Extended Distributed Framework for Feature Extraction in Remote Sensing Imagery with High Resolution

T. Naga Raju
Research Scholar, Nagarjuna University, India. E-mail: nagarajut514@gmail.com

Dr. Chittineni Suneetha
Associate Professor, Department of MCA, R.V.R. & J.C. College of Engineering, India.
E-mail: suneethachittineni@gmail.com

Received May 15, 2021; Accepted September 18, 2021
ISSN: 1735-188X
DOI: 10.14704/WEB/V18I2/WEB18372

Abstract

Remote Sensing imagery is used vastly in the areas of human activities investigation, environmental changes monitoring and geo-spatial data updation in a rapidly increasing way. Humans can easily and appropriately interpret the normally shot pictures but this is a difficult task for the computer to automatically interpret information from the given images. One of the prominent phases is in finding the way to extract the projected information from the given imagery and its conversion to wrath-ful data which can be used for further research. The motto is the generation of an algorithm which aims to be very efficient during of processing of huge images that include enhancement of efficiency in processing, correlation finding amongst given data and extraction of continuous features. In order to accomplish all these purposes as stated above, we hereby put forward an algorithm Extended Feature Extraction and Detection in High Resolution Remote Sensing (HRRS) Imagery to detect rivers. The proposed system is established with Hadoop Distributed Framework in order to enhance the efficiency of total system.

Keywords

Feature Extraction, Image Processing, Remote Sensing.

Introduction

Remote Sensing technology is one of the vital methodologies for instant and direct procurement of Earth's surface information. In the current years, with the advent of science of environmental information, data procured by remote Sensing played a vital role in several fields of research like Earth's crust evolution, volcanic phenomena, soil science,

environmental geology, water revolution, soil contamination, ecology and atmospheric science.

All the research requirements have paved the way for expansion of technologies in Earth observation. In this context several countries have quickly launched several satellites on their own. Summary of the total number of satellites used for remote Sensing launched by different countries during 1962- 2014 is given in Figure 1. It can be depicted that USA, India, Russia bagged top 3 positions in launching highest number of remote sensing satellites. During 2001 - 2014 majority of the regions and countries have launched their own satellites.

