A Big Data Framework for Satellite Images Processing using Apache Hadoop and RasterFrames: A Case Study of Surface Water Extraction in Phu Tho, Viet Nam

Dung Nguyen[1], Hong Anh Le[2]
Faculty of Information Technology
Hanoi University of Mining and Geology
18 Pho Vien, Bac Tu Liem, Ha Noi, Viet Nam

***Abstract*—Earth data, collected from many sources such as remote sensing imagery, social media, and sensors, are growing tremendously. Among them, satellite imagery which play an important roles for monitoring environment and natural changes are increased exponentially in term of both volume and speed. This paper introduces an approach to managing and analyzing such data sources based on Apache Hadoop and RasterFrames. First, it presents the architecture and the general flow of the proposed distributed framework. Based on this, we can implement and perform efficient computations on a big data in parallel without moving data to the center computer which might lead to network congestion. Finally, the paper presents a case study that analyzes the water surface of a Vietnam region using the proposed platform.**

***Keywords*—Satellite imagery; big data; water surface; Apache Hadoop; RasterFrames**

I. Introduction

Data generated have been increased exponentially in recent years. The amount of data in recent two years is equal the to amount data which has been generated before. Data format also varies from structured to unstructured types. The format includes database in relational database systems, while the later might be video, images, or log files. Hence, 'big data' term appeared with three Vs such as *volume*, *velocity*, and *variety*. *Volume* refers to the amount of data generated, *Velocity* mentions the generation speed of the data, and *Variety* indicates the diversity of the data source.

One of most frequent generated sources is from earth observation. The big earth data consists of all data related to the earth including sensors, satellite images. Among them, satellite imaginary volume amount grows rapidly with advancement of technologies. For instance, in 2019, Sentinel (Sentinel-1, Sentinel-2, Sentinel-3), Landsat-7, Landsat-8, MODIS produce around 5 PB data [1], [2], [3]. For this reason, some solutions are proposed for building the platform for storing, managing, and processing EO data such as Google Earth Engine (GEE) [4], Sentinel Hub [5], Open Data Cube (ODC) [6], OpenEO [7], etc.. GEE is a platform that provides petabytes of satellite imagery and large-scale applicability for analysis. Sentinel Hub is an cloud-based engine for processing multi petabytes of satellite data. It allows users easily browse, visualize, and analyze Earth observation imagery. ODC is an open source project of geospatial data management and analysis. It provides facilities to work with raster data using Python libraries and PostgreSQL. These platform provide facilities to store and process satellite images both in commercial and open source solutions. In fact, however, there are some restricted policies of the organization or government that do not allow to store EO data in the cloud for example high-resolution satellite imagery from military areas or restricted areas.

Analyzing these data sources plays an important role in monitoring natural environment changes for instance land cover, surface water, body water, etc. Among them, surface water information is significant factor indicating the urbanization of an area and the management of water resources in the developing countries such as VietNam. Remote sensing and GIS techniques have been exploited to effectively analyze these changes. Recently, many research work proposed to use deep learning techniques and satellite imagery for surface water extraction and detection [8], [9], [10]. It, however, raises one issue regarding the huge volume of data to store and processing in a long time. For this reason, a simple and low-cost solution for this problem is desirable.

In this paper, we propose a framework for processing big satellite imagery data based on HDFS and Rasterframes. It allows to flexibly store satellite images in the distributed manner and perform manipulation on the data in parallel without moving data to the center. The contribution of this paper are (i) proposing a salable, open source, and low-cost framework for big satellite imaginary processing ; (ii) implementing and performing surface water analysis using sentinel-2 images for PhuTho area, a Northern province of VietNam.

The paper is structured as followed. Section II briefly give the introduction of Apache Hadoop platform and RasterFrames which are used as the basis of the proposed framework. Section III summarizes the related work. In the next section, the proposed framework is introduced in detail with its architecture and main flows. A case study of surface water extraction based on this framework is given in Section V-A. Finally, Section VI concludes the paper and presents the future work.

