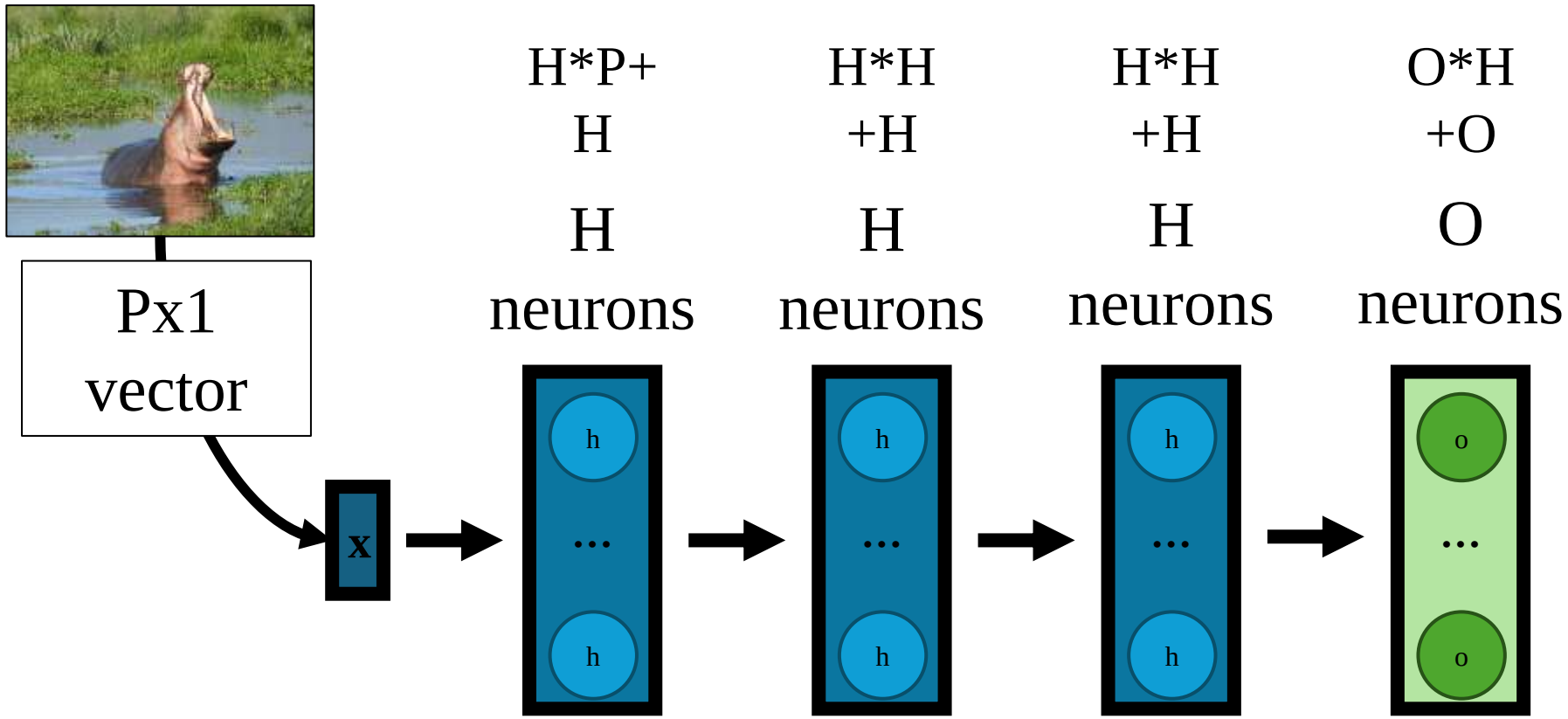
回顾：卷积神经网络

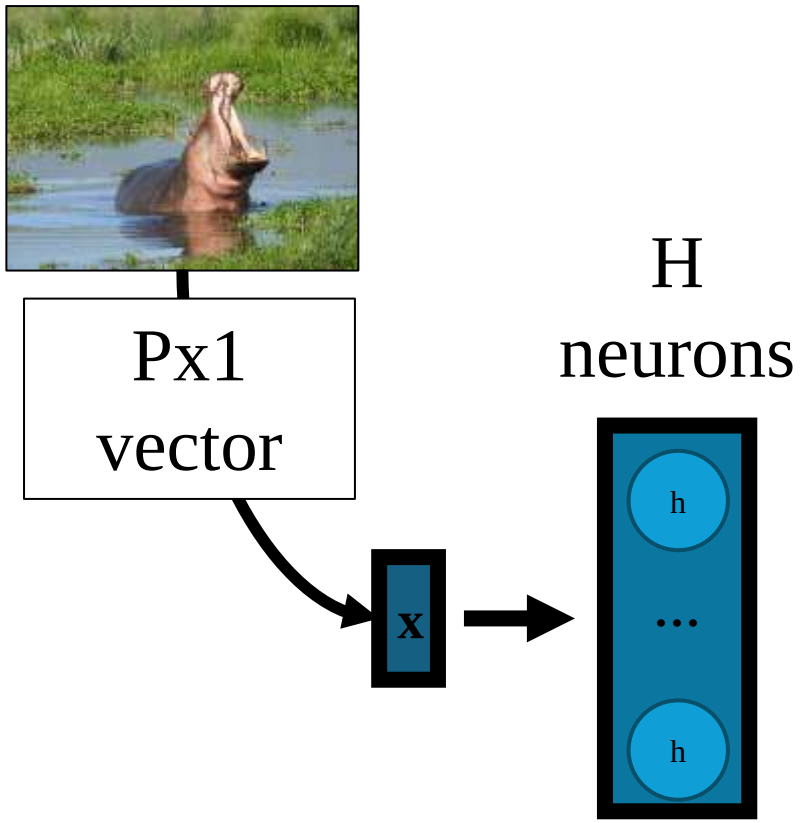
参数量



