PyTorch
What, why and how?

@shagunsodhani

FACEBOOK AI

Questions are welcome at all times :)

Agenda

1. PyTorch Framework
2. How to get started
3. PyTorch Ecosystem
4. PyTorch for production

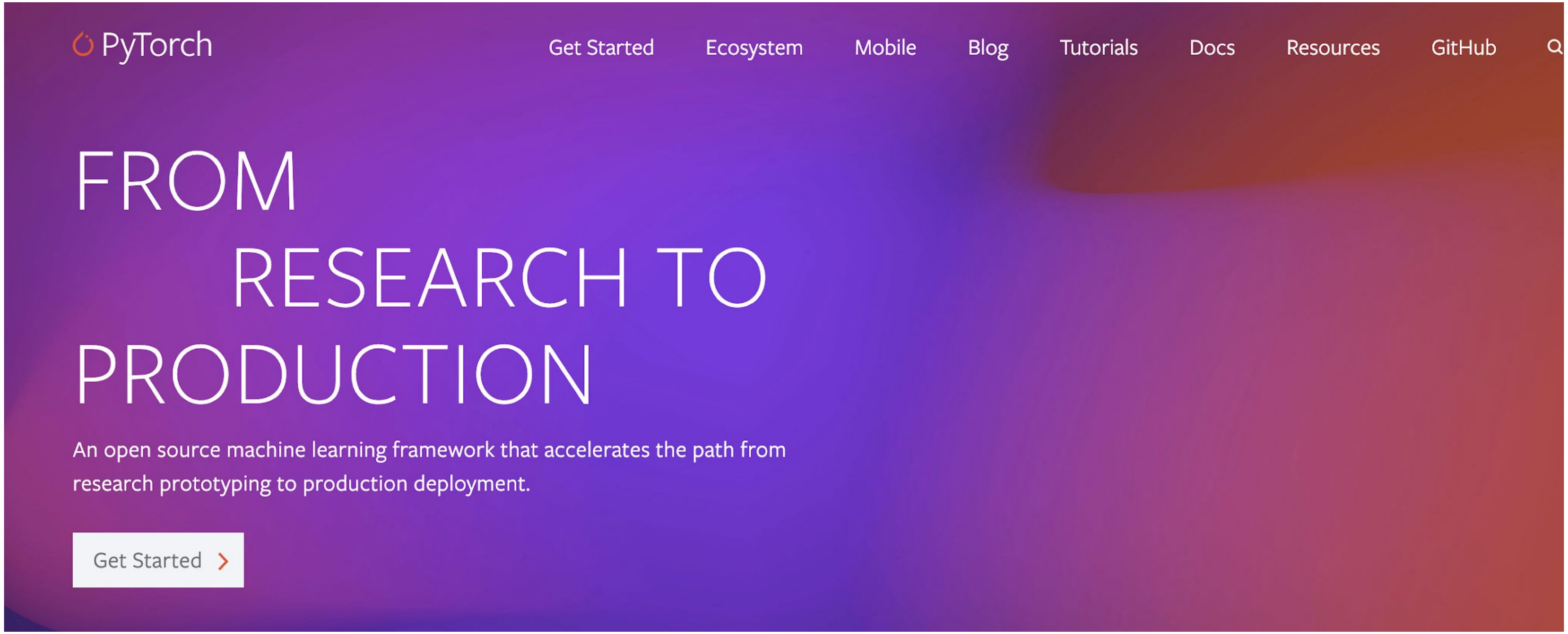
Questions are welcome at all times :)

Agenda

1. **PyTorch Framework**
2. How to get started
3. PyTorch Ecosystem
4. PyTorch for production

Questions are welcome at all times :)

https://pytorch.org



FROM RESEARCH TO PRODUCTION

An open source machine learning framework that accelerates the path from research prototyping to production deployment.

Taken from: https://pytorch.org/

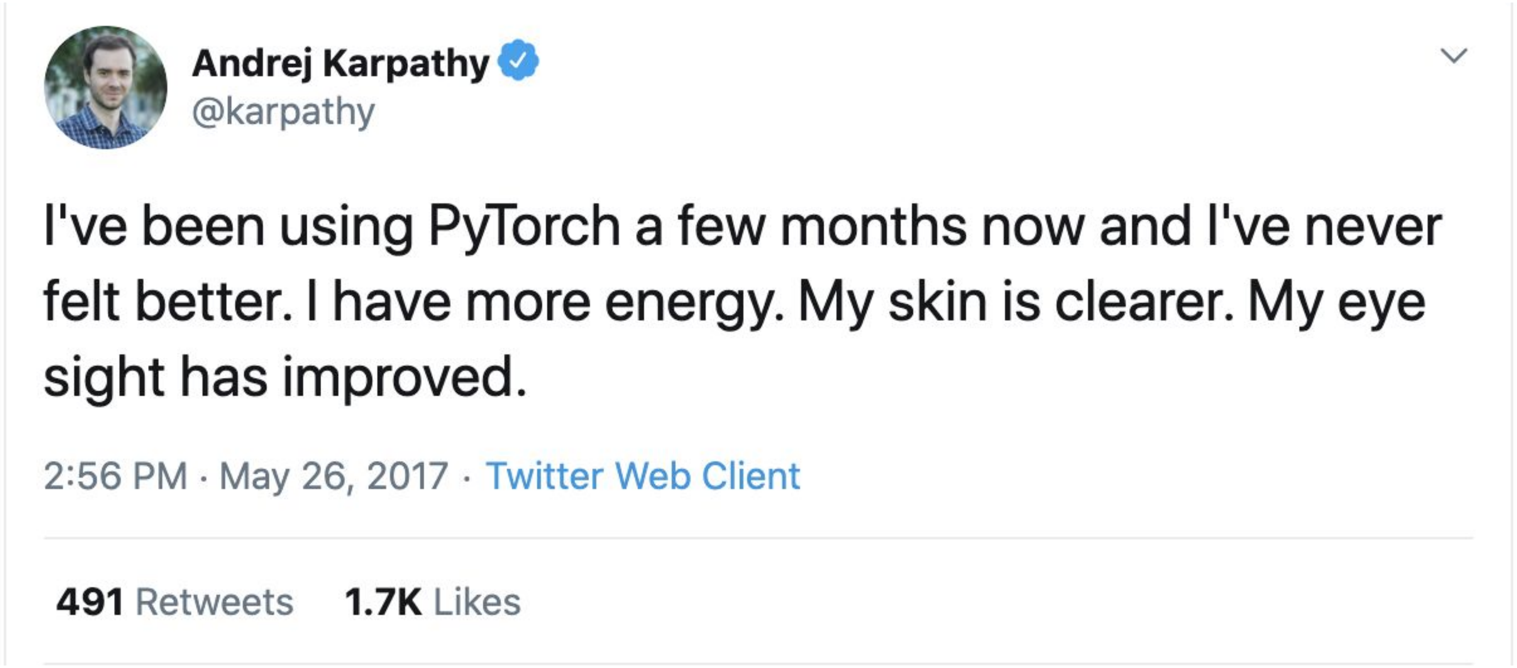
What is PyTorch

1. Open-source Machine Learning framework

2. Provides Numpy-like arrays with GPU acceleration

3. Enables training deep neural networks

Ease of Use



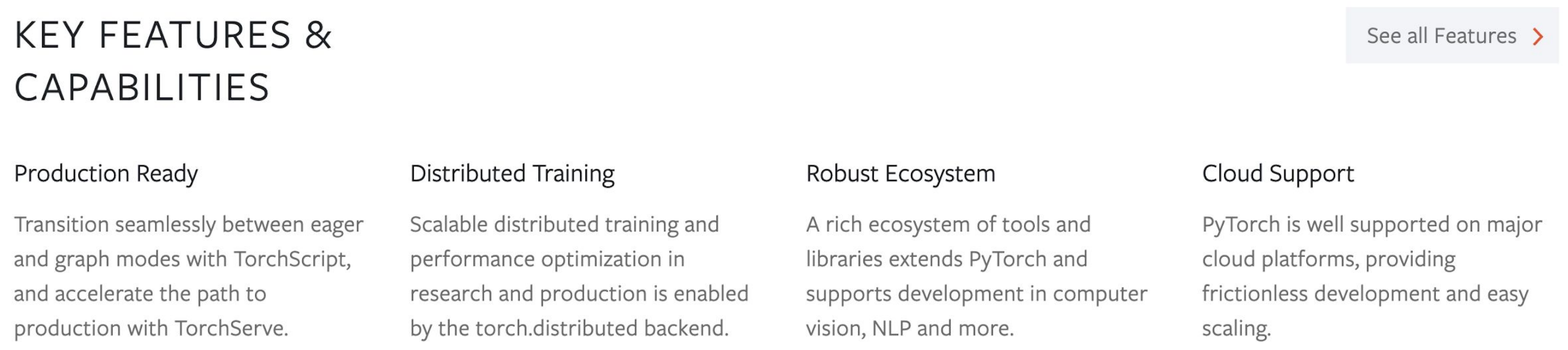
Andrej Karpathy
@karpathy

I've been using PyTorch a few months now and I've never felt better. I have more energy. My skin is clearer. My eye sight has improved.

2:56 PM · May 26, 2017 · Twitter Web Client

491 Retweets 1.7K Likes

More than just neural networks



KEY FEATURES & CAPABILITIES

Production Ready

Transition seamlessly between eager and graph modes with TorchScript, and accelerate the path to production with TorchServe.

Distributed Training

Scalable distributed training and performance optimization in research and production is enabled by the torch.distributed backend.

Robust Ecosystem

A rich ecosystem of tools and libraries extends PyTorch and supports development in computer vision, NLP and more.

Cloud Support

PyTorch is well supported on major cloud platforms, providing frictionless development and easy scaling.

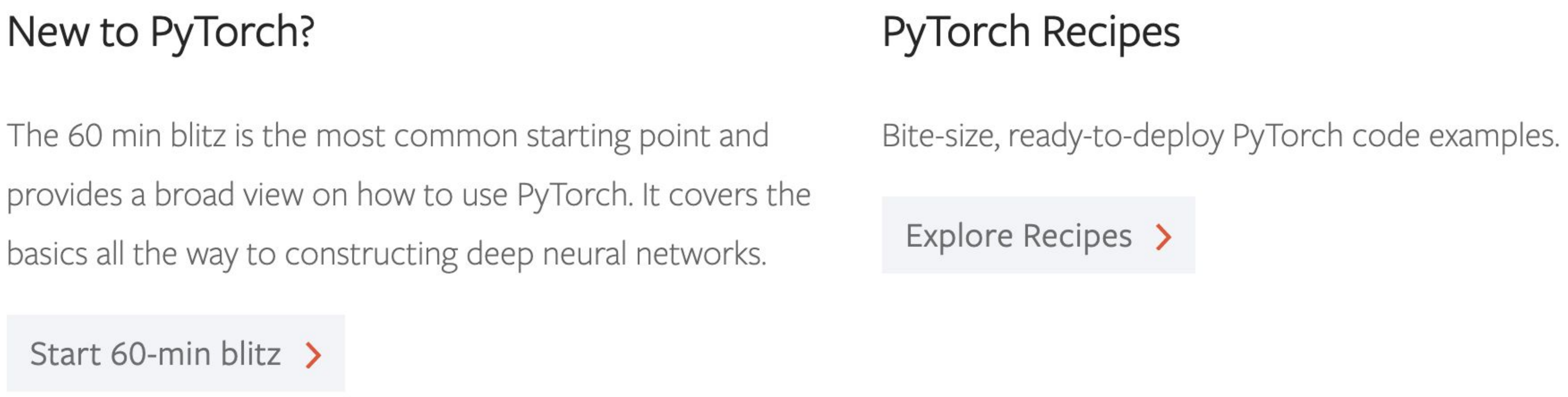
Taken from: https://pytorch.org/

Agenda

1. PyTorch Framework
2. **How to get started**
3. PyTorch Ecosystem
4. PyTorch for production

Questions are welcome at all times :)

PyTorch Tutorials



New to PyTorch?

The 60 min blitz is the most common starting point and provides a broad view on how to use PyTorch. It covers the basics all the way to constructing deep neural networks.

PyTorch Recipes

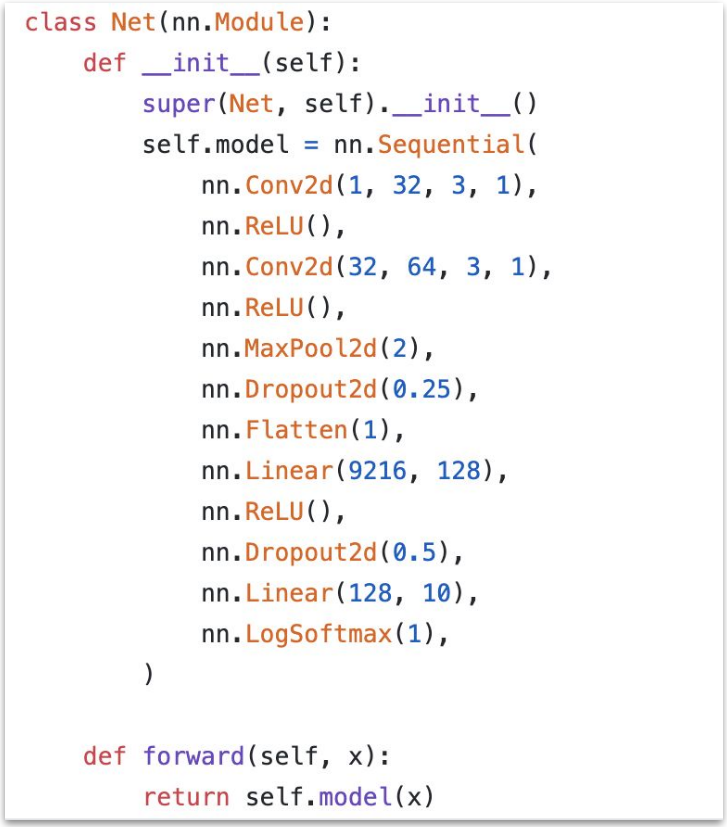
Bite-size, ready-to-deploy PyTorch code examples.

