

PyTorch Tutorials

Q&A

Search docs

PyTorch 1.10.0 (latest)

Deep Learning with PyTorch: A 6-Week Blitz

What is PyTorch?

PyTorch: automatic differentiation

Neural Networks

Training a classifier

What about data?

Training an image classifier

Training on GPUs

Training on multiple GPUs

Where did you go?

Optional: Data Parallelism

PyTorch for former Torch users

Learning PyTorch with Examples

Transfer Learning Tutorial

Deep Learning and Processing Tutorial

Deep learning for NLP with PyTorch

PyTorch on AWS

PyTorch on Google Cloud

Classifying Names with a Character-Level RNN

Generating Names with a Character-Level RNN

Translation with a Sequence-to-Sequence Network and Attention

Training a classifier

This is a 6-Week Blitz course to define neural networks, compute loss, and make updates to the weights of the networks.

How you might be thinking.

What about data?

Generally where you have a deal with images, text, audio or video data, you can use standard Python packages that load data into a numpy array. Then you can convert this array into a `torch.FloatTensor`.

- For images, packages such as Pillow, OpenCV are useful
- For audio, packages such as scipy and librosa
- For text, either raw Python or Cython-based loading or NLTK and SpaCy are useful

Specifically for video, we have created a package called `torchaudio`, that has data loaders for common datasets such as ImageNet, CIFAR10, MNIST, etc. and data transformers for images, [detailed here](#).

This provides a huge convenience and avoids writing boilerplate code.

For this tutorial, we will use the CIFAR10 dataset. It has the classes: 'airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck'. The images in CIFAR-10 are of size 32x32, in 3 channel color images of 256x256 pixels in total.



cifar10

Training an image classifier

We will do the following steps in order:

- Load and normalize the CIFAR10 training and test datasets using [torchvision](#).
- Define a Convolutional Neural Network.
- Define loss function
- Train the network on the training data
- Test the network on the test data

1. Loading and normalizing CIFAR10

Using [torchvision](#) it's extremely easy to load CIFAR10.

```
import torch
import torchvision
import torchvision.transforms as transforms

# Training data
trainset = torchvision.datasets.CIFAR10(root='./data', train=True,
                                       transform=transforms.Compose([
                                           transforms.RandomCrop(32),
                                           transforms.RandomHorizontalFlip(),
                                           transforms.ToTensor(),
                                           transforms.Normalize([0.4914, 0.4822, 0.4465], [0.203, 0.1994, 0.2019])
                                       ]), batch_size=4, shuffle=True, num_workers=4)

# Test data
testset = torchvision.datasets.CIFAR10(root='./data', train=False,
                                       transform=transforms.Compose([
                                           transforms.ToTensor(),
                                           transforms.Normalize([0.4914, 0.4822, 0.4465], [0.203, 0.1994, 0.2019])
                                       ]), batch_size=4, shuffle=False, num_workers=4)

classes = ('plane', 'bird', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck')

train_loader = torch.utils.data.DataLoader(trainset, batch_size=4, shuffle=True, num_workers=4)
test_loader = torch.utils.data.DataLoader(testset, batch_size=4, shuffle=False, num_workers=4)
```

Out:

```
Files of ready downloaded and verified
Files of ready downloaded and verified
```

Let us observe some of the training images, for fun.

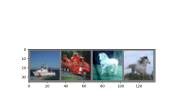
```
import random
import torchvision.transforms as transforms
import numpy as np

# Functions to show an image

def imshow(img):
    img = img / 2 + 0.5  # unnormalize
    npimg = img.numpy()
    plt.imshow(np.transpose(npimg, (1, 2, 0)))

# get some random training images
dataiter = iter(train_loader)
images, labels = dataiter.next()

# show images
imshow(torchvision.utils.make_grid(images))
# print labels
print(' '.join('%5s' % classes[labels[j]] for j in range(16)))
```