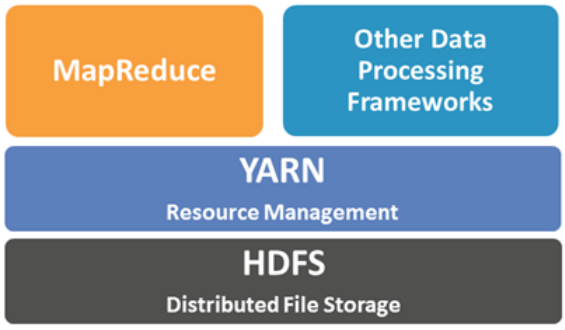

Fig. 1. Hadoop V2.0 Architecture

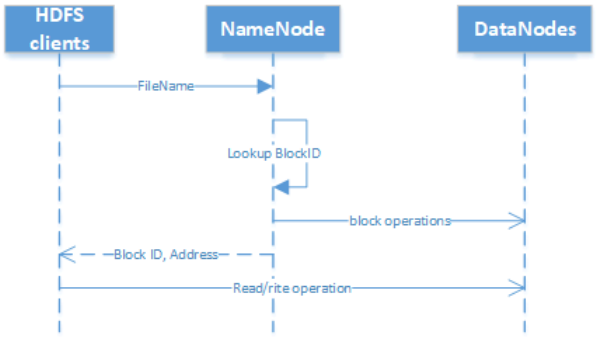

Fig. 2. Sequence Diagram of HDFS Operations

II. Background

A. Apache Hadoop

Apache Hadoop is an open source framework provides availability for distributed processing large data sets across single to thousands of nodes[11]. Hadoop was created by Nutch and a part of Apache Lunce. After implementing NDFS and MapReduce, in 2008, it became an independent one and rapidly reached the Apapche projects top-level in 2010. The basic archirecture of Apache Hadoop V2.0 is illustrated in Fig. 1. It has three main components including HDFS, MapReduce, and YARN.

- HDFS is designed for distributed file storage
- MapReduce provides features for parallel processing
- YARN stands for Yet Another Resources Negotiation that consists of ResourceManager and NodeManager. The former allocates resources between all application, the latter monitors the resource usage of the containers and report to ResourceManager.

Our proposed system utilizes HDFS as storage system, hence, this section introduces HDFS in detail. Files in HDFS are stored in independent block-size units that are much bigger than normal disk block size. A HDFS cluster has two different types of node namely namenode (or master node) and datanodes (worker nodes). NameNode manages namespace and block mapping to DataNodes, while data block units are stored in DataNodes. HDFS is designed to store data in large scale with reliability, therefore it makes a number of replication when files are ingested into the HDFS system. This factor is set to three as the default value. The HDFS reading and writing process of clients are illustrated in Fig. 2 that can be summarized as following steps

- HDFS clients send request to NameNode with a file name.
- NameNode looks up in its managed name space for data blocks ID of the requested file and returns to clients.
- NameNode frequently check DataNode with heartbeat.
- NameNode returns block ID and address to HDFS clients.
- Clients use a given block ID and perform reading and writing operations.

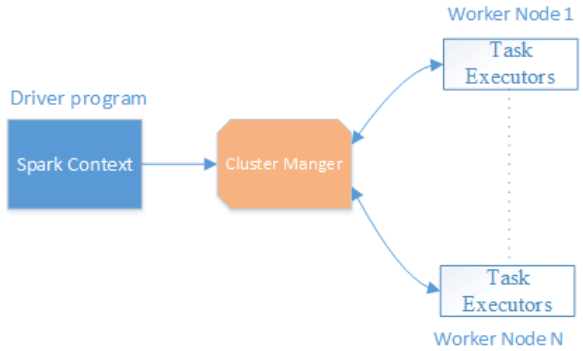

Fig. 3. Spark Applications Architecture

B. Apache Spark

Apache Spark is an analytics engine for large-scale data processing that supports high-level programming API in various languages and tools for querying, graph processing, and machine learning. It was started in 2009 in University of California, Berkeley RAD Lab and became an ASF top-level project in 2014. Spark performs 100 times faster than Hadoop Map Reduce for big data processing and can be deployed thourgh several other frameworks such as Mesos, Yarn, or standalone cluster. Fig. 3 depicts the basic architecture of Spark applications. The driver program in the Master Node first creates *Spark Context* that split jobs into tasks. Spark Context works with the *Cluster Manager* to mangage and distribute tasks over the clusters. Worder Nodes execute tasks and return resutls to Spark Context.