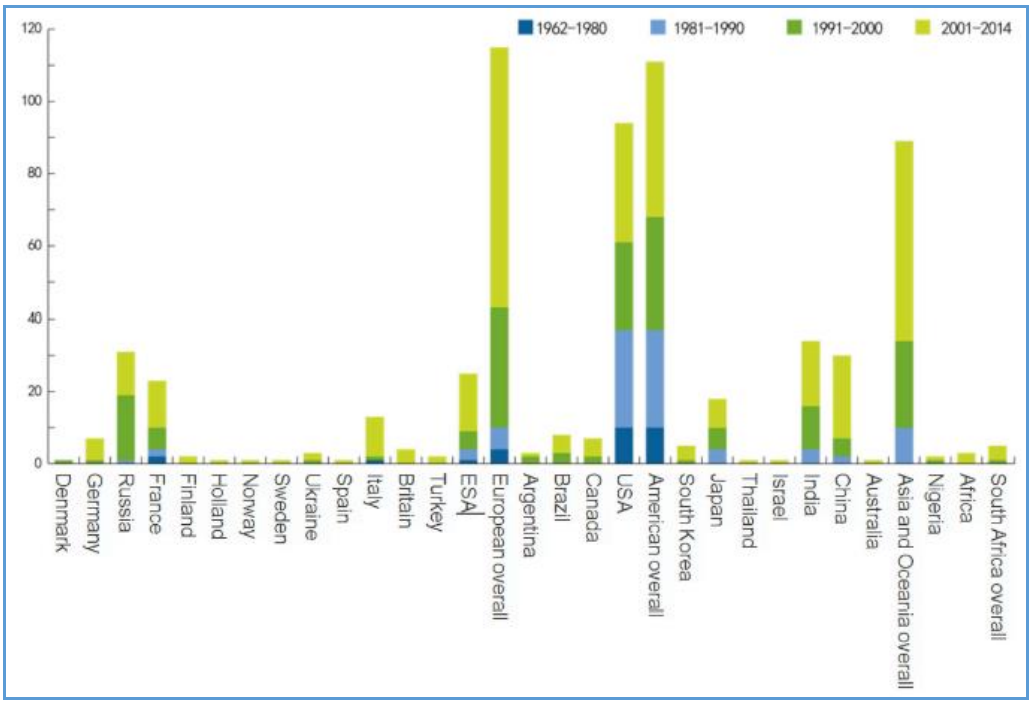

Figure 1 Approximate count of satellites launched by major nations or regions

The Remote Sensing data itself is voluminous. The capacity of data generated by remote sensing only for one scene will be as large as in terms of Giga Bytes. The data received from the satellites will be in the terms of Tera Bytes, Peta Bytes and Exa Bytes. The data acquisition rate must be very high because there exist many satellites in the Earth's orbit. In case of China the RSGS (Remote Sensing Satellite Ground Station) receives minimum of 1TB data per day. Hence this clearly shows that remote Sensing data is big data.

Remotely sensed data, vary in spatial, radiometric, spectral, and temporal resolutions. Understanding the strengths and weaknesses of different types of sensor data is essential for the selection of suitable remotely sensed data for image classification. Some previous

literature has reviewed the characteristics of major types of remote-sensing data (Barnsley 1999, Estes and Estes 1999, Althausen 2002, Lefsky and Cohen 2003). For example, Barnsley (1999) and Lefsky and Cohen (2003) summarized the characteristics of different remote-sensing data. The selection of suitable sensor data is the first important step for a successful classification for a specific purpose (Phinn 1998, Jensen and Cowen 1999, Qiu et al. 2000, Lefsky and Cohen 2003). Peng Lu, 2015 discussed that big data generated by Remote Sensing technology has special and materialistic characteristics such as non-linearity, isomer, dynamic state, high dimensional, multi-scale and multi-source. These days, with the launch of many satellites into the space with continuous enhancements in sensor Technology, the Remote Sensing data has obviously become high resolution data discussed by Wang et al, 2018.

The high resolution images has a finer spatial scale, which provides abundant quality data that consists of spectral features, textural features and geometric features which results in several new challenges as opposed to traditional image segmentation methodologies discussed by Yan Ma, 2014. The high resolution imagery (QuickBird, Landsat) provides us with most prominent featured like texture, shape and structure these are also rich in spectral data. These images have effective and vivid data sources for fine agriculture, land survey and ecological environment protection discussed by Lu, 2015. Due to the quality and detail of information, interference factor influence like small boundaries and targets are obvious. Binbin Chen, 2018 proposed that spectral features are utilised in classifying high-resolution remote sensing images with the phenomenon "Same object with different spectra" as well "Different objects with same spectrum" causes less accuracy in classification.

Wide research studies illustrates that wide range resolution of spectrum along with texture features will contribute in improvising feature extraction were discussed by Li, 2006 and Chen 2008. Zhang et al, 2016 researched on classification based on texture feature, spectral images and DEM information. Results conveyed that classification accuracy is improved by texture. Zhao, 2016 utilized both texture features and Spectrum in order to achieve higher accuracy extraction.

An efficient and novel methodology to retrieve appropriate images from huge databases is in need. The prominent approach of image retrieval is the context based approach. Content based image retrieval is very popular. This approach will index the image with help of spatial layout, faces, texture, shape and colour etc. These indexes will be extracted from image, and then represented in database repository. This approach makes extensive utilization of Local Tetra Patterns with feature descriptors which were initially announced

by Subramanyam et al, 2012. As this approach for feature vector matching and computation queries, entire thing will operate in an independent manner on the image; this approach can have parallelism in great extent.

