face Recognition Algorithm

3.10 Cropping of faces from the images

Fig. 25: Multiple input images for cropping face

We are giving two images in the input as shown in the figure. The code given below shows how the number of faces are detected from the input images and stored in the location provided. After executing the code the detected faces are cropped and stored in the folder given as the path for storing the cropped faces.

```
#Python 2.7.2
#Opencv 2.4.2
#PIL 1.1.7

import cv #Opencv
import Image #Image from PIL
import glob
import os

def DetectFace(image, faceCascade, returnImage=False):
    # This function takes a grey scale cv image and finds
    # the patterns defined in the haarcascade function
    #variables
    min_size = (20,20)
    haar_scale = 1.1
    min_neighbors = 3
    haar_flags = 0

    # Equalize the histogram
    cv.EqualizeHist(image, image)

    # Detect the faces
    faces = cv.HaarDetectObjects(
            image, faceCascade, cv.CreateMemStorage(0),
            haar_scale, min_neighbors, haar_flags, min_size
```

```
        )

    # If faces are found
    if faces and returnImage:
        for ((x, y, w, h), n) in faces:
            # Convert bounding box to two CvPoints
            pt1 = (int(x), int(y))
            pt2 = (int(x + w), int(y + h))
            cv.Rectangle(image, pt1, pt2, cv.RGB(255, 0, 0), 5, 8, 0)

    if returnImage:
        return image
    else:
        return faces

def pil2cvGrey(pil_im):
    # Convert a PIL image to a greyscale cv image

    pil_im = pil_im.convert('L')
    cv_im = cv.CreateImageHeader(pil_im.size, cv.IPL_DEPTH_8U, 1)
    cv.SetData(cv_im, pil_im.tostring(), pil_im.size[0]  )
    return cv_im

def cv2pil(cv_im):
    # Convert the cv image to a PIL image
    return Image.fromstring("L", cv.GetSize(cv_im), cv_im.tostring())
def imgCrop(image, cropBox, boxScale=1):
    # Crop a PIL image with the provided box [x(left), y(upper), w(width), h(height)]

    # Calculate scale factors
    xDelta=max(cropBox[2]*(boxScale-1),0)
    yDelta=max(cropBox[3]*(boxScale-1),0)

    # Convert cv box to PIL box [left, upper, right, lower]
    PIL_box=[cropBox[0]-xDelta, cropBox[1]-yDelta, cropBox[0]+cropBox[2]+xDelta,
cropBox[1]+cropBox[3]+yDelta]

    return image.crop(PIL_box)

def faceCrop(imagePattern,boxScale=1):
    faceCascade = cv.Load('haarcascade_frontalface_alt.xml')

    imgList=glob.glob(imagePattern)
    if len(imgList)<=0:
        print 'No Images Found'
        return

    for img in imgList:
        pil_im=Image.open(img)
        cv_im=pil2cvGrey(pil_im)
        faces=DetectFace(cv_im,faceCascade)
```

```
        if faces:
            n=1
            for face in faces:
                croppedImage=imgCrop(pil_im, face[0],boxScale=boxScale)
                fname,ext=os.path.splitext(img)
                croppedImage.save(fname+'_crop'+str(n)+ext)
                n+=1
        else:
            print 'No faces found:', img

def test(abc):
    pil_im=Image.open(abc)
    cv_im=pil2cvGrey(pil_im)

    faceCascade = cv.Load('haarcascade_frontalface_alt.xml')
    face_im=DetectFace(cv_im,faceCascade, returnImage=True)
    img=cv2pil(face_im)
    img.show()
    img.save('test.png')

# Crop all jpegs in a folder. Note: the code uses glob which follows unix shell rules.
# Use the boxScale to scale the cropping area. 1=opencv box, 2=2x the width and height
faceCrop('testPics/*.jpg',boxScale=1)
```

