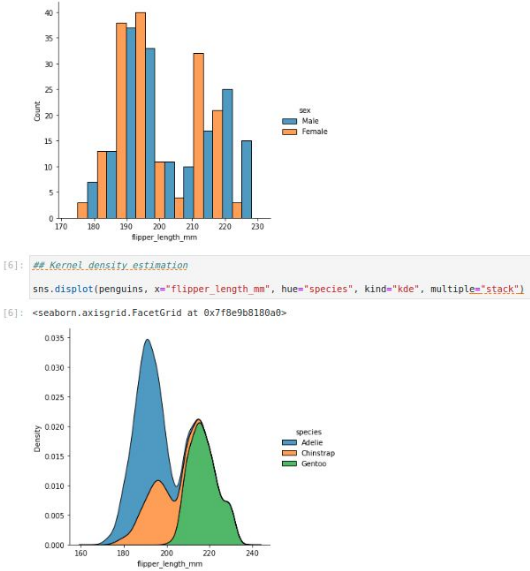
Galaxy Jupyterlab for AI

Anup Kumar

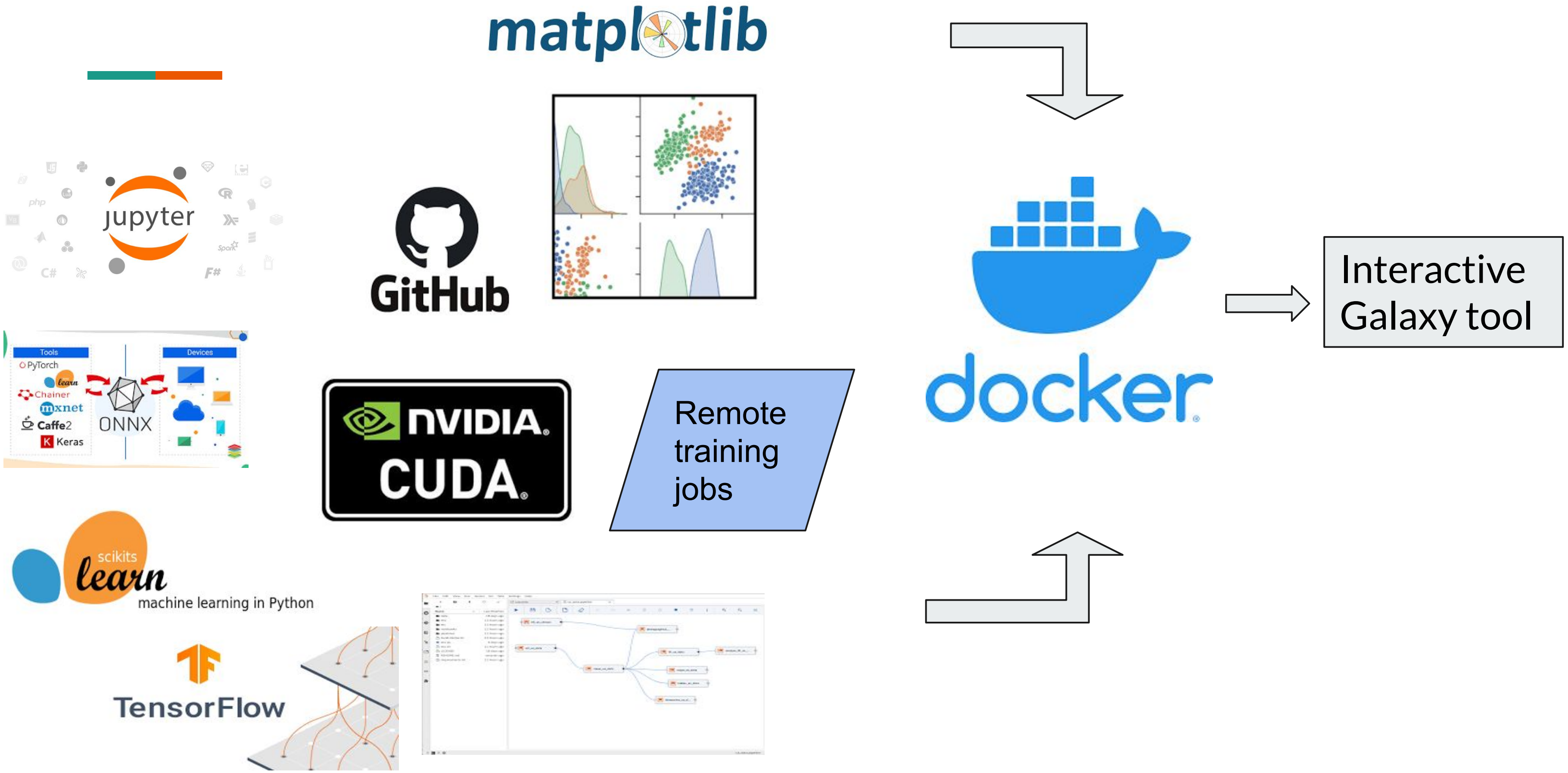
Freiburg Galaxy team, Bioinformatics group,
University of Freiburg, Germany

