Journal of Cardiovascular Disease Research

ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015

A REAL TIME EFFECTIVE STORAGE SYSTEM IN HADOOP BASED MEDICAL DATA ARCHITECTURE FOR DYNAMIC FILE SELECTION VIA BIGDATA ANALYTICS

Y. Sri Lalitha

Associate Professor, Gokaraju Rangaraju Institute of Engineering and Technology, Email: srilalitham.y@gmail.com

Abstract

Nowadays modern technologies have been depending on big data applications related to data storage; it is an important attribute for improving the storage of applications. The various surveys related to big-data applications simplify the file storage system. It is clearly observed that effective file handling mechanism do not recognize according to survey. The existed methods store the unstructured and structured files with less level of sensitivity. Therefore, an advanced file handling methodology is necessary for big-data analytics. In this research work map secure reduce layers (MSR) with elastic net regression (ENR) model has implementing. The hadoop distribute file system (HDFS) is taken as file handling platform; in this MSR-ENR techniques are verified. The proposed MSR-ENR can handle the all type of memory files with various extensions (.dox, .docs, .pdf, .rar and etc.). At final calculate the performance measures such as processing time, sensitivity, accuracy, throughput and recall. This proposed MSR-ENR method outperforms the simulation results and challenging the current technologies. The performance measures such as accuracy around 95%, sensitivity 98%, specificity 99%, Recall 95%, throughput 92% and processing time 420ms are achieved, and these results are challenging the present technologies. Dataset similar to enhance the processing of patient information, Electronic Medical Records (EMR) generated by multiple medical equipment and mobile apps will be introduced into MongoDB utilizing the Hadoop framework and improvised processing approach.

Keywords: File storage system, Hadoop, elastic net regression, MSR

1. INTRODUCTION

The modern model of big data does not specify any level of sensitivity of data for both structured and unstructured. So, need to incorporate Small file Size handling problem where the risk of personal information is statistically reduced. This research work introduced a Small file Size handling problem solution in HDFS System. The proposed methodology called as MSR model. The benefit of this work is data sharing privacy and the challenge of managing little files, as well as the privacy utility trade-off for data mining. HDFS's Small File Problem: Storing a large number of small files that are much smaller than the block size is impossible for HDFS to handle efficiently. Reading small files necessitates numerous searches and bouncing from data node to data node, resulting in inefficient data processing.

**HDFS -- Hadoop Distributed File System

**MSR -- Map Secured Reduce layer.

Today computing era most concentrates on special tools such as cloud computing, big data and data mining. In every domain efficient file handling mechanism is necessary, without any effective file

ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015

storage system the applications cannot perform it functionality properly. The hadoop file managing distributed system solves many encryption and filtration processes. The incoming files such as file type (text, pdf, document and video.avi) are handling effectively. But, necessary improvements are making the system interpretative to current technologies. These all limitations are overcome by MNR-ENR technique, >100kb file has directly rejected by normal storage system and transferred to HDFS. In order to secure the various data in cloud computing via bigdata applications handled by proposed methodology. In HDFS filtration and encryption mechanisms are handled by various software defined networks. This HDFS differentiate the file storage for both small files and large files. The source and destination users are can easily handle the available files in HDFS-ENR.

In bigdata various vast volumes of data has been organized for storing cloud or any server. Therefore traditional storage system or optimization algorithm is necessary to overcome this type of imitations [1]. The feature generation rapidly utilizes the bigdata applications, but these are facing many architectural problems for storing of large and small size of files. The semi structured and unstructured files are not properly utilized by present technologies. The prerequisite of bigdata applications are mainly depending upon file storage system. In this investigation conventional techniques are utilized to solve the following functionalities of bigdata such as, Jason XML and text email data. The text files, multimedia data and video files have been handled by conventional bigdata models. But, these are filed due to lack of proper file storage system [2]. After the conventional era the test art related to hadoop framework had been comes under file storage mechanism. It is an effective tool to store the large type of files and save the storage recovery space and processing time [3] (seen in figure 1).

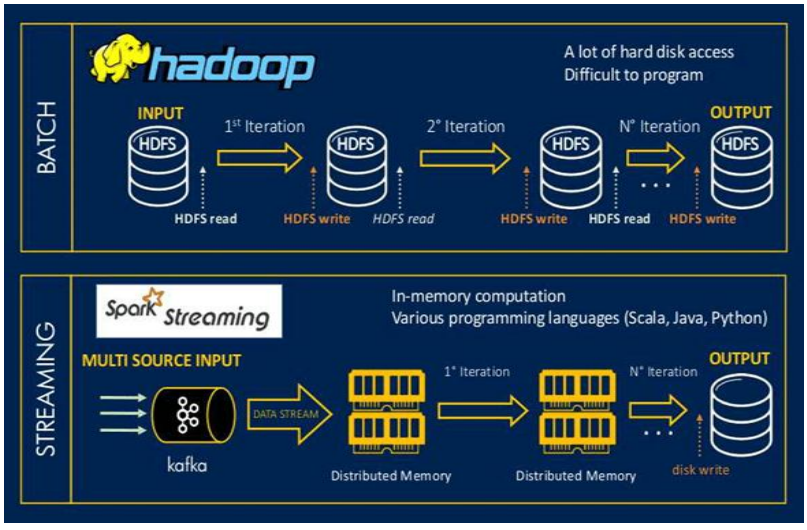


Figure 1 Example hadoop system.

For HDFS file reading and writing, this architecture observes streaming and batch files. Through iterations, the memory location modified in the HDFS system can accommodate distributed memory. These iterations begin with 1, 2,... The languages java, python, and HTML are used to implement disc writing and reading functionality. It is a Hadoop design that can manage a wide range of file storage issues using traditional methods. The memory computations and functionalities vary in selected program language. The over fitting problems and training datasets are handled based on regularization of hadoop system. It is an complex linear model and it can be reduces the operations by using feature extraction and regularization. But, these functionalities are failed due to unnecessary filtration and data access

ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015

points. In this work our proposed elastic net regression model with ridge regression and lasso net regression solve the above limitation.