Apache Hadoop in coordination with Map-Reduce programs with single point node in combination with sophisticated analysis will be used for implementation of proposed algorithm. Nagaraju et al, 2019 were discussed that Hadoop provides high performance computing and extensive parallelism with use of huge number of servers. Hence this is suitable to analyse huge amount of remote sensing imagery. In this paper, proposed an extended architecture based on distributed model proposed Nagaraju et al, 2019. So, utmost priority is given for Hadoop because of its sophisticated analysis, testing and algorithm development.

Methodology

Depending on the analysis and study done in the prior section on HRRS imagery, proposed system is developed to extract the prominent features. The system consists of the algorithm and complete implementation. Proposed system consists of several steps which include interpretation, merging, processing, segmentation, filtration and data collection. Methodology of the proposed model is diagrammatically represented in Figure 2.

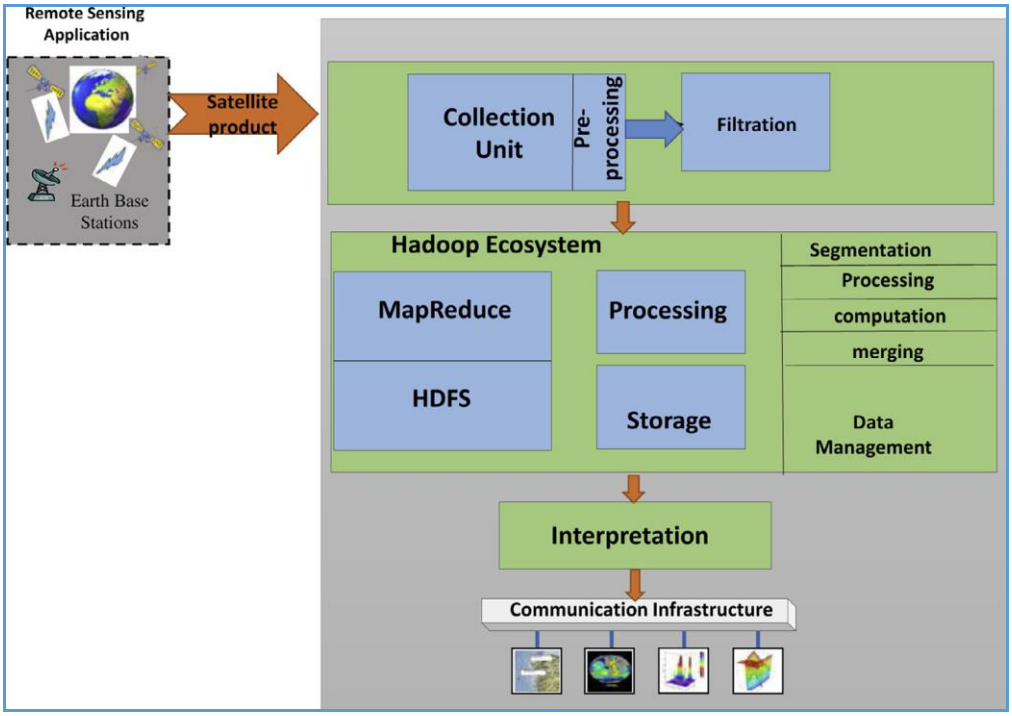

Figure 2 Proposed model

This consists of several units for each phase. The initial step was with the collection of data. The succeeding step is filtration, which will filter important information from the given data set and passes it to Hadoop system. Next goes the Hadoop system which utilizes Extended HDFS (EHDFS) system, which will act as the core and is accountable for data's key processing. At final stage, there lies the interpretation unit where it utilizes the results that are generated by Hadoop system for extraction of features from satellite imagery.

E-HDFS: Extended HDFS

Existing model proposed by Nagaraju et al, 2019 namely "Data Organization and Processing in Hadoop using HDFS and Map Reduce" are tailored in an ideal manner for applications of search and data in a parallel manner, where there exist nil data dependency amongst neighbouring / adjacent data. In this proposal, discussion of requirements of organisation of data overlapping and then proposal of E-HDFS to HDFS in order to address all the necessities will be done. We shall present Application Programming Interface (APIs) and also discuss the implementation details which are precise to HRRS imagery. E_HDFS consists of tiny overheads during storage of data as well I/O operations, but it will improve performance of the system in a great manner and also does the application development simple. Proposed system will work deprived of any modifications or variations in currently existing map-reduce models with nil overhead, this approach can also be utilised in domain specific applications that deal with overlapped data.

In this segment description of Extended – HDFS will be given; Hadoop's extended software package utilized for huge scale image processing. E-HDFS APIs for I/O are as follows.

HDFS I/O Extensions

Fig. 3 portrays steps in sequential order for R/W the images with E-HDFS library in Hadoop. At the initial step, client utilizes E-HDFS I/O methods for R/W of data. Client request will be interpreted into create ( ) / open ( ) by E-HDFS, then delivered DFS. DFS instances will call name node for determining the locations data blocks. For every block, name node will return address of data nodes to R/W data. The DFS will return FS Data I/O Stream that will be used by E-HDFS for R/W the data from/to data nodes. E-HDFS will check the file's format, if given format is of an image, metadata will be saved in the HBASE. This will simplify reading the header information when mandated by HBASE queries; else reading whole header in block wise manner will be very tough and time taking procedure.

Later, E-HDFS will call FS Data I/O Stream in order to R/W data from/to data nodes respectively. Then, a standard R/W HDFS data will be done in a pipelined manner. Every block will be written which consists header information belonging to respective blocks i.e. blockID, scan start, scan end, number of overlapped scan lines in block, total scan length, total size of block. At last, after R/W operation, request will be made to close the file, then the status shall be forwarded to name node.

Read operations of HDFS can be realized in either of both ways, one amongst them is to realize own split function, which ensures that boundaries are not split, and the other is to use FIXED LENGTH RECORD belonging to FixedLengthInputFormat class. Because the block sizes are fixed, we use record format fixed length. Below is the description of APIs used. The following commands show the Sample job configuration for MapReduce.

1. JobConfconf = new JobConf(ImageMapReduce.class);
2. conf.setWorkingDirectory(new Path("hdfs://namenode/user/hduser"));
3. conf.addResource(newPath("/home/hduser/hadoop-2.7.0/etc/hadoop/core-site.xml"));
4. conf.addResource(newPath("/home/hduser/hadoop-2.7.0/etc/hadoop/hdfs-site.xml"));
5. conf.setInt(FixedLengthInputFormat.FIXED_RECORD_LENGTH, blocklength);
6. conf.setInputFormat(FixedLengthInputFormat.class);

The input image will be organised in the form of divided blocks in HDFS, which consists some overlap amongst succeeding blocks. This block will be constructed in two manners as depicted in figure 4. During construction, an assurance that no splitting has occurred in the boundaries of pixel bytes has to be confirmed. In the unidirectional split mechanism, block will be constructed by segmentation of data in horizontal way. Every block will be appended with supplemented lines at block end. In Bidirectional splitting mechanism, the block will be split into both vertical and horizontal directions. The result will be such that start and end blocks will overlap with neighbouring blocks and all other blocks shall overlap through neighbouring four blocks. This sort of segmentation results in storing overheads which is merely doubled the capacity of unidirectional segmentation mechanism. This sort of organisation will be preferred when images consist of large length of scan lines.

In existing version of E-HDFS of, organisation of data has been given for unidirectional block segmentation. This mechanism can be formulated to work with bidirectional splits.

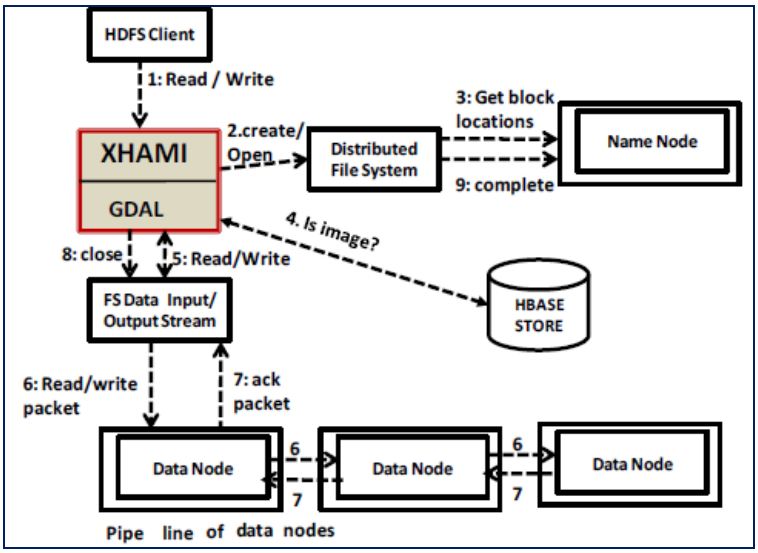

Figure 3 E-HDFS for read/write operations

The prominent E-HDFS APIs of Input Output associated procedures are defined in Table 1. E-HDFS reads and writes all image blocks in same format as of the original file, using Geographical Data Abstraction Layer (GDAL) library through I/O and MapReduce operations.

int xhmrWriteImage(String file, int overlap) Explanation: File contents will be written into HDFS along with the stated count of overlapped scan lines. This method is utilized for writing image sort files. Return value: 1 on success, 0 on failure.
int xhmrWriteFile(String file) Explanation: Non image content of files are written onto HDFS. Return value: 1 on success, 0 on failure.
String[] xhmrReadFile(String file) Explanation: to write non image files onto HDFS. Return value: file content as string
byte[] xhmrReadImage(String file) Explanation: read file content from HDFS. Return value: binary format file
int xhmrReadGetTotalScans(String file) Explanation: returns number of totally scanned lines in the image with file name. Return value: number of scanned lines
int xhmrReadGetTotalPixels(String file) Explanation: returns total pixel count of image. Return value: binary format of file.
byte[] xhmrReadGetRoi(String file, int startscan int start pixel,int blockwidth, int blockheight) Explanation: reads image's region of interest, starting at startpixel, with given block size represented by blockwidth and blockheight Return value: read file content's binary format.
byte[] xhmrReadGetBlockData(String file, int blocknumber) Description: returns bytes at given blocknumber of file. Return value: binary data of the block
String[] xhmrReadGetBlockHeader(String file, int blocknumber) Description: reads header of file at given blocknumber. Return status: header content of file at given block number.