P: 285x350 , H: 1000, O: 3

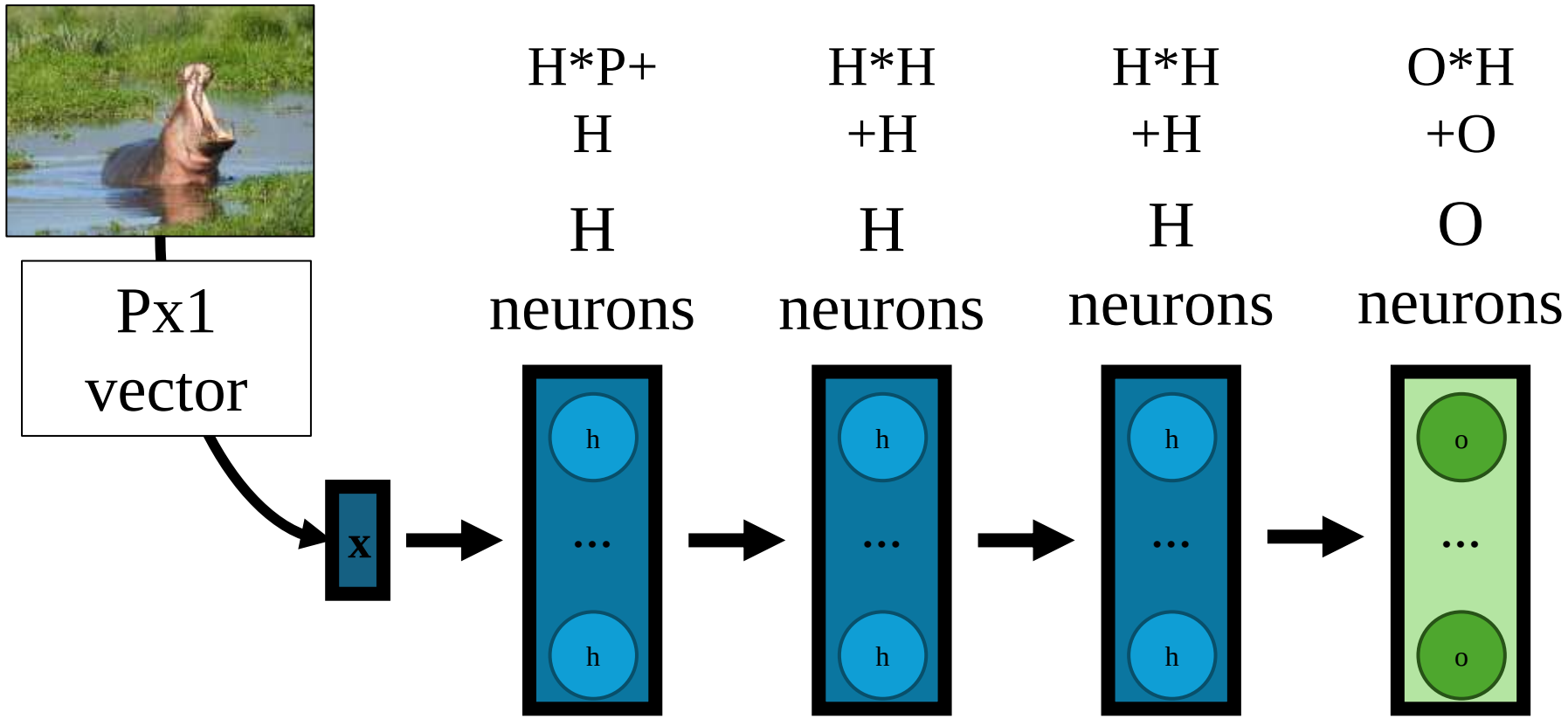
102 million 个 参数 (400MB)

参数量



- H 决定了网络的大小
- 第一层的参数量是P*H+H
- 如果我们要计算图像梯度dx/dy.**我们需要多少H?**
- **2P!**

参数量



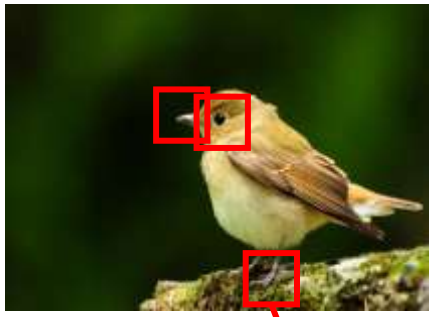
P: 285x350, H: 2P, O: 3

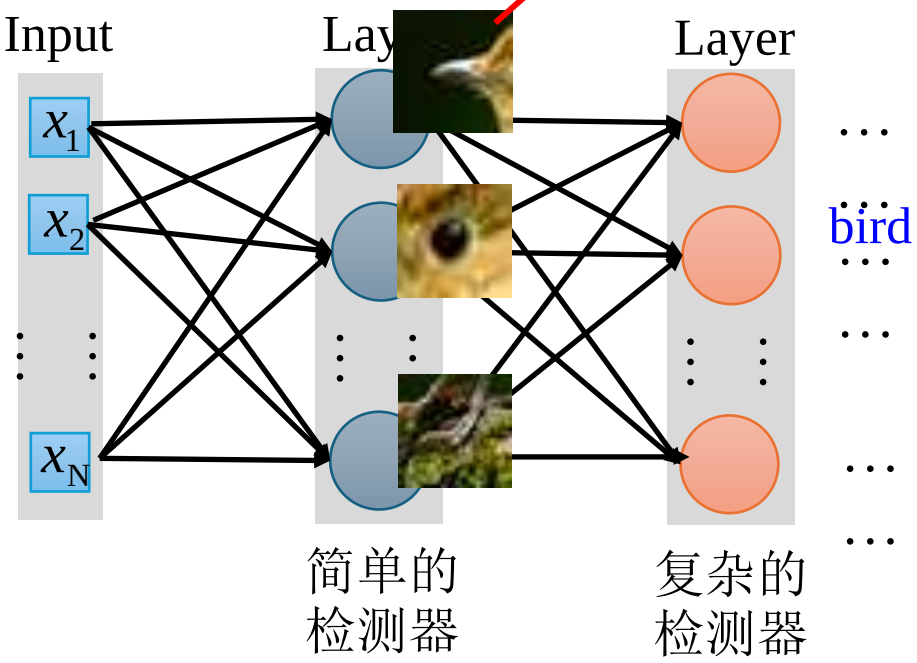
100 billion 个 参数 (400GB)

