DEEP-HybridDataCloud

DEEP LEARNING APPLICATION FOR MONITORING THROUGH SATELLITE IMAGERY

DELIVERABLE: D2.1 (ANNEX 2)

Document identifier:	DEEP-NA2-D2.1-Annex2-V8.0.odt
Date:	16/05/2018
Activity:	WP2
Lead partner:	HMGU
Status:	FINAL
Dissemination level:	PUBLIC
Permalink:	http://hdl.handle.net/10261/164311

Copyright Notice

Delivery Slip

	Name	Partner/Activity	Date
From	Wolfgang zu Castell	HMGU / WP2	16/05/2018
Reviewed by	Ignacio Blanquer Fernando Aguilar Álvaro López	UPV CSIC CSIC	23/04/2018
Approved by	Steering Commitee		27/04/2018

Document Log

Issue	Date	Comment	Author/Partner
V1.0	26/02/2018	First template version	Álvaro López / CSIC
V2.0	26/03/2018	TOC	Wolfgang zu Castell / HMGU Marcus Hardt / KIT Lara Lloret / CSIC
V3.0	29/03/2018	First version	Daniel García / CSIC Ignacio Heredia / CSIC
V4.0	20/04/2018	Use Case Description Updated	Daniel García / CSIC Ignacio Heredia / CSIC
V5.0	20/04/2018	Use Case Requirements Updated	Daniel García / CSIC Ignacio Heredia / CSIC
V6.0	23/04/2018	External Review	Ignacio Blanquer / UPV Fernando Aguilar / CSIC
V7.0	24/04/2018	Internal Review	Álvaro López / CSIC
V8.0	25/05/2018	Final Version	Daniel García / CSIC Ignacio Heredia / CSIC

Table of Contents

1. Executive Summary

This use case aims to explore potential applications of Deep Learning for automatic monitorization with satellite imagery, particularly with data coming from the Sentinel mission. As a first approach we plan to develop an application to reliably detect water from the satellites spectral bands using state-of-the-art convolutional neural networks. This could have huge benefits for ecosystem's automatic monitorization.

1.1. Identification

Name	Deep Learning application for monitoring through satellite imagery
Institution/Partner	CSIC
Contacts	• Daniel García (CSIC) garciad@ifca.unican.es • Ignacio Heredia (CSIC) iheredia@ifca.unican.es

1.2. Brief description of the Use Case

With the latest missions launched by ESA, such as Sentinel, equipped with the latest technologies in multispectral sensors, we face an unprecedented amount of data with spatial and temporal resolutions never before reached. Exploring the potential of this data with state-of-the-art AI techniques like deep learning, could potentially change the way we think about and protect our planet's resources.

Possible applications of the machine learning techniques range from remote object detection, terrain segmentation to meteorological prediction. Our use case will implement one of these applications to demonstrate the potential of combining satellite imagery and machine learning techniques in a cloud infrastructure.

1.3. Expectations in the framework of the Deep Hybrid Datacloud Project

- Integration of different tools that allow download and save large datasets automatically, as well as an environment to process these large datasets and save the derived product.
- Development of tools based on deep learning to analyze a large amount of data.
- Integration of different tools based on the cloud, that allow the management of the life cycle of the data, the production of data based on FAIR+R principles.

1.4. Expected results and derived impact

Expected results

- Develop a detection and prediction system that combines the latest deep learning techniques with satellite data. An environment where you can process and analyze different satellite maps, choosing the place, the date, etc.

Derived impact

- A new way of remote detection that helps monitor and manage the different resources of the planet

1.5. References useful to understand the Use Case

Although the references are not directly with lake monitoring, they are nevertheless useful examples of how satellite imagery and deep learning can be combined together.

- Combining satellite imagery and machine learning to predict poverty, N. Jean et al, *Science* (2016), Vol. 353, Issue 6301, pp. 790-794.
- Using convolutional networks and satellite imagery to identify patterns in urban environments at a large scale, A. Albert and M. Gonzalez, *Proceedings of the ACM KDD* (2017).

2. Introduction and Use Case

2.1. Presentation on the Use Case

This use case aims to explore potential applications of Deep Learning for automatic monitorization with satellite imagery, particularly with data coming from the Sentinel mission. As a first approach we plan to develop an application to reliably detect water from the satellites spectral bands using state-of-the-art convolutional neural networks. This could have huge benefits for ecosystem's automatic monitorization.

2.2. Description of the research community

The community is composed by developers and users:

- Manager/Developer: Configuration to preprocessing, the data ingestion on the models and the software for modeling.
- Researcher/User: Configure search parameters, place, date, etc. to obtain the predictions. Gets data/reports from special interest points.

2.3. Current Status and Plan for this Use Case

Current Status

- We have an earth observation web service with Sentinel images that allows us process the raw data of ESA and obtain RGB images or derived product such as different vegetation index.
- Knowledge of the latest deep learning technologies applied to image recognition techniques (aplicattion of image classification of plants with deep learning using images)

Plan

- Combine both techniques to further develop existing tools (more images from other satellites, create your own maps, etc.) and develop new tools that include deep learning techniques.

2.4. Identification of the KEY Scientific goals

2.5. Description of potential development

There are multiple possible beneficiaries of this tool:

- Governments could use these tools to better monitors ecosystems and therefore design better policies regarding the environment.
- European biodiversity platforms such as LifeWatch
- Private researcher for your research

3. Technical description of the Use Case

3.1. User categories and roles

- Manager/Developer: They take care of the preprocessing and ingestion of the data. They develop software to create the deep learning tools.
- Researcher/User: They use the tools (and input query parameters) to obtain the predictions. They get data/reports of special interest points.

3.2. General description of datasets/information/software used

Datasets

- Satellite data: Depending on the satellite, but information can usually be exported as images (TIFF). During the preprocessing the images are transformed into CSV or NetCDF4-HDF5. Standards on geographical information system web services will be taken into account.
- Environmental measurements: Used to validate the predictions made using satellite observations. They usually come in CSV format.

Software

Most analysis and development will be carried on in Python:

- Standard modules for scientific data analysis in Python are Numpy, Scipy, Pandas, matplotlib and os
- Modules for image processing like OpenCV and PIL.
- Deep learning tools will be developed in either one of the majors frameworks available: Tensorflow and Pytorch and wrappers around them.
- Earth observation python modules: snap-api, sentinel-sat and earthengine-api

It is also necessary to work with programs to process georeferenced images such as QGIS and GeoServer.

3.3. Technological (S/T) requirements

Scientific requirements

- Experts to validate the results produced by the developed tools.

Technological Requirements

- Storage in the order of TB to store satellite images.
- CPUs with 64~128GB of RAM to process satellite imagery.
- Powerful GPUs to efficiently train the deep learning applications.

3.4. Identification of required services

- Orchestration of containers to store the maps to analyze
- Data orchestration to control the flow of training images to the network

3.5. Description of the Use Case in terms of Workflows

4. Data Requirements

4.1. Access Control

With respect to Authentication and Authorization, the final solution must be compatible with the architecture designed in the context of the AARC2 project. Access based on roles or groups (developer, enabler, end user, researcher, etc).

4.1.1. Privacy

There are no privacy concerns regarding the data as the satellite imagery datasets we plan to use (Sentinel) are open-access.

Anyone can register online via self-registration. Registration grants access rights for searching and downloading Sentinels products. Sentinels products are available at no cost for anybody. The data available through the Data Hub is governed by the Legal Notice on the use of Copernicus Sentinel Data and Service Information , which the User is deemed to have accepted by using the Sentinel data.

The access and use of Copernicus Sentinel Data and Service Information is regulated under EU law. In particular, the law provides that users shall have a free, full and open access to Copernicus Sentinel Data and Service Information without any express or implied warranty, including as regards quality and suitability for any purpose.

4.1.2. Location

ın general, satellite data is stored in ESA (https://scihub.copernicus.eu/dhus) or NASA (https://earthexplorer.usgs.gov) repositories that can be accessed through an API. For training an application those data should partially be copied to a local server to allow for fast access from the GPU.

As well as the heterogeneous sources, the transfer patterns are also diverse. Depending on the source, the transfer should be done differently:

- Satellite data: In general, satellite data is stored in repositories (ESA, NASA) and accessible openly, under registration.
- Field observations: Usually stored in real time in a database or files. Later distributed in open-access catalogs/repositories.

4.1.3. Sharing

Due to the satellite imagery being open- access, we wouldn't share the raw data used for training but instead an identifiers to link to the original source of the ESA repositories. If found useful we could share derived representations of the data (like water maps constructed from raw spectral band informations) giving them PIDs for intermediate datasets and DOIs for public datasets.

For simplicity sake, field observations, while also being open access, could also be release as dataset if gathered from multiple sources .

EU law grants free access to Copernicus Sentinel Data and Service Information for the purpose of the following use in so far as it is lawful: (a) reproduction; (b) distribution; (c) communication to the public; (d) adaptation, modification and combination with other data and information; (e) any combination of points (a) to (d).

