Tutorial: DEEP tools for Deep Learning

Creating a new project using DEEP data science template

IBERGRID 2019, Santiago de Compostela

September 23-26, 2019

Lara Lloret Iglesias, Valentin Kozlov
lloret@ifca.unican.es, valentin.kozlov@kit.edu
IFCA-CSIC, KIT-SCC

DEEP-Hybrid-DataCloud is funded by the Horizon 2020 Framework Programme of the European union under grant agreement number 777435

DEEP Hybrid DataCloud

https://deep-hybrid-datacloud.eu/

https://marketplace.deep-hybrid-datacloud.eu/

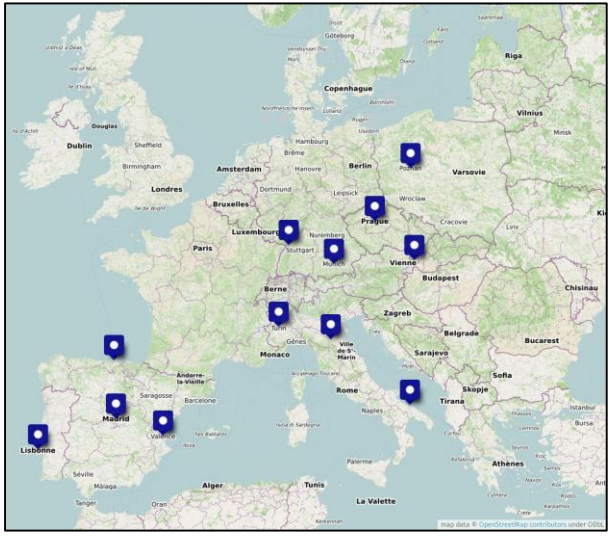
https://docs.deep-hybrid-datacloud.eu

https://www.youtube.com/playlist?list=PLJ9x9Zk1O-J_UZfNO2uWp2pFMmbwLvzXa

DEEP Hybrid DataCloud

Designing and **E**nabling **E**-Infrastructures for intensive data **P**rocessing in a Hybrid DataCloud

- Started as a spin-off project (together with XDC) from INDIGO-DataCloud technologies
- H2020 project, EINFRA-21 call
- Runs November 1st 2017 – April 2020
- 9 academic partners + 1 industrial partner: CSIC, LIP, INFN, PSNC, KIT, UPV, CESNET, IISAS, HMGU; Atos



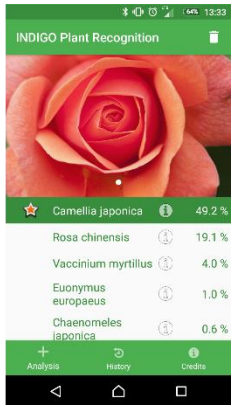
Ease and lower the entry barrier for scientists in using intensive computing techniques, e.g. deep learning, to exploit very large data sources

- Build ready to use modules and offer them through an open catalog or marketplace (Docker images)
- Implement common software development techniques also for scientist's applications (DevOps)
- Transparent execution on e-Infrastructures (pilot testbed, orchestration)

DEEP pilot use-cases

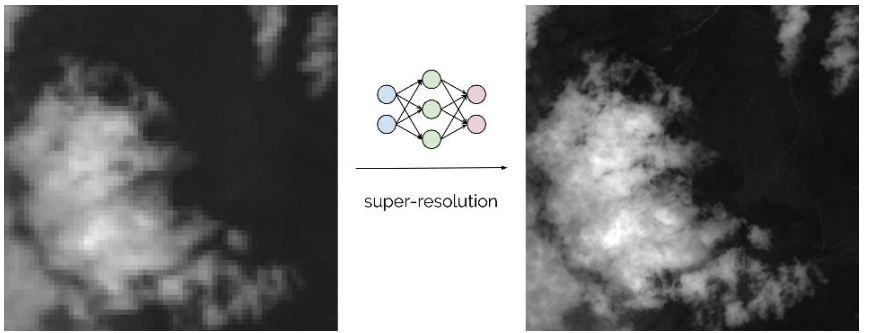
Image Classification

Plants, Conus marine snails, Seeds, Phytoplankton (CNN, TF)