Taken from: https://pytorch.org/

PyTorch Example

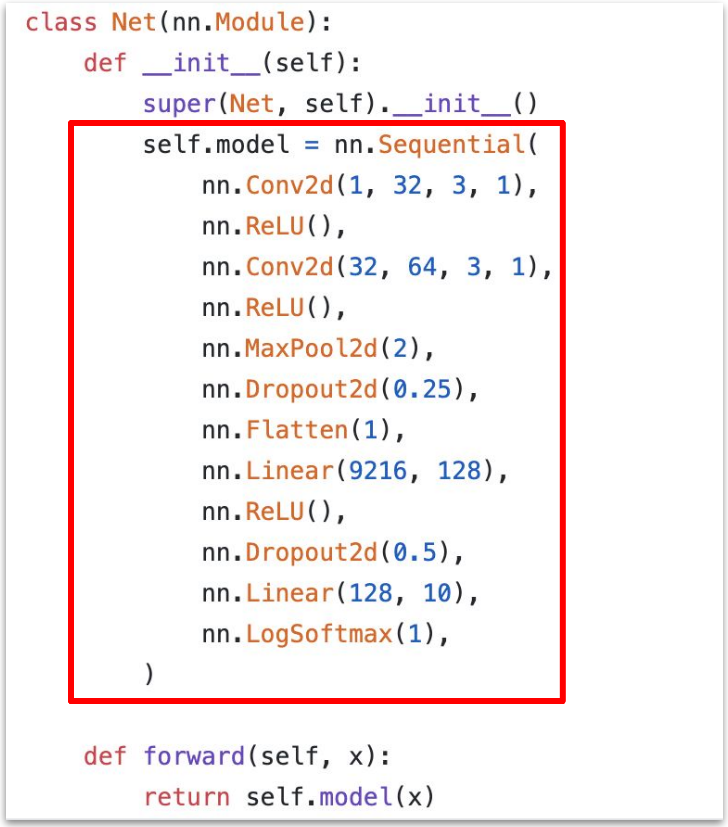
Taken from: https://pytorch.org/

PyTorch Example



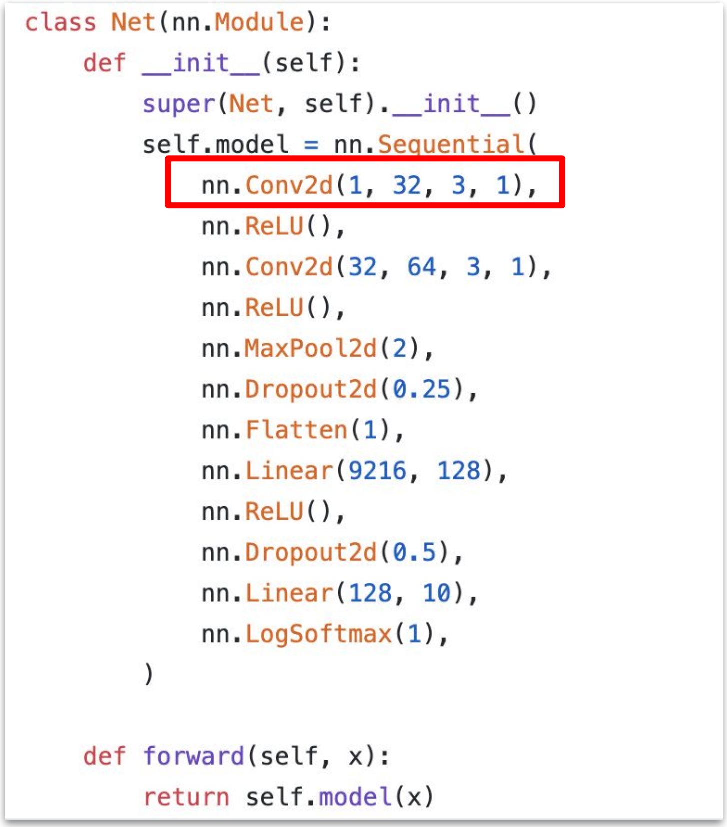
```python
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.model = nn.Sequential(
            nn.Conv2d(1, 32, 3, 1),
            nn.ReLU(),
            nn.Conv2d(32, 64, 3, 1),
            nn.ReLU(),
            nn.MaxPool2d(2),
            nn.Dropout2d(0.25),
            nn.Flatten(1),
            nn.Linear(9216, 128),
            nn.ReLU(),
            nn.Dropout2d(0.5),
            nn.Linear(128, 10),
            nn.LogSoftmax(1),
        )

    def forward(self, x):
        return self.model(x)
```

PyTorch Example



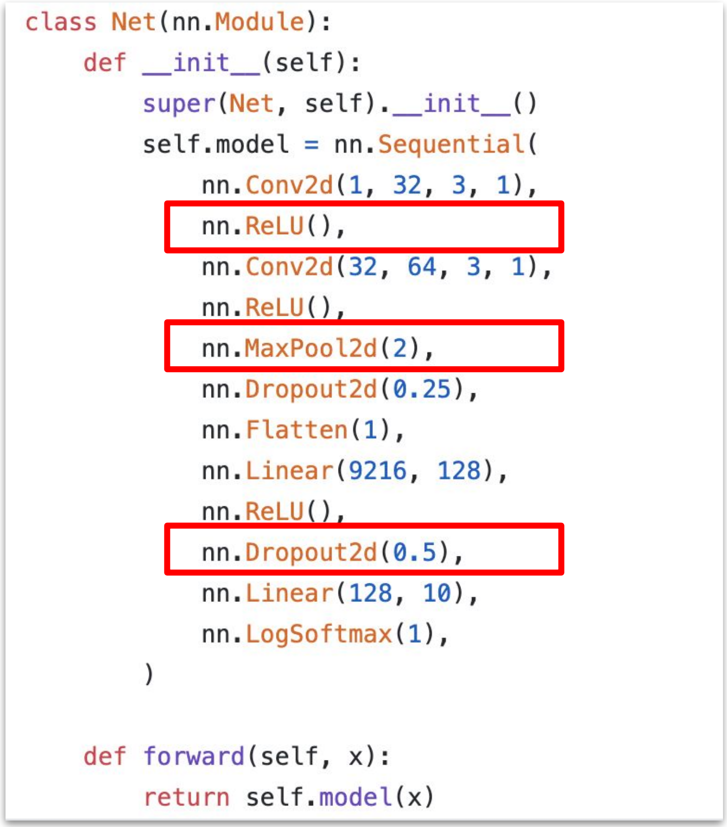
```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.model = nn.Sequential(
            nn.Conv2d(1, 32, 3, 1),
            nn.ReLU(),
            nn.Conv2d(32, 64, 3, 1),
            nn.ReLU(),
            nn.MaxPool2d(2),
            nn.Dropout2d(0.25),
            nn.Flatten(1),
            nn.Linear(9216, 128),
            nn.ReLU(),
            nn.Dropout2d(0.5),
            nn.Linear(128, 10),
            nn.LogSoftmax(1),
        )

    def forward(self, x):
        return self.model(x)
```

PyTorch Example



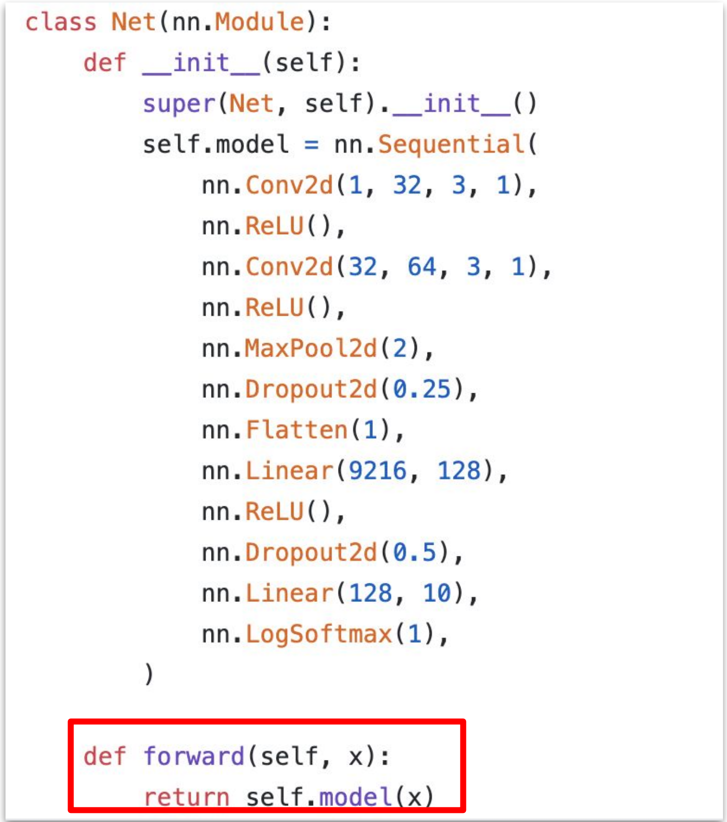
```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.model = nn.Sequential(
            nn.Conv2d(1, 32, 3, 1),
            nn.ReLU(),
            nn.Conv2d(32, 64, 3, 1),
            nn.ReLU(),
            nn.MaxPool2d(2),
            nn.Dropout2d(0.25),
            nn.Flatten(1),
            nn.Linear(9216, 128),
            nn.ReLU(),
            nn.Dropout2d(0.5),
            nn.Linear(128, 10),
            nn.LogSoftmax(1),
        )

    def forward(self, x):
        return self.model(x)
```

PyTorch Example



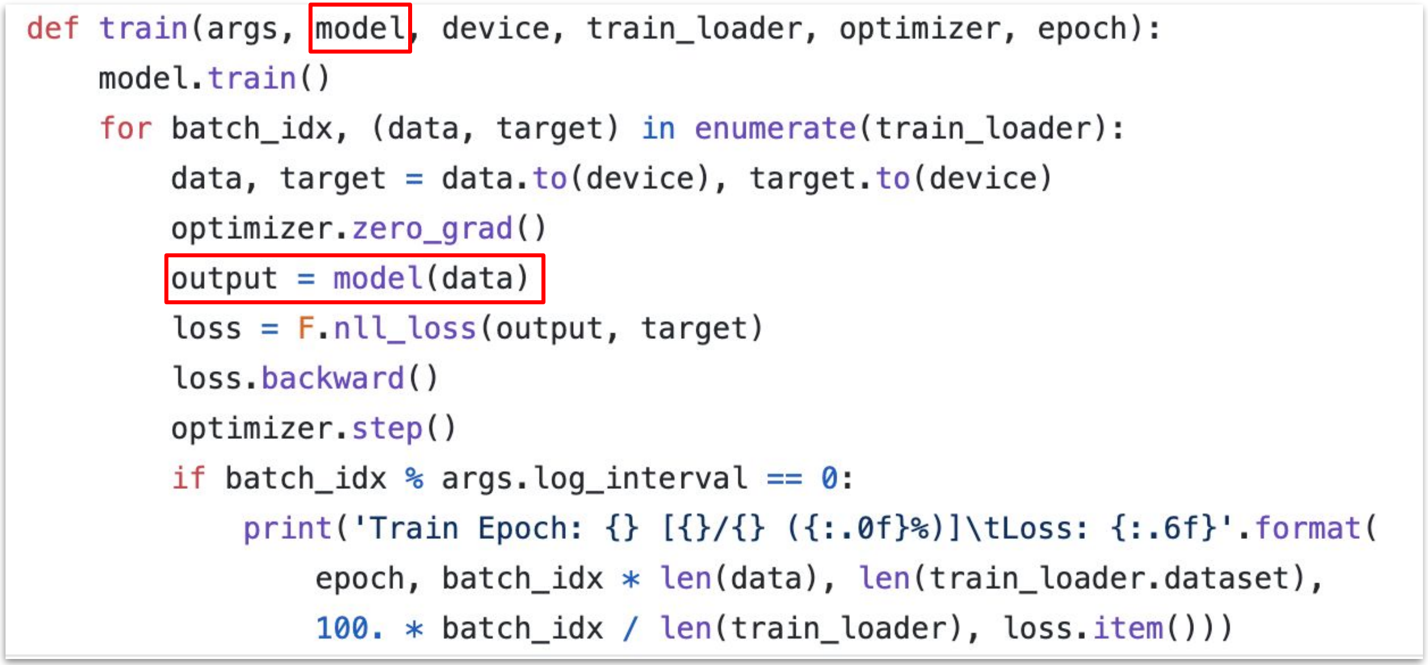
```python
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.model = nn.Sequential(
            nn.Conv2d(1, 32, 3, 1),
            nn.ReLU(),
            nn.Conv2d(32, 64, 3, 1),
            nn.ReLU(),
            nn.MaxPool2d(2),
            nn.Dropout2d(0.25),
            nn.Flatten(1),
            nn.Linear(9216, 128),
            nn.ReLU(),
            nn.Dropout2d(0.5),
            nn.Linear(128, 10),
            nn.LogSoftmax(1),
        )

    def forward(self, x):
        return self.model(x)
```

PyTorch Example



```python
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.model = nn.Sequential(
            nn.Conv2d(1, 32, 3, 1),
            nn.ReLU(),
            nn.Conv2d(32, 64, 3, 1),
            nn.ReLU(),
            nn.MaxPool2d(2),
            nn.Dropout2d(0.25),
            nn.Flatten(1),
            nn.Linear(9216, 128),
            nn.ReLU(),
            nn.Dropout2d(0.5),
            nn.Linear(128, 10),
            nn.LogSoftmax(1),
        )

    def forward(self, x):
        return self.model(x)
```