Jupyterlab

- Jupyter notebooks - popular editor
  - Data science
  - Scientific computing
  - Machine learning
  - Learn to code. E.g Python
- Simple and fast way to create prototypes
- No need for any package installation
- Easy to share an analysis
- Run on web



```
[6]: ## Kernel density estimation

     sns.displot(penguins, x="flipper_length_mm", hue="species", kind="kde", multiple="stack")

[6]: <seaborn.axisgrid.FacetGrid at 0x7f8e9b8180a0>
```

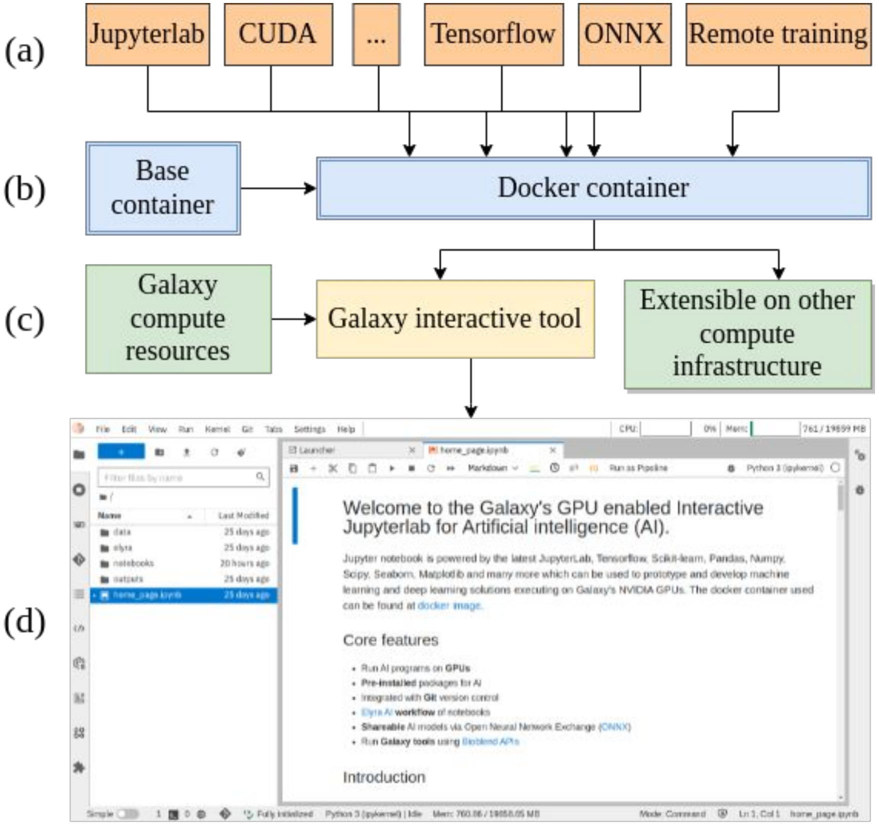


Remote training jobs

Interactive Galaxy tool

Features

- Faster computations using GPU
- Ready to use, pre-installed packages
  - ML: Scikit-learn, Tensorflow, CUDA, OpenCV, ONNX AI models
  - Data manipulation: Pandas, H5py, NumPy, Scipy, Nibabel, …
  - Visualizations: Matplotlib, Seaborn
- Git integration
- Workflows of notebooks (Elyra AI)
- Communicate with Galaxy (Bioblend)
- Remote training (using a separate Galaxy tool)
- Miscellaneous - resource dashboards, collapsible headers …



Welcome to the Galaxy's GPU enabled Interactive Jupyterlab for Artificial intelligence (AI).

Jupyter notebook is powered by the latest JupyterLab, Tensorflow, Scikit-learn, Pandas, Numpy, Scipy, Seaborn, Matplotlib and many more which can be used to prototype and develop machine learning and deep learning solutions executing on Galaxy's NVIDIA GPUs. The docker container used can be found at docker image.

