Developing a Big Data Framework for Processing Sentinel-2 Data in the Context of Nuclear Safeguards

Evaluation of Apache Airflow, Rasdaman and Google Earth Engine

Lisa Beumer and Irmgard Niemeyer

Forschungszentrum Jülich GmbH, Germany
E-mail: l.beumer@fz-juelich.de, i.niemeyer@fz-juelich.de

Abstract:

In the last years, Earth observation (EO) satellites have generated big amounts of geospatial data. Many providers offer their satellite data at low cost or even for free. For example, initiatives such as the Copernicus program, the European Union's Earth observation program, have revolutionized the market. The growing archives of satellite imagery open up a wide range of satellite EO applications, also in the field of nuclear verification where satellite imagery represents a key source of information for the implementation and verification of nuclear non-proliferation treaties [1]. The data collected, processed, analyzed, and managed for monitoring purposes is not only increasing in volume, but also becoming more and more heterogeneous, unstructured, and complex. However, Big Data is also accompanied with several issues related to capturing the data, sharing, transferring, updating, processing, and analyzing. To meet these demands, novel technologies have been developed. Apache Airflow for example has become a popular tool for defining, scheduling, visualizing, and monitoring Big Data related workflows [2]. For storing and accessing multidimensional raster data, such as satellite imagery, an array database management system, called Rasdaman, has become well established [3]. To analyze these large amounts of data effectively and efficiently, Google has developed a free-to-use cloud computing platform, known as Google Earth Engine (GEE) [4]. In this research an automated procedure for collecting, storing, processing, and analyzing satellite images based on the tools mentioned above was developed. Hereby, the strengths of Airflow in terms of the creation of dynamic workflows with high granularity and the log entries of execution became evident. Furthermore, Rasdaman provides indispensable advantages such as the open standards-based data-cube analytics possibilities. The usability and benefits of GEE with respect to big EO data management and analysis were evaluated through an analysis of two different machine learning algorithms, namely Random Forest (RF) and Classification and Regression Trees (CART). Regarding the target land over classes, the classification results of manual generation were compared with two by GEE provided land cover maps from the years 2017 and 2019. The overall accuracy of the RF and CART classifiers for the Sentinel-2 images was in the range of 87% to 98%, and 68% to 83%, respectively.

Keywords: Satellite Imagery; Big Data; Data Science; Airflow; Copernicus Hub; Rasdaman; Google Earth Engine

1. Introduction

Geospatial data, satellite imagery in particular, represents a key source of information for the implementation and verification of nuclear non-proliferation. In 1998, the IAEA started to investigate the potential use of commercial satellite imagery to support the safeguards implementation and nowadays it "[…] has become a very important source of information [...] especially with respect to sites to which the IAEA does not have access." [1]. Many applications of satellite imagery in the field of nuclear verification have been identified over time. With commercial satellite imagery available to the public, new opportunities are emerging to monitor nuclear activities at both known and undeclared nuclear facilities in a more proactive manner to verify compliance with non-proliferation agreements. As numerous studies have shown, satellite imagery provides analysts with clear insights into nuclear facilities and nuclear activities worldwide, for example, to confirm the status of an inoperable facility or declared production without having to visit the sites in person [1,5,6,7]. Moreover, use cases such as the recognition and monitoring of small-scale features for instance the construction of buildings, plant expansions or the preparation of underground facilities are also considered. The amount of available and heterogeneous satellite EO data is steadily increasing , as no longer only a few operators and government sources offer the data as primary source, but private companies are also investing in EO satellites, driven by technological advances that allow for higher resolution sensors and a higher return rate capacity. Many provider offer their satellite data at low cost or even for free. For example, initiatives such as the Copernicus program, the European Union's Earth observation program, have revolutionized the market. The demand for their immense amount of data is huge, as the Copernicus Sentinel Data Access Report of 2020 shows [8]. In 2020, a total data volume of 7.65 PiB was published, which is significantly more compared to the European Space Agency's (ESA's) entire collection of EO data from the pre-Copernicus era which amounts to 5.6 PB [8]. The average daily download volume of the Sentinel Data Access System was

405 TiB, resulting in a total of 82.8 PiB of products downloaded just in 2020 [8].

Due to the sheer volume and the velocity at which the amount of data is increasing, remote sensing data is referred to as Big Data. However, aspects such as diversity, complexity and trustworthiness also make this type of data Big Data. But what exactly does Big Data mean and to what extent does it apply to satellite imagery? The term Big Data refers to large data sets, whether structured, unstructured, or complex, that are difficult or even impossible to store, process and analyze using conventional methods. To define the term more precisely, several multi-V models were introduced in the last years starting with the 3-V model. In this paper, the 5-V model, characterized by the following propertied, is taken into consideration: Volume, velocity, variety, veracity and value. The term volume simply refers to the quantity of an existing and fast-moving amount of data. In this context, there is no upper boundary at which data is considered to be "big". The speed at which the data accumulates is summarized under the second V, namely velocity. Depending on the data source, a different data type is present, for example, data can be of an unstructured, semi-structured or structured nature. This characteristic is represented by the term variety. Veracity describes the data quality and its accuracy. Only data with known origin and quality, such as correctness and completeness, are generally considered to be reliable and can be trusted. In addition, data analysis can only provide a meaningful result if high quality data is available as input data. The last characteristic, referred to as value, refers to the usefulness of data. This raises the question of the benefit of high quality data if there is no use case in terms of a concrete example. So, one has to weigh whether to store all data or only useful data, the so-called smart data.

The data quantity and quality keeps moving forward with the aim of offering high and medium spatial resolution images on a daily basis. However, Big data is also accompanied with several issues related to capturing the data, storing, processing and analyzing it. In turn, this will create new challenges for the analyst to use the datasets appropriately and in a timely manner. No longer can visual interpretations of single satellite image scenes be expected to address the analysis requirements for such large repositories of satellite imagery datasets. To meet these demands, novel technologies have been developed. Apache Airflow for example enables the optimization of data processing and workflow management processes [2]. For storing and accessing multidimensional raster data, such as satellite imagery, an array database management system, called Rasdaman, has become well-established [3]. To analyze large amounts of data effectively and efficiently, Google has developed a free-to-use cloud computing platform called Google Earth Engine (GEE). The platform provides access to publicly available remote sensing imagery and machine learning algorithms [4]. In this research, these tools have been utilized to develop a semi-automated procedure for collecting, storing, processing and analyzing satellite images. The project plan is shown in Figure 1.

Within the scope of this work, Sentinel-2 data is obtained from the Copernicus program. Due to the diversity of possible data, a comprehensive preparation of the data in process usable formats is necessary to be able to use appropriate analysis algorithms. The data source is integrated into the Apache Airflow workflow management system capable of downloading, validating, preprocessing and storing the data into a Rasdaman database. Finally, the efficiency of the Google Earth Engine to effectively execute Big Data workflows using Google's provided machine learning techniques is explored. The potential of the developed framework is tested using case studies concerning nuclear fuel cycle related sites. Hereby the objective is to classify land cover use, as these features provide essential information for recognizing and monitoring for example changes of the operational status, constructions of new buildings and roads, plant expansions, etc.

2. Tool Fundamentals

2.1 Airflow

Apache Airflow is an open-source workflow management platform written in Python that enables the creation and management of data pipelines, as well as their automatic

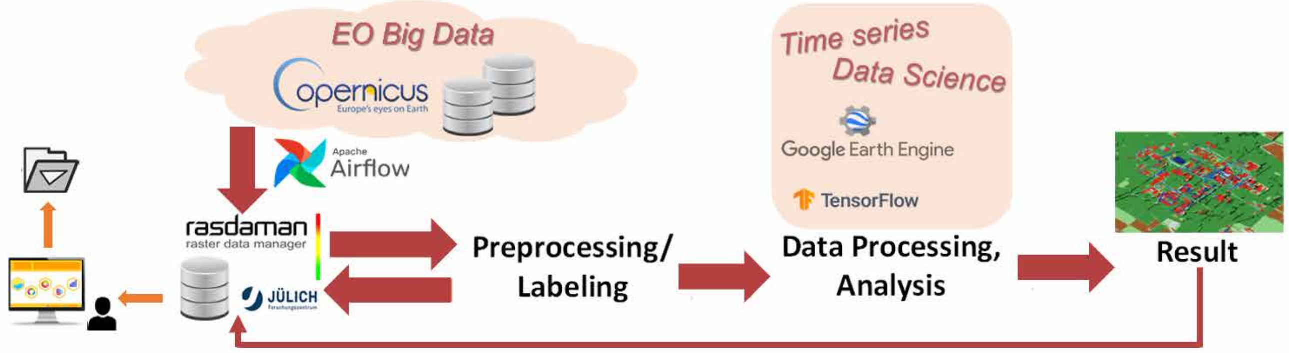


Figure 1: Project overview.

