# AI4EO Docker Instructions

The complete guide for using and exporting the official AI4EO Docker image

## Introduction

The docker contains the following libraries already installed and configured: python3.6, tensorflow-gpu, jupyter, gdal, scikit-learn, jupyterlab, keras, seaborn, matplotlib, numpy, scipy, pyarrow, fastparquet, pandas, xcube, rasterio, astropy, cartopy, eo-learn, torch, torchvision.

## Getting Started

To create a container, one can simply type:
$sudo docker run --detach -it --name ai4eo-container ai4eo-public

To connect with a running container's terminal, e.g. the one called ai4eo-container, type:
$sudo docker container attach ai4eo-container

To reconnect to a container, e.g. the one called ai4eo-container, one has to first make sure that the container has been started. For executing this operation, type:
$sudo docker container start ai4eo-container

There are further possible parameters that can be considered during the container creation.

1. By default, the container is configured to run with an unprivileged user that can only access to its own home directory (/home/participant). To change the user, type:
   $[..] -u <uid>:<guid>
2. To link a directory with the container, making all the files in that location accessible within the container, type:
   $ [..] -v <host_path>:<container_path>
3. To make the docker reachable from the host' network, for example for using jupyter lab with a browser on the host machine, type:
   $[..] --ip <ip-address> --port <host_port>:<container_port>
4. In case one has also nvidia-docker installed, type the following command to let the docker use a configured GPU on the host,
   $[..] --runtime=nvidia

## The Scripts

There are 3 suggested modes that have already been configured in 3 sh scripts which implements:

1. A standard unpriviledged user with read only access to the host working directory from which the script is executed. The GPU and the jupyter are enabled. To use this mode, it is needed to run the dedicated .sh script by typing:
   $sh run_ai4eo.sh
2. A mode similar to the previous one, but with a user that has also write access to the host working directory from which the script is executed. To use this mode, it is needed to run the dedicated .sh script by typing:
   $sh run_ai4eo_uhost.sh

3. [Dangerous and not recommended ] A mode with elevated privileges. To use this mode, it is needed to run the dedicated .sh script by typing:
   $sh run_ai4eo_root.sh

### Example

Example usage with the standard mode:

1. Download the docker image and the scripts to a directory on the host. Let 'app/' be the name of this directory.
2. $sudo docker load < ai4eo-public.tar
3. Open a terminal and type:
   $sh run_ai4eo.sh
4. Type:
   $sudo docker container attach ai4eo-container
5. From the container command line, type:
   $jupyter lab --ip 0.0.0.0 --port 8888 --no-browser
6. Open a browser from the host and navigate to "localhost:8888".
7. Fill in the token spawned in the container's terminal to access the jupyter lab.

## Exporting the container for submission

Follow these steps for exporting the docker container to a tarball:

1. $docker export <container name> > <team_name>_docker.tar
2. Add the tarball to a zip folder containing the other requirements for the submission

**IMPORTANT:**

Please bear in mind that the docker export command does not export the contents of volumes associated with the container. If a volume is mounted on top of an existing directory in the container, docker export will export the contents of the underlying directory, not the contents of the volume. Therefore, the documents that are required for the challenge (e.g. the jupyter notebook, and the trained models) should always be included in the internal directories of the container.

### Example:

1. $docker export ai4eo-container > team1_docker.tar
2. $gzip team1_docker.tar