In order to program in Spark environment, this section introduces some basic concepts such as Resilient Distributed Dataset (RDD), DataFrame, and Dataset. RDD is immutablity data object that has various types and are stored in memory. DataFrame is semilar to relational database tables that is a distributed collection of data. DataFrame constructed from RDDs, tables in Hive, or external databases is conceptually equivalent to Pandas data frames. Listing 1 illustrate a Spark DataFrame object is created from a Parquet file.

Listing 1: Create Spark DataFrame from Parquet file

```
from pyspark.sql import *
parquetDF = spark.read.parquet("example.parquet")
```

C. Raster Frames

RasterFrames, a part of LocationTech project, provides functionalities to access earth observation data, cloud computing, and data analysis[12]. It allows to load and analyze raster data in DataFrames, to perform spatial queries and operator, to integrate with Apache Spark with powerful features in processing big data and machine learning library. RasterFrames has capability to read various types of raster data including GeoTIFF, JP2000, MRF, HDF via many protocols such as HTTP, FTP, HDFS, S3. Besides, it also can work with multiple spatial vector data types (point, line, polygon) from many data sources such as GeoJSON, WKT/WKB. It also supports for preserving geometry column while converting GeoDataFrame to SparkFrame. The Listing 2 shows the code for reading a GeoJSON file and convert multipolygon data in form of GeoDataFrame to SparkFrame. RasterFrames introduces a new type of data named *Tile* which is subset of a scene. A *Tile* is usually a two-dimensional array and square.

Listing 2: Reading GeoJSON and converting data with RasterFrames

```
from pyspark import SparkFiles
import geopandas
from shapely.geometry import MultiPolygon

data_uri = 'http://<uri>/example.geojson'
spark.sparkContext.addFile(data_uri)
df = spark.read.geojson(SparkFiles.get('example'))

gdf = geopandas.read_file(data_uri)
gdf.geometry = gdf.geometry.apply(_multipolygons)
df2 = spark.createDataFrame(gdf)
```

In addition to features of handling with raster and vector data as well as sptial operator, RasterFrames also provides ability to train and predict with Apache Spark Machine Learning library(Spark MLLib) [13] that makes ML practical and easy. It consists of serveral libraries including ML algorithms (classification, regression, clustering, etc..), featurization, pipelines, persisence, and other ultilities. RasterFrames is built on top of Apache Spark, hence our proposed framework can be easily scaled up to run on cluster and cloud with the prorotyped solution developed on personal laptop.

III. Related Work

Gomes *et al.* [14] made a great review of seven platforms for big earth observation data management and analysis. Among them, ODC and SEPAN are open source ones which are similar to our proposed framework. The difference is that our proposed framework uses Hadoop and RasterFrames which can start with simple model but it can be extended later.

Rajak *et al.* [15] proposed a general framework for storing and processing high resolution satellite images using Apache Hadoop. Their work used MapReduce to perform computation and used HBase to store processed images. Our work is different and more concrete because we utilize RasterFrames with more powerfull features to analyze earth observation data.

Haifa Tamiminia*et al.* [16] made a systematic review of 349 articles that utilize Google Earth Engine for various big geodata applications. They showed the advantages of using GEE for big data processing with rich support features such as cloud storage, ML, DL algorithms.

Eken and Sayar [17] proposed a distributed big data framework for large scale raster mosaic images processing based on MapReduce programming model. The proposed framework consists of five steps including pre-processing, identifying mosaics, polygon coverage, stitching vectorised images, and object extraction and analysis.

Yanbo Huang *et al.* [18] presented a four-layer-twelve-level satellte remote sensing data management for agricultural remote sensing data including high-resolution satellites, UAV images, etc..