Core features

- Run AI programs on **GPUs**
- **Pre-installed** packages for AI
- Integrated with **Git** version control
- Elyra AI **workflow** of notebooks
- **Shareable** AI models via Open Neural Network Exchange (ONNX)
- Run **Galaxy tools** using Bioblend APIs

Introduction

Base container from Jupyter Docker stacks: jupyter/tensorflow-notebook:tensorflow-2.6.0

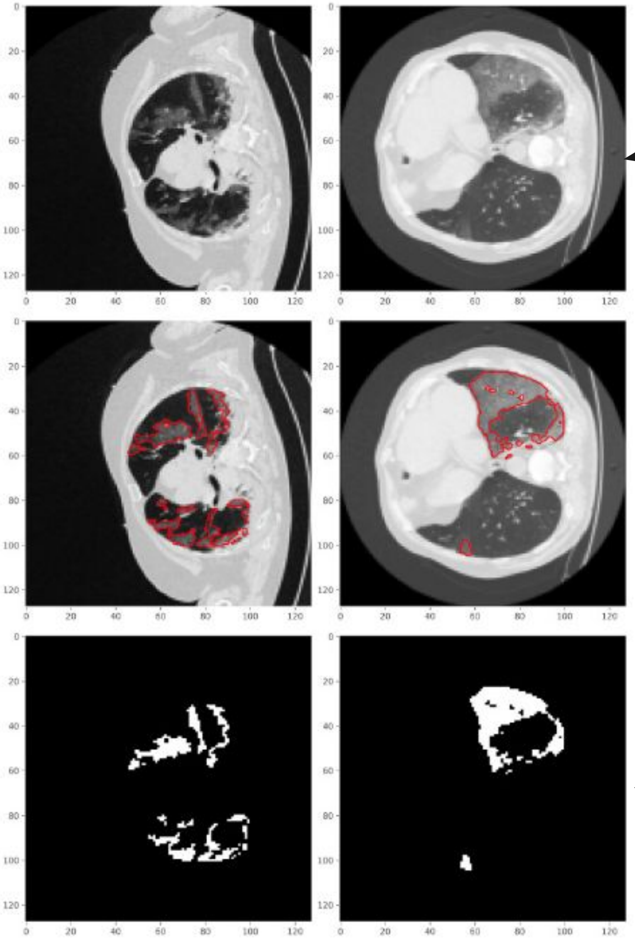
Comparison with other notebook infrastructures

	Google Colab	Kaggle Kernel	Galaxy Jupyterlab
Memory/disk space	~ 12 GB/70 GB	~ 16 GB/73 GB	~ **20** GB/**1** TB
GPU/TPU	Yes/**Yes**	Yes/**Yes**	Yes/No
Max usage time	12 hrs	12 hrs/session, 30 hrs of GPU/week, 20 hrs of TPU/week	**No time restriction** on GPU usage, notebook and job execution
Dynamic resources	Yes	Yes	**Fixed and guaranteed**
Remote model training	No	No	**Yes**

https://research.google.com/colaboratory/faq.html, https://www.kaggle.com/general/108481, https://www.kaggle.com/page/GPU-tips-and-tricks

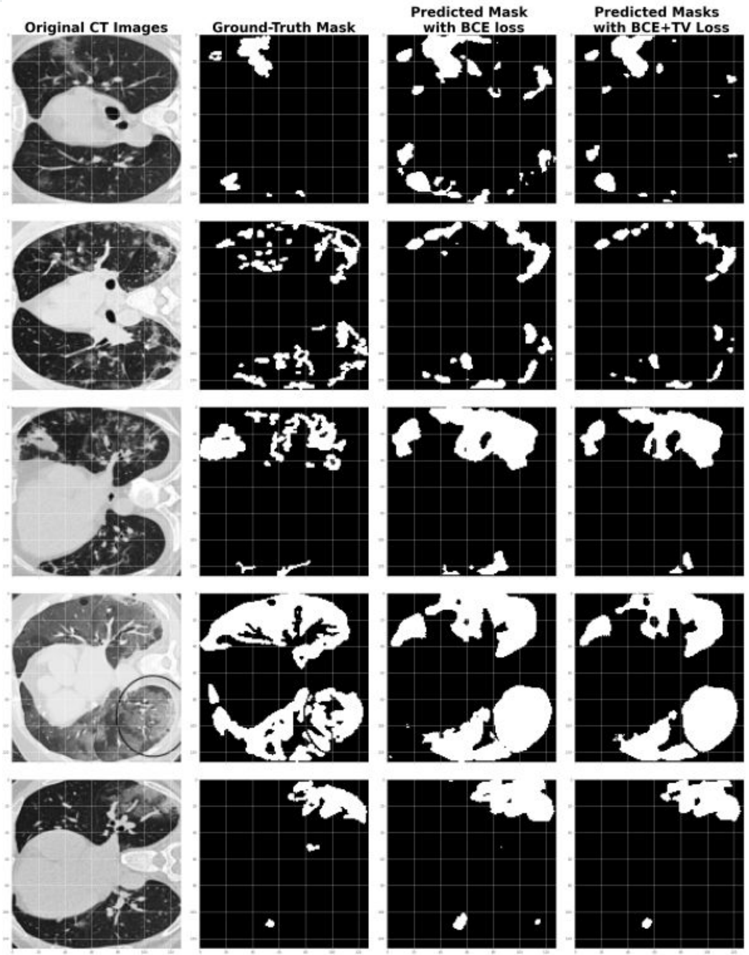
Use-case 1: COVID-19 CT scan image segmentation

