Using the DEEP-Hybrid-Datacloud platform

Remote webinar for EGI-ACE
01 December 2021

Ignacio Heredia
iheredia@ifca.unican.es
Instituto de Física de Cantabria
(CSIC-UC)

Introduction - The project

- The project was carried out with European Horizon 2020 funds.
- The project provides **new generation of e-infrastructures** that harness latest generation technologies, supporting deep learning and other intensive computing techniques to exploit very large data sources.
- It aims to **lower the adoption barriers** for new communities and users, satisfying the needs of both research, education communities and citizen science.

Project partners:

CSIC Consejo Superior de Investigaciones Científicas · LIP · INFN Istituto Nazionale di Fisica Nucleare · PSNC · KIT Karlsruhe Institute of Technology · cesnet · Ústav Informatiky Slovenská akadémia vied · Atos · Universitat Politècnica de València

Introduction - The users

Basic

No machine learning knowledge. Just give me a working model to make predictions.

We offer:

- ➔ a **catalogue** full of ready-to-use modules to perform inference with your data
- ➔ an **API** to easily interact with the services
- ➔ solutions to run the inference in local or **Cloud resources**
- ➔ the ability to develop complex topologies by **composing different modules**

Intermediate

I want to retrain a working model on my personal dataset.

We offer:

- ➔ the ability to train out-of-the-box a module of the **catalogue** on your personal dataset
- ➔ an **API** to easily interact with the model
- ➔ **data storage** resources to access your dataset (DEEP-Nextcloud, OneData, ...)
- ➔ the ability to deploy the developed service on **Cloud resources**
- ➔ the ability to **share the service** with other users in the user's catalogue

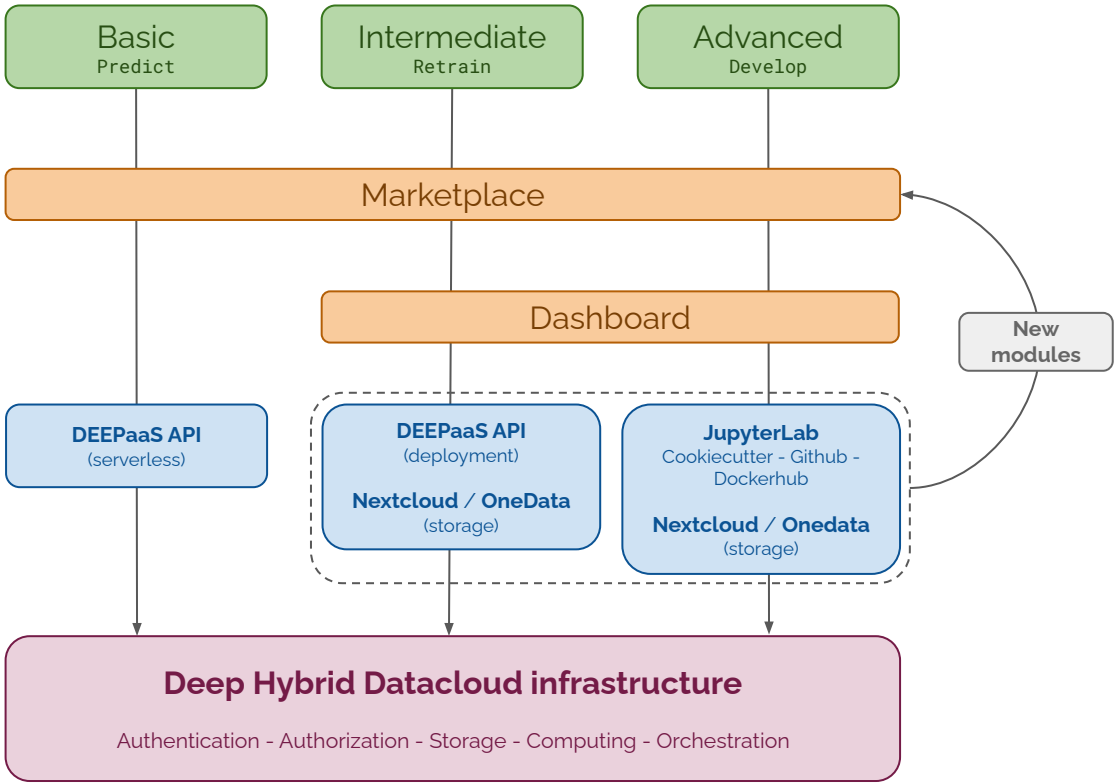
Advanced

I want to develop my custom Deep Learning model.

We offer:

- ➔ a ready-to-use environment with the **main DL frameworks** running in a dockerized solution running on different types of hardware (CPUs, GPUs, etc)
- ➔ **data storage** resources to access your dataset (DEEP-Nextcloud, OneData, ...)
- ➔ the ability to deploy the developed module on **Cloud resources**
- ➔ the ability to share the module with other users in the open **catalogue**
- ➔ the possibility to integrate your module with the **API** to enable easier user interaction

Introduction - The users



Introduction - Useful links

Homepage	https://deep-hybrid-datacloud.eu/
Marketplace	https://marketplace.deep-hybrid-datacloud.eu/
Dashboard	https://train.deep-hybrid-datacloud.eu/
Github	https://github.com/deephdc
DockerHub	https://hub.docker.com/u/deephdc/
Documentation	https://docs.deep-hybrid-datacloud.eu/en/latest/
NextCloud	https://nc.deep-hybrid-datacloud.eu/

(* these slides are available here)

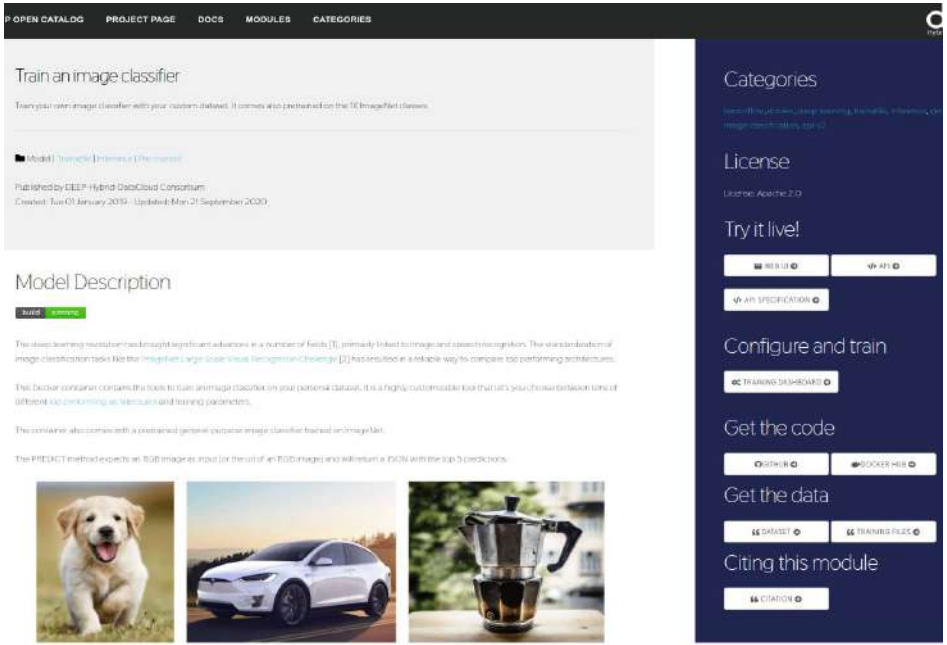
Introduction - Webinar outline

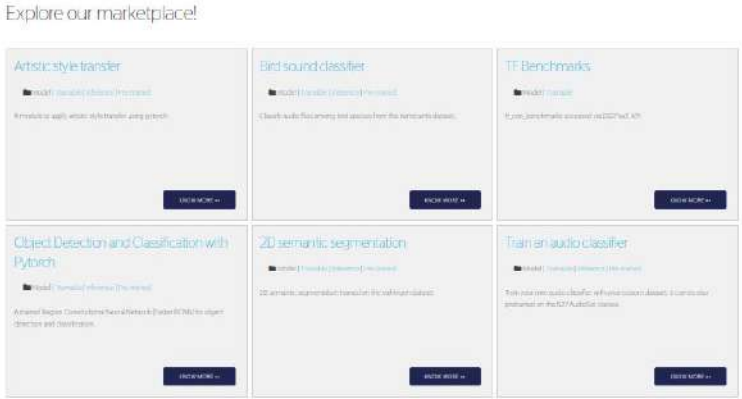
1. **Exploring the Marketplace**
2. **Using the Dashboard**
    a. Deploying a module
    b. Making inference
    c. Retraining a module on a new dataset
3. **Developing a new module**
    a. Deploying the DEEP development environment
    b. Using the cookiecutter
    c. Integrating it with DEEPaaS API
    d. Adding the model to the CI pipeline
    e. Adding the model to the Marketplace
4. **What's next?**
    a. New DEEPaaS features
    b. Friendlier UI for module inference
    c. Training Dashboard

1

Exploring the Marketplace

The Marketplace

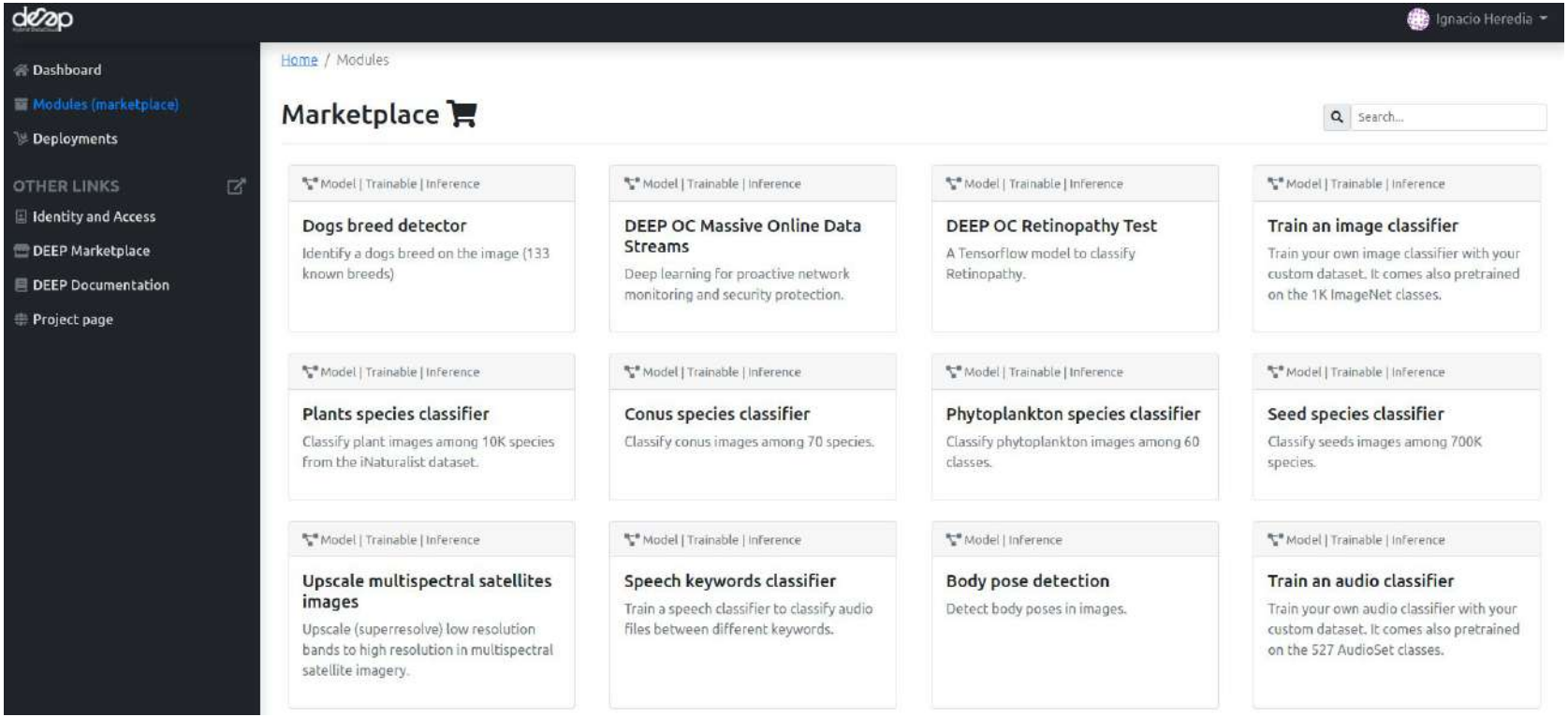




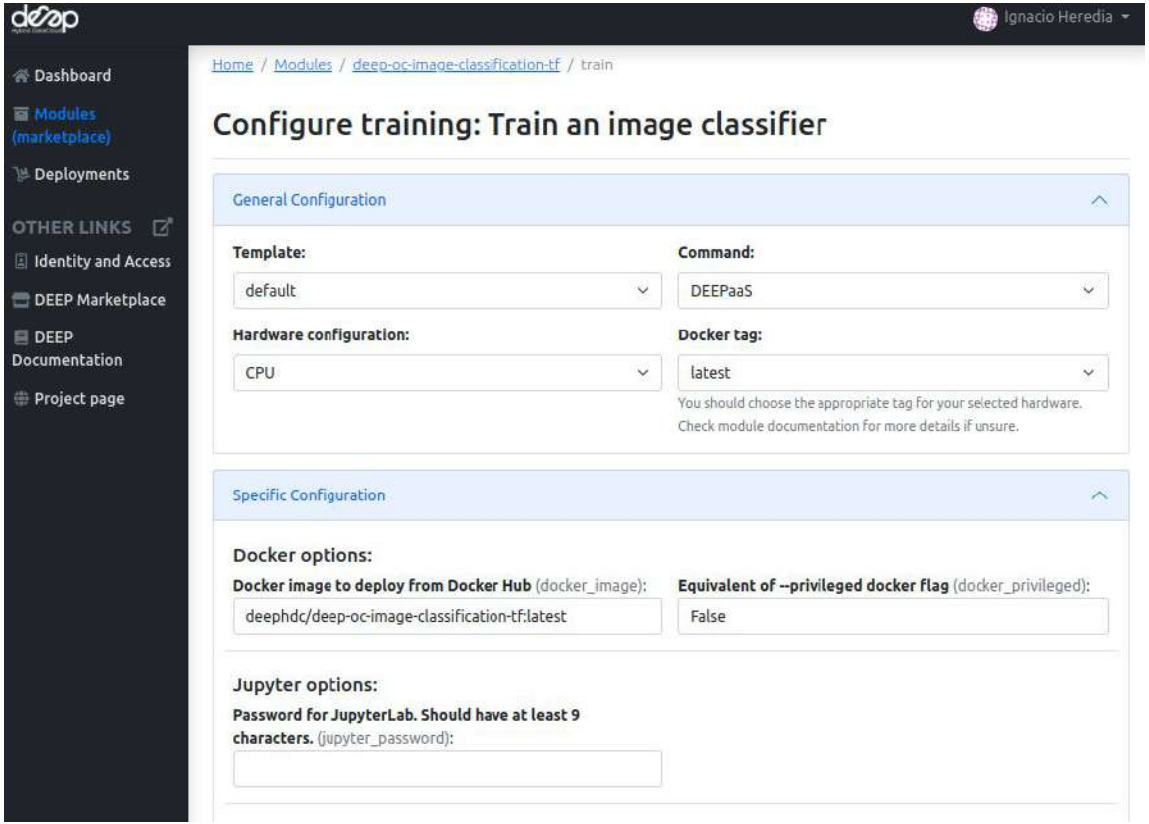
2

The Dashboard

The Dashboard - Module Overview



The Dashboard - Deploying a module



Configurable options

- **docker image** (from deep-oc, but also custom docker images)
- **hardware** (#cpus, #gpus, RAM)
- **storage** (OneData, Nextcloud volumes)
- **services** (DEEPaaS, JupyterLab)

The Dashboard - Making inference

Launch `image-classification-tf` module with DEEPaaS.

