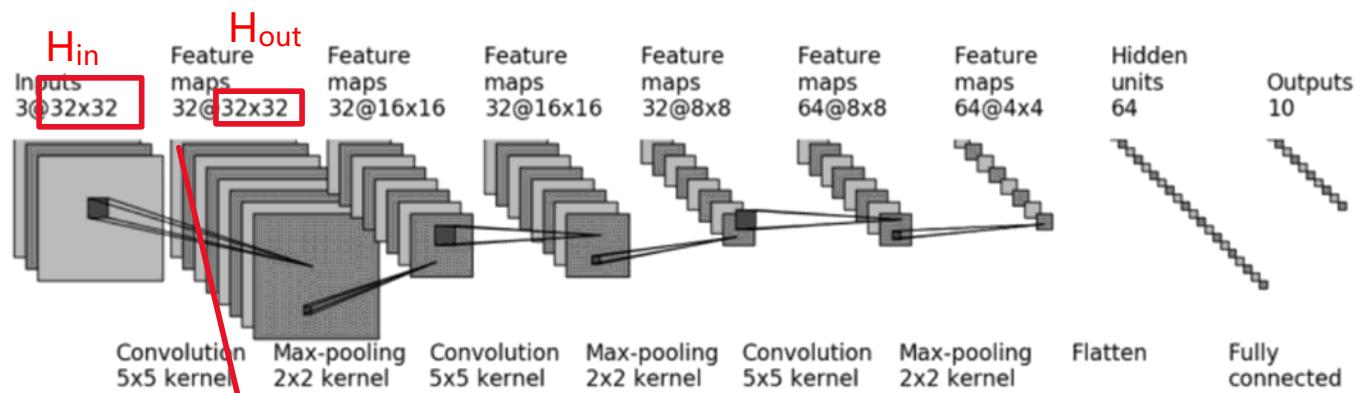


小实战—搭建cifar10神经网络

2025年11月23日 13:48



代入公式去算, Hin, Hout, dilation默认1, kernel_size5, stride默认1

Shape:

- Input: $(N, C_{in}, H_{in}, W_{in})$ or (C_{in}, H_{in}, W_{in})
- Output: $(N, C_{out}, H_{out}, W_{out})$ or $(C_{out}, H_{out}, W_{out})$, where

$H_{out} = \left\lfloor \frac{H_{in} + 2 \times \text{padding}[0] - \text{dilation}[0] \times (\text{kernel_size}[0] - 1) - 1}{\text{stride}[0]} + 1 \right\rfloor$

$W_{out} = \left\lfloor \frac{W_{in} + 2 \times \text{padding}[1] - \text{dilation}[1] \times (\text{kernel_size}[1] - 1) - 1}{\text{stride}[1]} + 1 \right\rfloor$

padding是2

```
12     self.maxpool3 = nn.MaxPool2d(2)
13     self.flatten = Flatten()
14     self.linear = Linear(in_features: 1024, out_features: 64)
15     self.linear2 = Linear(in_features: 64, out_features: 10)
16
17     def forward(self,x):
18         x = self.conv1(x)
19         x = self.maxpool(x)
20         x = self.conv2(x)
21         x = self.maxpool(x)
22         x = self.conv3(x)
23         x = self.maxpool(x)
24         x = self.flatten(x)
25         x = self.linear(x)
26         x = self.linear2(x)
27
28     Solmount = solmount()
29     print(Solmount)
```

运行 CIFAR10_construction

```
D:\anaconda3\envs\pytorch\python.exe "D:\pycharm_space\PycharmProjects\PythonProject1\nurse_net\CIFAR10_construction.py"
solmount(
(conv1): Conv2d(3, 32, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
(maxpool): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
(conv2): Conv2d(32, 32, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
(maxpool2): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
(conv3): Conv2d(32, 64, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
(maxpool3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
(flatten): Flatten(start_dim=1, end_dim=-1)
```

Print 以后，可以看这个网络中含有哪些层

验证

使用ones，创建一个想要的数据格式，输入进网络，看他输出来什么东西

The screenshot shows a PyCharm IDE interface. On the left, there's a tree view with nodes: Solmount, input, and output. The main window displays Python code for the Solmount class. Lines 27 through 34 are shown:

```
    27     return x
    28
    29     Solmount = solmount()
    30     print(Solmount)
    31     input = torch.ones(64,3,32,32)
    32     output = Solmount(input)
    33     print(output)
    34     print(output.shape)
```

Line 31 and line 34 are highlighted with red boxes. The output pane below shows the execution results:

```
行   CIFAR10_construction ×
:
      [-0.0301, -0.1075],
      [-0.0646, -0.0602, -0.0224,  0.1151, -0.0278, -0.0958, -0.1184,  0.0246,
       -0.0301, -0.1075],
      [-0.0646, -0.0602, -0.0224,  0.1151, -0.0278, -0.0958, -0.1184,  0.0246,
       0.0301,  0.1075]], grad_fn=<AddmmBackward0>
torch.Size([64, 10])
```

这只是搭建了cifar10的基本框架，离真正训练，还有一段的距离