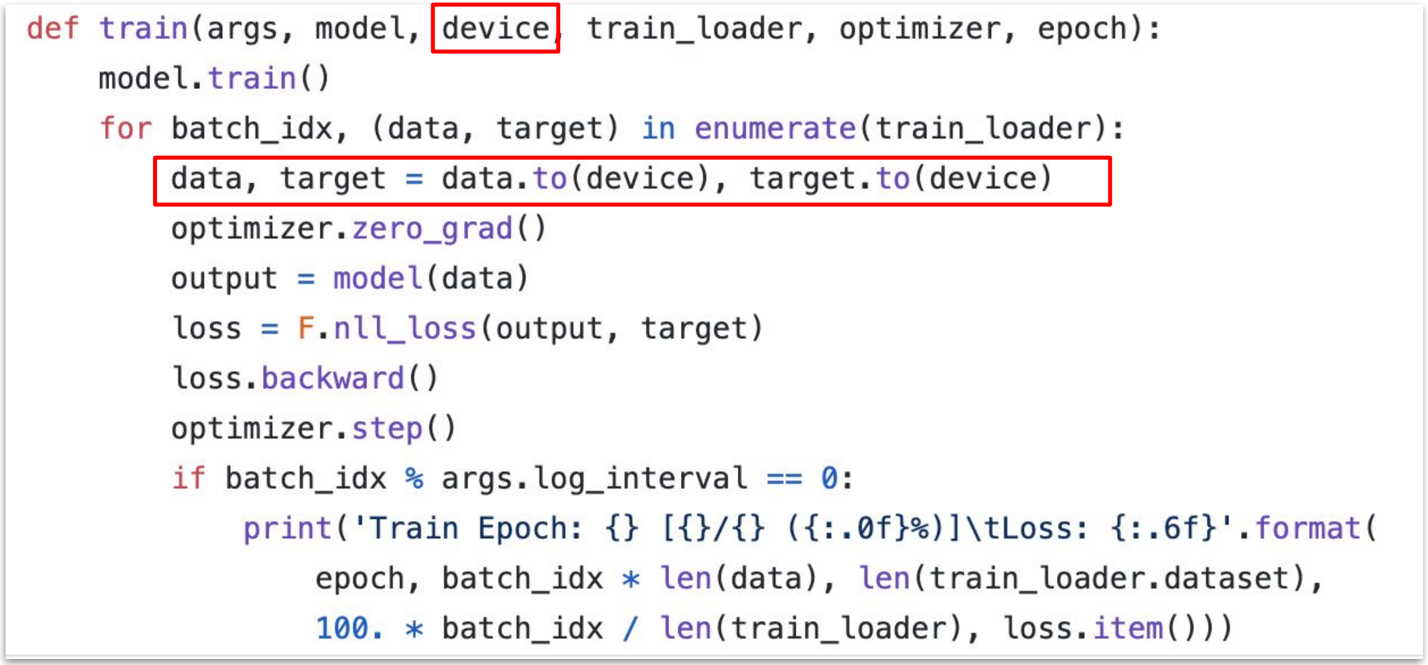
PyTorch Example



```
def train(args, model, device, train_loader, optimizer, epoch):
    model.train()
    for batch_idx, (data, target) in enumerate(train_loader):
        data, target = data.to(device), target.to(device)
        optimizer.zero_grad()
        output = model(data)
        loss = F.nll_loss(output, target)
        loss.backward()
        optimizer.step()
        if batch_idx % args.log_interval == 0:
            print('Train Epoch: {} [{}/{} ({:.0f}%)]\tLoss: {:.6f}'.format(
                epoch, batch_idx * len(data), len(train_loader.dataset),
                100. * batch_idx / len(train_loader), loss.item()))
```

Taken from: https://pytorch.org/

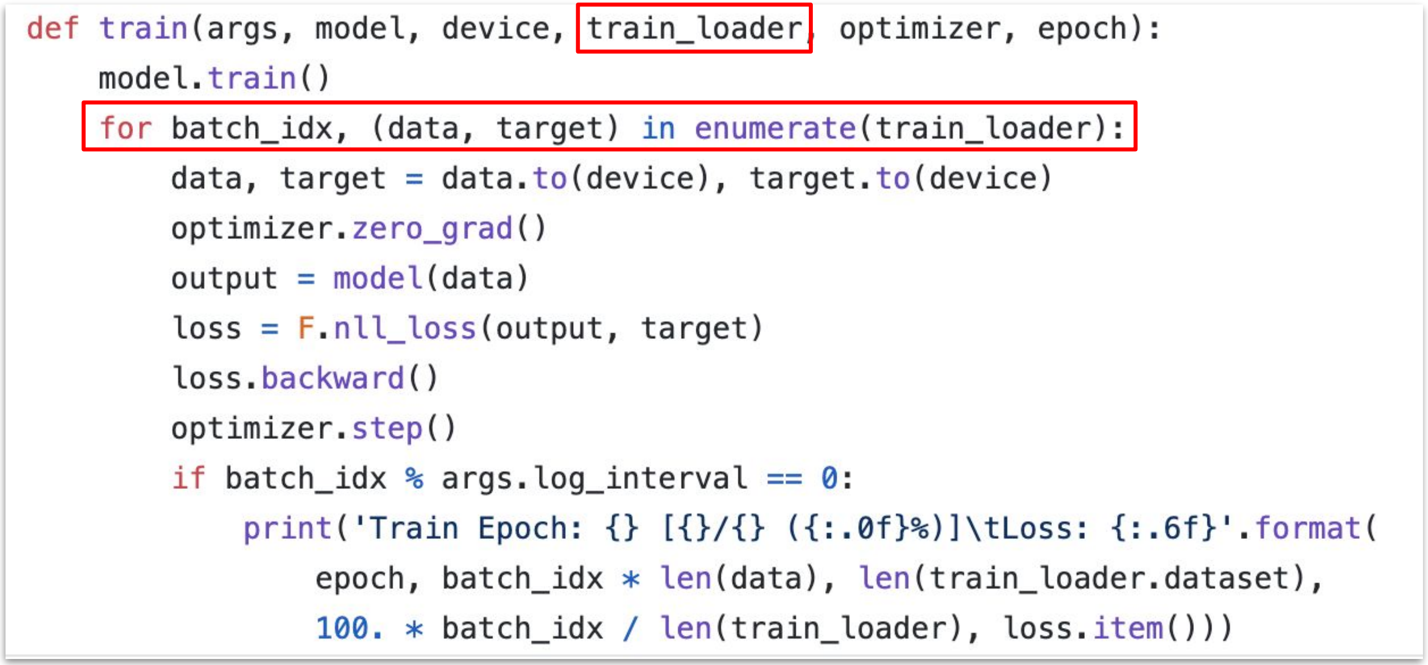
PyTorch Example



```python
def train(args, model, device, train_loader, optimizer, epoch):
    model.train()
    for batch_idx, (data, target) in enumerate(train_loader):
        data, target = data.to(device), target.to(device)
        optimizer.zero_grad()
        output = model(data)
        loss = F.nll_loss(output, target)
        loss.backward()
        optimizer.step()
        if batch_idx % args.log_interval == 0:
            print('Train Epoch: {} [{}/{} ({:.0f}%)]\tLoss: {:.6f}'.format(
                epoch, batch_idx * len(data), len(train_loader.dataset),
                100. * batch_idx / len(train_loader), loss.item()))
```

Taken from: https://pytorch.org/

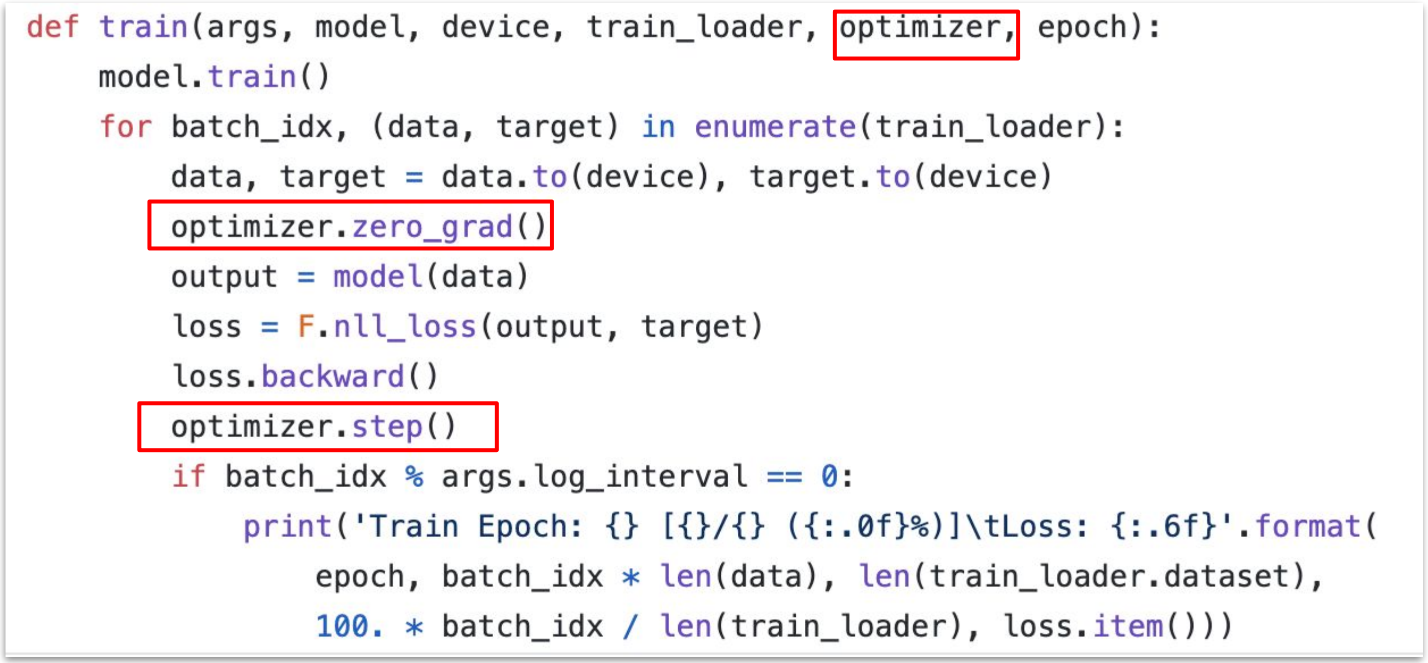
PyTorch Example



```python
def train(args, model, device, train_loader, optimizer, epoch):
    model.train()
    for batch_idx, (data, target) in enumerate(train_loader):
        data, target = data.to(device), target.to(device)
        optimizer.zero_grad()
        output = model(data)
        loss = F.nll_loss(output, target)
        loss.backward()
        optimizer.step()
        if batch_idx % args.log_interval == 0:
            print('Train Epoch: {} [{}/{} ({:.0f}%)]\tLoss: {:.6f}'.format(
                epoch, batch_idx * len(data), len(train_loader.dataset),
                100. * batch_idx / len(train_loader), loss.item()))
```

Taken from: https://pytorch.org/

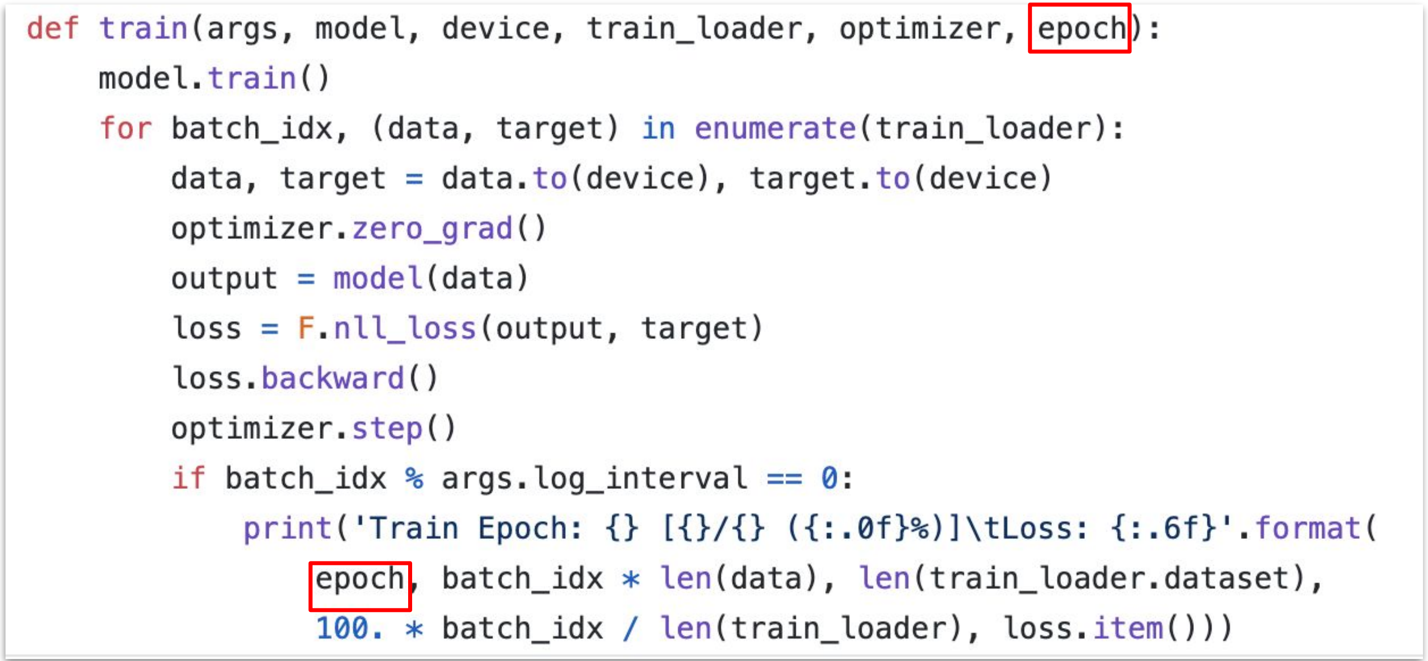
PyTorch Example



```python
def train(args, model, device, train_loader, optimizer, epoch):
    model.train()
    for batch_idx, (data, target) in enumerate(train_loader):
        data, target = data.to(device), target.to(device)
        optimizer.zero_grad()
        output = model(data)
        loss = F.nll_loss(output, target)
        loss.backward()
        optimizer.step()
        if batch_idx % args.log_interval == 0:
            print('Train Epoch: {} [{}/{} ({:.0f}%)]\tLoss: {:.6f}'.format(
                epoch, batch_idx * len(data), len(train_loader.dataset),
                100. * batch_idx / len(train_loader), loss.item()))
```

Taken from: https://pytorch.org/

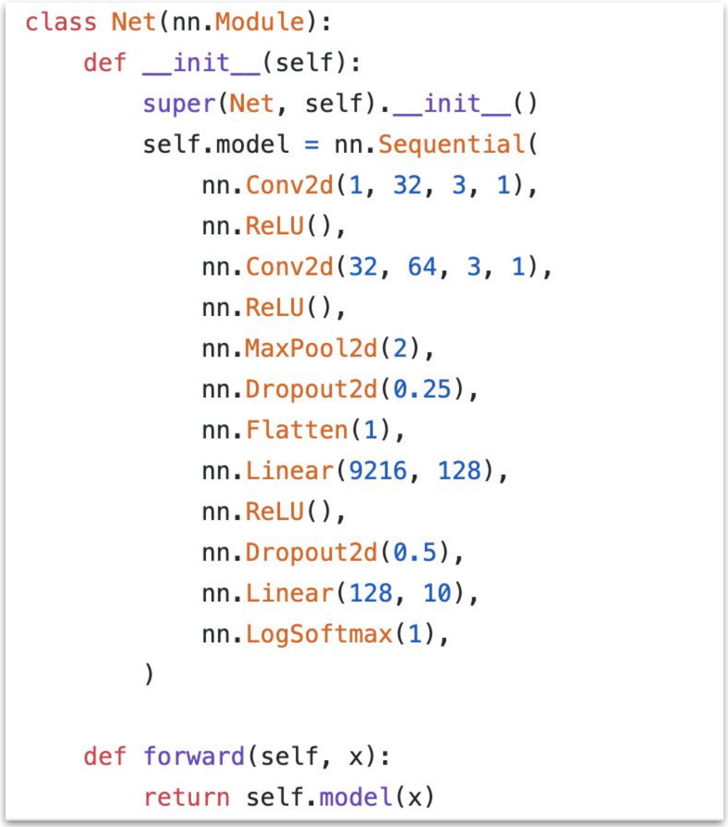
PyTorch Example



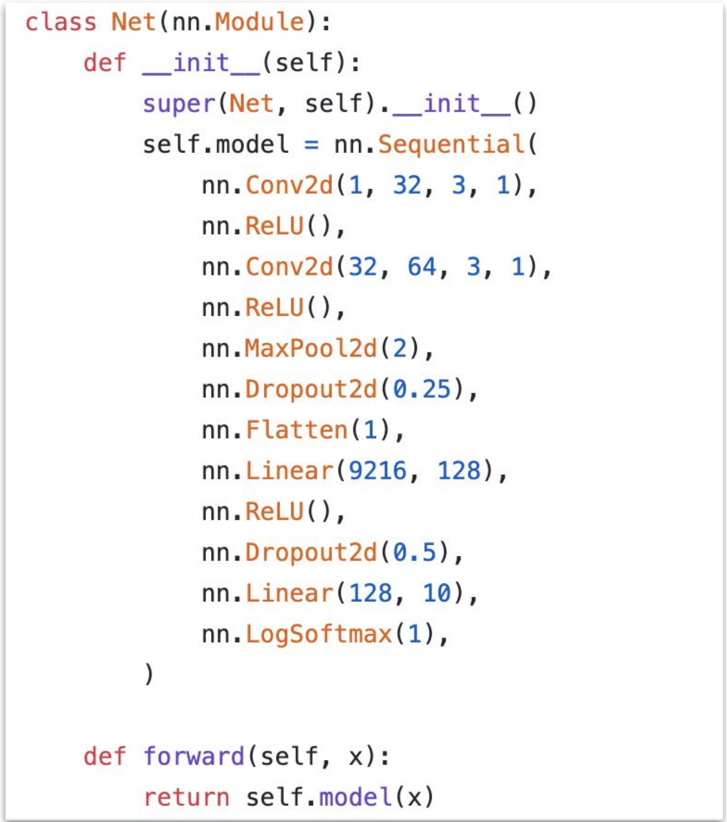
```
def train(args, model, device, train_loader, optimizer, epoch):
    model.train()
    for batch_idx, (data, target) in enumerate(train_loader):
        data, target = data.to(device), target.to(device)
        optimizer.zero_grad()
        output = model(data)
        loss = F.nll_loss(output, target)
        loss.backward()
        optimizer.step()
        if batch_idx % args.log_interval == 0:
            print('Train Epoch: {} [{}/{} ({:.0f}%)]\tLoss: {:.6f}'.format(
                epoch, batch_idx * len(data), len(train_loader.dataset),
                100. * batch_idx / len(train_loader), loss.item()))
```

Taken from: https://pytorch.org/

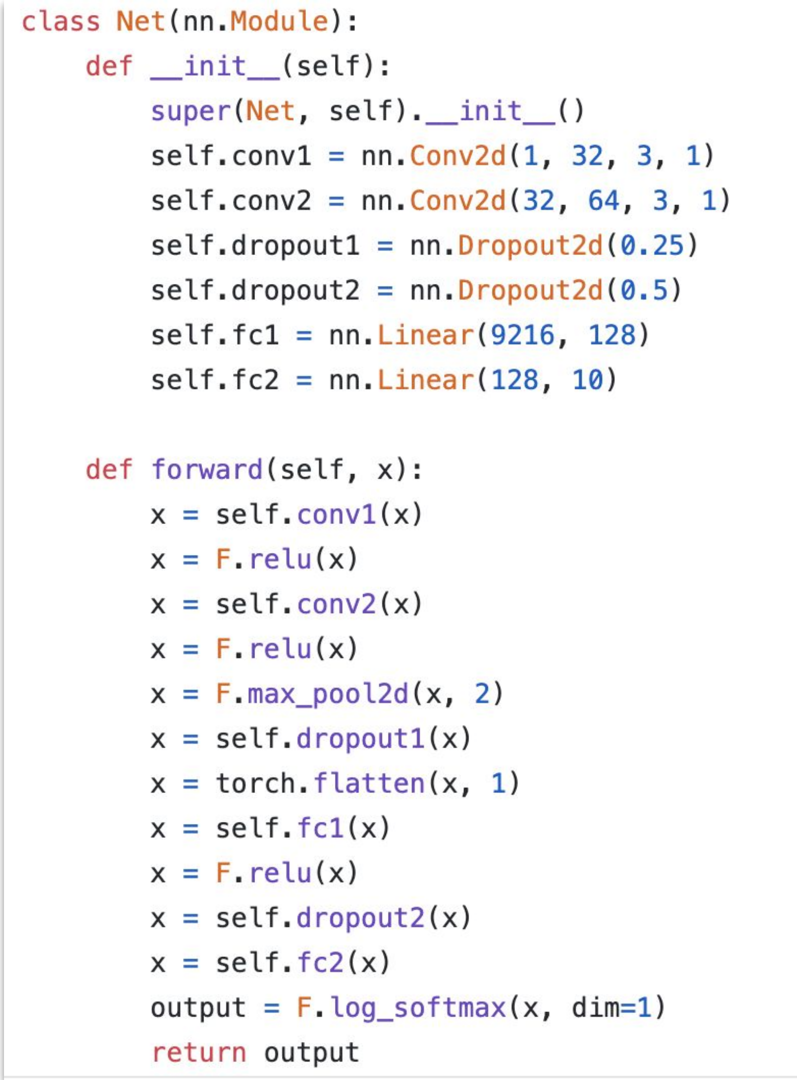
PyTorch Example


```python
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.model = nn.Sequential(
            nn.Conv2d(1, 32, 3, 1),
            nn.ReLU(),
            nn.Conv2d(32, 64, 3, 1),
            nn.ReLU(),
            nn.MaxPool2d(2),
            nn.Dropout2d(0.25),
            nn.Flatten(1),
            nn.Linear(9216, 128),
            nn.ReLU(),
            nn.Dropout2d(0.5),
            nn.Linear(128, 10),
            nn.LogSoftmax(1),
        )

    def forward(self, x):
        return self.model(x)
```

PyTorch Example

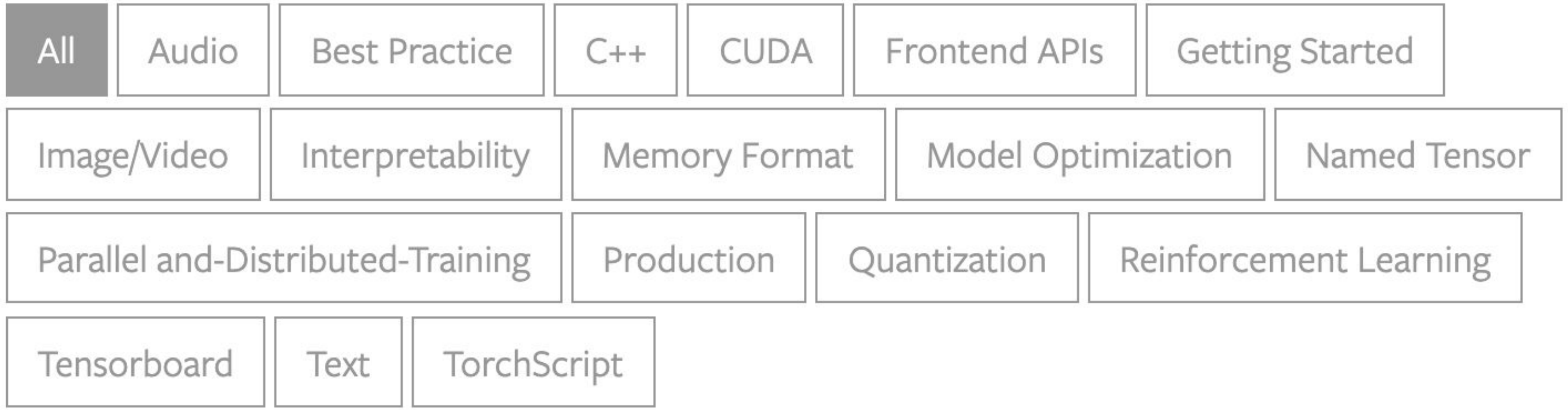


```python
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.model = nn.Sequential(
            nn.Conv2d(1, 32, 3, 1),
            nn.ReLU(),
            nn.Conv2d(32, 64, 3, 1),
            nn.ReLU(),
            nn.MaxPool2d(2),
            nn.Dropout2d(0.25),
            nn.Flatten(1),
            nn.Linear(9216, 128),
            nn.ReLU(),
            nn.Dropout2d(0.5),
            nn.Linear(128, 10),
            nn.LogSoftmax(1),
        )

    def forward(self, x):
        return self.model(x)
```



```python
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(1, 32, 3, 1)
        self.conv2 = nn.Conv2d(32, 64, 3, 1)
        self.dropout1 = nn.Dropout2d(0.25)
        self.dropout2 = nn.Dropout2d(0.5)
        self.fc1 = nn.Linear(9216, 128)
        self.fc2 = nn.Linear(128, 10)

    def forward(self, x):
        x = self.conv1(x)
        x = F.relu(x)
        x = self.conv2(x)
        x = F.relu(x)
        x = F.max_pool2d(x, 2)
        x = self.dropout1(x)
        x = torch.flatten(x, 1)
        x = self.fc1(x)
        x = F.relu(x)
        x = self.dropout2(x)
        x = self.fc2(x)
        output = F.log_softmax(x, dim=1)
        return output
```

PyTorch Tutorials



Taken from: https://pytorch.org/

Agenda

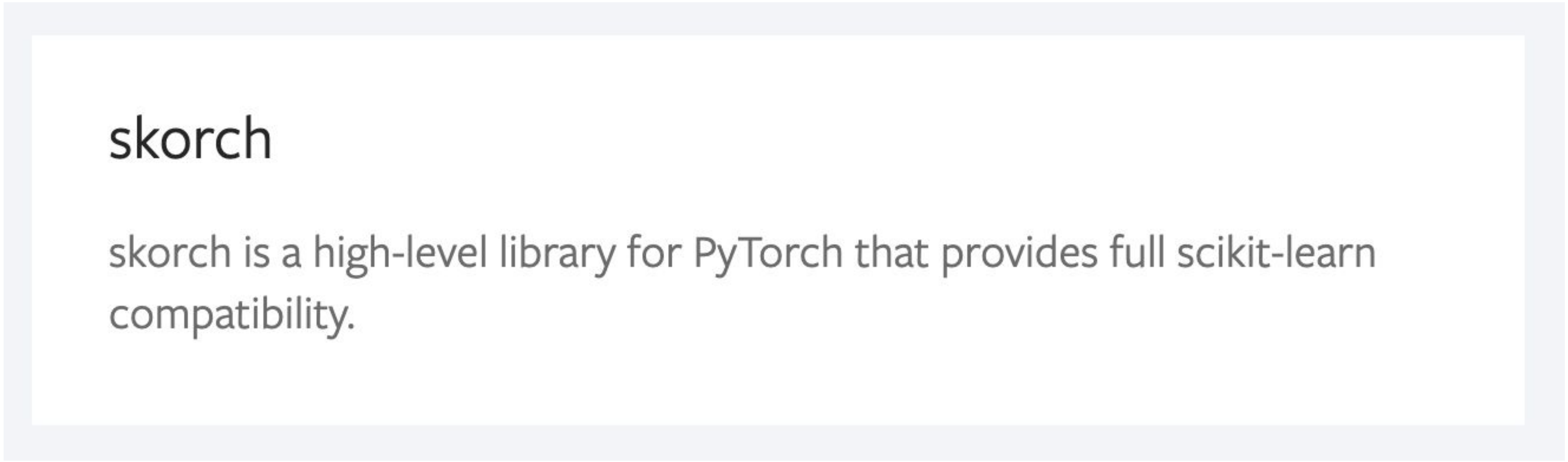
1. PyTorch Framework
2. How to get started
3. **PyTorch Ecosystem**
4. PyTorch for production

Questions are welcome at all times :)

PyTorch Ecosystem

- Around 40 featured projects, tools, and libraries
- Developed by researchers in academia and industry, application developers, and ML engineers.
- https://pytorch.org/ecosystem/

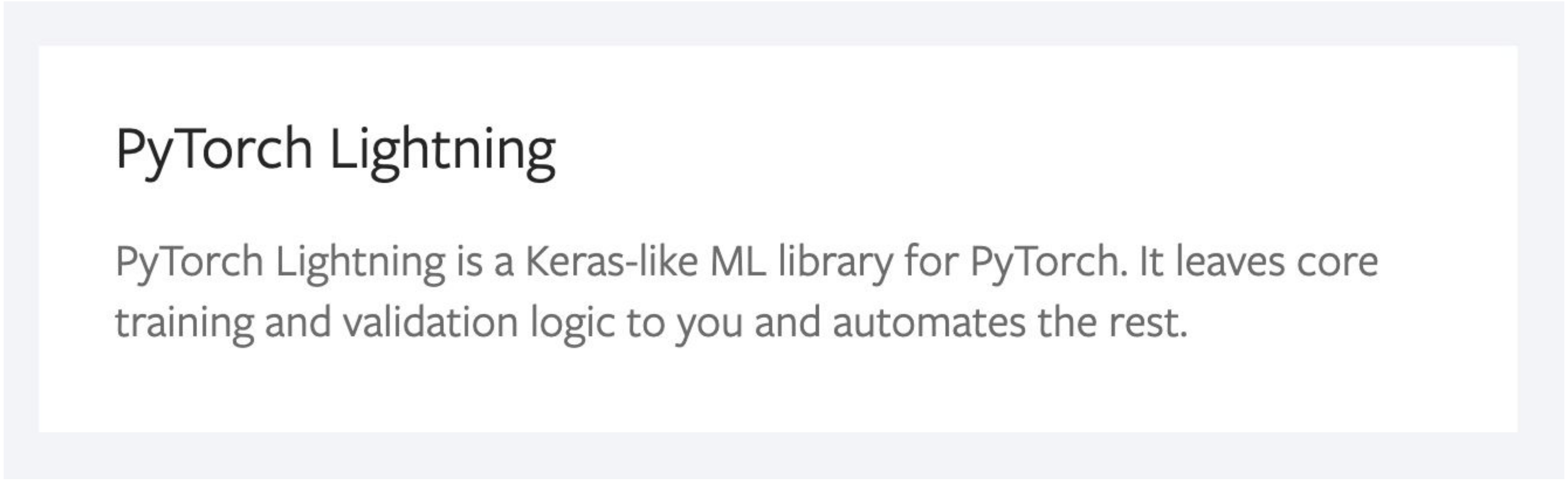
Machine Learning


skorch

skorch is a high-level library for PyTorch that provides full scikit-learn compatibility.