POST `/v2/models/imgclas/predict/` Make a prediction given the input data

Parameters

Name	Description
data **file** *(formData)*	Select the image you want to classify. *Default value* : null
urls **string** *(query)*	Select an URL of the image you want to classify. *Default value* : null
timestamp **string** *(query)*	Model timestamp to use for prediction. Group name: **testing** Choices: ['default_imagenet'] Type: str *Available values* : "default_imagenet" *Default value* : "default_imagenet"
ckpt_name **string** *(query)*	Checkpoint inside the timestamp to use for prediction. Group name: **testing** Type: str *Default value* : "final_model.h5"

Response body


```
{
  "status": "OK",
  "predictions": {
    "labels": [
      "lion",
      "fur_coat",
      "brown_bear",
      "hyena",
      "timber_wolf"
    ],
    "probabilities": [
      0.9763001203536987,
      0.005802886560559273,
      0.00480994675308466,
      0.0014703389024361968,
      0.001060070120729506
    ],
```

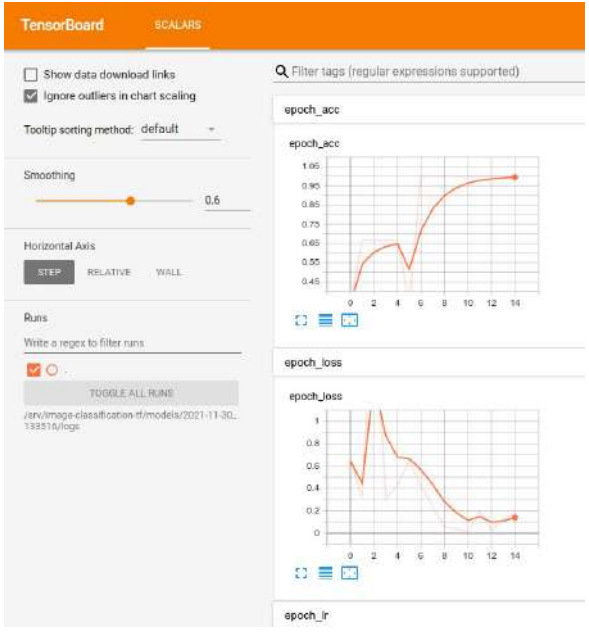
The Dashboard - Retraining a module

1) Launch `image-classification-tf` module with JupyterLab (remember adding password).
2) Copy some demo files to make a mock dataset.
3) Terminal: `deepaas-run --listen-ip 0.0.0.0` to launch DEEPaaS.

