Sentinel-2 change detection

Technical Documentation

Reinforcing the AI4EU Platform by Advancing Earth Observation Intelligence, Innovation and Adoption

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 101016798.

Table of Contents

1 Introduction

The AI4Copernicus consortium provides a set of services and resources made available from the Security, Agriculture, Energy and Health communities for the open calls winners.

The development of these bootstrapping services aimed to reduce the time and resources of the bidders in different tasks as data access (EO and ancillary data), pre-processing, labelling datasets, ML algorithm definition. The AI4Copernicus consortium support to the bidders allows to address open calls winner's effort on the development of innovative services based on AI.

The Security Bootstrapping services and resources have been developed considering the objective of the open calls, which was *"the development of EO applications leveraging on AI algorithms to detect, identify and/or predict features and events in response to current Security challenges. The applications are expected to exploit EO data, in conjunction with relevant collateral data sources as suitable (e.g. geolocalization, AIS, statistical data, climate/weather, in-situ sensors…) with the use of the latest technologies, also contributing to shape the development of a Digital Twin Earth (DTE) for Security"*.

The following section describe the Sentinel-2 change detection service, developed by SatCen in the frame of the Security domain.

2 Sentinel-2 change detection

2.1 Summary

The Sentinel-2 Change Detection pipeline is available as a dockerized application (see annex) that can be executed in any environment with a properly configured Docker client.

This pipeline computes (and classifies) the changes using as input a pair of S2-L2A products by using the Change Vector Analysis approach.

Several parameters are exposed (e.g. resolution, bands, number of change classes), including when possible, a default value to facilitate the use by non-expert users.

2.2 Input

The input of this pipeline is a pair of Sentinel-2 L2A products in their native SENTINEL-SAFE format (zipped or unzipped products are both supported). The inputs shall correspond to the same tile (e. g. same relative orbit).

2.3 Exposed Parameters

Table 1. Exposed parameters in Sentinel-2 change detection

Parameter	Valid values	Default Value
Resolution: output resolution in meters. The minimum value recommended is the default value (10m for IW products and 5 m for SM products)	Any in meters	10m (IW), 5m (SM)

Bands: list of S2 bands that are going to be used for computing the changes.	Any combination of S2 L2A bands	B2,B3,B4,B8
AoI: Area of Interest	WKT polygon or path to vector file	None
Projection: output projection	Any supported by SNAP	WGS84
NumberOfClasses: number of classes in which changes will be automatically classified.	Any integer	4
ReferenceVector: Reference vector to be used for computing the angle in CVA methodology.		1,0,0,…
Level of confidence:	Any float value lower than 100.	99,99
Output format: output format	Supported by GDAL and SNAP	GeoTiff

2.4 Processing

The standard approach when computing changes is simplified in figure below.

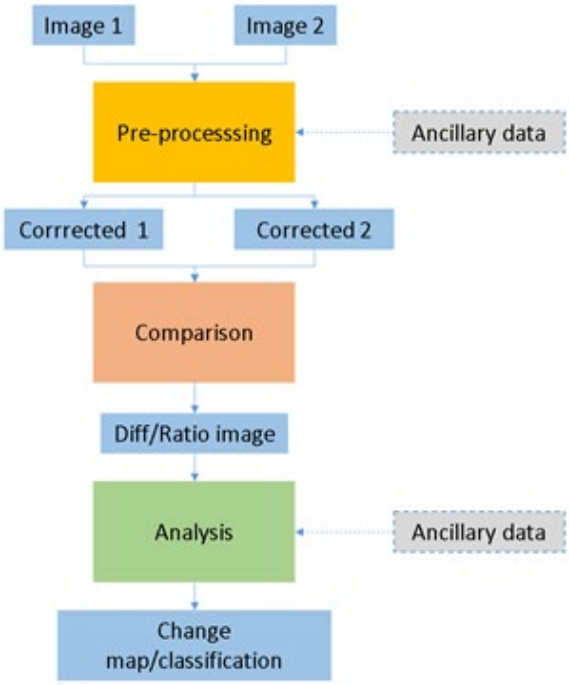


Figure 1. Sentinel-2 Change detection processing flow.

