Fast.ai and PyTorch

Sachin Satish Kumar

April 27, 2022

1 Introduction

Fast.ai is a machine learning repository that gives professionals elevated platforms for efficiently and simply delivering next-generation results in traditional neural network contexts, as well as minimal parameters which can be perfectly matched to generate new strategies. However, we will be discussing the library/framework in which Fast.ai operates on.

In October 2016, Facebook released PyTorch, a computer vision system. It's highly customizable, and it's built on the Torch repository. PyTorch is a neural network based application framework that offers a lot of versatility and power. Fast.ai is an added component of PyTorch that adds a wide selection of features to the existing algorithm, such as visual analytics approaches, increased. ways to upload and break information, and so on.

Today, we will be going over different aspects of the Pytorch library and what it has to offer! Furthermore, the syntax and installation of Pytorch and Fast.ai, as well as discussing comparisons between other libraries such as Keras and Tensorflow!

2 Pytorch

2.1 Overview and Syntax

PyTorch allows artificial intelligence to become simpler for Python users and includes helpful applications such as OOP assistance and adaptive computational diagrams. It goes beyond just acting as a normal Python program, as it can use simple Python syntax but implement different machine learning algorithms from its library.

Now onto syntax, there are a variety of syntax. For this lecture today, I will be going over the essential syntax needed plus additional visual aids presented. Such syntax will include tensors, neural networks, and computational graphs.

2.1.1 Tensors

Tensors in PyTorch are multifaceted array parameters that form the basis for all conventional learning in this library. Tensors, unlike normal nominal models, can be programmed to use either CPU or GPU to accelerate functions.

There are 5 data types used to initialize tensors in PyTorch listed below:

CPU tensor	Data Type
FloatTensor	32-bit float
DoubleTensor	64-bit float
HalfTensor	16-bit float
IntTensor	32-bit int
LongTensor	64-bit int

```
import torch
# initializing tensors
a = torch.tensor(2)
b = torch.tensor(1)
```

Figure 1: Displays initialization of variables

```
# addition
print(a+b)
# subtraction
print(b-a)
# multiplication
print(a*b)
# division
print(a/b)
```

Figure 2: Explains functions that may be used in PyTorch

```
if torch.cuda.is_available():
    x = x.cuda()
    y = y.cuda()
    x + y
```

Figure 3: Moving tensors to be handled by the GPU

```
ones_tensor = torch.ones((2, 2)) # tensor containing all ones
rand_tensor = torch.rand((2, 2)) # tensor containing random va
lues
```

Figure 4: Displaying matrix capabilities

2.2 Neural Networks

Because of its outstanding prediction model, such as image analysis and computational models,PyTorch is often used to construct the common convolutional neural networks (CNN). The **torch.nn** framework in PyTorch is used to construct neural networks, and it includes a series of nodes to describe every layer of the CNN. Each function receives input tensors and computes output tensors, which will then be combined to form the system. Loss functions are also specified in the torch.nn bundle, that we use to develop algorithms for training CNNs. The following are the steps in creating a neural network:

- Structure: Construct neural network frameworks, configure constraints, and assign biases.
- Forward Propagation: Utilizing variables, determine the total efficiency. By comparing the expected and real results, you can calculate the deviation from the accuracy.
- Back-propagation: Upon identifying the fault, calculate the mistake function's derivative in view with their neural network's variables. We will change some feature weights using reverse propagation.
- Recursive Automation: Use optimization techniques that change criteria by regression using gradient descent to minimize accuracy loss.

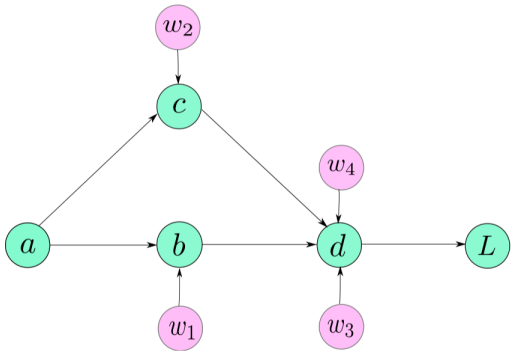

Figure 5: Graph of a simple neural network

2.3 Computational Graphs

It's crucial to experiment with computational diagrams in terms of understanding PyTorch and neural networks specifically CNNs. These plots display basically a concise summary of neural nets with a series of pro that show why the input affects the outcome of the network.

