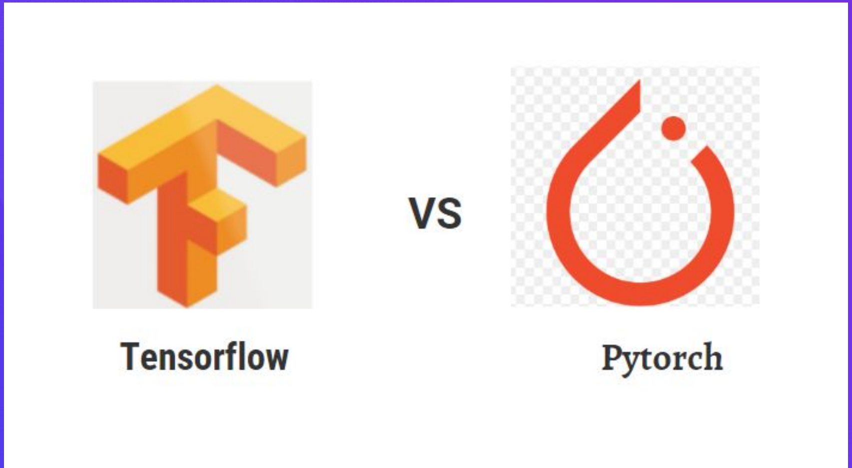
42 Wolfsburg / 42 Berlin AI Community

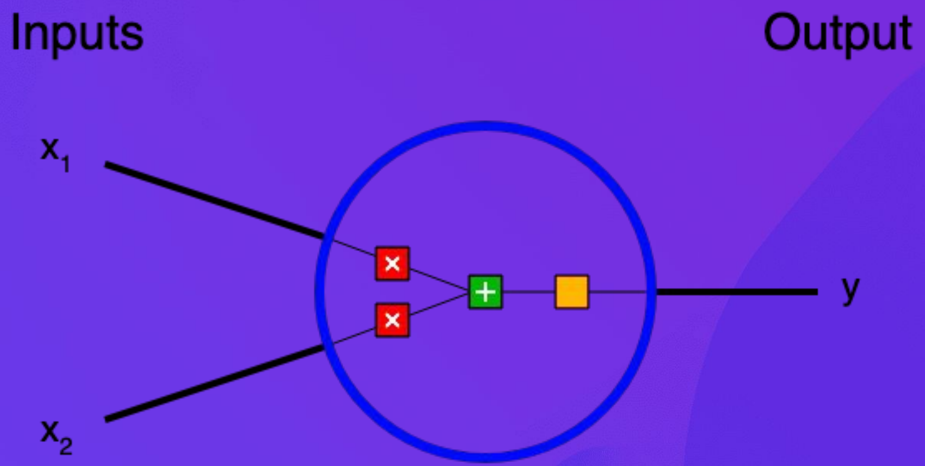
Comparing Deep Learning Frameworks
In
Simple Neural Network