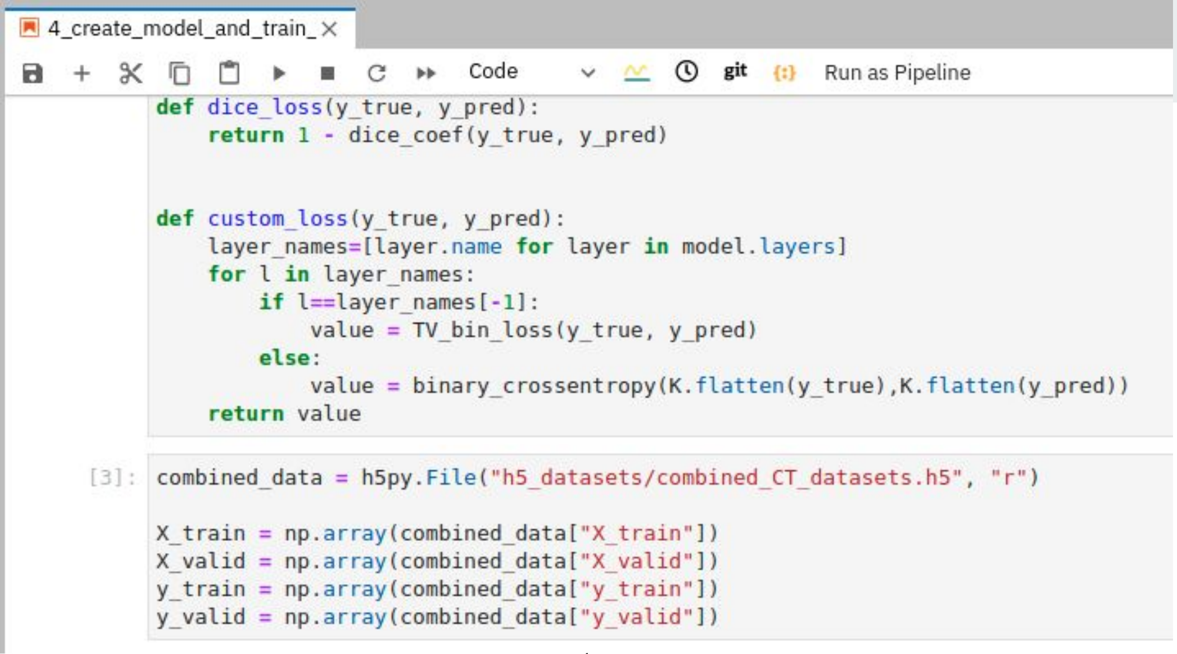
- Reproduce results from published work: "COVID TV-Unet: Segmenting COVID-19 chest CT images using connectivity imposed Unet" [1]
- Entire analysis in Galaxy Jupyterlab notebooks
- Save model as ONNX
- Fetch trained model (as ONNX file) from Galaxy and make predictions
- Remote model training or in notebooks using GPU
- For remote training: convert datasets to H5 (save as matrices)



CT scans

Masks


Original CT Images
Ground-Truth Mask
Predicted Mask with BCE loss
Predicted Masks with BCE+TV Loss

