PyTorch

PyTorch is a machine learning framework based on the Torch library,[4][5][6] used for applications such as computer vision and natural language processing,[7] originally developed by Meta AI and now part of the Linux Foundation umbrella.[8][9][10][11] It is free and open-source software released under the modified BSD license. Although the Python interface is more polished and the primary focus of development, PyTorch also has a C++ interface.[12]

A number of pieces of deep learning software are built on top of PyTorch, including Tesla Autopilot,[13] Uber's Pyro,[14] Hugging Face's Transformers,[15] PyTorch Lightning,[16][17] and Catalyst.[18][19]

PyTorch provides two high-level features:[20]

- Tensor computing (like NumPy) with strong acceleration via graphics processing units (GPU)
- Deep neural networks built on a tape-based automatic differentiation system

PyTorch

Original author(s)	Adam Paszke Sam Gross Soumith Chintala Gregory Chanan
Developer(s)	Meta AI
Initial release	September 2016[1]
Stable release	2.0[2] / 15 March 2023; 15 March 2023
Repository	github.com/pytorch/pytorch (https://github.com/pytorch/pytorch)
Written in	Python C++ CUDA
Operating system	Linux macOS Windows
Platform	IA-32, x86-64
Available in	English
Type	Library for machine learning and deep learning
License	BSD-3[3]
Website	pytorch.org (https://pytorch.org/)

History

Meta (formerly known as Facebook) operates both *PyTorch* and *Convolutional Architecture for Fast Feature Embedding* (Caffe2), but models defined by the two frameworks were mutually incompatible. The Open Neural Network Exchange (ONNX) project was created by Meta and Microsoft in September 2017 for converting models between frameworks. Caffe2 was merged into PyTorch at the end of March 2018.[21] In September 2022, Meta announced that *PyTorch* would be governed by PyTorch Foundation, a newly created independent organization – a subsidiary of Linux Foundation.[22]

PyTorch 2.0 has been released on 15 March 2023.[23]

PyTorch tensors

PyTorch defines a class called Tensor (`torch.Tensor`) to store and operate on homogeneous multidimensional rectangular arrays of numbers. PyTorch Tensors are similar to NumPy Arrays, but can also be operated on a CUDA-capable NVIDIA GPU. PyTorch has also been developing support for other GPU platforms, for example, AMD's ROCm and Apple's Metal Framework.[24]

PyTorch supports various sub-types of Tensors.[25]

Note that the term "tensor" here does not carry the same meaning as tensor in mathematics or physics. The meaning of the word in those areas, that is, a certain kind of object in linear algebra, is only tangentially related to the one in Machine Learning.

Modules

Autograd module

PyTorch uses a method called automatic differentiation. A recorder records what operations have performed, and then it replays it backward to compute the gradients. This method is especially powerful when building neural networks to save time on one epoch by calculating differentiation of the parameters at the forward pass.

`Optim` module

`torch.optim` is a module that implements various optimization algorithms used for building neural networks. Most of the commonly used methods are already supported, so there is no need to build them from scratch.

`nn` module

PyTorch autograd makes it easy to define computational graphs and take gradients, but raw autograd can be a bit too low-level for defining complex neural networks. This is where the `nn` module can help. The `nn` module provides layers and tools to easily create a neural networks by just defining the layers of the network.

PyTorch also contains many other useful submodules such as data loading utilities and distributed training functions.

Example

The following program shows the low-level functionality of the library with a simple example

```
import torch
dtype = torch.float
device = torch.device("cpu") # This executes all calculations on the CPU
# device = torch.device("cuda:0") # This executes all calculations on the GPU

# Creation of a tensor and filling of a tensor with random numbers
a = torch.randn(2, 3, device=device, dtype=dtype)
print(a) # Output of tensor A
# Output: tensor([[-1.1884,  0.8498, -1.7129],
#                 [-0.8816,  0.1944,  0.5847]])

# Creation of a tensor and filling of a tensor with random numbers
b = torch.randn(2, 3, device=device, dtype=dtype)
print(b) # Output of tensor B
# Output: tensor([[ 0.7178, -0.8453, -1.3403],
#                 [ 1.3262,  1.1512, -1.7070]])

print(a*b) # Output of a multiplication of the two tensors
# Output: tensor([[-0.8530, -0.7183,  2.58],
```

```
#                  [-1.1692,  0.2238, -0.9981]])

print(a.sum()) # Output of the sum of all elements in tensor A
# Output: tensor(-2.1540)

print(a[1,2]) # Output of the element in the third column of the second row (zero based)
# Output: tensor(0.5847)

print(a.max()) # Output of the maximum value in tensor A
# Output: tensor(-1.7129)
```

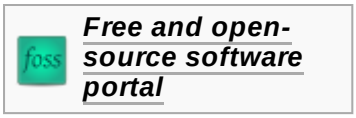
The following code-block shows an example of the higher level functionality provided `nn` module. A neural network with linear layers is defined in the example.

```
import torch
from torch import nn # Import the nn sub-module from PyTorch

class NeuralNetwork(nn.Module): # Neural networks are defined as classes
    def __init__(self): # Layers and variables are defined in the __init__ method
        super(NeuralNetwork, self).__init__() # Must be in every network.
        self.flatten = nn.Flatten() # Defining a flattening layer.
        self.linear_relu_stack = nn.Sequential( # Defining a stack of layers.
            nn.Linear(28*28, 512), # Linear Layers have an input and output shape
            nn.ReLU(), # ReLU is one of many activation functions provided by nn
            nn.Linear(512, 512),
            nn.ReLU(),
            nn.Linear(512, 10),
        )

    def forward(self, x): # This function defines the forward pass.
        x = self.flatten(x)
        logits = self.linear_relu_stack(x)
        return logits
```

See also



- Comparison of deep learning software
- Differentiable programming
- DeepSpeed
- Torch (machine learning)

References

1. Chintala, Soumith (1 September 2016). "PyTorch Alpha-1 release" (https://github.com/pytorch/pytorch/releases/tag/v0.1.1).
2. "PyTorch 2.0: Our next generation release that is faster, more Pythonic and Dynamic as ever" (https://github.com/pytorch/pytorch/releases/tag/v2.0.0). Retrieved 15 March 2023.
3. Claburn, Thomas (12 September 2022). "PyTorch gets lit under The Linux Foundation" (https://www.theregister.com/2022/09/12/pytorch_meta_linux_foundation/). *The Register*.
4. Yegulalp, Serdar (19 January 2017). "Facebook brings GPU-powered machine learning to Python" (https://www.infoworld.com/article/3159120/artificial-intelligence/facebook-brings-gpu-powered-machine-learning-to-python.html). *InfoWorld*. Retrieved 11 December 2017.