Satellite Imagery

Super-resolution service (e.g. DSen2; TF)

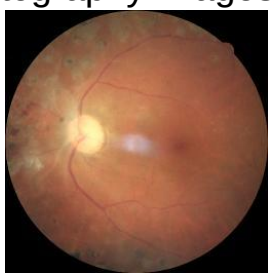


Retinopathy

DL to analyze color fundus retinal photography images (CNN, TF)

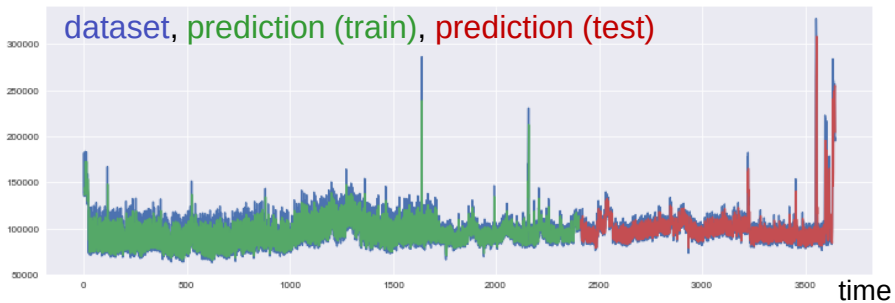
DR=0

DR=4



Massive Online Data Streams

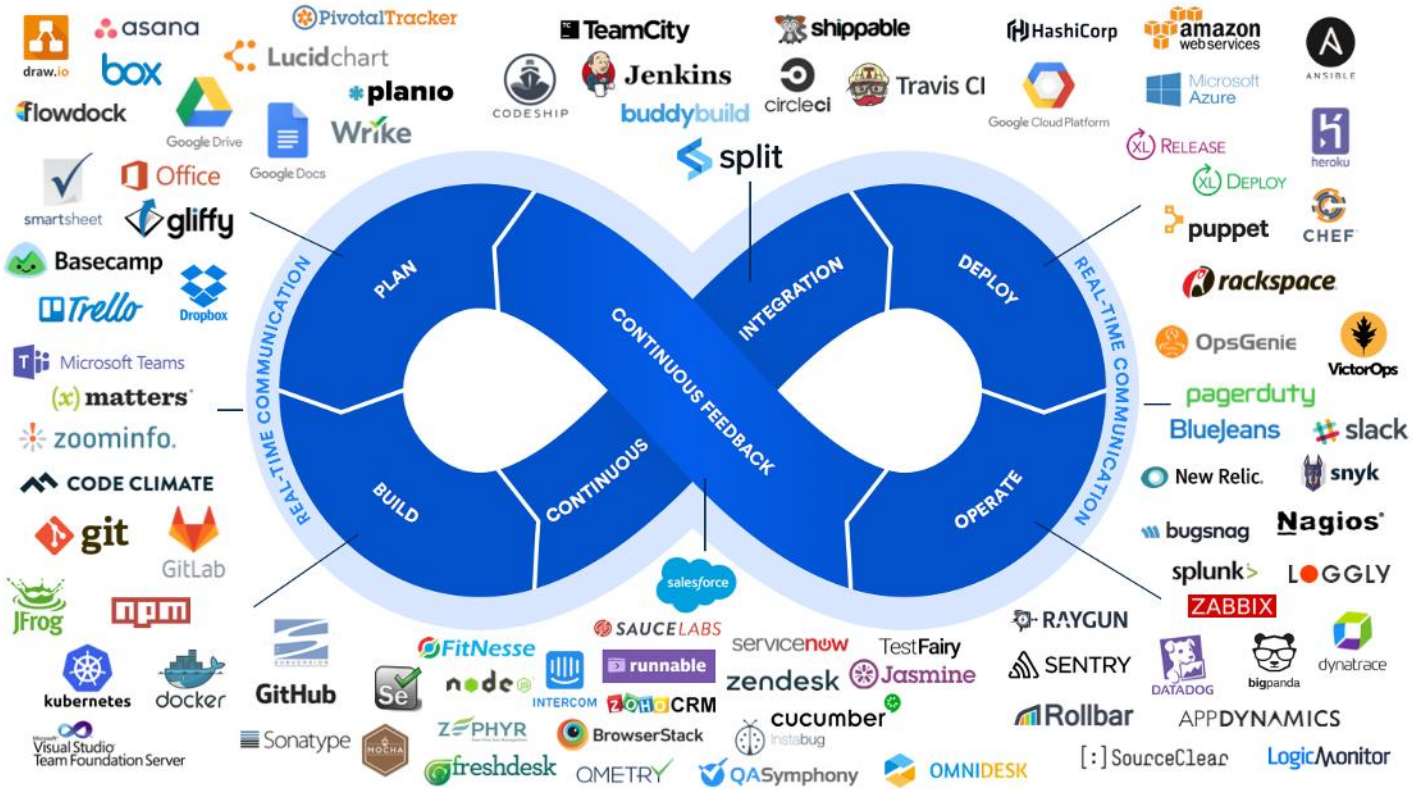
Intrusion detection systems (LSTM/GRU, TF)