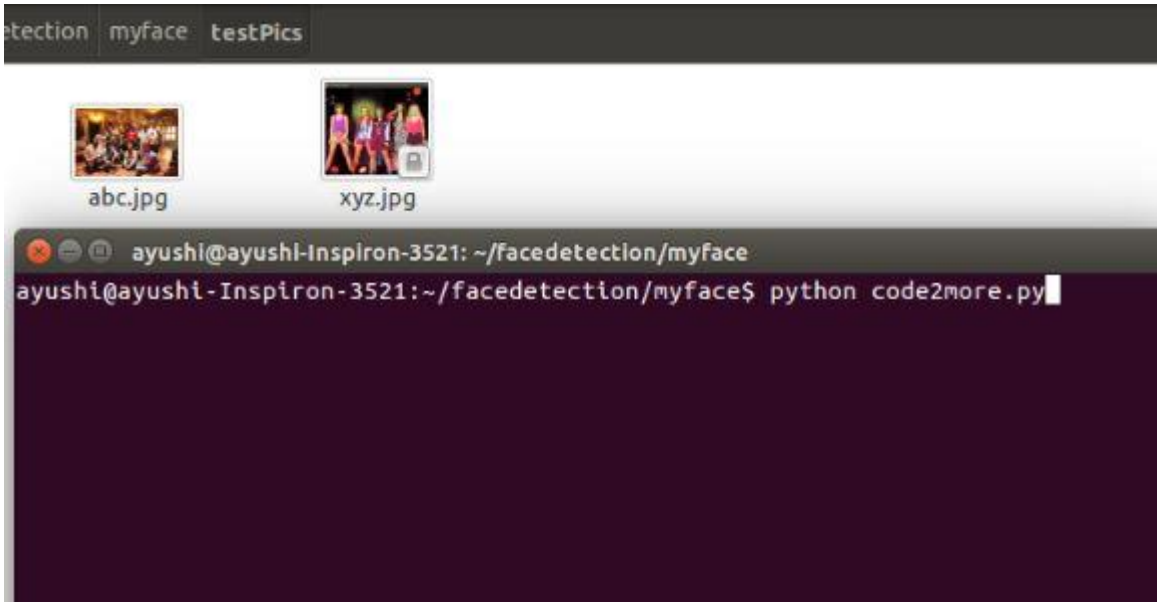

Fig. 26: Execution of code

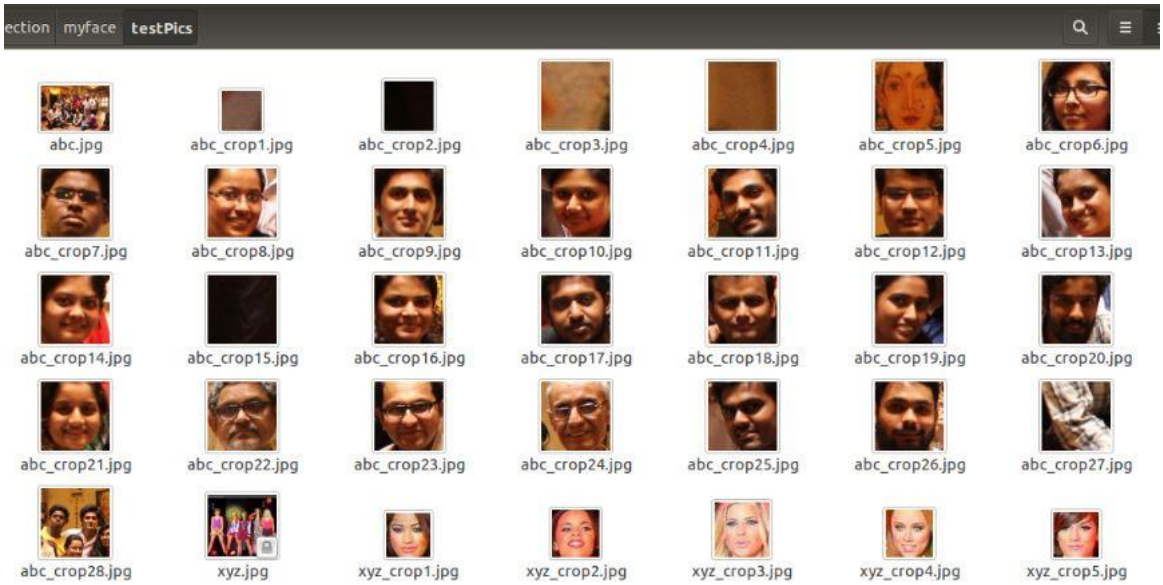

Fig. 27: Cropped faces as output

3.11 Comparing input image with images in the database

The following image is given as input and compared with the images in our database. If the match is found it gives the name of the suspect as output.