Taken from: https://pytorch.org/

Machine Learning

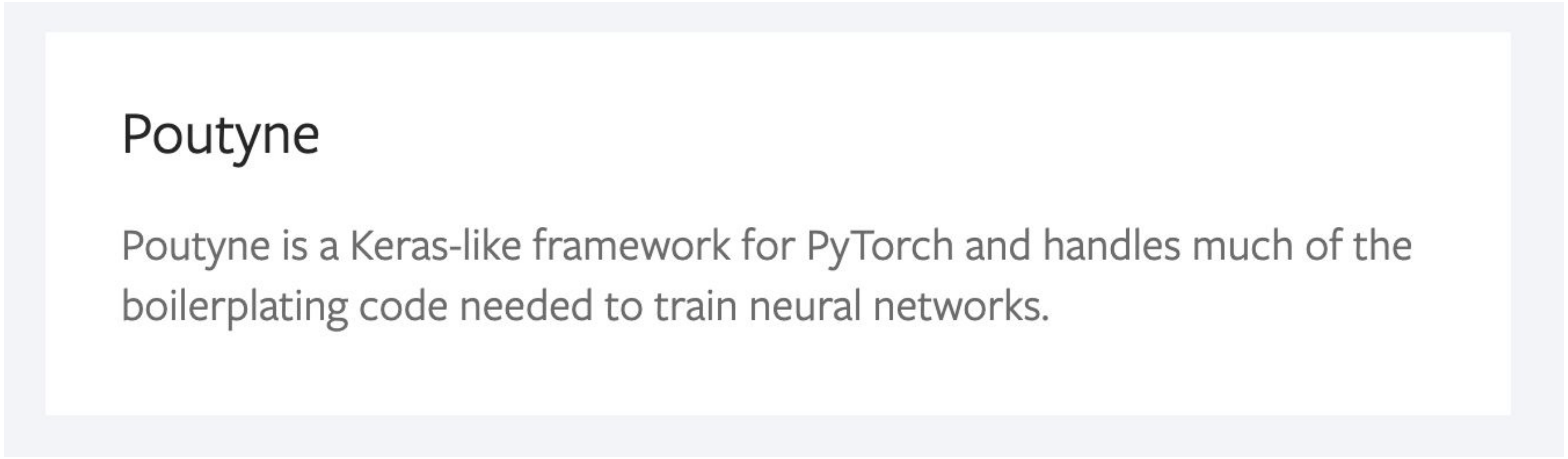


PyTorch Lightning

PyTorch Lightning is a Keras-like ML library for PyTorch. It leaves core training and validation logic to you and automates the rest.

Taken from: https://pytorch.org/

Machine Learning

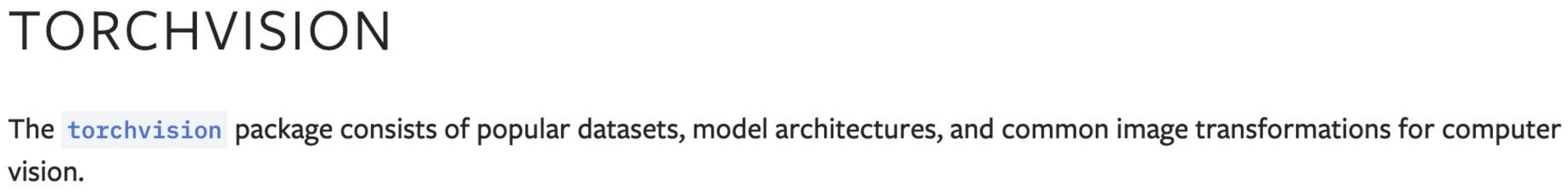


Poutyne

Poutyne is a Keras-like framework for PyTorch and handles much of the boilerplating code needed to train neural networks.

Taken from: https://pytorch.org/

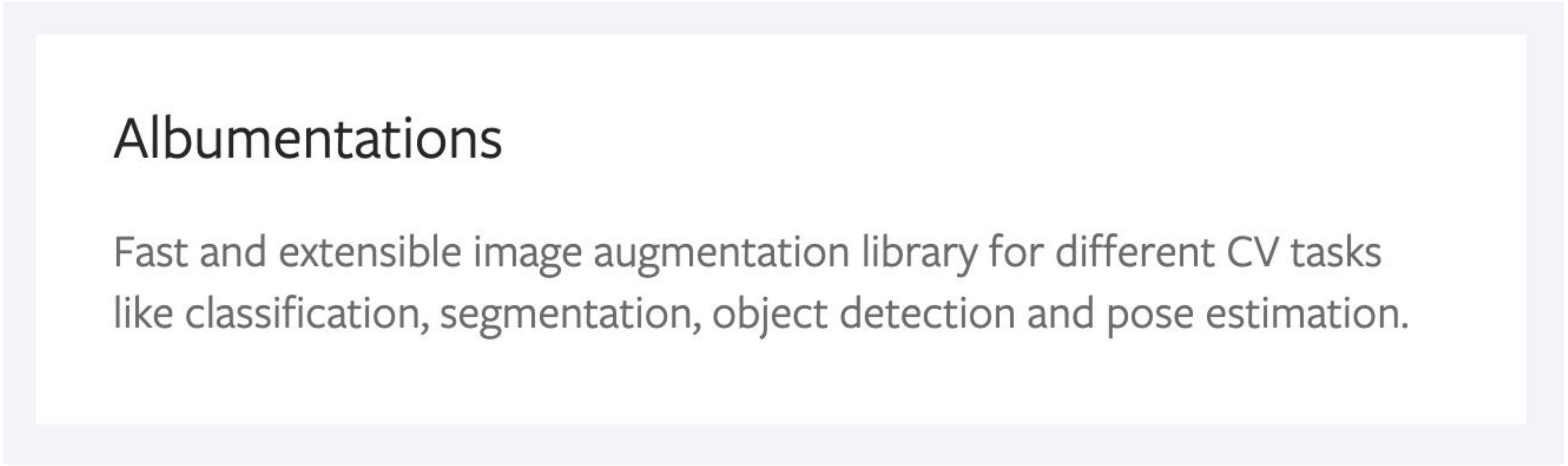
Vision


TORCHVISION

The `torchvision` package consists of popular datasets, model architectures, and common image transformations for computer vision.

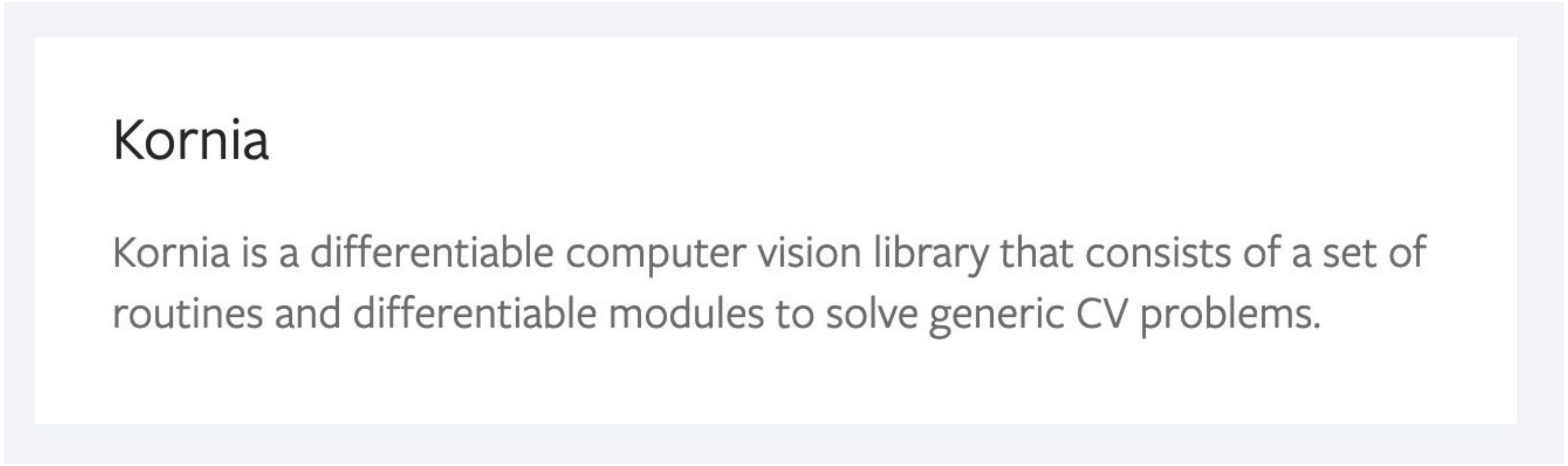
Taken from: https://pytorch.org/

Vision


Albumentations

Fast and extensible image augmentation library for different CV tasks like classification, segmentation, object detection and pose estimation.

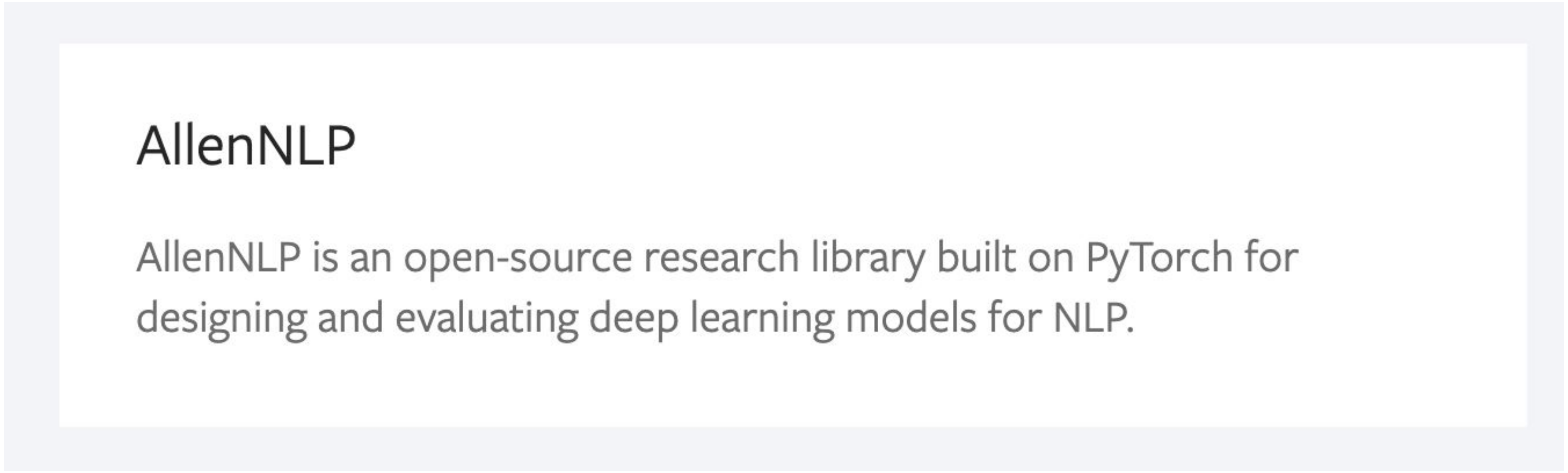
Vision



Kornia

Kornia is a differentiable computer vision library that consists of a set of routines and differentiable modules to solve generic CV problems.

NLP

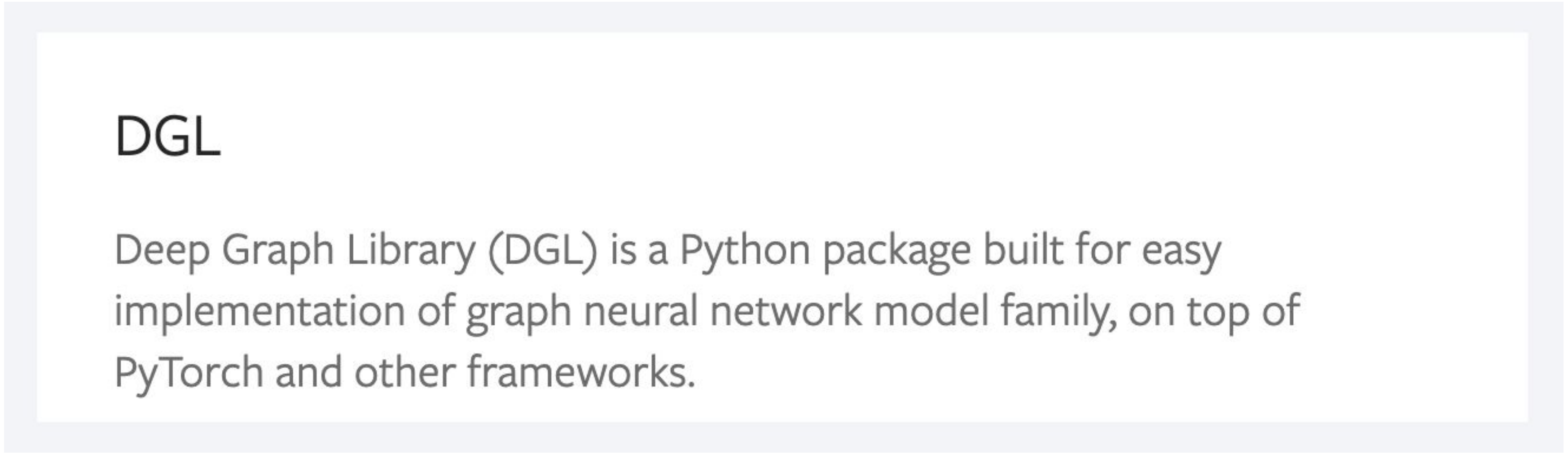


AllenNLP

AllenNLP is an open-source research library built on PyTorch for designing and evaluating deep learning models for NLP.