4.2. Capacity (Data Volume)

4.2.1. Test Data / Production Data

The data capacity will be in the order of TBs. For example each satellite image file corresponding to an area of 10000 km² weights about 1 GB. So monitoring an area over long periods of time with several band per image quickly builds to lots of storage.

4.2.2. Transfer rate requirements

Transfer during training from cloud storage unit to GPU

Data transfer to GPU is the main speed bottleneck when training deep learning tools so this should be as high as possible.

4.3. Preservation requirements

Raw satellite information for training an application would usually build up to several TB of information. From this raw data we would construct derived representations which would need far less storage (in the order of hundreds of GB). Once those derived representations are constructed, the raw data are no longer needed. Once the application is trained we wouldn't need any longer the derived representations.

Thus the preservation requirements vary in different levels:

- Level 1 (~1 GBs): In the most basic level, one would only store the produced application (the neural network).
- Level 2 (~100s GBs): In addition, for reproducibility purposes, one would also store the derived representations (maps).
- Level 3 (~10s TB): In addition, for reproducibility purposes, one would also store the raw satellite bands (maps). This would also make the data preprocessing of futures applications faster when data needed are stored in a local copy and therefore we didn't need to query ESA/LandSat servers.

We suggest that the more feasible approach, as a middle ground in terms of reproducibility and practical cost of storage, would be to save the intermediate representations (which is the actual training dataset) and the code used to derive them from the raw data (which should be listed with pointers to the original Esa/LandSat source). Analysis code would of course also be shared making the whole pipeline from raw data preprocessing to the application training is fully reproducible.

5. Infrastructure and technical requirements

5.1. Expectation regarding the advantage through the use of technology

5.2. Expectations regarding e-Infrastructure use

5.2.1. Networking

Accessibility of the data from both CPUs and GPUs infrastructure with low latency interconnections among cores and nodes.

5.2.2. Computing: Clusters, Grid, Cloud, Supercomputing resources.

Infrastructure for development

- Servers with CPUs with 64~128GB of RAM to be able to process satellite images.
- Servers with one or several GPU units for fast training of deep learning applications.

Infrastructure for deployment

- Servers for hosting the web services that will enable users to access the tools.
- In the case of services for serving the deep learning tools, servers should preferably have GPUs to make fast predictions (although they can function also with CPUs).

5.2.3. Storage

Infrastructure for development

The development of the tools needs in the order of 10 TB for storing the images.

Infrastructure for deployment

The deployment doesn't have any particular requirement in terms of storage.

5.3. On (user-facing) Monitoring (and Accounting)

Modelling and visualization tools based on OpenSource solutions.

5.4. On authentication and authorization Infrastructure (AAI)

6. Formal list of requirements

See table below.

7. Use case summary table

Use Case	Deep learning application for monitoring through satellite imagery
Software and services used	Python, orchestration of containers (Mesos, Kubernetes?), data orchestration, web services that will enable users to access the tools
Machine/Deep Learning tools	• Python: Numpy, Scipy, Pandas, matplotlib, os; snap-api, sentinel-sat, earthengine-api. • Image processing: OpenCV and PIL. • DL: start with Tensorflow + Keras, in future may go for pyTorch
Computing	CPUs and powerful GPUs for deep learning (8 GB GPU memory prefered)
Memory requirements	64-128 GB
Networking	As fast as possible, as data transfer rate between a cloud storage and GPU is the main bottleneck
Storage requirements (permanent, temporal)	Feasible approach: • 100s of GBs as permanent storage (intermediate representations, the code, and the neural network) • 10s of TBs as temporary storage (raw data)
External data access requirements	• ESA: https://scihub.copernicus.eu/dhus • NASA: https://earthexplorer.usgs.gov/ Accessed via a corresponding API to copy data locally
Privacy	No privacy concerns as the datasets (Sentinel) are open access
Other requirements	Authentication and Authorization must be compatible with the architecture designed in the context of the AARC2 project
Other comments	• A node with single GPU is probably good enough to start with. • Going to run/develop the use case in a local machine, later will need containerized solution.
Relevant references or	• https://scihub.copernicus.eu/userguide/

URLs	• https://sentinel.esa.int/documents/247904/690755/Sentinel_Data_Legal_Notice

8. References

Link to the ESA website:

https://scihub.copernicus.eu/userguide/

Link to the license document:

https://sentinel.esa.int/documents/247904/690755/Sentinel_Data_Legal_Notice