Digital Research Alliance of Canada

Working with Jupyter on Clusters

Jinhui Qin

SHARCNET | Compute Ontario | Digital Research Alliance of Canada
Western University

Western University · Canada

SHARCNET™

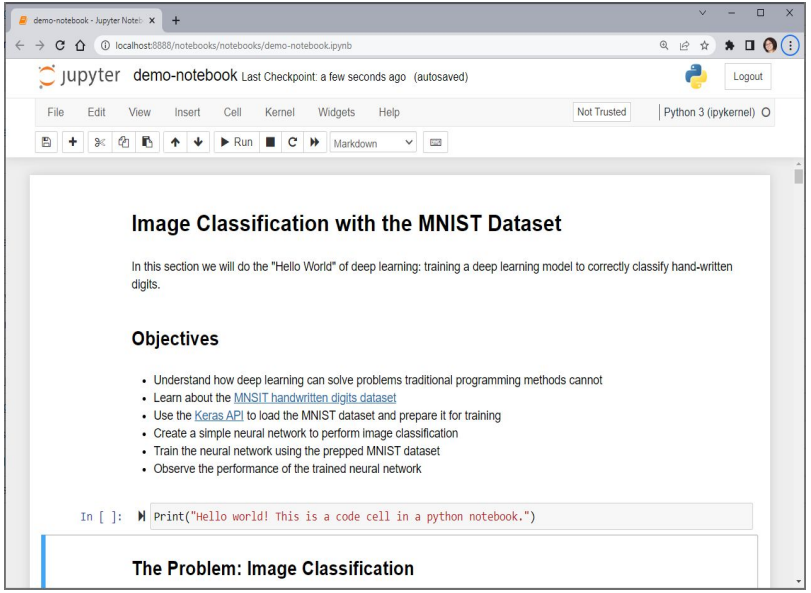
Compute • Calcul Ontario

Outline

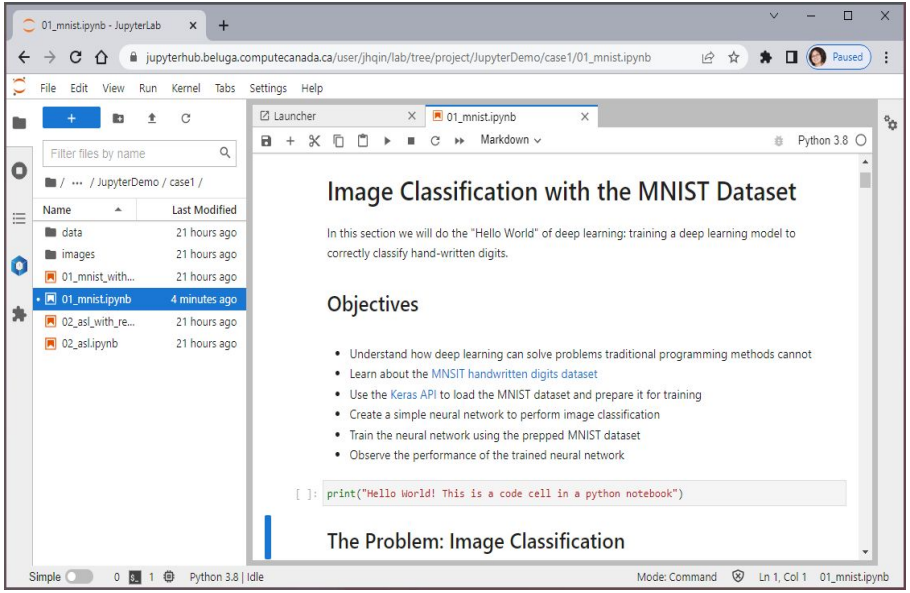
- Notebook interface
  - Jupyter Notebook
  - JupyterLab
- Launching a notebook server on a cluster
  - Via JupyterHub
  - Via Command line
- Use cases to demo
  - Case 1 : via JupyteHub without command line
  - Case 2: work with a commercial software package
  - Case 3: work with a container