Taken from: https://pytorch.org/

Graph

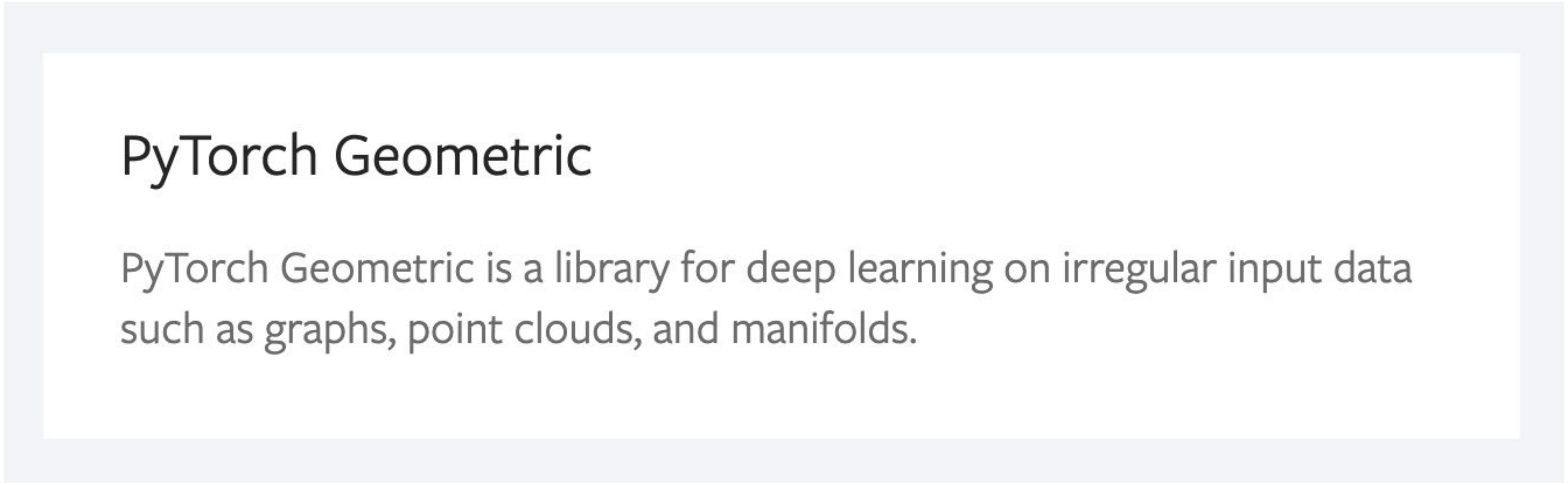


DGL

Deep Graph Library (DGL) is a Python package built for easy implementation of graph neural network model family, on top of PyTorch and other frameworks.

Taken from: https://pytorch.org/

Graph

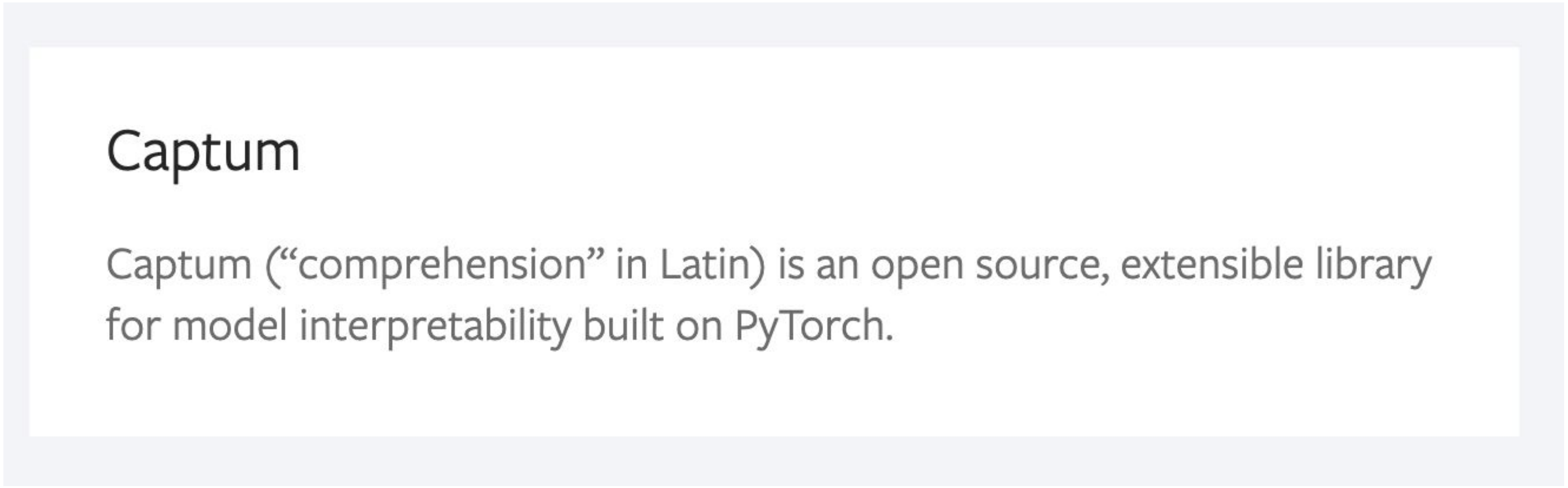


PyTorch Geometric

PyTorch Geometric is a library for deep learning on irregular input data such as graphs, point clouds, and manifolds.

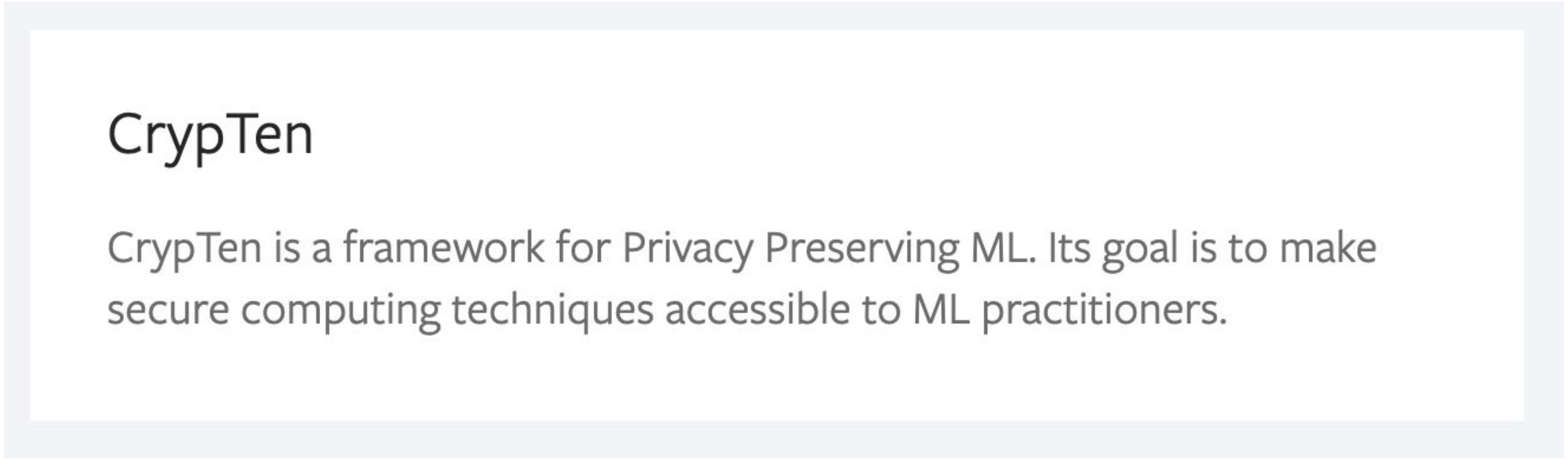
Taken from: https://pytorch.org/

Model Interpretability


Captum

Captum ("comprehension" in Latin) is an open source, extensible library for model interpretability built on PyTorch.

Privacy Preserving ML

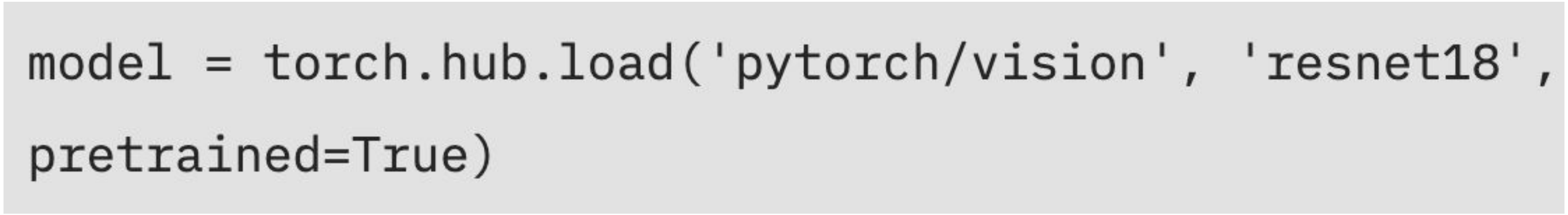


CrypTen

CrypTen is a framework for Privacy Preserving ML. Its goal is to make secure computing techniques accessible to ML practitioners.

Taken from: https://pytorch.org/

PyTorch Hub



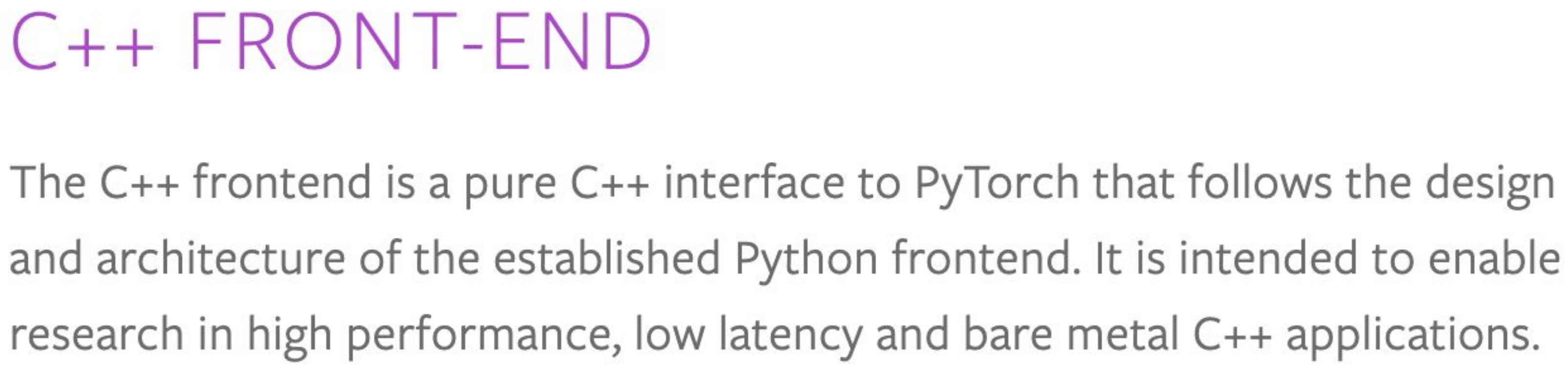
```
model = torch.hub.load('pytorch/vision', 'resnet18',
pretrained=True)
```

Agenda

1. PyTorch Framework
2. How to get started
3. PyTorch Ecosystem
4. **PyTorch for production**

Questions are welcome at all times :)

https://pytorch.org/cppdocs

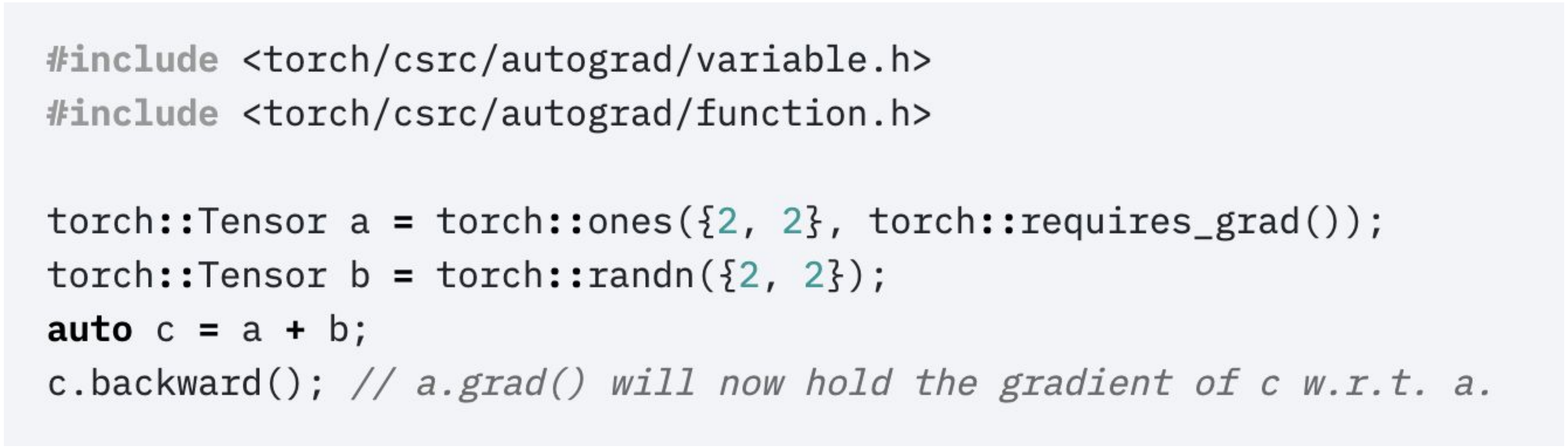


C++ FRONT-END

The C++ frontend is a pure C++ interface to PyTorch that follows the design and architecture of the established Python frontend. It is intended to enable research in high performance, low latency and bare metal C++ applications.

Taken from: https://pytorch.org/

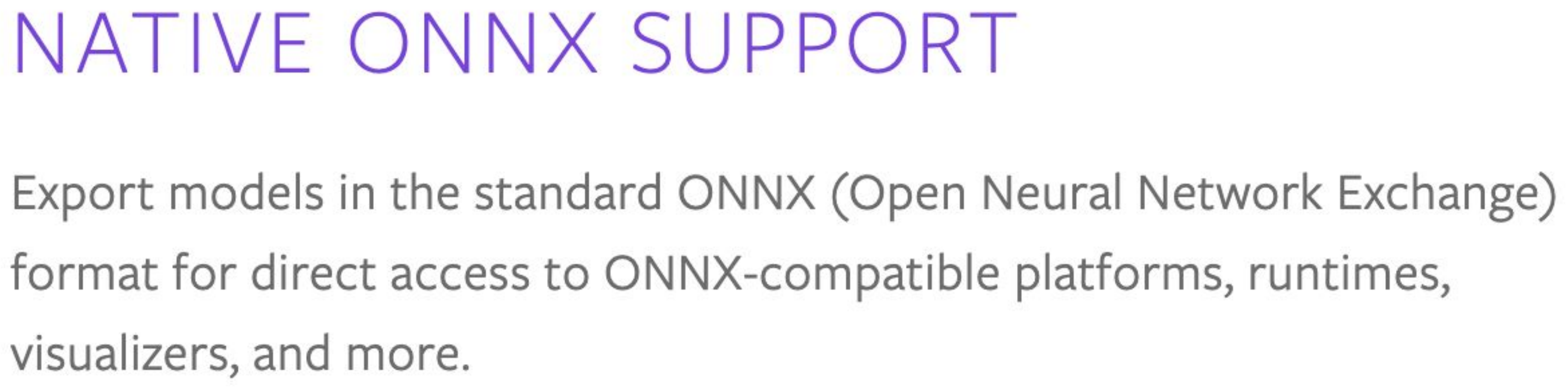
https://pytorch.org/cppdocs



```cpp
#include <torch/csrc/autograd/variable.h>
#include <torch/csrc/autograd/function.h>

torch::Tensor a = torch::ones({2, 2}, torch::requires_grad());
torch::Tensor b = torch::randn({2, 2});
auto c = a + b;
c.backward(); // a.grad() will now hold the gradient of c w.r.t. a.
```

Taken from: https://pytorch.org/

https://pytorch.org/docs/stable/onnx.html



NATIVE ONNX SUPPORT

Export models in the standard ONNX (Open Neural Network Exchange) format for direct access to ONNX-compatible platforms, runtimes, visualizers, and more.

