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

D5.2 Bootstrapping services and resources II

Grant Agreement ID	101016798	**Acronym**	AI4COPERNICUS
Project Title	Reinforcing the AI4EU Platform by Advancing Earth Observation Intelligence, Innovation and Adoption		
Start Date	01/01/2021	**Duration**	36 Months
Project URL	https://ai4copernicus-project.eu/		
Contractual due date	30/06/2022	**Actual submission date**	31/08/2022
Nature	DEM = Demonstrator	**Dissemination Level**	PU = Public
Author(s)	M. Lazzarini (SATCEN), O. Barrilero (SATCEN), M. Alexe (ECMWF), V. Baousis (ECMWF), G.Weikmann (UNITN), R. Hall (Equinor), A. Chietera (THA), A. Stoian (THA)		
Contributor(s)			
Reviewer(s)	T. Giannakopoulos (NCSR-D), P. Boza (INSEAD)		

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

Document Revision History *(including peer reviewing & quality control)*

Version	Date	Changes	Contributor(s)
V1.0	31/08/2022	First version of the document	M. Lazzarini, O. Barrilero, M. Alexe, V. Baousis, G. Weikmann, R. Hall, A. Chietera, A. Stoian, D. Hassan

Executive Summary

The deliverable will describe the set of services and resources made available from the Security, Agriculture, Energy and Health communities for the AI4Copernicus 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.

Each service is documented in the section of the domain responsible of its deployment: following a cooperative approach, each service can be used by different domains, if relevant. The description of the services (e.g. the purpose, the input needed and the output produced) referred to the application deployed in the AI4Copernicus infrastructure.

This document is a demonstrator: it is meant to provide valid support for the open calls winner to operate the services.

This document represents an update of D5.1: in this update, the bootstrapping resources made available for the open call winners were refined based on the feedback provided by the first batch of projects. Moreover, this document provides the instructions to access the docker registry where the bootstrapping services are catalogued.

Table of Contents

List of Tables

List of Figures

List of Terms & Abbreviations

Abbreviation	Definition
AC	Atmospheric Composition
ACD	Amplitude Change Detection
AoI	Area of Interest
AQ	Air Quality
ARD	Analysis Ready Data
AI	Artificial Intelligence
AIS	Automatic Identification System
BoA	Bottom of Atmosphere
C3S	Copernicus Climate Change Service
CAMS	Copernicus Atmosphere Monitoring Service
CD	Change Detection
CDS	Climate Data Store
CSV	Comma-Separated Values
CVA	Change Vector Analysis
DEM	Digital Elevation Model
DIAS	Data and Information Access Services
DL	Deep Learning
DTE	Digital Twin Earth
EC	European Commission
ECTL	Extract, Cleanse, Transform, Load
EO	Earth Observation
EU	European Union
GA	Grant Agreement
GAN	Generative Adversarial Network
GDAL	Geospatial Data Abstraction Library
GHG	Greenhouse gas
GPT	Graph Processing Tool
GRD	Ground Range Detected
IoT	Internet of Things
IW	Interferometric Wide
JRC	Joint Research Centre
JSON	JavaScript Object Notation
km	Kilometer
L1C	Level 1C
L2A	Level 2A
LAI	Leaf Area Index
LSTM	Long Short-Term Memory

ML	Machine Learning
MTC	Multi-Temporal Coherence
NN	Neural Network
OSM	Open Street Map
PM	Particulate Matter
S1	Sentinel-1
S2	Sentinel-2
SAR	Synthetic Aperture Radar
SLC	Single Look Complex
SM	Strip Map
SNAP	Sentinel Application Platform
SRGAN	Super-resolution GAN
SRTM	Shuttle Radar Topography Mission
TOPSAR	Terrain Observation Progressive Scans SAR
UTM	Universal Transverse Mercator
WGS	World Geodetic System
WKT	Well-known text
WP	Work Package
XML	eXtensible Markup Language
YAML	YAML Ain’t Markup Language

1 Introduction

1.1 Purpose and Scope

The deliverable describes the set of services and resources made available from the Security, Agriculture, Energy and Health communities forAI4Copernicus open calls winners.

In WP5 activities, each of the above domains oversaw and carried out the bootstrapping of AI4Copernicus with high-impact services and resources coming from the operational needs of users. Each domain task focused on the technical requirements, development of the services and deployment of the resources in the AI4Copernicus infrastructure. First versions of services and resources for each area were provided in M12 and they had incremental updates and refinements until current final status.

D5.2, as an update of D5.1, is a demonstrator: the description of the services referred to the application deployed in the AI4Copernicus infrastructure as support for the open calls winner to operate the services.

1.2 Approach for Work Package and Relation to other Work Packages and Deliverables

From an organisational point of view, the present D5.2 deliverable is a direct outcome of T5.1, T5.2, T5.3 and T5.4. These four tasks correspond to the bootstrapping services and resources of the targeted high-impact domains of AI4Copernicus and each task has a domain responsible: Security (SatCen), Agriculture (University of Trento), Energy (Equinor) and Health (ECMWF). All the tasks were performed with the support of other technical partners as Thales Six, University of Athens, NCSR “Demokritos” and Cloudferro.

The resources of WP5 have been initially described in the proposal and in the Grant Agreement (GA) These resources have been initially refined considering the users operational needs from WP2. The activities of WP5 have been constantly coordinated with the two other technical WPs: WP3 *Architecture and Technical Positioning*, to ensure the deployment of the services within the AI4Copernicus architecture; WP4 *Implementation, customisation, integration and testing,* to ensure the integration of the domain specific services with general services. For a comprehensive overview of the whole infrastructure please refer also to D3.1 and D4.1.

As indicated before, the main objective of the WP is to provide the services for the open calls (WP6). After the initial deployment of the services and their availability through a docker registry, the services described in D5.1 were refined based on an initial set of feedback from the project winners.

1.3 Methodology and Structure of the Deliverable

The deliverable is structured in accordance with the template and guidelines provided by the EC and is organised in an intuitive structure to directly refer the services to their domain. Section 2, 3, 4 and 5 are related to the services provided for Security (Section 2, SatCen), Agriculture (Section 3, University of Trento with the support of Thales Six), Energy (Section 4, Equinor), Health (Section 5, ECMWF). While the domain responsible was in charge to deploy the specific services, other domains can use the different services if considered useful for their own domain (e.g. pre-processing services from Security on Sentinel-1 and Sentinel-2 are useful also for other domains). This approach avoided the duplication of efforts to develop the same service and the simplification for the users.

Each section briefly introduces the approach followed to deploy the services and listed all the services available for the open calls. The description of the services describes the purpose, the structure, the input needed, the output produced and explains how to run them.

Finally, in Section 6, it is described the evolution of the services carried out due to feedback received. This deliverable, even though referred often to D5.1, has to be considered as a stand-alone document, since all the bootstrapping services and resources are described in an exhaustive way in Section 2, 3, 4 and 5 respectively.

1.4 Summary table of services and resources

The Table 1 summarizes the services and resources described in the next sections of this document. The use of the services has been formalized in the Subgrant Agreement between each project and AI4Copernicus. In Article 2 of the SubGrant Agreement, *all bootstrapping services* (see section 1.3.1 < PartB< Annex1 [SubProject] < SubGrant Agreement) *are offered solely during the 16 month period commencing from the Effective Date. Bootstrapping services beyond this term are subject to a separate agreement with the relevant AI4Copernicus partner and lie outside the scope of the SubGrant Agreement and the obligations of the AI4Copernicus Consortium there of.*

In the case of the service "Deep network for pixel-level classification of S2 patches", provided by Thales Six, an additional licence agreement to be signed was provided to the open call winners in order to access and use the related bootstrapping service.

Table 1. Summary table of services and resources

Resource	Summary	Domain	Origin
Sentinel-1 GRD pre-processing	This pipeline processes a S1 GRD product in native format to generate a terrain corrected image representing the calibrated backscatter in GeoTiff format.	Security/ General	AI4Copernicus
Sentinel-1 SLC pre-processing	This pipeline processes a S1 SLC product in native format to generate a terrain corrected image representing the calibrated backscatter in GeoTiff format.	Security/ General	AI4Copernicus
Sentinel-2 pre-processing	This pipeline processes a S2 product in native format to generate a product with a common resolution for all the bands in GeoTiff format. The process allows to apply a land/sea mask and a cloud mask in order to have an output product ready for analysis.	Security/ General	AI4Copernicus

Sentinel-1 Change detection– Amplitude Change Detection and Multi-temporal Coherence	This pipeline processes pairs of S1 SLC products in native format to generate a series of products to assess the changes between both images. These products include: coherence, ACD (Amplitude Change Detection), MTC (Multi-Temporal Coherence) and binary mask of changes.	Security/ General	AI4Copernicus
Sentinel-2 Change Detection	This pipeline computes (and classifies) the changes using as input a pair of S2-L2A products by using the Change Vector Analysis approach.	Security/ General	AI4Copernicus
Vector data of human features	SatCen has pre-processed and ingested in a data model the OSM data and can provide the data as a service in the scope of the project	Security	AI4Copernicus
Deep network for pixel-level classification of S2 patches	This service provides functionality for users to train a custom pixel-level classifier of Sentinel 2 patches. For example users can train a classifier for crop types (corn, sunflower, wheat, etc), land cover (urban vs. natural, water vs land), road extraction (road vs other).	Agriculture/ General	AI4Copernicus
TimeSen2Crop	TimeSen2Crop is a pixel-based dataset made up of more than 1 million crop type samples of Sentinel-2 time series. The dataset includes atmospherically corrected images and reports the snow, shadows, and clouds information per labelled unit, as well as the spectral signature of the samples of nine Sentinel-2 spectral bands at 10m of spatial resolution.	Agriculture	AI4Copernicus
Harmonization of pre-processed Time Series of Sentinel-2 data	The harmonization of pre-processed time series of Sentinel-2 data considers a statistic-based approach that computes the median for each pixel in the different images acquired in a particular month. The pixel composite approach to mosaic generation provides consistent results at large scale, allowing the processing of harmonized acquisitions.	Agriculture	AI4Copernicus
Long Short-Term Memory Neural	The Long Short Term-Memory architecture can be trained using samples selected by the user. The service exploits the data given by the user to train	Agriculture	AI4Copernicus

