Master in Computer Vision Barcelona

M5 Project: Cross-modal Retrieval

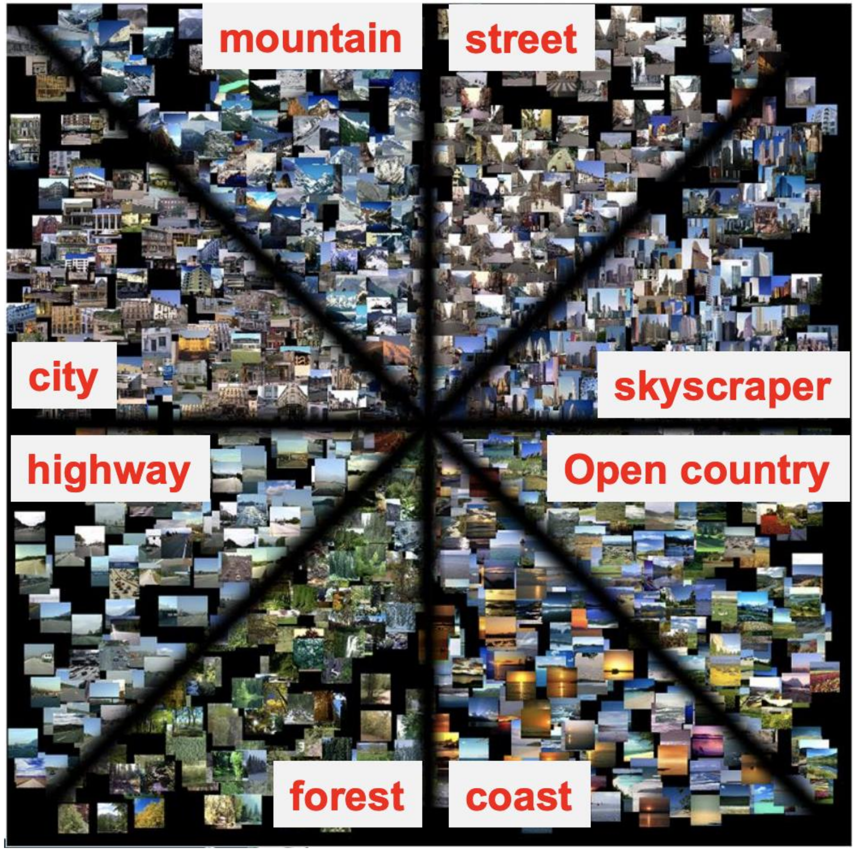
Week 1. Introduction to Pytorch

Rubèn Pérez Tito
rperez@cvc.uab.es

Ernest Valveny
ernest@cvc.uab.es

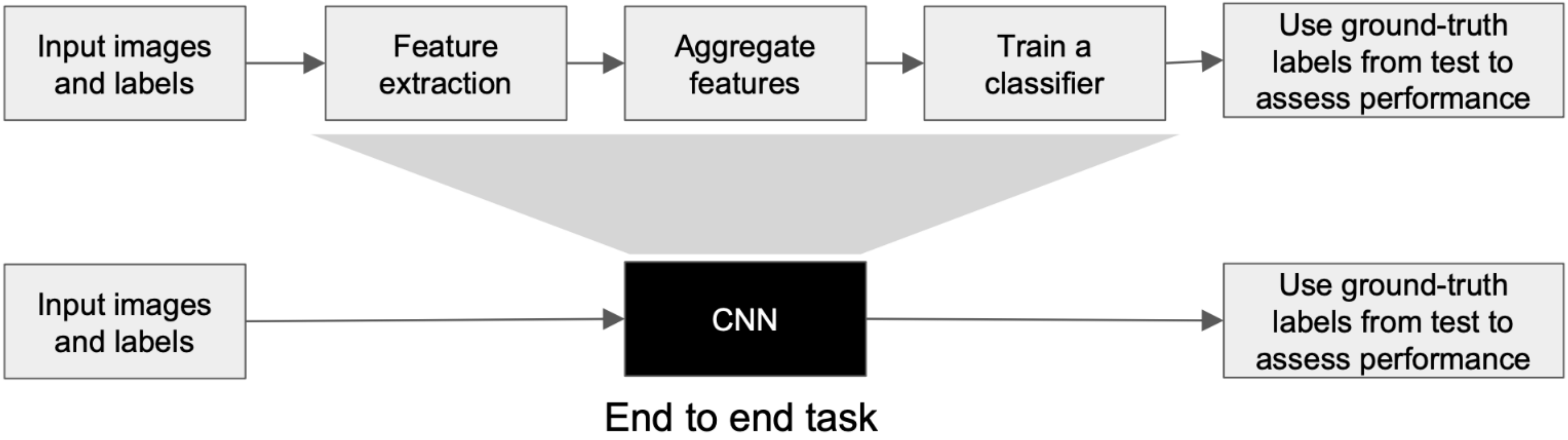
M3 GOAL REMINDER

The goal of M3 was to learn the techniques for category classification: handcrafted and learned features



M3 GOAL REMINDER

M3 tasks for last weeks was to train a CNN from scratch...

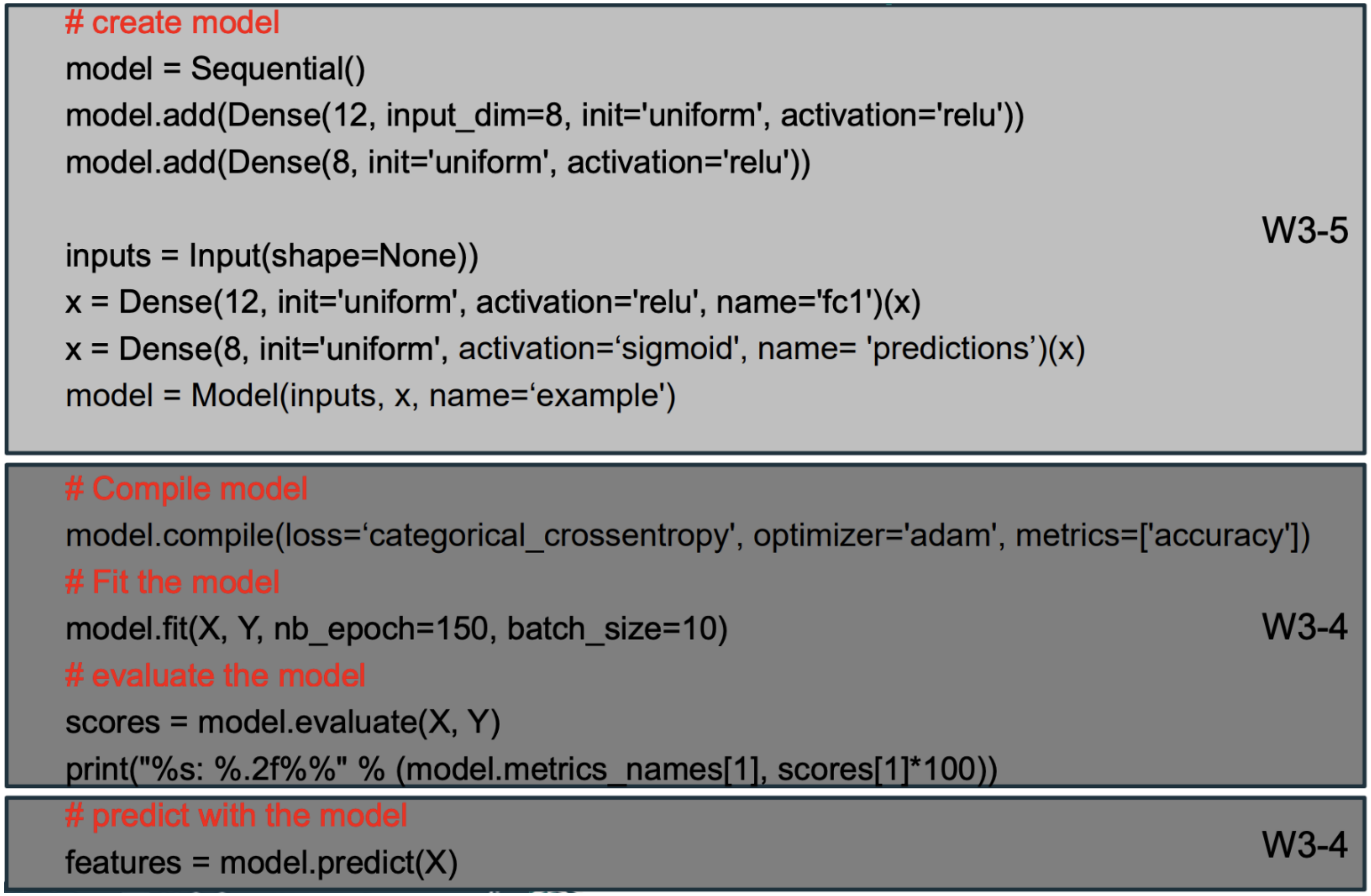


Machine learning for image classification:

- Data driven methods: Deep Convolutional Networks: 3 sessions
  - From hand-crafted to learnt features
  - Fine tuning of pre-trained CNNs
  - Training a CNN from scratch

M3 GOAL REMINDER

… and you did these tasks using Keras



```
# create model
model = Sequential()
model.add(Dense(12, input_dim=8, init='uniform', activation='relu'))
model.add(Dense(8, init='uniform', activation='relu'))

inputs = Input(shape=None))
x = Dense(12, init='uniform', activation='relu', name='fc1')(x)
x = Dense(8, init='uniform', activation='sigmoid', name= 'predictions')(x)
model = Model(inputs, x, name='example')
```

W3-5

```
# Compile model
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
# Fit the model
model.fit(X, Y, nb_epoch=150, batch_size=10)
# evaluate the model
scores = model.evaluate(X, Y)
print("%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
```

W3-4

```
# predict with the model
features = model.predict(X)
```

W3-4

M5 GOAL

Implementation of a model for **cross-modal retrieval**

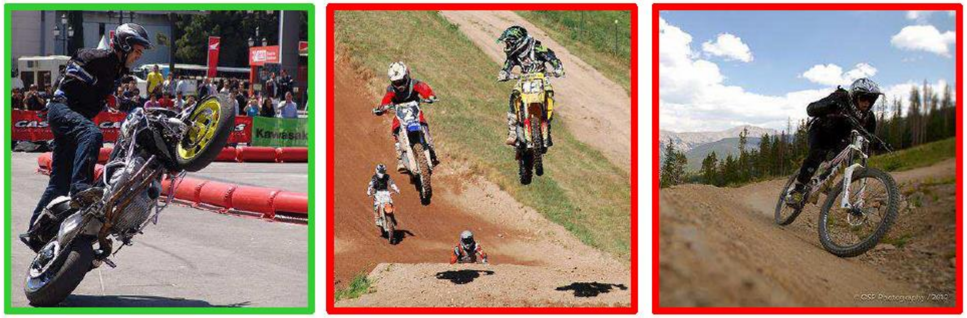
Image to Text

(c)

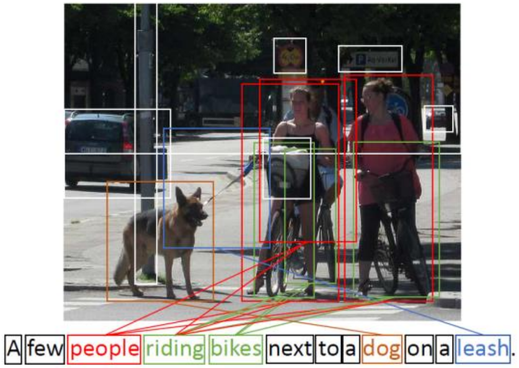
1:A female runner dressed in blue athletic wear is running in a competition , while spectators line the street . ✓
2:A lady dressed in blue running a marathon . ✓
3:A young woman is running a marathon in a light blue tank top and spandex shorts . ✓
4:A lady standing at a crosswalk . ✗
5:A woman who is running , with blue shorts . ✓

Text to Image

Query: *A man riding a motorcycle is performing a trick at a track .*



Sub-objective: **Object detection**



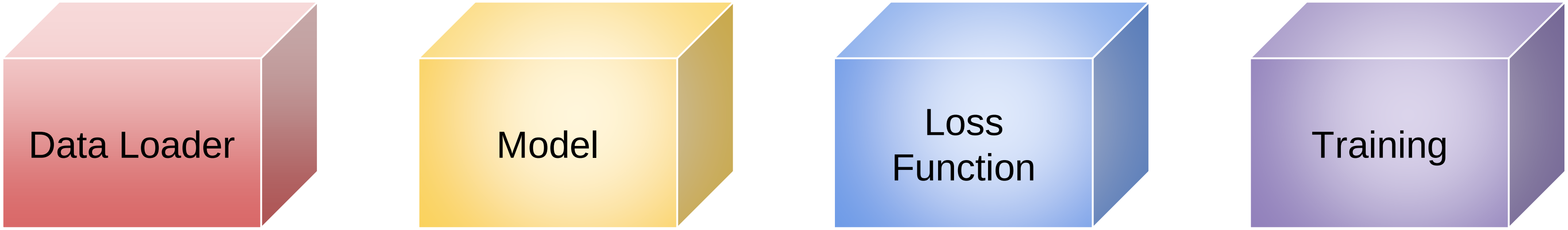
M5 Project Stages and Schedule

Week 1	**Introduction to Pytorch - Image Classification**
Week 2	**Object Detection, Recognition and Segmentation**
Week 3	
Week 4	**Image Retrieval**
Week 5	**Cross-modal Retrieval**
Week 6	**Presentation**

M5 INTRODUCTION TO PYTORCH

- In M5 Project, we will use Pytorch framework instead of Keras for object detection and segmentation.
- We will use Detectron2 framework from Facebook Artificial Intelligence Research (FAIR), which is a research platform for object detection and segmentation in Pytorch
  - https://github.com/facebookresearch/detectron2
  - More details about the project next week (W2)
- **First task**: Implementing the final model from M3 (Image Classification) in Pytorch

M5 INTRODUCTION TO PYTORCH



Data Loader

Model

Loss Function

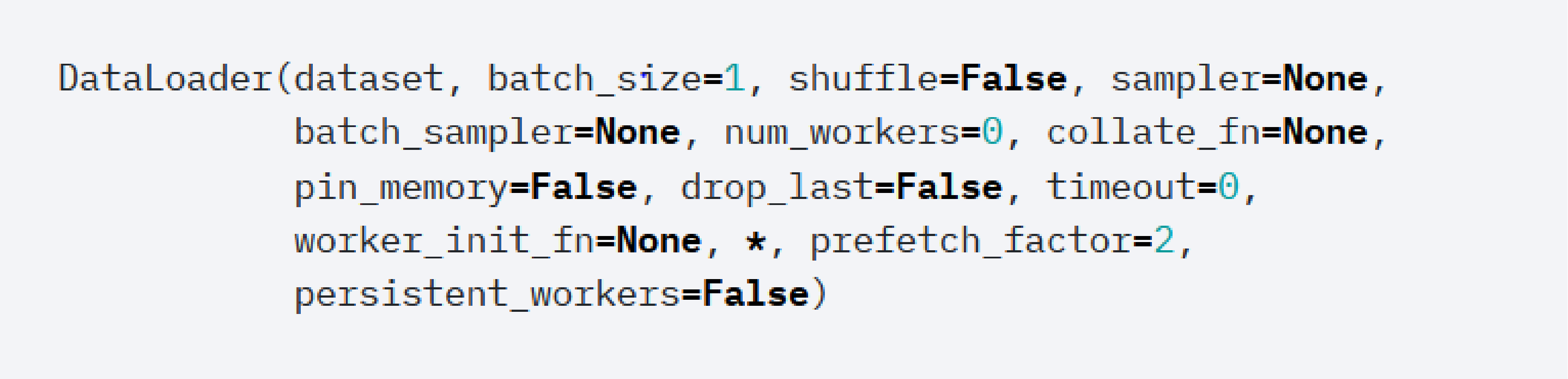
Training

M5 INTRODUCTION TO PYTORCH

- Few things to be considered in Pytorch:
  - **torch.utils.data.Dataset**
    - An abstract class representing a Dataset.
    - class MyDataset(Dataset):
      - def __init__(self):
      - def __len__(self):
      - def __getitem__(self, index)