DevOps & tools

ATLASSIAN

DevOps is a *set of practices* that ***automates*** the processes between *software development (Dev)* and *IT teams (Ops)*, in order that they can *build*, *test*, and *release* software ***faster*** and more ***reliably***.



DEEP DevOps toolset

Jenkins
+ jenkins-pipeline-library

Confluence

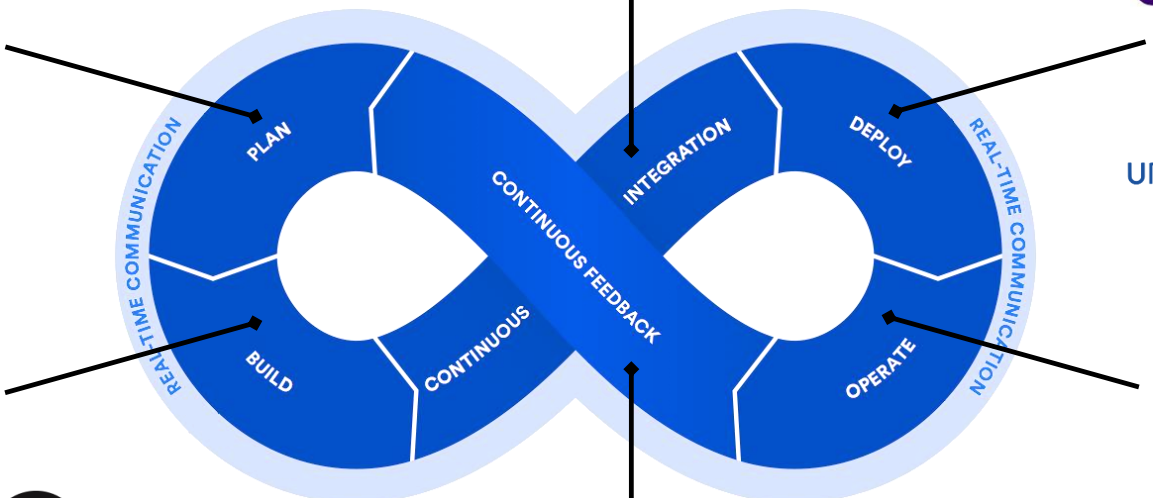
Jira Software
(user stories)