Network for Sentinel-2	from scratch an LSTM and stores the resulting weights. Several parameters are exposed to allow the user to custom the model		
Pre-Trained Long Short-Term Memory	The pre-trained Long Short Term-Memory architecture is already trained using the TimeSen2Crop database and is available in .h5 format. The service exploits a pre-trained architecture to classify the specified tile harmonized using the monthly composite approach	Agriculture	AI4Copernicus
Energy datasets	Meteorological data: ERA5 Example: offshore wind farms are located (training data) JRC Open Power Plants Database (JRC-PPDB-OPEN) Open data from the floating offshore wind farm, Hywind Scotland	Energy	External references
Probabilistic downscaling of CAMS air quality model data	This service generates high-resolution (currently ~ 10km) air quality maps from low-resolution (~40 - 80km) CAMS model (re)analysis and/or forecast output	Health	AI4Copernicus

2 Security bootstrapping services and resources

2.1 Introduction

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 target of the open calls highlights the need of AI algorithms to detect changes using EO and collateral data. To build the above-mentioned services, data and resources coming from EO repositories (e.g. DIASes) needed to be pre-processed for further advanced processing. The AI4Copernicus consortium therefore decided to contribute with the provision of *Analysis Ready Data (ARD)* and possible training datasets for AI algorithms. The development of these bootstrapping services aimed to reduce the time and resources of the bidders in the data preparation and allow them to focus on the development of innovative services based on AI. Moreover, in-house change detection algorithms were also provided as benchmarks to compare the accuracy of possible change detection algorithms provided by bidders.

The following sections describe the services developed by SatCen in the frame of the Security domain.

2.2 Services

2.2.1 Sentinel-1 GRD pre-processing

2.2.1.1 Summary

The Sentinel-1 GRD pre-processing pipeline is available as a dockerized application that can be executed in any environment with a properly configured Docker client.

This pipeline processes a S1 GRD product in native format to generate a terrain corrected image representing the calibrated backscatter in GeoTiff format. Several parameters will be exposed to the users (e.g. final resolution, polarization and projection). Where possible, a standard value will be set, to facilitate the use by the less-expert users.

2.2.1.2 Input

The input of this pipeline is a Level-1 Ground Range Detected (GRD) product in its native SENTINEL-SAFE [1] format (zipped or unzipped product).

Regarding the acquisition mode, they are supported:

- Stripmap (SM): *Stripmap (SM) mode acquires data with an 80 km swath at slightly better than 5 m by 5 m spatial resolution (single look). The ground swath is illuminated by a continuous sequence of pulses while the antenna beam is pointing to a fixed azimuth angle and an*

[1] https://sentinel.esa.int/web/sentinel/user-guides/sentinel-1-sar/data-formats/sar-formats

approximately fixed off-nadir angle (this is subject to small variations because of roll steering). SM images have continuous along track image quality at an approximately constant incidence angle.[2]

- Interferometric Wide Swath (IW): *the main acquisition mode over land and satisfies the majority of service requirements. It acquires data with a 250 km swath at 5 m by 20 m spatial resolution (single look). IW mode captures three sub-swaths using Terrain Observation with Progressive Scans SAR (TOPSAR). With the TOPSAR technique, in addition to steering the beam in range as in ScanSAR, the beam is also electronically steered from backward to forward in the azimuth direction for each burst, avoiding scalloping and resulting in homogeneous image quality throughout the swath*[3].

2.2.1.3 Exposed Parameters

Table 2. Exposed parameters in Sentinel-1 GRD pre-processing

Parameter	Valid values	Default Value
CalibrationOutput: backscatter convention selected for the output. Each value uses a different reference area. Values ending with DB will be converted to decibel.	Sigma, Gamma, Beta, SigmaDB, GammaDB, BetaDB	Sigma
Polarization	VV,VH… (valid values depend on the specific product)	VV
Speckle-filter (+ specific parameters if needed)	Supported by SNAP	Lee Sigma
AoI: Area of Interest	WKT polygon or path to vector file	None
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)
Land/Sea mask: type of pixels to be removed considering SRTM3Sec. If “Sea” is selected, all values with elevation=0 in SRTM 3Sec will be set to NoData.	Land/Sea/None	None
Projection: output projection	Any supported by SNAP	WGS84
Output format: output format	Supported by GDAL and SNAP	GeoTiff

[2] https://sentinels.copernicus.eu/web/sentinel/user-guides/sentinel-1-sar/acquisition-modes/stripmap

[3] https://sentinels.copernicus.eu/web/sentinel/user-guides/sentinel-1-sar/acquisition-modes/interferometric-wide-swath

2.2.1.4 Processing

The pipeline has been designed using SNAP[4], the common software platform and host for the Sentinel Toolboxes.

One of the components of SNAP, the Graph Processing Tool (GPT), allows to execute SNAP in batch-mode from command-line, with the possibility to create complex workflows using most of the SNAP operators (e.g. Readers, Subset, Reprojection, Band Math, Filters, Calibration…).

In the case of this pipeline, the SNAP graph executed is represented in Figure 1.

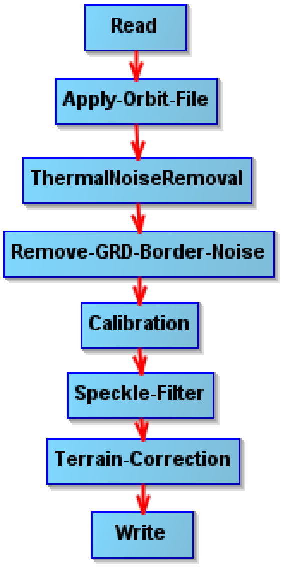


Figure 1. S1 GRD Processing graph

Where:

- Read: the operator in charge of reading a product to the SNAP internal data model.
- Apply-Orbit-File: search, download and apply the corrected information about the orbit that is provided some days after the S1 acquisition to improve the geolocation.
- ThermalNoiseRemoval: removes thermal noise
- Remove-GRD-Border-Noise: remove border noise
- Calibration: convert pixel values to calibrated sigma0, gamma0 or beta0
- Speckle-Filter: applies filter to reduce speckle
- Terrain-Correction: orthorectify the product
- Write: write the output product to the desired format.

(More information about the specific operators can be found in the SNAP help and documentation.)

[4] https://step.esa.int/main/toolboxes/snap/

2.2.1.5 Output

The output is a GeoTiff (by default) terrain-corrected image with one float32 band representing the calibrated backscatter (sigma0, gamma0 or beta0 depending on the selected parameters).

2.2.1.6 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:

```
S1-GRD-preprocess --input “VALUE” [--calibration “VALUE”] [--polarization “VALUE”]
[--speckle “VALUE”] [--AoI “WKT”] [--resolution “VALUE”] [--landseamask “VALUE”]
[--projection “VALUE”] [--output_format “VALUE”] --output_path “VALUE”
```