Out:

```
ship truck horse horse
```

2. Define a Convolutional Neural Network

Copy the neural network from the Neural Networks section before and modify it to take 3-channel images (instead of 1-channel images as it was defined).

```
import torch.nn as nn
import torch.nn.functional as F

class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(3, 6, 5)
        self.conv2 = nn.Conv2d(6, 16, 5)
        self.pool = nn.MaxPool2d(2, 2)
        self.conv3 = nn.Conv2d(16, 16, 5)
        self.conv4 = nn.Conv2d(16, 16, 5)
        self.conv5 = nn.Conv2d(16, 16, 5)
        self.conv6 = nn.Conv2d(16, 16, 5)
        self.fc1 = nn.Linear(1000, 1000)
        self.fc2 = nn.Linear(1000, 10)

    def forward(self, x):
        x = self.pool(self.conv1(self.pool(self.conv2(self.pool(self.conv3(x))))))
        x = self.pool(self.conv4(self.pool(self.conv5(self.pool(self.conv6(x))))))
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        return x

net = Net()
```

3. Define a Loss function and optimizer

Let's use a Classification Cross Entropy loss, and SGD with momentum.

```
import torch.optim as optim

criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(net.parameters(), lr=0.01, momentum=0.9)
```

4. Train the network

This is where things start to get interesting. We simply have to loop over our data iterator, and feed the inputs to the network and optimize.

```
for epoch in range(10):  # loop over the dataset multiple times
    running_loss = 0.0
    for i, data in enumerate(train_loader, 0):
        # get the inputs
        inputs, labels = data

        # zero the parameter gradients
        optimizer.zero_grad()

        # forward + backward + optimize
        outputs = net(inputs)
        loss = criterion(outputs, labels)
        loss.backward()

        # print statistics
        running_loss += loss.item()
        if i % 1000 == 0:    # print every 1000 mini-batches
            print('[%d, %5d] loss: %.3f' % (epoch + 1, i + 1, running_loss / 1000))
    running_loss = 0.0

print('Finished Training')
```