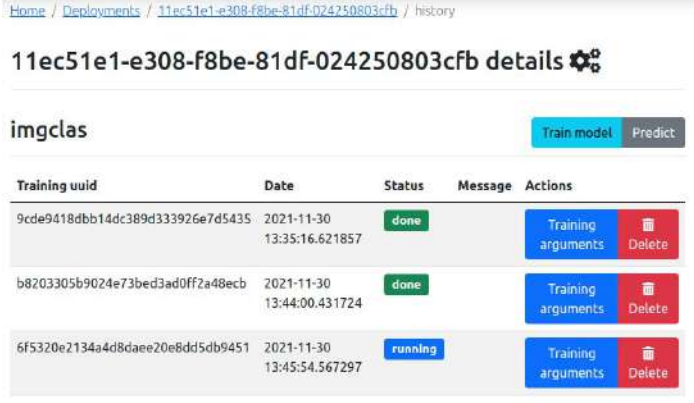
DEEPaaS

Monitor

TensorBoard

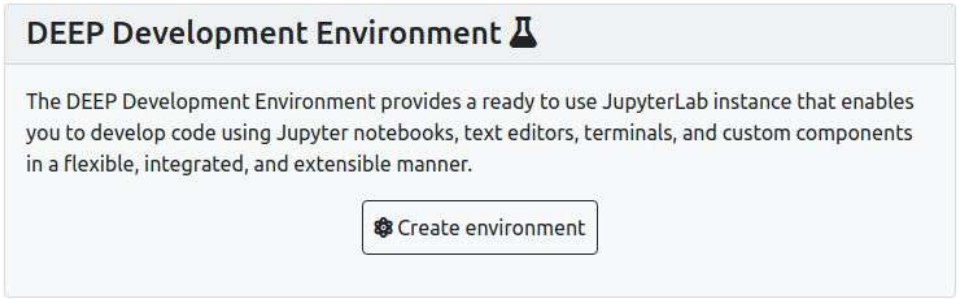


Training history



Develop your module

Developing - DEEP Development Environment



Configurable options

- **docker image** (from deep-oc, but also custom docker images). Eg:
  - **Tensorflow** docker
  - **Pytorch** docker
  - ...
- **hardware** (#cpus, #gpus, RAM)
- **storage** (OneData, Nextcloud volumes)
- **services** (DEEPaaS, **JupyterLab**)

Developing - DEEP Cookiecutter

This is the easiest way to develop any new module from scratch as it will take care of generating all the nitty-gritty details that we will cover in the following slides (entrypoints, files, Jenkinsfile, Dockerfile, etc).

- Use the command: `cookiecutter https://github.com/indigo-dc/cookiecutter-data-science`
- Answer questions:
  - Project name, description, version, license type
  - Author name, email, Github account
  - Dockerhub account, Docker base image
- This will generate two folders. Eg:
  - `mymodule`: This is where the project code is located

    → Example: https://github.com/deephdc/image-classification-tf
  - `DEEP-OC-mymodule`: This contains the Dockerfile of the project

    → Example: https://github.com/deephdc/DEEP-OC-image-classification-tf

Developing - Integrating with DEEPaaS

- Head over to `mymodule`. Any module that wants to integrate with DEEPaaS should have two minimum requirements:
  - it should define a file (eg. `mymodule/mymodule/api.py`) with the functions to interact with the module. These functions should define:
    - the model metadata
    - the input args for training
    - the input args for prediction
    - the response structure for prediction
    - the train function
    - the predict function
    - a model warming function for prediction

    `get_metadata()`
    `get_train_args()`
    `get_predict_args()`
    `schema`
    `train()`
    `predict ()`
    `warm()`

    → Minimal example: https://github.com/deephdc/demo_app/blob/master/demo_app/api.py

    → Full example: https://github.com/deephdc/image-classification-tf/blob/master/imgclas/api.py

  - it should define an entrypoint in `mymodule/setup.cfg` pointing to that file

    → Example: https://github.com/deephdc/demo_app/blob/master/setup.cfg#L25-L27

Developing - Customizing the Dockerfile

- Head over to `DEEP-OC-mymodule` and modify the Dockerfile following your needs:
  - install additional packages,
  - change the base image,
  - etc.

Developing - Continuous Integration

- Both `mymodule` and `DEEP-OC-mymodule` have their respective `Jenkinsfile` that define the actions to be taken when a change is committed to the repos.
- Typical workflows:
  - `mymodule/Jenkinsfile` will:
    - run PEP8 style analysis
    - trigger of `DEEP-OC-mymodule/Jenkinsfile`.

    → Example: https://github.com/deephdc/image-classification-tf/blob/master/Jenkinsfile

  - `DEEP-OC-mymodule/Jenkinsfile` will:
    - build Docker images for different branches (train/test) and different hardware (cpu/gpu)
    - upload the image to DockerHub
    - build Docker images of other dependent modules. For example, changes in the code of image-classification should rebuild all Docker images of applications that were trained with that code (plant classifier, seed classifier, etc).
    - refresh the module page in the Marketplace (see next step)

    → Example: https://github.com/deephdc/DEEP-OC-image-classification-tf/blob/master/Jenkinsfile

Developing - Integrating to the Marketplace

- Head over to `DEEP-OC-mymodule` and modify `metadata.json` with the info relevant to your module. This is the information that will appear in the Marketplace page.
- Make a Pull Request to add your module here. This will create the Jenkins pipeline for your module and will add the module to the Marketplace and the Training Dashboard.

Congratulations, you're done!

