
what have we learned at ALT-F1?

Release 0.2019.09.09 07.43

Abdelkrim Boujraf, Youssef Boujraf

Sep 09, 2019

Contents

1	Secure and Private AI	2
1.1	Deep learning with PyTorch	2
1.1.1	Install Python3	2
1.1.2	Install PyTorch	2
1.1.3	Launching Jupyter Notebook App	3
1.1.4	Udacity course : Deep Learning with PyTorch	3
1.2	Tools	3
2	Glossary	4
3	Indices and tables	6
	Index	7

Contact us : <https://by-systems.weebly.com>

course source : <https://classroom.udacity.com/courses/ud185>

other course : <https://www.udacity.com/school-of-ai>

1.1 Deep learning with PyTorch

- <https://research.fb.com/category/facebook-ai-research/>
- Notebooks : <https://github.com/udacity/deep-learning-v2-pytorch>

1.1.1 Install Python3

- create a python3.7.X environment : `conda create -n py37 python=3.7 anaconda`
- activate the environment `conda activate py37`
- deactivate the environment `conda deactivate`
- determining my environment : `conda info --envs`

1.1.2 Install PyTorch

- **Install Conda** [<https://conda.io/en/latest/>]
 - install Anaconda : <https://docs.anaconda.com/anaconda/install/>
 - or install Miniconda : <https://docs.conda.io/en/latest/miniconda.html>
 - **Some commands**
 - * managing environments : <https://conda.io/projects/conda/en/latest/user-guide/getting-started.html#managing-environments>
 - * example of commands : `conda search scipy`, `conda install scipy`, `conda build my_fun_package`, `conda update conda`
- **Install PyTorch** <https://pytorch.org/get-started/locally/>
 - `conda install pytorch torchvision cudatoolkit=10.0 -c pytorch`

- **Install numpy, jupyter and notebook :**
 - *conda install numpy jupyter notebook*

1.1.3 Launching Jupyter Notebook App

- *jupyter notebook* - <https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/execute.html>

1.1.4 Udacity course : Deep Learning with PyTorch

This repo contains notebooks and related code for Udacity's Deep Learning with PyTorch lesson. This lesson appears in our [AI Programming with Python Nanodegree program](<https://www.udacity.com/course/ai-programming-python-nanodegree-nd089>).

- **Part 1:** Introduction to PyTorch and using tensors
- **Part 2:** Building fully-connected neural networks with PyTorch
- **Part 3:** How to train a fully-connected network with backpropagation on MNIST
- **Part 4:** Exercise - train a neural network on Fashion-MNIST
- **Part 5:** Using a trained network for making predictions and validating networks
- **Part 6:** How to save and load trained models
- **Part 7:** Load image data with torchvision, also data augmentation
- **Part 8:** Use transfer learning to train a state-of-the-art image classifier for dogs and cats

1.2 Tools

- Gym is a toolkit for developing and comparing reinforcement learning algorithms. It supports teaching agents everything from walking to playing games like Pong or Pinball. <https://gym.openai.com/>
- ONNX is an open format to represent deep learning models. With ONNX, AI developers can more easily move models between state-of-the-art tools and choose the combination that is best for them. ONNX is developed and supported by a community of partners. <https://onnx.ai/>

alt-f1 ALT-F1 designs, implements, deploys and supports secure, large-scale software solutions for diverse industries: Manufacturing, MRO, Warehouse, Broadcasting, Bank, Insurance, Law Enforcement, Counter terrorism, Justice & Serious International Crime

autograd Module that PyTorch uses to calculate gradients for training neural networks See <https://pytorch.org/docs/stable/notes/autograd.html>

BY Systems See <https://by-systems.weebly.com>

Conda Package, dependency and environment management for any language—Python, R, Ruby, Lua, Scala, Java, JavaScript, C/ C++, FORTRAN

CUDA PyTorch uses a library called CUDA to accelerate operations using the GPU

Gradients A gradient is a partial derivative — why partial? Because one computes it with respect to (w.r.t.) a single parameter. We have two parameters, a and b, so we must compute two partial derivatives See <https://towardsdatascience.com/understanding-pytorch-with-an-example-a-step-by-step-tutorial-81fc5f8c4e8e>

Layers The first layer shown on the bottom here are the inputs, understandably called the input layer. The middle layer is called the hidden layer, and the final layer (on the right) is the output layer. Source : <https://classroom.udacity.com/courses/ud185/lessons/8a993162-65c4-4a80-bd35-47d9f3a6f5bc/concepts/70526adf-40d3-4446-ac32-d3f798739745>

NumPy Interacts with PyTorch

OpenMined OpenMined is an open-source community focused on researching, developing, and promoting tools for secure, privacy-preserving, value-aligned artificial intelligence. <https://www.openmined.org>

PyTorch To build a network and run data forward through it

Session Initiation Protocol See https://en.wikipedia.org/wiki/Session_Initiation_Protocol

Sigmoid function A sigmoid function is a mathematical function having a characteristic “S”-shaped curve or sigmoid curve. See <https://en.wikipedia.org/wiki/Sigmoid>

SIP See *Session Initiation Protocol*

SIREMIS Web Management Interface for Kamailio (OpenSER) SIP Server See <https://siremis.asipto.com>

tensors The main data structure of PyTorch. the tensor is an array. A vector is a 1-dimensional tensor, a matrix is a 2-dimensional tensor, an array with three indices is a 3-dimensional tensor (RGB color images for example)

torchvision The torchvision package consists of popular datasets, model architectures, and common image transformations for computer vision. See <https://pytorch.org/docs/stable/torchvision/index.html>

Validation the action of checking or proving the validity or accuracy of the model generated by the Artificial Intelligence

Validation Dataset The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters. The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration. See <https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7>

CHAPTER 3

Indices and tables

- `genindex`
- `search`

A

alt-f1, [4](#)
autograd, [4](#)

B

BY Systems, [4](#)

C

Conda, [4](#)
CUDA, [4](#)

G

Gradients, [4](#)

L

Layers, [4](#)

N

NumPy, [4](#)

O

OpenMined, [4](#)

P

PyTorch, [4](#)

S

Session Initiation Protocol, [4](#)
Sigmoid function, [4](#)
SIP, [4](#)
SIREMIS, [4](#)

T

tensors, [4](#)
torchvision, [5](#)

V

Validation, [5](#)
Validation Dataset, [5](#)