Out:

```
[0, 2000] loss: 1.289
[0, 4000] loss: 1.054
[0, 6000] loss: 1.008
[0, 8000] loss: 1.008
[0, 10000] loss: 1.008
[0, 12000] loss: 1.008
[0, 14000] loss: 1.008
[0, 16000] loss: 1.008
[0, 18000] loss: 1.008
[0, 20000] loss: 1.008
[0, 22000] loss: 1.008
[0, 24000] loss: 1.008
[0, 26000] loss: 1.008
[0, 28000] loss: 1.008
[0, 30000] loss: 1.008
[0, 32000] loss: 1.008
[0, 34000] loss: 1.008
[0, 36000] loss: 1.008
[0, 38000] loss: 1.008
[0, 40000] loss: 1.008
[0, 42000] loss: 1.008
[0, 44000] loss: 1.008
[0, 46000] loss: 1.008
[0, 48000] loss: 1.008
[0, 50000] loss: 1.008
[0, 52000] loss: 1.008
[0, 54000] loss: 1.008
[0, 56000] loss: 1.008
[0, 58000] loss: 1.008
[0, 60000] loss: 1.008
[0, 62000] loss: 1.008
[0, 64000] loss: 1.008
[0, 66000] loss: 1.008
[0, 68000] loss: 1.008
[0, 70000] loss: 1.008
[0, 72000] loss: 1.008
[0, 74000] loss: 1.008
[0, 76000] loss: 1.008
[0, 78000] loss: 1.008
[0, 80000] loss: 1.008
[0, 82000] loss: 1.008
[0, 84000] loss: 1.008
[0, 86000] loss: 1.008
[0, 88000] loss: 1.008
[0, 90000] loss: 1.008
[0, 92000] loss: 1.008
[0, 94000] loss: 1.008
[0, 96000] loss: 1.008
[0, 98000] loss: 1.008
[0, 100000] loss: 1.008
[0, 102000] loss: 1.008
[0, 104000] loss: 1.008
[0, 106000] loss: 1.008
[0, 108000] loss: 1.008
[0, 110000] loss: 1.008
[0, 112000] loss: 1.008
[0, 114000] loss: 1.008
[0, 116000] loss: 1.008
[0, 118000] loss: 1.008
[0, 120000] loss: 1.008
[0, 122000] loss: 1.008
[0, 124000] loss: 1.008
[0, 126000] loss: 1.008
[0, 128000] loss: 1.008
[0, 130000] loss: 1.008
[0, 132000] loss: 1.008
[0, 134000] loss: 1.008
[0, 136000] loss: 1.008
[0, 138000] loss: 1.008
[0, 140000] loss: 1.008
[0, 142000] loss: 1.008
[0, 144000] loss: 1.008
[0, 146000] loss: 1.008
[0, 148000] loss: 1.008
[0, 150000] loss: 1.008
[0, 152000] loss: 1.008
[0, 154000] loss: 1.008
[0, 156000] loss: 1.008
[0, 158000] loss: 1.008
[0, 160000] loss: 1.008
[0, 162000] loss: 1.008
[0, 164000] loss: 1.008
[0, 166000] loss: 1.008
[0, 168000] loss: 1.008
[0, 170000] loss: 1.008
[0, 172000] loss: 1.008
[0, 174000] loss: 1.008
[0, 176000] loss: 1.008
[0, 178000] loss: 1.008
[0, 180000] loss: 1.008
[0, 182000] loss: 1.008
[0, 184000] loss: 1.008
[0, 186000] loss: 1.008
[0, 188000] loss: 1.008
[0, 190000] loss: 1.008
[0, 192000] loss: 1.008
[0, 194000] loss: 1.008
[0, 196000] loss: 1.008
[0, 198000] loss: 1.008
[0, 200000] loss: 1.008
[0, 202000] loss: 1.008
[0, 204000] loss: 1.008
[0, 206000] loss: 1.008
[0, 208000] loss: 1.008
[0, 210000] loss: 1.008
[0, 212000] loss: 1.008
[0, 214000] loss: 1.008
[0, 216000] loss: 1.008
[0, 218000] loss: 1.008
[0, 220000] loss: 1.008
[0, 222000] loss: 1.008
[0, 224000] loss: 1.008
[0, 226000] loss: 1.008
[0, 228000] loss: 1.008
[0, 230000] loss: 1.008
[0, 232000] loss: 1.008
[0, 234000] loss: 1.008
[0, 236000] loss: 1.008
[0, 238000] loss: 1.008
[0, 240000] loss: 1.008
[0, 242000] loss: 1.008
[0, 244000] loss: 1.008
[0, 246000] loss: 1.008
[0, 248000] loss: 1.008
[0, 250000] loss: 1.008
[0, 252000] loss: 1.008
[0, 254000] loss: 1.008
[0, 256000] loss: 1.008
[0, 258000] loss: 1.008
[0, 260000] loss: 1.008
[0, 262000] loss: 1.008
[0, 264000] loss: 1.008
[0, 266000] loss: 1.008
[0, 268000] loss: 1.008
[0, 270000] loss: 1.008
[0, 272000] loss: 1.008
[0, 274000] loss: 1.008
[0, 276000] loss: 1.008
[0, 278000] loss: 1.008
[0, 280000] loss: 1.008
[0, 282000] loss: 1.008
[0, 284000] loss: 1.008
[0, 286000] loss: 1.008
[0, 288000] loss: 1.008
[0, 290000] loss: 1.008
[0, 292000] loss: 1.008
[0, 294000] loss: 1.008
[0, 296000] loss: 1.008
[0, 298000] loss: 1.008
[0, 300000] loss: 1.008
[0, 302000] loss: 1.008
[0, 304000] loss: 1.008
[0, 306000] loss: 1.008
[0, 308000] loss: 1.008
[0, 310000] loss: 1.008
[0, 312000] loss: 1.008
[0, 314000] loss: 1.008
[0, 316000] loss: 1.008
[0, 318000] loss: 1.008
[0, 320000] loss: 1.008
[0, 322000] loss: 1.008
[0, 324000] loss: 1.008
[0, 326000] loss: 1.008
[0, 328000] loss: 1.008
[0, 330000] loss: 1.008
[0, 332000] loss: 1.008
[0, 334000] loss: 1.008
[0, 336000] loss: 1.008
[0, 338000] loss: 1.008
[0, 340000] loss: 1.008
[0, 342000] loss: 1.008
[0, 344000] loss: 1.008
[0, 346000] loss: 1.008
[0, 348000] loss: 1.008
[0, 350000] loss: 1.008
[0, 352000] loss: 1.008
[0, 354000] loss: 1.008
[0, 356000] loss: 1.008
[0, 358000] loss: 1.008
[0, 360000] loss: 1.008
[0, 362000] loss: 1.008
[0, 364000] loss: 1.008
[0, 366000] loss: 1.008
[0, 368000] loss: 1.008
[0, 370000] loss: 1.008
[0, 372000] loss: 1.008
[0, 374000] loss: 1.008
[0, 376000] loss: 1.008
[0, 378000] loss: 1.008
[0, 380000] loss: 1.008
[0, 382000] loss: 1.008
[0, 384000] loss: 1.008
[0, 386000] loss: 1.008
[0, 388000] loss: 1.008
[0, 390000] loss: 1.008
[0, 392000] loss: 1.008
[0, 394000] loss: 1.008
[0, 396000] loss: 1.008
[0, 398000] loss: 1.008
[0, 400000] loss: 1.008
[0, 402000] loss: 1.008
[0, 404000] loss: 1.008
[0, 406000] loss: 1.008
[0, 408000] loss: 1.008
[0, 410000] loss: 1.008
[0, 412000] loss: 1.008
[0, 414000] loss: 1.008
[0, 416000] loss: 1.008
[0, 418000] loss: 1.008
[0, 420000] loss: 1.008
[0, 422000] loss: 1.008
[0, 424000] loss: 1.008
[0, 426000] loss: 1.008
[0, 428000] loss: 1.008
[0, 430000] loss: 1.008
[0, 432000] loss: 1.008
[0, 434000] loss: 1.008
[0, 436000] loss: 1.008
[0, 438000] loss: 1.008
[0, 440000] loss: 1.008
[0, 442000] loss: 1.008
[0, 444000] loss: 1.008
[0, 446000] loss: 1.008
[0, 448000] loss: 1.008
[0, 450000] loss: 1.008
[0, 452000] loss: 1.008
[0, 454000] loss: 1.008
[0, 456000] loss: 1.008
[0, 458000] loss: 1.008
[0, 460000] loss: 1.008
[0, 462000] loss: 1.008
[0, 464000] loss: 1.008
[0, 466000] loss: 1.008
[0, 468000] loss: 1.008
[0, 470000] loss: 1.008
[0, 472000] loss: 1.008