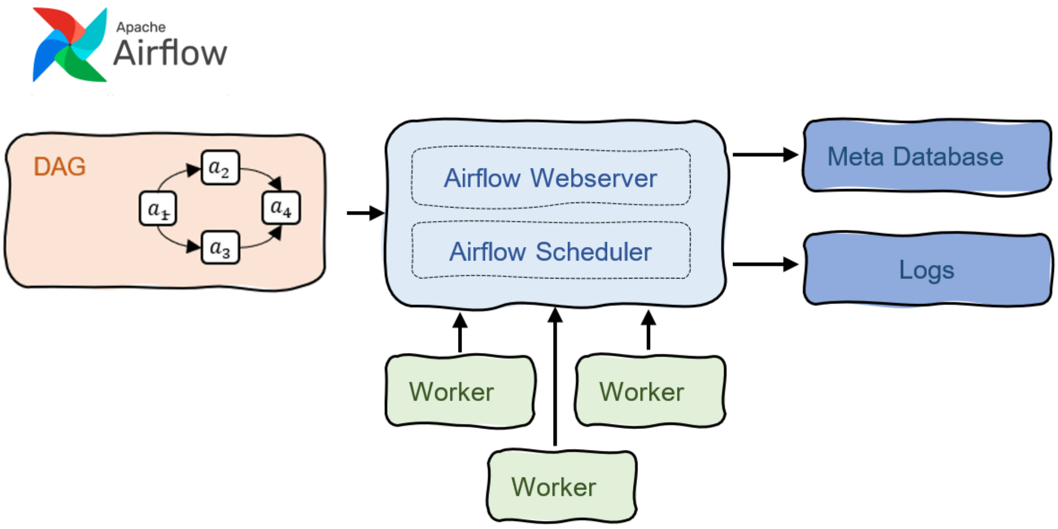


Figure 2: Terminology used in Airflow. The key concept are direct acyclic graphs (DAGs) managed by a webserver and a scheduler.

execution [1]. The key terminology used in Airflow is shown in Figure 2.

The core concept of Airflow is represented by Directed Acyclic Graphs (DAGs), which collect tasks together. A DAG forms an abstract structure consisting of nodes and edges. The nodes represent the individual work tasks and the edges the connections between them, having a direction. According to [9] a direct graph is defined as $G = \{V, E\}$, where $V = \{a_1, a_2, \dots, a_N\}$ is a finite set of nodes and E a finite set of directed edges. It holds that $E = \{a_j \to a_{j'} | a_j, a_{j'} \in V, a_j \neq a_{j'}$ This graph is called acyclic if there does not exist

$$(j, j^{(1)}, j^{(2)}, \dots, j^{(n)})\ s.t. \{a_j \to a_{j^{(1)}}, a_{j^{(1)}} \\ \to a_{j^{(2)}}, \dots, a_{j^{(n-1)}} \to a_{j^{(n)}}, a_{j^{(n)}} \to a_j\} \subset E \quad .$$

An example of a DAG is given in Figure 3.

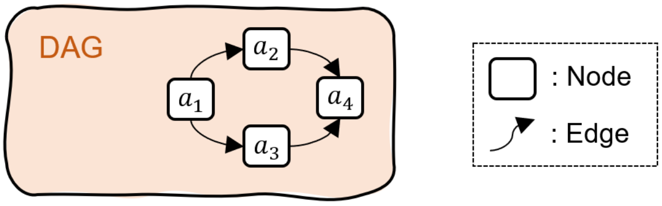


Figure 3: Simple example of a DAG $G = \{V, E\}$with the set of events $V = \{a_1, a_2, a_3 , \mathrm{a}_4\}$ and $E = \{a_1 \to a_2, a_1 \to a_3, \mathrm{a}_2 \to \mathrm{a}_4, \mathrm{a}_3 \to \mathrm{a}_4\}$ representing the set of directed edges.

The management of the whole system is performed by a graphical user interface (webserver) and a scheduler. The webserver enables the creation of workflows and their management. The detailed status of each workflow can be displayed, thus making a live monitoring possible. Apache Airflow manages and controls workflows via schedulers, whereas both sequential and parallel schedulers are supported. Workflows are executed according to a predefined schedule or trigger events. Once the schedule is created, according to which the tasks of the defined rules can be processed, the scheduler assigns them to the workers, which are responsible for the actual processing of the individual tasks according to their respective Python description. For documentation purposes, all task information is stored in the meta database. Log files can be used for debugging, error analysis or documentation.

2.2 Copernicus Data Hub

Copernicus, formerly known as GMES (Global Monitoring for Environment and Security), is a European Union program aimed at establishing a European capacity for global environment and security monitoring [10]. The Program is funded, coordinated, and managed by the European Commission in cooperation with partners such as ESA (European Space Agency) and EUMETSAT (European Organisation for the Exploitation of Meteorological Satellites). The program provides data from its own fleet of satellites, called Sentinel, in-situ data and data from national and commercial satellites [11]. The Sentinel satellites consist of six missions: Sentinel -1 (High Resolution Radar), -2 (Optical for Vegetation), -3 (Optical/Thermal for Oceans), -5P (An Eye for Air) and -6 (Sea Level Elevation) are stand-alone satellites, while Sentinel-4 and -5 are dedicated measurement instruments installed on EUMETSAT. There are currently eight Sentinels in space, namely Sentinel-1A & -1B (2014, 2016), Sentinel-2A & -2B (2015,2017), Sentinel-3A & -B (2016, 2018), Sentinel-5P (2017), Sentinel-6 (2021). The Sentinel data and Copernicus services are free of charge and are provided through ESA's Copernicus Open Access Hub, previously known as Sentinels Scientific Data Hub.

One has the possibility to download the data through a graphical interface or via two different application programming interfaces (APIs), OData and Open Search (Solr). In this work, OData, a data access protocol built on the HyperText Transfer Protocol (HTTP) and the Representational State Transfer (REST), was used because it can be easily integrated in Python using Client for URLs (cURL) or Wget. The data resources to be queried are uniquely identifiable via so-called Uniform Resource Identifiers (URIs) and can be requested via HTTP messages. As shown in Figure 4, a URI is composed of up to three components: (1) Service Root URI, (2) Resource path and (3) Query options that control the amount and order of the data.

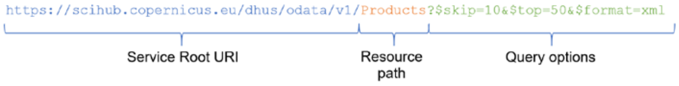

https://scihub.copernicus.eu/dhus/odata/v1/Products?$skip=10&$top=50&$format=xml
Service Root URI
Resource path
Query options

Figure 4: Example of an OData URI. Source [12].

Since the queries used in this work are too complex and thus vulnerable to cURL and Wget errors, a generic bash script was implemented. This is based on the script dhusget.sh provided by Copernicus, which is a simple demo script illustrating how to use OData and OpenSearch APIs to query and download products from any Data Hub Service [13].

2.3 Rasdaman

Originally, databases were developed to store, manage, and query alphanumeric data efficiently. However, data storage requirements have changed over the years. For example, when looking at satellite imagery, it is necessary to be able to store multi-dimensional data. Array database management systems (array DBMSs) provide database services specifically for raster data and aim to provide a flexible and scalable management of this kind of data. In the context of this work, the array DBMS Rasdaman is reviewed, which aimed to form a comprehensive DBMS support for raster data of arbitrary size and dimension over arbitrary base types, so-called multidimensional discrete data (MDD) [3]. Before the system architecture is described, the logical data model used by Rasdaman is explained whereby the declaration of [14] is used.