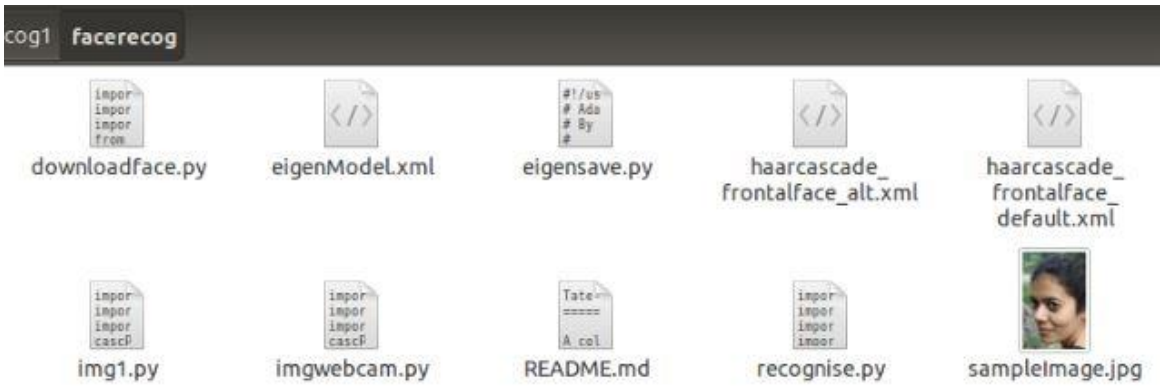

Fig. 28: Image of the suspect as input

Fig. 29: Database

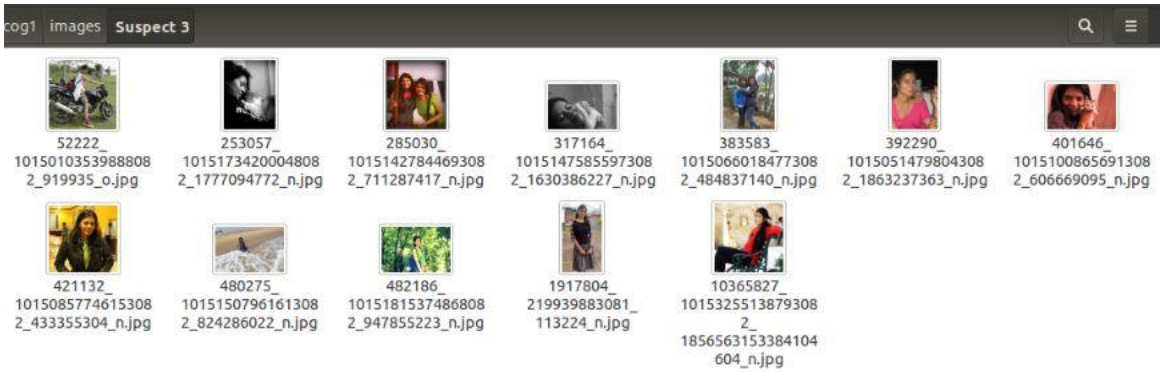

Fig. 30: Suspect detected as output

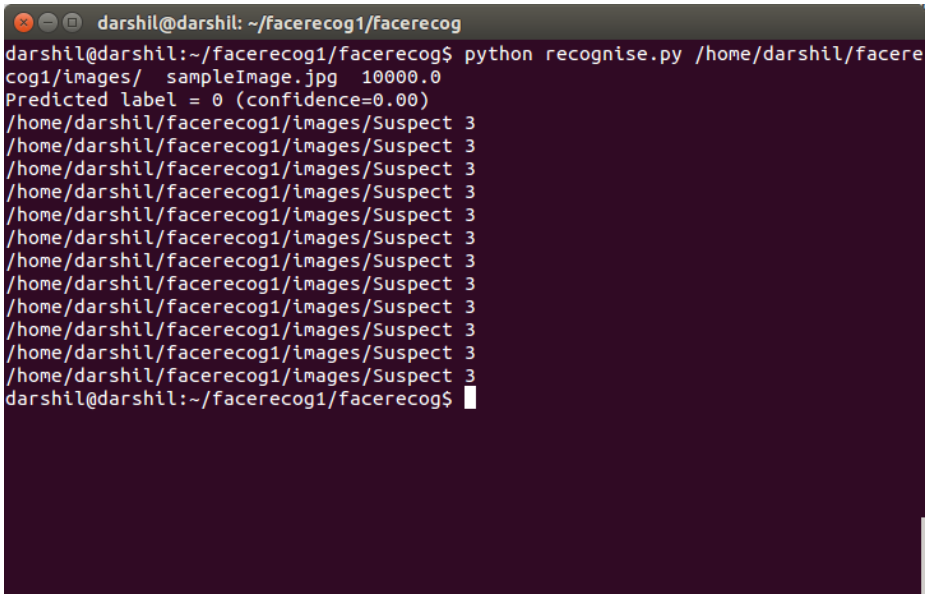

Fig. 31: Output of Face comparison Algorithm

CHAPTER-4 CONCLUSION AND FUTURE WORK

4.1 Objectives completed in this semester

- Edited the previous python code which now detects faces from a given image i.e. frames and then stores the new image which marks the face with a rectangle border.
- The detected faces through the previous code are now cropped by executing a new code. The code detects the faces, crops them and stores them in a given folder.
- An image containing the face of a suspect is given as input and compared with the images in our database to find a match.
- To detect faces throughout the video and store that images at one place i.e. file or folder in the system.
- Created a multi-node Hadoop cluster and deployed the whole video processing task on it.
- Our aim to justify our title and complete our project definition has been successfully achieved.

4.2 Future Work

The future scope of this project is to develop a system where a fast and accurate data processing is required. Where the user has the facility to search the media file from the tons of storage of which the user does not know the name of.

REFERENCES

1. [Alex Holmes, “Hadoop In Practice”, Manning Publications Co., New York, 2012]

2. [Tom White, “Hadoop – The Definitive Guide”, O’Reilly Publications, May 2012]

3. [S. Abiteboul, I. Manolescu, P. Rigaux, M. Rousset and P. Senellart, “Web Data Management”, Cambridge University Press, 2012].

4. Berlinska, J.; M. Drozdowski. (2011); Scheduling Divisible MapReduce Computations, Journal

of Parallel and Distributed Computing, 71(3): 450-459.

[3] Dean, J.; S. Ghemawat. (2010); MapReduce: A Flexible Data Processing Tool, Communications

of the ACM, 53(1): 72-77.

5. Dean, J.; S. Ghemawat. (2008); MapReduce: Simplified Data Processing on Large Clusters,

Communications of the ACM, 51(1): 1-13.

6. Dong, B.; et al. (2012); An Optimized Approach for Storing and Accessing Small Files on

Cloud Storage, Journal of Network and Computer Applications, 35(6): 1847-1862.