[0, 474000] loss: 1.008
[0, 476000] loss: 1.008
[0, 478000] loss: 1.008
[0, 480000] loss: 1.008
[0, 482000] loss: 1.008
[0, 484000] loss: 1.008
[0, 486000] loss: 1.008
[0, 488000] loss: 1.008
[0, 490000] loss: 1.008
[0, 492000] loss: 1.008
[0, 494000] loss: 1.008
[0, 496000] loss: 1.008
[0, 498000] loss: 1.008
[0, 500000] loss: 1.008
[0, 502000] loss: 1.008
[0, 504000] loss: 1.008
[0, 506000] loss: 1.008
[0, 508000] loss: 1.008
[0, 510000] loss: 1.008
[0, 512000] loss: 1.008
[0, 514000] loss: 1.008
[0, 516000] loss: 1.008
[0, 518000] loss: 1.008
[0, 520000] loss: 1.008
[0, 522000] loss: 1.008
[0, 524000] loss: 1.008
[0, 526000] loss: 1.008
[0, 528000] loss: 1.008
[0, 530000] loss: 1.008
[0, 532000] loss: 1.008
[0, 534000] loss: 1.008
[0, 536000] loss: 1.008
[0, 538000] loss: 1.008
[0, 540000] loss: 1.008
[0, 542000] loss: 1.008
[0, 544000] loss: 1.008
[0, 546000] loss: 1.008
[0, 548000] loss: 1.008
[0, 550000] loss: 1.008
[0, 552000] loss: 1.008
[0, 554000] loss: 1.008
[0, 556000] loss: 1.008
[0, 558000] loss: 1.008
[0, 560000] loss: 1.008
[0, 562000] loss: 1.008
[0, 564000] loss: 1.008
[0, 566000] loss: 1.008
[0, 568000] loss: 1.008
[0, 570000] loss: 1.008
[0, 572000] loss: 1.008
[0, 574000] loss: 1.008
[0, 576000] loss: 1.008
[0, 578000] loss: 1.008
[0, 580000] loss: 1.008
[0, 582000] loss: 1.008
[0, 584000] loss: 1.008
[0, 586000] loss: 1.008
[0, 588000] loss: 1.008
[0, 590000] loss: 1.008
[0, 592000] loss: 1.008
[0, 594000] loss: 1.008
[0, 596000] loss: 1.008
[0, 598000] loss: 1.008
[0, 600000] loss: 1.008
[0, 602000] loss: 1.008
[0, 604000] loss: 1.008
[0, 606000] loss: 1.008
[0, 608000] loss: 1.008
[0, 610000] loss: 1.008
[0, 612000] loss: 1.008
[0, 614000] loss: 1.008
[0, 616000] loss: 1.008
[0, 618000] loss: 1.008
[0, 620000] loss: 1.008
[0, 622000] loss: 1.008
[0, 624000] loss: 1.008
[0, 626000] loss: 1.008
[0, 628000] loss: 1.008
[0, 630000] loss: 1.008
[0, 632000] loss: 1.008
[0, 634000] loss: 1.008
[0, 636000] loss: 1.008
[0, 638000] loss: 1.008
[0, 640000] loss: 1.008
[0, 642000] loss: 1.008
[0, 644000] loss: 1.008
[0, 646000] loss: 1.008
[0, 648000] loss: 1.008
[0, 650000] loss: 1.008
[0, 652000] loss: 1.008
[0, 654000] loss: 1.008
[0, 656000] loss: 1.008
[0, 658000] loss: 1.008
[0, 660000] loss: 1.008
[0, 662000] loss: 1.008
[0, 664000] loss: 1.008
[0, 666000] loss: 1.008
[0, 668000] loss: 1.008
[0, 670000] loss: 1.008
[0, 672000] loss: 1.008
[0, 674000] loss: 1.008
[0, 676000] loss: 1.008
[0, 678000] loss: 1.008
[0, 680000] loss: 1.008
[0, 682000] loss: 1.008
[0, 684000] loss: 1.008
[0, 686000] loss: 1.008
[0, 688000] loss: 1.008
[0, 690000] loss: 1.008
[0, 692000] loss: 1.008
[0, 694000] loss: 1.008
[0, 696000] loss: 1.008
[0, 698000] loss: 1.008
[0, 700000] loss: 1.008
[0, 702000] loss: 1.008
[0, 704000] loss: 1.008
[0, 706000] loss: 1.008