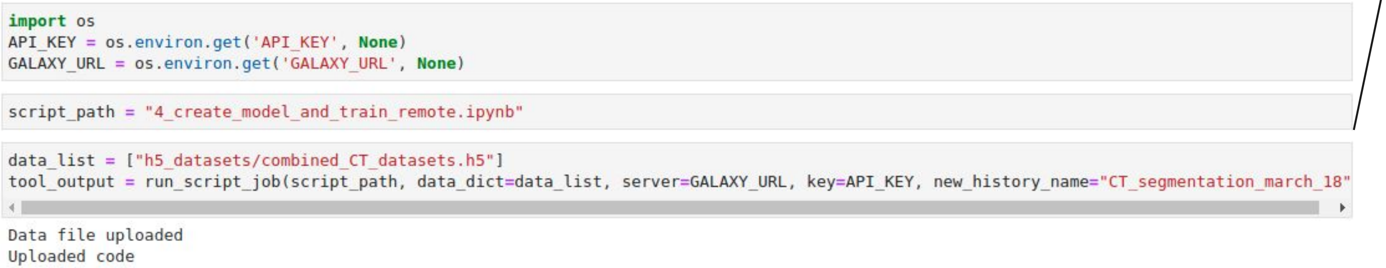


```
def dice_loss(y_true, y_pred):
    return 1 - dice_coef(y_true, y_pred)


def custom_loss(y_true, y_pred):
    layer_names=[layer.name for layer in model.layers]
    for l in layer_names:
        if l==layer_names[-1]:
            value = TV_bin_loss(y_true, y_pred)
        else:
            value = binary_crossentropy(K.flatten(y_true),K.flatten(y_pred))
    return value
```

```
combined_data = h5py.File("h5_datasets/combined_CT_datasets.h5", "r")

X_train = np.array(combined_data["X_train"])
X_valid = np.array(combined_data["X_valid"])
y_train = np.array(combined_data["y_train"])
y_valid = np.array(combined_data["y_valid"])
```

1. Script to run

2. Send script to run remotely



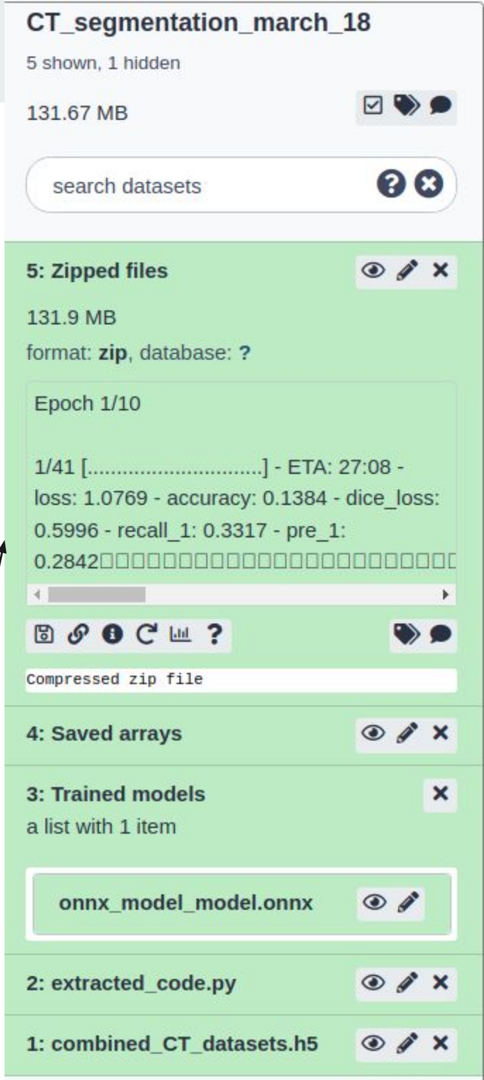
```
import os
API_KEY = os.environ.get('API_KEY', None)
GALAXY_URL = os.environ.get('GALAXY_URL', None)
```

```
script_path = "4_create_model_and_train_remote.ipynb"
```

```
data_list = ["h5_datasets/combined_CT_datasets.h5"]
tool_output = run_script_job(script_path, data_dict=data_list, server=GALAXY_URL, key=API_KEY, new_history_name="CT_segmentation_march_18"
```

```
Data file uploaded
Uploaded code
```

3. History



CT_segmentation_march_18

5 shown, 1 hidden

131.67 MB

search datasets

5: Zipped files

131.9 MB

format: **zip**, database: **?**

```
Epoch 1/10

1/41 [..............................] - ETA: 27:08 -
loss: 1.0769 - accuracy: 0.1384 - dice_loss:
0.5996 - recall_1: 0.3317 - pre_1:
0.2842□□□□□□□□□□□□□□□□□□□□□□□
```

```
Compressed zip file
```