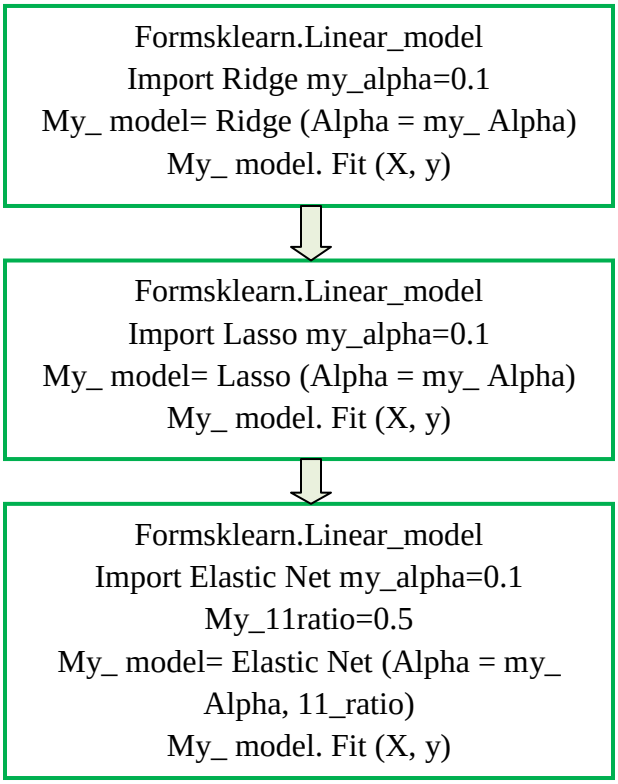


Figure 2 HDFS-ENR

Figure 2 illustrates the elastic net regression model using lasso and ridge regression, which effectively solves various file storage challenges. But, have significant limits. The alpha module uses zero coefficient analysis to tackle the file shrinking problem. This approach does not address the current concerns when it comes to linear and non-linear models. Coming too real to practical applications the cross validation can estimate the exact problem in this model. Necessary improvements make the system efficient, this can possible by MSR-ENR. The structured data is an specified format associated with database, it can easily handle by present mechanisms. Coming to unstructured data more than 95% of content generated by various social networks, these are websites and log files. These files cannot handle by present mechanisms [4 5].

1.1 Literature Survey

In big data file storage management is an critical parameter, it is mainly depending on following elements. Such as data analysis, data acquisition, data diagnosis, data storage and data usage. In data acquisition model select the all type of data models, they are structured and unstructured, event processing, real-time and data streams. In data analysis semantic search, machine learning, data discovery and information extraction processes are analyzed. The data diagnosis and storage are the key parameters; these are handled data quality, data validation, human interaction, memory adjustment, and SQL cloud storage portioning and privacy statement. At final data usage can simulate the decision

ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015

support from designed architecture. This entire chain adjusts the storage system and makes the file storage location efficient.

Reference No	Technique	Key points
[6]	Hadoop distributed file system	In this work an yahoo text classification has been done using HDFS file storage system. A 25petabytes memory is adjusted using this architecture and results show that designed method is more improved. **create file 5700 **delete file 20800 **Rename file 8400
[7]	Block replication model	The HDFS data storage system can interpret the space reclamation step. In this crash dictionary have deleted the files depending on nodal technique. Decrease replication factor plays an important role in name node structure. It is handle the free space in the structure between source and destination.
[8]	Map reduce configuration_ Yahoo press	Map reducing mechanism is an relational database model, it is eliminate the over fit and structural fit problems. In this technique giga bytes are converts into peta bytes read and write mechanism into write once read many times mechanism. Because of this map reduce can allot the more space compared to another conventional model.
[9]	Quant cast file storage system	In this research IoT related big data analysisand its normalization factors are analyzed. Hadoop make the system efficient in terms of performance and streaming of files.
[10]	Hadoop enhancement of HDFS interruption	In this work time delay handling, solution quality, comparison of reading and writing are estimated with following HDFS technique. H+S+B method achieves significant improvement compared to conventional model.

The above all survey explains about distributed file system related to file storage, when the technology moving forward storage system needs to be update. Therefore designed systems are not suitable for large data files and bulk file processing. Because of many limitations the existed methods we are moving to MSR-ENR methods.

ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015

1.2 Objective

To provide a mechanism for solving the "Small size file handling problem" present in the existing framework