PaaS Orchestrator

OASIS TOSCA

UDOCKER

Alien4Cloud



DEEP Data Science Template

COOKIECUTTER

DEEPaaS API

metadata.json

GitHub

docker

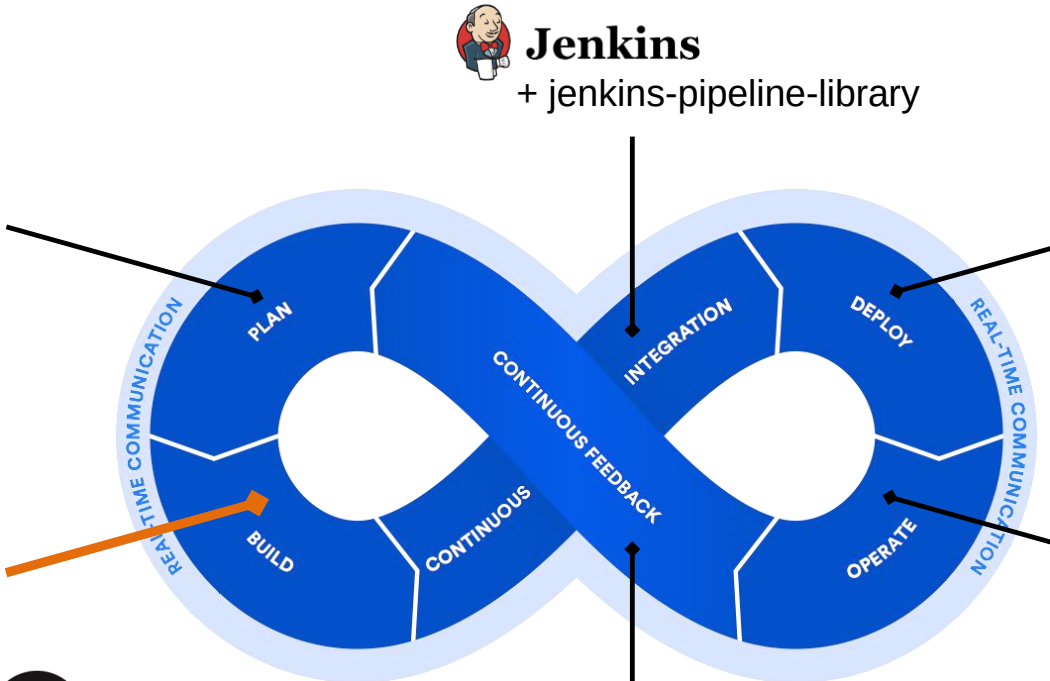
User-defined logging

Jira Software

issue tracking

GitHub

DEEP DevOps toolset



Confluence

Jira Software

(user stories)

Jenkins

+ jenkins-pipeline-library

PaaS Orchestrator

OASIS TOSCA

UDOCKER

Alien4Cloud

DEEP Data Science Template

COOKIECUTTER

DEEPaaS API

metadata.json

GitHub

docker

Jira Software

issue tracking

GitHub

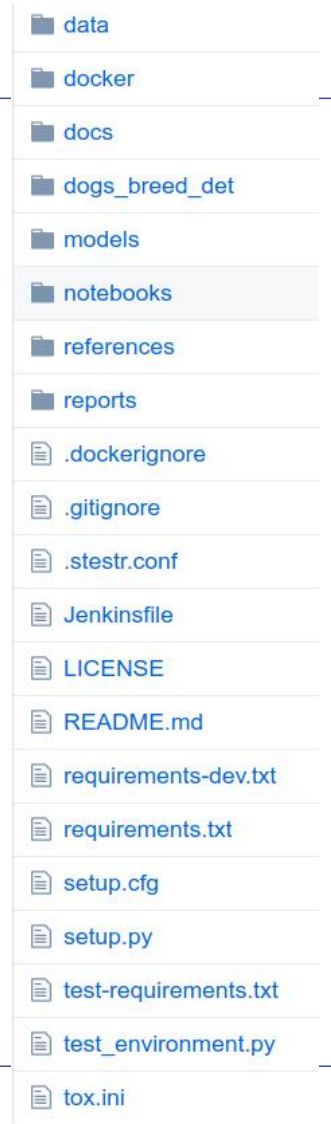
User-defined logging

I - Data science template

1. Install cookiecutter: $ pip install cookiecutter
2. Run cookiecutter with DEEP template
3. Two directories created:
   <user_project> and <DEEP-OC-user_project>

   - This are also git repositories.
   - Both have ‘master’ and ‘test’ branches.
   - Dockerfile, Jenkinsfile, and
     python files necessary for DEEPaaS API integration
     are pre-populated
   - .stestr.conf, tox.ini for testing in Python