简化全连接层

每个神经元并不需要看到所有像素点（回顾特征尺度）

我们需要看整张图来分类吗？（回顾特征点）





有些关键特征也许很小（回顾特征尺度）

神经元通常只对图像中的局部模式或特征进行响应，而不是整张图像。

我们可以可视化滤波来推断他们的功能

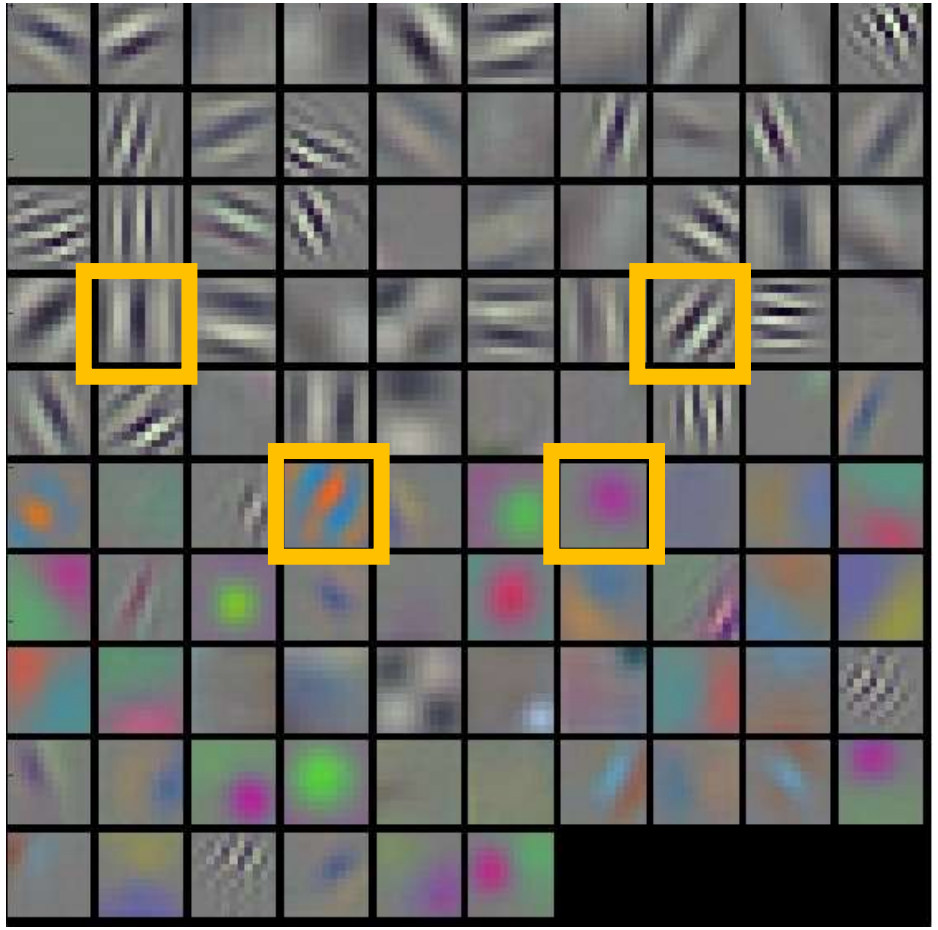


Figure Credit: Karpathy and Fei-Fei

每一层的输出

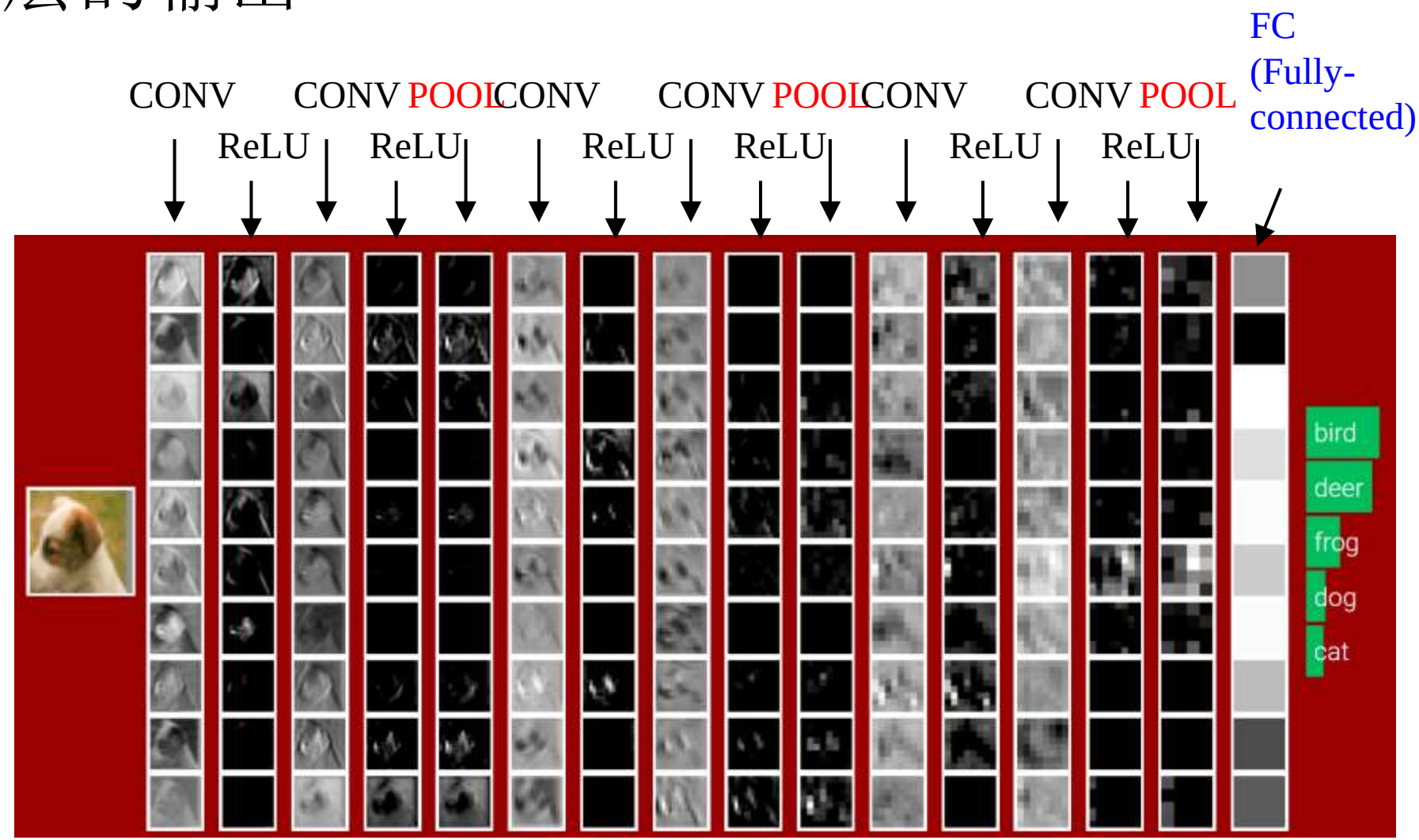
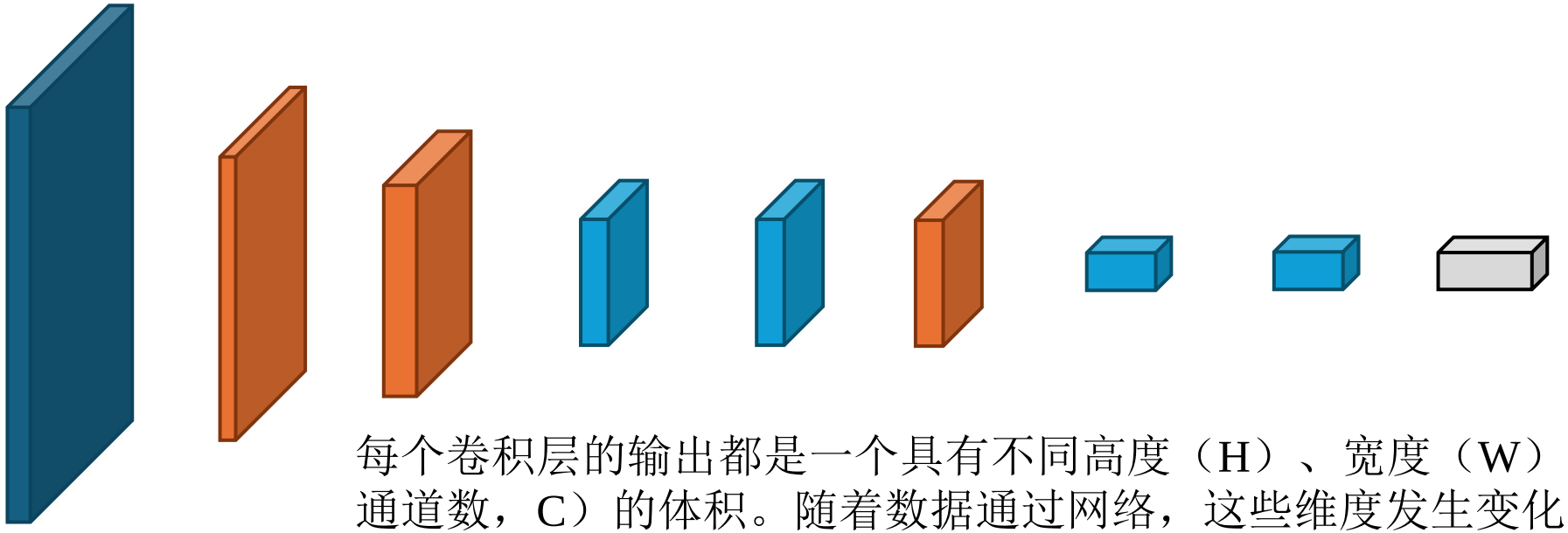


