深度学习快速入门

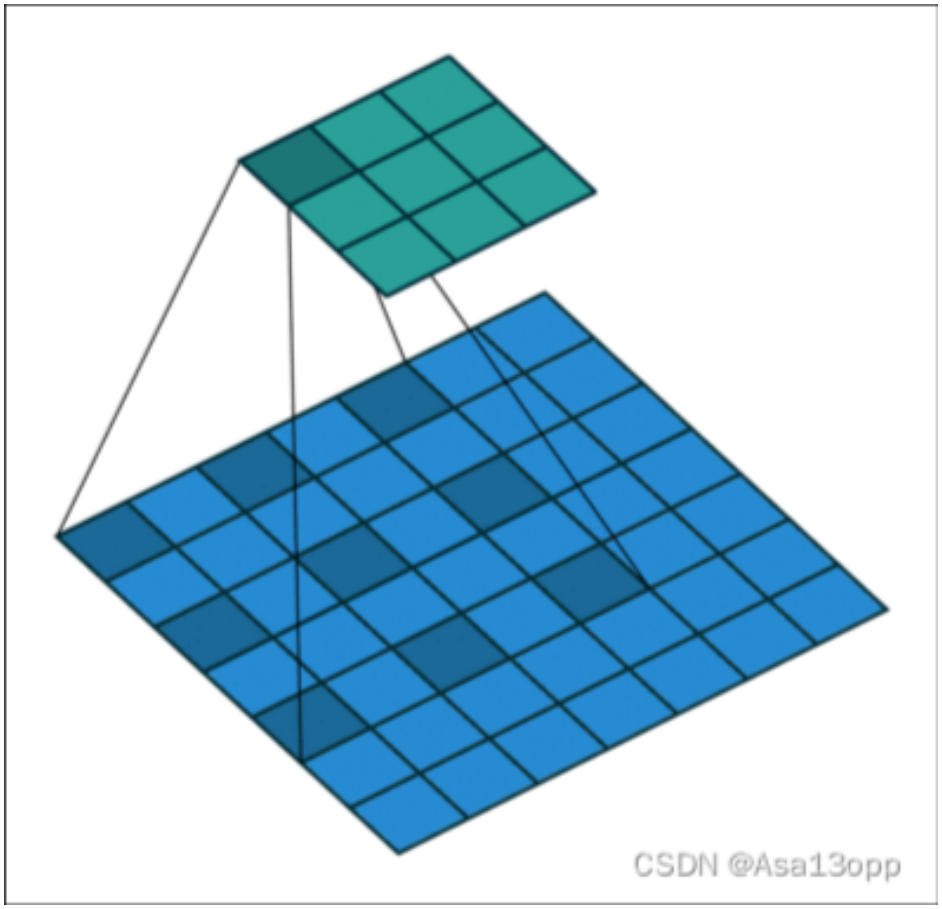
09 池化层

POET

2024 年 2 月 13 日

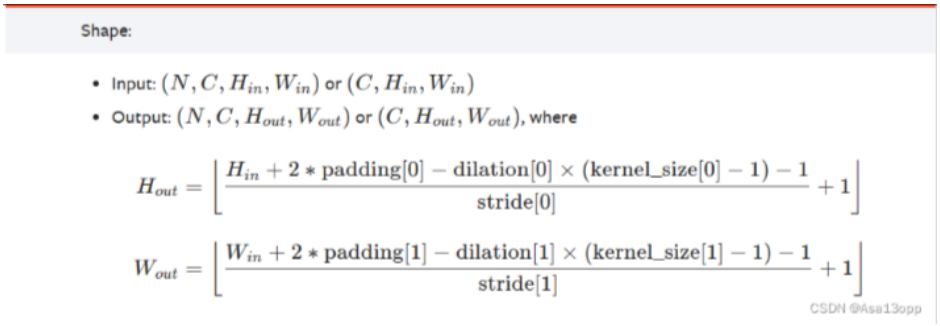
1 池化层原理

1. 最大池化层有时也被称为下采样。
2. dilation 为空洞卷积，如下图所示。
3. Ceil_ model 为当超出区域时，只取最左上角的值。
4. 池化使得数据由 5 * 5 变为 3 * 3, 甚至 1 * 1 的，这样导致计算的参数会大大减小。例如 1080P 的电影经过池化的转为 720P 的电影、或 360P 的电影后，同样的网速下，视频更为不卡。



2 torch.nn.MaxPool2d

```
torch.nn.MaxPool2d(kernel_size, stride=None, padding=0, dilation=1,

return_indices=False, ceil_mode=False)

    kernel_size (Union[int, Tuple[int, int]]) - the size of the window to take a max over

    stride (Union[int, Tuple[int, int]]) - the stride of the window. Default value is kernel_size

    padding (Union[int, Tuple[int, int]]) - Implicit negative infinity padding to be added on both sides

    dilation (Union[int, Tuple[int, int]]) - a parameter that controls the stride of elements in the window

    return_indices (bool) - if True, will return the max indices along with the outputs. Useful for torch.nn.MaxUnpool2d later

    ceil_mode (bool) - when True, will use ceil instead of floor to compute the output shape

    当 ceil_mode=True 时，如果滑动窗口在左填充内边距内启动，则允许它们越界或输入。将在右侧填充区域开始的滑动窗口将被忽略。
```



Shape:

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

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

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

3 池化层处理图片

```
import torch
import torchvision
from torch import nn
from torch.nn import MaxPool2d
from torch.utils.data import DataLoader
from torch.utils.tensorboard import SummaryWriter

dataset = torchvision.datasets.CIFAR10("./dataset",train=False,transform=torchvision.transforms.ToTensor(),download=True)
dataloader = DataLoader(dataset, batch_size=64)

class Tudui(nn.Module):
    def __init__(self):
        super(Tudui, self).__init__()
        self.maxpool = MaxPool2d(kernel_size=3, ceil_mode=True)

    def forward(self, input):
        output = self.maxpool(input)
        return output
```

```
19
20      tudui = Tudui()
21      writer = SummaryWriter("logs")
22      step = 0
23
24      for data in dataloader:
25          imgs, targets = data
26          writer.add_images("input", imgs, step)
27          output = tudui(imgs)
28          writer.add_images("output", output, step)
29          step = step + 1
```