It can also be launched 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  S1-GRD-preprocess --
input “VALUE” [--calibration “VALUE”] [--polarization “VALUE”] [--speckle “VALUE”]
[--AoI “WKT”] [--resolution “VALUE”] [--landseamask “VALUE”]  [--projection
“VALUE”] [--output_format “VALUE”] --output_path “VALUE”
```

In the case of any customization is needed in graph, it can be found in the source code in the docker and could be adapted by the users and executed directly using the SNAP’s Graph Processing Tool (gpt).

2.2.2 Sentinel-1 SLC pre-processing

2.2.2.1 Summary

The Sentinel-1 SLC pre-processing pipeline is available as a dockerized application that can be executed in any environment with a properly configured Docker client.

This pipeline processes a S1 SLC product in native format to generate a terrain corrected image representing the calibrated backscatter in GeoTiff format. Several parameters are exposed (e.g. final resolution, polarization and projection), including when possible, a default value to facilitate the use by non-expert users.

2.2.2.2 Input

The input of this pipeline is a Level-1 Single Look Complex (SLC) product in its native SENTINEL-SAFE [5] format (zipped or unzipped product are both supported).

Regarding the acquisition mode, they are supported SM and IW (see 2.2.1.2 for a description of these modes).

[5] https://sentinel.esa.int/web/sentinel/user-guides/sentinel-1-sar/data-formats/sar-formats

2.2.2.3 Exposed Parameters

Table 3. Exposed parameters in Sentinel-1 SLC pre-processing

Parameter	Valid values	Default Value
CalibrationOutput: backscatter convention selected for the output. Each value uses a different reference area. Values ending with DB will be converted to decibel.	Sigma, Gamma, Beta, SigmaDB, GammaDB, BetaDB	Sigma
Polarization	VV,VH… (valid values depend on the specific product)	VV
Speckle-filter (+ specific parameters if needed)	Supported by SNAP	Lee Sigma
AoI: Area of Interest	WKT polygon or path to vector file	None
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)
Land/Sea mask: type of pixels to be removed considering SRTM3Sec. If "Sea" is selected, al values with elevation=0 in SRTM 3Sec will be set to NoData.	Land/Sea/None	None
Projection: output projection	Any supported by SNAP	WGS84
Output format: output format	Supported by GDAL and SNAP	GeoTiff

2.2.2.4 Processing

The main pipeline has been designed using SNAP. The SNAP graph executed for IW products is represented in Figure 2. For SM products, the graph varies a little since operators like TOPSAR-Deburst are not needed.

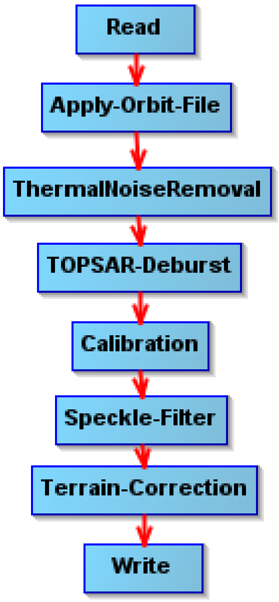


Figure 2. S1 SLC pre-processing graph for IW products.

Where:

- Read: the operator in charge of reading a product to the SNAP internal data model.
- Apply-Orbit-File: search, download and apply the corrected information about the orbit that is provided some days after the S1 acquisition to improve the geolocation.
- ThermalNoiseRemoval: removes thermal noise
- TOPSAR-Deburst: merge the bursts
- Calibration: convert pixel values to calibrated sigma0, gamma0 or beta0
- Speckle-Filter: applies filter to reduce speckle
- Terrain-Correction: orthorectify the product
- Write: write the output product to the desired format.

(More information about the specific operators can be found in the SNAP help and documentation.)

2.2.2.5 Output

The output is a GeoTiff (by default) terrain-corrected image with one float32 band representing the calibrated backscatter (sigma0, gamma0 or beta0 depending on the selected parameters).

2.2.2.6 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:

```
S1-SLC-preprocess --input “VALUE” [--calibration “VALUE”] [--polarization “VALUE”]
[--speckle “VALUE”] [--AoI “WKT”] [--resolution “VALUE”] [--landseamask “VALUE”]
[--projection “VALUE”] [--output_format “VALUE”] --output_path “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   S1-SLC-preprocess  --
input “VALUE” [--calibration “VALUE”] [--polarization “VALUE”] [--speckle “VALUE”]
[--AoI  “WKT”]  [--resolution  “VALUE”]  [--landseamask  “VALUE”]   [--projection
“VALUE”] [--output_format “VALUE”] --output_path “VALUE”
```

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

2.2.3 Sentinel-2 pre-processing

2.2.3.1 Summary

The Sentinel-2 pre-processing pipeline is available as a dockerized application that can be executed in any environment with a properly configured Docker client.

This pipeline processes a S2 product in native format to generate a product with a common resolution for all the bands in GeoTiff format. The process allows to apply a land/sea mask and a cloud mask in order to have an output product ready for analysis.

Several parameters are exposed (e.g. DEM, cloud mask type), including when possible, a default value to facilitate the use by non-expert users.

2.2.3.2 Input

The input of this pipeline is a Sentinel-2 L2A product in its native SENTINEL-SAFE format (zipped or unzipped products are both supported).

They are also supported Sentinel-2 L1c products in its native SENTINEL-SAFE format. In this case, *sen2cor* tool is used internally to process the L2A product before preprocessing.

2.2.3.3 Exposed Parameters

Table 4. Exposed parameters in Sentinel-2 pre-processing.

Parameter	Valid values	Default Value
Resolution: output resolution in meters. The minimum value recommended is the default value (10m)	Any in meters	10
Bands	Any combination of S2 L2A bands: B1,B2, B3, B4, B5, B6, B7, B8, B8A, B9, B10, B11, B12	B2, B3, B4, B8
Land/Sea mask: type of pixels to be removed considering SRTM3Sec. If “Sea” is selected, al values with elevation=0 in SRTM 3Sec will be set to NoData.	Land/Sea/None	None
AoI: Area of Interest	WKT polygon or path to vector file	None

CloudMask: assign NoData to cloudy pixels according to the cloud mask type selected	L1C/L2A/other?/None	None
Upsampling method	Nearest/Bilinear/Bicubic	Nearest
Downsampling method	First,Min,Max,Mean,Median	First
Flag Downsampling method	First,FlagAnd,FlagOr,FlagMedianAnd,FlagMedianOr	First
Projection: output projection	Any supported by SNAP	UTM(Auto)
Output format: output format	Supported by GDAL and SNAP	GeoTiff

2.2.3.4 Processing

The main workflow has been designed using SNAP. The SNAP graph executed is represented in Figure 3.

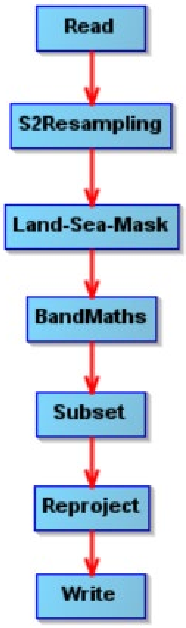


Figure 3. S2 pre-processing graph.

Where:

- Read: the operator in charge of reading a product to the SNAP internal data model.
- S2Resampling: this operator resamples the product to a common resolution.
- Land/Sea Mask: It applies land/sea mask based on srtm 3sec.
- BandMaths: it is used to compute the cloud mask when needed.
- Subset: it filters out the non-desired bands and crop to the AoI.
- Reproject: reprojects to the selected projection.
- Write: write the output product to the desired format.

(More information about the specific operators can be found in the SNAP help and documentation.)

2.2.3.5 *Output*

The output is a GeoTiff (by default) terrain-corrected image containing the selected bands. Depending on the selected parameters, pixels in sea/land or/and cloudy have been set to NoData.

2.2.3.6 *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-preprocess --input “VALUE” [--bands “XX,XX,XX”] [--landseamask “VALUE”] [--
cloudmask “VALUE”] [--AoI “WKT”] [--resolution “VALUE”] [--upsampling “VALUE”] [-
-downsampling “VALUE”] [--projection “VALUE”] [--output_format “VALUE”] --
output_path “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-preprocess --input
“VALUE” [--bands “XX,XX,XX”] [--landseamask “VALUE”] [--cloudmask “VALUE”] [--AoI
“WKT”] [--resolution “VALUE”] [--upsampling “VALUE”] [--downsampling “VALUE”] [--
projection “VALUE”] [--output_format “VALUE”] --output_path “VALUE”
```

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

2.2.4 Sentinel-1 Change detection– Amplitude Change Detection and Multi-temporal Coherence

2.2.4.1 *Summary*

The Sentinel-1 Change Detection resource is available as a dockerized application that can be executed in any environment with a properly configured Docker client.

This pipeline processes pairs of S1 SLC products in native format to generate a series of products to assess the changes between both images. These products include:

- the coherence (the amplitude of correlation between the images),
- the ACD (Amplitude Change Detection), which is a RGB composite of the backscatter of the input images
- the MTC (Multi-Temporal Coherence), which is a RGB composite of the backscatters and the coherence
- binary mask of changes

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

2.2.4.2 *Input*

The input of this pipeline is a pair of Sentinel-1 SLC product in their native SENTINEL-SAFE format (zipped or unzipped product are both supported). The inputs shall be acquired with the same acquisition geometry (i.e. same relative orbit), which means in practice that they are acquired in a multiple of six days (6, 12, 18 days) of difference (considering both satellites, S1A and S1B). The validity of the inputs will be checked by the pipeline and when the requirements are not fulfilled, an error will be raised.

2.2.4.3 Exposed Parameters

Table 5. Exposed parameters in S1 Change detection– Amplitude Change Detection and Multi-temporal Coherence

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)
CalibrationOutput: backscatter convention selected for the output. Each value uses a different reference area.	Sigma, Gamma, Beta	Sigma
Polarization	VV,VH… (valid values depend on the specific product)	VV
Land/Sea mask: type of pixels to be removed considering SRTM3Sec. If “Sea” is selected, all values with elevation=0 in SRTM 3Sec will be set to NoData.	Land/Sea/None	None
AoI: Area of Interest	WKT polygon or path to vector file	None
Speckle-filter (+ specific parameters if needed)	Supported by SNAP	Lee Sigma
Projection: output projection	Any supported by SNAP	WGS84
Output format: output format	Supported by GDAL and SNAP	GeoTiff

2.2.4.4 Processing

The pipeline has been designed mostly using SNAP and GDAL. The SNAP graphs executed to generate the coherence are represented in Figure 4 and Figure 5.