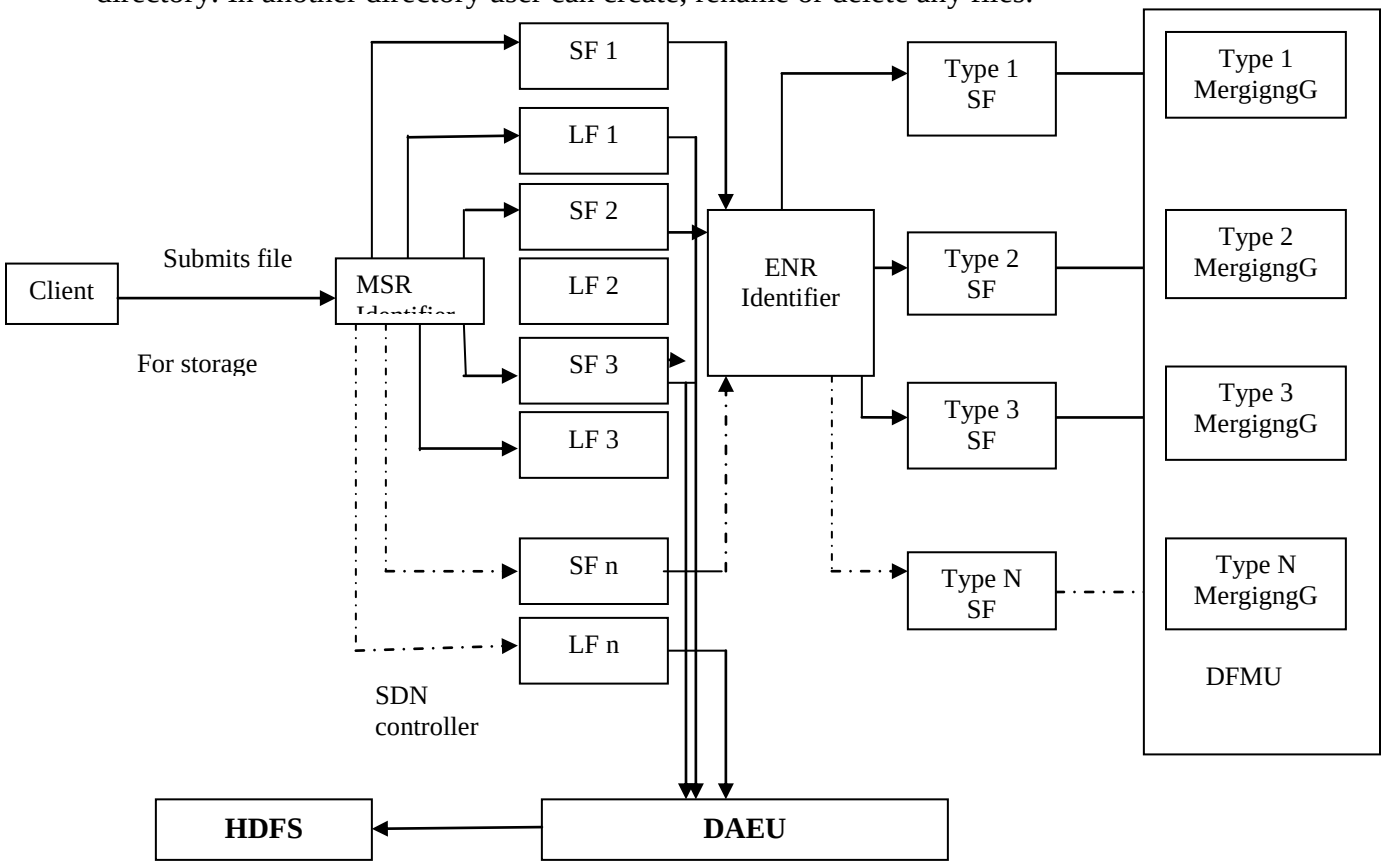
2. METHODOLOGY

In this research work the proposed methodology MSR-ENR has been implemented on java, python and HTML programming languages on core python software. The HDFS is a prominent network it organizing the file managing system when the large data identified on any platform such as cloud, server and any storage space. The HDFS is a advanced file handling system along with MSR-ENR can handle hundreds of applications. Coming to streaming data access t is an interactive data handling mechanism easily runs on HDFS semantic key area. A large datasets are identified on clusters can automatically MSR-ENR converts less bytes of datasets. It is an tuned process and supporting the large files even though bandwidth availability is very less.

The clinical standard files from hospitals and practises that are relevant to health and patients are the input for the suggested framework. Big data processing and storage will be a part of this strategy. This level takes into account the importance of document volume or size as well as the complexity of the files or records.

2.1 Name Node and Data Node

On the client server, HDFS is a self-architecture with a single name node and data nodes. These platform run on HDFS area when the file system consist of greater than the threshold size. Read and write requests, as well as creation, deletion, and replication, are handled by the data nodes. The data node and name node software's are runs on windows machine supported by java-8u-121. The proposed MSR consists of single name node simplifier architecture; it is designed user data cannot flow through the user name node area. The HDFS is an file organization mechanism, it can create dictionaries, files on existing platforms. Anyone can create or delete files using HDFS system from one directory to another directory. In another directory user can create, rename or delete any files.



ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015

Figure 3 Proposed HDFS-MSR-ENR

The above figure 3 clearly explains about proposed methodology block diagram. In this at primary stage we can create the HDFS platform using core java tool. The input is applied as any extension file (.pdf, .docx, doc and etc..), this file analyzed by MSR-HDFS file managing system. If the uploaded file contains more than 100kb then automatically HDFS server encoding the data and saved in hadoop server. With the help of MSR differentiate the small size and large size files for further decision. After MSR-ENR operation hadoop function handle the entire mechanism.

2.2 MSR (MAP secure reduce layer)

MSR is introduced for security and privacy purpose, this layer working between HDFS and ENR. This concept benefits the data sharing knowledge mining applications. The MSR challenges the privacy and security of selected file in the HDFS system, because of this scalability, maintenance and privacy utilizations are attained efficiently. This tool helps the traditional bigdata applications like sensitivity of data selection such as unstructured and semi structured. When increasing the data volumes through social networks and mobile devises, this MSR easily handle it.

MSR applications:

A. Distributed applications maintained securely
B. Non-relational data security.
C. Transition logistic data handling
D. Input and output validation and filtering.
E. Real-time security maintenance.
F. Data accessibility.

The many organizations are facing above type of applications without maintenance by hadoop system. Due to MSR-hadoop it can easily handle the map reducing layers for high volumes of files. This concept is helpful for bigdata architectures, cloud computing and data mining applications. The private cloud and public cloud are simultaneously access the large files with some interaction. But, sometimes these applications are hold by struck at errors; this problem is easily handling the MSR mechanism.

Algorithm: MAP secure reduce layer

Input: any document or file (any type of format)

Output: size decreased folder, encoded folder

Mapping

Step: 1 dividing

Step: 2 reading ()

Step: 3 tokenizing ()

Step: 4 covert to MAP secure reduce layer model

Reducing MAP secure reduce layer:

Step: 5 attaching the folder

Step: 6 calculating the memory

Step: 7 conclusions

Step: 8 stop the model

The above MSR algorithm partition in the files into various type of modules using HDFS software. If any file >100kb the MSR mapper identifies the file using read, write, tokenize parameter, using reduce

ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015

MSR option encoded the files into HDFS location. For this MSR operation 64bit operating system, 8 GB of ram and 1tb of memory elements are required. At secondary stage java, hadoop, executed HTML workstation has been arranged. This MSR calculating the memory loss, memory usage, CPU utilization and runtime mechanisms are utilized.

$$I = \sum_{i=0}^{n} \sum_{j=0}^{m} \frac{| upper_{ij} - lower_{ij}}{m.n \mid max_j - min_j} \qquad (1)$$

Using the minimum and maximum file sizes, equation 1 calculates the upper and lower bond characteristics. If the file is larger than 100 kilobytes, our system will immediately move it to the Hadoop network; otherwise, it will be stored on the same server. In MSR-HDFS, the above figure 4 is employed as a file merging strategy, and the files and their values are clearly modified by the proposed technique.

2.3 ENR (Elastic net regression):

It is a machine learning process can handle any type of applications with effective manner, the free encyclopedia; multimedia applications are large in size. These are handled by our proposed technique with less accuracy by using proposed non-linear methods. Therefore elastic net regression model with the help of linear methods L1 and L2 are over fit the problems faced by file handling system.

**L1= Lasso

**L2 = Ridge

The ENR is a machine learning model, it reduces the zero value coefficients for empirical file handling system.

$$\hat{\beta} = \text{argmin}\left\{\frac{1}{2}\sum_{i=1}^{N}\left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2 + \lambda\sum_{j=1}^{p}\left(\alpha|\beta_j| + (1-\alpha)\beta_j^2\right)\right\} \qquad (2)$$

In this equation2 calculate the argument of various files with the help of λ coefficient, here λ is source penalty coefficient and allowed the number of α&β values. If α = 0 it is a ridge type function, else α = 1 tense to lasso function. The entire equation depending on regression variables correlated with multi linearity results.

$$TL(\hat{f}, \lambda) = \frac{1}{N}\sum_{i=1}^{N} L\left(y_i, \hat{f}^{-k(i)}(x_i, \lambda)\right) \qquad (3)$$

The equation 3 clearly explains about training set model of elastic net regression, using this equation estimate the α & λ value. When this values less than cross validation then elastic net regression handle the function, otherwise quit the training set.