M5 INTRODUCTION TO PYTORCH

- **Dataloader**



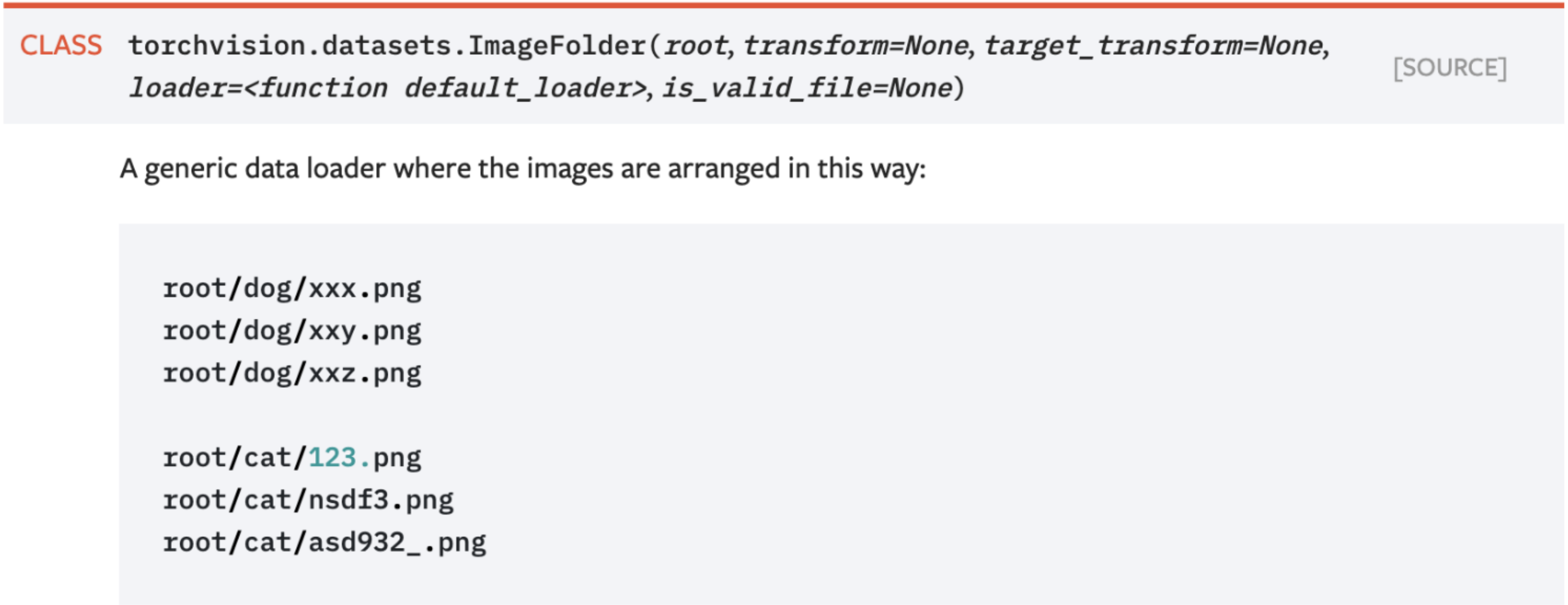
```
DataLoader(dataset, batch_size=1, shuffle=False, sampler=None,
           batch_sampler=None, num_workers=0, collate_fn=None,
           pin_memory=False, drop_last=False, timeout=0,
           worker_init_fn=None, *, prefetch_factor=2,
           persistent_workers=False)
```

M5 INTRODUCTION TO PYTORCH

- **Torchvision**: Package which consists of popular datasets, model architectures, and common image transformations for computer vision.
  - MODEL ZOO
    - AlexNet
    - VGG
    - ResNet
    - Inception v3 ….
  - Transforms
    - Many transforms function, CenterCrop, Normalize, RandomCrop, Flip, VerticalFlip, etc.
    - You can “**append**” them together with Compose.
      - torchvision.transforms.Compose(transforms.CenterCrop(10), transforms.Normalize((0.485, 0.456, 0.406), (0.229, 0.224, 0.225)))

M5 INTRODUCTION TO PYTORCH

- Few things to be considered in Pytorch:
    - **Dataloader - ImageFolder**



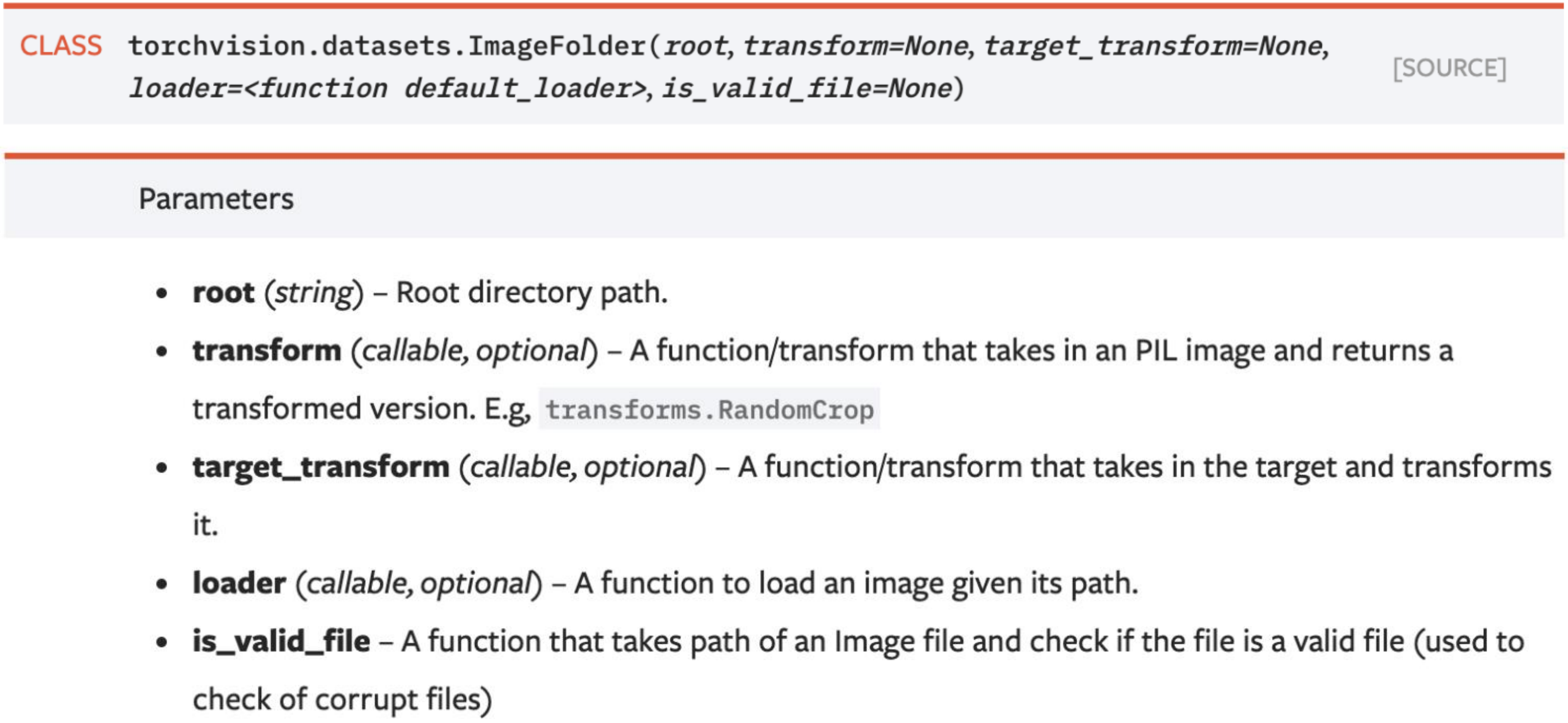
CLASS `torchvision.datasets.ImageFolder(`*`root`*`, `*`transform=None`*`, `*`target_transform=None`*`, `*`loader=<function default_loader>`*`, `*`is_valid_file=None`*`)` [SOURCE]

A generic data loader where the images are arranged in this way:

```
root/dog/xxx.png
root/dog/xxy.png
root/dog/xxz.png

root/cat/123.png
root/cat/nsdf3.png
root/cat/asd932_.png
```

M5 INTRODUCTION TO PYTORCH

- Few things to be considered in Pytorch:
  - **Dataloader - ImageFolder**

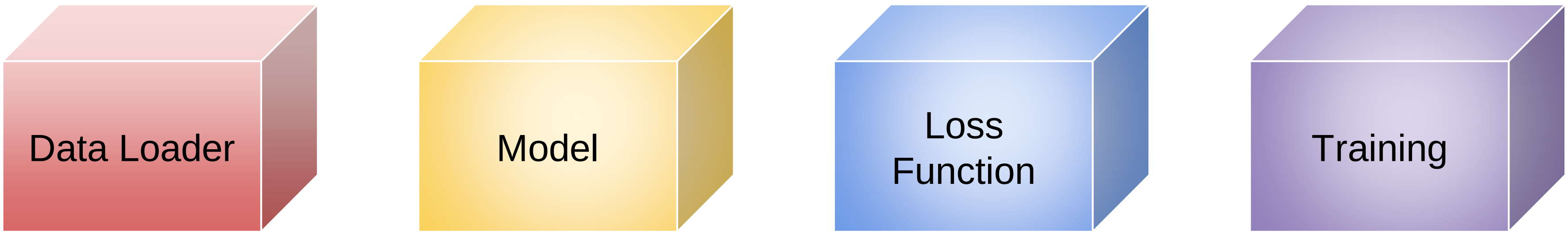


CLASS `torchvision.datasets.ImageFolder(`*`root`*`, `*`transform=None`*`, `*`target_transform=None`*`, `*`loader=<function default_loader>`*`, `*`is_valid_file=None`*`)` [SOURCE]

Parameters

- **root** (*string*) – Root directory path.
- **transform** (*callable, optional*) – A function/transform that takes in an PIL image and returns a transformed version. E.g, `transforms.RandomCrop`
- **target_transform** (*callable, optional*) – A function/transform that takes in the target and transforms it.
- **loader** (*callable, optional*) – A function to load an image given its path.
- **is_valid_file** – A function that takes path of an Image file and check if the file is a valid file (used to check of corrupt files)

M5 INTRODUCTION TO PYTORCH



Data Loader

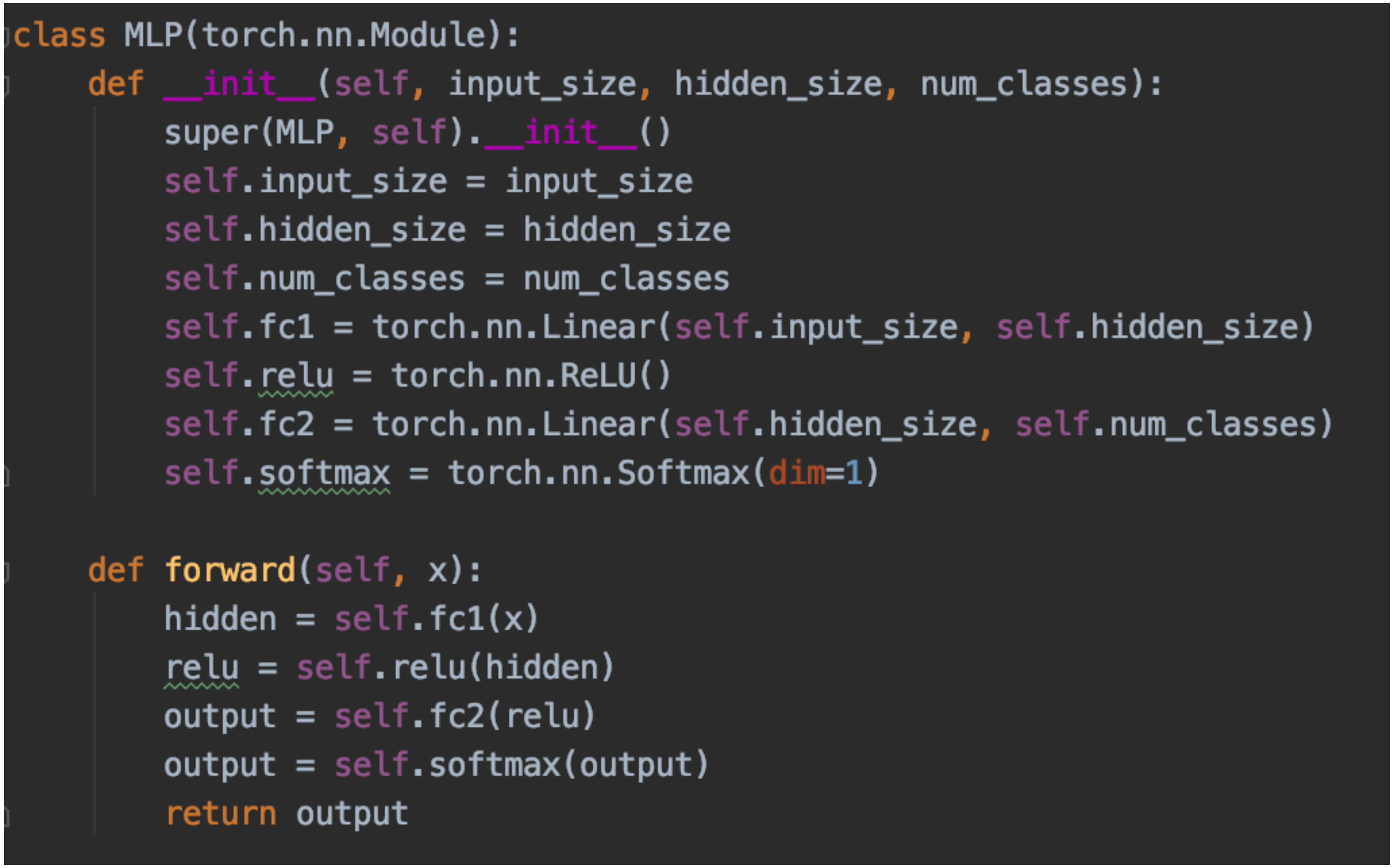
Model

Loss Function

Training

M5 INTRODUCTION TO PYTORCH

- **Model architecture**



```python
class MLP(torch.nn.Module):
    def __init__(self, input_size, hidden_size, num_classes):
        super(MLP, self).__init__()
        self.input_size = input_size
        self.hidden_size = hidden_size
        self.num_classes = num_classes
        self.fc1 = torch.nn.Linear(self.input_size, self.hidden_size)
        self.relu = torch.nn.ReLU()
        self.fc2 = torch.nn.Linear(self.hidden_size, self.num_classes)
        self.softmax = torch.nn.Softmax(dim=1)

    def forward(self, x):
        hidden = self.fc1(x)
        relu = self.relu(hidden)
        output = self.fc2(relu)
        output = self.softmax(output)
        return output
```

M5 INTRODUCTION TO PYTORCH

- **Using GPU**
  - Setting your GPU device
    - torch.cuda.set_device(device=gpu_id)
  - Converting your model to CUDA tensors:
    - model.cuda() **(CUDA_VISIBLE_DEVICES=gpu_id python)**
  - Converting your inputs and targets to CUDA tensors:
    - inputs = inputs.cuda()
    - targets = targets.cuda()

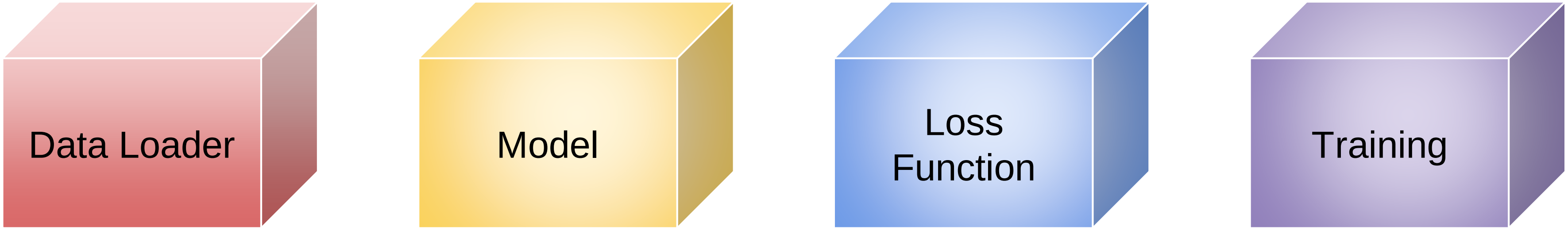
M5 INTRODUCTION TO PYTORCH

- **Using GPU**
  - Setting your GPU device
    - device = 'cpu' || 'cuda' (default) || 'cuda:0' (gpu 0)
  - Converting your model to CUDA tensors:
    - model.to(device)
  - Converting your inputs and targets to CUDA tensors:
    - inputs = inputs.to(device)
    - targets = targets.to(device)

M5 INTRODUCTION TO PYTORCH

- **Using GPU**
  - Checking GPU is available:
    - torch.cuda.is_available()
  - Numpy cannot work on CUDA Tensors, you need to send them to CPU before performing any operation on numpy:
    - var_cpu = var_gpu.cpu()
  - Once all required operations on numpy have been done, remember to transform your data again to CUDA Tensors:
    - var_gpu = var_cpu.cuda()

M5 INTRODUCTION TO PYTORCH



Data Loader

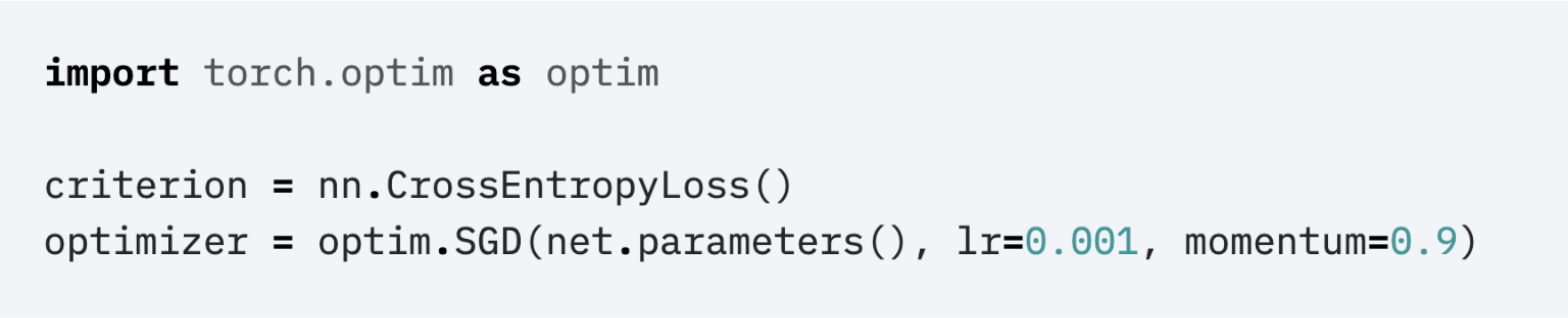
Model

Loss Function

Training

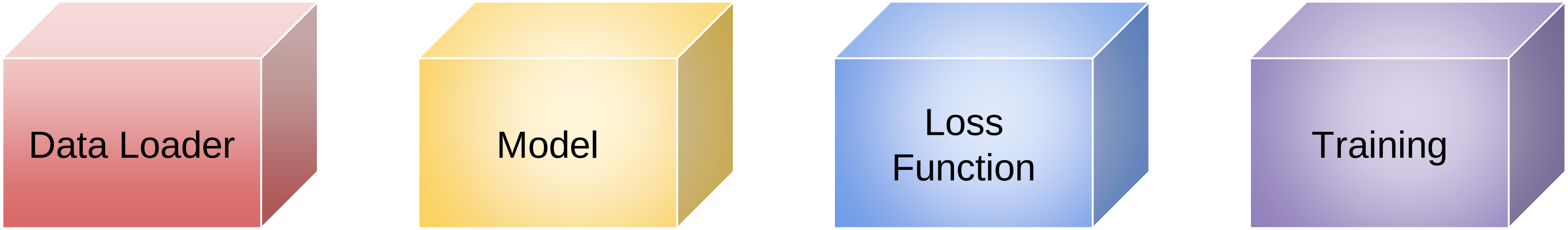
M5 INTRODUCTION TO PYTORCH

- **Optimizer and Loss**
  - Pytorch loss functions
    - MSE
    - CrossEntropyLoss
    - BCELoss
    - …



```
import torch.optim as optim

criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(net.parameters(), lr=0.001, momentum=0.9)
```

M5 INTRODUCTION TO PYTORCH



Data Loader

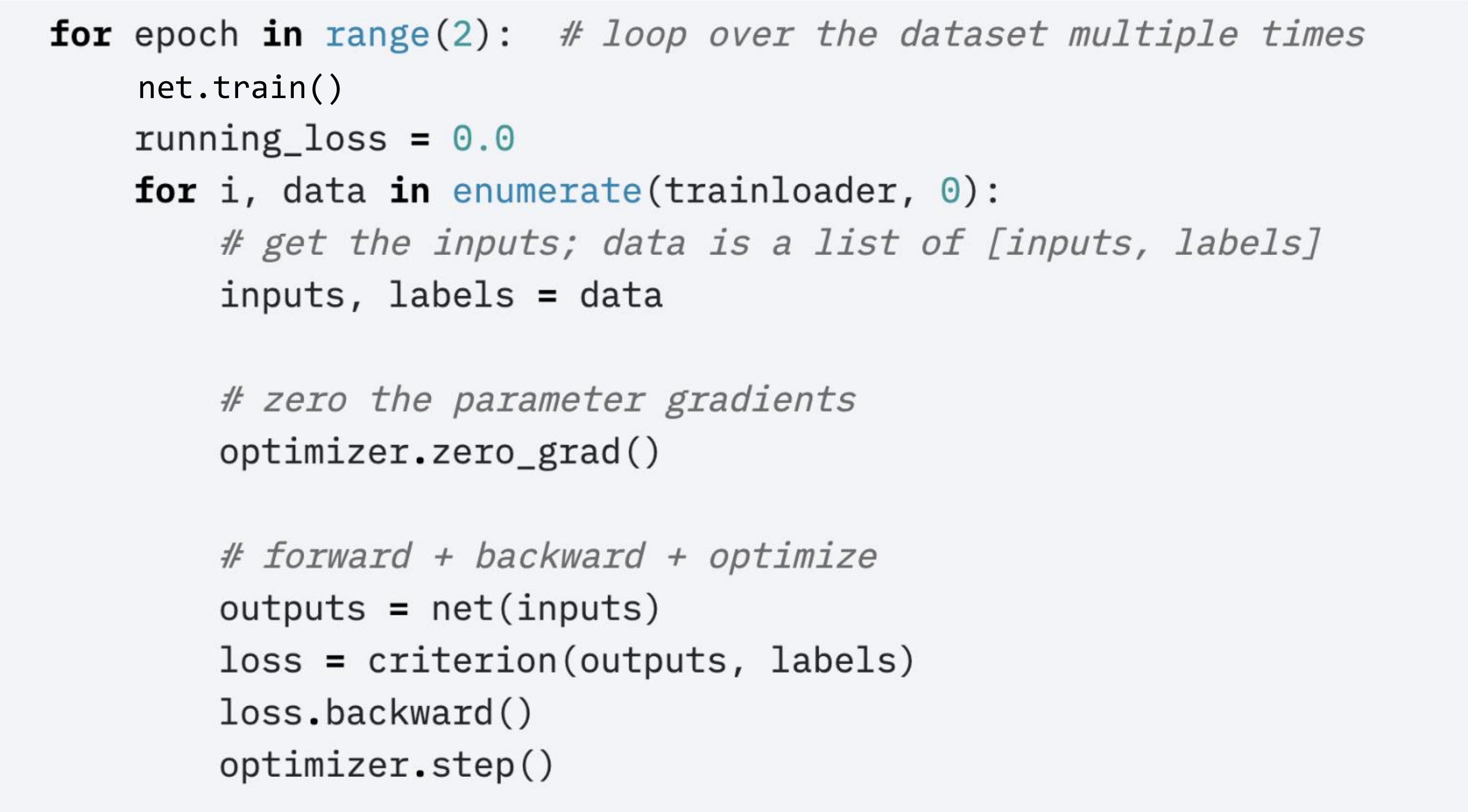
Model

Loss Function

Training

M5 INTRODUCTION TO PYTORCH

- **Training your network**

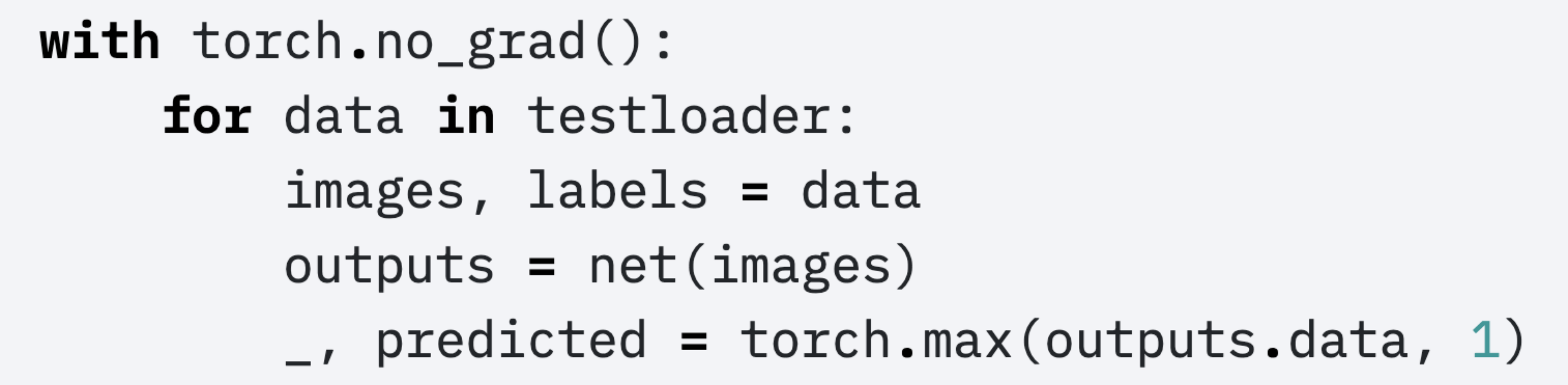


```
for epoch in range(2):  # loop over the dataset multiple times
    net.train()
    running_loss = 0.0
    for i, data in enumerate(trainloader, 0):
        # get the inputs; data is a list of [inputs, labels]
        inputs, labels = data

        # zero the parameter gradients
        optimizer.zero_grad()

        # forward + backward + optimize
        outputs = net(inputs)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()
```

M5 INTRODUCTION TO PYTORCH

- **Testing your network**

```
net.eval()
```



```
with torch.no_grad():
    for data in testloader:
        images, labels = data
        outputs = net(images)
        _, predicted = torch.max(outputs.data, 1)
```

M5 INTRODUCTION TO PYTORCH

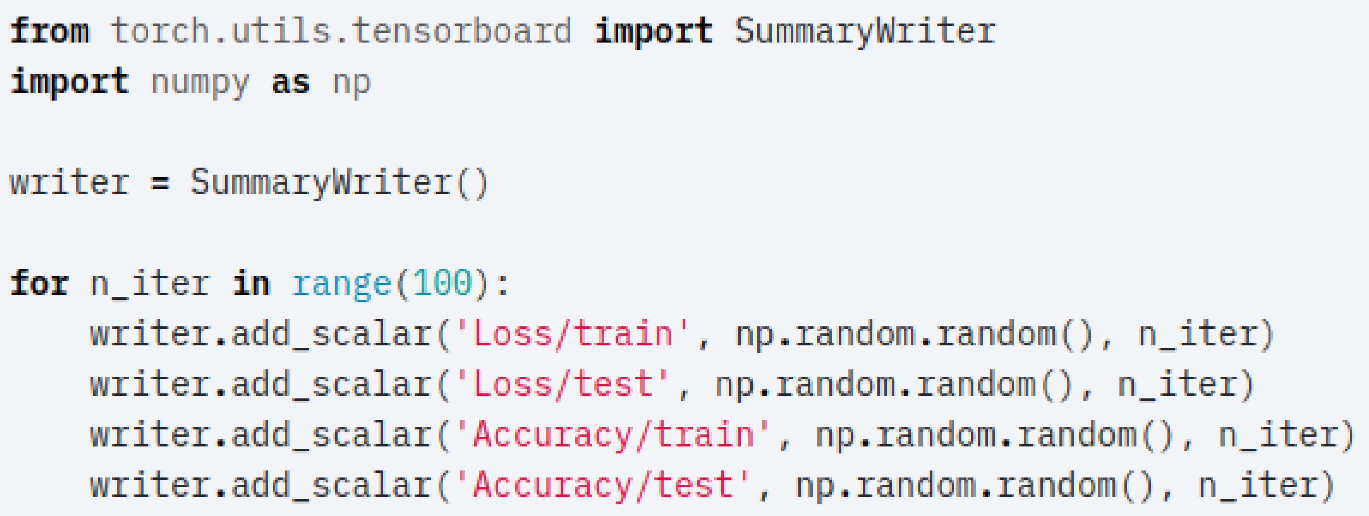
- **Monitoring your training**:
    - With Tensorboard
        - Pytorch Tensorboard
        - **Installation in server:**

            -
              ```
              conda install -c conda-forge tensorboardx
              ```
        - **How to run it:**

            -
              ```
              tensorboard --logdir=/path/to/summary/file --port XXXX (--bind_all)
              ```

M5 INTRODUCTION TO PYTORCH

- **Monitoring your training**:
    - With Tensorboard



```
from torch.utils.tensorboard import SummaryWriter
import numpy as np

writer = SummaryWriter()

for n_iter in range(100):
    writer.add_scalar('Loss/train', np.random.random(), n_iter)
    writer.add_scalar('Loss/test', np.random.random(), n_iter)
    writer.add_scalar('Accuracy/train', np.random.random(), n_iter)
    writer.add_scalar('Accuracy/test', np.random.random(), n_iter)
```

M5 INTRODUCTION TO PYTORCH

- **Monitoring your training**:
  - With Weights and Bias (WandB)

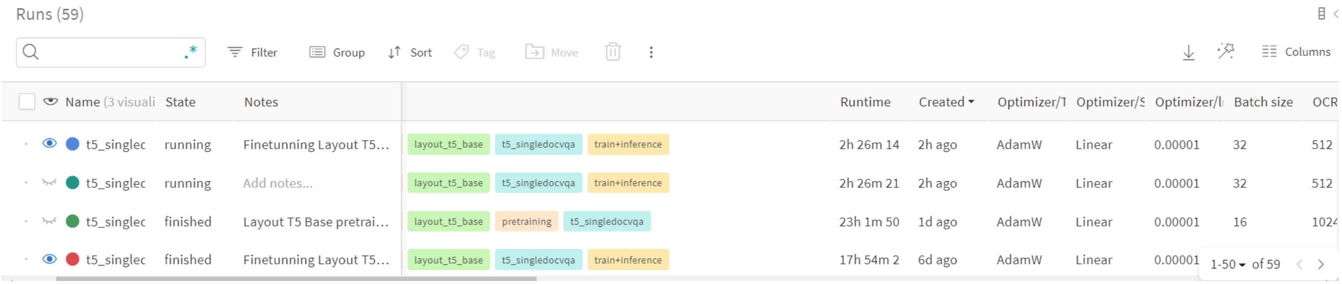
```
pip install wandb
```

```
import wandb as wb

writer = wb.init(name=“My experiment 1”, project=“P1”,
                 config={“bs”: 2, “lr”: 1e-5}, entity=“G1”)

writer.log({‘Loss/train’: net_loss, ‘Accuracy/train’: net_acc}, n_iter)
```

```
writer = wb.init(name=“My experiment 1”, project=“P1”,
                 config={“bs”: 2, “lr”: 1e-5}, entity=“G1”,
                 sync_tensorboard=True)
```

M5 INTRODUCTION TO PYTORCH

- Few things to be considered in Pytorch:
  - **Monitoring your training with WandB**

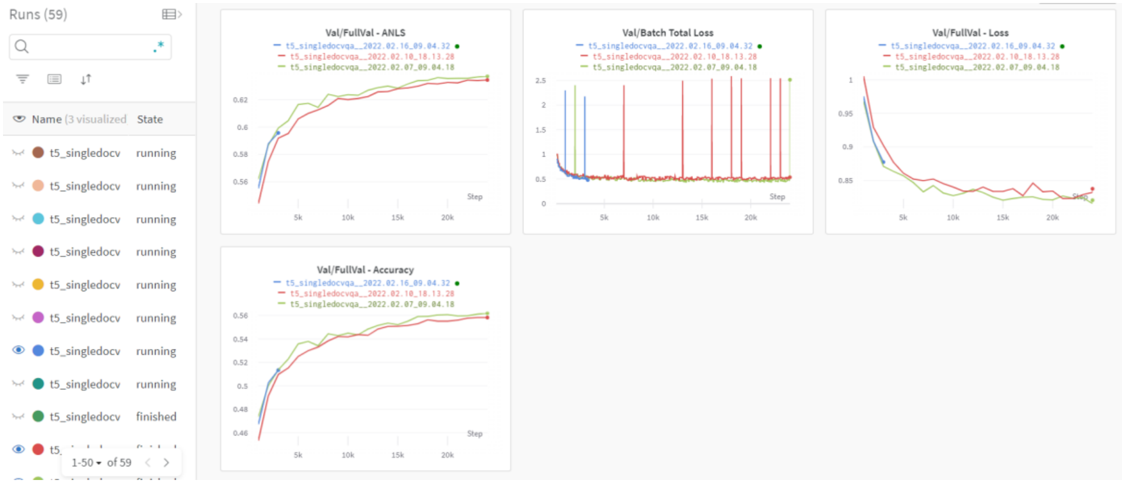


Runs (59)

Filter Group Sort Tag Move Columns

□ Name (3 visuali	State	Notes		Runtime	Created ▾	Optimizer/T	Optimizer/S	Optimizer/l	Batch size	OCR
t5_singlec	running	Finetunning Layout T5…	layout_t5_base t5_singledocvqa train+inference	2h 26m 14	2h ago	AdamW	Linear	0.00001	32	512
t5_singlec	running	Add notes…	layout_t5_base t5_singledocvqa train+inference	2h 26m 21	2h ago	AdamW	Linear	0.00001	32	512
t5_singlec	finished	Layout T5 Base pretrai…	layout_t5_base pretraining t5_singledocvqa	23h 1m 50	1d ago	AdamW	Linear	0.00001	16	1024
t5_singlec	finished	Finetunning Layout T5…	layout_t5_base t5_singledocvqa train+inference	17h 54m 2	6d ago	AdamW	Linear	0.00001		

1-50 ▾ of 59



Runs (59)

Name (3 visualized)	State
t5_singledocv	running
t5_singledocv	running
t5_singledocv	running
t5_singledocv	running
t5_singledocv	running
t5_singledocv	running
t5_singledocv	running
t5_singledocv	running
t5_singledocv	finished
t5_	

1-50 ▾ of 59

Val/FullVal - ANLS
t5_singledocvqa__2022.02.16_09.04.32
t5_singledocvqa__2022.02.10_18.13.28
t5_singledocvqa__2022.02.07_09.04.18
0.62
0.6
0.58
0.56
Step
5k
10k
15k
20k
Val/Batch Total Loss
t5_singledocvqa__2022.02.16_09.04.32
t5_singledocvqa__2022.02.10_18.13.28
t5_singledocvqa__2022.02.07_09.04.18
2.5
2
1.5
1
0.5
0
Step
5k
10k
15k
20k
Val/FullVal - Loss
t5_singledocvqa__2022.02.16_09.04.32
t5_singledocvqa__2022.02.10_18.13.28
t5_singledocvqa__2022.02.07_09.04.18
1
0.95
0.9
0.85
Step
5k
10k
15k
20k
Val/FullVal - Accuracy
t5_singledocvqa__2022.02.16_09.04.32
t5_singledocvqa__2022.02.10_18.13.28
t5_singledocvqa__2022.02.07_09.04.18
0.56
0.54
0.52
0.5
0.48
0.46
Step
5k
10k
15k
20k

M5 INTRODUCTION TO PYTORCH

GPU Cluster use:

- Cluster information
  - Host: 158.109.75.50
  - SSH Port: 22
  - Username: group{01,02,...,10}
  - Passwd: *****

- Code edition/management:
  - Edit directly in the server. Connecting with MobaXterm you can right-click and edit with local editor.
  - Edit in local, send the new files to the server.
  - Edit in local, push/pull with GitHub

GPU Cluster use:

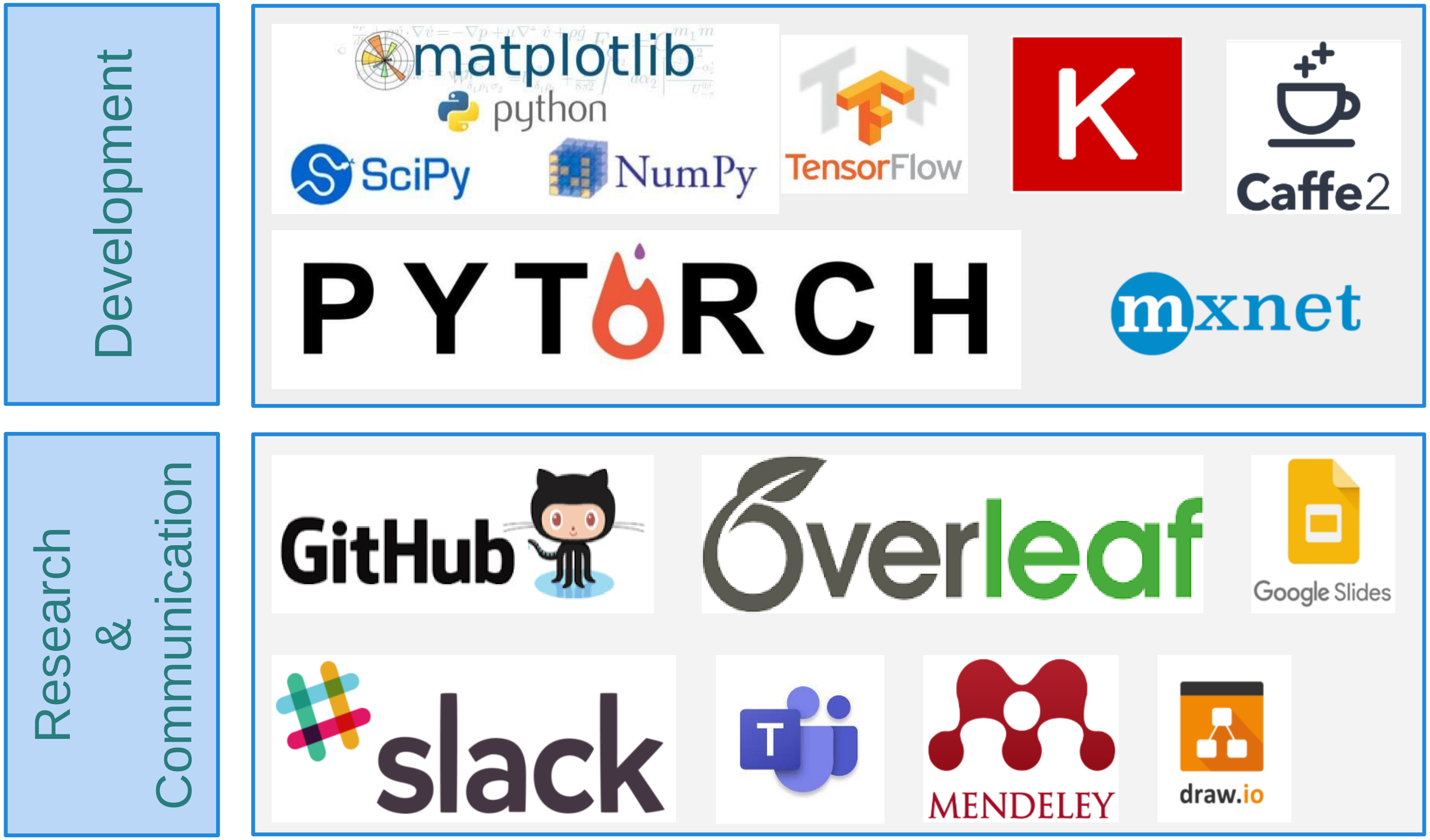
- Job management:
  - SLURM
  - /home/example/Graphics DCC Cluster User's Guide for MCV V2.pdf
    - Cluster information is deprecated (IP, port, etc.)
  - /home/example/ mtgpulow.sh | tgpu.sh | mtgpuhigh.sh
  - Partition (-p) and QOS (-q) especially important. Check documentation. Use case C is recommended.
  - **IMPORTANT**: Save temporal results (weights) during training every N iterations/epochs. Always can happen something that breaks, kills your process.

M5 INTRODUCTION TO PYTORCH

GPU Cluster use:

- When the job is launched:
  - Check constantly during the first ~2 minutes everything is OK.
    - sinfo, squeue, tensorboard/wandb logger
    - watch –n **2** nvidia-smi

M5 Project: Required tools



Development

Research & Communication

Details on tasks, deliverables, and marks for this week

Week 1. Introduction to Python

M5 Project Stages and Schedule

Week 1	**Introduction to Pytorch - Image Classification**
Week 2	**Object Detection, Recognition and Segmentation**
Week 3	
Week 4	**Image Retrieval**
Week 5	**Cross-modal Retrieval**
Week 6	**Presentation**

M5 Project: Goals per week

Week 1: Introduction to Pytorch

Goals

(a) Form groups.
(b) Install and setup the development framework.
(c) Set Up collaborative tools.
(d) Understand Pytorch framework.
(e) Implement Image Classification network from M3 in Pytorch
(f) Compute loss graphs and compare them with yours from Keras
(g) Compute accuracy graphs and compare them with yours from Keras

Marks

C - Achieve (a)-(e) goals
B - Achieve (a)-(f) goals
A - Achieve (a)-(g) goals

Deliverable (for next week)

- **Github** repository with readme.md (Members of the group, code explanation & instructions)
- Presentation with loss and accuracy graphs as well as table with accuracy

Week 1: Introduction to Pytorch

- This week there is an assignment with its own deliverables for introduction to Pytorch based on the task on Image Classification done in M3.

- It is **important** to do the proposed tasks because they will:
    - Help you to understand Pytorch framework.
    - Prepare you to start next week's project.

- In the following slides we detail the tasks to be done.

Week 1: Introduction to Pytorch (Tasks)

Task (a): **Form teams**

- There will be a maximum of 8 groups.
- No more than 4 people per group.
- Follow the link in Campus Virtual to fill in information about the group.

Important

Each member of the team should/must contribute equally to each assignment. Moreover, we will ask you who did what.

Week 1: Introduction to Pytorch (Tasks)

Task (b): **Install and setup the development framework**

- If you use the master GPU cluster everything should be already installed (basic software and datasets). **Check it!**
    - Host: 158.109.75.50
    - SSH Port: 22
    - Username: group{01,02,...,10}
    - Passwd: *****

- Browse images in the dataset directory in the GPU cluster
    - /home/mcv/m5/datasets/MIT_split/
    - This is the same dataset as used in M3

Week 1: Introduction to Pytorch (Tasks)

Task (c): **Setup collaboration tools**

- **GitHub** repository for the code management.
  - Create your own github repository (one per group)
  - Structure the github according to weeks.

- **Overleaf**: Project for the reports (next week tasks)

- **PPT / Google Slides:** Project for the presentations.

Week 1: Introduction to Pytorch (Deliverable)

- **All the deliverables should be accessible through your group GitHub project.**
- We recommend you to make public your GitHub project so you can share your results with other groups.
- Add Rubèn and Ernest (rubenpt91, evalveny) as contributors.
- Add the link to your GitHub project in the link with information about groups in Campus Virtual

Week 1: Introduction to Pytorch (Deliverable)

Your GitHub should contain a README.md file with:

- Title of the project.
- Name of the group.
- Name and contact email of all the team members.
- Link to the Overleaf article (Non-editable link) at the moment no content yet.
- Links to the presentations with the summary of your weekly work.

Week 1: Introduction to Pytorch (Tasks)

Task (d): **Understand Pytorch framework**

- **Check Pytorch documentation**
  - Deep learning with Pytorch:
    - Deep learning 60min blitz tutorial
    - Training a classifier: CIFAR 10 tutorial
    - Torchvision documentation main page
- **Install Pytorch framework**
  - Follow instructions from /home/mcv/installing_m5.txt

Week 1: Introduction to Pytorch (Tasks)

Task (e): **Implement Image Classification network from M3 in Pytorch**

- **GitHub** repository for the code management.
  - Include your implementation in your github repository
- **Project presentation**.
  - Include the description of your network architecture in your presentation (diagrams are welcome).

Week 1: Introduction to Pytorch (Tasks)

Task (f): **Compute losses graphs and compare them with yours from Keras**

- **Project presentation**:
  - Include your training and validation losses graphes from Keras and Pytorch in your presentation.

Week 1: Introduction to Pytorch (Tasks)

Task (g): **Compute accuracy graphs and compare them with yours from Keras**

- **Project presentation**.
  - Include your accuracy graphs from Keras and Pytorch i your presentation.
  - Include your final results in a accuracy table that will find at the end in your presentation.

Week 1: Introduction to Pytorch (Tasks)

Project presentation

- Include 1 final slide with 2 items:
  - Main results and conclusions
  - Main difficulties and problems (if any)
- One of the group members will have to present this slide in 1-2 minutes in the follow-up session next week.

Week 1: Introduction to Pytorch (Deadline)

Due date: 14th of March, Monday, before 10:00 AM