10

January 23, 2024

[1]:
```
%pylab inline
import torch
import sys
sys.path.append('..')
sys.path.append('../..')
from data import load
train_data, train_label = load.get_dogs_and_cats_data(resize=(128,128),␣
 ↪n_images=10)
device = torch.device('cuda') if torch.cuda.is_available() else torch.
 ↪device('cpu')
print('device = ', device)
```

```
%pylab is deprecated, use %matplotlib inline and import the required libraries.
Populating the interactive namespace from numpy and matplotlib
device =  cuda
```

[2]:
```
class ConvNet(torch.nn.Module):
    class Block(torch.nn.Module):
        def __init__(self, n_input, n_output, stride=1):
            super().__init__()
            self.net = torch.nn.Sequential(
              torch.nn.Conv2d(n_input, n_output, kernel_size=3, padding=1,␣
 ↪stride=stride),
              torch.nn.ReLU(),
              torch.nn.Conv2d(n_output, n_output, kernel_size=3, padding=1),
              torch.nn.ReLU()
            )

        def forward(self, x):
            return self.net(x)

    def __init__(self, layers=[32,64,128], n_input_channels=3):
        super().__init__()
        L = [torch.nn.Conv2d(n_input_channels, 32, kernel_size=7, padding=3,␣
 ↪stride=2),
             torch.nn.ReLU(),
             torch.nn.MaxPool2d(kernel_size=3, stride=2, padding=1)]
```

```
        c = 32
        for l in layers:
            L.append(self.Block(c, l, stride=2))
            c = l
        self.network = torch.nn.Sequential(*L)
        self.classifier = torch.nn.Linear(c, 1)

    def forward(self, x):
        # Compute the features
        z = self.network(x)
        # Global average pooling
        z = z.mean(dim=[2,3])
        # Classify
        return self.classifier(z)[:,0]

net = ConvNet()
```

[3]:
```
z = net(train_data[:1])
print(z)
```

```
tensor([0.0218], grad_fn=<SelectBackward0>)
```

[4]:
```
%load_ext tensorboard
import tempfile
log_dir = tempfile.mkdtemp()
%tensorboard --logdir {log_dir} --reload_interval 1
```

```
<IPython.core.display.HTML object>
```

[5]:
```
from util import train
train.train(net, batch_size=128, resize=(128,128), log_dir=log_dir,␣
 ↪device=device, n_epochs=100)
```

```
WARNING:root:loading dataset
WARNING:root:loading done
```

```
  0%|          | 0/100 [00:00<?, ?it/s]
```

[ ]:
```

```