Figure Credit: Karpathy and Fei-Fei; see http://cs231n.stanford.edu/

AlexNet

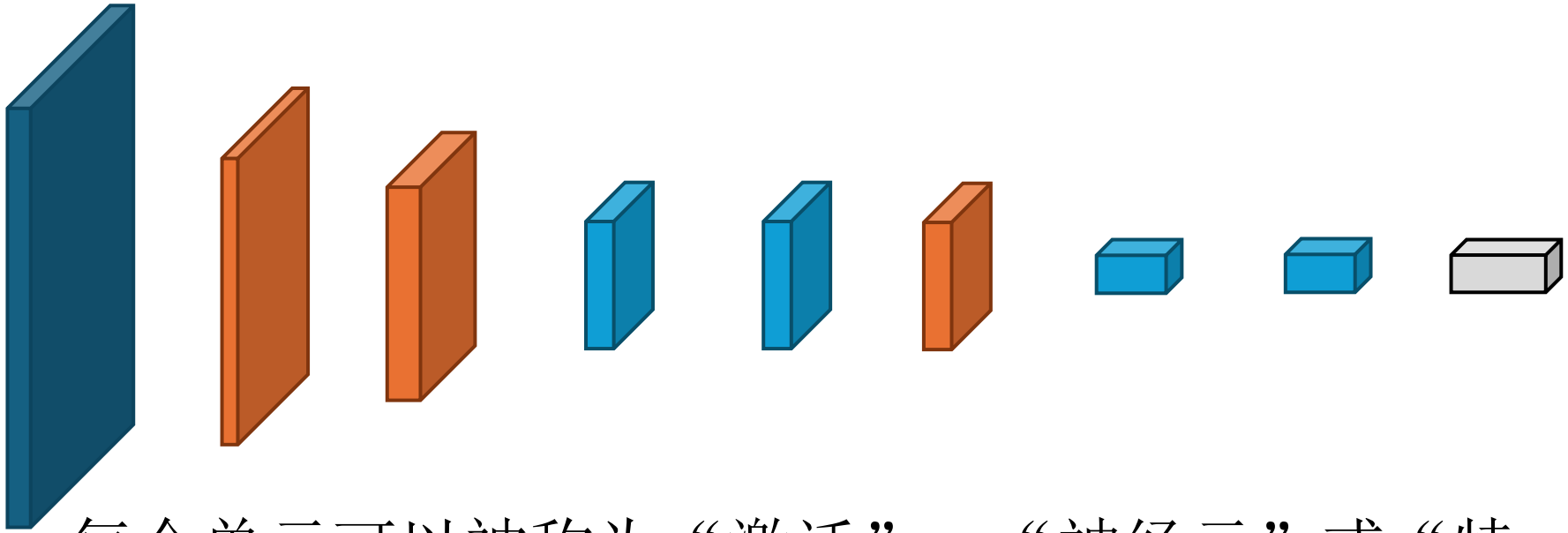
Input	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	FC 6	FC 7	Output
227x227 3	55x55 96	27x27 256	13x13 384	13x13 384	13x13 256	1x1 4096	1x1 4096	1x1 1000