5. Lorica, Ben (3 August 2017). "Why AI and machine learning researchers are beginning to embrace PyTorch" (https://www.oreilly.com/ideas/why-ai-and-machine-learning-researchers-are-beginning-to-embrace-pytorch). O'Reilly Media. Retrieved 11 December 2017.
6. Ketkar, Nikhil (2017). "Introduction to PyTorch". *Deep Learning with Python*. Apress, Berkeley, CA. pp. 195–208. doi:10.1007/978-1-4842-2766-4_12 (https://doi.org/10.1007%2F978-1-4842-2766-4_12). ISBN 9781484227657.
7. "Natural Language Processing (NLP) with PyTorch – NLP with PyTorch documentation" (http://dl4nlp.info/en/latest/). *dl4nlp.info*. Retrieved 2017-12-18.
8. Patel, Mo (2017-12-07). "When two trends fuse: PyTorch and recommender systems" (https://www.oreilly.com/ideas/when-two-trends-fuse-pytorch-and-recommender-systems). *O'Reilly Media*. Retrieved 2017-12-18.
9. Mannes, John. "Facebook and Microsoft collaborate to simplify conversions from PyTorch to Caffe2" (https://techcrunch.com/2017/09/07/facebook-and-microsoft-collaborate-to-simplify-conversions-from-pytorch-to-caffe2/). *TechCrunch*. Retrieved 2017-12-18. "FAIR is accustomed to working with PyTorch – a deep learning framework optimized for achieving state of the art results in research, regardless of resource constraints. Unfortunately in the real world, most of us are limited by the computational capabilities of our smartphones and computers."
10. Arakelyan, Sophia (2017-11-29). "Tech giants are using open source frameworks to dominate the AI community" (https://venturebeat.com/2017/11/29/tech-giants-are-using-open-source-frameworks-to-dominate-the-ai-community/). *VentureBeat*. Retrieved 2017-12-18.
11. "PyTorch strengthens its governance by joining the Linux Foundation" (https://pytorch.org/blog/PyTorchfoundation/). *pytorch.org*. Retrieved 2022-09-13.
12. "The C++ Frontend" (https://pytorch.org/cppdocs/frontend.html). *PyTorch Master Documentation*. Retrieved 2019-07-29.
13. Karpathy, Andrej. "PyTorch at Tesla - Andrej Karpathy, Tesla" (https://www.youtube.com/watch?v=oBklltKXtDE).
14. "Uber AI Labs Open Sources Pyro, a Deep Probabilistic Programming Language" (https://eng.uber.com/pyro/). *Uber Engineering Blog*. 2017-11-03. Retrieved 2017-12-18.
15. *PYTORCH-TRANSFORMERS: PyTorch implementations of popular NLP Transformers* (https://pytorch.org/hub/huggingface_pytorch-transformers/), PyTorch Hub, 2019-12-01, retrieved 2019-12-01
16. *PYTORCH-Lightning: The lightweight PyTorch wrapper for ML researchers. Scale your models. Write less boilerplate* (https://github.com/PyTorchLightning/pytorch-lightning/), Lightning-Team, 2020-06-18, retrieved 2020-06-18
17. "Ecosystem Tools" (https://pytorch.org/ecosystem/). *pytorch.org*. Retrieved 2020-06-18.
18. *GitHub - catalyst-team/catalyst: Accelerated DL & RL* (https://github.com/catalyst-team/catalyst), Catalyst-Team, 2019-12-05, retrieved 2019-12-05
19. "Ecosystem Tools" (https://pytorch.org/ecosystem/). *pytorch.org*. Retrieved 2020-04-04.
20. "PyTorch – About" (https://web.archive.org/web/20180615190804/https://pytorch.org/about/). *pytorch.org*. Archived from the original (https://pytorch.org/about/) on 2018-06-15. Retrieved 2018-06-11.
21. "Caffe2 Merges With PyTorch" (https://medium.com/@Synced/caffe2-merges-with-pytorch-a89c70ad9eb7). 2018-04-02.
22. Edwards, Benj (2022-09-12). "Meta spins off PyTorch Foundation to make AI framework vendor neutral" (https://arstechnica.com/information-technology/2022/09/meta-spins-off-pytorch-foundation-to-make-ai-framework-vendor-neutral/). *Ars Technica*.

23. "PyTorch 2.0 brings new fire to open-source machine learning" (https://venturebeat.com/ai/pytorch-2-0-brings-new-fire-to-open-source-machine-learning/). *VentureBeat*. 15 March 2023. Retrieved 16 March 2023.
24. "Introducing Accelerated PyTorch Training on Mac" (https://pytorch.org/blog/introducing-accelerated-pytorch-training-on-mac/). *pytorch.org*. Retrieved 2022-06-04.
25. "An Introduction to PyTorch – A Simple yet Powerful Deep Learning Library" (https://www.analyticsvidhya.com/blog/2018/02/pytorch-tutorial/). *analyticsvidhya.com*. 2018-02-22. Retrieved 2018-06-11.

External links

- Official website (https://pytorch.org)

Retrieved from "https://en.wikipedia.org/w/index.php?title=PyTorch&oldid=1144867403"