

Deep Learning Course

Picsart Academy

Vitali Avagyan

Senior Data Scientist, TurinTech

December 22, 2022

Outline

- 1 Outline
- 2 Deep Learning
- 3 What is PyTorch?
- 4 Linear and Logistic Regressions as NNs
- 5 Deep Neural Networks
- 6 Convolutional Neural Networks
- 7 Natural Language Processing
- 8 Recurrent Neural Networks
- 9 JAX and Distributed Training

Session 1

Deep Learning

What is Deep Learning?

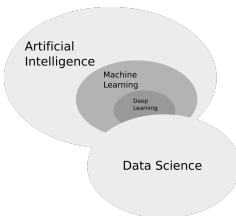


Figure 1: Where does Deep Learning stand in AI?

"Deep learning is a specific subfield of machine learning: a new take on learning representations from data that puts an emphasis on learning successive layers of increasingly meaningful representations. The "deep" in "deep learning" isn't a reference to any kind of deeper understanding achieved by the approach; rather, it stands for this idea of successive layers of representations."

François Chollet in Deep Learning with Python, Second Edition

Frameworks

What is **Deep Learning Framework**?

"Deep learning (DL) frameworks offer building blocks for designing, training, and validating deep neural networks through a high-level programming interface."

Nvidia

Most popular:

- **PyTorch** ← gaining momentum
- **TensorFlow** and **Keras**
- **MXNet**
- **JAX** ← gaining momentum

Introduction: Recommended Material

- Chapter 1, [Deep Learning with Python, Second Edition](#) by François Chollet
- Chapter 1, [Dive into Deep Learning](#) by Zhang A. et al.
- YouTube: [INTRODUCTION TO PYTORCH](#)
- Chapter 1, [Neural Networks and Deep Learning](#) by Michael Nielsen
- Introduction, [Deep Learning](#) by Yoshua Bengio, Ian Goodfellow and Aaron Courville
- GitHub: [Awesome Deep Learning](#)
- Chapter 1, [Deep Learning with PyTorch: A practical approach to building neural network models using PyTorch](#) by Vishnu Subramanian
- Chapter 1, [Deep Learning with PyTorch: Build, Train, and Tune Neural Networks Using Python Tools](#) by Eli Stevens, Luca Antiga, Thomas Viehmann

What is PyTorch?



"An open source machine learning framework that accelerates the path from research prototyping to production deployment"

[PyTorch Webpage](#)

- Tensors
- Datasets, Dataloaders and Transforms
- Autograd
- Vectorisation
- Computational Graph

PyTorch Ecosystem Tools

- Python API
- Ecosystem Tools
 - [Lightning](#): Simplified PyTorch for Research
 - [pyro](#) and [numpyro](#): Deep Universal Probabilistic Programming
 - [BoTorch](#): Bayesian Optimization in PyTorch
 - [fastai](#): fastai simplifies training fast and accurate neural nets using modern best practices
 - [ONNX Runtime](#): Cross-platform inference and training machine-learning accelerator
 - [Transformers](#) by HuggingFace
 - [Ray](#): A unified framework for scaling AI and Python applications
 - [PyTorch NLP](#): NLP library in Python
 - [detectron2](#): State-of-the-art object detection and segmentation algorithms
 - [Optuna](#): Hyperparameter optimization framework

PyTorch Ecosystem Libraries



- [torchaudio](#): audio and signal processing
- [torchvision](#): popular datasets, model architectures, and common image transformations for computer vision
- [torchtext](#): data processing utilities and popular datasets for NLP
- [torchserve](#): model serving

Introduction to PyTorch, tensors, and operations

What is [Tensor](#)?

"A PyTorch Tensor is basically the same as a numpy array: it does not know anything about deep learning or computational graphs or gradients, and is just a generic n-dimensional array to be used for arbitrary numeric computation."

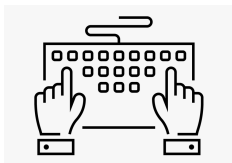
Source: [PYTORCH: TENSORS](#)

Torch tensor:

- Runs on either CPU or GPU
 - For GPU, cast tensor to a [cuda](#) datatype
 - More info on [cuda python](#) and [accelerated computing](#)
- Optimised for automatic differentiation; `grad_fn` property references the backward propagation function

Get used to [numpy library](#) and [numpy array](#) before moving on!

Session 2



Tensors and Operations

GitHub: [tensors](#)

Session 3



Autograd and Vectorisation

GitHub

- autograd
- vectorisation

Session 4 and 5

Linear and Logistic Regressions as Neural Nets

Steps to build a Neural Net:

- Model
- Loss function
- Optimiser
- Training

Hyperparameters for training:

- **Number of Epochs** - the number times to iterate over the dataset
- **Batch Size** - the number of data samples propagated through the network before the parameters are updated
- **Learning Rate** - how much to update models parameters at each batch/epoch ([SGD for Linear Regression at MLU](#))

Implementations:

- [Linear Regression](#)
- [Logistic Regression](#)

Outline
○○

Deep Learning
○○○

What is PyTorch?
○○○○○○○○○○

Linear and Logistic Regressions as NNs
○

Deep Neural Networks
●○○

Convolutional
○○○○○

Deep Neural Networks

Neural Networks and Forward Propagation

Homework 1

Build a simple neural network using PyTorch to classify MNIST digits

Convolutional Neural Networks

Introduction to Convolutional Layers and Image Classification

Transfer Learning and Fine-tuning Pre-trained Models

Object Detection and Instance Segmentation with PyTorch

Homework 2

- Use a pre-trained model to classify images from the CIFAR-10 dataset
- Use a pre-trained model to detect and classify objects in an image

Convolutional Neural Networks: Recommended Reading

- Deep Learning with PyTorch: A practical approach to building neural networks, Chapter 3
- PyTorch for Deep Learning and Computer Vision, Chapter 4
- Hands-On Computer Vision with PyTorch, Chapter 3

Natural Language Processing

Introduction to Word Embeddings and Language Modelling

Sequence labeling and text classification with PyTorch

Machine translation and generation with PyTorch

Homework 3

- Use a pre-trained word embedding model to classify sentences
- Use a pre-trained machine translation model to translate a sentence from English to Armenian

Natural Language Processing: Recommended Reading

- Deep Learning with PyTorch: A practical approach to building neural networks, Chapter 4
- Natural Language Processing with PyTorch, Chapter 2
- Deep Learning with PyTorch, Chapter 5

Recurrent Neural Networks

Introduction to Word Embeddings and Language Modelling

Sequence labeling and text classification with PyTorch

Machine translation and generation with PyTorch

Homework 4

- Use a pre-trained model to generate text based on a given prompt

Recurrent Neural Networks: Recommended Reading

- Deep Learning with PyTorch: A practical approach to building neural networks, Chapter 5
- PyTorch for Deep Learning and Computer Vision, Chapter 5
- Hands-On Computer Vision with PyTorch, Chapter 4

JAX and Distributed Training

Introduction to JAX and its differences from PyTorch

Distributed training with PyTorch and JAX

JAX best practices and advanced techniques

Homework 5

- Use JAX to train a simple neural network on the MNIST dataset

JAX and Distributed Training: Recommended Reading

- Deep Learning with PyTorch: A practical approach to building neural networks, Chapter 6
- JAX: High-performance machine learning with NumPy-style functions, Chapter 3
- Deep Learning with PyTorch, Chapter 6