每个卷积层的输出都是一个具有不同高度（H）、宽度（W）、和深度（即通道数，C）的体积。随着数据通过网络，这些维度发生变化，最终导致全连接层，并得到最终的输出。每个块（block）表示数据在网络中流动时的体积，通过卷积操作将一个体积变换到另一个体积。

CNN基本术语

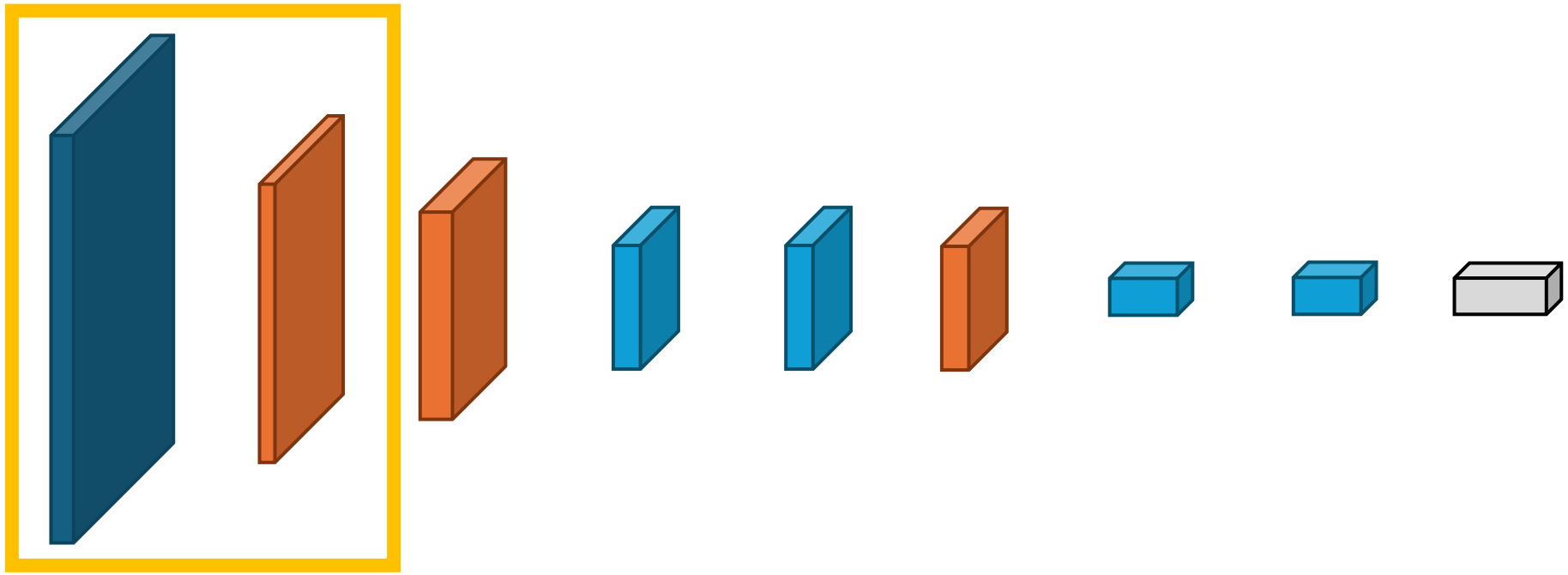
Input	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	FC 6	FC 7	Output
227x227 3	55x55 96	27x272 56	13x13 384	13x13 384	13x13 256	1x1 4096	1x1 4096	1x1 1000



