

# Learning Deep Learning with PyTorch

## (3) Knowing PyTorch

Qiyang Hu  
UCLA IDRE  
April 23, 2020

# What is **PYTORCH**

- An open-source Python-based deep learning framework
  - Primarily developed by Facebook's AI Research lab (FAIR)
  - Replacement for Numpy with supporting GPUs
  - A full set of deep learning libraries
- History
  - Lua-based Torch (2002 - 2011)
  - PyTorch 0.1 (2016): THNN
  - PyTorch 1.0 (2018): merging Caffe2
  - PyTorch 1.5 (Apr 21, 2020) [release note](#)
- About Fast.ai
  - Sitting on top of PyTorch
  - Fast.ai for PyTorch is not what Keras is for TF
  - Still unclear for the usage popularity

# Why **PYTORCH**

- Simplicity & Flexibility from “Py”

- Feels like Numpy
- GPU acceleration

- Immediate execution mode

- Defined by run
- Tape-based autograd
- Awesome debugging

- Hybrid front-end

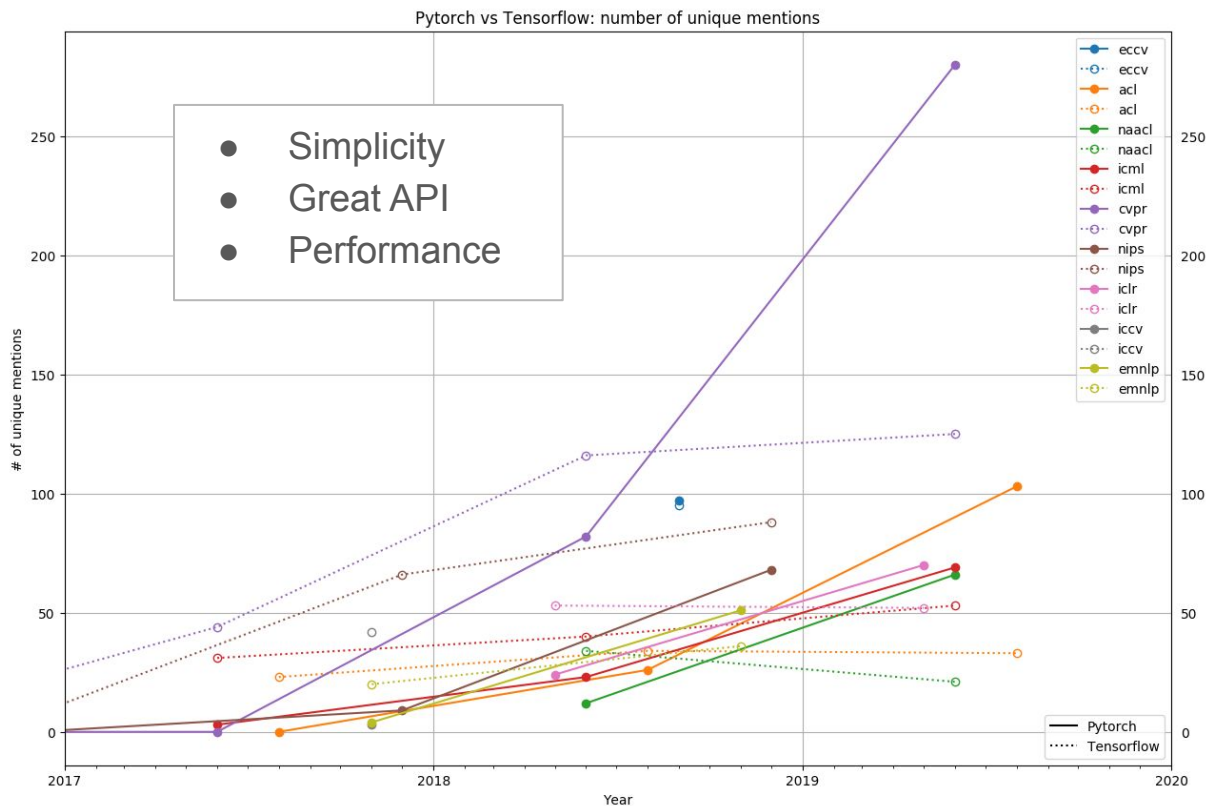
- JIT: TorchScript, Tracing
- Seamlessly switch between
  - Modes
  - Distributed training
  - Mobile deployment

A graph is created on the fly

```
from torch.autograd import Variable  
  
x = Variable(torch.randn(1, 10))  
prev_h = Variable(torch.randn(1, 20))  
W_h = Variable(torch.randn(20, 20))  
W_x = Variable(torch.randn(20, 10))
```



# PyTorch is increasing dominance in research



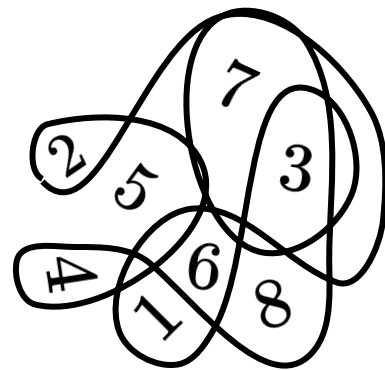
[Source](#)

# Tensors as building blocks

1

$$\begin{bmatrix} 2 \\ 5 \\ 4 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 1 & 3 \\ 5 & 7 & 9 \\ 4 & 8 & 6 \end{bmatrix}$$

$$\begin{bmatrix} \begin{bmatrix} 4 & 6 & 9 \\ 1 & 2 & 5 \\ 8 & 7 & 3 \end{bmatrix} \end{bmatrix}$$


Scalar  
0-D

Vector  
1-D

Matrix  
2-D

Tensor  
3-D

Tensor  
 $n$ -D

$$X = 1$$

$$X[1] = 5$$

$$X[2, 1] = 8$$

$$X[0, 1, 2] = 5$$

$$X[\underbrace{2, 3, \dots, 1}_{N \text{ indices}}] = 6$$

# “Py” and “Non-Py” in PyTorch

- Tensors and Numpy arrays
  - Easy conversion
  - Zero copy: share their underlying memory locations (if on CPU)
  - Unboxed contiguous memory blocks containing unboxed C numeric types
    - Not python objects as in lists or tuples of numbers
    - Views by tensor metadata, e.g. sizes, strides, offsets
- PyTorch = Python + C/C++ + CUDA
  - Python extension objects in C/C++
  - Code base components:
    - The core Torch libraries: TH, THC, THNN, THCUNN
    - Vendor libraries: CuDNN, NCCL
    - Python Extension libraries
    - Additional third-party libraries: NumPy, MKL, LAPACK, DLPack

# Colab Hands-on

[bit.ly/LDDL\\_01](https://bit.ly/LDDL_01)

# Automatic differentiation

- Autograd package
  - Track all operations of tensors
  - Compute derivatives analytically via back-prop
  - Natively loaded in torch module
  - Can be used in other scientific domains
- Simple usage
  - Set tensor's `.requires_grad` as `TRUE`
  - Call `.backward()`
    - Gradient accumulated into `.grad` attribute
    - Tensor's creation function recorded in `.grad_fn` attribute
- Stop a tensor from tracking history
  - `.detach()`
  - Wrap the code block in with `torch.no_grad()`



# Neural Networks in PyTorch

- [torch.nn](#) package
  - Contains all building blocks for NN architectures
  - All blocks subclassed from `nn.Module` (e.g. `nn.Linear`)
- Define a network
  - For simple networks:
    - concatenate modules through a `nn.Sequential` container
  - For complex networks:
    - Subclassing `nn.Module`
- `nn.Module` package expects first index as first batch size of samples
  - Need to reshape the input by `.unsqueeze()`
- Loss functions in `torch.nn`:
  - `L1Loss`, `MSELoss`, `CrossEntropyLoss`, `MarginRankingLoss`, ...

# Optimizers in PyTorch

- [torch.optim](#) package
  - Provides various optimization algorithms
  - Need to move model to GPU before constructing optimizers
  - Must zero the gradient explicitly:
    - `optimizer.zero_grad()`
  - Take an optimization step:
    - `optimizer.step()` in GD method
    - `optimizer.step(closure)` in CG or LBFGS method
  - Optional: adjust the learning rate based on the number of epochs.
    - `optimizer.lr_scheduler`

# Don't forget to

- Github Repo:
  - <https://github.com/huqy/idre-learning-deep-learning-pytorch>
- Slack workspace:
  - [bit.ly/join-LDL](https://bit.ly/join-LDL)
- Contact me
  - [huqy@idre.ucla.edu](mailto:huqy@idre.ucla.edu)
  - Direct message in Slack
- IF you don't have plan to attend the rest of workshops, :
  - Please fill out our series survey: [bit.ly/2X2phyS](https://bit.ly/2X2phyS)