2.3.1 Logical data model

Multidimensional array data, also known as multidimensional discrete data (MDD) is located in a discrete space $\mathbb{Z}^d$. Figure 5 illustrates a three-dimensional MDD in a discrete space $\mathbb{Z}^3$.

A multidimensional object α is the mapping of a value of the base type to each vector of its domain, i.e., the multidimensional interval it takes:
$\alpha \coloneqq \{(x, v(x)) | v(x) \in T, x \in D\}$, where x describes the cell and $v(x)$ the corresponding cell value. The domain is spanned by an interval D of dimensionality d, where each dimension i has a lower bound l_i and an upper bound j_i:

$$D \coloneqq \overset{d}{\underset{i=1}{\mathrm{X}}} \{x | 1_i \leq x \leq h_i, x \in \mathbb{Z}\} = [1_l : h_l] \times \ldots \times [1_d : h_d],$$

$$l, h \in \mathbb{Z}^d \; and \; l_i \leq h_i \forall i \in \{1, \ldots d\}\,.$$

A single cell value $v(x)$ can be assigned a base type T, which may be of atomic or composite data types

$$T \coloneqq \overset{n}{\underset{i=1}{\mathrm{X}}} \{t_i | t_i \in \zeta \cup T\}; n \in \mathbb{N};\ \zeta \in \mathbb{N}_0 \cup \mathbb{Z} \cup \mathbb{R} \cup \mathbb{B} \ldots .$$

A cell can represent a single value such as a gray value or a composite value, for example the red, green, and blue components of a color image. In addition to the basic data type T, an MDD has a data type M, which is described by D and T, M=<D,T>. A set of MDD of the same type M are called collection, defined by $C \subset \{\alpha \mid type(\alpha) =< D, T >$. If operations are now applied, a distinction is made between those on MDD and those on collections. Geometric

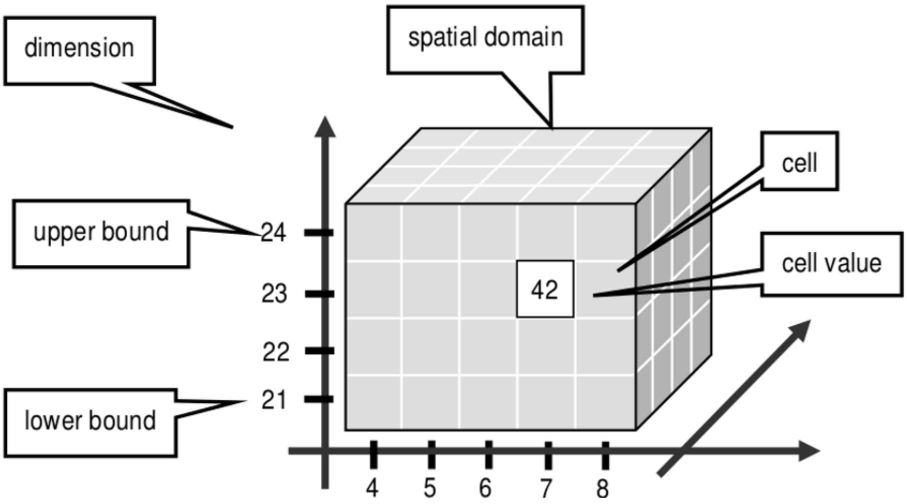


Figure 5: Constituents of a three-dimensional MDD. Source [15].

operations, induced operations, aggregate operations, and cell operations can be applied to multidimensional objects. In case of collections, relational operations such as application, selection, cross product, among others, can be applied.

Big Data cubes can be created in Rasdaman via an OGC (Open Geospatial Consortium) Web Coverage Service - Transaction Extension (WCS-T) standard interface that allows users and machines to insert, update, and delete data via simple web requests. A Python tool wcst_import is provided for this purpose. This tool is based on two concepts: (1) Recipes and (2) Ingredients. The recipe defines how data files are combined into a coverage. All information needed to create a data cube are specified in an ingredient file. This is a JavaScript Object Notation (JSON) file based on a recipe which translates the files and information specified in the ingredient file into the data cubes. If wcst_import is run again with a different set of files to be imported, the data cube will be updated at the correct positions.

2.3.2 Rasdaman architecture

The Rasdaman storage concept relies on a separate data storage. The raster data is stored in the file system and the metadata in a separate database. The client-server system can be summarized as a four-layer architecture as shown in Figure 6.

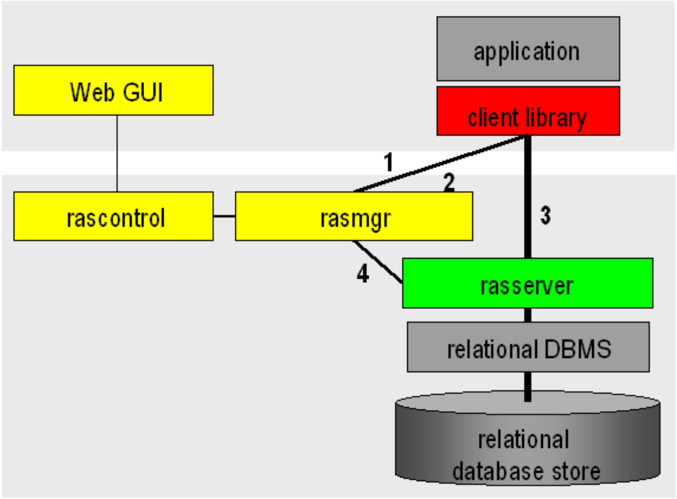


Figure 6: Rasdaman architecture. Source [16].

The foundation is formed by a conventional relational DBMS, which allows efficient storage of large volumes of data. The second layer is the Rasdaman server (rasserver), which provides various functions such as an interface to the relational database, metadata management and query processing. Furthermore, the server interacts with a Rasdaman manager (rasmgr). This manager handles tasks such as security functions, authentication, or multi-user operation by allocating requests to different Rasdaman servers. The fourth layer is the client. An application initially builds a connection to the Rasdaman manager, which then establishes communication between a server and the application.

2.4 Google Earth Engine

In 2010, Google Inc. launched the development of a platform called Google Earth Engine (GEE), offering cloud computations for EO products, initially focusing on forest monitoring using satellite imagery, but later expanding to a variety of applications related to Earth Observation [4]. These include case studies such as Map of Life, Global Surface Water, or Collect Earth [17]. There exist a series of applications in the field of Earth surface analysis [18] but very few suitable for our purposes [19]. The Google Earth Engine is free for research, education, and nonprofit use. Since the platform is a browser-based IDE (Integrated Development Environment), no separate software needs to be downloaded and maintained. To use GEE, a JavaScript-based code editor is provided. Furthermore, data can be requested and analyzed using the Earth Engine (EE) Python API. To process the data Google infrastructure is provided consisting of a large pool of servers, co-located with the data that allows for fast data processing. In addition to the cloud computational capabilities, GEE offers an exhaustive catalog of remote sensing datasets including multispectral, radar, aerial, climate, land cover and vector data including data from satellite missions such as Landsat, Sentinel, MODIS as well as high-resolution imagery data sets [20]. The data is updated and expanded daily. When working with data from the GEE data catalog, three dataset types have to be distinguished: features, images and collections. A feature *(ee.Feature)* is a geometric object containing a list of properties. Images *(ee.Image)* are like features but may include several bands. A combination of features or images is called collection *(ee.ImageCollection)*. Machine Learning is supported via EE API methods and export and import functions for TensorFlowRecord files. The EE API provides methods such as *ee.Classifier, ee.Clusterer or ee.Reducer*. After performing the analysis, it is then possible to export a resulting *ee.Image* as a GeoTIFF to Google Drive or the local machine.

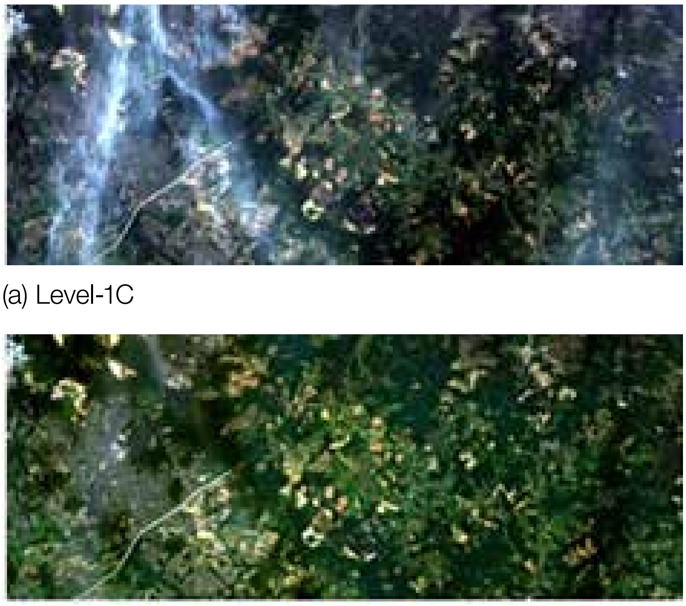
2.4.1 Google Colaboratory

Google Colaboratory is a cloud based hosted Jupyter Notebook service developed by Google specifically for machine learning applications. It allows users to develop, execute and share python code within Google Drive. It provides limited and up to a certain point free access to central processing units (CPU), graphical processing units (GPU), and tensor processing units (TPU). The EE Python API can be easily deployed in a Google Colaboratory notebook.

3. Case Study

3.1 Data Acquisition using Apache Airflow

The starting point of the developed framework is the workflow management implementation using Apache Airflow. To ensure flexibility and scalability, it has been implemented dynamically. Dynamic DAGs are usually better for dynamically loading configuration options or changing operator options. In this case, the DAG is built dynamically based on two configuration files. The first one contains an overview of areas of interest (AOIs), defined by a name, the corresponding polygon, and the satellite from which the data should be requested. The corresponding satellite configuration is stored in another file, where its name, provider, and product type define each satellite. The Open Access Hub offers data starting from Level-1C. As a product type, Level-2A was chosen for this project because only atmospheric corrected and orthorectified data are only available for Level-2A (see Fig. 7).



(a) Level-1C

(b) Level-2A

Figure 7: Two different processing stages of Sentinel-2 data. Source [21].

In addition, the maximum cloud cover percentage that the image may have can be specified as well as it's geometric resolution. This satellite provides a set of 13 spectral bands spanning from the visible (4 bands) and near infrared (6 bands) to the shortwave infrared (3 bands) with a resolution of 10m, 20m, 60m depending on the wavelength. As test examples, we defined six different areas from which we only request Sentinel-2 bands having a geometric resolution of 10m or 20m and a cloud cover percentage less than ten percent. For each satellite provider, one DAG is created. The resulting structure can be seen in Figure 8.

The DAG consist of as many tasks as specified AOIs. Since Airflow allows the execution of parallel tasks, the graph adopts a tree structure. However, since Copernicus Hub can only handle two server requests at a time, the node CopernicusHub_start has only two child nodes. In the user interface (UI), the blue nodes represent task groups, which are a UI grouping concept and useful for creating repeating patterns. In each task group, the same sequence of tasks is executed accordingly to the different areas. The tasks are shown in Figure 9.

At the beginning, it is checked whether the corresponding data entry already exists in the database. If so, the execution of the task group is terminated. Otherwise, it is verified whether the data is available on the platform. For this purpose, the modified dhusget.sh script is executed. If no data is available, no further tasks of the group will be executed. If the data is available, the task *checkDownload* passes the output of the executed script to the next task by using a cross-communications message (xcom). The script response may look like *id('2b17b57d-fff4-4645-b539-91f305c27c69')* which represents an individual entity given by the UUID (Universally Unique Identifier) *'2b17b57d-fff4-4645-b539-91f305c27c69'.* The next step is to determine whether the data is available for download or not. The availability of online products on the Data Hub can be identified by means of an OData query. If the data is online, it is downloaded directly. Otherwise, the download request automatically triggers the request for restoring the data from

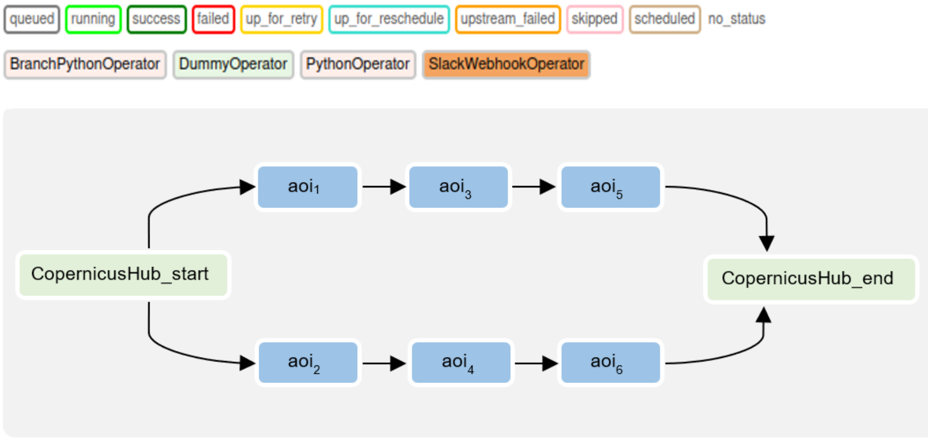


Figure 8: Dynamically created DAG based on the two configuration files.