The S2 change detection pipeline developed implements the following steps:

1. Pre-processing:

. Resampling: Bands that are needed for the processing are resampled to the common selected resolution. These bands include the bands selected by the user and the scene classification bands. This step is performed using SNAP.

a. Radiometric correction/histogram matching: in order to minimize errors caused by not accurate radiometric corrections (including atmospheric), a relative radiometric correction is applied to one of the inputs. For this, it is computed a linear regression using as references the pixels with less changes after removing the ones affected by clouds or where the land cover is different.

b. Crop the image to the AoI.

c. Generate cloud masks.

2. Computation of normalized vector of differences in pre-processed images
3. Compute the module of the vector and angle with respect to the reference vector.
4. Compute binary mask of changes by assuming that difference between bands follow gaussian distributions and the module of the change follows a chi-squared distribution:
   - After having applied the radiometric correction/histogram matching in 1.b, it is assumed that the difference of the same band in two images is following a Gaussian distribution. The values that cannot be explained with this distribution are classified as changes.
   - When taking into account the full set of bands selected for the processing, the amplitude of the change is computed with the normalized differences and it is assumed that it follows a chi-squared distribution.
5. Classify changes using K-means algorithm in the polar representation of the vector of differences (amplitude and angle).

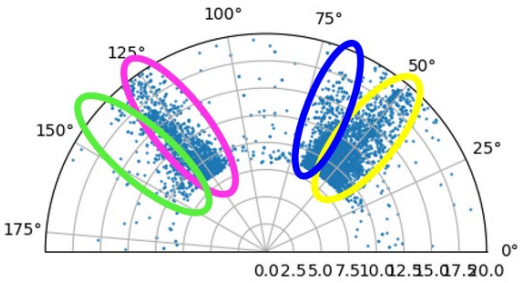


Figure 2. Example of classified changes in the polar representation using K-means.

2.5 Output

The outputs are:

- CVA: GeoTiff image with two bands. The first band is the amplitude of the change and the second band is the angle with respect to the reference vector.
- S2-CD: GeoTiff image with one band with pixel type Byte. It represents the classes of the detected changes.

2.6 How to use

Minimum requirements: 16GB of RAM.

Inside the docker, the pipeline can be found in `/app/pipelines` and can be executed with the following command:

```
S2-CD --input1 "VALUE" --input2 "VALUE" [--bands "XX,XX,XX"] [--AoI "WKT"] [--resolution "VALUE"] [--projection "VALUE"] [--numberClasses "VALUE"] [--referenceVector "VALUE"] [--levelConfidence "VALUE] [--output_format "VALUE"] --outdir "VALUE"
```

It can be also executed with "*docker run*" taking into account that a volume has to be mounted in order to write on it the output file so it is accessible at the end of the processing.