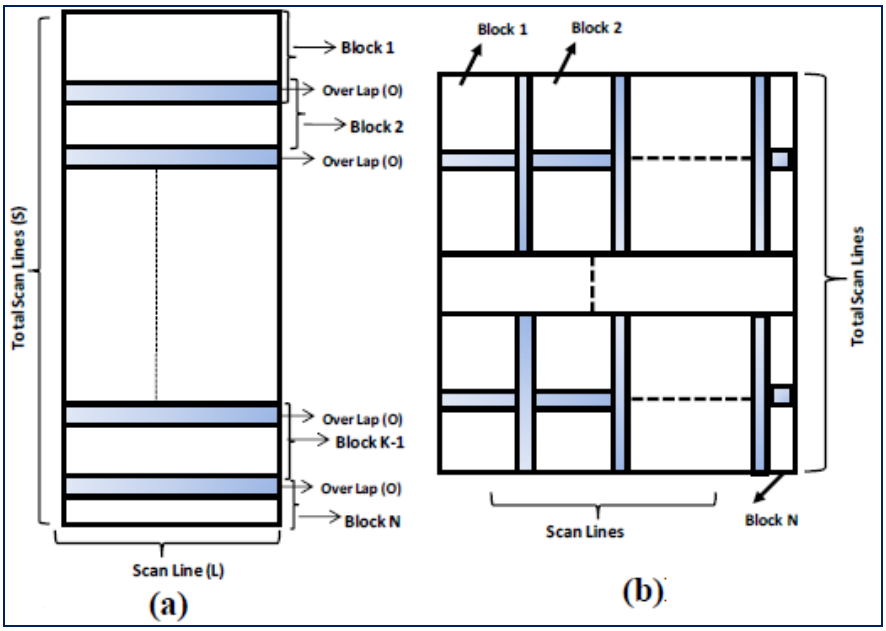

Figure 4 Block construction methods a) Unidirectional b) Bidirectional

Experimental Results

The data utilised for this experiment is WorldView-2 data that includes panchromatic and multispectral images. Spatial resolution of the panchromatic and multispectral images is 0.5 and 1.8 metres respectively. The considered area of study is depicted in Figure 5 shot from IRS-1A satellite. Software packages utilised for this study are HDFS Map Reduce with JDK Version 6.31 on Hadoop Yarn 2.3.0, which works as an open source framework facilitating distributed processing and storage applications. MATLAB technology is utilised as a complementary tool for development and maintenance of images during pre-processing phase. All the experiments are delivered on Linux based OS, Intel Core Processor i7 with RAM of 8GB.

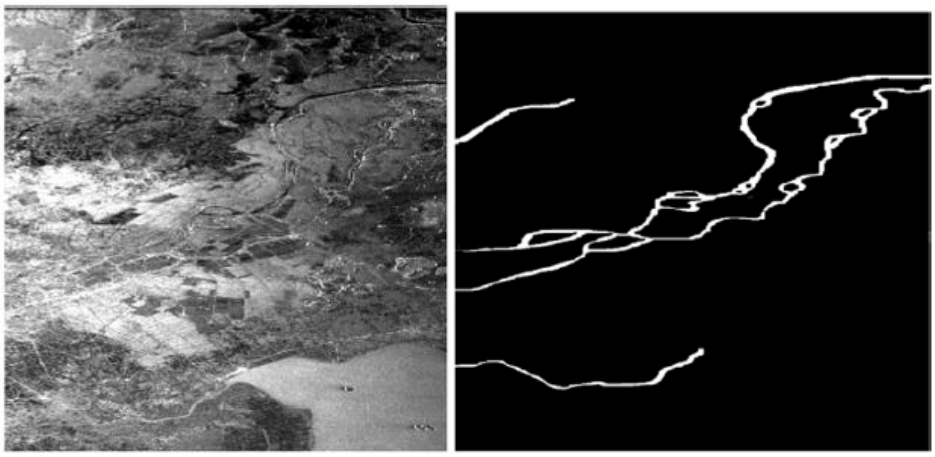

Figure 5 a) Original HRRS image b)Segmented image

The proposed framework has operated efficiently for different sets of all HRRS imagery which were employed for testing. Considering the input image is of small traditional image file, input imagery reaches wide scale that cluster can't timely process and manage it, formerly succeeding input has to be deferred, this will lead a gain in sharp boost of input time as depicted in figure 6. In disparity, well-formed big image will effectively release the constraint by merging all small images into one large image, then partition that large file into several blocks that are to be parallelly processed. Hence, this will be equal to processing multiple files where number of input images will not be a matter. According to this perspective, increasing the image scale has a tiny influence on efficiency of the input file of big image.

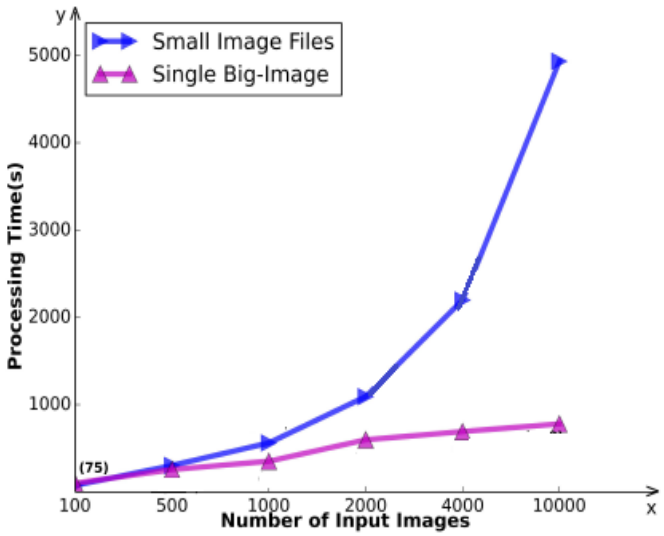

Figure 6 Scale of comparison between efficiency amongst Big-Image input and small image file input