4: Saved arrays

3: Trained models

a list with 1 item

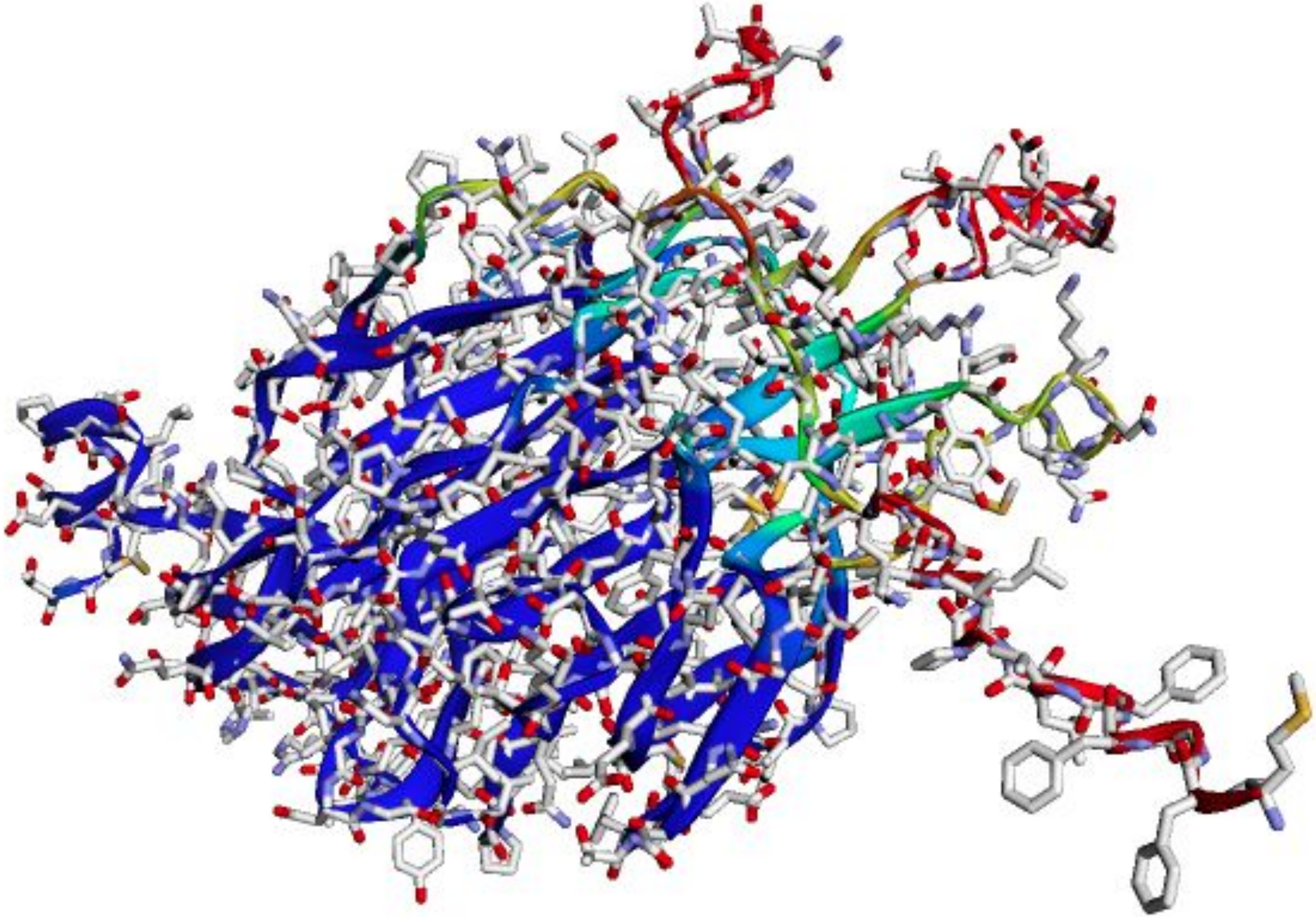
onnx_model_model.onnx

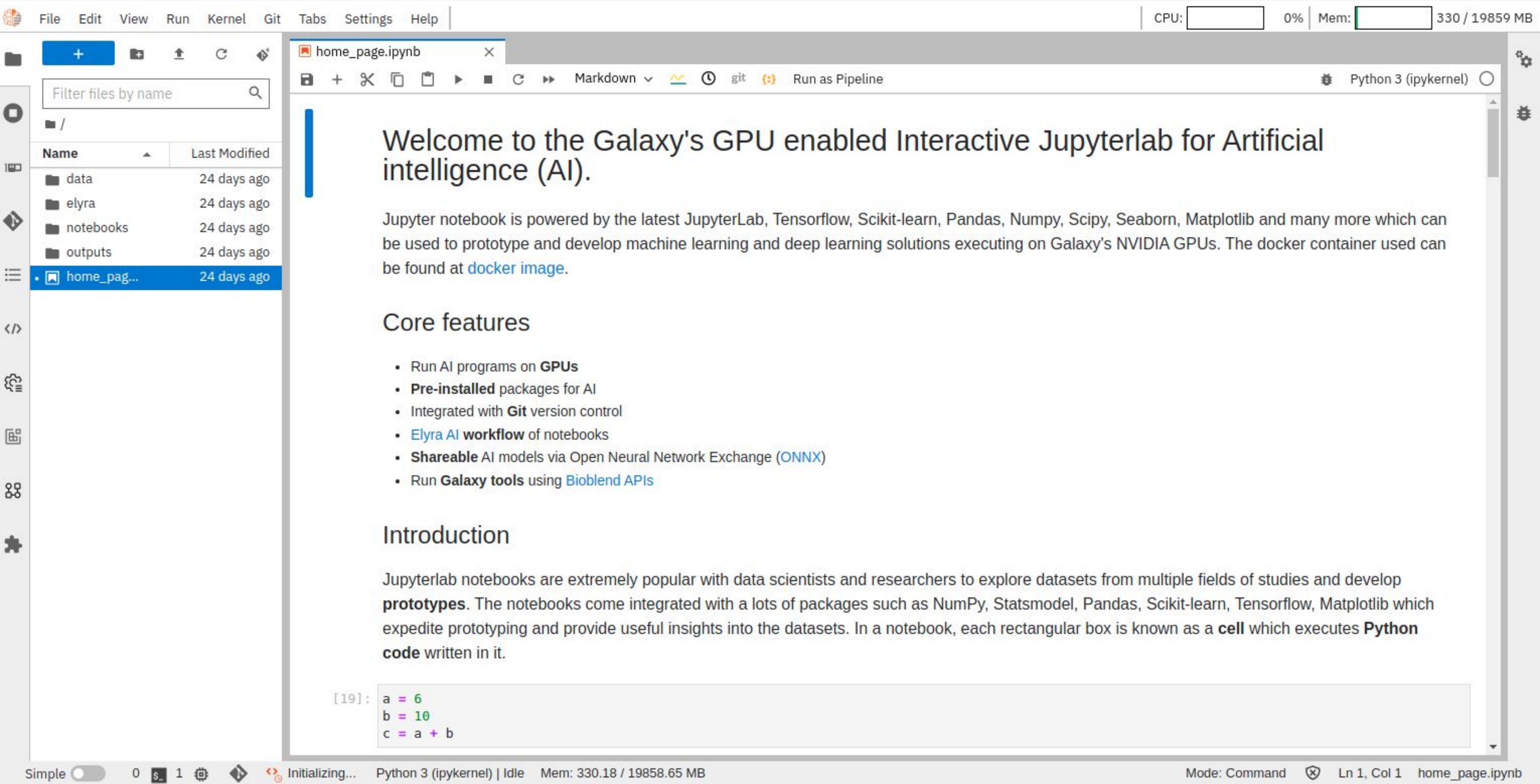
2: extracted_code.py

1: combined_CT_datasets.h5

Use-case 2: Predict protein 3D structures using Colabfold

- Colabfold: Predict 3D structures of proteins using only sequences [2]
- Less memory intensive than Alphafold2, faster prediction
- Use Alphafold2 weights
- Addition of only two packages in Galaxy Jupyterlab - colabfold and JAX
- Acceleration of the prediction of 3D conformation via GPU
- Notebook
- Next slide: Predicted 3D structure of 300 amino-acid long spike protein of SARS-CoV-2





Welcome to the Galaxy's GPU enabled Interactive Jupyterlab for Artificial intelligence (AI).

Jupyter notebook is powered by the latest JupyterLab, Tensorflow, Scikit-learn, Pandas, Numpy, Scipy, Seaborn, Matplotlib and many more which can be used to prototype and develop machine learning and deep learning solutions executing on Galaxy's NVIDIA GPUs. The docker container used can be found at docker image.

Core features

- Run AI programs on **GPUs**
- **Pre-installed** packages for AI
- Integrated with **Git** version control
- Elyra AI **workflow** of notebooks
- **Shareable** AI models via Open Neural Network Exchange (ONNX)
- Run **Galaxy tools** using Bioblend APIs

