Politecnico di Torino

1859

VANDAL AI

PyTorch Bootcamp

Machine Learning and Deep Learning

A. A. 2024/25

Teaching Assistants: *Davide Buoso, Arda Eren Dogru*

Slide credits: *Chiara Plizzari, Gabriele Rosi*

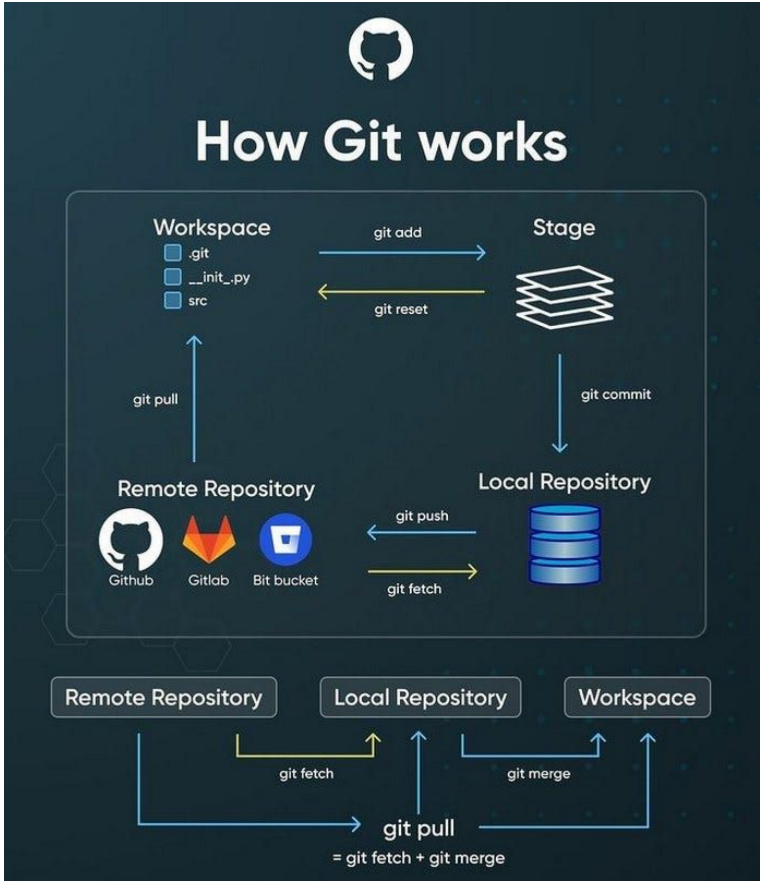
How to setup a project from scratch

Lesson 3

Github

- Cloud-based platform
- Born for collaboration and version control

Repositories are cloud-saved folders
saving remotely all your files



clone -> add -> commit -> push
pull=sync

Github

Main commands:

- **Clone**: copies a repository to your machine (execute the first time)
- **Status**: tells us about the status of our repo
- **Add**: adds the local changes to the “to send remotely” files
- **Commit**: prepares “the package” of changes to send (and let you add a message, representing what you did)
- **Push**: Really pushes everything to your remote repo

- **Pull**: sync your local changes with the changes happened in the remote repo (useful to bring in your local folder the changes a team member pushed)

https://docs.github.com/en/get-started

Github

clone -> change : add, commit -m
push

Typical Workflow

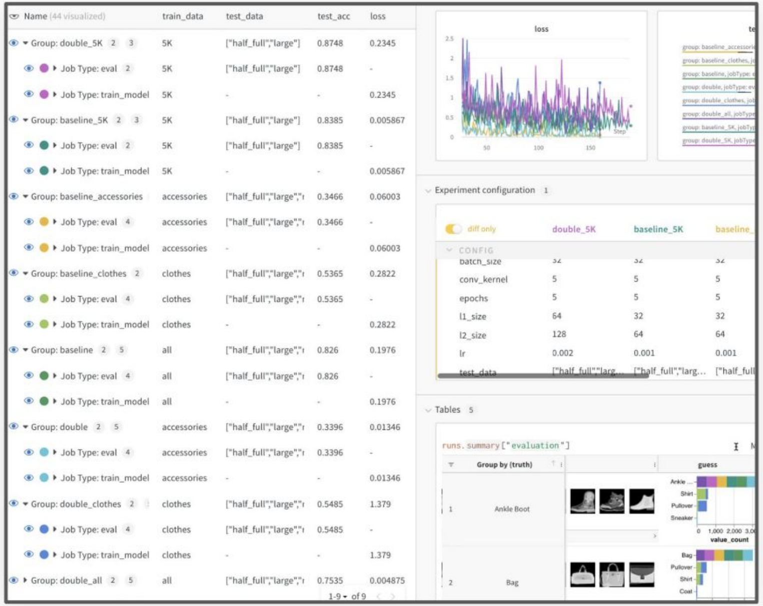
```
git clone <repo>


For every big change you make:

     … local changes …

     git add .

     git commit -m 'Explain message'


     git push
```

How to keep track of your results?