A novel technique for feature extraction is implemented with Hadoop and MATLAB to implement divide and merge mechanism. The result of river feature extraction is presented in Figure 5(b). The proposed algorithm which uses E-HDFS with Map Reduce is far more efficient than implementation with MATLAB. Throughput of the application has been tested while growing the count of images in implementation of Hadoop. Initially the throughput of the work is quite less because the image size was very small and the image cannot support further division into chunks so as to store it on multiple nodes. There exist numerous switching contexts between Map and Reduce functions during its execution. If the given image capacity is very small, application will take more computing time during switches rather than on computation. Hence overall throughput will be lessened. The throughput can be increased if number of images is increased. For huge image datasets, the input data will be divided in form of chunks, every chunk will be

stored on different data nodes. All the changes will be processed parallelly as Hadoop map reduce programming nature itself is parallel. As a result the throughput will increase as depicted in figure 7.

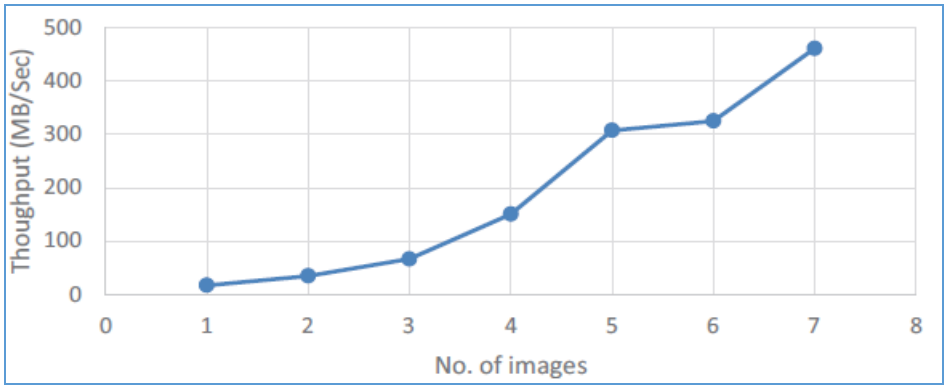

Figure 7 Increase of system throughput by increase in number of input satellite images

In this concern, we consider river as a continuous feature for feature extraction. The implementation model consists of different phases and units which include collecting, filtering, segmenting, computing, processing, merging as well interpretation. Proposed algorithm has proved its efficacy in theoretical testing by terms of complexity, practical testing by system throughput and average computing time. The proposed algorithm implementation using MapReduce provides good results when compared to simple Java implementation. Mapreduce implementation took <0.7s execution time average in order to process data of 1MB from most of IRS-1A, and SPOT remote sensing images.

Conclusions

We came up with E-HDFS interface as well a Map Reduce interface for remote sensing applications. HDFS's and Map Reduce's extended Library are used to analyse huge scale images along with higher level abstraction over image R/W. APIs provided for basic R/W image queries.

Numerous experimentations are accompanied on a sample set of 6 data sets along with a big image which varies 288 MB upto 9.1 GB. Numerous experimentations are directed to write and read images containing and not containing overlap using XHAMI. The yielded results are related with conventional Hadoop system, which depicted that, though the given E-HDFS suffers minimal R/W overheads because of data overlapping, application performance has increased linearly and also reduced the programming complexity in a significant way. Presently, the application is realized with the static images; in the nearest prospect, proposal will be to utilize the split function in a customized manner for

processing that allows spawning of more map functions. Conversely, addressing of challenges in organizing the sequence of executed map functions for aggregations will be done. We further have plans to implement bi-directional split in upcoming system, such that it addressed large scale canvas images. All API's of MapReduce can be formulated for several such Image processing and Computer vision working modules.