** TL = Tune length

ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015

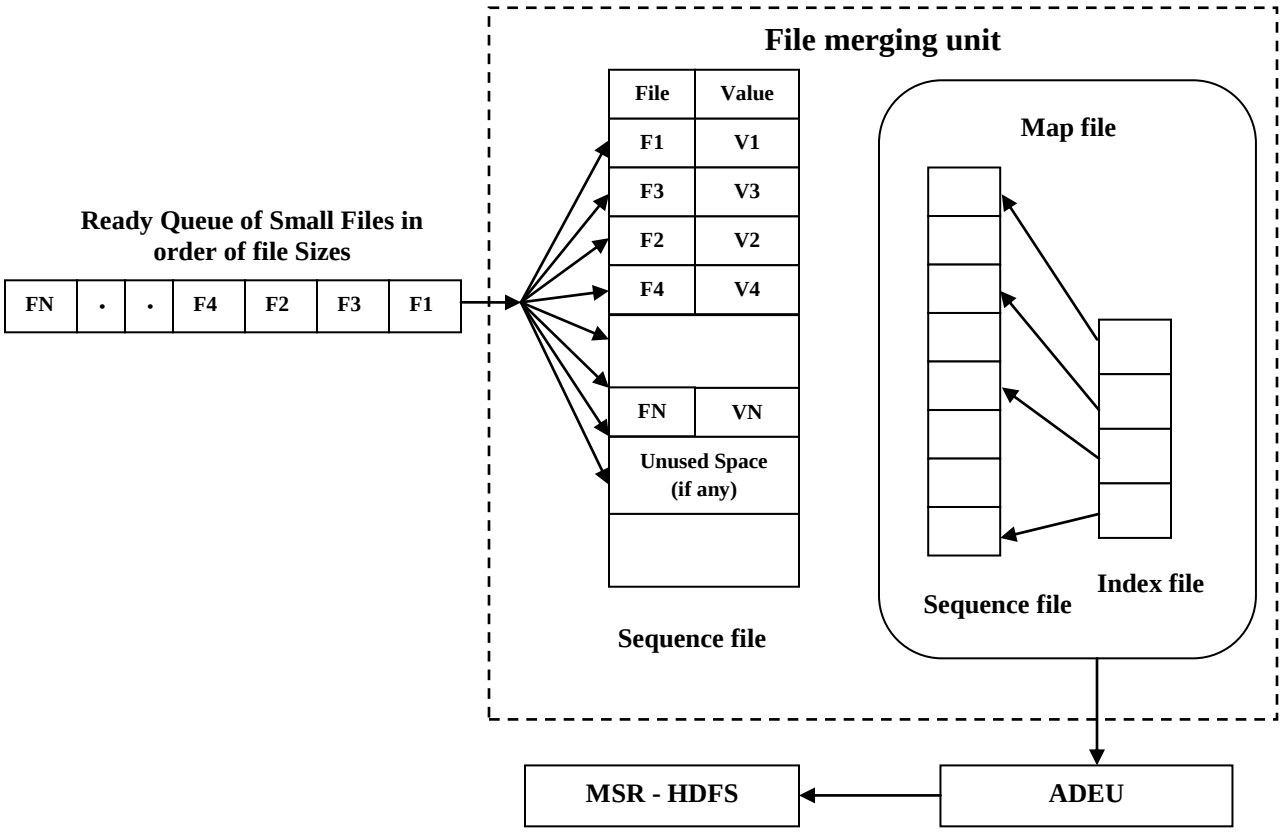


Figure 4 Architecture of automatic File Merging Unit

$$\log\frac{P(G=1|X=x)}{P(G=0|X=x)} = \beta_0 + \beta^T x = Tt \qquad (4)$$

Equation 4 explains about different performance metrics related to root mean square error, efficiency and recall.

$$\max\{\beta_{0\in}R, \beta_{\in}R^p\}\left[\frac{1}{N}\sum_{i=1}^{N}\log P(G_i|x_i) - \lambda\sum_{j=1}^{p}\left(\alpha|\beta_j| + (1-\alpha)\beta_j^2/2\right)\right] \qquad (5)$$

The above equation 5 is calculating the maximum file size location, using this differentiate the file storage space (server or hadoop).

ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015

In this project ENR-MSR is describing concept to merge short file in HADOOP distributed storage. Sometime small file storage will consume lots of memory to store chunks details metadata and author is proposing dynamic merging concept where small files less than 100KB will be merging with appropriate file type and due to this merging we can save chunks details memory storage. In propose method will check file size and if size less than 100KB then we will encrypt file data and merge with other files in HADOOP. All small files are merging with sequence file. If file size is greater than 100KB then file will be encrypted directly and send to HADOOP for storage

3. Materials and methods

Install python and java in your system's C directory & configure the path and python home variables.

Place the hadoop path in your system's C directory, set the path to C:/hadoop/bin, and then set the Hadoop_home to

Variable name = HADOOP_HOME

Variable value = C:/hadoop

Follow below instructions to run hadoop server

Go cd/hadoop/bin and execute command

HDFS_name_node -format

Now set position in CMD

C:/hadoop/sbin and run command start-dfs

Above facility will start Hadoop path

Then open browser and enter url http://127.0.0.1:50070/dfshealth.jsp to get below hadoop home page

Create a Python folder in your system's C directory, then place the "Dynamic Merging" folder in C:/Python. Figure 5 depicts the Name node point in relation to the local host system.