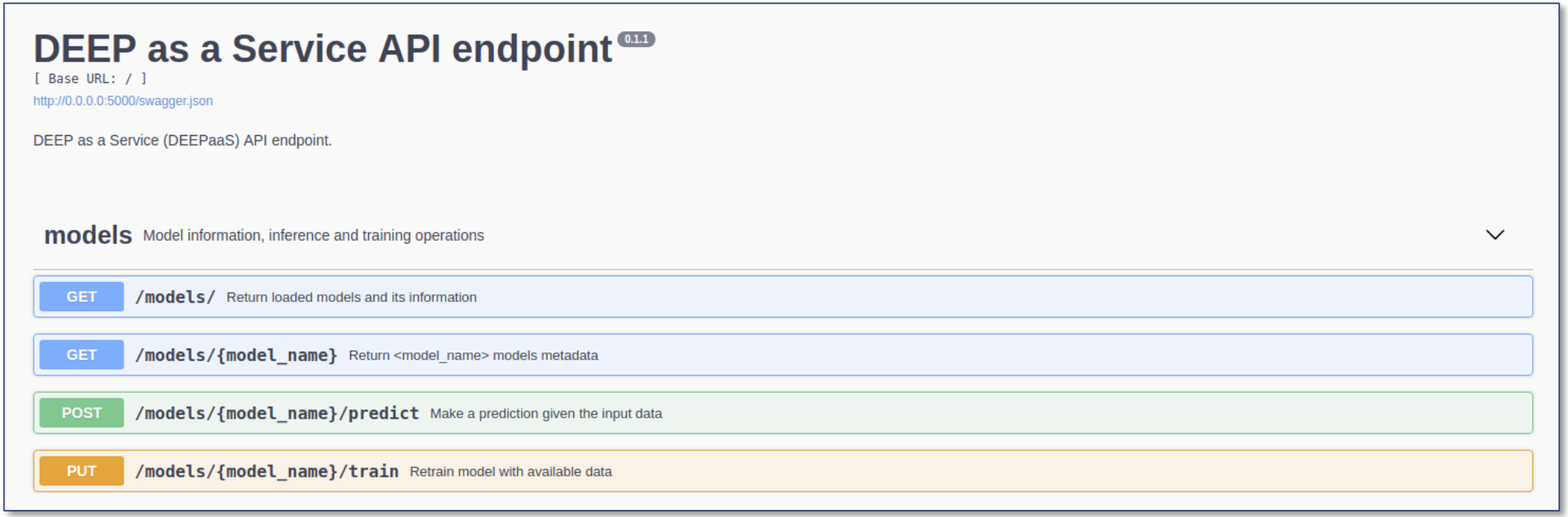
<user_project>



<DEEP-OC-user_project>

II - DEEPaaS API

get_metadata()	Return metadata from the exposed model	predict_file(path, **kwargs)	Perform a prediction from a file in the local filesystem
get_train_args(*args)	Function to expose to the API what are the typical training parameters	predict_data(*args, **kwargs)	Perform a prediction from the data passed in the arguments.
train(*args)	Function to train your model.	predict_url(*args)	Perform a prediction from a remote URL



I - Data science template

https://www.youtube.com/watch?v=akf7AdgQFN0

Prepare the environment

1. You have github and Docker accounts
2. SSH to VM (user ‘ubuntu’):
   ```
   $ ssh ubuntu@given_ip_address
   ```
3. Once logged, configure git:
   ```
   $ git config --global user.email you@example.com
   ```
4. Login to your Docker account:
   ```
   $ sudo docker login
   ```
5. Check pip availability and version:
   ```
   $ pip --version
   ```
6. Install cookiecutter with pip:
   ```
   $ pip install cookiecutter
   ```
7. Check now cookiecutter availability and the version (need >= 1.4) (avoid installing from Linux repo):
   ```
   $ cookiecutter --version
   ```

Answer template questions

N.B.: Items starting with ‘Help_...’ are just to explain next parameter

1. Run cookiecutter with DEEP template (we use ‘tutorial’ branch):

```
$ cookiecutter https://github.com/indigo-dc/cookiecutter-data-science --checkout tutorial
```