[0, 708000] loss: 1.008
[0, 710000] loss: 1.008
[0, 712000] loss: 1.008
[0, 714000] loss: 1.008
[0, 716000] loss: 1.008
[0, 718000] loss: 1.008
[0, 720000] loss: 1.008
[0, 722000] loss: 1.008
[0, 724000] loss: 1.008
[0, 726000] loss: 1.008
[0, 728000] loss: 1.008
[0, 730000] loss: 1.008
[0, 732000] loss: 1.008
[0, 734000] loss: 1.008
[0, 736000] loss: 1.008
[0, 738000] loss: 1.008
[0, 740000] loss: 1.008
[0, 742000] loss: 1.008
[0, 744000] loss: 1.008
[0, 746000] loss: 1.008
[0, 748000] loss: 1.008
[0, 750000] loss: 1.008
[0, 752000] loss: 1.008
[0, 754000] loss: 1.008
[0, 756000] loss: 1.008
[0, 758000] loss: 1.008
[0, 760000] loss: 1.008
[0, 762000] loss: 1.008
[0, 764000] loss: 1.008
[0, 766000] loss: 1.008
[0, 768000] loss: 1.008
[0, 770000] loss: 1.008
[0, 772000] loss: 1.008
[0, 774000] loss: 1.008
[0, 776000] loss: 1.008
[0, 778000] loss: 1.008
[0, 780000] loss: 1.008
[0, 782000] loss: 1.008
[0, 784000] loss: 1.008
[0, 786000] loss: 1.008
[0, 788000] loss: 1.008
[0, 790000] loss: 1.008
[0, 792000] loss: 1.008
[0, 794000] loss: 1.008
[0, 796000] loss: 1.008
[0, 798000] loss: 1.008
[0, 800000] loss: 1.008
[0, 802000] loss: 1.008
[0, 804000] loss: 1.008
[0, 806000] loss: 1.008
[0, 808000] loss: 1.008
[0, 810000] loss: 1.008
[0, 812000] loss: 1.008
[0, 814000] loss: 1.008
[0, 816000] loss: 1.008
[0, 818000] loss: 1.008
[0, 820000] loss: 1.008
[0, 822000] loss: 1.008
[0, 824000] loss: 1.008
[0, 826000] loss: 1.008
[0, 828000] loss: 1.008
[0, 830000] loss: 1.008
[0, 832000] loss: 1.008
[0, 834000] loss: 1.008
[0, 836000] loss: 1.008
[0, 838000] loss: 1.008
[0, 840000] loss: 1.008
[0, 842000] loss: 1.008
[0, 844000] loss: 1.008
[0, 846000] loss: 1.008
[0, 848000] loss: 1.008
[0, 850000] loss: 1.008
[0, 852000] loss: 1.008
[0, 854000] loss: 1.008
[0, 856000] loss: 1.008
[0, 858000] loss: 1.008
[0, 860000] loss: 1.008
[0, 862000] loss: 1.008
[0, 864000] loss: 1.008
[0, 866000] loss: 1.008
[0, 868000] loss: 1.008
[0, 870000] loss: 1.008
[0, 872000] loss: 1.008
[0, 874000] loss: 1.008
[0, 876000] loss: 1.008
[0, 878000] loss: 1.008
[0, 880000] loss: 1.008
[0, 882000] loss: 1.008
[0, 884000] loss: 1.008
[0, 886000] loss: 1.008
[0, 888000] loss: 1.008
[0, 890000] loss: 1.008
[0, 892000] loss: 1.008
[0, 894000] loss: 1.008
[0, 896000] loss: 1.008
[0, 898000] loss: 1.008
[0, 900000] loss: 1.008
[0, 902000] loss: 1.008
[0, 904000] loss: 1.008
[0, 906000] loss: 1.008
[0, 908000] loss: 1.008
[0, 910000] loss: 1.008
[0, 912000] loss: 1.008
[0, 914000] loss: 1.008
[0, 916000] loss: 1.008
[0, 918000] loss: 1.008
[0, 920000] loss: 1.008
[0, 922000] loss: 1.008
[0, 924000] loss: 1.008
[0, 926000] loss: 1.008
[0, 928000] loss: 1.008
[0, 930000] loss: 1.008
[0, 932000] loss: 1.008
[0, 934000] loss: 1.008
[0, 936000] loss: 1.008
[0, 938000] loss: 1.008
[0, 940000] loss: 1.008
[0, 942000]
```