Wandb: record all your results

Get started in 60 seconds

```
!pip install wandb    # Install W&B

wandb.init()          # Start experiment
wandb.log(metrics)    # Log metrics + more!
```



Name (44 visualized)	train_data	test_data	test_acc	loss
Group: double_5K 2 3	5K	["half_full","large"]	0.8748	0.2345
● Job Type: eval 2	5K	["half_full","large"]	0.8748	-
● Job Type: train_model	5K	-	-	0.2345
Group: baseline_5K 2 3	5K	["half_full","large"]	0.8385	0.005867
● Job Type: eval 2	5K	["half_full","large"]	0.8385	-
● Job Type: train_model	5K	-	-	0.005867
Group: baseline_accessories	accessories	["half_full","large","r	0.3466	0.06003
● Job Type: eval 4	accessories	["half_full","large","r	0.3466	-
● Job Type: train_model	accessories	-	-	0.06003
Group: baseline_clothes 2	clothes	["half_full","large","r	0.5365	0.2822
● Job Type: eval 4	clothes	["half_full","large","r	0.5365	-
● Job Type: train_model	clothes	-	-	0.2822
Group: baseline 2 5	all	["half_full","large","r	0.826	0.1976
● Job Type: eval 4	all	["half_full","large","r	0.826	-
● Job Type: train_model	all	-	-	0.1976
Group: double 2 5	accessories	["half_full","large","r	0.3396	0.01346
● Job Type: eval 4	accessories	["half_full","large","r	0.3396	-
● Job Type: train_model	accessories	-	-	0.01346
Group: double_clothes 2	clothes	["half_full","large","r	0.5485	1.379
● Job Type: eval 4	clothes	["half_full","large","r	0.5485	-
● Job Type: train_model	clothes	-	-	1.379
Group: double_all 2 5	all	["half_full","large","r	0.7535	0.004875

1-9 of 9

loss
2.5
2
1.5
1
0.5
0
50
100
150
Step
te
group: baseline_accessori
group: baseline_clothes, jo
group: baseline, jobType: e
group: double, jobType: ev
group: double_clothes, job
group: double_all, jobType
group: baseline_5K, jobTyp
group: double_5K, jobType

Experiment configuration 1

diff only	double_5K	baseline_5K	baseline_
CONFIG			
batch_size	32	32	32
conv_kernel	5	5	5
epochs	5	5	5
l1_size	64	32	32
l2_size	128	64	64
lr	0.002	0.001	0.001
test_data	["half_full","larg...	["half_full","larg...	["half_full

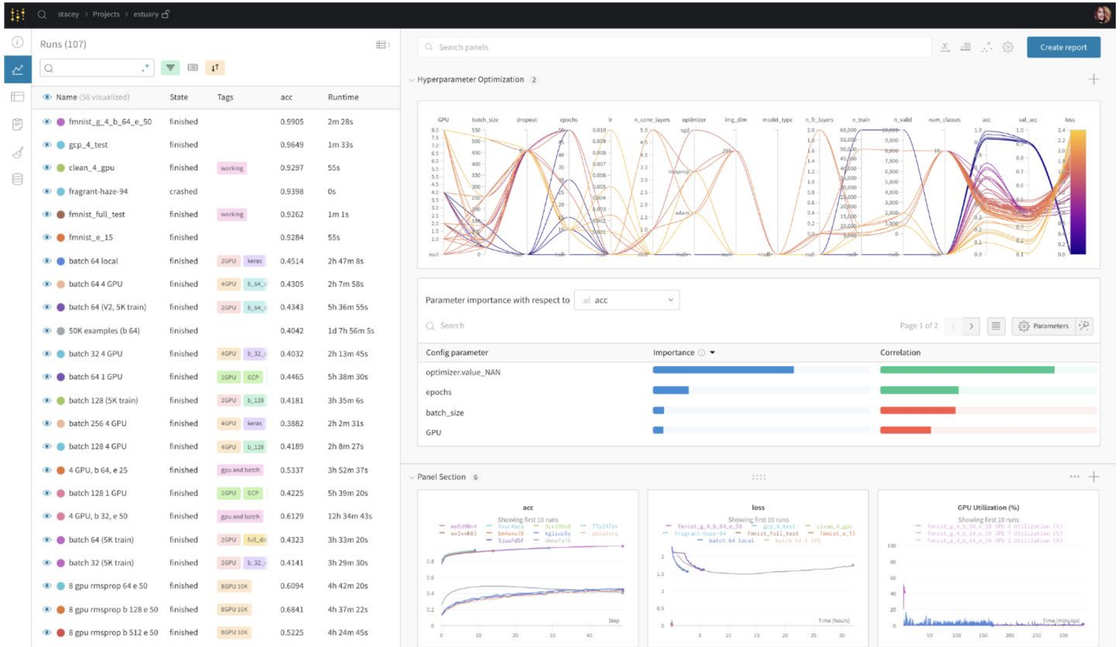
Tables 5

```
runs.summary["evaluation"]
```

	Group by (truth)		guess
1	Ankle Boot		
2	Bag		

Never lose your progress

Save everything you need to debug, compare and reproduce your models — architecture, hyperparameters, weights, model predictions, GPU usage, git commits, and even datasets — with a few lines of code.



Runs (107)

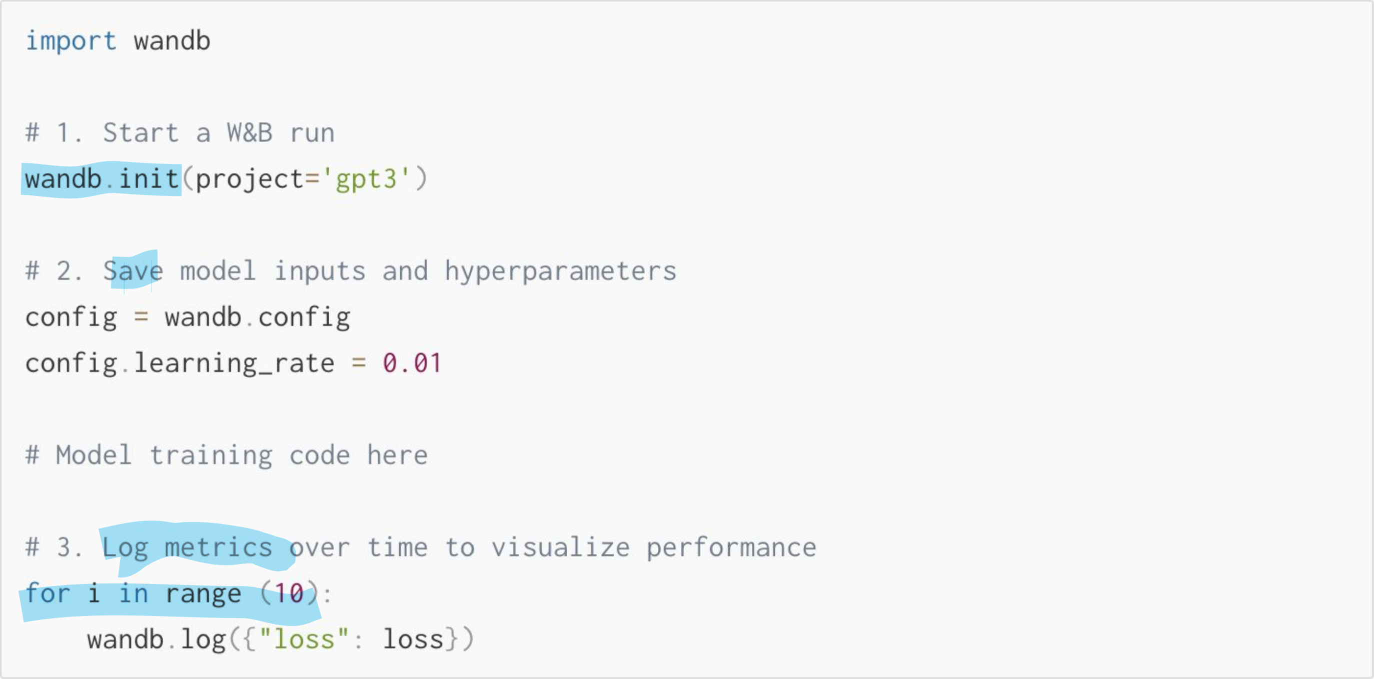
Name (56 visualized)	State	Tags	acc	Runtime
fmnist_g_4_b_64_e_50	finished		0.9905	2m 28s
gcp_4_test	finished		0.9649	1m 33s
clean_4_gpu	finished	working	0.9297	55s
fragrant-haze-94	crashed		0.9398	0s
fmnist_full_test	finished	working	0.9262	1m 1s
fmnist_e_15	finished		0.9284	55s
batch 64 local	finished	2GPU keras	0.4514	2h 47m 8s
batch 64 4 GPU	finished	4GPU b_64_c	0.4305	2h 7m 58s
batch 64 (V2, 5K train)	finished	2GPU b_64_c	0.4343	5h 36m 55s
50K examples (b 64)	finished		0.4042	1d 7h 56m 5s
batch 32 4 GPU	finished	4GPU b_32_	0.4032	2h 13m 45s
batch 64 1 GPU	finished	1GPU GCP	0.4465	5h 38m 30s
batch 128 (5K train)	finished	2GPU b_128	0.4181	3h 35m 6s
batch 256 4 GPU	finished	4GPU keras	0.3882	2h 2m 31s
batch 128 4 GPU	finished	4GPU b_128	0.4189	2h 8m 27s
4 GPU, b 64, e 25	finished	gpu and batch	0.5337	3h 52m 37s
batch 128 1 GPU	finished	1GPU GCP	0.4225	5h 39m 20s
4 GPU, b 32, e 50	finished	gpu and batch	0.6129	12h 34m 43s
batch 64 (5K train)	finished	2GPU full_da	0.4323	3h 33m 20s
batch 32 (5K train)	finished	2GPU b_32_c	0.4141	3h 29m 30s
8 gpu rmsprop 64 e 50	finished	8GPU 10K	0.6094	4h 42m 20s
8 gpu rmsprop b 128 e 50	finished	8GPU 10K	0.6841	4h 37m 22s
8 gpu rmsprop b 512 e 50	finished	8GPU 10K	0.5225	4h 24m 45s

Parameter importance with respect to acc

Search   Page 1 of 2   Parameters

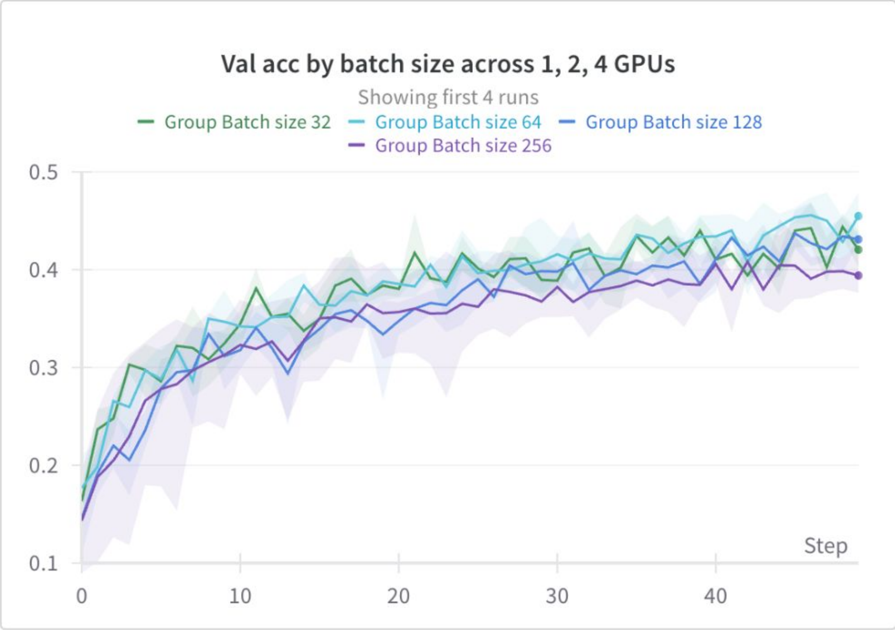
Config parameter	Importance	Correlation
optimizer.value_NAN		
epochs		
batch_size		
GPU		

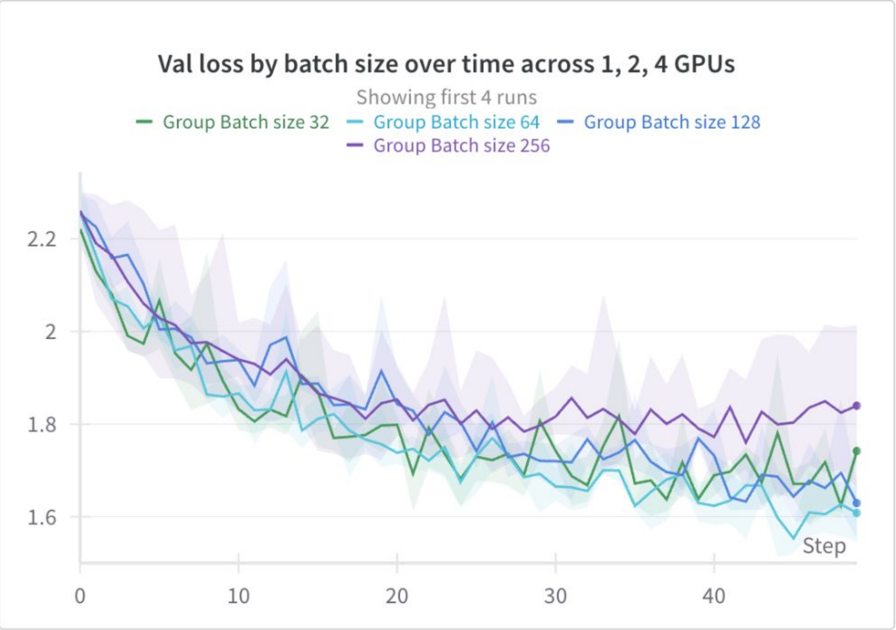
Track your model training



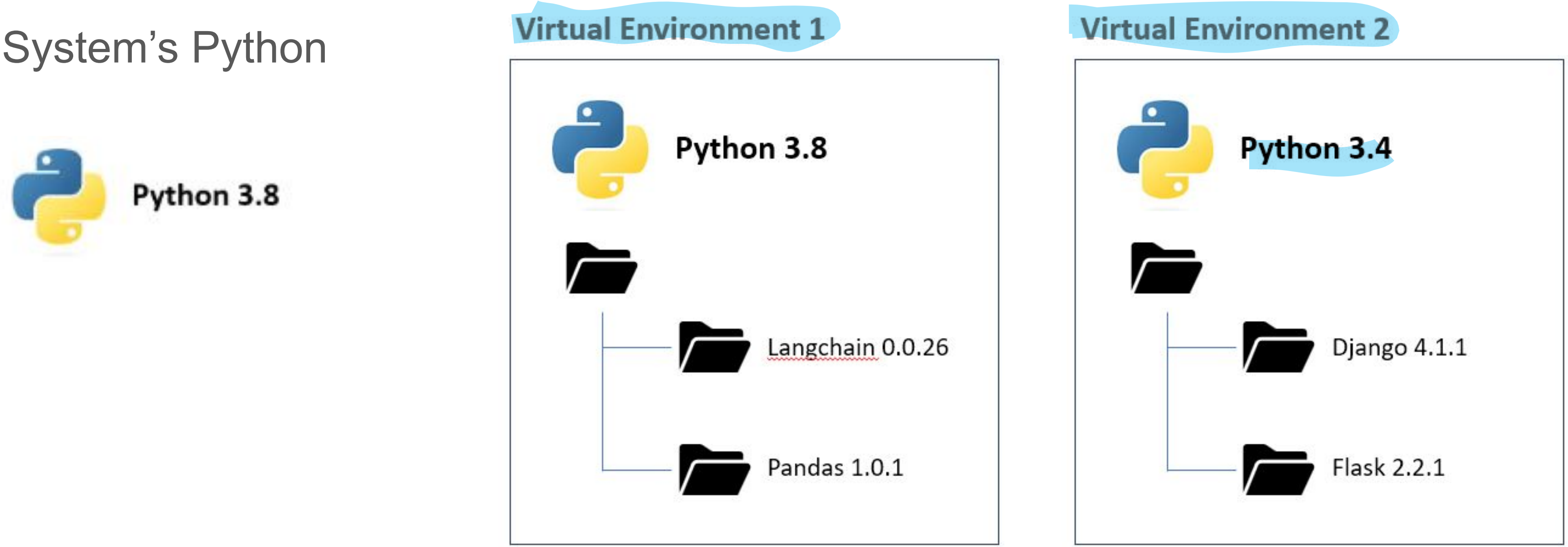
```
import wandb

# 1. Start a W&B run
wandb.init(project='gpt3')

# 2. Save model inputs and hyperparameters
config = wandb.config
config.learning_rate = 0.01

# Model training code here

# 3. Log metrics over time to visualize performance
for i in range (10):
    wandb.log({"loss": loss})
```

Track your model training





Virtual Envs and Pip Package Manager



System's Python

Python 3.8

Virtual Environment 1

Python 3.8

Langchain 0.0.26

Pandas 1.0.1

Virtual Environment 2

Python 3.4

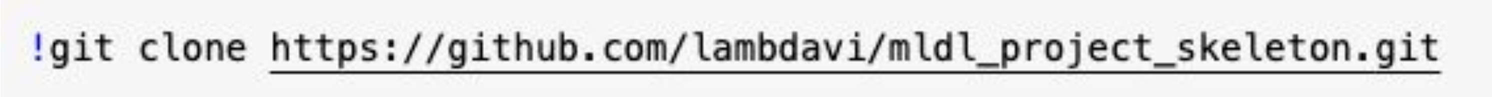
Django 4.1.1

Flask 2.2.1

https://packaging.python.org/en/latest/guides/installing-using-pip-and-virtual-environments/

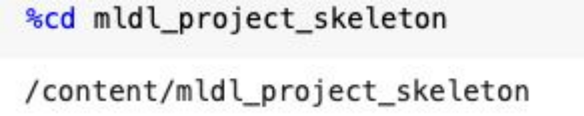
Colab special commands

! <command>: executes a bash command (e.g ls, cp, mv, git..)



```
!git clone https://github.com/lambdavi/mldl_project_skeleton.git
```

% <command>: executes a special command (e.g. cd)

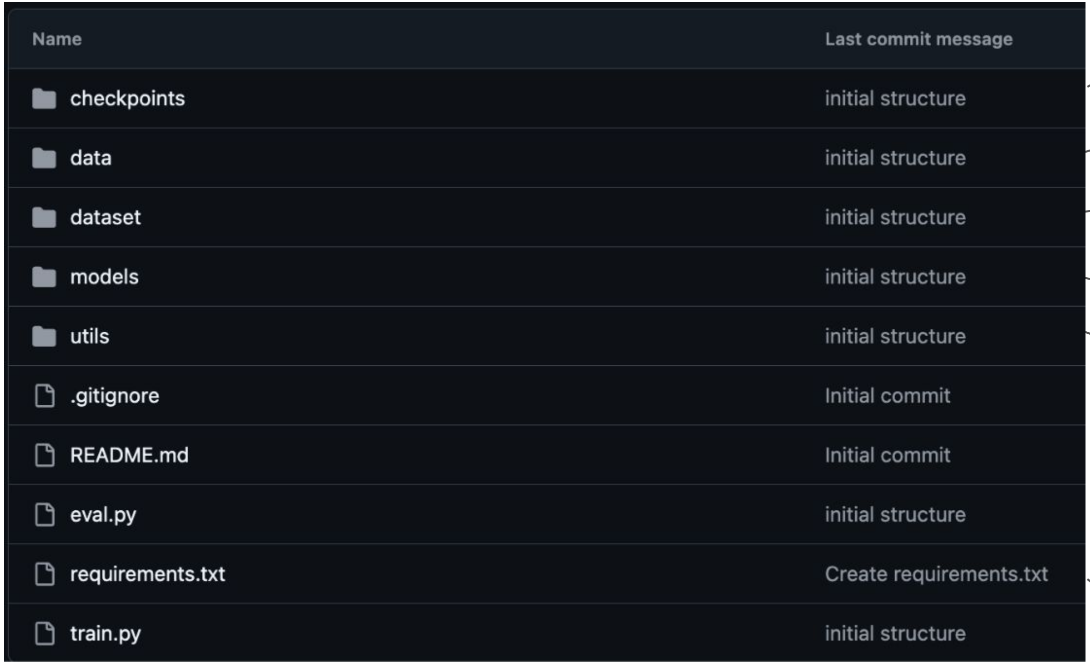


```
%cd mldl_project_skeleton
```

```
/content/mldl_project_skeleton
```

Goal of this lab:

Use Github, Wandb and Pip to setup a structured project.



Name	Last commit message
checkpoints	initial structure
data	initial structure
dataset	initial structure
models	initial structure
utils	initial structure
.gitignore	Initial commit
README.md	Initial commit
eval.py	initial structure
requirements.txt	Create requirements.txt
train.py	initial structure

Here you will save your saved models

Here you will save your dataloader/datase classes

Here you will save your actual dataset(s)

Here you will save your architectures (Pytorch)

Here you will save utils that support your code (e.g. scripts/visualization etc)

eval logic

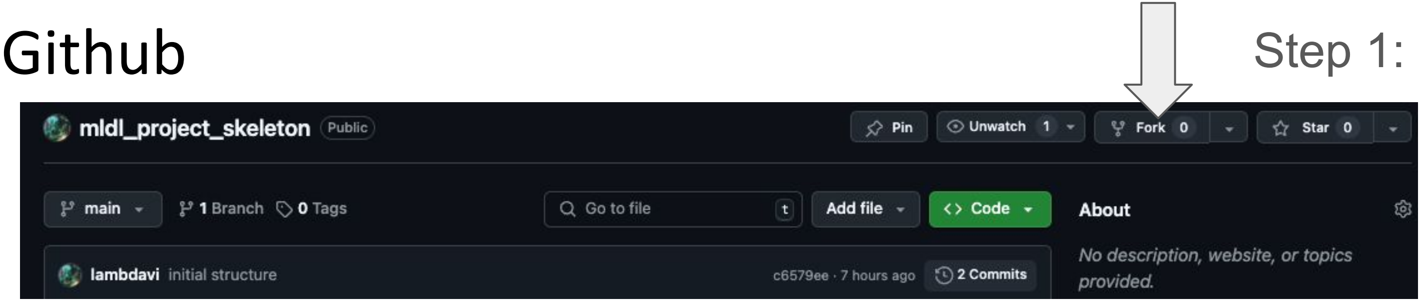
file containing the pip packages you installed (for reproducibility)

training logic

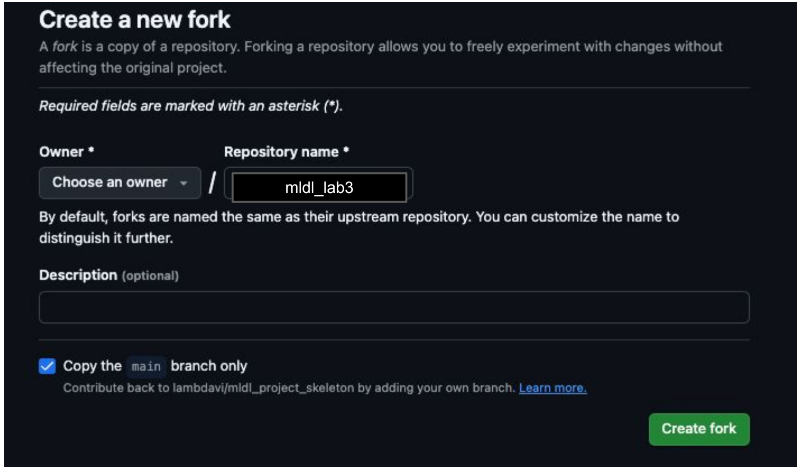
https://github.com/lambdavi/mldl_project_skeleton

Github

Step 1: Fork the project



No description, website, or topics provided.



Create a new fork

A *fork* is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project.

Required fields are marked with an asterisk (*).

Owner * : Choose an owner / Repository name * : mldl_lab3

By default, forks are named the same as their upstream repository. You can customize the name to distinguish it further.

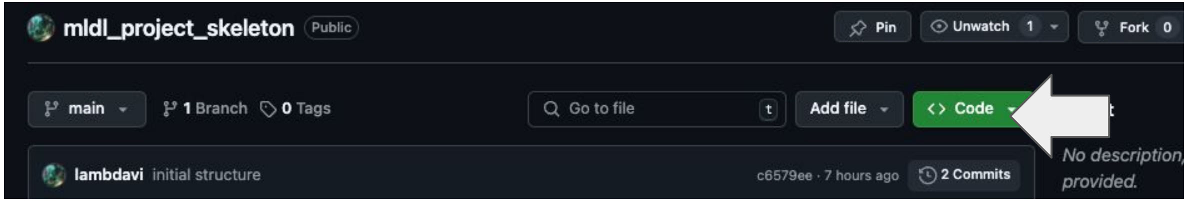
Description (optional)

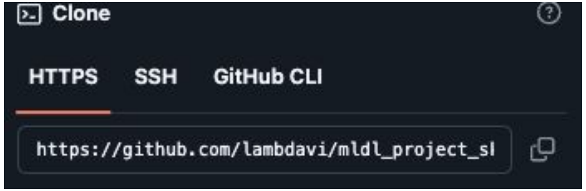
☑ Copy the `main` branch only

Contribute back to lambdavi/mldl_project_skeleton by adding your own branch. Learn more.

Step 2: name your repository (e.g. mldl_lab3)

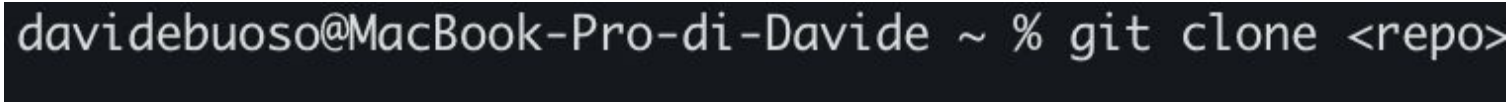
Step 3: Copy the link to the repo





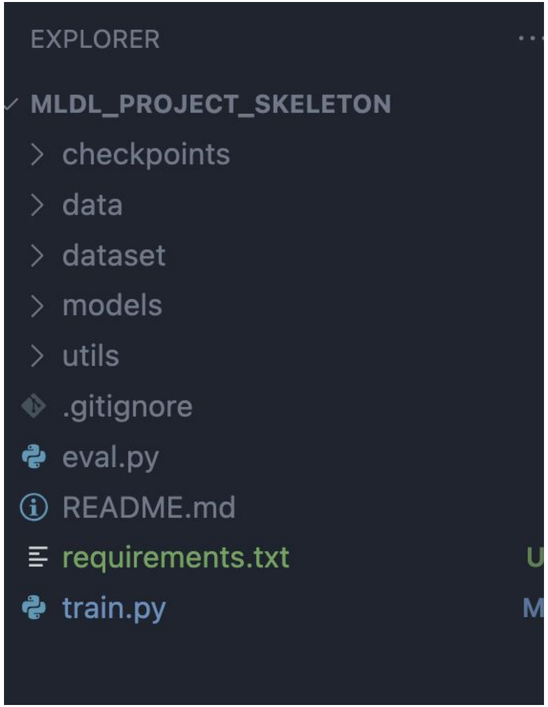
```
https://github.com/lambdavi/mldl_project_sl
```

Step 4: Use CMD/Terminal to clone the repository you forked (you may have to install git if it says “command not found”)



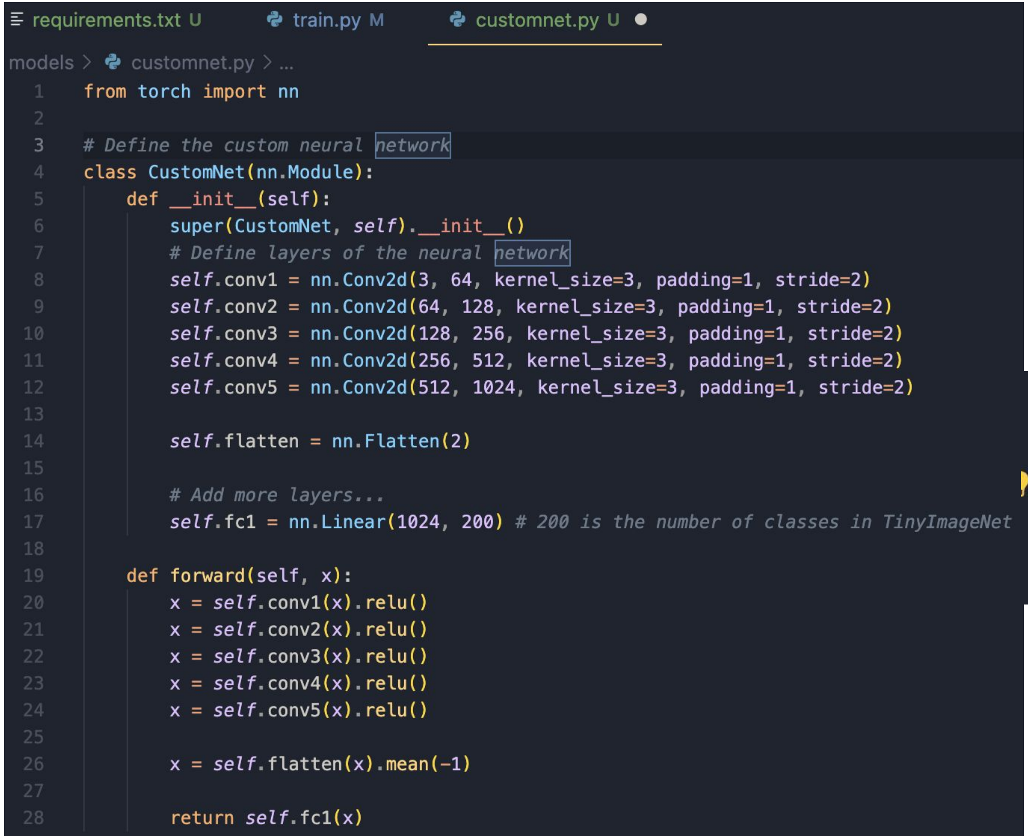
```
davidebuoso@MacBook-Pro-di-Davide ~ % git clone <repo>
```

Step 5: Open the folder on your favourite code editor (VSCode etc)

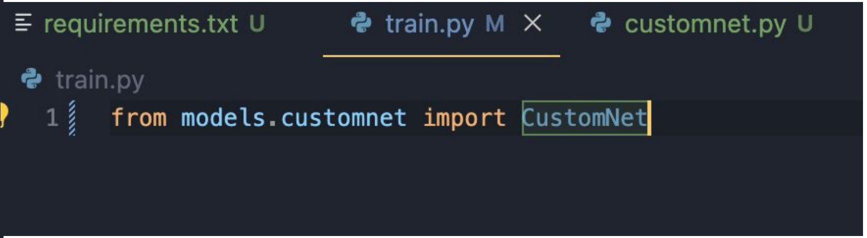


```
EXPLORER                          ···
MLDL_PROJECT_SKELETON
  > checkpoints
  > data
  > dataset
  > models
  > utils
  .gitignore
  eval.py
  README.md
  requirements.txt                U
  train.py                        M
```

Step 6: Make the changes to the files, add new files etc



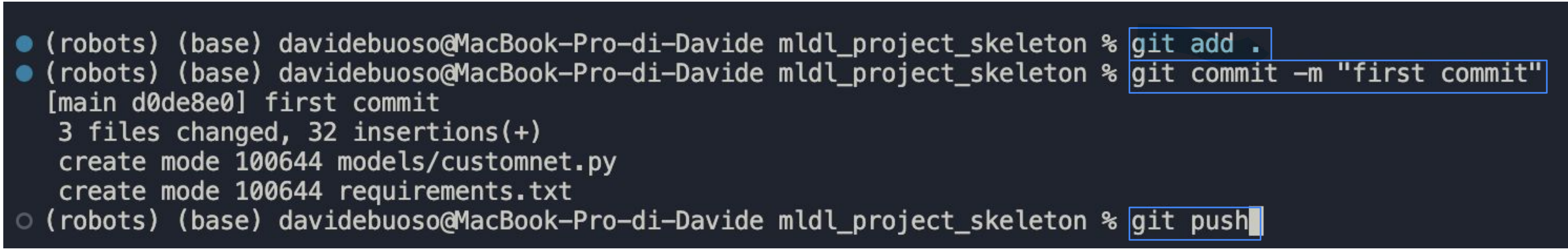
```python
from torch import nn

# Define the custom neural network
class CustomNet(nn.Module):
    def __init__(self):
        super(CustomNet, self).__init__()
        # Define layers of the neural network
        self.conv1 = nn.Conv2d(3, 64, kernel_size=3, padding=1, stride=2)
        self.conv2 = nn.Conv2d(64, 128, kernel_size=3, padding=1, stride=2)
        self.conv3 = nn.Conv2d(128, 256, kernel_size=3, padding=1, stride=2)
        self.conv4 = nn.Conv2d(256, 512, kernel_size=3, padding=1, stride=2)
        self.conv5 = nn.Conv2d(512, 1024, kernel_size=3, padding=1, stride=2)

        self.flatten = nn.Flatten(2)

        # Add more layers...
        self.fc1 = nn.Linear(1024, 200) # 200 is the number of classes in TinyImageNet

    def forward(self, x):
        x = self.conv1(x).relu()
        x = self.conv2(x).relu()
        x = self.conv3(x).relu()
        x = self.conv4(x).relu()
        x = self.conv5(x).relu()

        x = self.flatten(x).mean(-1)

        return self.fc1(x)
```



```python
from models.customnet import CustomNet
```

add -> commmit -m --> push

Step 7: Use add, commit and push to put your files back on github



```
● (robots) (base) davidebuoso@MacBook-Pro-di-Davide mldl_project_skeleton % git add .
● (robots) (base) davidebuoso@MacBook-Pro-di-Davide mldl_project_skeleton % git commit -m "first commit"
  [main d0de8e0] first commit
   3 files changed, 32 insertions(+)
   create mode 100644 models/customnet.py
   create mode 100644 requirements.txt
○ (robots) (base) davidebuoso@MacBook-Pro-di-Davide mldl_project_skeleton % git push
```

NOTE: Do not push any dataset on Github!

Once you finished to setup your project locally and pushed everything on Github you should be able to see your new files in the repository!

Now you can open the Colab, clone your repo, download the datasets (on colab) and run your code.

Your turn!

It’s time to setup your project!

- At this link https://github.com/lambdavi/mldl_project_skeleton you can find the starting repo (TO FORK).
- Fork our repo and then clone your repo on your computer.
- Take your code from Lab1/Lab2 and populate the repo with Python files in the right directories.
- Add visualization logic.
- Test your code on Colab.

Notebook @ https://colab.research.google.com/drive/1sLHv8x3p_SOI2Y6wo5sf1XaId329HxH4?usp=sharing