PyTorch - A deep learning framework

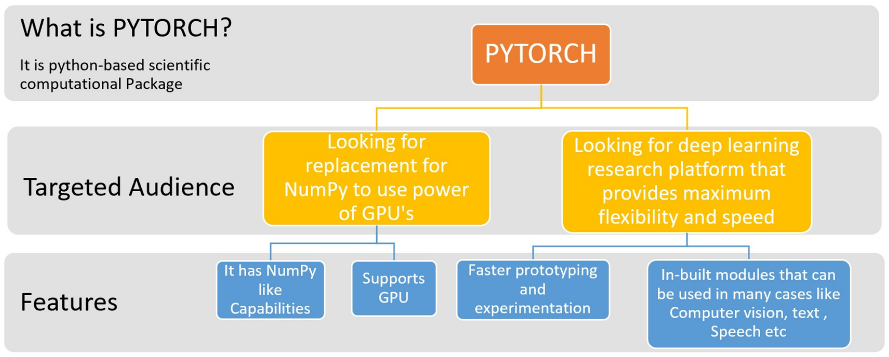
Published on March 18, 2019 Edit article | View stats

Shashi Chilukuri
Data Scientist who is passionate about Artificial Intelligence & solving business problems with it
2 articles

29 2 0 0

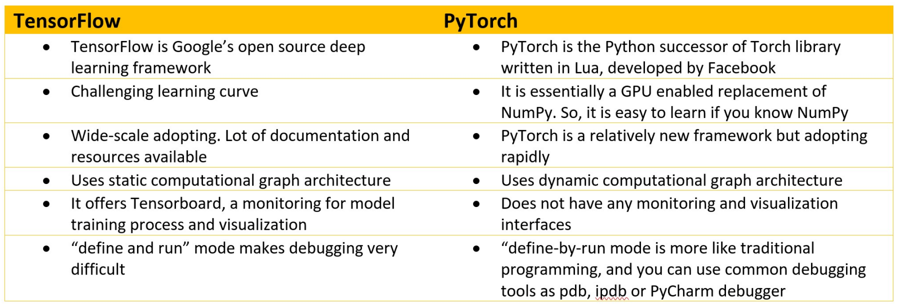
We have many frameworks to create deep learning models, to name some: Tensorflow, PyTorch, Caffe, MXNet, Keras etc. Each of them have their own architecture style and of course with some pros and cons. Lets have a quick look at one of the popular framework "PyTorch".

What is PyTorch?


What is PYTORCH?
It is python-based scientific computational Package
PYTORCH
Targeted Audience
Looking for replacement for NumPy to use power of GPU's
Looking for deep learning research platform that provides maximum flexibility and speed
Features
It has NumPy like Capabilities
Supports GPU
Faster prototyping and experimentation
In-built modules that can be used in many cases like Computer vision, text , Speech etc

So in a nutshell, it is a python based package used for creating deep learning models, that has NumPy (Machine Learning Library) capabilities and supports GPU.

Now lets compare with most popular framework 'TensorFlow' to check how it will standout.



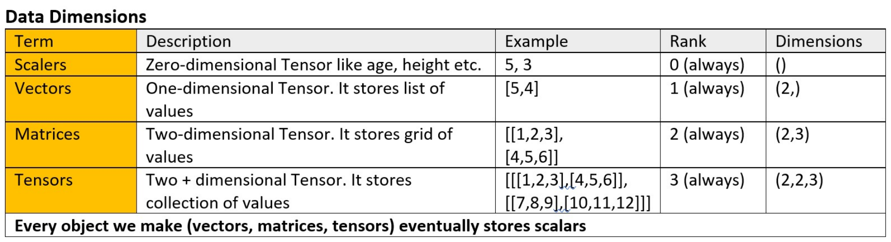
TensorFlow	PyTorch
• TensorFlow is Google's open source deep learning framework	• PyTorch is the Python successor of Torch library written in Lua, developed by Facebook
• Challenging learning curve	• It is essentially a GPU enabled replacement of NumPy. So, it is easy to learn if you know NumPy
• Wide-scale adopting. Lot of documentation and resources available	• PyTorch is a relatively new framework but adopting rapidly
• Uses static computational graph architecture	• Uses dynamic computational graph architecture
• It offers Tensorboard, a monitoring for model training process and visualization	• Does not have any monitoring and visualization interfaces
• "define and run" mode makes debugging very difficult	• "define-by-run mode is more like traditional programming, and you can use common debugging tools as pdb, ipdb or PyCharm debugger

We can clearly see PyTorch out performs TensorFlow in some areas.

Core Components of PyTorch

1.Tensors

Before we get into 'Tensors', lets see what data dimensions we have.



Data Dimensions

Term	Description	Example	Rank	Dimensions
Scalers	Zero-dimensional Tensor like age, height etc.	5, 3	0 (always)	()
Vectors	One-dimensional Tensor. It stores list of values	[5,4]	1 (always)	(2,)
Matrices	Two-dimensional Tensor. It stores grid of values	[[1,2,3], [4,5,6]]	2 (always)	(2,3)
Tensors	Two + dimensional Tensor. It stores collection of values	[[[1,2,3],[4,5,6]], [[7,8,9],[10,11,12]]]	3 (always)	(2,2,3)

Every object we make (vectors, matrices, tensors) eventually stores scalars

Going back to tensors...

- Tensor as noted above is a multi-dimensional matrix which contain elements of a single data type. These are part of the PyTorch's torch package.

- PyTorch's 'torch package' provides a flexible N-dimensional array or Tensor, which supports basic routines for indexing, slicing, transposing, type-casting, resizing, sharing storage and cloning. The Tensor also supports mathematical operations like max, min, sum, statistical distributions like uniform, normal and multinomial, and BLAS operations like dot product, matrix-vector multiplication, matrix-matrix multiplication, matrix-vector product and matrix product.

So why do we need to use PyTorch tensors when we have Numpy ndarrays?

One good reason is, PyTorch can utilize power of GPU's to accelerate numerical calculations. Otherwise both are conceptually identical. Tensors in PyTorch and arrays in numpy will share their underlying memory locations, and changing one will change the other. We can convert a Tensor to array and vice versa very easily.

2. Computational Graphs

Computational graphs are another core component of PyTorch deep learning framework. It states the sequence of operations that occur between the variables. These chain comp in the neural network will result in output prediction. Neural networks are heavily dependent on computational graph during the training process. It helps to calculate the gradient (which is all the partial derivatives of error function with respect to weights) by efficiently applying the chain rule.

PyTorch uses dynamic computational graph (as opposed to conventional static computational graph) to calculate the gradient during back propagation. Here are the differences between static and dynamic computational graphs

No alt text provided for this image

Here is the example of computational graphs (by PyTorch.org) to show how back propagation works using PyTorch's autograd package. This brings us to next topic — "**Autograd**"

No alt text provided for this image

3. AUTOGRAD

PyTorch's Autograd package provides reverse automatic differentiation (back propagation) for all operations on Tensors to calculate the gradient. Since, PyTorch follows "define by run" dynamic framework, back propagation will run the desired computation as opposed to specifying a static graph structure.

PyTorch Packages

Here are some of the PyTorch packages that are used in creating a neural network

No alt text provided for this image

Steps to create neural network

Here are the high level steps involved in creating neural network in PyTorch (for overview on deep learning / neural networks check this link)

1. Convert the input data into torch tensors

No alt text provided for this image

2. Create a model with sequence of input, output and hidden layers and perform forward propagation to generate predicted output
3. Compare predicted output with actual
4. Calculate error (difference between predicted vs actual)
5. Perform gradient decent using back propagation with PyTorch's Autograd to spread the error until the model gets better output
6. Check the performance of the model with different metrics.

PyTorch Installation

Messaging

No alt text provided for this image

- For installation go to https://pytorch.org/
- It has option of installing locally or on cloud
- Based on your system configuration, you can select options in interactive option selector
- Once you select the preferences, you will get the command to run
- As a prerequisite, you need to have access to a system with Python, NumPy installed

Sources and Interesting articles:

- Quick Introduction to Deep Learning
- About Torch package
- PyTorch Documentation
- Pytorch dynamic computational graphs
- Automatic differentiation in PyTorch
- Exploring PyTorch

Shashi Chilukuri
Data Scientist who is passionate about Artificial Intelligence & solving business problems with it

Messaging