Huang *et al.* [19] introduced a scalable, efficient for heterogeneous remote sensing big data and management in distributed environment. The authors also used HDFS and HBase as storage layer. To query metadata contents, they constructed the index based on Elasticsearch.

Sedona *et al.* [20] proposed to use High Performance Computing machines to compute classification algorithms in parallel with CNNs.

Recently, Dalton Dunga [21] proposed RESFlow framework that includes several modular components such as clustering and embedding, image-bucket assignment, image gallery, model gallery, accelerated inference, application space,and image analytic. The framework provides parallel computing using Spark.

Arushi Patni [22] *et al.* introduced a model for satellite mages classification based on HDFS and Deep Neural Network. The result, however, just presented the architectural model with processing flow. It did not show any specific framework for processing satellite data and give any concrete example for their proposed model.

Hai Lan *et al.* [23] developed a Spark-based large-scale for processing remote sensing images. The authors used GAL to extract raster data and to transform to Spark RDDs. They proposed algorithms to caculate NDVI and NDWI values for classification and changing detection. The research also performed theree experiments with Landsat 8 and MODIS datasets on Amazon S3 and GEE.

Nguyen *et al.* [24] developed an end-to-end analysis pipeline using Caffe deep learning library along with two distributed platforms such as Spark and Dark.

Sukanta Roy*et al.* [25] also proposed to use Apache Hadoop as a solution to store remote sensing data. They implmented MapReduce programs for preprocessing Synthetic Aperture Radar and Multispectural data.

In comparision to these works, our approach is different because we use HDFS as infrastrucre along with Apache Spark and RasterFrames to process and analyze raster data at scale.

IV. Big Data Framework for Satellite Images Processing

Firstly, the general approach for processing big satellite imaginary data is presented, we then introduce the surface water extraction following the proposed approach for illustration purpose.

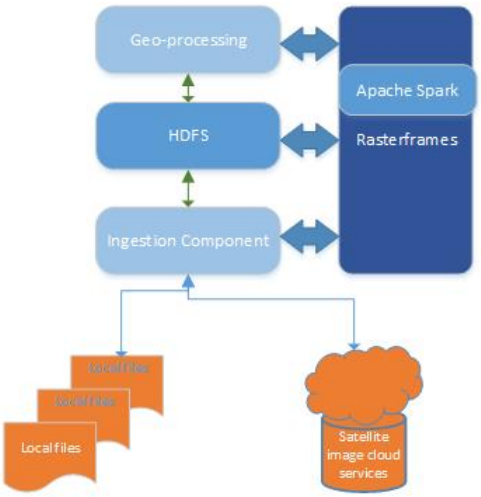

Fig. 4. Framework Architecture

A. Framework Architecture

Figure 4 depicts the architecture of for big satellite imagery processing framework. It based on HDFS and RasterFrame consists of two main layers that are describe as follows.

- The ingestion component: retrieves satellite imaginary data from local files or many cloud sources including Amazon S3, Sentinel-hub, etc.. This feature is implemented by using catalog and raster readers in RasterFrame. The *Ingestion* component ingests data and stores in HDFS clusters.
- The geo-processing component: proceeds and analyzes the raster data from HDFS or external sources. This component also reuse machine learning models with Spark ML Pipelines. These facilities are provided with RasterFrame.

B. Processing Workflow

Based on the proposed architecture, this section introduces the workflow of two main components: *Ingestion* and *Geoprocessing*.

Fig. 5 illustrates the flow of Ingestion. The input of the process is eitheir local raster files, external catalog in cloud-based services, or built-in catalogs supported by RasterFrames. These data are loaded in parallel into Spark DataFrames that can handle big data files. After loading to memories, if users are able process the in-memory data by invoking geoprocessing component in case of neccesity. Finally. it allows to store in HDFS with original file format or Apache Parquet files.

Fig. 6 depicts the general flow of information mining from imaginery. At the first step, it loads data stored in HDFS or external catalog from AWS Public Data Set (PDS). Next two steps are data preparation and data quality enhancement. In train and predict step, we choose the appropriated algorithms or models that provided SparkMLLib for each specific task. In case of supervised learnining models, label data is required and joined with the prepared data. After prediction, in the last step, we evaluate the model, visualize the results, or serialize to the proposed HDFS system.

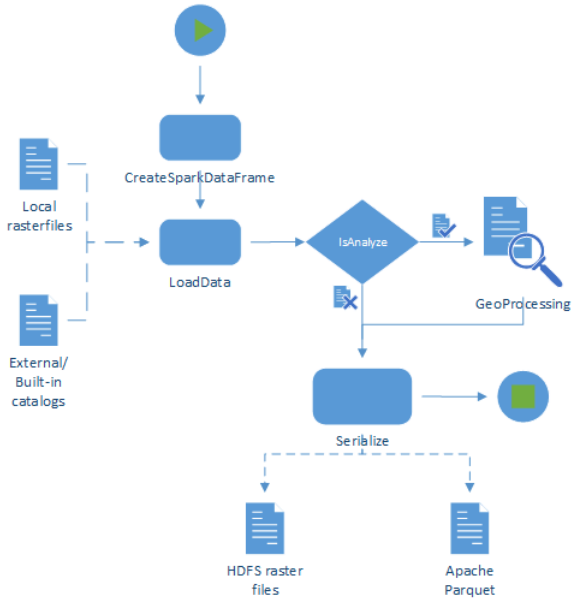

Fig. 5. Ingestion Work Flow

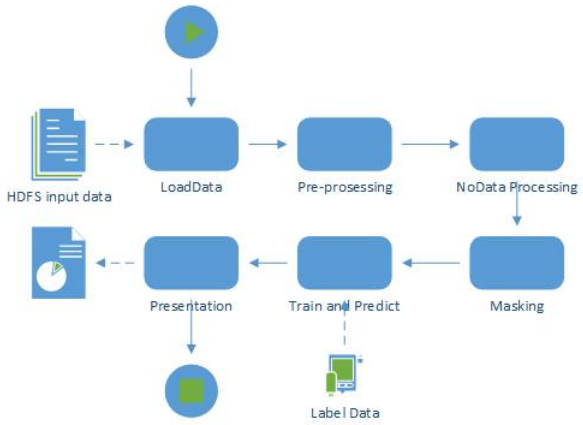

Fig. 6. Geoprocessing Work Flow

V. A Case Study: Surface Water Extraction in Phu Tho Province, VietNam

This section presents a case study of surface water extraction with Sentinel-2 images following the approach introduced in section IV-A. First, we introduce the region and used data. The data sources are Sentinel-2 images of the study area which are stored as local files in single computer nodes. These images are stored into HDFS clusters for processing later using ingestion layer. For surface water extraction, we follow the proposed framework and use a supervised machine learning technique which is built inside RasterFrames.

A. Study Area and Used Data

Phu Tho is a mountainous province in North located at the centre of Vietnam's western Northeast. Phu Tho has 13 district, city and township with five large rivers including Chay, Hong, Da, Lo, and Bua rivers and many branches which provide more than 2,500 ha surface water are for agriculture development.

It also has various group of residents and variety of living culture. Phu Tho has been a industrial province with theree regions Viet Tr, Bai Bang - Lam Thao, and Thanh - Ha Hoa. Currently, surface water quality in PhuTho has been quickly polluted at the alarm level caused by many factors such as industrialization, urbanization, and living styles.

The case study uses Sentinel-2 images as the data source which were collected in range from 2019 to 2020 and pushed into HDFS using *Ingestion* component. The Sentinel-2 mission consists of two satellite systems including Sentinel-2A and Sentinel-2B, which were launched on 23 June 2015 and 07 March 2017 respectively. These satellites provide a global coverage of the Earth's land surface every five days.

B. Main Process

Figure 7 depicts the detailed work flow of extraction process that consists of the following steps

- Load data from HDFS: After the local Sentinel-2 images are ingested to HDFS, the selected are data are loaded into data frames.
- Remove low quality data: It is the masking operation to set values to missing data or cloudy cells. The SCL (scene classification) band is used to check if a cell contains no data, saturated, cloud probability, or vegetation i.e., the cell value is in range of 0,1,4,8,9,10. It can be implemented by invoking function *rf_local_is_in[0,1,4,8,9,10]*.
- Create label data: Creating polygon regions in form of GeoJSON to indicate surface water areas. This small DataFrame is joined later with raster DataFrame loaded from HDFS.
- Create SparkML pipelines: This step utilizes Apache Spark ML pipelines to create a pipeline which consists of four stages including tile exploding, data filtering, assembler and classifier. For surface water extraction, we use Decision Tree and Random Forest algorithms to evaluate. Listing V-B shows the snippet of Python code to construct the pipeline with Random Forest algorithm.

Listing 3: Create SparkML pipelines

```
classifier = RandomForestClassifier() \
  .setLabelCol('label') \
  .setFeaturesCol(assembler.getOutputCol())
pipeline = Pipeline() \
  .setStages([tileExploder, dataFilter, \
        assembler, classifier])
```

- Training and Prediction: Execute the pipeline created in the previous step and make model prediction applying *transformation* method on the pipeline as follows.

```
model = pipeline.fit(model_input)
pre = model.transform(df_mask) \
.drop(assembler.getOutputCol()).cache()
```

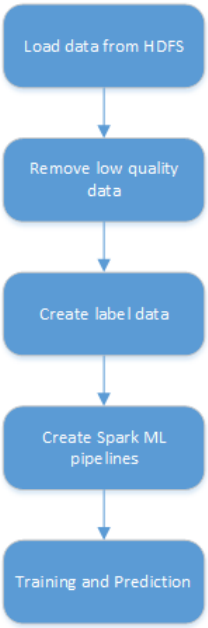

Fig. 7. Surface Water Extraction Flow

Fig. 8. Create Label Data for Surface Water

C. Implementation and Results

As mentioned in the previous section, we implement the surface water in Phu Tho province. After reading raster data of the study area from HDFS, we create a small set of data including 26 sample label of surface water in Phu Tho across the several Northern provinces in VietNam. These data can be created by add features in geojson.io illustrated in Fig. 8.

Fig. 9 shows the three extracted scenes and the prediction results of the study region that use DecisionTree algorithms and satellite images collected in April 2020.

We also can evaluate the correctness of the models by using the following the snippets (Listing 4) to show the accuracy and confusion matrix of the model.

Listing 4: Evaluate the model

```
model_eval = MulticlassClassificationEvaluator(
prediction=classifier.getPredictionCol(),
```