Neural Networks

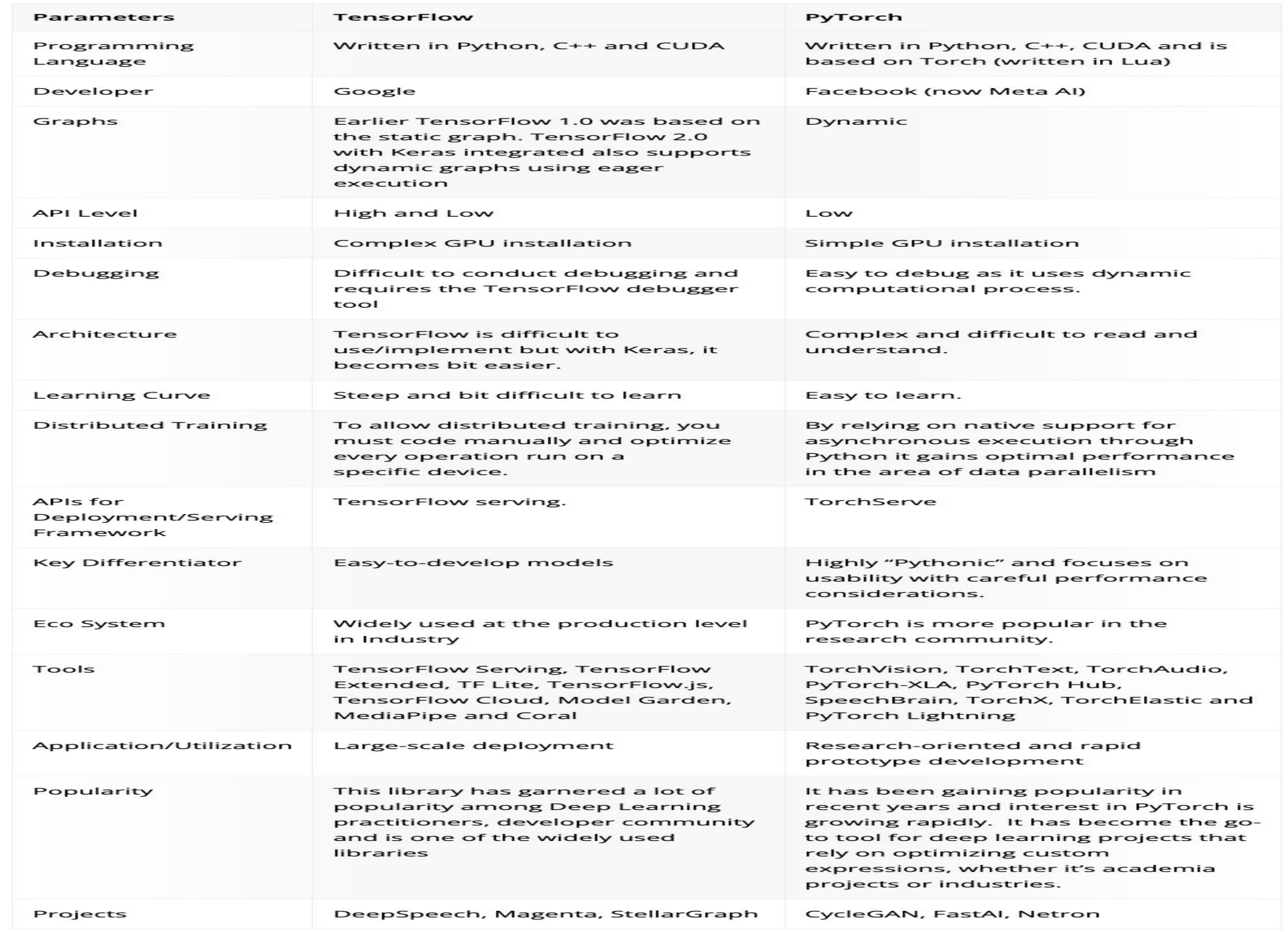
Overview and Applications

- Neural networks are a type of machine learning algorithm inspired by the structure and function of the human brain
- They are used for a wide range of applications, including image recognition, natural language processing, and predictive analytics



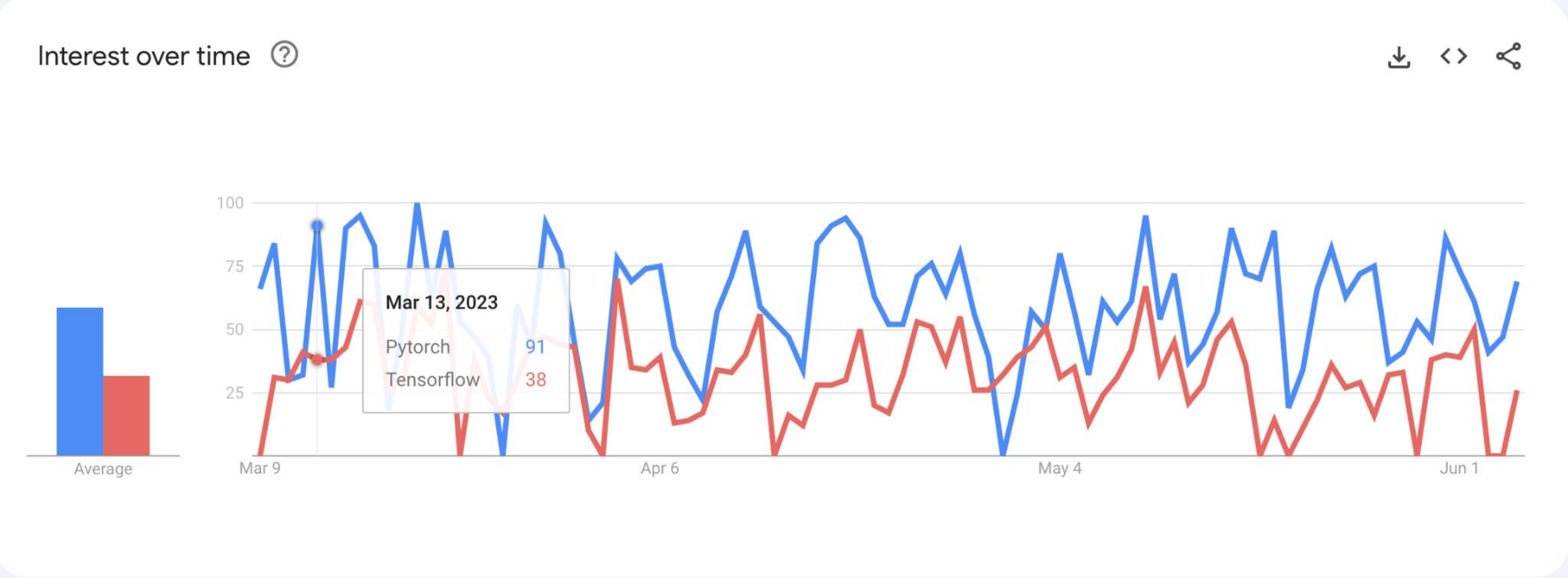
Source :Machine Learning for Beginners: An Introduction to Neural Networks - victorzhou.com

PyTorch vs Tensorflow : Under the Hood



Parameters	TensorFlow	PyTorch
Programming Language	Written in Python, C++ and CUDA	Written in Python, C++, CUDA and is based on Torch (written in Lua)
Developer	Google	Facebook (now Meta AI)
Graphs	Earlier TensorFlow 1.0 was based on the static graph. TensorFlow 2.0 with Keras integrated also supports dynamic graphs using eager execution	Dynamic
API Level	High and Low	Low
Installation	Complex GPU installation	Simple GPU installation
Debugging	Difficult to conduct debugging and requires the TensorFlow debugger tool	Easy to debug as it uses dynamic computational process.
Architecture	TensorFlow is difficult to use/implement but with Keras, it becomes bit easier.	Complex and difficult to read and understand.
Learning Curve	Steep and bit difficult to learn	Easy to learn.
Distributed Training	To allow distributed training, you must code manually and optimize every operation run on a specific device.	By relying on native support for asynchronous execution through Python it gains optimal performance in the area of data parallelism
APIs for Deployment/Serving Framework	TensorFlow serving.	TorchServe
Key Differentiator	Easy-to-develop models	Highly "Pythonic" and focuses on usability with careful performance considerations.
Eco System	Widely used at the production level in Industry	PyTorch is more popular in the research community.
Tools	TensorFlow Serving, TensorFlow Extended, TF Lite, TensorFlow.js, TensorFlow Cloud, Model Garden, MediaPipe and Coral	TorchVision, TorchText, TorchAudio, PyTorch-XLA, PyTorch Hub, SpeechBrain, TorchX, TorchElastic and PyTorch Lightning
Application/Utilization	Large-scale deployment	Research-oriented and rapid prototype development
Popularity	This library has garnered a lot of popularity among Deep Learning practitioners, developer community and is one of the widely used libraries	It has been gaining popularity in recent years and interest in PyTorch is growing rapidly. It has become the go-to tool for deep learning projects that rely on optimizing custom expressions, whether it's academia projects or industries.
Projects	DeepSpeech, Magenta, StellarGraph	CycleGAN, FastAI, Netron

PyTorch vs Tensorflow : Trends in 2023



Interest over time

Mar 13, 2023

Pytorch 91

Tensorflow 38

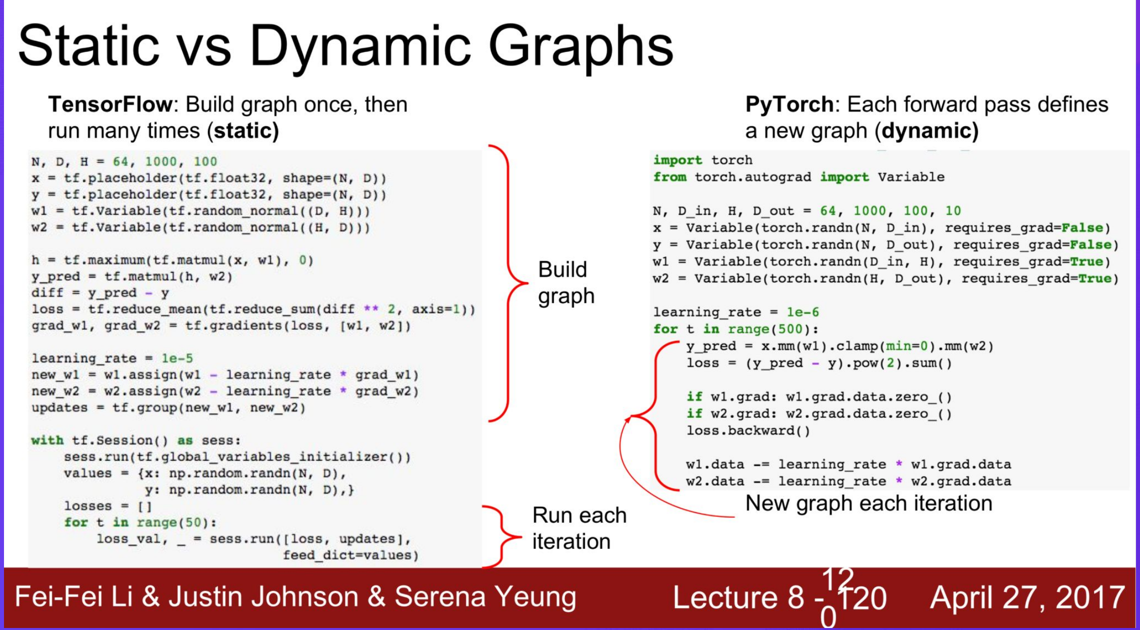
Pytorch vs Tensorflow : Architecture comparison

Static graphs:

- It's a fixed layer architecture.
- The tower is the map and the blocks are the data.

Dynamic graphs:

- A dynamic layer architecture.
- The tower is the map and the blocks are the data,but the map is not fixed and changes with the data.



Static vs Dynamic Graphs

TensorFlow: Build graph once, then run many times (**static**)

```
N, D, H = 64, 1000, 100
x = tf.placeholder(tf.float32, shape=(N, D))
y = tf.placeholder(tf.float32, shape=(N, D))
w1 = tf.Variable(tf.random_normal((D, H)))
w2 = tf.Variable(tf.random_normal((H, D)))

h = tf.maximum(tf.matmul(x, w1), 0)
y_pred = tf.matmul(h, w2)
diff = y_pred - y
loss = tf.reduce_mean(tf.reduce_sum(diff ** 2, axis=1))
grad_w1, grad_w2 = tf.gradients(loss, [w1, w2])

learning_rate = 1e-5
new_w1 = w1.assign(w1 - learning_rate * grad_w1)
new_w2 = w2.assign(w2 - learning_rate * grad_w2)
updates = tf.group(new_w1, new_w2)

with tf.Session() as sess:
    sess.run(tf.global_variables_initializer())
    values = {x: np.random.randn(N, D),
              y: np.random.randn(N, D),}
    losses = []
    for t in range(50):
        loss_val, _ = sess.run([loss, updates],
                               feed_dict=values)
```

Build graph

Run each iteration

PyTorch: Each forward pass defines a new graph (**dynamic**)

```
import torch
from torch.autograd import Variable

N, D_in, H, D_out = 64, 1000, 100, 10
x = Variable(torch.randn(N, D_in), requires_grad=False)
y = Variable(torch.randn(N, D_out), requires_grad=False)
w1 = Variable(torch.randn(D_in, H), requires_grad=True)
w2 = Variable(torch.randn(H, D_out), requires_grad=True)

learning_rate = 1e-6
for t in range(500):
    y_pred = x.mm(w1).clamp(min=0).mm(w2)
    loss = (y_pred - y).pow(2).sum()

    if w1.grad: w1.grad.data.zero_()
    if w2.grad: w2.grad.data.zero_()
    loss.backward()

    w1.data -= learning_rate * w1.grad.data
    w2.data -= learning_rate * w2.grad.data
```

New graph each iteration

Fei-Fei Li & Justin Johnson & Serena Yeung

Lecture 8 - 120

April 27, 2017

Notebook / Places to Code

1. Jupyter Notebook – https://jupyter.org/
2. Google Collab – https://colab.research.google.com/)
3. Pycharm ← VS Code (with Notebook extension)

- PyTorch:
  - Pythonic and intuitive syntax
  - Flexible and creative with dynamic graphs
  - Strong community and rich tools for research
  - Lacks some features for production and deployment

- TensorFlow:
  - Mature and comprehensive ecosystem
  - Various options for production and deployment
  - Better performance and scalability with static graphs and distributed training
  - Steeper learning curve and more verbose and complex syntax than PyTorch
  - Less flexibility and creativity with static graphs
  - May require more debugging for some tasks
  - Good for building robust and scalable applications

Sources for Documentation

- PyTorch documentation
- TensorFlow documentation
- PyTorch vs. TensorFlow for Deep Learning in 2023

Conclusion

Key Similarities and Differences, and Recommendations

- PyTorch and TensorFlow are both powerful machine learning frameworks with their own unique strengths and weaknesses
- Choosing the best framework for a given project depends on the specific requirements and constraints of the project
- By understanding the similarities and differences between PyTorch and TensorFlow, we can make informed decisions about which framework to use for a given project