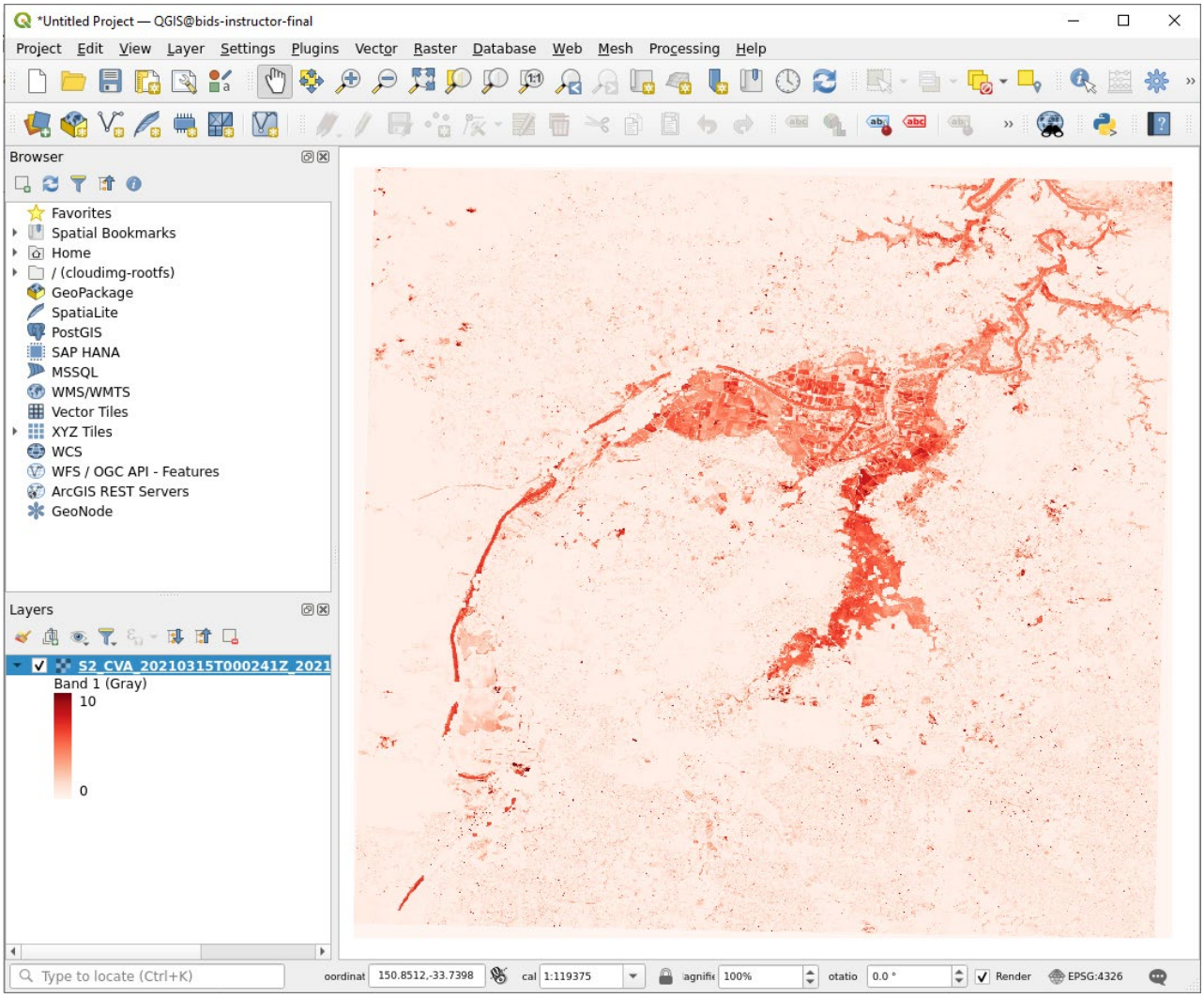
```
docker run -v [local_path]:[container_path] DOCKER_IMAGE  S2-CD --input1 "VALUE" --input2 "VALUE" [--bands "XX,XX,XX"] [--AoI "WKT"] [--resolution "VALUE"] [--projection "VALUE"] [--numberClasses "VALUE"] [--referenceVector "VALUE"] [--levelConfidence "VALUE] [--output_format "VALUE"] --outdir "VALUE"
```

If any customization is needed in the processing graph, it can be found in the docker and could be adapted by the users and executed directly using *gpt*.

2.6.1 Examples

- Computation of S2 change detection to estimate area affected by a flooding that took place close to Sydney in March 2021

```
S2-CD                                                                                -i1 /output/S2A_MSIL2A_20210315T000241_N0214_R030_T56HKH_20210315T020346.zip      -i2 /output/S2A_MSIL2A_20210325T000241_N0214_R030_T56HKH_20210325T022649.zip       -b B3,B4,B8 -r 20 -p "POLYGON((150.6266 -33.49, 150.952 -33.49, 150.952 -33.795, 150.6266 -33.795, 150.6266 -33.49))" -outdir /output/
```