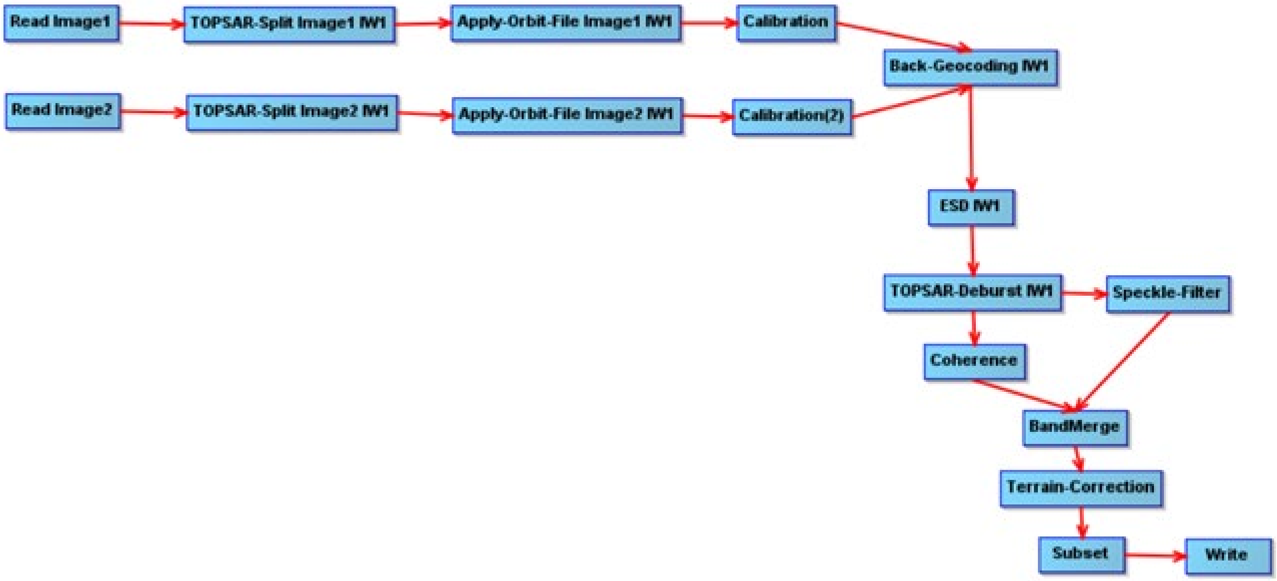


Figure 4. Sentinel-1 IW processing.

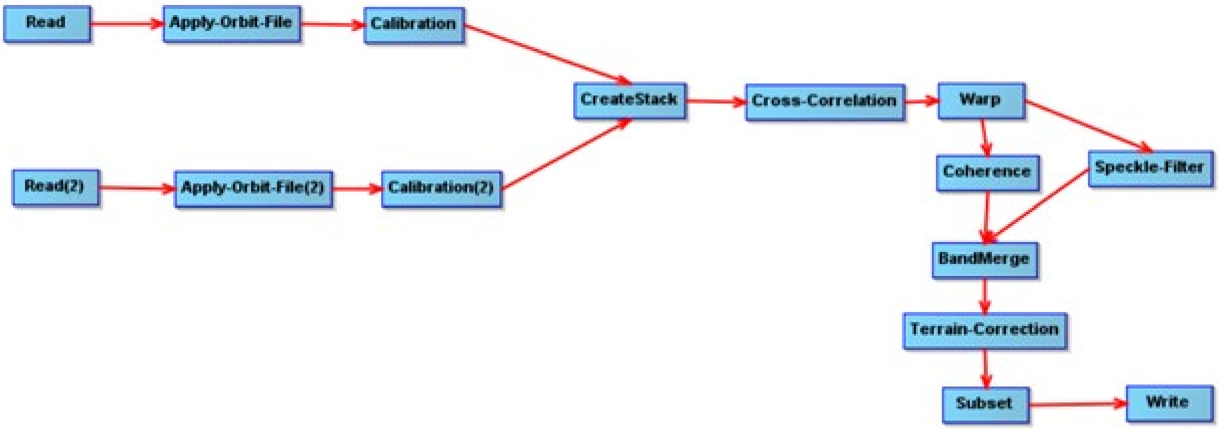


Figure 5. Sentinel-1 SM processing.

Where:

- Read: the operator in charge of reading a product to the SNAP internal data model.
- Apply-Orbit-File: search, download and apply the corrected information about the orbit that is provided some days after the S1 acquisition to improve the geolocation.
- TOPSAR-Split: split the product using the indicated subswath.
- Back-Geocoding: co-register the input products.
- ESD: Enhance the coregistration when more than one burst have been selected.
- Calibration: convert pixel values to calibrated sigma0, gamma0 or beta0
- Speckle-Filter: applies filter to reduce speckle
- Coherence: estimates the coherence

- Terrain-Correction: orthorectify the product
- CreateStack + Cross-Correlation + Warp: coregister the input products.
- Write: write the output product to the desired format.

(More information about the specific operators can be found in the SNAP help and documentation.)

2.2.4.5 Output

The outputs are:

- Coherence: GeoTiff image that represents the amplitude of correlation between the images. The pixel type is float32.
- Backscatters: 2 GeoTiff products (one for each of the input images) with one float32 band representing the calibrated backscatter.
- ACD: a RGB composite of the backscatter of the input images.
  - GeoTiff with three bands. R: backscatter of image 1; G : backscatter of image 2; B: backscatter of image 2
  - Pixel type is Byte, where byte value is computed by data conversion of float values using a linear interpolation taking as min and max values the percentiles 2.5 and 7.5.
- MTC (Multi-Temporal Coherence): a RGB composite of the backscatters and the coherence
  - GeoTiff with three bands. R: backscatter of image 1; G : backscatter of image 2; B: coherence
  - Pixel type is Byte, where byte value is computed by data conversion of float values using a linear interpolation taking as min and max values the percentiles 2.5 and 7.5 for backscatter band. For the coherence band the linear conversion is (0,1)->(0,255)
- Binary mask of changes. The pixel type is byte.

2.2.4.6 How to use

Minimum requirements: 32GB of RAM.

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

```
S1-CD --input1 “VALUE” --input2 “VALUE” [--calibration “VALUE”] [--polarization “VALUE”] [--landseamask “VALUE”] [--speckle “VALUE”] [--AoI “WKT”] [--resolution “VALUE”] [--projection “VALUE”] [--output_format “VALUE”] --output_path “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  S1-CD --input1 “VALUE” --input2 “VALUE” [--calibration “VALUE”] [--polarization “VALUE”] [--landseamask “VALUE”] [--speckle “VALUE”] [--AoI “WKT”] [--resolution “VALUE”] [--projection “VALUE”] [--output_format “VALUE”] --output_path “VALUE”
```

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

2.2.5 Sentinel-2 Change detection

2.2.5.1 Summary

The Sentinel-2 Change Detection pipeline is available as a dockerized application 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.

2.2.5.2 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).

2.2.5.3 Exposed Parameters

Table 6. 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

2.2.5.4 Processing

The standard approach when computing changes is simplified in Figure 6.

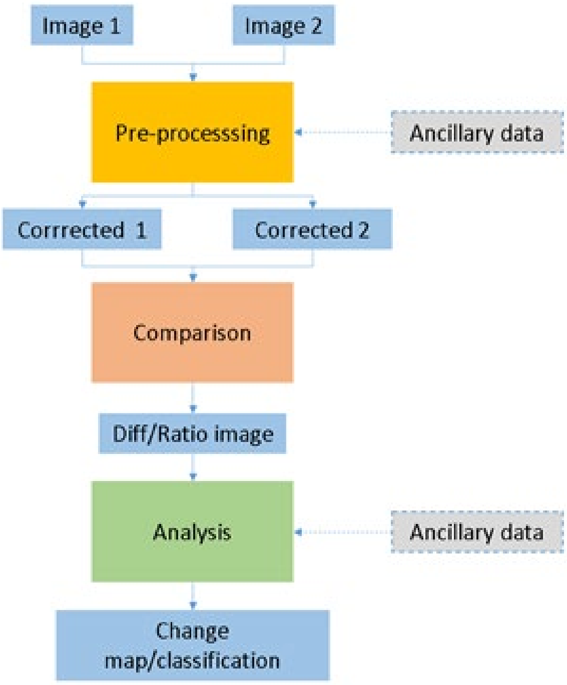


Figure 6. S2 Change detection processing flow.

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).

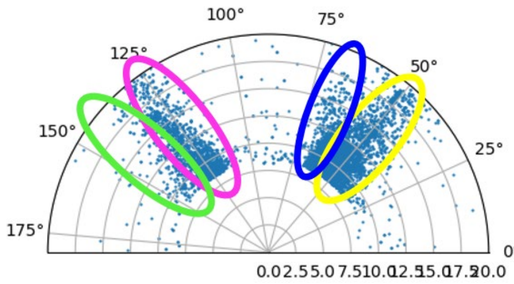


Figure 7. Example of classified changes in the polar representation using K-means.

2.2.5.5 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.

2.2.5.6 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”] --
output_path “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”] --output_path “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*.

2.2.6 Vector data of human features

2.2.6.1 Summary

The vector data of human features service is based in OpenStreetMap (OSM). OSM is a collaborative project to create a free editable map of the world with an open-content license. The OSM License allows free (or almost free) access to the map images and all of the underlying map data. But this access is not always easy since there are limitations in the servers and APIs. Moreover, the data structure is not the preferred by Security domain.

SatCen has pre-processed and ingested in their own data model (SatCen Data Dictionary) the OSM data and can provide the data as a service in the scope of the project.

2.2.6.2 Input and parameters

The input data in which the service is based is the OSM data, but it is offered to the users in SatCen's data model.

For obtaining the data, the user has to indicate the features of interest (e.g. airport) and the area of interest.

2.2.6.3 Processing

The process can be considered a ECTL process (Extract, Cleanse, Transform and Load workflow). It was developed in FME and based in schema mapping documents.

2.2.6.4 Output

The outputs are the desired features in the AoI in geojson format.