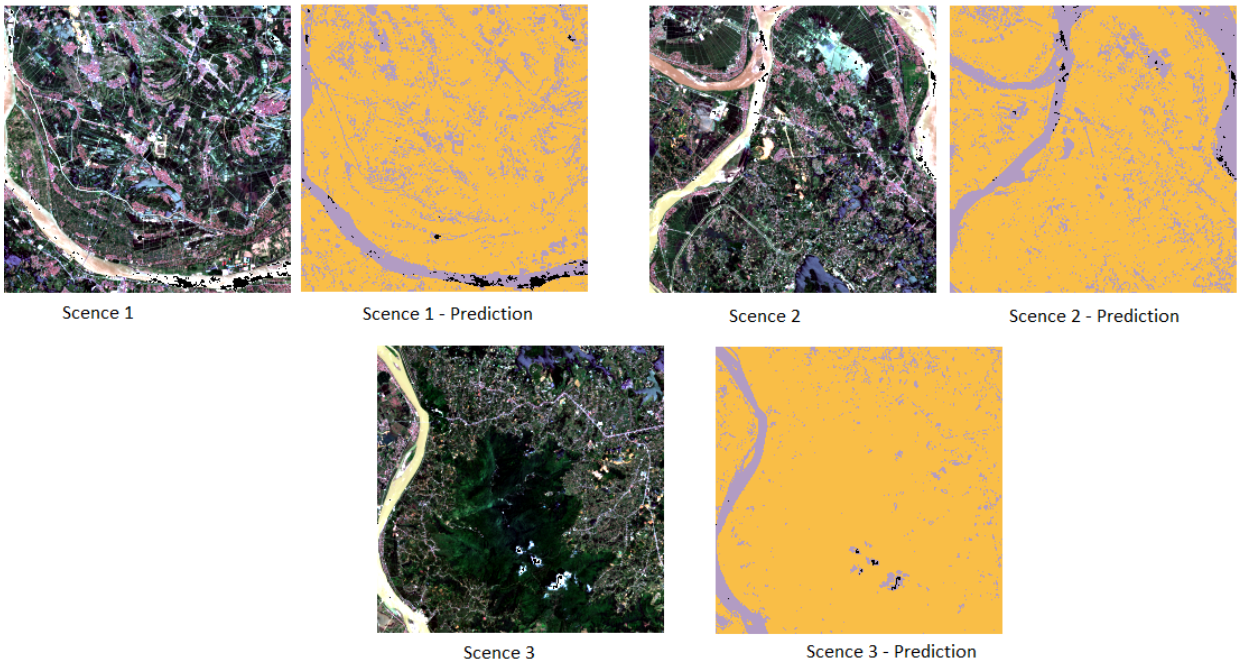

Fig. 9. Prediction results of extracted scenes

```
label=classifier.getLabelCol(),
metricName='accuracy')
accuracy = model_eval.evaluate(prediction_ret)
confusion_matrix = prediction_df.groupBy
(classifier.getPredictionCol()) \
.pivot(classifier.getLabelCol()) \
.count() \
.sort(classifier.getPredictionCol())
```

The prediction accuracy of DecisionTree model is slightly better with 0.959 while RandomForest gives the result at 0.954 those confusion matrixes are shown in Table I and Table II respectively.

TABLE I. CONFUSION MATRIX WITH DECISION TREE ALGORITHM

prediction	Water	Non-Water
Water	354	17
Non-Water	46	

TABLE II. CONFUSION MATRIX WITH RANDOM FOREST ALGORITHM

prediction	Water	Non-Water
Water	353	18
Non-Water	1	45

VI. CONCLUSION AND FUTURE WORK

In this paper, we propose a framework for satellite images processing in large scale. It based on the open source platforms such as Apache Hadoop and RasterFrames consists of two major components including storage layer with HDFS architecture and processing layer that combines Spark and RasterFrames. The proposed framework can be built on the commodity hardware and has the low cost for deployment. The paper also implements a surface water extraction with Sentinel-2 images in PhuTho province Viet Nam with high accuracy for illustration purpose. The case study shows the flexibility of the proposed framework. We intend to develop the framework with more powerful features such as deep learning libraries for object segmentation and detection.

ACKNOWLEDGMENT

This research is funded by Hanoi University of Mining and Geology, Vietnam (Project No. T19-29).

REFERENCES

[1] J. Salazar Loor and P. Fdez-Arroyabe, *Aerial and Satellite Imagery and Big Data: Blending Old Technologies with New Trends*. Cham: Springer International Publishing, 2019, pp. 39–59.

[2] H.-D. Guo, L. Zhang, and L.-W. Zhu, "Earth observation big data for climate change research," *Advances in Climate Change Research*, vol. 6, no. 2, pp. 108 – 117, 2015, special issue on advances in Future Earth research. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1674927815000519

[3] Y. Ma, H. Wu, L. Wang, B. Huang, R. Ranjan, A. Zomaya, and W. Jie, "Remote sensing big data computing: Challenges and opportunities," *Future Generation Computer Systems*, vol. 51, pp. 47 – 60, 2015, special Section: A Note on New Trends in Data-Aware Scheduling and Resource Provisioning in Modern HPC Systems. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X14002234

[4] (2020) Google earth engine. https://earthengine.google.com/.

[5] (2020) Sentinel Hub. https://sentinel-hub.com/.

[6] (2020) Open Data Cube. https://opendatacube.org/.

[7] (2020) OpenEO. https://openeo.org/.