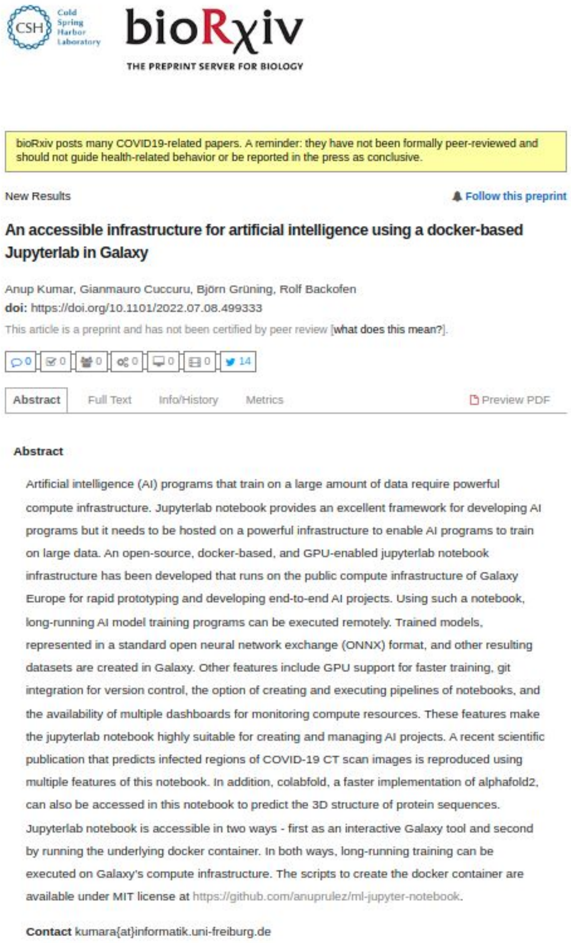
Introduction

Jupyterlab notebooks are extremely popular with data scientists and researchers to explore datasets from multiple fields of studies and develop **prototypes**. The notebooks come integrated with a lots of packages such as NumPy, Statsmodel, Pandas, Scikit-learn, Tensorflow, Matplotlib which expedite prototyping and provide useful insights into the datasets. In a notebook, each rectangular box is known as a **cell** which executes **Python code** written in it.

```
[19]: a = 6
      b = 10
      c = a + b
```

Running instance, GTN tutorial..

- Running instance
- GTN tutorial
- Preprint
- Dockerfile
- Submitted to GigaScience and is under review
- Thanks to Gianmauro, Bjoern and Rolf



CSH Cold Spring Harbor Laboratory

bioRχiv

THE PREPRINT SERVER FOR BIOLOGY

bioRxiv posts many COVID19-related papers. A reminder: they have not been formally peer-reviewed and should not guide health-related behavior or be reported in the press as conclusive.

New Results

Follow this preprint

An accessible infrastructure for artificial intelligence using a docker-based Jupyterlab in Galaxy

Anup Kumar, Gianmauro Cuccuru, Björn Grüning, Rolf Backofen

doi: https://doi.org/10.1101/2022.07.08.499333

This article is a preprint and has not been certified by peer review [what does this mean?].

Abstract

Artificial intelligence (AI) programs that train on a large amount of data require powerful compute infrastructure. Jupyterlab notebook provides an excellent framework for developing AI programs but it needs to be hosted on a powerful infrastructure to enable AI programs to train on large data. An open-source, docker-based, and GPU-enabled jupyterlab notebook infrastructure has been developed that runs on the public compute infrastructure of Galaxy Europe for rapid prototyping and developing end-to-end AI projects. Using such a notebook, long-running AI model training programs can be executed remotely. Trained models, represented in a standard open neural network exchange (ONNX) format, and other resulting datasets are created in Galaxy. Other features include GPU support for faster training, git integration for version control, the option of creating and executing pipelines of notebooks, and the availability of multiple dashboards for monitoring compute resources. These features make the jupyterlab notebook highly suitable for creating and managing AI projects. A recent scientific publication that predicts infected regions of COVID-19 CT scan images is reproduced using multiple features of this notebook. In addition, colabfold, a faster implementation of alphafold2, can also be accessed in this notebook to predict the 3D structure of protein sequences. Jupyterlab notebook is accessible in two ways - first as an interactive Galaxy tool and second by running the underlying docker container. In both ways, long-running training can be executed on Galaxy's compute infrastructure. The scripts to create the docker container are available under MIT license at https://github.com/anuprulez/ml-jupyter-notebook.

Contact kumara{at}informatik.uni-freiburg.de

Thank you for your attention!

Questions?

References

1. Image segmentation: https://www.sciencedirect.com/science/article/pii/S2666990021000069?via%3Dihub
2. Colabfold: https://www.nature.com/articles/s41592-022-01488-1
3. Docker image: https://github.com/anuprulez/ml-jupyter-notebook
4. https://www.docker.com/
5. https://www.nvidia.com/en-in/
6. https://jupyter.org/
7. https://www.tensorflow.org/
8. https://scikit-learn.org/stable/
9. https://matplotlib.org/
10. https://github.com/
11. https://pandas.pydata.org/
12. https://keras.io/examples/nlp/text_classification_with_transformer/
13. https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf