

CLOUD COMPUTING WITH GOOGLE CLOUD & DOCKER

AI FRAMEWORKS

Brendan Guillouet

November ??, 2020

Institut National des Sciences Appliquées

TABLE OF CONTENTS

Python Script

Google Cloud

Pipeline d'exécution

Docker

PYTHON SCRIPT

IA FRAMEWORKS - TOOLS

ML Python
Libraries



Python
Environment



Viz' Python Libraries



Framework
& Tool



docker



Google Cloud Platform



git

WHY USING SCRIPT?

Jupyter limits :

- it's an exploration tool
 - but cloud machine are accounted on an hourly base.
- Non-linear workflow.
 - Easy to write messy code.
- Not designed to handle large-scale experiment.
- Not designed for production.
 - Can't be run from terminal, no test procedure.

WHY USING SCRIPT?

Jupyter limits :

- it's an exploration tool
 - but cloud machine are accounted on an hourly base.
- Non-linear workflow.
 - Easy to write messy code.
- Not designed to handle large-scale experiment.
- Not designed for production.
 - Can't be run from terminal, no test procedure.

⇒ Exploration work : *Jupyter*.

WHY USING SCRIPT?

Jupyter limits :

- it's an exploration tool
 - but cloud machine are accounted on an hourly base.
- Non-linear workflow.
 - Easy to write messy code.
- Not designed to handle large-scale experiment.
- Not designed for production.
 - Can't be run from terminal, no test procedure.

⇒ Exploration work : *Jupyter*.

⇒ Large-scale/production work : *Write Script!*

SCRIPT EXECUTION

File *script.py*

```
a = 5
b = 3
c = a + b
print("The answer is %d" %c)
```

Terminal

```
bguillou $> python script.py
bguillou $> The answer is 8
```


Write two scripts :

- **learning.py** : to learn a model, save it in the *model* directory, save results in the *results* directory and
- **prediction.py** : to generate prediction and save it in the *results* directory

on *CatsVsDogs* data.

⇒ Ensure that the **complete workflow is working** locally before pushing the code on the instance.

LIBRAIRIE ARGPARSE

File *script.py*

```
import argparse

parser = argparse.ArgumentParser()
parser.add_argument('--a', type=int, default=5)
parser.add_argument('--b', type=int, default=3)

args = parser.parse_args()

c= args.a + args.b
print("The answer is %d" %c)
```

Terminal

```
bguillou $> python script.py
bguillou $> The answer is 8
bguillou $> python script.py --a 4
bguillou $> The answer is 7
bguillou $> python script.py --a 4 --b 2
bguillou $> The answer is 7
```

LIBRAIRIE PICKLE

File *learning.py*

```
import pickle
...
results = {"learning_time" : lt, "accuracy" : acc}
pickle.dump(results, open("/User/bguillouet/data/results.pkl", "wb"))
```

File *explore_results.py*

```
import pickle
results = pickle.load(open("/User/bguillouet/data/results.pkl", "rb"))
print(results)
```

Terminal

```
bguillou $> ls data/
bguillou $>
bguillou $> python learning.py
bguillou $> ls data/
bguillou $> results.pkl
bguillou $> python explore_results.py
bguillou $> {"learning_time" : lt, "accuracy" : acc}
```


GOOGLE CLOUD

IA FRAMEWORKS - TOOLS

ML Python
Libraries



Python
Environment



Viz' Python Libraries



Framework
& Tool



docker



Google Cloud Platform



git

Suite of cloud computing services with more than 90 products.

- Power computation , Database, AI, Networking, Security *etc..*

Suite of cloud computing services with more than 90 products.

- **Power computation**, Database, AI, Networking, Security etc..

Suite of cloud computing services with more than 90 products.

- **Power computation**, Database, AI, Networking, Security etc..
 - \implies Google Cloud Engine

Suite of cloud computing services with more than 90 products.

- **Power computation**, Database, AI, Networking, Security etc..
 - \implies Google Cloud Engine

Why not IA tools?

- *AutoML* or *ML Engine* are tools that provide solution for non-expert.

Suite of cloud computing services with more than 90 products.

- **Power computation**, Database, AI, Networking, Security etc..
 - \implies Google Cloud Engine

Why not IA tools?

- *AutoML* or *ML Engine* are tools that provide solution for non-expert.

What do I need to know to use Google CCloud Engine?

- Basic knowledge of Google cloud interface.
- Basic use of terminal command (no graphic interface).
- *gcloud SDK*.

<https://cloud.google.com/sdk/docs/quickstart-debian-ubuntu>

Command line tool which allow to manage VM instance.

- *gcloud init*. To be used at first utilization.
- *gcloud compute instances start/stop/delete instance_name*. To start/stop/remove instance.
- *gcloud compute scp --recurse CopyFrom CopyTo*
 - *--recurse* (optional) : To be used if directory is copied.
 - *CopyFrom* : Location of the file or directory to be copied.
 - *CopyTo* : Location of the directory where the file or directory will be copied.
 - *Syntax* : `[[[USER@]INSTANCE :]DIR]` . To send file on the instance.

- *gcloud compute ssh --ssh-key-file LocationOfSSHKey --zone europe-west1-b*. To set ssh-connection to the instance.
- *gcloud compute ssh --command 'COMMAND'*. To execute command on the instance.
 - *gcloud compute ssh --command 'mkdir data'*
 - *gcloud compute ssh --command 'python learning.py'*

- Python
- Cuda
- Docker
- Nvidia-docker

bash_script for installation in *utils/bash_utils_on_gpu*.

PIPELINE D'EXECUTION

1. Write script in your *local* machine.
2. Turn you *instance on*.
3. Build environment(if first used).
4. Send latest version of your code to the instance.
5. Send data to the instance.
7. Run the script on the instance.
8. Copy the results you want to analyze from the instance to your local machine.
10. Turn your instance off.

1. Write script in your *local* machine.
2. Turn you *instance on*.
3. Build environment(if first used).
4. Send latest version of your code to the instance.
5. Send data to the instance.
6. Run container.
7. Run the script on the instance.
8. Copy the results you want to analyze from the instance to your local machine.
9. Stop and remove container.
10. Turn your instance off.

PIPELINE - COMMAND

1. Write script in your *local* machine.
2. `gcloud compute instances start ..`
3. `gcloud compute ssh --command 'mkdir data'`
4. `gcloud compute scp script.py bguillou@instance-gpu :/home/`
5. `gcloud compute scp --recurse data bguillou@instance-gpu :/home/`
6. `gcloud compute scp --recurse data bguillou@instance-gpu :/home/`
7. `gcloud compute ssh --command 'python script.py -a 3'`
8. `gcloud compute scp --recurse bguillou@instance-gpu :/home/results/
/home/`
9. `gcloud compute instances start ..`
10. `gcloud compute instances stop ..`

PIPELINE - COMMAND

1. Write script in your *local* machine.
2. `gcloud compute instances start ..`
3. `gcloud compute ssh --command 'mkdir data'`
4. `gcloud compute scp script.py bguillou@instance-gpu :/home/`
5. `gcloud compute scp --recurse data bguillou@instance-gpu :/home/`
6. `gcloud compute ssh --command 'sudo nvidia-docker run ...'`
7. `gcloud compute ssh --command 'python script.py -a 3'`
8. `gcloud compute scp --recurse bguillou@instance-gpu :/home/results/
/home/`
9. `gcloud compute ssh --command 'sudo nvidia-docker container stop ...'`
10. `gcloud compute instances stop ..`

PIPELINE - COMMAND

1. Write script in your *local* machine.
2. `gcloud compute instances start ..`
3. `gcloud compute ssh --command 'mkdir data'`
4. `gcloud compute scp script.py bguillou@instance-gpu :/home/`
5. `gcloud compute scp --recurse data bguillou@instance-gpu :/home/`
6. `gcloud compute ssh --command 'sudo nvidia-docker run ...'`
7. `gcloud compute ssh --command 'python script.py -a 3'`
8. `gcloud compute scp --recurse bguillou@instance-gpu :/home/results/
/home/`
9. `gcloud compute ssh --command 'sudo nvidia-docker container stop ...'`
10. `gcloud compute instances stop ..`

Too many possibility to make a mistake.

PIPELINE - COMMAND

1. Write script in your *local* machine.
2. `gcloud compute instances start ..`
3. `gcloud compute ssh --command 'mkdir data'`
4. `gcloud compute scp script.py bguillou@instance-gpu :/home/`
5. `gcloud compute scp --recurse data bguillou@instance-gpu :/home/`
6. `gcloud compute ssh --command 'sudo nvidia-docker run ...'`
7. `gcloud compute ssh --command 'python script.py -a 3'`
8. `gcloud compute scp --recurse bguillou@instance-gpu :/home/results/
/home/`
9. `gcloud compute ssh --command 'sudo nvidia-docker container stop ...'`
10. `gcloud compute instances stop ..`

Too many possibility to make a mistake.

⇒ **Pipeline**

We will use a tool composed of 3 *python script* and a *.yaml* file :

- **conf.yaml** : This file works like a dictionary which contains global variables such as location of directory.
- **instances.py** : This script defines a Python class *InstanceManager* which encapsulates calls to *gcloud*. For example *list()* function of this class calls this command in terminal :

```
gcloud compute instances list
```

- **project.py** This script defines a Python class *ProjectManager* which contains specific functions to manage your project.
For example, the *update_data(self, zip_file)* function allows to send a *zip_file* from your *data* directory from your local machine to the *data* directory on your instance.
- **main.py** This is a python script which contains all the command to execute your pipeline.

DOCKER

IA FRAMEWORKS - TOOLS

ML Python
Libraries



Python
Environment



Viz' Python Libraries



Framework
& Tool



docker



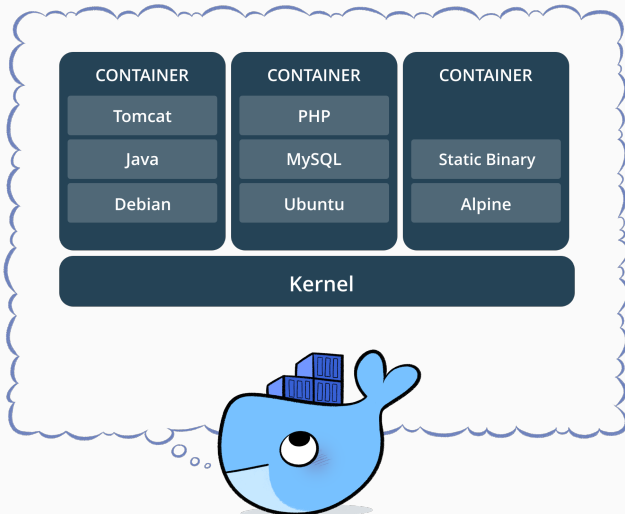
Google Cloud Platform



git

WHAT IS DOCKER?

Docker enables to build Virtual machines which are easy to re-create on different compute environment.



WHY SHOULD I USE DOCKER AS A DATA SCIENTIST?

- **Reproducibility** : wrapping all you environment in a Docker container ensure the possibility to recreate your environment and makes your work more accessible.
- **Portability of your compute environment** To move your code and your model easily on machine with more computational power.
- **Enlarge your possibility** : being comfortable with Docker can allow you to use various solution available with docker.

- **Image** : Its like a turned-off VM which contains the tools you want. Ex : Ubuntu + TensorFlow with Nvidia Drivers and a running Jupyter Server.
- **Container** : Is an instantiation of an image. You can have multiple copies of the same image running.
- **Dockerfile** : Recipe for creating an Image.
- **DockerHub / Image Registry** : Place where people/organization can post public (or private) docker images to facilitate collaboration and sharing.

<https://hub.docker.com/>

WHAT WE'LL DO IN THIS TP

- Write a *Dockerfile* which is based on the official *Tensorflow Dockerfile* available on *DockerHub*.
- Use the *Dockerfile* to build a *image*.
- Launch *container* with different option from the build *image*.

Dockerfile of the image we'll build :

```
FROM tensorflow/tensorflow:latest-devel-gpu-py3
RUN apt-get update && apt-get install -y
python-opencv python-tk vim
RUN pip install h5py keras pytest scikit-image
seaborn tqdm gensim
```

- **FROM** : Specifies the base image you want to build on top of. Docker will look in your local environment for the image you called and if it cannot find it locally it will search it in *DockerHub*.
- **RUN** : Is followed by normal commands that would be directly run on terminal to install librairies or framework.

Run the BUILD command in order to build your **IMAGE**.

```
docker build -t ImageName -f /Docker/Dockerfile /Docker/
```

Run the BUILD command in order to build your **IMAGE**.

Name of
the image


docker build -t ImageName -f /Docker/Dockerfile /Docker/

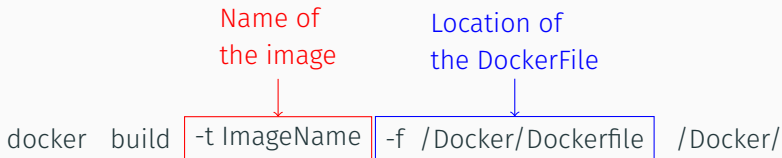
BUILD IMAGE

Run the BUILD command in order to build your **IMAGE**.

docker build -t ImageName -f /Docker/Dockerfile /Docker/

Name of
the image

Location of
the DockerFile



BUILD IMAGE

Run the BUILD command in order to build your **IMAGE**.

Diagram illustrating the components of the `docker build` command:

```
docker build -t ImageName -f /Docker/Dockerfile /Docker/
```

The components are labeled as follows:

- Name of the image** (red text) points to `-t ImageName` (red box).
- Location of the DockerFile** (blue text) points to `-f /Docker/Dockerfile` (blue box).
- Location of the build context** (orange text) points to `/Docker/` (orange box).

The build context is the location of the folder to which the **ADD** statement will reference. This means that all external files required by the *Dockerfile* will be located here.