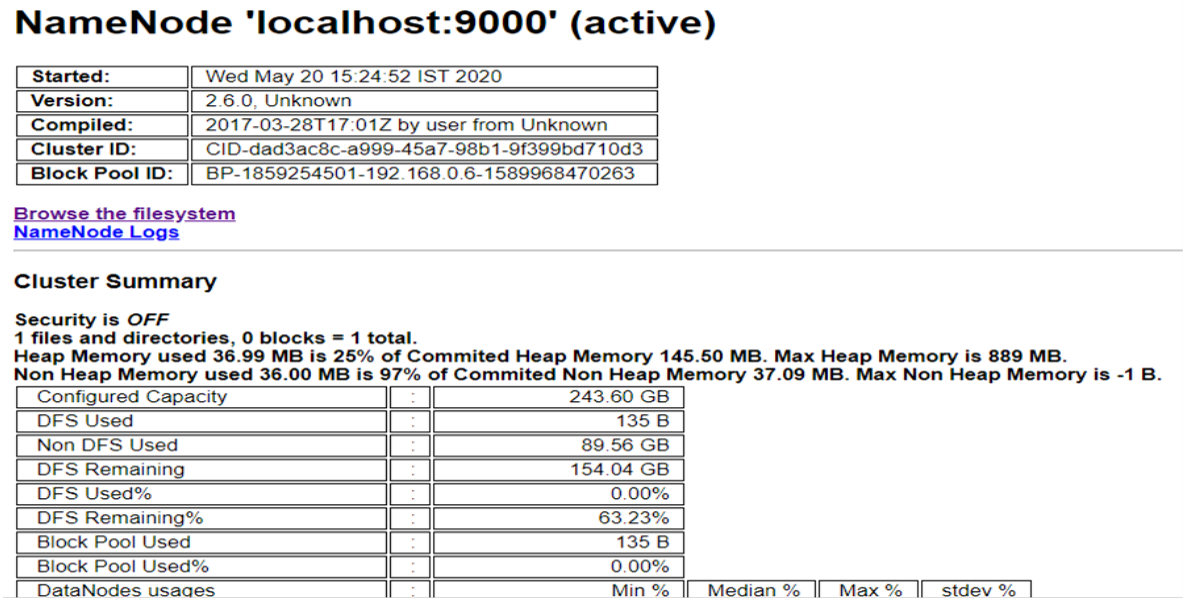


Figure 5 Name node allocation

ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015

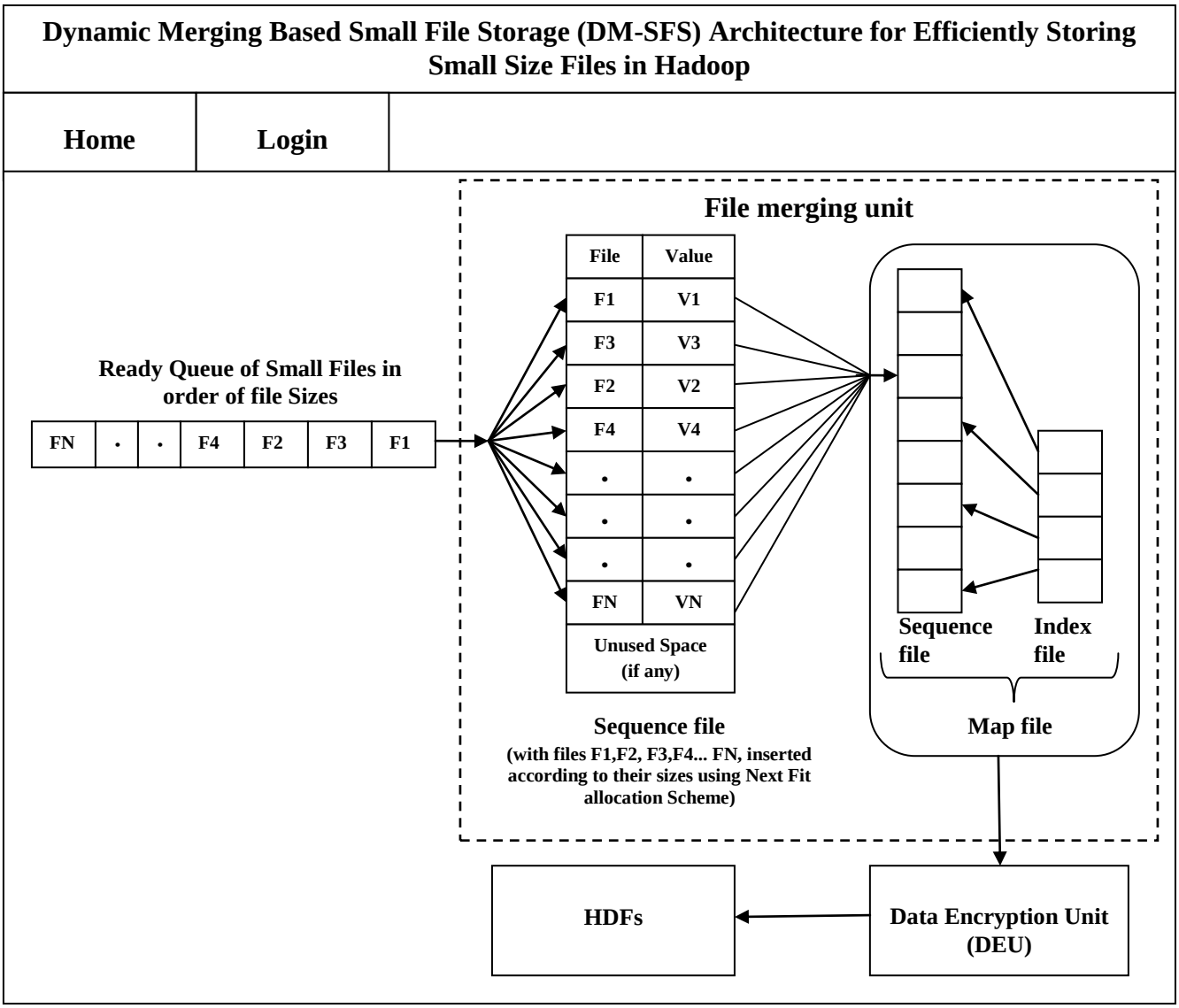


Figure 6 HDFS-MSR-ESR

The automatic file managing unit for MSR-ENR in the HDFS system is depicted in figure.6. With the elastic net regression model, it can successfully tackle the challenges. It's a machine learning engine, and the entire analysis process is handled by cluster memory. Click the Login link in the above panel to go to the below screen, where users can log in to the Hadoop environment using their credentials.

To access the below screen, enter the username "admin" and the password "admin" on the above screen.

ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015

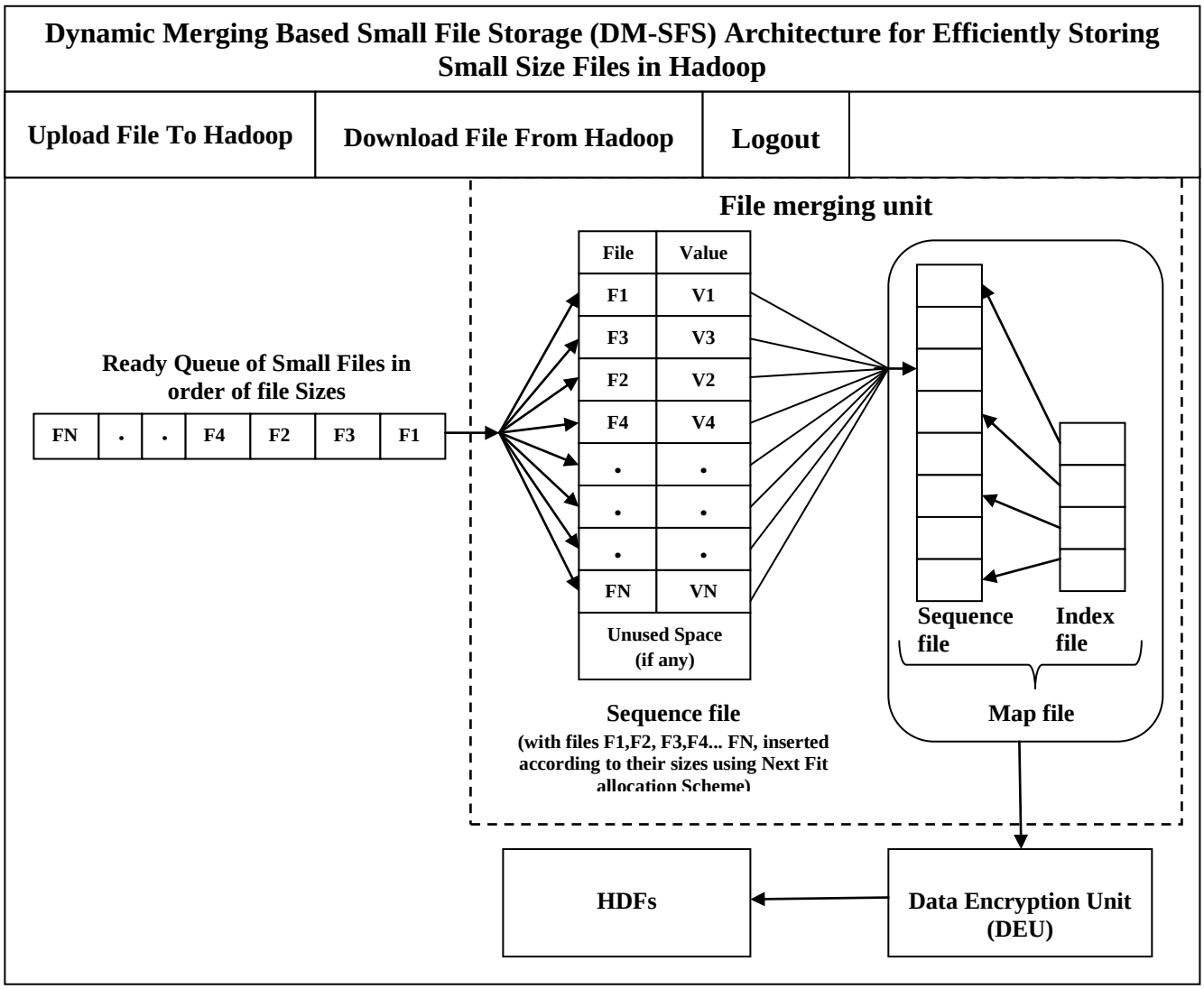


Figure 7 File upload

To upload a file to Hadoop, go to the previous screen and click the ‘Upload File to Hadoop' link; figure 7 explains the upload and verification options of ENR technology. When the file size exceeds 100kb after uploading, the file is immediately transferred to HDFS. For reduced storage activation, the encoding operation is performed on the same file.

ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015

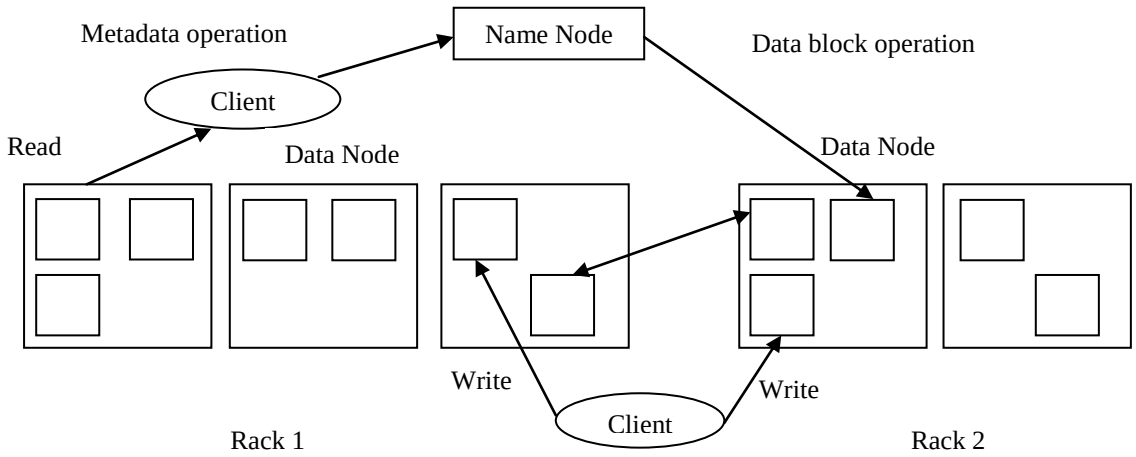


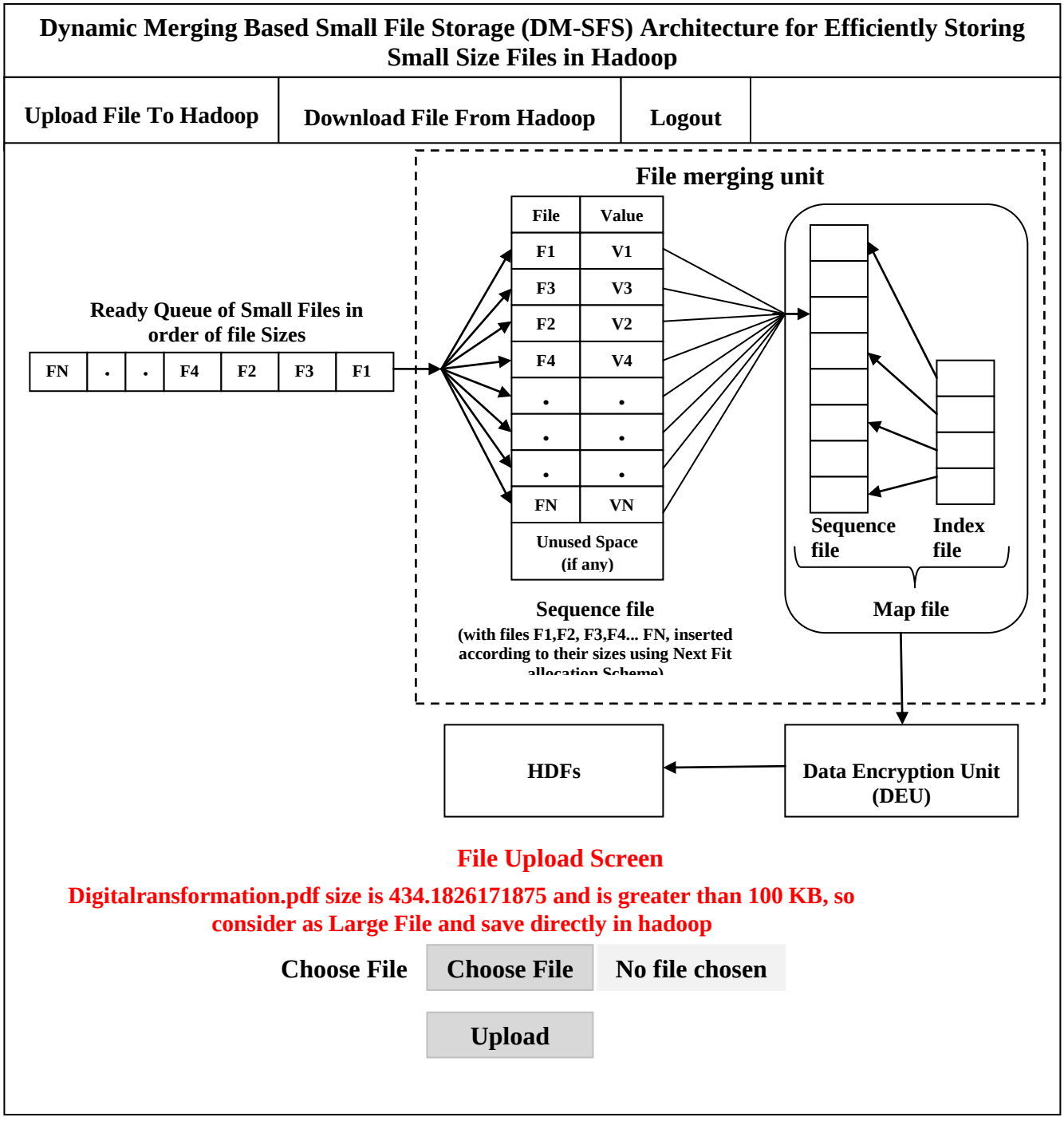
Figure 8 data node and name nodes

On the previous screen, we uploaded one pdf and then clicked on the ‘Open' button before clicking on the ‘Upload' button to store the file at Hadoop.
The process of name nodes and data nodes is explained in detail in Figure 8, and it is noted that block activities are separated. For HDFS function verification, the data and name nodes are gathered.

4. RESULTS AND DISCUSSION

The pdf file size is 436 KB, which is larger than 100.00 KB, so it will be considered a big file and stored straight in Hadoop, with the file being encrypted using the two fish algorithm before storage. Now, using the URL ‘http://127.0.0.1:50070/dfshealth.jsp', you may view the file on the Hadoop server.

ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015



ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015

Figure 9 Results of file

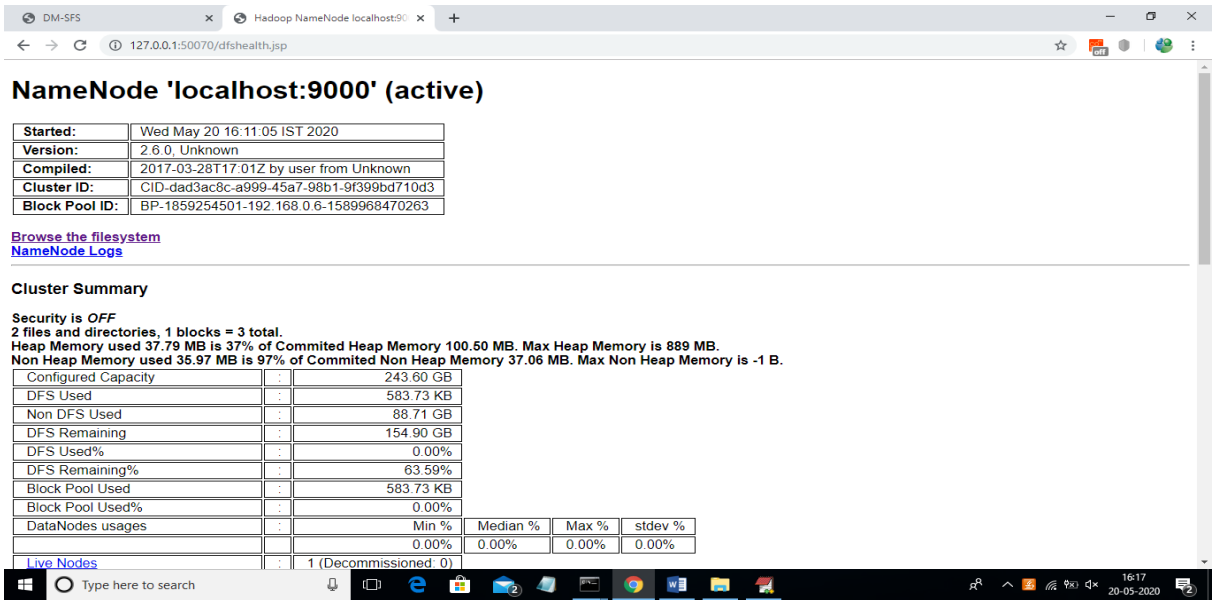


Figure 10 Name node local host

Click the ‘Browse the File System' option in the above page to access files in the below screen, as shown in figure 9, which explain the date, time, cluster id, & block pool id. This type of data can help the system be more efficient in the future when it comes to file management.

On the previous screen, it could see that the file and folder name that can be uploaded from the request is stored at the Hadoop local server, & that we can now connect on that folder with file name to see its gratified as encoded in figure 10.

In the following figure, it can see that the PDF is encoded. Now, return to the request and save a little file. In Hadoop, the little files would be combining into a individual sequence file.

The multiclass.txt file on the previous screen is a small file that will be merged into the sequence file, and then we will submit another short file for further action recognition.

We're uploading another txt file on the above page, and the storage result shows that it's a little file.

In the following screen, cloud answers.txt is a little file that will be merged into the sequence file, and now we can view all of the store files depicted on the Hadoop server.

We can see one pdf file and one sequence file in the above screen, and we actually uploaded one pdf file and two little txt files, however the small txt files were combined into the sequence. We only got one file for two little files because we used a pkl file. Return to the programme and click the ‘Get File from Hadoop' link to download those files.

ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015

Table 1. Results comparison

Methods	Accuracy	Sensitivity	Specificity	Recall	Through Put
Misclassification[15]	68.23	59.67	87.39	83.42	74.90
HDFS[16]	72.83	62.07	82.77	74.58	77.10
H+S [17]	76.75	63.00	86.02	81.45	80.45
HSC [17]	79.060	73.230	83.340	85.78	84.68
Proposed	92.740	95.870	98.650	92.560	94.48