Notebook Interface

Jupyter Notebook
— the classic interface



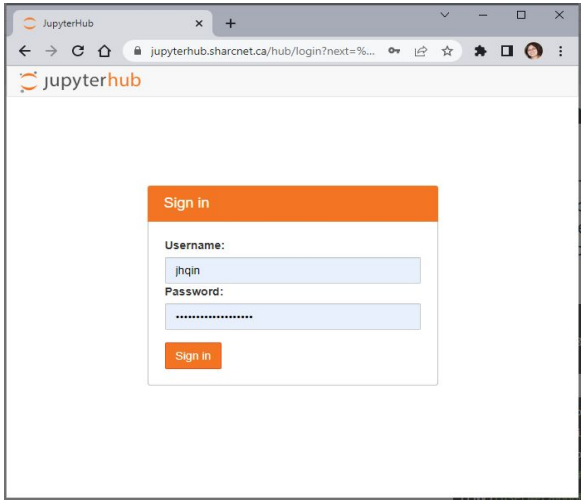
JupyterLab
— the new interface

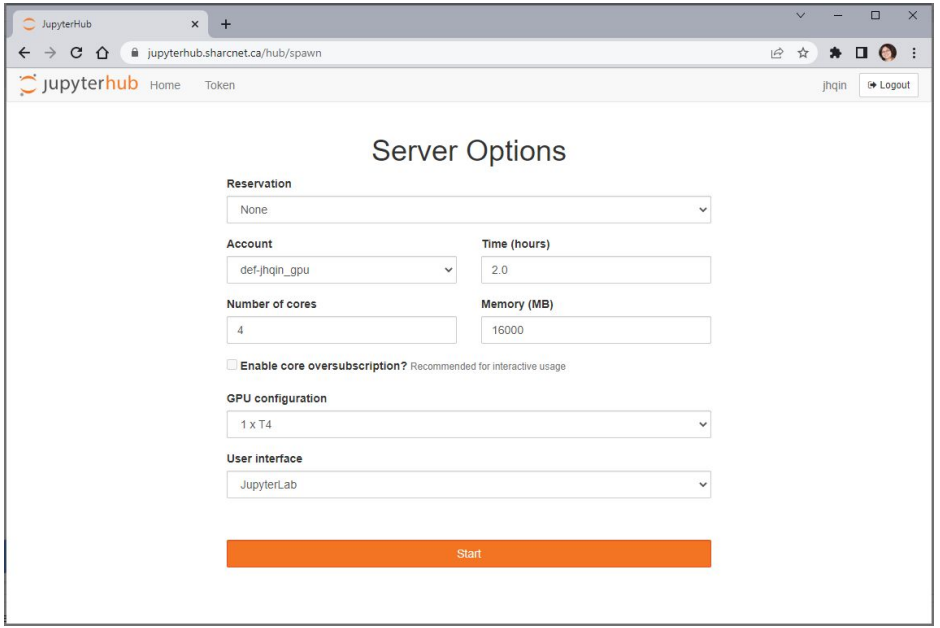


Launching a notebook server on a cluster

Via JupyterHub -- a web portal for launching a notebook server

- https://docs.alliancecan.ca/wiki/JupyterHub#JupyterHub_on_clusters





Launching a notebook server on a cluster

Via Command line

- ssh to a cluster
- load a python module and other modules if needed, e.g. scipy-stack
- create a python virtual env and Install jupyerlab and other libs if needed
- request an interactive session on a compute node
- launch jupyter-lab from the virtual env.

```
[you@local]$ ssh user@cluster.computecanada.ca
```

```
[user@cluster-login]$ module load python scipy-stack <other-modules>
[user@cluster-login]$ virtualenv --no-download ENV
[user@cluster-login]$  source ENV/bin/activate
(ENV)[user@cluster-login]$ pip install jupyterlab <other-libs>
(ENV)[user@cluster-login]$ deactivate
```

```
user@cluster-login]$ salloc --account=def-user --cpus-per-task=4 \
--mem=16G --gres=gpu:t4:1 --time=1:0:0
[user@node###]$ module load python scipy-stack <other-modules>
[user@node###]$ source ENV/bin/activate
(ENV)[user@node###]$ jupyter-lab --no-browser --ip $(hostname -f)
```

Features

	JupyterHub	Command line
Connecting	Automatically	*Need to setup a SSH tunnel*
Default environment	Pre-configured, *temporary*	Self-made, *persistent*
Internet access	No (on most clusters)	No (on most clusters)
Accessing files on a cluster	Yes	Yes
Installing packages	Yes (only those pre-built on the system)	Yes
Working with a self-built virtual env	Possible	Yes
Using modules	Yes	Yes
Using GPUs	Yes	Yes
Working with a container	No	Yes