RUN CONTAINER


Run the RUN command in order to run your **CONTAINER**.

```
docker run -it --gpus all --name ContainerName  
-v ~/CatsVsDogs/ :/root/CatsVsDogs ImageName
```

RUN CONTAINER

Run the RUN command in order to run your **CONTAINER**.

Mode in which
the container
will be launched

```
docker run  -it -gpu all -name ContainerName  
-v ~/CatsVsDogs/ :/root/CatsVsDogs ImageName
```

Mode of the container could be :


- -it : interactive mode,
- -dt : detached mode.

RUN CONTAINER

Run the RUN command in order to run your **CONTAINER**.

Mode in which
the container
will be launched

gpus
option



```
docker run -it -gpus all -name ContainerName  
-v ~/CatsVsDogs/ :/root/CatsVsDogs ImageName
```

Mode of the container could be :

- -it : interactive mode,
- -dt : detached mode.

RUN CONTAINER

Run the RUN command in order to run your **CONTAINER**.

Mode in which
the container
will be launched

gpus
option

Name of
the container

```
docker run -it -gpus all -name ContainerName  
-v ~/CatsVsDogs/ :/root/CatsVsDogs ImageName
```

Mode of the container could be :

- -it : interactive mode,
- -dt : detached mode.

RUN CONTAINER

Run the RUN command in order to run your **CONTAINER**.

Mode in which
the container
will be launched

gpus
option

Name of
the container

```
docker run -it -gpus all -name ContainerName  
-v ~/CatsVsDogs/ :/root/CatsVsDogs ImageName
```

Name of
the image

A diagram showing the command 'docker run -it -gpus all -name ContainerName -v ~/CatsVsDogs/ :/root/CatsVsDogs ImageName'. The options are grouped into boxes: '-it' in a red box, '-gpus all' in a green box, and '-name ContainerName' in a blue box. Arrows point from descriptive text to these boxes: 'Mode in which the container will be launched' (red) to the red box, 'gpus option' (green) to the green box, and 'Name of the container' (blue) to the blue box. The volume mount '-v ~/CatsVsDogs/ :/root/CatsVsDogs' is followed by 'ImageName' in a red box, with an arrow pointing from 'Name of the image' (red) to it.

Mode of the container could be :

- -it : interactive mode,
- -dt : detached mode.

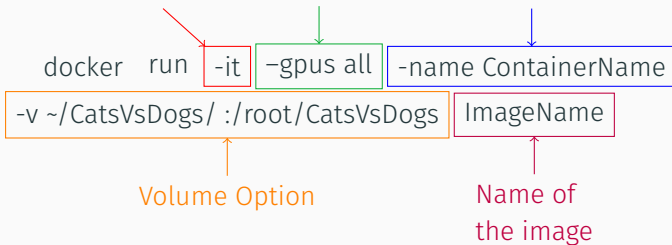
RUN CONTAINER

Run the RUN command in order to run your **CONTAINER**.

Mode in which
the container
will be launched

gpus
option

Name of
the container



Mode of the container could be :

- `-it` : interactive mode,
- `-dt` : detached mode.

The `-v` option allow you to use some data you have in your machine within a container.

```
-v ~/CatsVsDogs/ :/root/CatsVsDogs
```


The `-v` option allow you to use some data you have in your machine within a container.

Declare
Option



`-v` `~/CatsVsDogs/ :/root/CatsVsDogs`

MOUNTED VOLUME

The `-v` option allow you to use some data you have in your machine within a container.

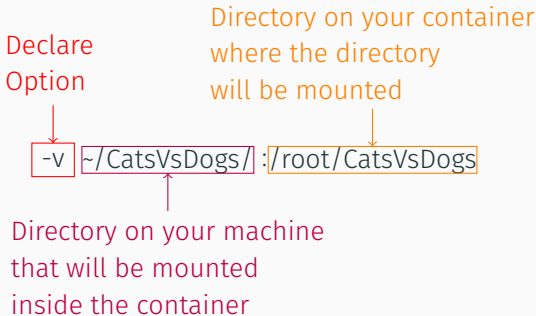
Declare
Option

`-v` `~/CatsVsDogs/` `:/root/CatsVsDogs`

Directory on your machine
that will be mounted
inside the container

MOUNTED VOLUME

The `-v` option allow you to use some data you have in your machine within a container.



- `sudo docker image ls -a`
- `sudo docker container ls -a`
- `sudo docker start/stop/rm container_name -a`
- `sudo docker exec container_name 'Command to execute in container' -a`

- Use Jupyter on the instance (via ssh connection)
- Google Colab
- Image Gcloud

<https://towardsdatascience.com/how-docker-can-help-you-become-a-more-effective-data-scientist-7fc048ef91d5>

RÉFÉRENCES