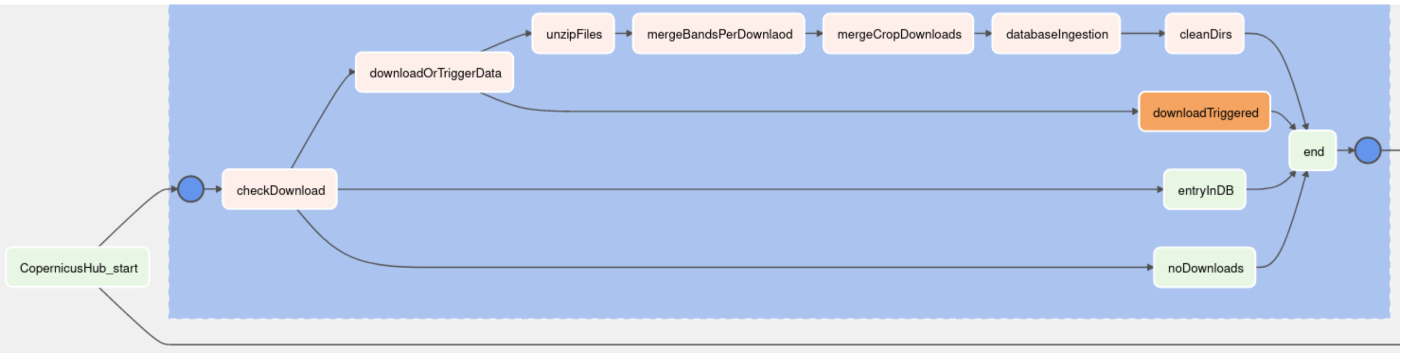


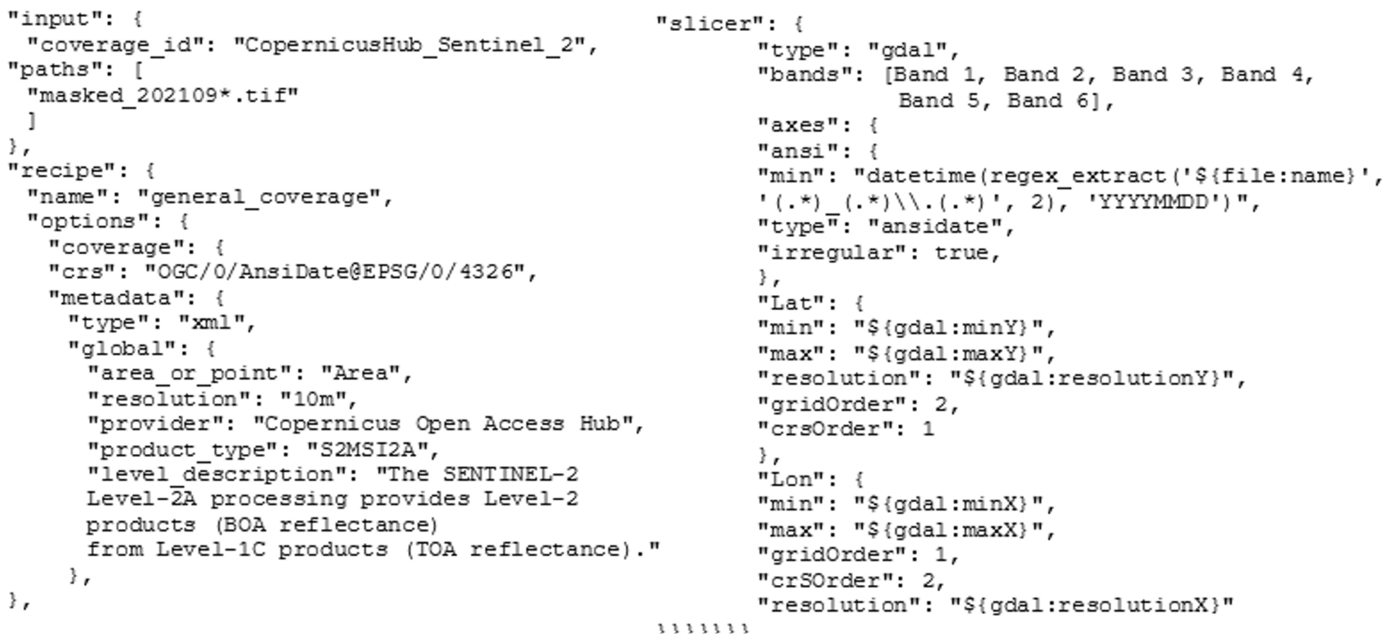
Figure 9: The actual tasks to be performed per area organized into Task Groups.

the archive. Restored data is then kept online for at least three days. In case of a download, the file is then checked for completeness. For this, the MD5 (Message-Digest Algorithm 5) checksum provided by the Data Hub is compared with the MD5 value of the download. Sentinel-2 data is provided in the form of data packages (tiles) with a size of 100x100 km2. Therefore, it may happen that several zip files are downloaded for one query which consequently have to be merged. The result is one image per geometric resolution cropped to the AOI. The last step is the insertion of the images into the Rasdaman database using the tool *wcst_import.* The corresponding recipe is dynamically filled with all file related information and is shown in Listing 1.

This recipe contains all the necessary information, such as the size of the image data, the associated coverage ID and the resolution. The input section contains information about the source files to be considered. The structural information such as the data cube type, the timestamp and metadata are part of the recipe section.

3.2 Analysis using Google Earth Engine

To analyze this data, artificial intelligence (AI), in particular machine learning (ML) is applied, aiming at building and improving a generalizing system based on relevant data that automatically identifies patterns of data not previously introduced. There exist different types of machine learning algorithms such as supervised, unsupervised, semi-supervised, and reinforcement learning. A number of studies have already been conducted on the application of supervised and unsupervised methods with respect to our use case [22]. In this paper, we provide an overview of two machine learning algorithms, namely Random Forest (RF) and Classification and Regression Trees (CART) provided by GEE. The decision tree algorithm CART developed by [23] provides decision trees for classification, as well as for regression. The key to this algorithm is to find an optimal binary separation. For this purpose, a univariate binary decision tree is built by the algorithm. The Gini index is used as an impurity measure and minimal cost-complexity pruning is



```
"input": {
  "coverage_id": "CopernicusHub_Sentinel_2",
"paths": [
  "masked_202109*.tif"
  ]
},
"recipe": {
  "name": "general_coverage",
  "options": {
    "coverage": {
    "crs": "OGC/0/AnsiDate@EPSG/0/4326",
    "metadata": {
      "type": "xml",
      "global": {
        "area_or_point": "Area",
        "resolution": "10m",
        "provider": "Copernicus Open Access Hub",
        "product_type": "S2MSI2A",
        "level_description": "The SENTINEL-2
        Level-2A processing provides Level-2
        products (BOA reflectance)
        from Level-1C products (TOA reflectance)."
      },
},
```

```
"slicer": {
        "type": "gdal",
        "bands": [Band 1, Band 2, Band 3, Band 4,
                  Band 5, Band 6],
        "axes": {
        "ansi": {
        "min": "datetime(regex_extract('${file:name}',
        '(.*)_(.*)\\.(.*)', 2), 'YYYYMMDD')",
        "type": "ansidate",
        "irregular": true,
        },
        "Lat": {
        "min": "${gdal:minY}",
        "max": "${gdal:maxY}",
        "resolution": "${gdal:resolutionY}",
        "gridOrder": 2,
        "crsOrder": 1
        },
        "Lon": {
        "min": "${gdal:minX}",
        "max": "${gdal:maxX}",
        "gridOrder": 1,
        "crSOrder": 2,
        "resolution": "${gdal:resolutionX}"
}}}}}}}
```

Listing 1: Sentinel-2 L2A recipe.

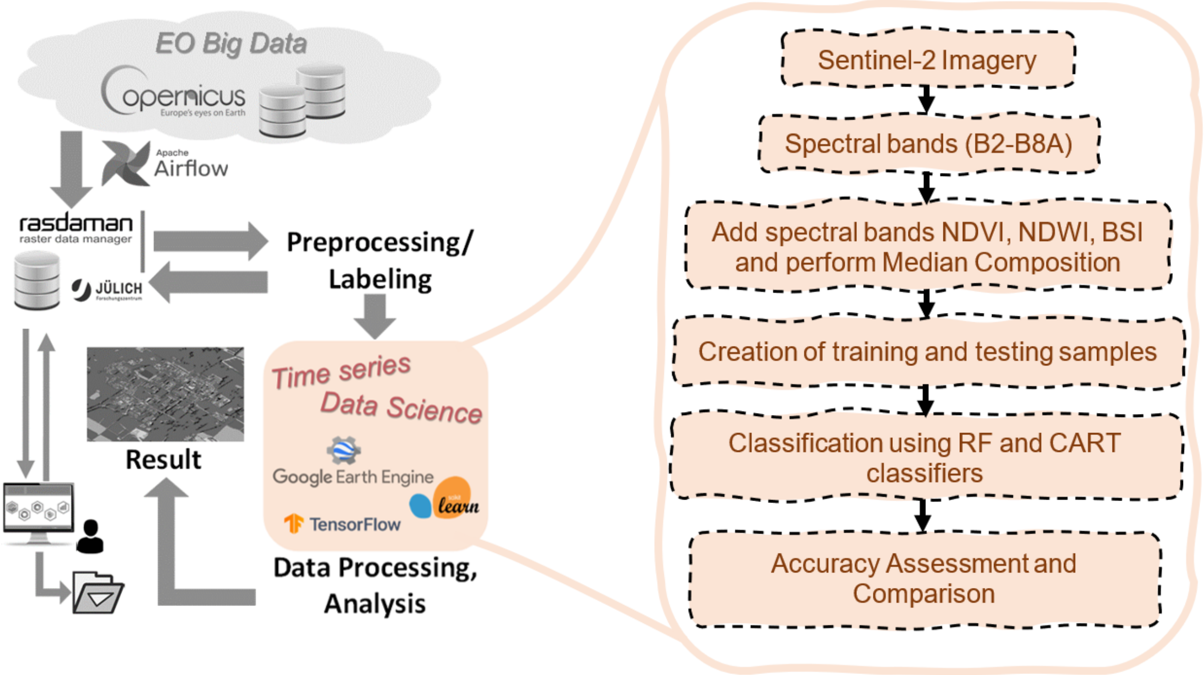


Figure 10: Methodology for classification on the GEE platform using the machine learning algorithms RF and CART.

used after the tree is built. The GEE library provides the technique classifier.smileCart. Random Forest [24] is a well-known supervised machine learning method, which is based on decision trees and is used for classification and regression tasks. In Random Forests, many decision trees are created randomly based on so-called bootstrap datasets. Each tree makes individual decisions on its own. Classification is done by repeatedly applying a learning procedure to bootstrap samples of the training data and then aggregating the individual results. Since the individual decision trees can be built and trained quickly and in parallel, the overall algorithm also trains fast. In this study GEE's technique classifier.smileRandomForest was used. Their performance is compared using accuracy assessment. The methodology used for training the classifier is shown in Fig. 10.

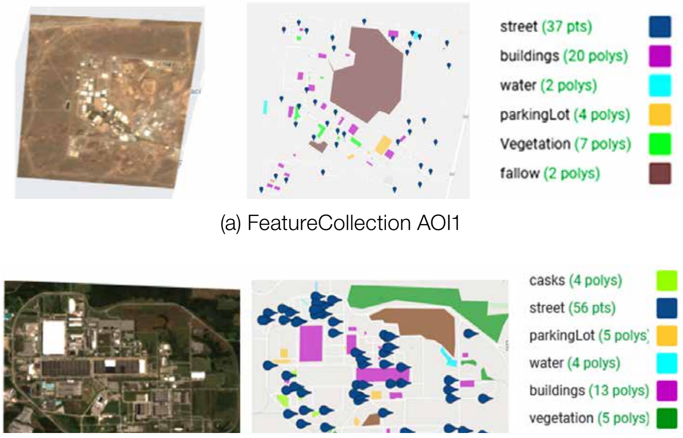
As previously mentioned, Sentinel-2 imagery with a cloud cover of less than ten percent is collected for six different AOIs. The median was used to compose the Sentinel-2 images for the entire years of 2017, 2019 and 2021. The bands B2-B8A as well as the normalized difference water index (NDWI), the normalized difference vegetation index (NDVI) and the bare soil index (BSI), calculated as follows

$$NDVI = \frac{(NIR-Red)}{(NIR+Red)}, NDWI = \frac{(Green-NIR)}{(Green+NIR)}, BSI = \frac{((Red+SWIR)-(NIR+Blue))}{((Red+SWIR)+(NIR+Blue))}$$

with near infrared (NIR), and short-wave infrared (SWIR), were used as input features.

Since the selected machine learning methods are supervised algorithms and the Sentinel-2 data do not contain labels, training data must be collected. Two different methods are compared for this purpose. On the one hand, *FeatureCollections* for two AOIs are created manually using the GEE drawing tool and on the other target labels extracted from two different tagged land cover datasets [25, 26] provided by GEE were used for all six AOIs. An overview of available labeled land cover datasets can be found in [27].

For the manually extracted features, 72 feature polygons were selected distributed throughout the first study area and 89 for the second, covering six different classes, seven respectively. The features and the corresponding study areas are shown in Figure 11.



(a) FeatureCollection AOI1

(b) FeatureCollection AOI2

Figure 11: Manually created FeatureCollections consisting of 72 and 89 features

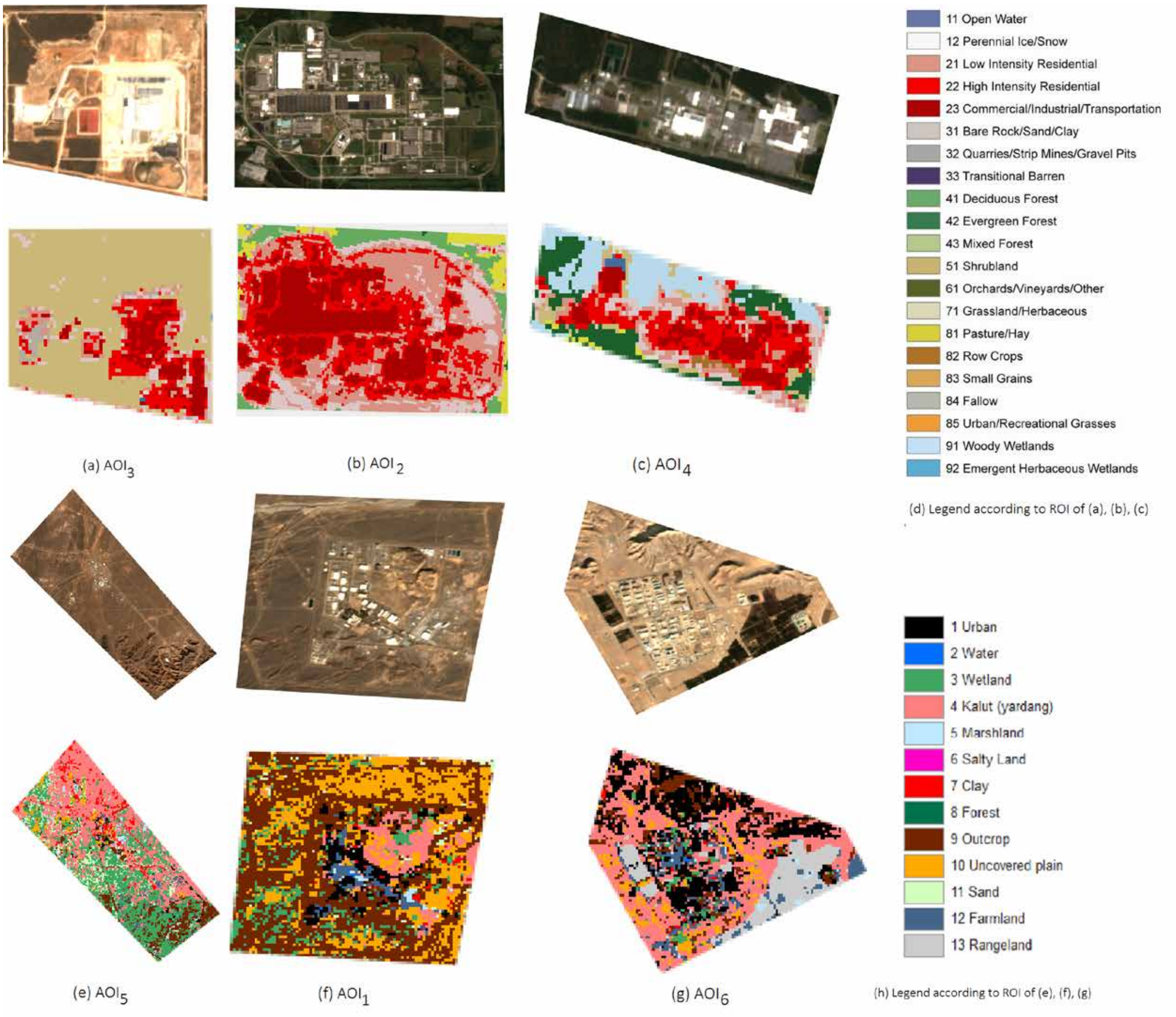


Figure 12: Class labels based on two different land cover datasets provided by GEE.

For the first area, a land cover of GEE was used, which spans eight different epochs and contains 20 different land cover classes. The 2019 release was used in this study. A single mosaic dataset from 2017 containing 13 classes was used as the second land cover. The land cover maps according to their AOI are shown in Figure 12.

Training and validation datasets were generated using a stratified random sampling approach which were then used to build, train, and classify a RF and a CART classifier. In this study, a total of 170 trees combined with a minimum leaf population of 3 and a fraction of input to bag per tree of 0.9 yielded good results for RF. In terms of the CART algorithm, the best cross validation factor was determined to be 5.

3.3 Results

Figure 13 shows the classification results obtained by both classifiers based on the manually created *FeatureCollections.*

Figure 13 (b) shows that for the year 2017 the classification by CART resulted in a misclassification of fallow land to asphalt roads and parking lots in a less extend. In addition, much of the fallow land was misclassified as vegetation. Water was incorrectly declared as vegetation. Furthermore, road sections were classified as buildings. Also in the 2019 case, there were more misclassifications of CART compared to RF. Again, vegetation was misclassified as parking lot. Here, the wasteland has been identified as roads and parking lots. In both cases, Random Forest performed well compared to the second algorithm. Almost all buildings and vegetation are correctly classified. But also, here some road sections were misclassified. Figure 14 shows the classification maps of all six AOIs where the target classes were assigned to the predefined land cover datasets.

Since pasture/hay areas had very few pixels and thus insufficient for efficient training, this class was misclassified as a developed class. Furthermore, several issues were encountered with the classification of woody wetlands and shrub/shrub classes, which were classified as forest in the first case and developed area in the second case.