References

Althausen, J.D. (2002). *What remote sensing system should be used to collect the data?* In Manual of Geospatial Science and Technology, Edited by: Bossler, J.D, Jensen, J.R, McMaster, R.B and Rizos, C. 276–297. New York: Taylor and Francis.

Barnsley, M.J. (1999). *Digital remote sensing data and their characteristics.* In Geographical Information Systems: Principles, techniques, applications, and management, 2nd edn, Edited by: Longley, P, Good child, M, Maguire, D.J and Rhind, D.W. New York: John Wiley and Sons, 451–466.

Chen, B., Li, C., & Zhou, Z. (2018). *Classification of High Resolution Remote Sensing Images Based on PCA, HSV and Texture Feature.* Key Laboratory of Quantitative Remote Sensing Information Technology, Chinese Academy of Sciences, Beijing, China.

Chen, M., Su, W., Li, L., Zhang, C., Yue, A., & Li, H. (2009). Comparison of pixel-based and object-oriented knowledge-based classification methods using SPOT5 imagery. *WSEAS Transactions on Information Science and Applications, 3*(6), 477-489.

Estes, J.E., & Loveland, T.R. (1999). *Characteristics, sources, and management of remotely-sensed data. Geographical Information Systems: Principles, Techniques, Applications, and Management,* 2nd edn, Edited by: Longley, P, Goodchild, M, Maguire, D.J and Rhind, D.W., New York: John Wiley and Sons. 667-675.

Lefsky, M.A., & Cohen, W.B. (2003). *Selection of remotely sensed data.* In Remote Sensing of Forest Environments: Concepts and case studies, Edited by: Wulder, M. A and Franklin, S.E. 13–46. Boston: Kluwer Academic Publishers.

Li, J., He, L., Dai, J., & Li, J. (2006). Extract enclosure culture in lakes based on remote sensing image texture information. *Journal of Lake Sciences*, *18*(4), 337-342.

Yewei, L., Qiangzi, L., Du Xin, W.H., & Jilei, L. (2015). A method of coastal aquaculture area automatic extraction with high spatial resolution images. *Remote Sensing Technology and Application*, *30*(3), 486-494.

Nagaraju, T., & Ch. Suneetha, (2019). Distributed Framework for Processing High-Resolution Remote Sensing Images. *International Journal of Engineering and Advanced Technology (IJEAT), 9*(1), 4287-4292.

Phinn, S.R. (1998). A framework for selecting appropriate remotely sensed data dimensions for environmental monitoring and management. *International Journal of Remote Sensing, 19*(17), 3457-3463.

Liu, P. (2015). A survey of remote-sensing big data. *Frontiers in Environmental Science, 3,* 45.

Qiu, F., & Jensen, J.R. (2004). Opening the black box of neural networks for remote sensing image classification. *International Journal of Remote Sensing*, *25*(9), 1749-1768.

Murala, S., Maheshwari, R.P., & Balasubramanian, R. (2012). Local tetra patterns: a new feature descriptor for content-based image retrieval. *IEEE transactions on image processing*, *21*(5), 2874-2886.

Wang, C., Xu, A., & Li, X. (2018). Supervised classification high-resolution remote-sensing image based on interval type-2 fuzzy membership function. *Remote Sensing*, *10*(5), 710.

Ma, Y., Wu, H., Wang, L., Huang, B., Ranjan, R., Zomaya, A., & Jie, W. (2015). Remote sensing big data computing: Challenges and opportunities. *Future Generation Computer Systems*, *51*, 47-60.

Zhang, S., Jianfei, C.H.E.N., & Jianzhou, G.O.N.G. (2016). Object-oriented classification based on C5. 0 algorithm. *Science of Surveying and Mapping*, *41*(6), 117-121.

Zhao, L., Meng, L., Zhang, Y., Feng, W., & Zhang, J. (2016). Research on joint spectral and texture features svm tidal classification and extraction algorithm. *Engineering of Surveying and Mapping*, *25*(1), 43-46.

Keshavarz, H. (2020). Web self-efficacy: A psychological prerequisite for web literacy. *Webology*, *17*(1), 81-98.