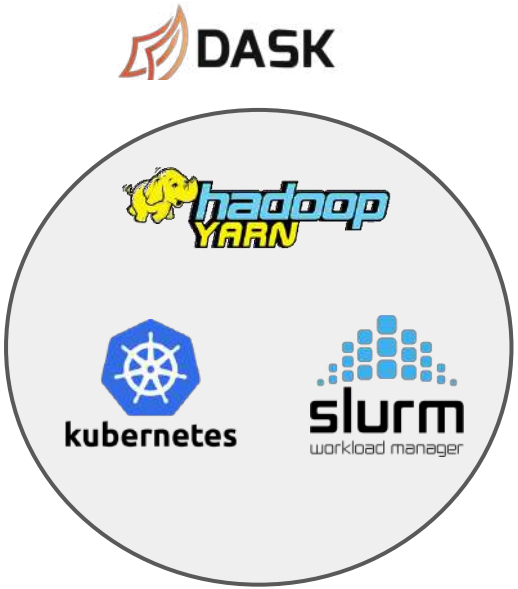
4

What's next?

What's next? - New DEEPaaS features

- Integration with Dask

  Mature



- Easier module integration via decorators/hints

  Midterm

Before (webargs)

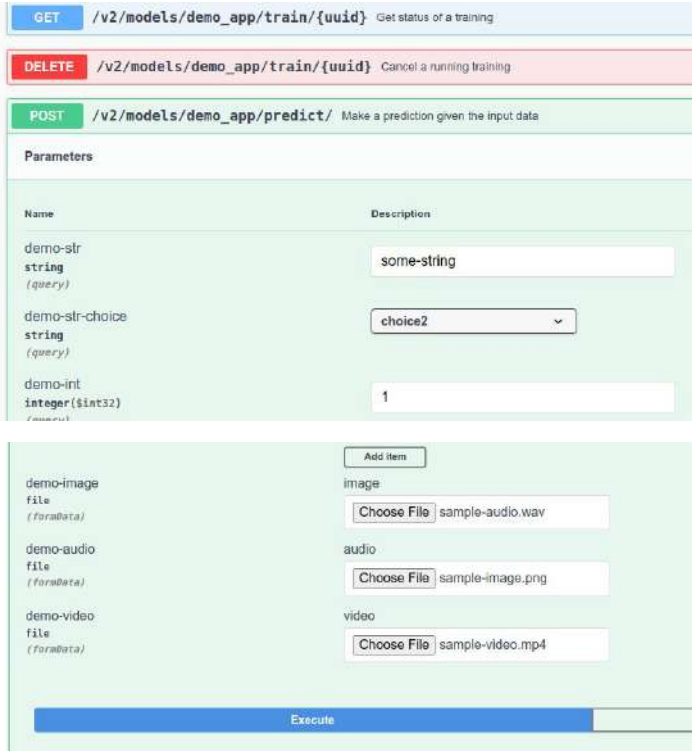
```python
from webargs import fields, validate

def get_predict_args():
    arg_dict = {
        "demo-str": fields.Str(
            required=False,
            missing='some-string',
        ),
        "demo-int": fields.Int(
            required=False,
            missing=1,
        ),
        }


schema = {
    "demo-list": fields.List(
        fields.Float()
        ),
}


def predict(**kwargs):
    return {"demo-list": [1, 2, 3]}
```

After (type hints)

```python
def predict("demo-str": str,
            "demo-int": int,
            ) -> dict:
    return {"demo-list": [1, 2, 3]}
```

What's next? - Friendlier inference UI

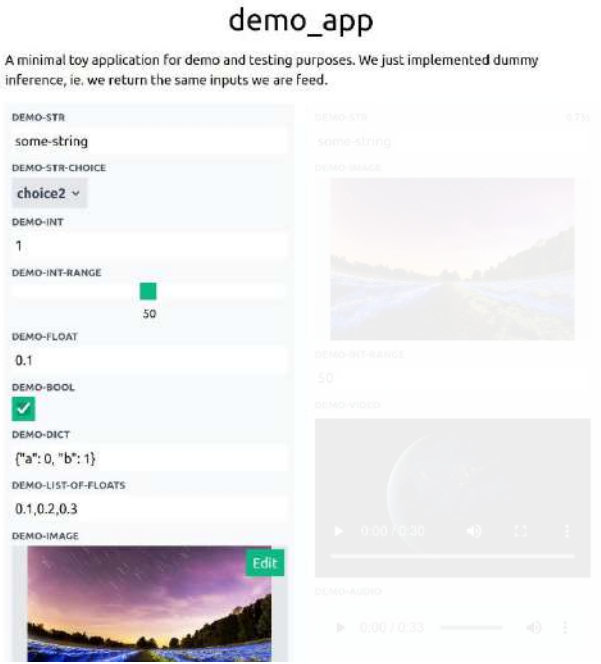
Before (Swagger UI)