2.2.6.5 How to use

This service is available as a web service where the users can select the desired feature and define the AoI.

3 Agriculture bootstrapping services and resources

3.1 Introduction

The Agriculture Bootstrapping service and resources have been developed to support the development of EO applications leveraging on AI algorithms for the food security and agriculture fields. The services aim to facilitate the integration of the proposals of the bidders in the AI4Copernicus environment.

To be effective at large scale, data and resources need to be properly pre-processed to accurately perform crop dynamic monitoring. The bootstrapping services described in this section allow the definition of a generic processing pipeline to perform crop type mapping, and to support the testing and implementation of innovative AI algorithms.

3.2 Services

The services described are available as a dockerized application that can be executed in any environment with a properly configured Docker client.

The services have not been modified in the second update of the resources due to the absence of feedback and requests to change. Minor updates have been implemented to improve the QoL of the services.

3.2.1 Deep network for pixel-level classification of S2 patches

3.2.1.1 Summary

This service provides functionality for users to train a custom pixel-level classifier of Sentinel 2 patches. For example users can train a classifier for crop types (corn, sunflower, wheat, etc), land cover (urban vs. natural, water vs land), road extraction (road vs other).

The service will be available as a dockerized application that can be executed in any environment with a properly configured Docker client.

The service will implement code to train a U-NET model on Sentinel2 hyperspectral images (that can be first converted to smaller patches). The weights of a part of the U-NET model will be pre-trained with a self- supervised approach, increasing final classification performance.

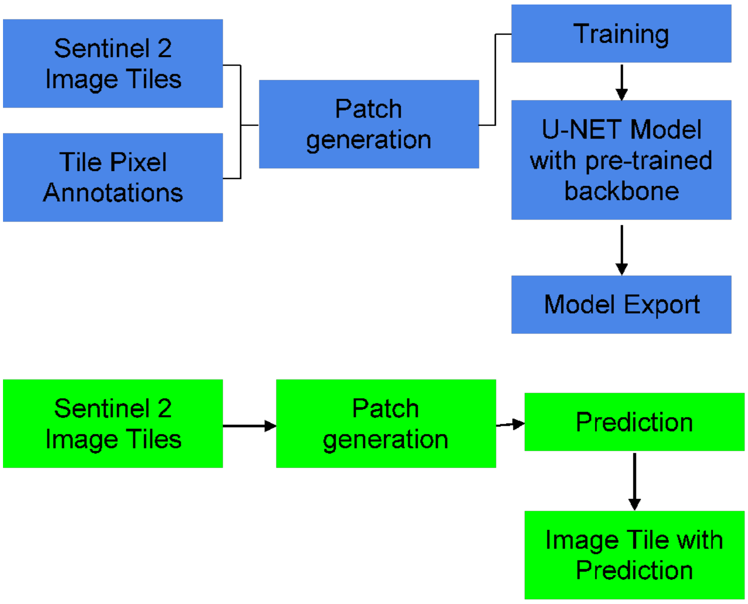


Figure 8. Deep network for pixel-level classification of S2 patches.

3.2.1.2 Input/output

The docker container will take as input a directory containing patches and their labels (patches with pixels = 1..N labels). The output will be a U-NET model which makes the classification by pixel with these N classes.

3.2.1.3 Pipeline

The service design is based on a network architecture with a backbone, a ResNet-50, which is pre-trained separately on SEN12MS. By using self supervised learning, an emerging unsupervised training procedure, we will learn good features on Sentinel-2 images, without requiring labelling, to initialize our network's backbone (*Ref. Citation: Ciocarlan, A.; Stoian, A. Ship Detection in Sentinel 2 Multi-Spectral Images with Self-Supervised Learning. Remote Sens. 2021, 13, 4255. https://doi.org/10.3390/rs13214255)*

Users provide their own data under the Sentinel format and can train a segmentation model. A various number of parameters are available to customise the training phase (see table below). The service also provides the code to apply the trained model to a new set of data.

3.2.1.4 Exposed parameters

Table 7. Exposed parameters in Deep network for pixel-level classification of S2 patches.

Parameter	Valid values	Default Value
-action: action to perform. One among "full", "extract_data", train","predict"'	full, extract_data, train, predict	_
-criterion: loss criterion used during training	cross_entropy, focal_loss	cross_entropy
-pretrained: enable pretrained backbone	True/False	False
-bands: comma separated list of bands:	Any combination of S2 L2A bands: B1,B2, B3, B4, B5, B6, B7, B8, B8A, B9, B10, B11, B12	_
-selected-classes: comma separated id of classes.	1-255	1
-data_aug: enable data augmentation	True/False	False
-simplified: enable simplified label representation for the Sen12MS dataset	True/False	False

-b, --batch-size: batch size for training	2-64	2
-epochs: number of epochs for training	1-99	20
-patch_size: size of input data for training	256/64	256
-lr: initial learning rate	Any non-zero positive real value	0,0001
-model_format: model save format	Pytorch, onnx	Pytorch
-model_file: model file name	String value	_

3.2.1.5 How to use

Recommended requirements : 16GB of RAM, RTX 2080ti.

First of all the user should obtain a data set of S2 image tiles with any combination of bands. Furthermore, the user should then provide pixel level annotation of these image tiles. These tiles can be of any size, with the lowest size being 64x64 pixels. Larger image tiles will be cut into 64x64 patches by the service, automatically.

The annotations do not need to cover all image pixels, some pixels that have unknown/irrelevant types will have class 0. This can be considered a background class. For example if the user wants to segment roads, she/he will annotate all road pixels with class 1 and leave the rest of the pixels as class 0.

To train a model the user will invoke the dockerized applications through the command line or eventually through the cloud interface. He/she will point the application to a directory containing Sentinel 2 images and their annotations. After training, the application will produce a “trained model” that can produce predictions on new image data. Calling the application in “Predict” mode while supplying such a saved trained model will allow the user to predict pixel level classification on new data.

3.2.2 TimeSen2Crop

3.2.2.1 *Summary*

TimeSen2Crop is a pixel-based dataset made up of more than 1 million crop type samples of Sentinel-2 time series. The dataset includes atmospherically corrected images and reports the snow, shadows, and clouds information per labelled unit, as well as the spectral signature of the samples of nine Sentinel-2 spectral bands at 10m of spatial resolution.

3.2.2.2 *Dataset Description*

The dataset is organized hierarchically, as shown in Figure 8. The data are organized per Sentinel-2 tiles, i.e. 16 folders. Each folder contains 16 sub-folders, each associated to a particular crop type. The multispectral temporal signature is stored in a .csv file, which provides a matrix where each row indicates the acquisition date, and each column indicates the spectral band associated. The last column in the file shows the condition of the pixel. A csv containing the list of the dates in which the samples have been acquired is stored inside the tile folder.

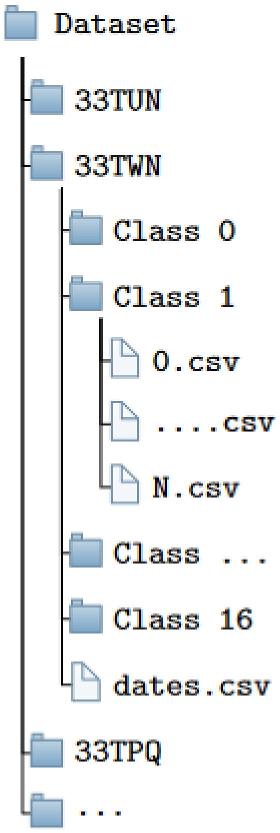


Figure 9. Hierarchical folder structure of TimeSen2Crop.

Table 8. Dataset specifics for TimeSen2Crop.

Dataset Specifics	Values
Spatial Resolution	10 [m]

Spectral Resolution	B2-490 [nm], B3-560 [nm], B4-665 [nm], B5-705 [nm], B6-740 [nm], B7-783 [nm], B8A-865 [nm], B11-1610 [nm], B12-2190 [nm]
Pixel condition	clear [0], cloud [1], shadow [2], snow [3]
Agronomic Year	September 2017 - August 2018
Crop Types – Classification Scheme	Legumes [0], Grassland [1], Maize [2], Potato [3], Sunflower [4], Soy [5], Winter Barley [6], Winter Caraway [7], Rye [8], Rapeseed [9], Beet [10], Spring Cereals [11], Winter Wheat [12], Winter Triticale [13], Permanent Plantations [14], Other Crops [15]
Sentinel-2 Tiles	32TNT, 32TPT, 32TQT, 33TUM, 33TUN, 33TVM, 33TVN, 33TWM, 33TWN, 33TXN, 33UUP, 33UVP, 33UWP, 33UWQ, 33UXP

3.2.3 Harmonization of pre-processed Time Series of Sentinel-2 data

3.2.3.1 Summary

One of the challenges of crop type mapping at large scale is the definition of a regular temporal sampling grid to perform analysis in a standardized manner. This optical pre-processing harmonizes the irregular time series of images and mitigates the cloud coverage problem. The harmonization of pre-processed time series of Sentinel-2 data considers a statistic-based approach that computes the median for each pixel in the different images acquired in a particular month. The pixel composite approach to mosaic generation provides consistent results at large scale, allowing the processing of harmonized acquisitions.

3.2.3.2 Input

The inputs of this service are a Sentinel-2 Bottom of Atmosphere time series and the LAI images associated. The input to this service can be obtained using the pre-processing pipeline defined in the previous section. The `input_dir` specified to execute the service must contain two folders, “REFBOA” and “LAI”, each containing the .tif images.

3.2.3.3 Exposed Parameters

Table 9. Exposed parameters in Harmonization of pre-processed Time Series of Sentinel-2 data.

Parameter	Valid Values	Default Value
tile: UTM code associated with the selected tile (i.e. “33UVP”)	string	None
season_start: Year at which the agronomic year starts defined from September (A) to August (B).	number	2017

3.2.3.4 Processing

The integrated algorithm has been designed using python. The script reads the LAIs of the images in a particular month, sets to 0 the corresponding pixels in the BOAs, and performs the median operation masking the cloudy samples.

3.2.3.5 Output

The outputs of the Harmonization step are 12 monthly composite GeoTiff images in uint16 representing the agronomic year specified by the user.

3.2.3.6 How to use

Minimum Requirements: 16GB RAM, GPU: None

Inside the docker, the pipeline can be executed with the following command:

```
python3 AI4C_MComp.py --input_dir="VALUE" --output_dir="VALUE" --
season_start=”VALUE” --tile=”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 -u $(id -u):$(id -g) -v $(pwd):$(pwd) -w $(pwd) docker_name python3
AI4C_MComp.py --input_dir=”VALUE” --output_dir=”VALUE --season_start=”VALUE” --
tile=”VALUE”
```

The user can check the help page to have an overview of the variables that can be defined and a short description of each parameter.

```
monthlycomposite --help
```

3.2.4 Long Short-Term Memory Neural Network for Sentinel-2

3.2.4.1 Summary

The Long Short Term-Memory architecture can be trained using samples selected by the user. The service exploits the data given by the user to train from scratch an LSTM and stores the resulting weights. Several parameters are exposed to allow the user to custom the model.

3.2.4.2 Input

The service requires two different .npy files stored inside the `input_dir` specified, one containing the multitemporal spectral signature of the samples used to train the LSTM (“train.npy”), and a .npy file containing the labels associated with the samples (“y_train.npy”). The dimension of the array containing the samples used to train the model must be [N,ts,b], where N is the number of samples, ts the number of time steps in the Time Series, and b the number of bands.

3.2.4.3 Exposed Parameters

Table 10. Exposed parameters in Long Short-Term Memory Neural Network for Sentinel-2.

Parameter	Valid Values	Default Value
n_epochs: number of epochs to update the internal model parameters	Any number >1	50
batch_size: number of samples used to update the internal models	Any number >2 and power of 2	32
class_weights: flag used in the training to give different weights based on the a-priori probability of each class.	0 (not used) - 1 (used)	0
learning_rate: number describing the step size.	Any number > 0 and < 1	1e-3
val: string pointing to the path where a validation set is stored.	path to folder containing the .npy data	None

3.2.4.4 Processing

The service has been designed using python. The script performs a normalization on the input data and trains the neural network.

3.2.4.5 Output

The output of the Long Short-Term Memory network is a .h5 file containing the weights and the model configuration.

3.2.4.6 How to use

Minimum Requirements: 16GB RAM, 1 GPU x worker

Inside the docker, the pipeline can be executed with the following command:

```
python3 AI4C_LSTMTrain.py --input_dir=‟VALUE” --output_dir=‟VALUE” --
n_epochs=”VALUE” --batch_size=”VALUE” --class_weights=”VALUE” --
learning_rate=”VALUE” --val=”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 -u $(id -u):$(id -g) -v $(pwd):$(pwd) -w $(pwd) docker_name python3
AI4C_LSTMTrain.py --input_dir=”VALUE” --output_dir=”VALUE” --n_epochs=”VALUE” --
batch_size=”VALUE” --class_weights=”VALUE” --learning_rate=”VALUE” --val=”VALUE”
```

The user can check the help page to have an overview of the variables that can be defined and a short description of each parameter.

```
train_lstm --help
```

3.2.5 Pre-Trained Long Short-Term Memory

3.2.5.1 *Summary*

The pre-trained Long Short Term-Memory architecture is already trained using the TimeSen2Crop database and is available in .h5 format. The service exploits a pre-trained architecture to classify the specified tile harmonized using the monthly composite approach.

3.2.5.2 *Input*

The input of this service is a GeoTiff image containing the information regarding the agricultural samples that the LSTM must classify (if no mask is provided, the service will classify the entire image), and a monthly composite time series. The .tif image and the .h5 pre-trained network must be in the --input_dir folder.

3.2.5.3 *Processing*

The service has been designed using python. The monthly composite time series is classified according to the TimeSen2Crop classification scheme. The GeoTiff image defined by the user is used to mask the samples not related to agriculture and the samples where classification is not needed. If the binary crop mask is specified, pixels with 0 value will be classified, while pixels with other values will be ignored. The script performs a block-wise classification of the monthly composites and applies morphological operators to the result.

3.2.5.4 *Output*

The output is the crop type map associated to the monthly composite time series as a GeoTiff image in uint8.

3.2.5.5 *How to use*

Inside the docker, the pipeline can be executed with the following command:

```
python3 AI4C_LSTMInference.py --input_dir=‟VALUE” --output_dir=‟VALUE” --
mc_dir=”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 -u $(id -u):$(id -g) -v $(pwd):$(pwd) -w $(pwd) docker_name python3
AI4C_LSTMInference.py --input_dir=‟VALUE” --output_dir=‟VALUE” --mc_dir=”VALUE”
```

The user can check the help page to have an overview of the variables that can be defined and a short description of each parameter.

```
tile_classifier --help
```

4 Energy bootstrapping services and resources

4.1 Introduction

The AI4Copernicus Energy Bootstrapping services have been identified to support the application of AI technology towards satellite data to produce models or forecasts that address one of the three broad energy questions.

The three broad questions are:

- As a user I want to know where I can and cannot build low carbon and renewable energy infrastructure
- As a user I want a better understanding of energy consumption and energy needs of a society
- As a user I want to know where to carry out precision, pre-emptive, maintenance in my energy infrastructure

The main function of the services are labelled data sets used to train machine learning and artificial intelligence services. In addition to these domain specific resources, the energy domain will also exploit the bootstrapping services from other domains (e.g. pre-processing services of Sentinel-1 and -2 from the security) if requested by open calls winners. In order not to duplicate the information, the details of other generic services can be found in the other domains' sections.

4.2 Resources

Table 11. Labelled datasets for Energy domain.

Meteorological data: ERA5	CREODIAS - ERA5 product home page How to order CDS ERA5 products - FAQ Answer - CREODIAS
Example: offshore wind farms are located (training data)	EMODNET Data Catalogue EMODnet Product Catalogue - EMODnet (europa.eu)
JRC Open Power Plants Database (JRC-PPDB-OPEN)	WMS service identified (https://data.europa.eu/data/datasets/9810feeb-f062-49cd-8e76-8d8cfd488a05?locale=en)
Open data from the floating offshore wind farm, Hywind Scotland	Platform for Operational Data (POD) | POD (catapult.org.uk)

5 Health bootstrapping services and resources

5.1 Introduction

The AI4Copernicus Health Bootstrapping services and resources are being developed to address current public health and air pollution (or air quality) challenges using Earth observation and in-situ measurement data. The services are focused on probabilistic downscaling (super-resolution) of air quality (AQ) and atmospheric composition (AC) model output. Current AC and AQ models output forecasts at relatively low-resolution (e.g., ca. 10 to 40 km for the CAMS EU and global domains). Previous research has demonstrated that it is possible to make use of Earth observation (EO) and in-situ measurement data to downscale (i.e., increase the spatio-temporal resolution) of dynamical model output, allowing the identification of pollution or greenhouse gas (GHG) emission hotspots. This service is well aligned with all proposals submitted to the AI4Copernicus first open call under the health domain. These proposals have identified the need and market potential for hyper-local short to medium-term air pollution forecasts, with up to street-scale resolution in densely populated areas that are subject to heavy pollution episodes. Real-time knowledge of AQ plus high-resolution forecasts up to a few days ahead allow policymakers to make science-informed decisions (e.g. limit traffic, issue health advisories, deploy additional health personnel, etc.). Citizens can also benefit by reducing their exposure to air pollutants - staying indoors during aerosol peaks, for example, or using AQ-aware routing when navigating through traffic-congested cities.

The health services also provide an estimate of the systematic uncertainties present in the downscaled product.

It is important to note that the technical specification of the service described below is subject to changes. The main objectives are to (1) provide a health bootstrap service that is useful for the winning bidders of the first open call and (2) ensure interoperability with other services described in this document if advantageous for end users.

5.2 Services

5.2.1 Probabilistic downscaling of CAMS air quality model data

This service generates high-resolution (currently ~ 10km) air quality maps from low-resolution (~40 - 80km) CAMS model (re)analysis and/or forecast output.

5.2.1.1 Input data

The input data sources for the probabilistic downscaling service are summarized in the table below.

Table 12. Input data for Probabilistic downscaling of CAMS air quality model data.