Demo on Cluster
Case 1

Use Cases

Case 1: run a python notebook for image classification on GPUs

- Launch a JupyterLab on a GPU node from a JupyterHub
- Work in the pre-configured environment
- Load modules, e.g. scipy-stack for using numpy, pandas, matplotlib, etc.
- Install additional python libraries, e.g. tensorflow
- Check GPU status if available
- Upload / Download / Export files
- No need to work in command line on a terminal

JupyteHub on clusters: https://docs.alliancecan.ca/wiki/JupyterHub#JupyterHub_on_clusters

Demo on Cluster
Case 2

Use Cases

Case 2: run a notebook to work with a commercial package via a *persistent* python virtual environment

- An example notebook to work with Gurobi optimizer with machine learning
- Build a python env on a cluster with required packages installed, e.g. gurobipy, sklearn, etc.
- Two options for launching a JupyterLab, ie. via JupyterHub and via Command line

```
[user@cluster-login]$ module load python scipy-stack gurobi
[user@cluster-login]$ virtualenv --no-download gurobiENV
[user@cluster-login]$ source gurobiENV/bin/activate
(gurobiENV)[user@cluster-login]$ cd ${GUROBI_HOME}
(gurobiENV)[user@cluster-login]$ python setup.py build --build-base /tmp/${USER} install
(gurobiENV)[user@cluster-login]$ pip install --no-index scikit-learn seaborn
```

Use Cases

Case 2: run a notebook to work with a commercial package via a *persistent* python virtual environment

- Option 1. launching a JupterLab from a JupyterHub
  - Add a customized kernel from *a command line on the cluster*

```
(gurobiENV)[user@cluster-login]$ pip install --no-index ipykernel
(gurobiENV)[user@cluster-login]$ python -m ipykernel install --user --name gurobiEnv
```

  - Check available Jupyter kernels, remove a kernel if no longer needed

```
(gurobiENV)[user@cluster-login]$ jupyter kernelspec list
(gurobiENV)[user@cluster-login]$ jupyter kernelspec uninstall <kernel-name>
```

Use Cases

Case 2: running a notebook for using a commercial package via a python virtual environment

- Option 2. launching a JupterLab from a command line on a compute node
  - Need to install jupyterlab *in the same virtual env*

```
(gurobiENV)[user@cluster-login]$ pip install --no-index jupyterlab
(gurobiENV)[user@cluster-login]$ deactivate

[user@cluster-login]$ salloc --account=def-user --cpus-per-task=4 --mem=16G --time=1:0:0

[user@node###]$ module load python scipy-stack gurobi
[user@node###]$ source gurobiENV/bin/activate
(gurobiENV)[user@node####]$ jupyter-lab --no-browser --ip $(hostname -f)
```

Use Cases

Case 2: running a notebook for using a commercial package via a python virtual environment

- Option 2. launching a JupterLab from a command line on a compute node

```
(gurobiENV)[user@node###]$ jupyter-lab --no-browser --ip $(hostname -f)
... ...
    http://<node-hostname>:8888/lab?token=109c369980ab346bcd129eb...
```

  - Setup a SSH tunnel on a second terminal from your local computer

```
[you@local]$ ssh -L 9999:<node-hostname>:8888 user@cluster.computecanada.ca
[user@cluster-login]$
```

  - The URL to connect on a local web browser would be:

    `http://localhost:9999/lab?token=109c369980ab346bcd129eb...`

Demo on Cluster
Case 3

Use Cases

Case 3: run a notebook using packages in a container

- Working with RAPIDS via a Singularity container on a GPU node (a previous demo)
- Working with Anaconda via a Singularity container
  - Based from a Docker image for Anaconda3 on Dockerhub
  - Request an interactive session on a compute node
  - Load singularity module and access the container shell

```
[user@gra-login]$ salloc --account=def-user --cpus-per-task=4 --mem=16G --time=1:0:0
[user@node###]$ module load singularity
[user@node###]$ singularity shell -B /home -B /project -B /scratch mycontainer.sif
Singularity>
```

Use Cases

Case 3: run a notebook using packages in a container

- Launch a jupyerLab from the container shell

```
Singularity> jupyter-lab --no-browser --ip $(hostname -f)
... ...
    http://<node-hostname>:8888/lab?token=109c369980ab346bcd129eb…
```

- Setup a SSH tunnel on a second terminal from your local computer

```
[you@local]$ ssh -L 9999:<node-hostname>:8888 user@cluster.computecanada.ca
[user@cluster-login]$
```

- The URL to connect on a local web browser would be:

  `http://localhost:9999/lab?token=109c369980ab346bcd129eb…`