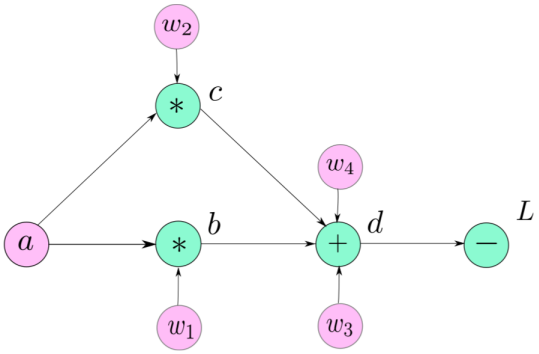

Figure 6: Computational diagram of neural network shown above

2.4 Optimizers

Optimizers assist in reducing loss by iteratively updating biases within the model. This leads to making changes to your model despite attempting to recreate it entirely.

The **torch.optim** kit contains all PyTorch optimization techniques, each of which is configured to be helpful in a particular scenario. By forwarding a collection of parameters to the **torch.optim** function, you can construct an arbitrary application developers. PyTorch has a large number of optimization methods to pick from, so you'll also often find one which suit your requirements.

```
import torch.optim as optim
params = torch.tensor([1.0, 0.0], requires_grad=True)
learning_rate = 1e-3
## SGD
optimizer = optim.SGD([params], lr=learning_rate)
```

Figure 7: Observing the initialization of optimizers

2.5 PyTorch and Competing Libraries

There are some libraries out there that are head-to-head with the new and upcoming library, PyTorch. A few of these libraries is termed Keras and Tensorflow, but the library we will be discussing is Keras due to its high-level API . Keras is a pleasant, efficient application for addressing computer vision issues, with an emphasis on advanced machine learning. It offers basic simplifications and basic structure for designing and exporting high-iteration-rate deep learning algorithms.

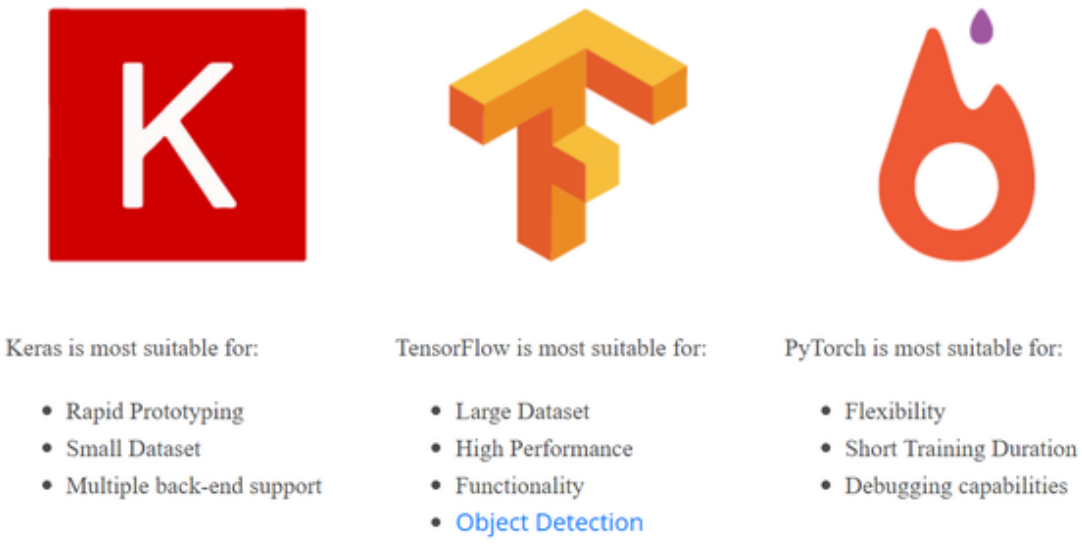

Figure 8: Analyzing differences of machine learning libraries

2.6 Applications of PyTorch

PyTorch is a machine learning platform that is accessible and is used to incorporate network architectures such as RNN, CNN, LSTM, as well as other elevated protocols. Furthermore, it is used by

academics, businesses, and ML and AI groups.

