UNC CHARLOTTE

The WILLIAM STATES LEE COLLEGE *of* ENGINEERING

TECSAR

Transformative Computer Systems and Architecture Research

Real-time AI
Lecture 0: Introduction

Hamed Tabkhi

Department of Electrical and Computer Engineering,

University of North Carolina Charlotte (UNCC)

htabkhiv@uncc.edu

Topics

Chapter 1 introduces PyTorch as a library and its place in the deep learning revolution, and touches on what sets PyTorch apart from other deep learning frameworks.

Chapter 2 shows PyTorch in action by running examples of pretrained networks; it demonstrates how to download and run models in PyTorch Hub.

Chapter 3 introduces the basic building block of PyTorch—the tensor—showing its API and going behind the scenes with some implementation details.

Chapter 4 demonstrates how different kinds of data can be represented as tensors and how deep learning models expects tensors to be shaped.

Chapter 5 walks through the mechanics of learning through gradient descent and how PyTorch enables it with automatic differentiation.

Chapter 6 shows the process of building and training a neural network for regression in PyTorch using the `nn` and `optim` modules.

Chapter 7 builds on the previous chapter to create a fully connected model for image classification and expand the knowledge of the PyTorch API.

Chapter 8 introduces convolutional neural networks and touches on more advanced concepts for building neural network models and their PyTorch implementation.

Main repository of course

https://github.com/deep-learning-with-pytorch/dlwpt-code

Other Major Resources:

https://pytorch.org/
https://docs.fast.ai/
https://docs.nvidia.com/deeplearning/tensorrt/developer-guide/index.html
https://developer.nvidia.com/EMBEDDED/jetson-nano-developer-kit
https://docs.nvidia.com/deeplearning/tensorrt/developer-guide/index.html#c_topics

Requirements

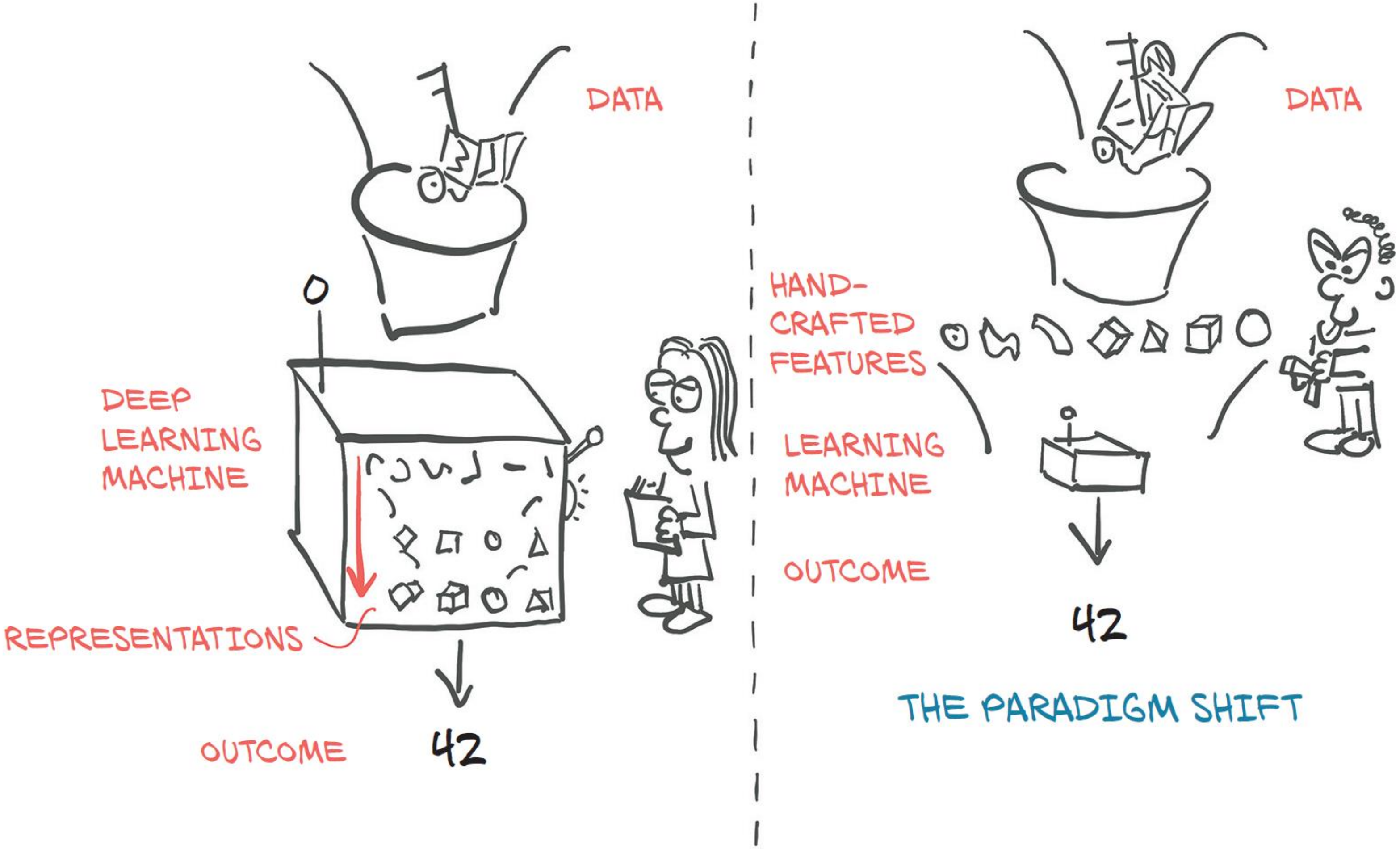
Create a google co-lab account:
https://course19.fast.ai/start_colab.html

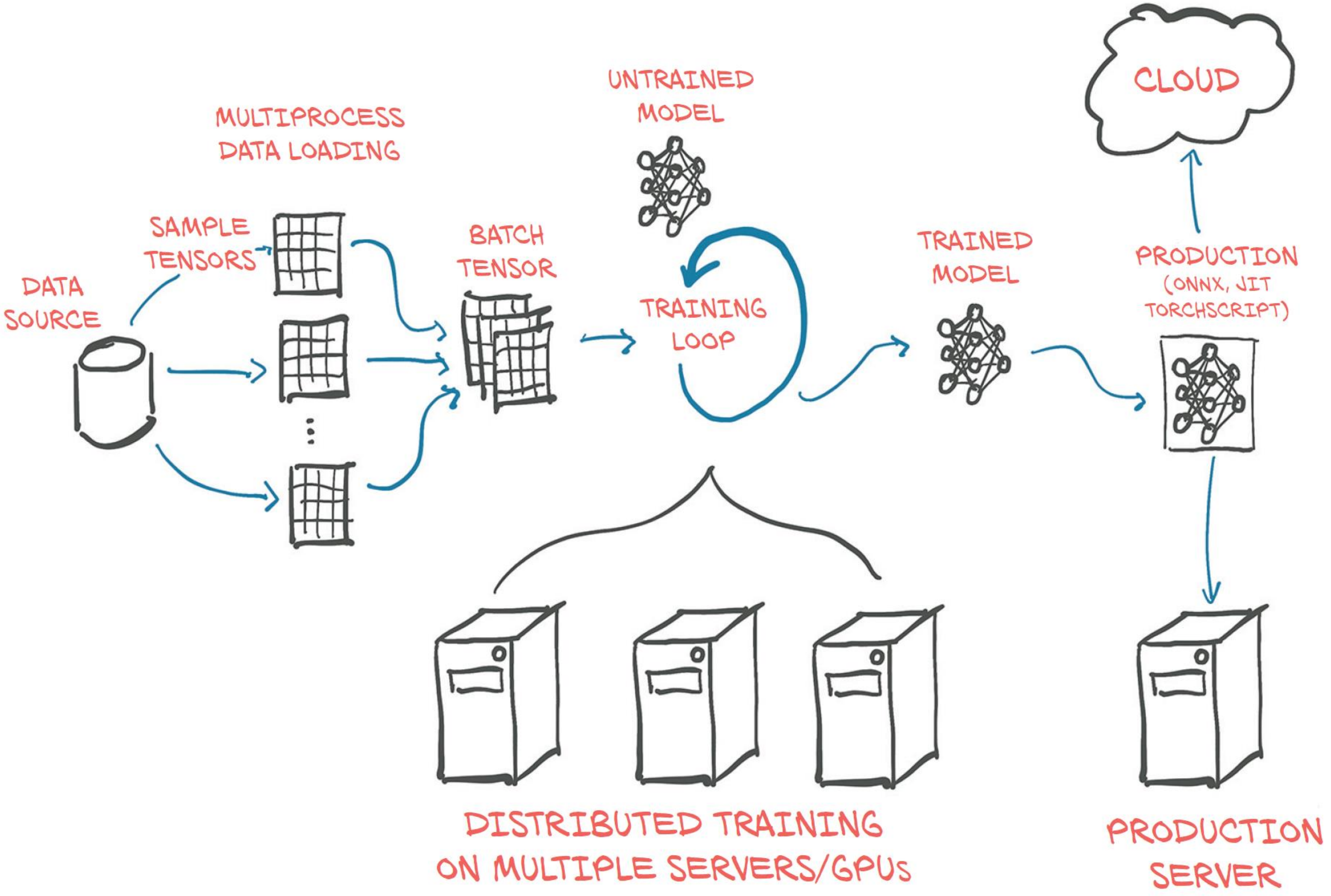
Purchase Nvidia Jetson Nano development kit:
https://www.amazon.com/dp/B08J157LHH

Having a public GitHub Repository, using your UNCC email account:
https://github.com/
https://guides.github.com/

Nvidia Jetson Nano Development Kit

https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-2gb-devkit#prepare


DATA
DEEP LEARNING MACHINE
REPRESENTATIONS
OUTCOME
42
DATA
HAND-CRAFTED FEATURES
LEARNING MACHINE
OUTCOME
42
THE PARADIGM SHIFT


MULTIPROCESS DATA LOADING
UNTRAINED MODEL
CLOUD
SAMPLE TENSORS
DATA SOURCE
BATCH TENSOR
TRAINING LOOP
TRAINED MODEL
PRODUCTION (ONNX, JIT TORCHSCRIPT)
DISTRIBUTED TRAINING ON MULTIPLE SERVERS/GPUs
PRODUCTION SERVER

Deep learning competitive landscape

⍰ TensorFlow:
– Consumed Keras entirely, promoting it to a first-class API
– Provided an immediate-execution “eager mode” that is somewhat similar to how PyTorch approaches computation
– Released TF 2.0 with eager mode by default

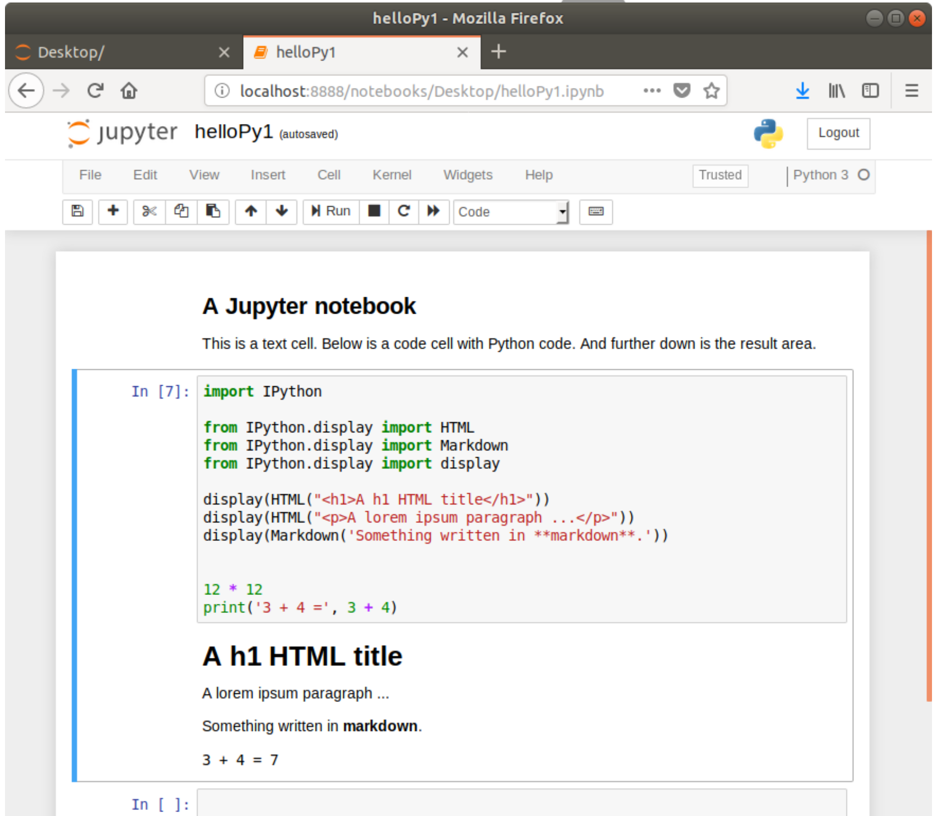
PyTorch:
– Consumed Caffe2 for its backend
– Replaced most of the low-level code reused from the Lua-based Torch project
– Added support for ONNX, a vendor-neutral model description and exchange format
– Added a delayed-execution “graph mode” runtime called *TorchScript*
– Released version 1.0

PyTorch for deep learning

- PyTorch is a library for Python programs that facilitates building deep learning projects.
- It emphasizes flexibility and allows deep learning models to be expressed in idiomatic Python.
- This approachability and ease of use found early adopters in the research community, and in the years since its first release.
- It has grown into one of the most prominent deep learning tools across a broad range of applications.
- PyTorch is easy to recommend because of its simplicity.
- It provides accelerated computation using graphical processing units (GPUs).
- PyTorch provides facilities that support numerical optimization on generic mathematical expressions, which deep learning uses for training.
- A design driver for PyTorch is expressivity, allowing a developer to implement complicated models without undue complexity being imposed by the library (it’s not a framework!).
- PyTorch has been equipped with a high-performance C++ runtime that can be used to deploy models for inference without relying on Python, and can be used for designing and training models in C++.

Jupyter Notebooks

- Jupyter Notebook shows itself as a page in the browser through which we can run code interactively.
- Code is evaluated by a *kernel*, a process running on a server that is ready to receive code to execute and send back the results, which are then rendered inline on the page.
- A notebook maintains the state of the kernel, like variables defined during the evaluation of code, in memory until it is terminated or restarted.
- The fundamental unit with which we interact with a notebook is a
- *cell*: a box on the page where we can type code and have the kernel evaluate it You can read everything about Jupyter Notebooks on the project website (https://jupyter.org).
- Google Colab is just a specialized version of the Jupyter Notebook, which runs on the cloud and offers free computing resources.


helloPy1 - Mozilla Firefox
Desktop/
helloPy1
localhost:8888/notebooks/Desktop/helloPy1.ipynb
jupyter helloPy1 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 ○
Run
Code

A Jupyter notebook

This is a text cell. Below is a code cell with Python code. And further down is the result area.

```
In [7]: import IPython

        from IPython.display import HTML
        from IPython.display import Markdown
        from IPython.display import display

        display(HTML("<h1>A h1 HTML title</h1>"))
        display(HTML("<p>A lorem ipsum paragraph ...</p>"))
        display(Markdown('Something written in **markdown**.'))


        12 * 12
        print('3 + 4 =', 3 + 4)
```

A h1 HTML title

A lorem ipsum paragraph ...

Something written in **markdown**.

```
3 + 4 = 7
```

```
In [ ]:
```

Summary

- Deep learning models automatically learn to associate inputs and desired outputs from examples.
- Libraries like PyTorch allow you to build and train neural network models efficiently.
- PyTorch minimizes cognitive overhead while focusing on flexibility and speed. It also defaults to immediate execution for operations.
- TorchScript allows us to precompile models and invoke them not only from Python but also from C++ programs and on mobile devices.
- Since the release of PyTorch in early 2017, the deep learning tooling ecosystem has consolidated significantly.
- PyTorch provides a number of utility libraries to facilitate deep learning projects.