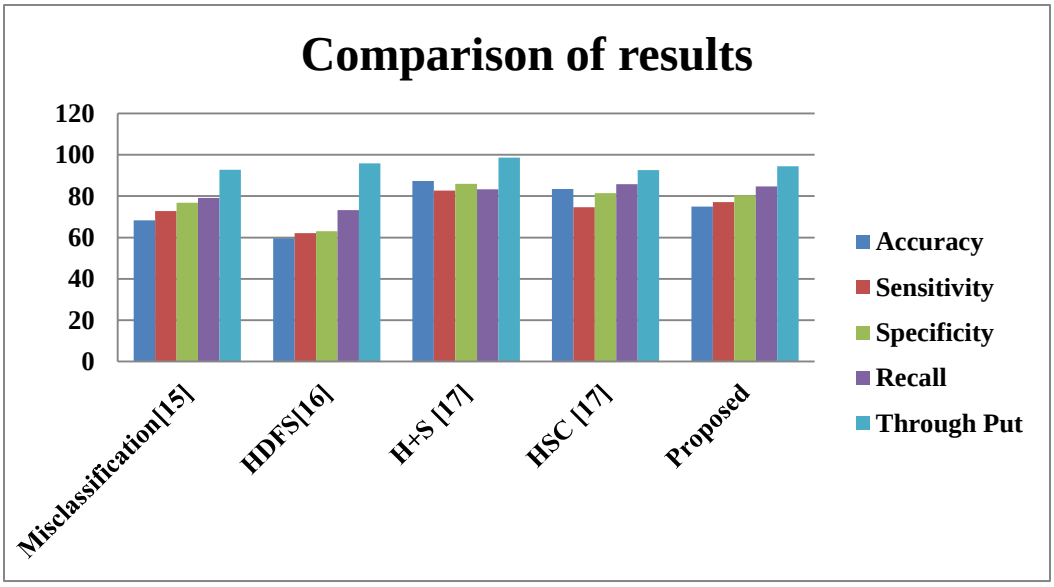


Figure 11 Comparison of results

Figure 11 and Table 1 describes all existing performance metrics as well as the suggested approach MSR+ENR, which clearly explains how the proposed method provides more improvement.

Table 2. Time required for processing

Processing Time	
methods	Processing Time in ms
Misclassification[15]	6200
HDFS[16]	5600
H+S [17]	600
HSC [17]	900
Proposed	460

ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015

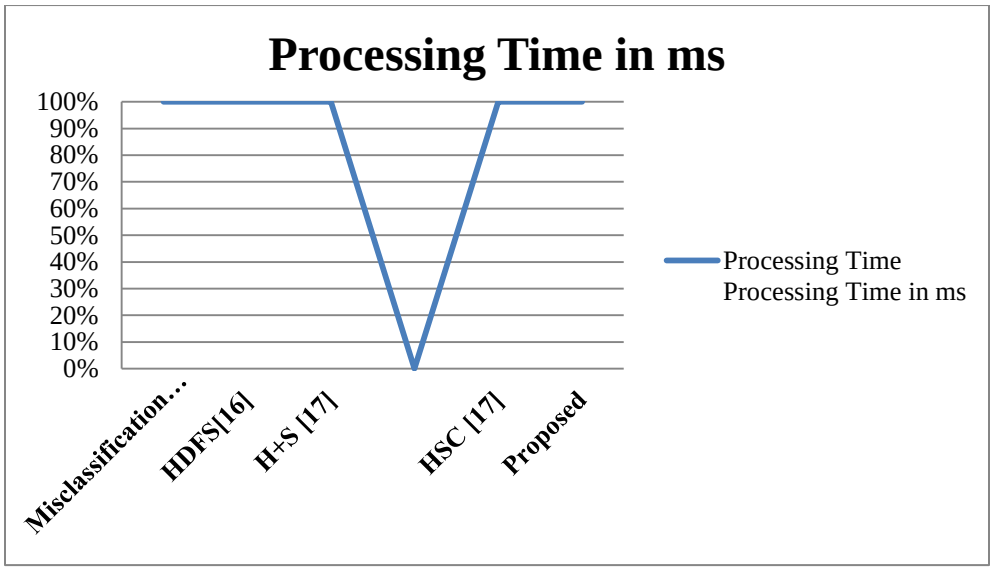


Figure 12 Processing time

The processing time allocation to each approach is explained in Figure 12 and Table 2, and the suggested method MSR+ENR achieves higher improvement than existing methods.

5. CONCLUSION

In this research work implemented an advanced file handling method i.e. HDFS+ENR+MSR. It is a modern technology dealing the Name-node and Data-node problems related to files. The problem identification from survey design an objective, it is solve the basis of Java-8u-121 extension, python 3.7 software. This implemented design is dealing the Large and small size files effectively. The data storage approach is mainly divided into two type, if the selected file more than the 100kB then automatically it is recognize as small file and store into general area else save into Hadoop server at merge option. In comparison to existing approaches, the proposed MSR+ENR achieve higher improvement in this study. The performance measures such as accuracy around 95%, sensitivity 98%, specificity 99%, Recall 95%, throughput 92% and processing time 420ms are achieved, and these results are challenging the present technologies.

References:

[1] Halevi, G., & Moed, H. (2012). The evolution of big data as a research and scientific topic: Overview of the literature. Research trends, 30(1), 3-6.

[2] Li, G., Ooi, B. C., Feng, J., Wang, J., & Zhou, L. (2008, June). Ease: an effective 3-in-1 keyword search method for unstructured, semi-structured and structured data. In Proceedings of the 2008 ACM SIGMOD international conference on Management of data (pp. 903-914).

[3] Zikopoulos, P., & Eaton, C. (2011). Understanding big data: Analytics for enterprise class hadoop and streaming data. McGraw-Hill Osborne Media.

[4] Federal Big Data Commission. (2012). Demystifying big data: A practical guide to transforming the business of government. Technical report, TechAmerica Foundation.

ISSN:0975 -3583,0976-2833 VOL06, ISSUE 04, 2015

[5] Shvachko, K., Kuang, H., Radia, S., & Chansler, R. (2010, May). The hadoop distributed file system. In 2010 IEEE 26th symposium on mass storage systems and technologies (MSST) (pp. 1-10). Ieee.

[6] Borthakur, D. (2008). HDFS architecture guide. Hadoop apache project, 53(1-13), 2.

[7] White, T. (2012). Hadoop: The definitive guide. " O'Reilly Media, Inc.".

[8] Hua, X., Wu, H., Li, Z., & Ren, S. (2014). Enhancing throughput of the Hadoop Distributed File System for interaction-intensive tasks. Journal of Parallel and Distributed Computing, 74(8), 2770-2779.

[9] Jia, M., Xu, H., Wang, J., Bai, Y., Liu, B., & Wang, J. (2015). Handling big data of online social networks on a small machine. Computational Social Networks, 2(1), 1-12.

[10] Kreutz, D., Ramos, F. M., Verissimo, P. E., Rothenberg, C. E., Azodolmolky, S., & Uhlig, S. (2014). Software-defined networking: A comprehensive survey. Proceedings of the IEEE, 103(1), 14-76.

[11] Poggi, N., Carrera, D., Call, A., Mendoza, S., Becerra, Y., Torres, J., ... & Blakeley, J. (2014, October). ALOJA: A systematic study of hadoop deployment variables to enable automated characterization of cost-effectiveness. In 2014 IEEE International Conference on Big Data (Big Data) (pp. 905-913). IEEE.

[12] Hua, X., Wu, H., Li, Z., & Ren, S. (2014). Enhancing throughput of the Hadoop Distributed File System for interaction-intensive tasks. Journal of Parallel and Distributed Computing, 74(8), 2770-2779.