

## DCNN

生理学启发 局部特征 软件模块周围神经元附近的输出

结构  $\rightarrow$  简单 Cell + 复杂 Cell

局部感受野 (局部性)

视场中自适应感受野 (权值共享)

全局下采样率 (2到3) Conv核kernel对局部像素加权求和

Sobel算子用于边缘检测 作用: 局部算子; 平移同质性; 特征增强; 去噪  
数学上, 给定  $x$  输入及滤波器  $W$  有输出特征:  $y = \sum x * W$

特征

Conv

Pooling

Non-Linear Activation

Normalise

FC / classifier

Poolinging!

又称为下采样 / 池化  $\rightarrow$  降低特征图空间分辨率

剪枝

常用方法: Max pooling, Average pooling

通道数不变, 不利用参数作用于局部化不重要

(特征多样性); 增大感受野; 不依赖于输入, 而应完全相同

防止过拟合

按步长=2进行2x2的最大池化

若: 池化步长=1

$\begin{bmatrix} 1 & 0 & 1 & 0 \\ 2 & 1 & 3 & 2 \\ 1 & 2 & 0 & 4 \\ 2 & 2 & 3 & 0 \end{bmatrix}$

$\begin{bmatrix} 6 & 8 \\ 3 & 4 \end{bmatrix}$

max pool with 2x2 filters and stride 2

输出:  $\begin{bmatrix} 6 & 8 \\ 3 & 4 \end{bmatrix}$

按步长=2进行2x2的最大池化

若: 池化步长=1

$\begin{bmatrix} 1 & 0 & 1 & 0 \\ 2 & 1 & 3 & 2 \\ 1 & 2 & 0 & 4 \\ 2 & 2 & 3 & 0 \end{bmatrix}$

$\begin{bmatrix} 6 & 8 \\ 3 & 4 \end{bmatrix}$

max pool with 2x2 filters and stride 2

输出:  $\begin{bmatrix} 6 & 8 \\ 3 & 4 \end{bmatrix}$

按步长=2进行2x2的最大池化

若: 池化步长=1

$\begin{bmatrix} 1 & 0 & 1 & 0 \\ 2 & 1 & 3 & 2 \\ 1 & 2 & 0 & 4 \\ 2 & 2 & 3 & 0 \end{bmatrix}$

$\begin{bmatrix} 6 & 8 \\ 3 & 4 \end{bmatrix}$

max pool with 2x2 filters and stride 2

输出:  $\begin{bmatrix} 6 & 8 \\ 3 & 4 \end{bmatrix}$

优势: 局部连接, 参数减少

权值共享, 再减少

在不同位置工作原理同  
响应随目标位置变化而变化

Conv核kernel对局部像素加权求和  
Sobel算子用于边缘检测 作用: 局部算子; 平移同质性; 特征增强; 去噪  
数学上, 给定  $x$  输入及滤波器  $W$  有输出特征:

$y = \sum x * W$

超参数 ① kernel size 卷积核和场大小

② Stride  $W$  在  $x$  上的步进数

③ Padding 边缘处理

一个简单的例子

• 3x3 的滤波器作用于5x5的输入

• 思考: 当 padding 和 kernel size 满足什么关系时, 卷积前后输入的尺寸不变?

• 试计算 C, D 和 E 的卷积结果

与 FeatureMap 数与卷积核数量同

即输出长宽  $Hout = \frac{Hin + 2 * padding - dilation * (kernel size - 1)}{stride} + 1$

即输出长宽  $Hout = \frac{Hin + 2 * padding - kernel size}{stride} + 1$

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

## CNN典型架构

### ① LeNet

LeNet-5 Document Recognition System

INPUT IMAGE

C1: 16 maps 10x10x1

S2: 1 map 5x5x16

S3: 16 maps 5x5x1

F4: 120 maps 1x1x16

F5: 10 maps 1x1x120

OUTPUT

Gaussian connections

• 平均池化

• 全连接层是用于分类

• 在60,000个训练样本的MNIST手写体识别数据集上训练

### ② AlexNet

CNN典型架构之 AlexNet

2012 ImageNet 比赛冠军!

与LeNet结构相似, 但:

• 更大的模型 (8层)

• 采用ReLU激活函数

• 更多的数据 ( $10^6$  vs  $10^3$  images)

• 更好的正则化 (DropOut)

• GPU实现 (超过CPU 50倍的加速)

• 7个隐含层, 650,000 神经元, 60,000,000 参数

• 在2块GPUs上训练一周

与LeNet结构相似, 但:

• 更大的模型 (8层)

• 采用ReLU激活函数

• 更多的数据 ( $10^6$  vs  $10^3$  images)

• 更好的正则化 (DropOut)

• GPU实现 (超过CPU 50倍的加速)

• 7个隐含层, 650,000 神经元, 60,000,000 参数

• 在2块GPUs上训练一周

与LeNet结构相似, 但:

• 更大的模型 (8层)

• 采用ReLU激活函数

• 更多的数据 ( $10^6$  vs  $10^3$  images)

• 更好的正则化 (DropOut)

• GPU实现 (超过CPU 50倍的加速)

• 7个隐含层, 650,000 神经元, 60,000,000 参数

• 在2块GPUs上训练一周

与LeNet结构相似, 但:

• 更大的模型 (8层)

• 采用ReLU激活函数

• 更多的数据 ( $10^6$  vs  $10^3$  images)

• 更好的正则化 (DropOut)

• GPU实现 (超过CPU 50倍的加速)

• 7个隐含层, 650,000 神经元, 60,000,000 参数

• 在2块GPUs上训练一周

与LeNet结构相似, 但:

• 更大的模型 (8层)

• 采用ReLU激活函数

• 更多的数据 ( $10^6$  vs  $10^3$  images)

• 更好的正则化 (DropOut)

• GPU实现 (超过CPU 50倍的加速)

• 7个隐含层, 650,000 神经元, 60,000,000 参数

• 在2块GPUs上训练一周

与LeNet结构相似, 但:

• 更大的模型 (8层)

• 采用ReLU激活函数

• 更多的数据 ( $10^6$  vs  $10^3$  images)

• 更好的正则化 (DropOut)

• GPU实现 (超过CPU 50倍的加速)

• 7个隐含层, 650,000 神经元, 60,000,000 参数

• 在2块GPUs上训练一周

与LeNet结构相似, 但:

• 更大的模型 (8层)

• 采用ReLU激活函数

• 更多的数据 ( $10^6$  vs  $10^3$  images)

• 更好的正则化 (DropOut)

• GPU实现 (超过CPU 50倍的加速)

• 7个隐含层, 650,000 神经元, 60,000,000 参数

• 在2块GPUs上训练一周

与LeNet结构相似, 但:

• 更大的模型 (8层)

• 采用ReLU激活函数

• 更多的数据 ( $10^6$  vs  $10^3$  images)

• 更好的正则化 (DropOut)

• GPU实现 (超过CPU 50倍的加速)

• 7个隐含层, 650,000 神经元, 60,000,000 参数

• 在2块GPUs上训练一周

与LeNet结构相似, 但:

• 更大的模型 (8层)

• 采用ReLU激活函数

• 更多的数据 ( $10^6$  vs  $10^3$  images)

• 更好的正则化 (DropOut)

• GPU实现 (超过CPU 50倍的加速)

• 7个隐含层, 650,000 神经元, 60,000,000 参数

• 在2块GPUs上训练一周

与LeNet结构相似, 但:

• 更大的模型 (8层)

• 采用ReLU激活函数

• 更多的数据 ( $10^6$  vs  $10^3$  images)

• 更好的正则化 (DropOut)

• GPU实现 (超过CPU 50倍的加速)

• 7个隐含层, 650,000 神经元, 60,000,000 参数

• 在2块GPUs上训练一周

与LeNet结构相似, 但:

• 更大的模型 (8层)

• 采用ReLU激活函数

• 更多的数据 ( $10^6$  vs  $10^3$  images)

• 更好的正则化 (DropOut)

• GPU实现 (超过CPU 50倍的加速)

• 7个隐含层, 650,000 神经元, 60,000,000 参数

• 在2块GPUs上训练一周

与LeNet结构相似, 但: