**Pytorch 获取网络权重参数、每一层权重参数**

import torch

import torch.nn as nn

class Net(torch.nn.Module):

def \_\_init\_\_(self):

super(Net, self).\_\_init\_\_()

# kernel

# 输入是1个通道的灰度图，输出6个通道(feature map)，使用5x5的卷积核

self.conv1 = nn.Conv2d(1, 6, 5)

# 第二个卷积层也是5x5，有16个通道

self.conv2 = nn.Conv2d(6, 16, 5)

# 全连接层

# an affine operation: y = Wx + b

self.fc1 = nn.Linear(in\_features=16 \* 5 \* 5, out\_features=120)

self.fc2 = nn.Linear(in\_features=120, out\_features=84)

self.fc3 = nn.Linear(in\_features=84, out\_features=10)

if \_\_name\_\_ == '\_\_main\_\_':

model = Net()

# 1、查看每层对应的名称

print("#---------------- 查看每层对应的名称 ----------------#")

# 方式1：

for name in model.state\_dict():

print(name)

# 方式2：推荐★

# print(model)

print("#---------------- 可训练参数个数 ----------------#")

parameters = list(model.parameters())

print("可训练参数个数 = ", len(parameters))

# 2、输出相应层的权重

print("#---------------- 输出相应层的权重 ----------------#")

# 方式1：

# print(model.state\_dict()['conv1.weight'])

print(model.state\_dict()['conv1.bias'])

# 方式2：推荐★

# print(model.conv1.weight.grad)

# print(model.conv1.bias.grad)

# 3、打印模块名字和参数大小

print("#---------------- 打印模块名字和参数大小 ----------------#")

for name, parameters in model.named\_parameters():

print(name, ';', parameters.size())

# print(name, ';', parameters, '\n')

输出：

#---------------- 查看每层对应的名称 ----------------#

conv1.weight

conv1.bias

conv2.weight

conv2.bias

fc1.weight

fc1.bias

fc2.weight

fc2.bias

fc3.weight

fc3.bias

#---------------- 可训练参数个数 ----------------#

可训练参数个数 = 10

#---------------- 输出相应层的权重 ----------------#

tensor([-0.0433, 0.1646, 0.0730, -0.0692, -0.0767, -0.1820])

#---------------- 打印模块名字和参数大小 ----------------#

conv1.weight ; torch.Size([6, 1, 5, 5])

conv1.bias ; torch.Size([6])

conv2.weight ; torch.Size([16, 6, 5, 5])

conv2.bias ; torch.Size([16])

fc1.weight ; torch.Size([120, 400])

fc1.bias ; torch.Size([120])

fc2.weight ; torch.Size([84, 120])

fc2.bias ; torch.Size([84])

fc3.weight ; torch.Size([10, 84])

fc3.bias ; torch.Size([10])