*Untitled Project — QGIS@bids-instructor-final
Project Edit View Layer Settings Plugins Vector Raster Database Web Mesh Processing Help
Browser
Favorites
Spatial Bookmarks
Home
/ (cloudimg-rootfs)
GeoPackage
SpatiaLite
PostGIS
SAP HANA
MSSQL
WMS/WMTS
Vector Tiles
XYZ Tiles
WCS
WFS / OGC API - Features
ArcGIS REST Servers
GeoNode
Layers
S2_CVA_20210315T000241Z_2021
Band 1 (Gray)
10
0
Type to locate (Ctrl+K)
150.8512,-33.7398
1:119375
100%
0.0 °
Render
EPSG:4326

Appendix: docker registry access

A Docker registry is a storage and distribution system for Docker images. It is organised in Docker repositories that contain all the versions published of a specific image. It allows the developers/providers to tag and push their images that can be pulled by the users to run them.

CloudFerro has deployed an instance of Harbor (goharbor.io), which is an open source registry that can be accessed in https://harborai4c.cloudferro.com/ .

Different users have been created for the service providers (with 'Developer' role) and another user for the funded projects with 'Guest' role that allows them to pull the images.

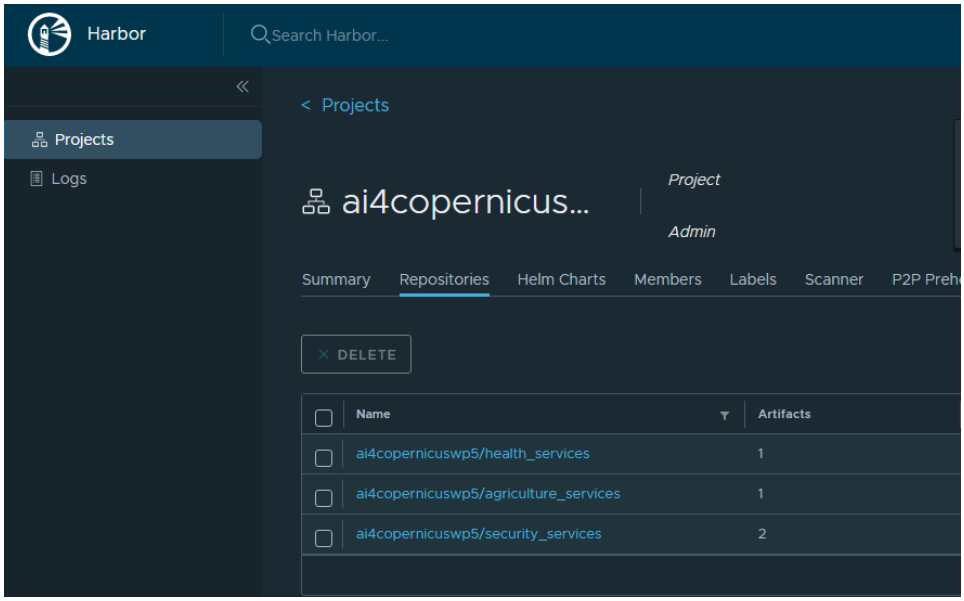


Figure 3. Docker registry screenshot.

The typical steps for pulling and running the services are:

- Login to registry

```
docker login -u=[YOUR_USER] -p=[PASSWORD] harborai4c.cloudferro.com
```

- Pull images (example with security services image)

```
docker pull harborai4c.cloudferro.com/ai4copernicuswp5/security_services:1.0.1
```

- Run a container

```
docker run -it harborai4c.cloudferro.com/ai4copernicuswp5/security_services:1.0.1 bash
```

- Run a container with a volume (local folder mounted in container)

```
docker run -it -v /tmp/example_products/:/output harborai4c.cloudferro.com/ai4copernicuswp5/security_services:1.0.1 bash
```

where /tmp/example_products is a local (Docker host) folder and /output is the folder in the container

- Copy files from/to the container

from Container to Docker Host

`docker cp {options} CONTAINER:SRC_PATH DEST_PATH`

from Docker Host to Container

`docker cp {options} SRC_PATH CONTAINER:DEST_PATH`

where the container can be obtained from `docker ps`