Dataset	Summary	Format(s)
CAMS European air quality analysis and forecasts	This dataset provides daily air quality analyses and forecasts for Europe (east boundary=25.0° W, west=45.0° E,	netCDF

	south=30.0° N, north=72.0° N) at a spatial resolution of 0.1 degrees (ca. 10km). The set of atmospheric variables includes particulate matter, pollen, ozone, dust, CO, NO, SO2, NO2 and ammonia. The downscaling service currently uses only surface data. Other vertical levels may be incorporated depending on user need.	
CAMS global (re-)analysis	The EAC4 reanalysis. The downscaling service currently uses only surface data. Other vertical levels may be incorporated depending on user need and/or impact on downscaled field quality.	netCDF
Sentinel-2	The use of Level-2A Sentinel-2 image data for identification of pollution hotspots is currently being investigated. Satellite-retrieved aerosol optical depth (AOD)[6] can be correlated to aerosol conditions over urban environments at a fine scale. Furthermore, it has been demonstrated that data from the Sentinel-2 MultiSpectral Instrument can be used to detect and quantify anomalously large methane point sources.[7]	GeoTIFF (can be the output of the Sentinel-2 pipeline described for the "Security" domain above)
Sentinel-5P[8]	The Copernicus Sentinel-5 Precursor mission provides atmospheric measurements of air quality and atmospheric composition (aerosol, ozone, NO2, CH4, CO, HCHO and SO2) with high temporal and spatial resolution (7 x 7 km at nadir resolution). Sentinel-5P NO2 total columns are assimilated operationally into the CAMS-IFS 4D-Var model. The data are available in near-real-time (within three hours of sensing, except for total ozone which has a 24hrs delay).	xml (header) and netCDF (data)
CDS weather data	The Climate Data Store (CDS, http://cds.climate.copernicus.eu) is the operational data access portal of the Copernicus Climate Change Service (C3S), which is implemented by ECMWF on behalf of the European Commission. It provides access to the C3S portfolio of products through a web interface and API.	netCDF, GRIB

[6] https://sentinels.copernicus.eu/web/sentinel/user-guides/sentinel-3-slstr/product-types/level-2-aod (product under development)

[7] https://amt.copernicus.org/articles/14/2771/2021/ and https://www.esa.int/Applications/Observing_the_Earth/Copernicus/Sentinel-5P/Mapping_high-resolution_methane_emissions_from_space

[8] https://sentinel.esa.int/web/sentinel/sentinel-data-access

	The CDS includes high-resolution weather and climate reanalysis datasets such as the ERA5 and UERRA (EU-regional) multi-decadal re-analyses. Air quality and atmospheric composition are affected by weather, hence the addition of weather information often leads to an improvement in the accuracy of the downscaled data product.	
In-situ data: e.g. low-cost sensors, IoT devices, weather stations, ...	These data will be made available and incorporated into the service at a later date, subject to licensing agreements.	XML, JSON, CSV, …
Population density and built area maps	Regridded to 0.1 degrees. Original data available at 1km global resolution.	netCDF
Orography (surface elevation)	Regridded to 0.1 degrees. High-resolution data available e.g. from EU-DEM[9] for the European domain (25m pixels)	netCDF, Geotiff

5.2.1.2 Algorithm

The probabilistic downscaling engine is based on a generative adversarial network (GAN). Recent GAN-based models have achieved impressive performance in artificial high-resolution "image" generation for the Earth sciences[10]. Crucially, GANs allow the generation of "image" (i.e., 2D dataset) ensembles that quantify the uncertainty in the high-resolution output fields.

5.2.1.3 Output Data

Table 13. Output data for Probabilistic downscaling of CAMS air quality model data.

High-resolution air quality data	10km (or finer) spatial resolution for the initial deployment of the service on WEkEO and CREODIAS. This is envisaged to go up to street-level after hyper-local data sources are incorporated into the machine learning algorithms. PM2.5 is currently supported. Other variables to be added in subsequent iterations.	netCDF, CSV (for point locations of interest)
High-resolution atmospheric composition data	10km (or finer resolution). NO2	netCDF, CSV

[9] https://land.copernicus.eu/user-corner/publications/eu-dem-flyer/view

[10] e.g., Ledig et al., 2016: https://arxiv.org/abs/1609.04802 ; Stengel et al., 2020: https://www.pnas.org/content/117/29/16805 ; Leinonen et al., 2020: https://arxiv.org/abs/2005.10374 ;

5.2.1.4 How to use

The downscaling pipeline is available as a dockerized application that can be executed in any environment with a properly configured Docker client. Inside the docker, the pipeline can be executed with the following commands:

Table 14. Commands to execute Probabilistic downscaling of CAMS air quality model data.

Pre-process input data	`aqgan-preproc --start-date <YYYYMMDDHH> --end-date <YYYYMMDDHH> --dataset <DATASET> --config config.yaml`	The YAML configuration file contains important model settings, e.g. the size of a data "tile" (hi-res: 64 x 64, low-res: 8 x 8). An example config file is provided in the source tree under aqgan/src/config
Train the GAN model on a predefined data interval	`aqgan-train --model srgan --start-date <YYYYMMDDHH> --end-date <YYYYMMDDHH> --config config.yaml`	Start and end dates for training must be provided. Currently only the srgan (super-resolution GAN) model is supported.
Generate new high-resolution frames with a pre-trained SR-GAN model	`aqgan-generate --model srgan --start-date <YYYYMMDDHH> --end-date <YYYYMMDDHH> --config config.yaml`	

5.2.1.5 Compute requirements

It is recommended to run the downscaling pipeline on an NVIDIA GPU (CUDA 10.2 or newer) with a minimum of 16 GB RAM. The minimum CPU memory required is 32GB, but this number can vary according to the size of the input dataset (we recommend 128GB or more for maximum performance). Should the total size of the input dataset exceed the amount of available CPU RAM, input data can be loaded lazily using a library such as Dask[11].

[11] https://www.dask.org/

5.2.1.6 Exposed parameters

The user can change the model settings through the YAML configuration file. This way the model architecture can be adapted to the task at hand, new input data sources can be ingested, etc.

Table 15. Exposed parameters in Probabilistic downscaling of CAMS air quality model data.

Parameter	Default Value
Input and output frame size	8 x 8 and 64 x 64
Type of super-resolution model	srgan[12]
Area of Interest (AoI)	lat-lon bounding box
Output path	netCDF
Number of filters in the generator and discriminator	64 and 32
Learning rate	10^-5
Batch size	128
Type of adversarial loss	Wasserstein[13]
Low-resolution inputs	"pm2p5", "u10", "v10", "t2m", "lsm", "sp", "z", "pm10"
High-resolution (static) inputs	"urban_frac" (urban fraction), "orog_scal" (min-max scaled orography)
High-resolution output(s)	“pm2p5” (PM2.5)
Input, output data paths	See the config file.
Output “ensemble” size	10

6 Evolution of services

After having provided the bootstrapping services to the open call winners via a docker registry (described in the Appendix), instructions on how to access the registry were provided and dedicated

[12] inspired by https://arxiv.org/abs/1609.04802

[13] Arjovsky et al., 2017: https://arxiv.org/abs/1701.07875

meetings (e.g. demo sessions) were organised with the projects that required additional technical assistance on how to access the registry and run the dockers.

Successively, a testing period was granted to the projects to test the services of their interest and provide therefore useful insight for their evolution and fine-tuning.

6.1 Feedback from users and internal partners

An initial set of feedback was provided by the project, either via a direct interface with the Support Project Officer or using the table provided below to the project winners.

Please note that the exhaustive feedback on the use of bootstrapping services from the projects will be carried out in D2.3 and D2.4: this first evaluation was mainly performed from a technical point of view in order to evaluate the access to the services and to fine tune the current services.

Resource	**Feedback and comments**
Security_services	**Useful for your project?** - Test of the Sentinel-1 GRD preprocessing service, useful for the project we are working on, to access a fully automatic pre-processing of a large amount of Sentinel-1 images **(SR4C3 and VALENS)** - Some services might be useful e.g. land/sea and cloud masks, or change detection. However, the services are designed for Sentinel data while we mostly use geostationary satellite pictures, so it is not clear whether it would work well or not **(SLIDE)** **-** The services listed are mainly basic services useful for companies with limited experience in Sentinel data processing. We have already developed our own processing chain with efficient performance validated through existing applications **(HumanityWatch)**
	Easy to Access and to integrate in your project? -Easy access, combined with automatic scripts could be really interesting **(SR4C3)** **-** Access to the harbour and downloading of image was easy **(VALENS)** - Very nice that the projects are available as containerized images, that are platform independent **(VALENS)** - Getting the code from the docker registry is pretty straightforward **(SLIDE)**
	Any Missing functionalities to improve? - Some functionalities could be interesting : 1) the use of the path of a geojson/shapefile where the WKT will be extracted, to avoid looking for the WKT format of a file. 2) Concerning the pre-processing of Sentinel-1, decibel values are also a lot used, adding a small conversion step could be interesting to diversify the use of the SAR data **(SR4C3)** - the technical documentation doesn't include how to access the image or

	navigate to run the scripts. This should be included **(VALENS)** **-** We have pre-trained a SAR ship identification model based on training data from the DIU Xview3 challenge. It would be really nice for AI4Copernicus to evaluate how the S1-GRD-Preprocessing compares to the preprocessing in that project (and make sensible adaptations if found beneficial). Link to the project and data is found here: https://iuu.xview.us/dataset **(VALENS)** -It could be made clearer which of the output files (preGRD.... or S1_GRD ...) the script produces is the image to use for further processing **(VALENS)** - The technical documentation doesn't include the minimum requirements or any tips on how to monitor the process as it is running **(VALENS)** **-** The logging of the processing pipeline is sometimes missing, or comes at the end, when the processing is done **(VALENS)** - Our use case requires us to process a lot of full-size SAR imagery, in a time sensitive matter, so any optimization with regards to speed or resource use is very welcome. Right now, with a VM with 32 gig ram, it takes 30 minutes per picture **(VALENS)** - As a step in the preprocessing, a landmask filtering (only interested in the sea mask) is a required addition in the S1-GRD-preprocessing pipeline. Hopefully, filtering out the landmask will have a positive impact on performance as well. The addition of a tolerance parameter would be perfect in our case, as we will trade keeping some of the landmask in order to be sure to capture all the sea pixels **(VALENS)** - Maybe providing a detailed guide for users that are not familiar with UNIX systems, or a pre-built service if possible **(SLIDE)**
Agriculture_services	**Useful for your project?** Initial discussed with UNITN but eventually not using it, since the ML model had to be re-trained on another testing area (e.g. Ukraine) and it might required time to be adapted **(SR4C3)** - Not really useful a priori. We could use the pre-trained LSTM in comparison to our model, but the tasks seem quite different (crop classification vs. video forecasting), which means that the LSTM architecture might not work well, even if we re-train it on our datasets **(SLIDE)** - We need to generate a monitoring of arable planted fields and potential harvested fields in the Sahel region to check the ability of the local population in crisis areas to feed themselves. That is why we have selected 2 promising services: LSTM neural network for Sentinel-2 and Deep network for pixel-level classification of S2 patches **(HumanityWatch)**
	Easy to Access and to integrate in your project? Some blocking points have been encountered with the Deep network for pixel-level classification of S2 patches (**HumaniityWatch**):