ONNX

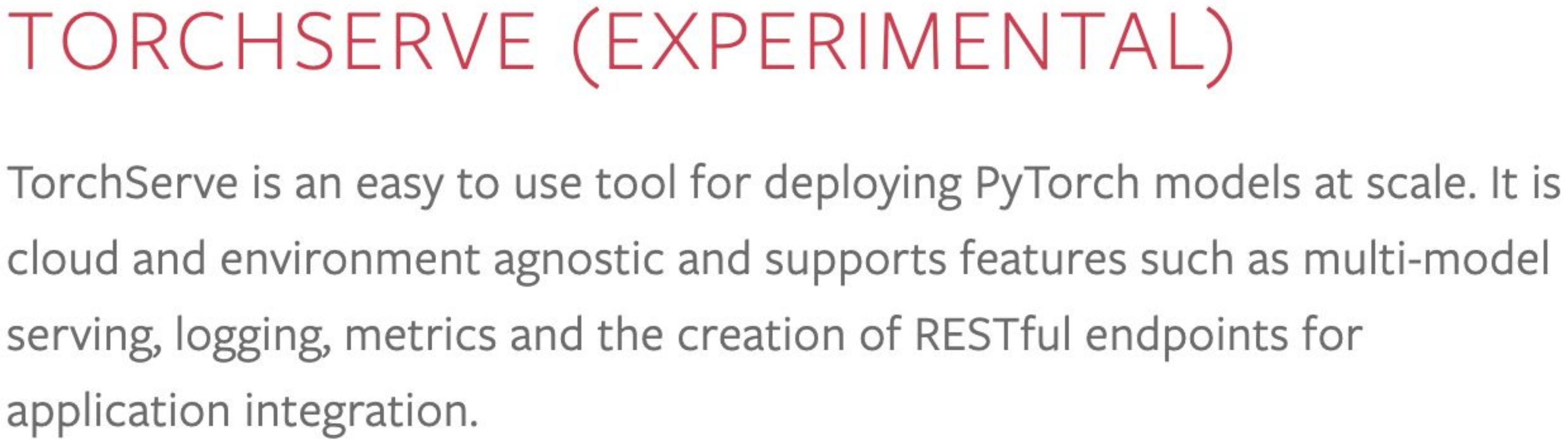
1. Standard for exchanging ML models

2. Supports interoperability between frameworks

3. Train with framework X, deploy with framework Y

4. Supports PyTorch, TensorFlow, Keras, Scikit-Learn, mxnet,....

https://pytorch.org/serve/



TORCHSERVE (EXPERIMENTAL)

TorchServe is an easy to use tool for deploying PyTorch models at scale. It is cloud and environment agnostic and supports features such as multi-model serving, logging, metrics and the creation of RESTful endpoints for application integration.

TorchServe

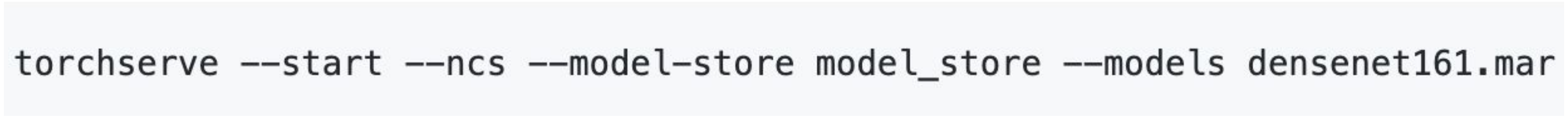
1. Supports Python-based and TorchScript-based models

2. Model versioning + rollback

3. Batches inference requests

4. Dockerfile for easy deployment

TorchServe



```
torchserve --start --ncs --model-store model_store --models densenet161.mar
```

TorchServe

```
torchserve --start --ncs --model-store model_store --models densenet161.mar
```

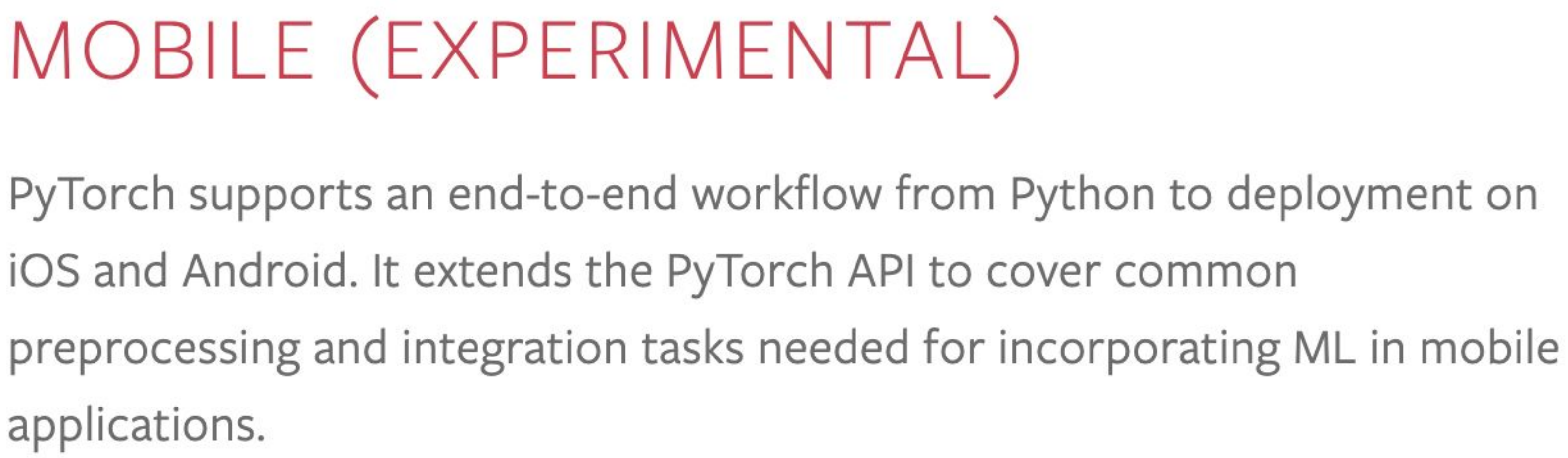
```
curl -O https://s3.amazonaws.com/model-server/inputs/kitten.jpg
curl http://127.0.0.1:8080/predictions/densenet161 -T kitten.jpg
```

TorchServe

```
[
  {
    "tiger_cat": 0.46933549642562866
  },
  {
    "tabby": 0.4633878469467163
  },
  {
    "Egyptian_cat": 0.06456148624420166
  },
  {
    "lynx": 0.0012828214094042778
  },
  {
    "plastic_bag": 0.00023323034110944718
  }
]
```

Taken from: https://pytorch.org/

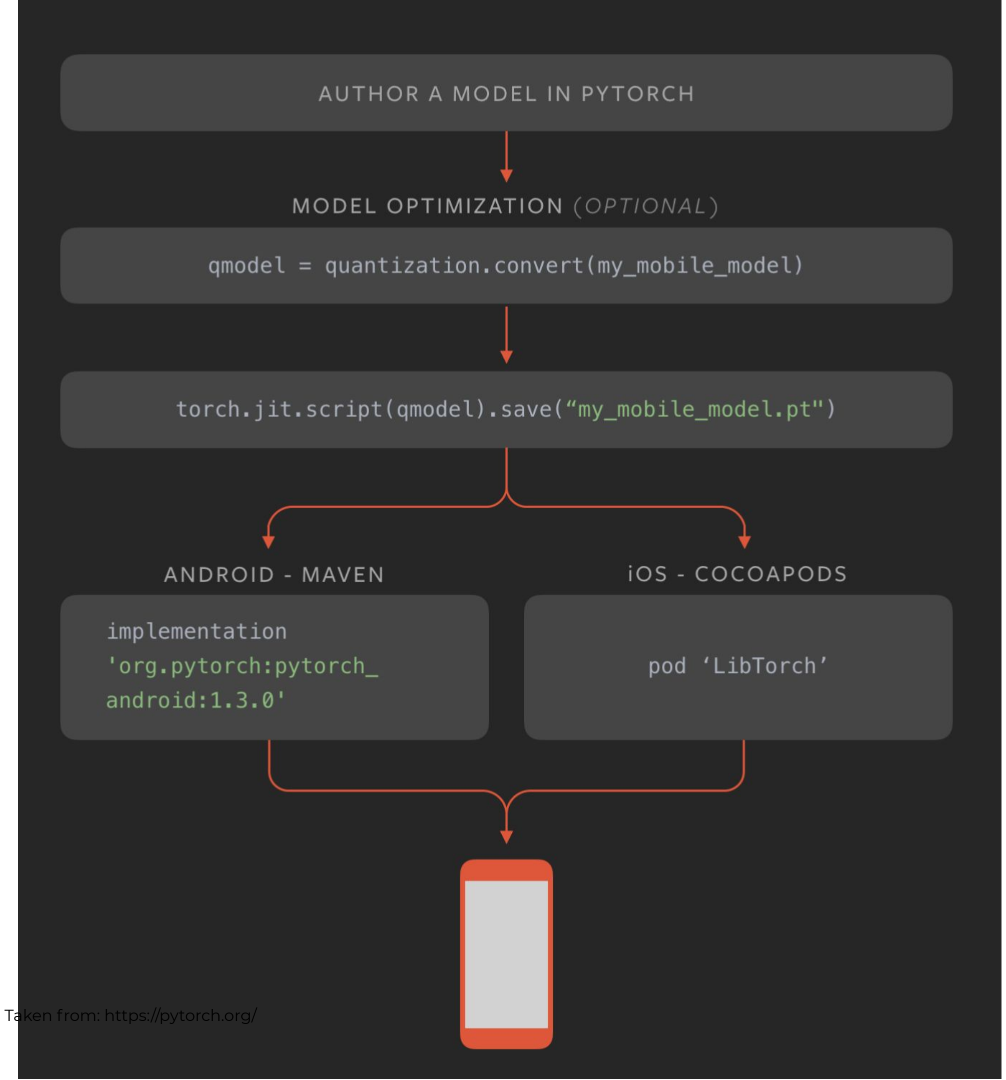
https://pytorch.org/mobile



MOBILE (EXPERIMENTAL)

PyTorch supports an end-to-end workflow from Python to deployment on iOS and Android. It extends the PyTorch API to cover common preprocessing and integration tasks needed for incorporating ML in mobile applications.

Taken from: https://pytorch.org/



```
qmodel = quantization.convert(my_mobile_model)
```

```
torch.jit.script(qmodel).save(“my_mobile_model.pt")
```

```
implementation
'org.pytorch:pytorch_
android:1.3.0'
```

```
pod ‘LibTorch’
```

Taken from: https://pytorch.org/

TorchScript

TorchScript

1. TorchScript is an intermediate representation of a PyTorch model.

2. It can be run in a high-performance environment such as C++.

Quantization

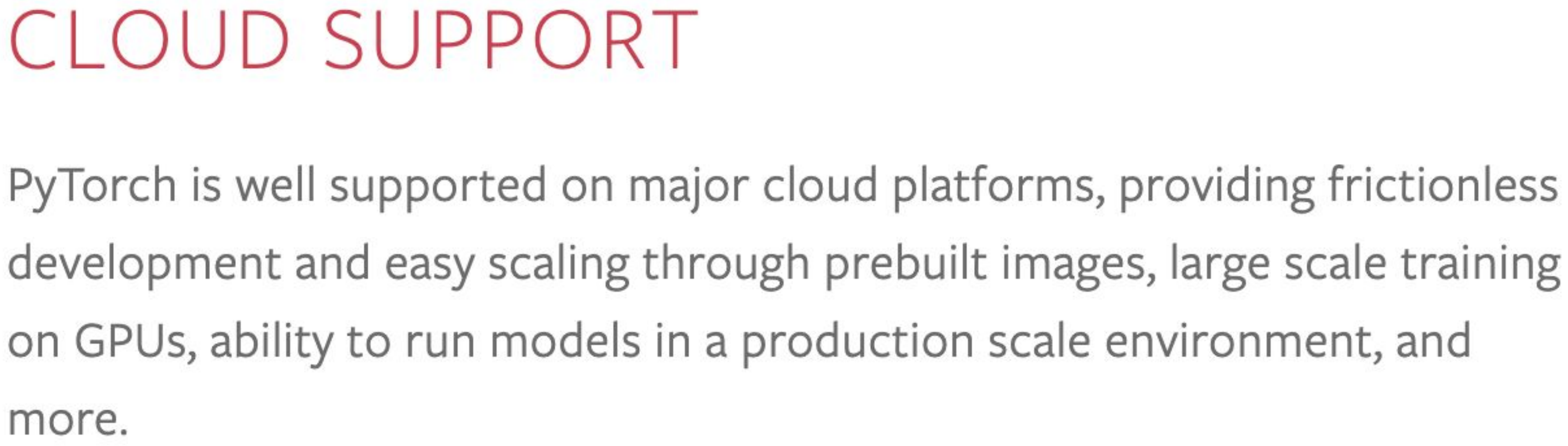
Quantization

1. Lower precision data (int8)

2. Savings in model size, memory bandwidth, and inference time

3. PyTorch supports:
   - Dynamic Quantization
   - Post-Training Static Quantization
   - Quantization Aware Training

https://pytorch.org/get-started/cloud-partners

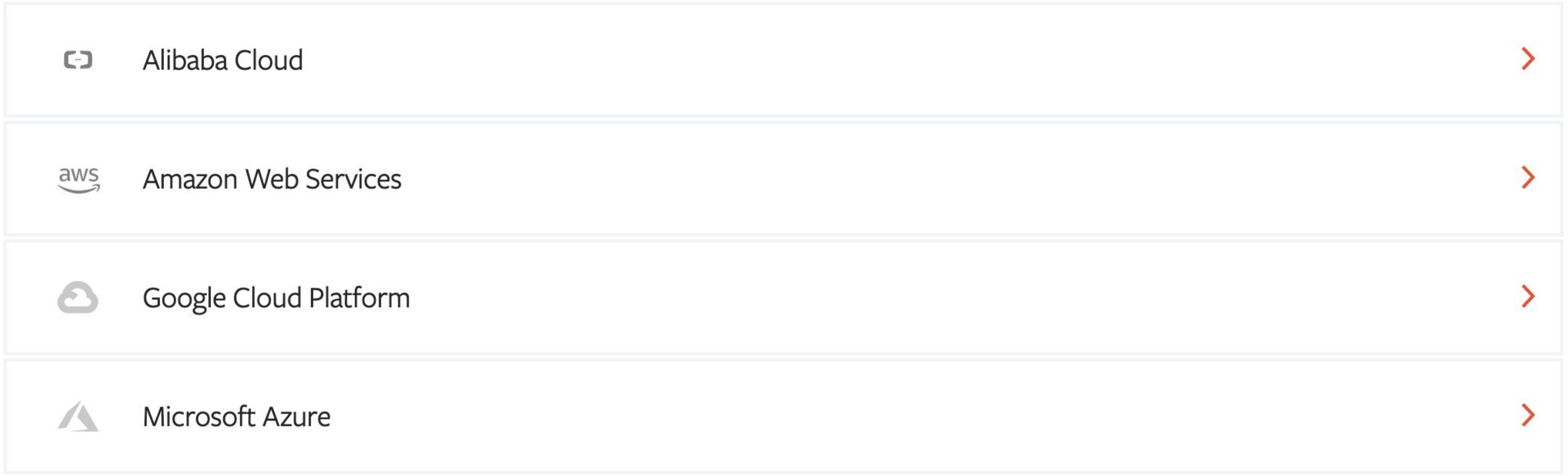


CLOUD SUPPORT

PyTorch is well supported on major cloud platforms, providing frictionless development and easy scaling through prebuilt images, large scale training on GPUs, ability to run models in a production scale environment, and more.

Taken from: https://pytorch.org/

https://pytorch.org/get-started/cloud-partners



Alibaba Cloud

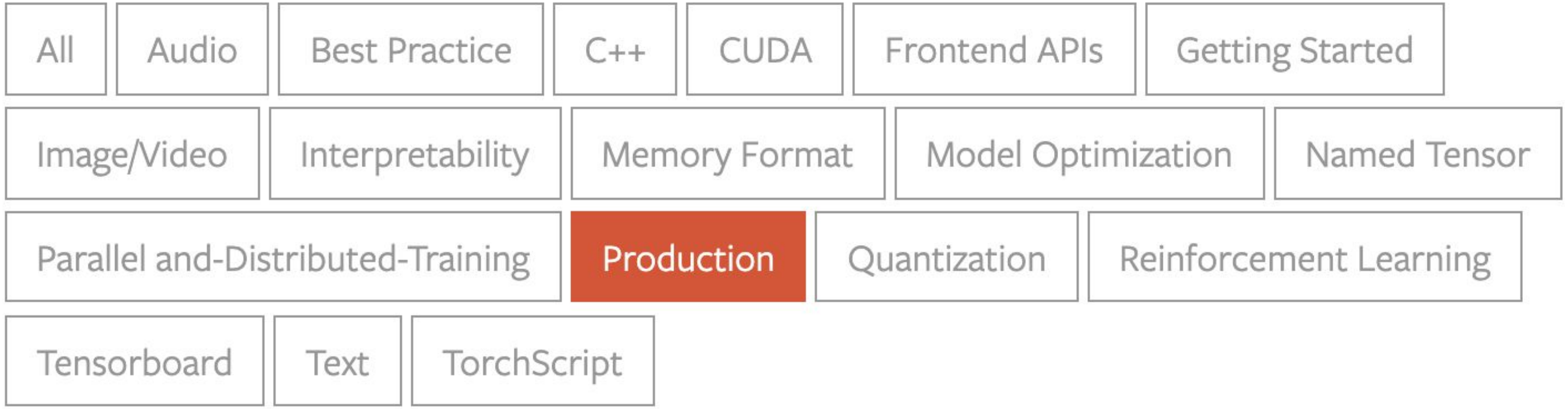
Amazon Web Services

Google Cloud Platform

Microsoft Azure

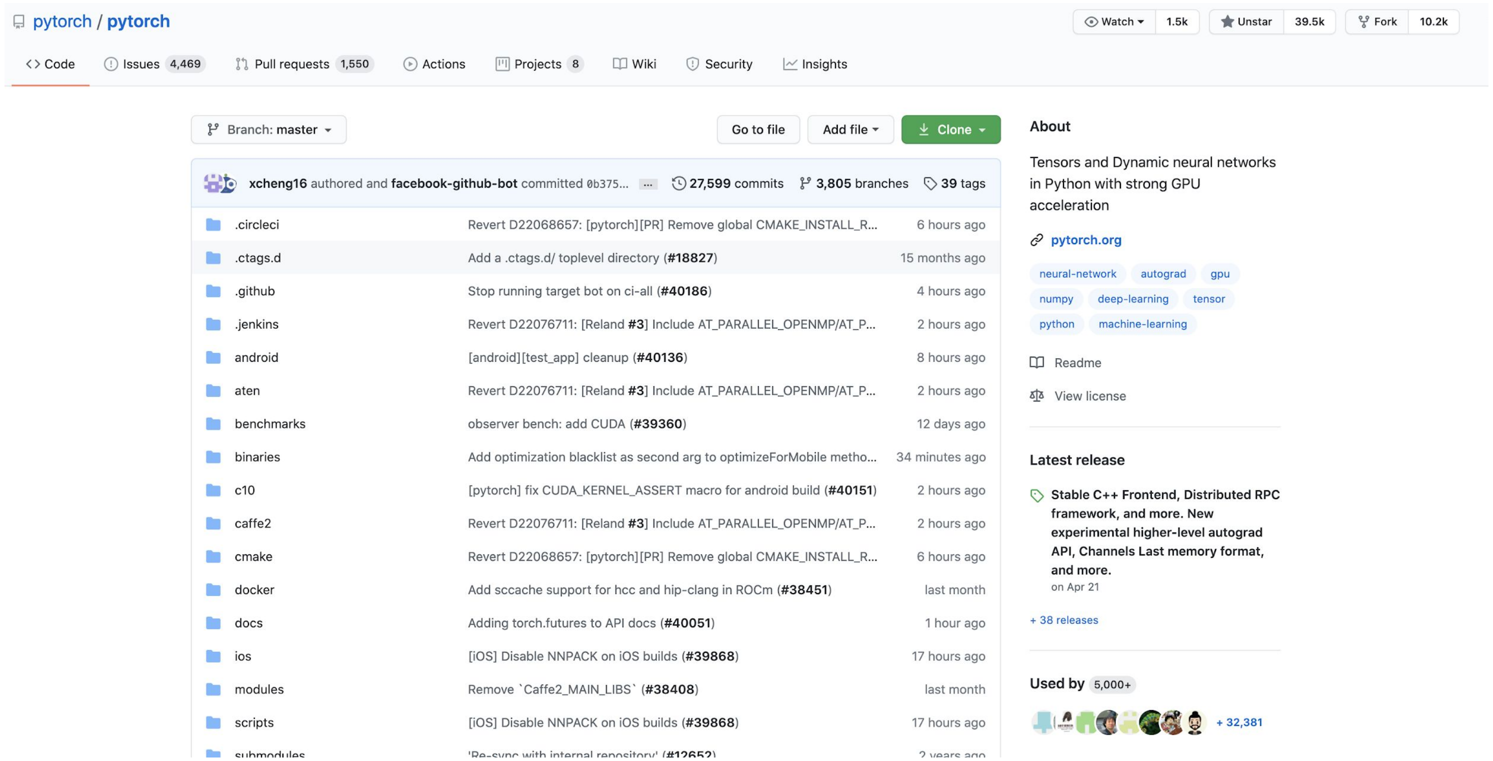
Taken from: https://pytorch.org/

https://pytorch.org/tutorials/


All
Audio
Best Practice
C++
CUDA
Frontend APIs
Getting Started
Image/Video
Interpretability
Memory Format
Model Optimization
Named Tensor
Parallel and-Distributed-Training
Production
Quantization
Reinforcement Learning
Tensorboard
Text
TorchScript

Taken from: https://pytorch.org/

Community



pytorch / pytorch

Watch 1.5k | Unstar 39.5k | Fork 10.2k

Code | Issues 4,469 | Pull requests 1,550 | Actions | Projects 8 | Wiki | Security | Insights

Branch: master | Go to file | Add file | Clone

xcheng16 authored and facebook-github-bot committed 0b375... | 27,599 commits | 3,805 branches | 39 tags

.circleci	Revert D22068657: [pytorch][PR] Remove global CMAKE_INSTALL_R...	6 hours ago
.ctags.d	Add a .ctags.d/ toplevel directory (#18827)	15 months ago
.github	Stop running target bot on ci-all (#40186)	4 hours ago
.jenkins	Revert D22076711: [Reland #3] Include AT_PARALLEL_OPENMP/AT_P...	2 hours ago
android	[android][test_app] cleanup (#40136)	8 hours ago
aten	Revert D22076711: [Reland #3] Include AT_PARALLEL_OPENMP/AT_P...	2 hours ago
benchmarks	observer bench: add CUDA (#39360)	12 days ago
binaries	Add optimization blacklist as second arg to optimizeForMobile metho...	34 minutes ago
c10	[pytorch] fix CUDA_KERNEL_ASSERT macro for android build (#40151)	2 hours ago
caffe2	Revert D22076711: [Reland #3] Include AT_PARALLEL_OPENMP/AT_P...	2 hours ago
cmake	Revert D22068657: [pytorch][PR] Remove global CMAKE_INSTALL_R...	6 hours ago
docker	Add sccache support for hcc and hip-clang in ROCm (#38451)	last month
docs	Adding torch.futures to API docs (#40051)	1 hour ago
ios	[iOS] Disable NNPACK on iOS builds (#39868)	17 hours ago
modules	Remove `Caffe2_MAIN_LIBS` (#38408)	last month
scripts	[iOS] Disable NNPACK on iOS builds (#39868)	17 hours ago
submodules	'Re-sync with internal repository' (#12652)	2 years ago

About

Tensors and Dynamic neural networks in Python with strong GPU acceleration

pytorch.org

neural-network | autograd | gpu | numpy | deep-learning | tensor | python | machine-learning

Readme

View license

Latest release

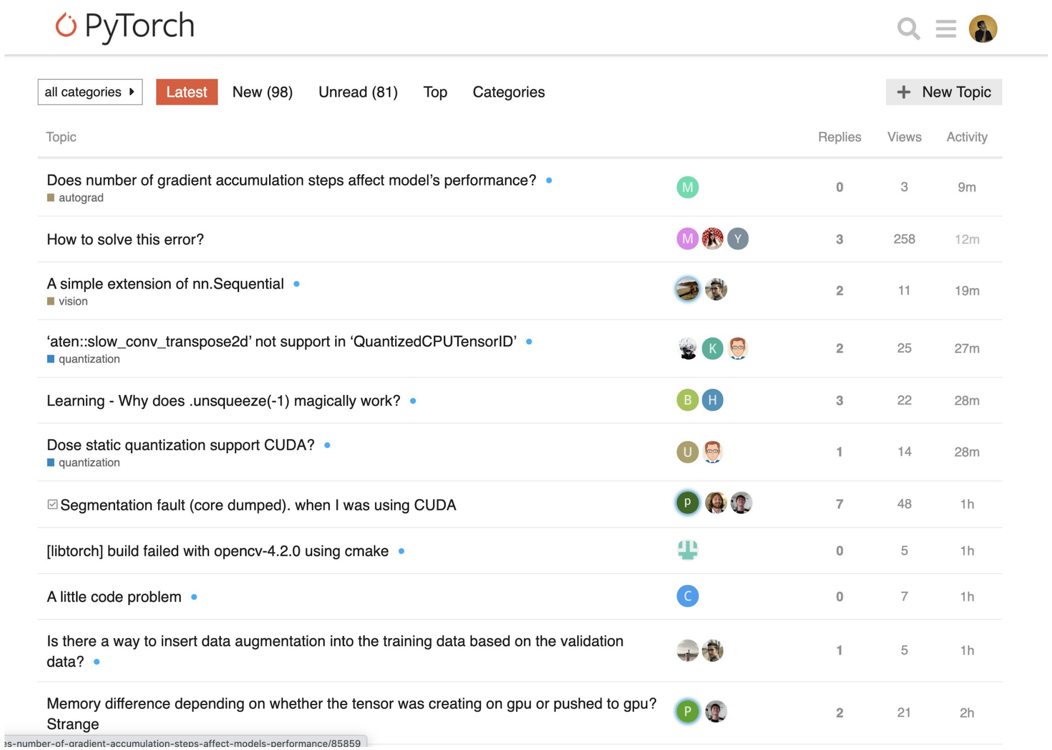
Stable C++ Frontend, Distributed RPC framework, and more. New experimental higher-level autograd API, Channels Last memory format, and more.
on Apr 21

+ 38 releases

Used by 5,000+

+ 32,381

Community



PyTorch

all categories | Latest | New (98) | Unread (81) | Top | Categories | New Topic

Topic	Replies	Views	Activity
Does number of gradient accumulation steps affect model's performance? autograd	0	3	9m
How to solve this error?	3	258	12m
A simple extension of nn.Sequential vision	2	11	19m
'aten::slow_conv_transpose2d' not support in 'QuantizedCPUTensorID' quantization	2	25	27m
Learning - Why does .unsqueeze(-1) magically work?	3	22	28m
Dose static quantization support CUDA? quantization	1	14	28m
☑ Segmentation fault (core dumped). when I was using CUDA	7	48	1h
[libtorch] build failed with opencv-4.2.0 using cmake	0	5	1h
A little code problem	0	7	1h
Is there a way to insert data augmentation into the training data based on the validation data?	1	5	1h
Memory difference depending on whether the tensor was creating on gpu or pushed to gpu? Strange	2	21	2h

es-number-of-gradient-accumulation-steps-affect-models-performance/85859

Agenda

1. PyTorch Framework
2. How to get started
3. PyTorch Ecosystem
4. PyTorch for production

Questions are welcome at all times :)

Thank you

@shagunsodhani

FACEBOOK AI