- git_base_url – base URL for your remote git repository, e.g. https://github.com/vykozlov
- project_name - name of the project being created (any name).
- repo_name – repository name (no spaces) (later referred as <user_project>)
- author_name – list of project authors
- author_email – (one) email address of the corresponding author
- description – (long) description of the project
- app_version - application version (Major.Minor.Patch). Tutorial: use default 0.0.0
- open_source_license – any of MIT, BSD-3-Clause, No license. Tutorial: use default MIT
- python_interpreter – either python3 or python2. Tutorial: use default python3

Answer template questions (continued)

- dockerhub_user – User account at hub.docker.com, e.g. 'deephdc' in https://hub.docker.com/u/deephdc
- docker_baseimage - Base Docker image for Dockerfile (FROM XX). Tutorial: tensorflow/tensorflow
- baseimage_cpu_tag - CPU tag for the baseimage. Tutorial: use default 1.10.0-py3
- baseimage_gpu_tag - GPU tag for the baseimage. Tutorial: use default 1.10.0-gpu-py3

Two repositories should be created: DEEP-OC-<user_project> and <user_project>.

Each has ‚master‘ and ‚test‘ branch

Post-actions

- Go to your https://github.com/<user> space and create two corresponding repositories: DEEP-OC-<user_project> and <user_project>
- Go back to you local machine/VM and push two repositories:
  ```
  <user_project>~$ git push origin --all
  DEEP-OC-<user_project>~$ git push origin --all
  ```
- Go to DEEP-OC-<user_project> and build Docker image:
  ```
  $ sudo docker build -t dockerhub_user/deep-oc-<user_project> .
  ```
- Get into the shell of your newly built Docker image:
  ```
  $ sudo docker run -ti -p 5000:5000 -p 8888:8888 dockerhub_user/deep-oc-<user_project> /bin/bash
  ```
- Verify that you have another environment and have /srv/<user_project> directory

Start/stop DEEPaaS API

- Inside the started container, execute deepaas-run:
  ```
  $ deepaas-run --listen-ip=0.0.0.0
  ```
- **FYI**: If the remote ports 5000, 8888 are closed, you may built SSH tunnel from your laptop, e.g.:
  ```
  $ ssh -L 5000:localhost:5000 ubuntu@194.210.120.XXX
  ```
- Point your laptop‘s web-browser to : http://given_ip_address:5000
- Verify that the info you typed when running cookiecutter appears in /models/ → „Try it out“ → Execute
- Check that /predict/ and /train/ can be called but produce „Not implemented“ message
- Stop deepaas-run (simply Ctrl-C in the corresponding terminal)

Start JupyterLab

Similar to deepaas-run:

- Inside the started container, execute:
  `$ /srv/.jupyter/run_jupyter.sh --allow-root`
  check terminal output to obtain printed token
- **FYI**: If the remote ports 5000, 8888 are closed, you may built SSH tunnel from your laptop, e.g.:
  `$ ssh -L 8888:localhost:8888 ubuntu@194.210.120.XXX`
- Point your laptop‘s web-browser to : http://given_ip_address:8888 , login
- Get acquainted with the JupyterLab interface: e.g.
  - Browser the code
  - Start Terminal emulation
  - From the emulated Terminal you can start/stop deepaas-run as in the previous exercise
- Leave JupyterLab running, we use it in the Part II of the tutorial

‚Solutions‘

- Git repositories of ‚cifar10‘ and ‚DEEP-OC-cifar10‘:

  ```
  $ git clone https://github.com/vykozlov/cifar10.git
  $ git clone https://github.com/vykozlov/DEEP-OC-cifar10.git
  ```

- Docker hub image (cpu)

  ```
  $ sudo docker pull vykozlov/deep-oc-cifar10
  ```

- This part of the tutorial: https://www.youtube.com/watch?v=akf7AdgQFN0

Thank you

DEEP Project:	https://deep-hybrid-datacloud.eu
DEEP Open Catalog:	https://marketplace.deep-hybrid-datacloud.eu
More docs:	https://docs.deep-hybrid-datacloud.eu

https://deep-hybrid-datacloud.eu

DEEP-Hybrid-DataCloud is funded by the Horizon 2020 Framework Programme of the European union under grant agreement number 777435