	● The service was not really ready for operational use at the beginning of the project, we got the docker around June. ● The licence agreement was too restrictive. It was only available for the AI4Copernicus project duration and the pricing model for longer usage was not already established. This blocking point is in contradiction with the wishes to transform the prototype into an operational service which commits the business model ● The licence agreement, from Humanitywatch point of view, is not suitable for an SME in front of a large group like Thales. Through the licences, Thales has the right to audit Pixstart about the usage of the services. In the context of a non-mature market where the usage can be the value of a SME like Pixstart, we can not communicate the information with a company like Thales. ● The licence agreement does not include a warranty (the software is delivered "as is") and the Thales expert told us that they don't know if the system runs in the Sahel region, they only have tested it in the European region **(HumanityWatch)** The additional agreement was signed but, requiring a further step in the acceptance,does not allow a fast access to the docker (**Sen4Weeds**).
	Any Missing functionalities to improve? - We don't have TimeSen2Crop equivalent data in the Sahel Region which is mandatory to train the model. (LSTM NN for S2) **(HumanityWatch)**
Health_services	**Useful for your project?** Downscaling of data fields might be useful for finer resolution of the satellite video forecasts. For now it has not been extensively tested since output of the SLIDE model is still blurry and thus the input superscaling would probably be lost. Another option would be to only superscale the output. However, we do not have superscaled irradiation measures to begin with, so only the pre-trained model is available. We do not know yet if the superscaling of Air quality data transfers well to that of irradiation data. Also, we might use the reanalysis data in the second phase of the project when we will try to integrate weather data in the model **(SLIDE)**
	Easy to Access and to integrate in your project? Getting the code from the docker registry is pretty straightforward. However, the code for the SLIDE model is already inside a docker so integrating the two dockers in a seamless manner is not very easy since I am not familiar at all with kubernetes.In the current architecture, the easiest way would be to superscale only the output of the SLIDE model (if it works well) and thus run the two dockers/models consecutively. The API for getting the ERA5 data is also easy to use **(SLIDE)**
	Any Missing functionalities to improve?

	Maybe some more documentation on the expected format of the input and how to call the "predict.py" code on a specific input ? For now, I think all these informations have to be inferred by looking at the code and config.yaml files. Also, a simpler "predict" module that would directly call the pre-trained model using a file path as argument to point to whatever input would be nice to quickly test the model on data it was not designed for **(SLIDE)**
Energy datasets	**Useful for your project?** The energy-specific data sets have so far not been utilised. However, data sets from other domains, e.g. Health has been utilised by "Energy" projects like SLIDE. Not used for our project (not relevant) **(HumanityWatch)**

6.2 Evolution of services to address feedback received

Table 16. Evolution of services.

Service	**Improvement**
Sentinel-1 GRD pre-processing	- Add Land/Sea Mask - Remove intermediate products - Support file (e.g. shapefile) as input for "Area of Interest" parameter - Add parameter to allow output in decibel - Improve logging - Some changes to improve performance
Sentinel-1 SLC pre-processing	- Add Land/Sea Mask - Remove intermediate products - Support file (e.g. shapefile) as input for "Area of Interest" parameter - Add parameter to allow output in decibel - Improve logging - Some changes to improve performance
Sentinel-2 pre-processing	- More parameters exposed (e.g. resampling methods) - Support L1C inputs - Support latest S2 format version - Remove intermediate products - Support file (e.g. shapefile) as input for Area of Interest parameter - Improve logging
Sentinel-1 Change Detection	- Merge IW subswath

	- Add Land/Sea Mask - Remove intermediate products - Support file (e.g. shapefile) as input for "Area of Interest" parameter - Improve logging
Sentinel-2 Change Detection	- Remove intermediate products - Support file (e.g. shapefile) as input for "Area of Interest" parameter - Improve logging

6.3 Evolution for integration and future work

As stated previously in the document, services are offered as docker images that can be run in any cloud (e.g. CREODIAS) or local infrastructure. The decision of packaging the services as dockers was made having in mind a future integration with AI4EU Experiments. AI4EU Experiments is an open source platform for the development, training, sharing and deployment of AI models whose goal is to make AI and machine learning accessible to a wider audience by creating an extensible marketplace of reusable solutions.

It is expected that the integration of some of the services as models in the AI4EU Experiments marketplace will take advantage of the new features (e.g. shared folders) that make it more convenient for EO data processing. For this, developments for the integration with CREODIAS are needed to facilitate the testing and implementation.

7 Conclusions

The present deliverable offers an overview of the status of services and resources made available from the AI4Copernicus 4 domains (Security, Agriculture, Energy and Health): each specific section described the services and resources to support the AI4Copernicus open calls winners. These services, described also in D5.1, have been updated and fine tuned (when required) based on the initial set of feedback from the first batch of open calls winners and internal users.

This document will be used, together with other technical documents provided by WP3 and WP4, as technical documentation to access the whole suite of resources and services provided by AI4Copernicus (general services for all the domains, geospatial tools, user manual for the EO clusters, domain specific resources).

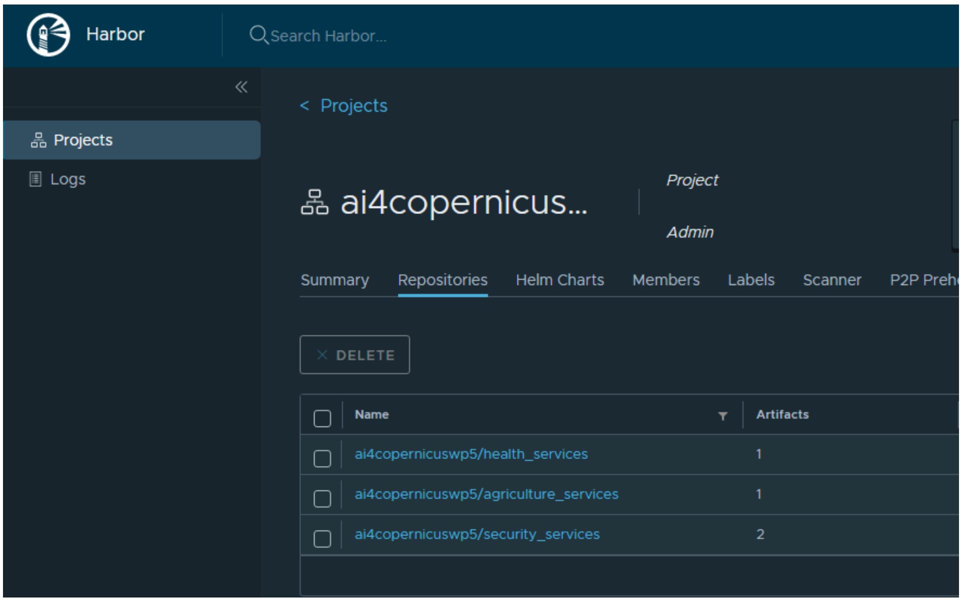
A deeper integration with AI4EU platform is expected in the second part of the project: an initial set of requirements from AI4Copernicus was already submitted to AI4EU consortium, in order to take into account the needs from the EO sector in the development of AI4EU architecture.

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.



Name	Artifacts
ai4copernicuswp5/health_services	1
ai4copernicuswp5/agriculture_services	1
ai4copernicuswp5/security_services	2

Figure 10. 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

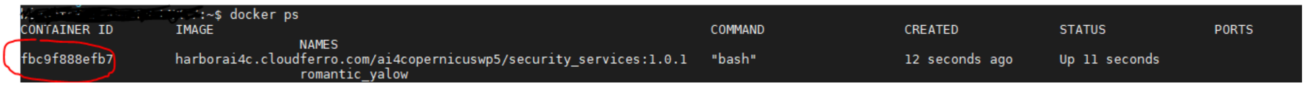
# from Container to Docker Host

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



```
[redacted]:~$ docker ps
CONTAINER ID        IMAGE                                                        COMMAND                  CREATED             STATUS              PORTS
                                     NAMES
fbc9f888efb7        harborai4c.cloudferro.com/ai4copernicuswp5/security_services:1.0.1   "bash"                   12 seconds ago      Up 11 seconds
                                     romantic_yalow
```