7. Dong, B.; et al. (2010); A Novel Approach to Improving the Efficiency of Storing and Accessing

Small Files on Hadoop: a Case Study by PowerPoint Files, IEEE International Conference

on Services Computing (SCC), Florida, USA: IEEE, DOI:10.1109/SCC.2010.72.

8. Golpayegani, N.; M. Halem. (2009); Cloud Computing for Satellite Data Processing on High

End Compute Clusters, IEEE International Conference on Cloud Computing, Bangalore,

India: IEEE, 88-92, DOI:10.1109/CLOUD.2009.71.

9. Ghemawat, S.; H. Gobioff.; S. T. Leung.(2003); The Google File System, Proceedings of the 19th ACM Symposium on Operating System Principles, NY, USA: ACM, DOI:10.1145/945445.945450.

10. H Rowley, S Baluja, and T Kanade, "Neural network-based face detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 1, pp. 23–38, Jan. 1998.

11. H Zhao and P C Yuen, "Incremental linear discriminant analysis for face recognition," IEEE Transactions on System Man and Cybernetics B, vol. 38, no. 1, pp. 210–221, Feb. 2008.

12. http://opencv.org

13. http://wiki.apache.org/hadoop/SequenceFile

14. https://hadoop.apache.org/docs/current/api/org/apache/hadoop/mapred/lib/CombineFileInputFormat.html

15. http://blog.pivotal.io/data-science-pivotal/products/using-hadoop-mapreduce-for-distributed-video-transcoding

16. http://www.academia.edu/6264430/The_Research_of_Smart_Surveillance_System_Using_Hadoop_Based_On_Craniofacial_Identification

17. http://createdigitalmotion.com/2009/02/processing-tutorials-getting-started-with-video-processing-via-opencv/

18. http://in.mathworks.com/help/vision/examples/face-detection-and-tracking-using-camshift.html

19. http://in.mathworks.com/help/vision/examples/object-detection-in-a-cluttered-scene-using-point-feature-matching.html

20. http://hadoop.apache.org/.

Hadoop Optimization for Massive Image Processing: Case Study Face Detection 671

21. http://face-rec.org/databases

22. http://vision.ucsd.edu/~leekc/ExtYaleDatabase/

23. J Lu, X. Yuan, and T. Yahagi, "A method of face recognition based on

fuzzy c-means clustering and associated sub-NNs," IEEE Transactions on

Neural Networks, vol. 18, no. 1, pp. 150–160, Jan. 2007.

24. Krishna, M.; et al. (2010); Implementation and Performance Evaluation of a Hybrid Distributed

System for Storing and Processing Images from the Web, 2nd IEEE International

Conference on Cloud Computing Technology and Science, Indianapolis, USA: IEEE, 762-767,

DOI:10.1109/CloudCom.2010.116.

25. Kocakulak, H.; T. T. Temizel. (2011); MapReduce: A Hadoop Solution for Ballistic Image

Analysis and Recognition, International Conference on High Performance Computing and

Simulation (HPCS),

.

lstanbul, Turkey, 836-842, DOI:10.1109/HPCSim.2011.5999917.

26. K K Sung, and T Poggio, "Example-based learning for view-based human

face detection," IEEE Transactions on Pattern Analysis and Machine

Intelligence, vol. 20, no. 1, pp. 39–51, Jan. 1998

27. Liu, X.; et al. (2009), Implementing WebGIS on Hadoop: A Case Study of Improving Small

File I/O Performance on HDFS, IEEE International Conference on Cluster Computing and

Workshops, Louisiana USA: IEEE, 1-8, DOI:10.1109/CLUSTR.2009.5289196.

28. M H Yang, D J Kriegman, , and N Ahuja, "Detecting faces in images: a

survey," IEEE Transactions on Pattern Analysis and Machine Intelligence,

vol. 24, no. 1, pp. 34–58, Jan. 2002

29. W Zhao, R Chellappa, P J Phillips, A Rosenfeld, “Face recognition: A literature survey,” ACM computing surveys, vol. 35, no. 4, pp. 399 – 458,

30. White, T. (2009); The Definitive Guide. 2009: O’Reilly Media.

31. www.hadoop.apache.org/ releases.html#Download

32. www.wiki.apache.org/hadoop

33. www.michael-noll.com/tutorials/

34. www.3pillarglobal.com/blog/how-configure-apache-hadoop-standalone-mode Dec 2003

APPENDIX

Periodic Progress Report-1

5/15/2015 Periodic Progress Report (PPR) Details

Periodic Progess Report : **First PPR**

Project Video Processing Using Hadoop

:

Status : Reviewed (Freeze)

What Progress you have made in the Project ?

We have edited the python code so that it now detects faces from a given image i.e. frames and then stores the new image which marks the face with a rectangle border. So we get all the faces in images, highlighted with rectangular borders around each one of them.

What challenge you have faced ?

We had to go through numerous image processing websites to attain accuracy with respect to our task.

What support you need ?

Assistance is required for implementing python programs onto Hadoop cluster.

Which literature you have referred ?

Face Recognition with OpenCV - opencv documentation How to do face recognition with python and opencv stackoverflow.com Face detection in Python using a webcam - Real Python by realpython.com **Comment by Internal Guide :**

OK

Periodic Progress Report-2

5/15/2015 Periodic Progress Report (PPR) Details

Periodic Progess Report : Second PPR

Project Video Processing Using Hadoop

:

Status : Reviewed (Freeze)

What Progress you have made in the Project ?

The detected faces are now cropped by executing a new code. The code detects the faces, crops them and stores them in a given folder.

What challenge you have faced ?

We had to go through various face cropping algorithms to decide upon the suitable match for our requirement.

What support you need ?

For now we

not need

any support.

Which

literature you

have referred

?

stackoverflow.com-detect-face-then-autocrop-pictures
chairnerd.seatgeek.com-opencv-face-detection-for-croppingfaces

Comment by Internal Guide :

Good

Periodic Progress Report-3

5/15/2015 Periodic Progress Report (PPR) Details

Periodic Progess Report : Third PPR

Project Video Processing Using Hadoop

:

Status : Reviewed (Freeze)

What Progress you have made in the Project ?

We have given an image containing the face of a suspect as input and compared with the images in our database to find a match. After the successful match a folder with the data of that suspect is provided to the user as output.
We have detected faces throughout the video and stored that images at one place i.e. file or folder in the system.

What challenge you have faced ?

We had network problems. So it took a lot of time to do a task which actually could have completed earlier.

What support you need ?

We need help in creating a multinode Hadoop cluster and guidance in how to run python code on Hadoop.

Which literature you have referred ?

Face comparison algorithms.

Comment by Internal Guide :

OK

Periodic Progress Report-4

5/15/2015 Periodic Progress Report (PPR) Details

Periodic Progess Report : **Forth PPR**

Project Video Processing Using Hadoop

:

Status : Reviewed (Freeze)

What Progress you have made in the Project ?

We have successfully implemented our video processing tasks on multi-node Hadoop cluster so that the output obtained now takes much lesser time as compared to those implemented normally. Hence, we have completed our project definition successfully.

What challenge you have faced ?

We had to generate mapper and reducer code in python which is not available worldwide, and so we had challenges in creating the same.

What support you need ?

We are almost towards the completion of our project, still we would like some help regarding the challenge we have faced and content related to that if it can be provided.

Which literature you have referred ?

1. Automatic Face Detection Using Color Based Segmentation-Yogesh Tayal, Ruchika Lamba, Subhransu Padhee

Department of Electrical and Instrumentation Engineering Thapar University, Patiala-147004, Punjab, India 2. N

APPROACH FOR FAST AND PARALLEL VIDEO PROCESSING ON APACHE HADOOP CLUSTERS Hanlin Tan,

Lidong Chen College of Information System and Management, National University of Defense Technology, China 3.

The research of smart surveillance system using hadoop based on craniofacial identification Venkata lakshmi.k , Venkateswaran.s

Comment by Internal Guide :

GOOD

Business Model Canvas

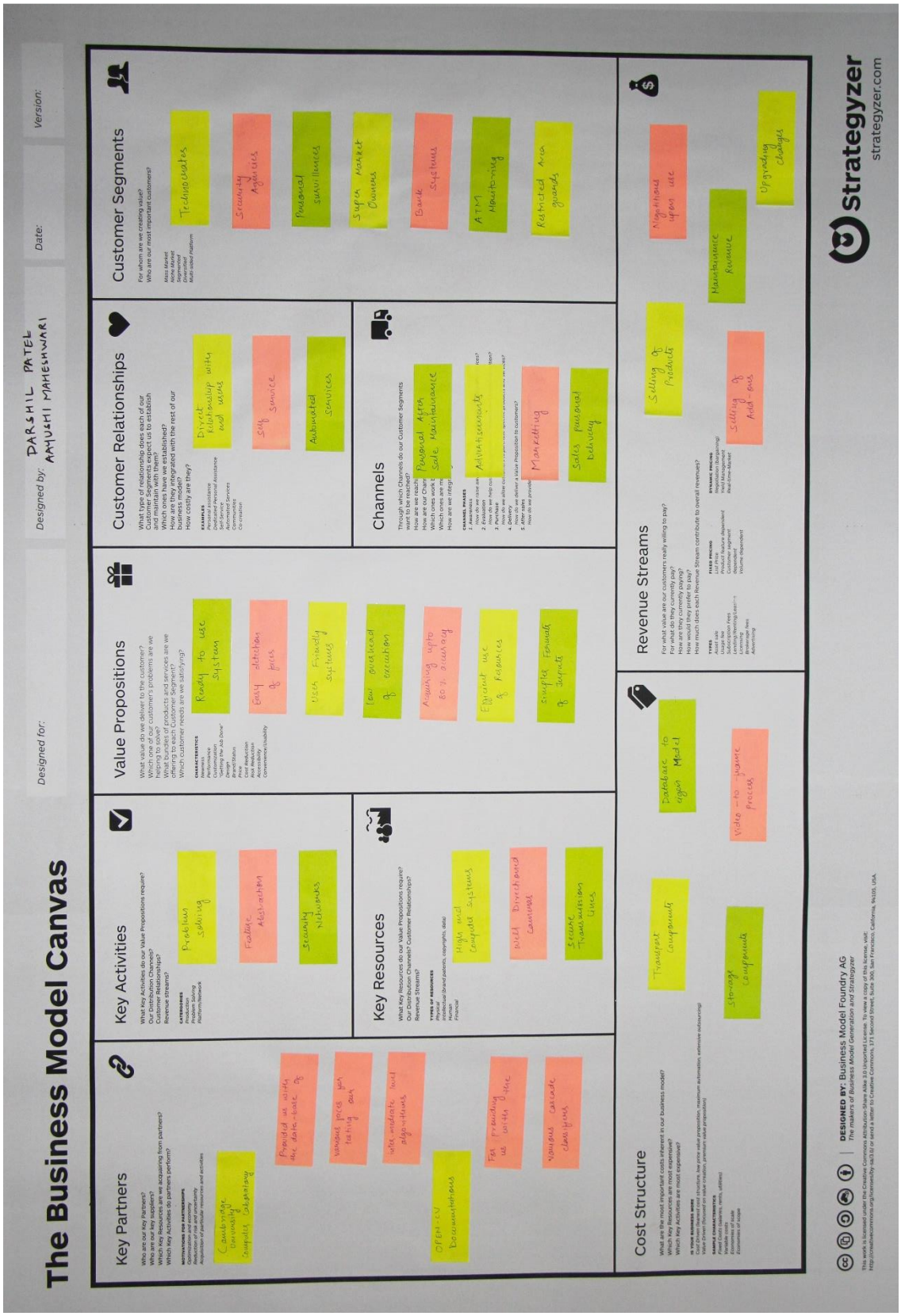

Fig. 32: Business Model Canvas

Business Model Canvas Report

1) Key Partners:-

Our key partners that have helped us in either making the project or indirectly have been responsible for the completion of our project are as follows:

a) Cambridge University (Computer Laboratory): It has provided us with the database which atmost contains 600 pictures of various people, which we have utilized for testing our algorithms.

b) OpenCV Documentation: It has helped us to get the cascade classifiers which is a major help in our project.

2) Key Activities:-

a) Problem Solving

b) Feature Abstraction

c) Security Network

3) Key Resources:-

a) High-end computer systems: The most fundamental part of our project is high end computers which help us in making the distributed system.

b) Well directioned camers: It is very important to aling and attach the cameras such that very clear and proper video can be captured.

c) Secure Transmission lines: We have to be very carefull about the security of the transmission that takes place between the webcamers and the projected system.

4) Value Propositions:-

a) Ready to use system

b) Easy detection of faces

c) User Friendly systems

d) Low overhead of execution

e) Acquiring upto 80% accuracy

f) Efficient use of resources

g) Simpler Formats of inputs

5) Customer Relationships:-

a) Direct relationship with the end users

b) Self Service

c) Automates Services

6) Channels:-

a) Personal after sale maintainance

b) Advertisement

c) Marketting

d) Sales personal delivery

7) Customer Segments:-

a) Technocrates

b) Security Agencies

c) Personal Survillances

d) Super market Owners

e) Bank systems

f) ATM Monitiring

g) Restricted Area Guards

8) Cost Structure:-

a) Transport Components

b) Storage components

c) Database to eigen Model

d) Video to frame process

9) Revenue Streams:-

a) Selling of the product

b) Selling of the Add-ons

c) Maintainence Revenues

d) Negotitions upon use

e) Upgrading Charges

Form-1

GTU Innovation Council

Patent Drafting Exercise (PDE)

FORM 1 THE PATENTS ACT 1970 (39 OF 1970) & THE PATENTS RULES, 2003 APPLICATION FOR GRANT OF PATENT	(FOR OFFICE USE ONLY) Application No: Filing Date: Amount of Fee paid: CBR No: _____

1. Applicant(s) :

ID	Name	Nationality	Address	Mobile No.	Email
1	Aayushi Jagdish Maheshwari	Indian	Computer Engineering , A. D. Patel Institute Of Technology, Karamsad , Gujarat Technologycal University.	9429950749	ayushimaheshwari1412@gmail.com
2	Darshil Dinesh Bhai Patel	Indian	Computer Engineering , A. D. Patel Institute Of Technology, Karamsad , Gujarat Technologycal University.	8460584110	darshil_dpatel@yahoo.co.in

2. Inventor(s):

Note : This is just a mock Patent Drafting Exercise (PDE) for semester 8, BE students of GTU.
These documents are not to be submitted with any patent office.

Page 1 of 4

ID	Name	Nationality	Address	Mobile No.	Email
1	Aayushi Jagdish Maheshwari	Indian	Computer Engineering , A. D. Patel Institute Of Technology, Karamsad , Gujarat Technologycal University.	9429950749	ayushimaheshwari1412@gmail.com
2	Darshil Dinesh Bhai Patel	Indian	Computer Engineering , A. D. Patel Institute Of Technology, Karamsad , Gujarat Technologycal University.	8460584110	darshil_dpatel@yahoo.co.in

3. Title of invention/Project:

Video Processing Using Hadoop

4. Address for correspondence of applicant/authorized patent agent in India

Name: Darshil Dinesh Bhai Patel

Address: Computer Engineering , A. D. Patel Institute Of Technology, Karamsad , Gujarat Technological University.

Mobile: 8460584110

Email ID: darshil_dpatel@yahoo.co.in

5. Priority particulars of the application(s) field in convention country

Country	Application No.	Filing Date	Name of the Applicant	Title of the Invention
N/A	*N/A*	*N/A*	*N/A*	*N/A*

6. Particulars for filing patent co-operation treaty (pct) national phase Application

International application number	International filing date as alloted by the receiving office
N/A	N/A

7. Particulars for filing divisional application

Original(First) Application Number	Date of filing of Original (first) application
N/A	**N/A**

Note : This is just a mock Patent Drafting Exercise (PDE) for semester 8, BE students of GTU.
These documents are not to be submitted with any patent office.

8. Particulars for filing patent of addition

Original(First) Application Number	Date of filing of Original (first) application
N/A	N/A

9. DECLARATIONS:

(I) Declaration by the inventor(s)

I/We, the above named inventor(s) is/are true & first inventor(s) for this invention and declare that the applicant(s).
herein is/are my/our assignee or legal representative.
Date : 16 - May - 2015

	Name	Signature & Date
1	Aayushi Jagdish Maheshwari	_____
2	Darshil Dinesh Bhai Patel	_____

(II) Declaration by the applicant(s) in the convention country

I/We, the applicant (s) in the convention country declare that the applicant(s) herein is/are my/our assignee or legal representative applicant(s)

(III) Declaration by the applicant(s)

I/We, the applicant(s) hereby declare(s) that:-

- [✓] I am/We in possession of the above mentioned invention.
- [✓] The provisional/complete specification relating to the invention is filed with this aplication.
- [✗] The invention as disclosed in the spcification uses the biological material from India and the necessary permission from the competent authority shall be submitted by me/us before the grant of patent to me/us.
- [✗] There is no lawful ground of objection to the grant of the patent to me/us.
- [✓] I am/we are the assignee or the legal representative of true & first inventors.
- [✗] The application or each of the application,particulars of each are given in the para 5 was the first applicatn in the convention country/countries in respect of my/our invention.
- [✗] The application or each of the application,particulars of each are given in the para 5 was the first applicatn in the convention country/countries in respect of my/our invention.
- [✗] I/we claim the priority from the above mentioned applications(s) filed in the convention country/countries & state that no application for protection in respect of invention had been made in a convention country before that date by me/us or by any person
- [✗] My/Our application in India is based on international application under Patent Cooperation Treaty (PCT) as mentioned in para 6
- [✗] The application is divided out of my/our application(s) particulars of which are given in para 7 and pray that this application may be treated as deemed to have been filed on _____under section 16 of the Act.
- [✗] **The said invention is an improvement in or modification of the invention particulars of ehivh are given in para 8.**

Note : This is just a mock Patent Drafting Exercise (PDE) for semester 8, BE students of GTU.
These documents are not to be submitted with any patent office.

10. Following are the attachments with the application:

- [✓] (a) Provisional specification/Complete specification
- [x] (b) Complete specification(In confirmation with the international application) / as amended befo International Preliminary Examination Authority (IPEA),as applicable(2 copies),No.of pages.. claims.....
- [✓] (c) Drawings (In confirmation with the international application)/as amended before the inter Preliminary Examination Authority(IPEA),as applicable(2 copies),No.of sheets....
- [x] (d) Priority documents
- [x] (e) Translations of priority documents/specification/international search reports
- [✓] (f) Statement and undertaking on Form 3
- [x] (g) Power of Authority
- [x] (h) Declaration of inventorship on Form 5
- [x] (I) Sequence listing in electronic Form
- [x] (j) ..... Fees Rs.XXX in Cash /Cheque/Bank Draft bearin No.XXX Date: XXX o Bank.

I/We hereby declare that to the best of my /our knowledge, information and belief the fact and mtters stated herein are correct and I/We request that a patent may be granted to me/us for the said invention.

Dated this 16 day of May , 2015

	Name	Signature & Date
1	Aayushi Jagdish Maheshwari	_____
2	Darshil Dinesh Bhai Patel	_____

Note : This is just a mock Patent Drafting Exercise (PDE) for semester 8, BE students of GTU.
These documents are not to be submitted with any patent office.

Page 4 of 4

Form-2

FORM 2
THE PATENTS ACT, 1970
(39 OF 1970)
&
THE PATENTS RULES, 2003
PROVISIONAL SPECIFICATION

1. Title of the project/Invention :

Video Processing Using Hadoop

2. Applicant(s) :

Aayushi Jagdish Maheshwari , (Indian)
Address :Computer Engineering , A. D. Patel Institute Of Technology, Karamsad , Gujarat Technologyal University.

Darshil Dinesh Bhai Patel , (Indian)
Address :Computer Engineering , A. D. Patel Institute Of Technology, Karamsad , Gujarat Technologyal University.

3. Preamble to the description :

The following specification describes the invention.

4. Description :

a. Field of Application / Project / Invention :

Image Processing and Face Detection

b. Prior Art / Background of the Invention / References :

This project works onto the task of extracting the faces out of the video reducing the overhead of watching the whole video. This task can be made faster by implementing it onto some kind of cluster which here in our case is hadoop.

c. Summary of the Invention/Project :

We here are majorly focusing on two tasks: massive data storage and
faster processing. We want to use open source tools to accomplish this task.The project basically extracts the faces from the video and then compares the same with the database of faces that we have with us .By this way we can know of the persons in the video without actually watching the video which saves a lot of time. Plus the feature of face comparison is done more efficiently by computers than the human brain .Thus this system is of immense usefulness.

d. Objects of the Invention/Project :

Cluster Computing (Hadoop).
Image processing (Opencv).
Facial Recognition.

e. Drawing(s) :

f. Description of the Invention

We want to use open source to make the process of extraction faces from video easier. First of all we convert video into frames and then we process that frames of video that have faces into it .Now we detect faces from the images and then store all the faces into a specific folder .Now we can select a face image which we want to identify and then compare it with the database of the faces that we have with us. This kind of arrangement can help various customers who have to make a check of the people by video means.

g. Examples

h. Unique Features of the Project

Less time consuming.
Simple to use.

5. Date & Signature :

Date : 18 - May - 2015

Sign and Date Aayushi Jagdish Maheshwari	Sign and Date Darshil Dinesh Bhai Patel

8. Abstract of the project / invention :

Recent research area projects usually include heavy processing of data which is a cumbersome, tedious and time consuming process. Hadoop is an open-source software framework for storing and processing big data in a distributed fashion on large clusters ofcommodity hardware. Essentially, it accomplishes two tasks: massive data storage andfaster processing.

Open-source software. Open source software differs from commercial software due to the broad and open network of developers that create and manage the programs. Traditionally, it's free to download, use and contribute to, though more and more commercial versions of Hadoop are becoming available.

Framework. In this case, it means everything you need to develop and run, your software applications is provided – programs, tool sets, connections, etc.

Distributed. Data is divided and stored across multiple computers, and computations can be run in parallel across multiple connected machines.

Massive storage. The Hadoop framework can store huge amounts of data by breaking the data into blocks and storing it on clusters of lower-cost commodity hardware.

Faster processing. Hadoop processes large amounts of data in parallel across clusters of tightly connected low-cost computers for quick results.

With the ability to economically store and process any kind of data (not just numerical or structured data), organizations of all sizes are taking cues from the corporate web giants that have used Hadoop to their advantage (Google, Yahoo, Etsy, eBay, Twitter, etc.).

Moving forward to next phase of this project is to detect faces out from the video, which can be of great help reducing a lot of time scanning a large video. OpenCV is a very useful library which can be integrated along with the python programming language. OpenCV again is an open source library that is developed in C++. There are many situations where an organization has to scan the video footage of its various departments or there retail stores, so that they can make out how many and who all visited the respective department

or that retail store. So that company can be ready in the situations like thefts or some disasters. Hereby we are trying to create a system which can help us achieve the above mentioned goal. Thus amalgamation of Big Data and Python along with OpenCV can be of great help. We have taken a step towards the enlightenment of this domain by means of our final year project.

Drawing Attachments :

Note : This is just a mock Patent Drafting Exercise (PDE) for semester 8, BE students of GTU.
These documents are not to be submitted with any patent office.

Page 4 of 4

Form-3

FORM 3
THE PATENTS ACT, 1970
(39 OF 1970)
&
THE PATENTS RULES, 2003
STATEMENT AND UNDERTAKING UNDER SECTION 8

1. Declaration :

I/We, Aayushi Jagdish Maheshwari ,
Darshil Dinesh Bhai Patel ,

2. Name, Address and Nationality of the joint Applicant :

Aayushi Jagdish Maheshwari (Indian)
Address : Computer Engineering , A. D. Patel Institute Of Technolc Karamsad , Gujarat Technologycal University.

Darshil Dinesh Bhai Patel (Indian)
Address : Computer Engineering , A. D. Patel Institute Of Technolc Karamsad , Gujarat Technologycal University.

Here by declare:

(I) that I/We have not made any application for the same/substantially the same invention outside India.
(II) that the right in the application(s) has/have been assigned to,

Name of the Country	*Date of Application*	*Application Number*	*Status of the Application*	*Date of Publication*	*Date of Grant*
N/A	*N/A*	*N/A*	*N/A*	*N/A*	*N/A*

(III) that I/We undertake that up to the date of grant of patent by the Controller , I/We would keep him inform in writing the details regarding corresponding application(s) for patents filed outside India within 3 months from the date of filing of such application.

Dated this 18 day of May , 2015.

3. Signature of Applicants :

Sign and Date
Aayushi Jagdish
Maheshwari

Sign and Date
Darshil Dinesh Bhai Patel

To
The Controller of Patent
The Patent Office, at **Mumbai.**

Note : This is just a mock Patent Drafting Exercise (PDE) for semester 8, BE students of GTU.
These documents are not to be submitted with any patent office.