Inputs

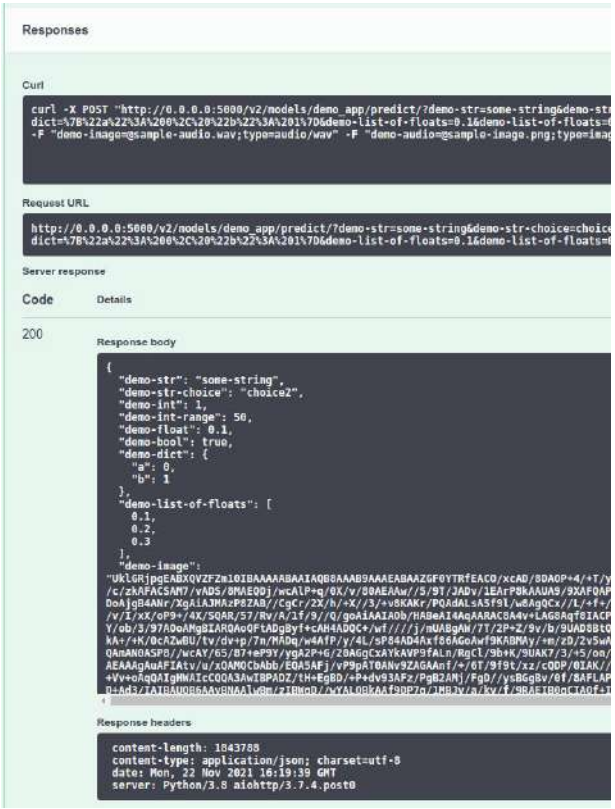
After (Gradio based)

Mature



What's next? - Friendlier inference UI

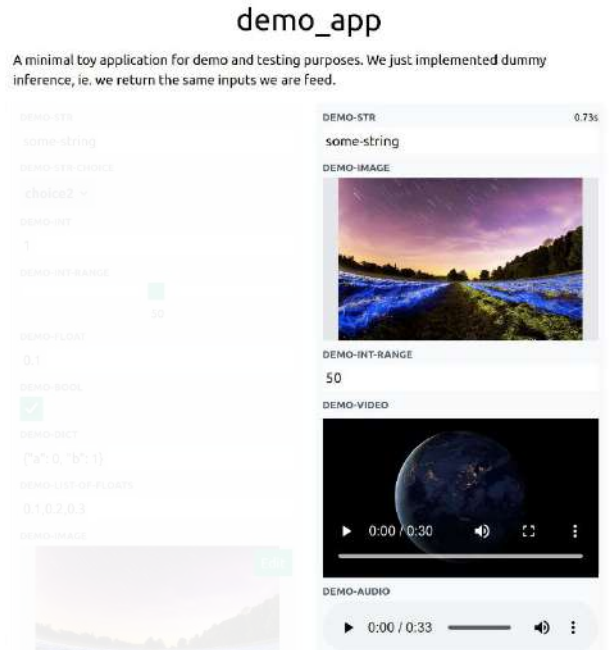
Before (Swagger UI)



Outputs

After (Gradio based)

Mature



What's next? - Training dashboard

- Organizing training run in experiments — Midterm
  - hyperparameter optimization
  - easier side-by-side comparison of training runs

- Richer module metadata language, to keep track of: — Project
  - training datasets
  - models
  - training execution pipelines

Questions

Horizon 2020
European Union funding
for Research & Innovation

DEEP-Hybryd-DataCloud has received funding from the European Horizon 2020 research and innovation programme under grant agreement N°777435