[8] T. D. Acharya, A. Subedi, H. Huang, and D. H. Lee, "Classification of surface water using machine learning methods from landsat data in nepal," *Proceedings*, vol. 4, no. 1, 2019. [Online]. Available: https://www.mdpi.com/2504-3900/4/1/43

[9] W. Huang, B. DeVries, C. Huang, M. Lang, J. Jones, I. Creed, and M. Carroll, "Automated extraction of surface water extent from sentinel-1 data," *Remote Sensing*, vol. 10, no. 5, p. 797, May 2018. [Online]. Available: http://dx.doi.org/10.3390/rs10050797

[10] A. Sekertekin, S. Y. Cicekli, and N. Arslan, "Index-based identification of surface water resources using sentinel-2 satellite imagery," in *2018 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, 2018, pp. 1–5.

[11] (2020) Apache Hadoop. https://hadoop.apache.org.

[12] (2020) Rasterframes. https://rasterframes.io/.

[13] (2020) Apache Spark. https://spark.apache.org/.

[14] V. C. F. Gomes, G. R. Queiroz, and K. R. Ferreira, "An overview of platforms for big earth observation data management and analysis," *Remote Sensing*, vol. 12, no. 8, 2020. [Online]. Available: https://www.mdpi.com/2072-4292/12/8/1253

[15] R. Rajak, D. Raveendran, M. C. Bh, and S. S. Medasani, "High resolution satellite image processing using hadoop framework," in *2015 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, 2015, pp. 16–21.

[16] H. Tamiminia, B. Salehi, M. Mahdianpari, L. Quackenbush, S. Adeli, and B. Brisco, "Google earth engine for geo-big data applications: A meta-analysis and systematic review," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 164, pp. 152 – 170, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0924271620300927

[17] S. Eken and A. Sayar, "A mapreduce based big-data framework for object extraction from mosaic satellite images," 2018.

[18] Y. Huang, Z. xin CHEN, T. YU, X. zhi HUANG, and X. fa GU, "Agricultural remote sensing big data: Management and applications," *Journal of Integrative Agriculture*, vol. 17, no. 9, pp. 1915 – 1931, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2095311917618598

[19] X. Huang, L. Wang, J. Yan, Z. Deng, S. Wang, and Y. Ma, "Towards building a distributed data management architecture to integrate multi-sources remote sensing big data," in *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, 2018, pp. 83–90.

[20] R. Sedona, G. Cavallaro, J. Jitsev, A. Strube, M. Riedel, and J. A. Benediktsson, "Remote sensing big data classification with high performance distributed deep learning," *Remote Sensing*, vol. 11, no. 24, 2019.

[21] D. Lunga, J. Gerrand, L. Yang, C. Layton, and R. Stewart, "Apache spark accelerated deep learning inference for large scale satellite image analytics," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 271–283, 2020.

[22] A. Patni, K. Chandelkar, R. K. Chakrawarti, and A. Rajavat, "Classification of satellite images on hdfs using deep neural networks," in *International Conference on Advanced Computing Networking and Informatics*, R. Kamal, M. Henshaw, and P. S. Nair, Eds. Singapore: Springer Singapore, 2019, pp. 49–55.

[23] H. Lan, X. Zheng, and P. M. Torrens, "Spark sensing: A cloud computing framework to unfold processing efficiencies for large and multiscale remotely sensed data, with examples on landsat 8 and modis data," *Journal of Sensors*, vol. 2018, p. 2075057, Aug 2018. [Online]. Available: https://doi.org/10.1155/2018/2075057

[24] M. H. Nguyen, J. Li, D. Crawl, J. Block, and I. Altintas, "Scaling deep learning-based analysis of high-resolution satellite imagery with distributed processing," in *2019 IEEE International Conference on Big Data (Big Data)*, 2019, pp. 5437–5443.

[25] S. Roy, S. Gupta, and S. Omkar, "Case study on: Scalability of preprocessing procedure of remote sensing in hadoop," *Procedia Computer Science*, vol. 108, pp. 1672 – 1681, 2017, international Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050917305598