Comparing the classification results with the underlying land cover map, which is shown in Figure 12, the RF algorithm again provides significantly better results than CART. The effectiveness of the different classifiers was evaluated based on accuracy. The most used metrics for evaluating

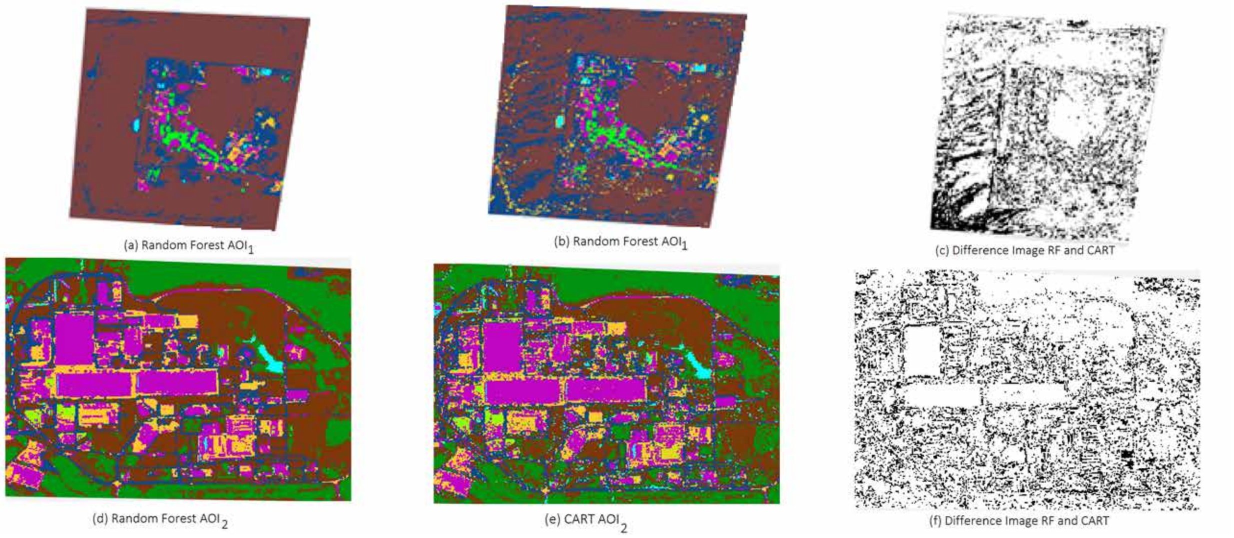


Figure 13: Classification maps using RF and CART classifiers for the years 2017 of AOI$_1$ and 2019 of AOI$_2$.

the accuracy and effectiveness of each classifier is the overall accuracy (OA) representing the percentage of correctly classified instances out of all instances and the Kappa coefficient used to test reliability.

The RF classifier outperformed the CART classifier with an average overall accuracy of 91.40 % in contrast to 74.57 %. The average kappa coefficients for RF, and CART classifiers were 0.85 and 0.64, respectively.

In addition, there are a few issues that should be investigated if the proposed method is applied not only for verification purposes. First, the spatial resolution of the Sentinel-2 satellite is limited to 10m, which results in mixed pixels containing different surface classes. This has an impact on the FeatureCollection creation and the classification. If these images had been used for feature selection, the low resolution would have caused problems regarding its concrete location. Therefore, a basemap of high-resolution reference imagery available directly within GEE was used. The downside is that this is a mosaic of images, with no information available on the date of acquisition. On the other hand, the classification performance is not accurate enough for these pixels. So, existing land cover maps were used for feature collection. However, since these maps capture only the land cover situation for one specific year the application for verification purposes is limited.

4. Conclusion

Today satellite imagery is an integral part of the IAEA's monitoring and verification efforts. The images can be used in a variety of ways to confirm that a country's nuclear facilities are in compliance with the specifications in internationally signed treaties and declarations made by the member states. Remote sensing data is for example well suited for planning on-site inspections and recognizing as well as monitoring features of interest within nuclear facilities in order to detect suspicious activities at an early stage. Thanks to the ongoing development of new satellite systems and the amount of data which will further increase in the coming years, even more applications are conceivable. Many providers offer their satellite data at low cost or even for free. For example, initiatives such as the Copernicus program, the European Union's Earth observation program, have revolutionized the market. Remote sensing has thus arrived in the Big Data era posing challenges regarding data management, processing, and analysis. The ever growing quantity of data and its properties require the further automation of processing and the development of quantitative techniques that have the potential to improve upon traditional techniques in terms of computational cost, reliability and objectivity. Several novel technologies have been developed to meet these challenges . In this research, three tools namely Apache Airflow, Rasdaman and google Earth Engine have been utilized to develop a

	AOI1		AOI2		AOI3		AOI4		AOI5		AOI6	
	CART	RF	CART	RF	CART	RF	CART	RF	CART	RF	CART	RF
Overall Accuracy (%)	83.07	91.02	76.63	98.34	68.95	86.65	72.05	89.78	73.67	92.61	73.05	89.98
Kappa Coefficient	0.65	0.76	0.70	0.92	0.63	0.81	0.59	0.85	0.60	0.89	0.64	0.86
Correct Area (%)	78.35	84.76	58.12	81.39	61.86	73.95	53.45	64.77	55.13	72.24	57.16	75.12

Table 1: Overall accuracy and Kappa statistic of CART and RF classifiers based on land cover tagged maps.

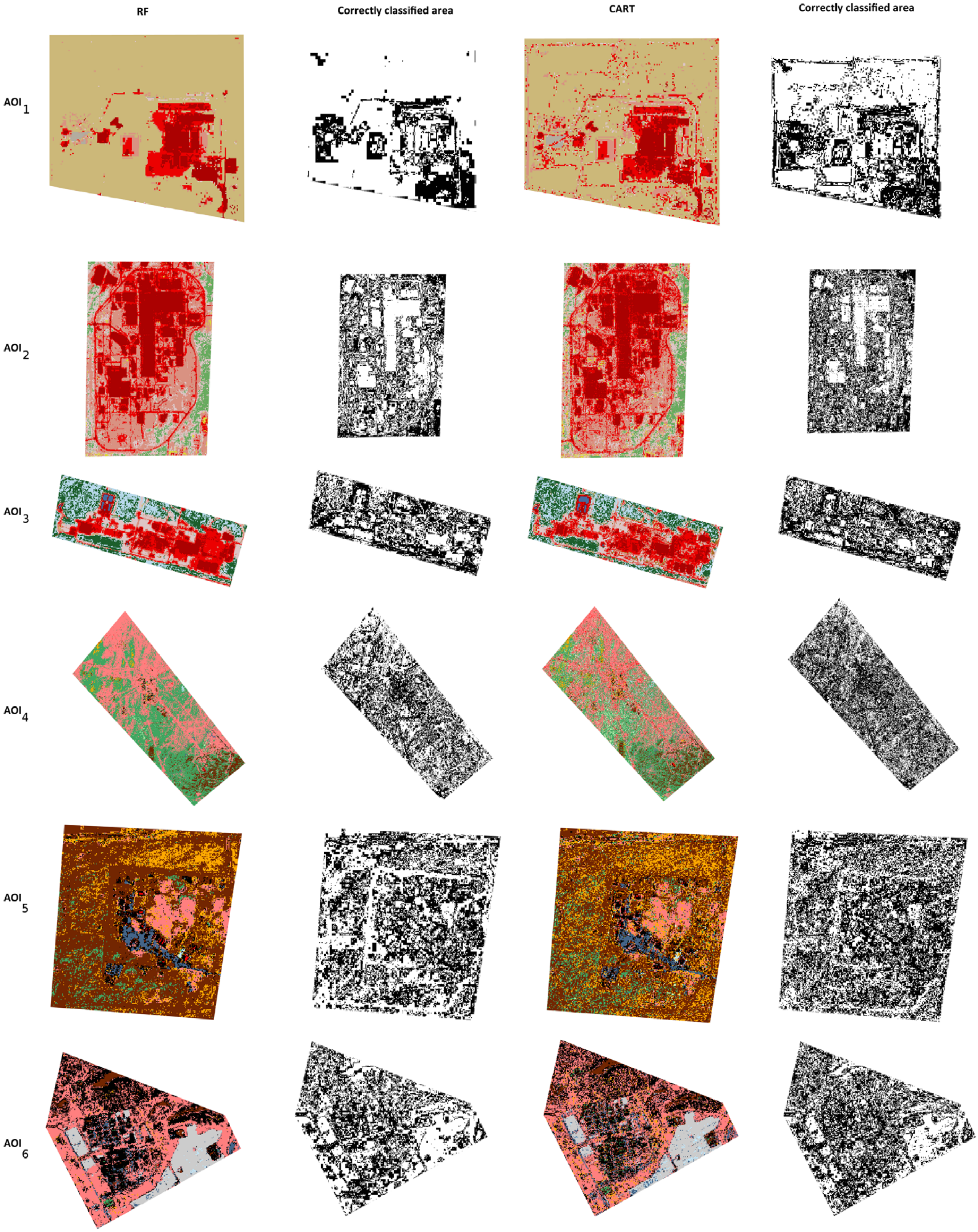


Figure 14: Classification maps using RF and CART classifiers based on two different predefined land cover classes provided by GEE. The corresponding land cover map legend is shown in Figure 13.