Companies Currently Using PyTorch

Download CSV Sample (25 companies)

COMPANY NAME	WEBSITE	HQ ADDRESS	CITY	STATE	ZIP	COUNTRY	TOP LEVEL INDUSTRY	SUB LEVEL INDUSTRY
JPMorgan Chase	jpmorganchase.com	383 Madison Ave	New York	NY	10179-0...	US	Finance	Banking
Comcast	xfinity.com	One Comcast Center	Philadelphia	PA	19103	US	Telecommunications	Telephony & Wireless
USAA	usaa.com	9800 Fredericksbur...	San Antonio	TX	78288	US	Insurance	Insurance
Amgen	amgen.com	One Amgen Center ...	Thousand Oaks	CA	91320-1...	US	Healthcare	Biotechnology
IBM	ibm.com	New Orchard Road	Armonk	NY	10504	US	Technical	Diversified Technology, Products & Serv
SparkCognition	sparkcognition.com	4030 West Braker L...	Austin	TX	78759	US	Technical	Software Manufacturers

Figure 9: Presenting top companies that utilize PyTorch

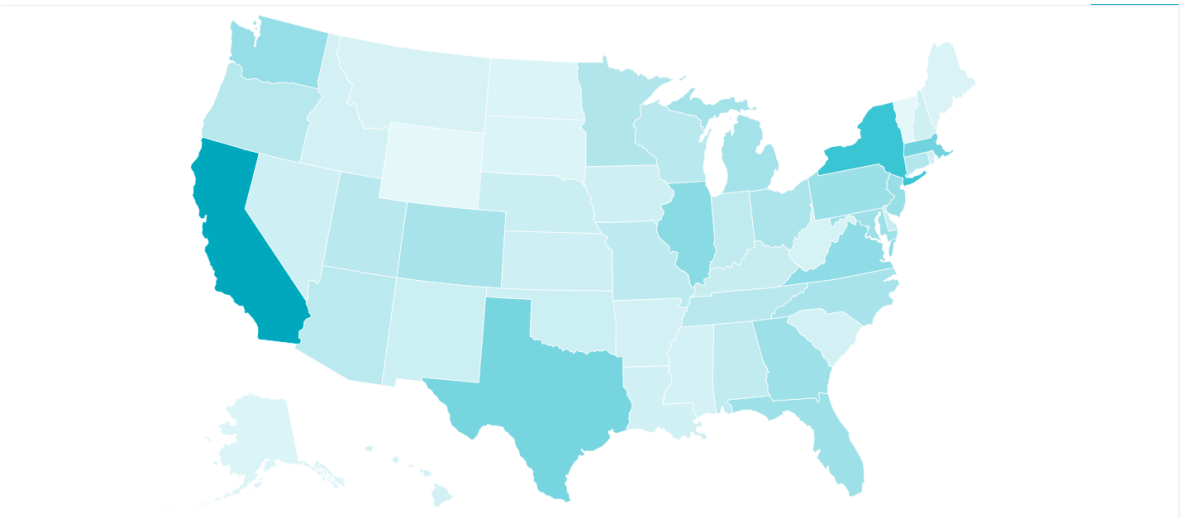

Figure 10: Illustration of a map of what states use PyTorch frequently

2.7 Installation process

2.7.1 Windows

For the installation process:
1. Install Anaconda
2. Open Anaconda Prompt (NOT Anaconda Navigator)
3. Type in the prompt: conda install pytorch -c pytorch
4. Type in the prompt: pip install torchvision
5. Finally, Add environment to ipykernel

2.7.2 Mac

1. Open Command Prompt for Mac
2. Dependent on your Python edition, choose one of the following two commands to install PyTorch using pip:
- Python 3.x: pip3 install torch torchvision

```
import torch
x = torch.rand(5, 3)
print(x)
```

Figure 11: Example code to test installation successsion

```
tensor([[0.3380, 0.3845, 0.3217],
        [0.8337, 0.9050, 0.2650],
        [0.2979, 0.7141, 0.9069],
        [0.1449, 0.1132, 0.1375],
        [0.4675, 0.3947, 0.1426]])
```

Figure 12: Output of example code above

3 References

fastai-A Layered API for Deep Learning. fastai-A Layered API for Deep Learning ·. (n.d.). https://www.fast.ai/2020/02/13
A-Layered-API-for-Deep-Learning/.

Kathuria, A. (2021, April 9). PyTorch Basics: Understanding Autograd and Computation Graphs.
Paperspace Blog. https://blog.paperspace.com/pytorch-101-understanding-graphs-and-automatic-differentiation/.

Learn the Basics¶. Learn the Basics - PyTorch Tutorials 1.8.1+cu102 documentation.(n.d.).https://pytorch.org/tutorials

PyTorch tutorial: a quick guide for new learners. Educative. (n.d.). https://www.educative.io/blog/pytorch-
tutorial.