semi-automated procedure for collecting, storing, processing and analyzing satellite images. The potential of the developed framework is tested using case studies concerning nuclear fuel cycle related sites. Hereby the objective is to classify land cover use, as these features provide essential information for recognizing and monitoring for example changes of the operational status, constructions of new buildings and roads, plant expansions, etc.

It has been shown that Apache Airflow enables the optimization of data processing and workflow management processes and can be easily adapted in Python. The well-established array database management system Rasdaman offers the benefits of using OGC standards and storing as well as functionality scalability, among others. Finally, the potential of GEE in terms of data analysis capabilities was evaluated. The platform offers powerful capabilities in handling large volumes of remote sensing imagery and provides several machine learning algorithms. Here, the use of Google Earth Engine was demonstrated using two supervised learning algorithms, namely Random Forest and Classification and Regression Trees, provided along with GEE to classify Sentinel-2 images. To evaluate the classifiers performance, accuracy assessment was carried out. It has been shown that RF is the most suitable classifier for any given scenario reaching an overall accuracy of 91.40%. In a next step, we will evaluate the application of unsupervised learning methods that do not require training data with respect to the classification of Sentinel-2 images.

5. Acknowledgements

The project on which this paper is based was funded by the German Foundation for Peace Research under the funding code FP 09/17 – SP 05/05-2017 and by the German Federal Ministry for Economic Affairs and Climate Action under the funding code 02W6279. The responsibility for the content of this presentation lies with the authors

6. References

[1] Quevenco R; *Completing the picture: using satellite imagery to enhance IAEA safeguards capabilities;* IAEA Bulletin 2016 57-2; 2016; pp. 24-25–405.

[2] Apache Airflow Documentation – Version 2.2.4; URL: https://airflow.apache.org/docs/apache-airflow/stable/index.html.

[3] Baumann P, Dehmel A, Furtado P, Ritsch R, Widmann N; *The Multidimensional Database System RasDaMan*; Sigmod Record 27; 1998; pp. 575-577; DOI: 10.1145/276304.276386.

[4] Gorelick N, Hancher M, Dixon M, Ilyushchenko S, Thau D, Moore R; *Google Earth Engine: Planetary-scale geospatial analysis for everyone;* Remote Sensing of Environment; 2017.

[5] Glaser A, Niemeyer I; *Nuclear Monitoring and Verification Without Onsite Access.* In: Toward Nuclear Disarmament: Building up Transparency and Verification; Ed. by M. Göttsche and A. Glaser; German Federal Foreign Office, 2021, pp. 86–115; DOI: 10.18154/RWTH- 2021- 04949; URL: https://publications.rwth-aachen.de/record/819269.

[6] Jo S.Y, Shin J.S; *Status and Application of Spatial Information Technology as a Nuclear Verification Tool;* Transactions of the Korean Nuclear Society Spring Meeting Pyeongchang, Korea; 2010.

[7] Zhang H; *Strengthening IAEA safeguards using high-resolution commercial imagery*; Symposium on International Safeguards: Verification and Nuclear Material Security, Vienna, Austria; 2001.

[8] Castriotta A.G, Volpi F; *Copernicus Sentinel Data Access Annual Report 2020*; URL: https://sentinels.copernicus.eu/web/sentinel/-/copernicus-sentinel-data-access-annual-report-2020/1.0?redirect=%2Fweb%2Fsentinel%2Fsentinel-data-access.

[9] Dehmer M, Emmert-Streib F, Chen Z, Li X, Shi Y; *Mathematical Foundations and Applications of Graph Theory*; 2017; ISBN: 9783527339099.

[10] *Copernicus*; URL: https://www.copernicus.eu/en.

[11] *Copernicus in detail*; URL: https://www.copernicus.eu/en/about-copernicus/copernicus-detail.

[12] *Copernicus Open Access Hub* - OData API; URL: https://scihub.copernicus.eu/userguide/ODataAPI

[13] *Copernicus Open Access Hub* – Batch Scripting; URL: https://scihub.copernicus.eu/userguide/BatchScripting

[14] Baumann P; *A Database Array Algebra for Spatio-Temporal Data and Beyond*; Proceedings of the 4th International Workshop on Next Generation Information Technologies and Systems (NGITS), pp. 76 – 93; 1999.

[15] *Rasdaman Documentation* – Query Language Guide; URL: https://doc.rasdaman.org/04_ql-guide.html

[16] *Rasdaman Documentation* – Server Architecture; URL: https://doc.rasdaman.org/02_inst-guide.html#server-architecture.

[17] *Google Earth Engine – Case Studies*; URL: https://earthengine.google.com/case_studies/.

[18] Canty M.J, Nielsen A.A. Conradsen K, Skriver H; *Statistical Analysis of Changes in Sentinel-1 Time Series on the Google Earth Engine*; Remote Sens. 2020; pp. 12, 46; DOI: 10.3390/rs12010046.

[19] Rutkowski J, Canty M.J, Nielsen A; *Site Monitoring with Sentinel-1 Dual Polarization SAR Imagery Using Google Earth Engine*; Journal of Nuclear Materials Management; 2018; pp. 48-59.

[20] *Google Earth Engine – Data Catalogue*; URL: https://developers.google.com/earth-engine/datasets/catalog

[21] Ehlert I, Schweitzer C; *Copernicus für das Umweltmonitoring – Eine Einführung*; p. 62; 2018; URL: https://www.bsh.de/DE/PUBLIKATIONEN/_Anlagen/Downloads/Meer_und_Umwelt/Weitere_Publikationen/Copernicus-fuer-das-Umweltmonitoring.html.

[22] Rutkowski J, Niemeyer I; *Remote Sensing Data Processing and Analysis Techniques for Nuclear Non-proliferation*; In: Niemeyer I, Dreicer M, Stein G (eds); Nuclear Non-proliferation and Arms Control Verification, Springer; Mar. 2020; pp. 339–350; ISBN: 978-3-030-29536-3; DOI: 10.1007/978-3-030-29537-0_23.

[23] Breiman L, Friedman J.H, Olshen R.A, Stone, *C.J; Classification And Regression Trees;* Routledge; 1984; DOI: 10.1201/9781315139470.

[24] Breiman L; *Random Forests; Machine Learning 45*; 2001; pp. 5–32; DOI: 10.1023/A:1010933404324.

[25] Ghorbanian A, Kakooei M, Amani M, Mahdavi S, Mohammadzadeh A, Hasanlou M; *Improved land cover map of Iran using Sentinel imagery within Google Earth Engine and a novel automatic workflow for land cover classification using migrated training samples*. *ISPRS Journal of Photogrammetry and Remote Sensing*, 167, 276-288; 2020; DOI: 10.1016/j.isprsjprs.2020.07.013.

[26] Dewitz J and U.S. Geological Survey; *National Land Cover Database (NLCD) 2019 Products (ver. 2.0, June 2021): U.S. Geological Survey data release; 2021*; DOI: 10.5066/P9KZCM54.

[27] *Google Earth Engine – Datasets tagged landcover in Earth Engine*; URL: https://developers.google.com/earth-engine/datasets/tags/landcover.