每个单元可以被称为“激活”、“神经元”或“特征”。

AlexNet

Input	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	FC 6	FC 7	Output
227x227 3	55x55 96	27x27 256	13x13 384	13x13 384	13x13 256	1x1 4096	1x1 4096	1x1 1000



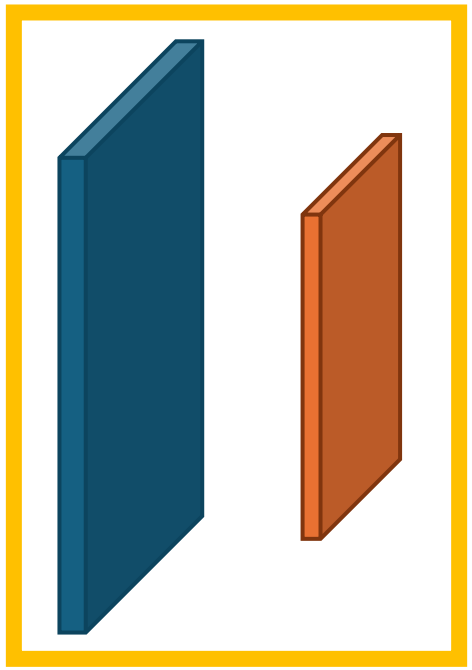
AlexNet

Input

227x227
3

Conv
1

55x55
96

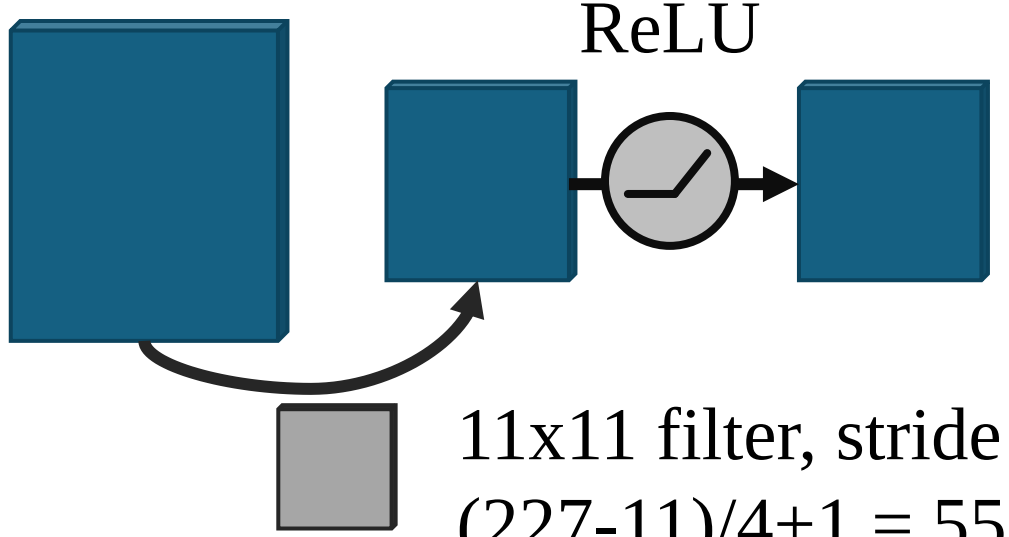


227x227
3

55x55
96

ReLU

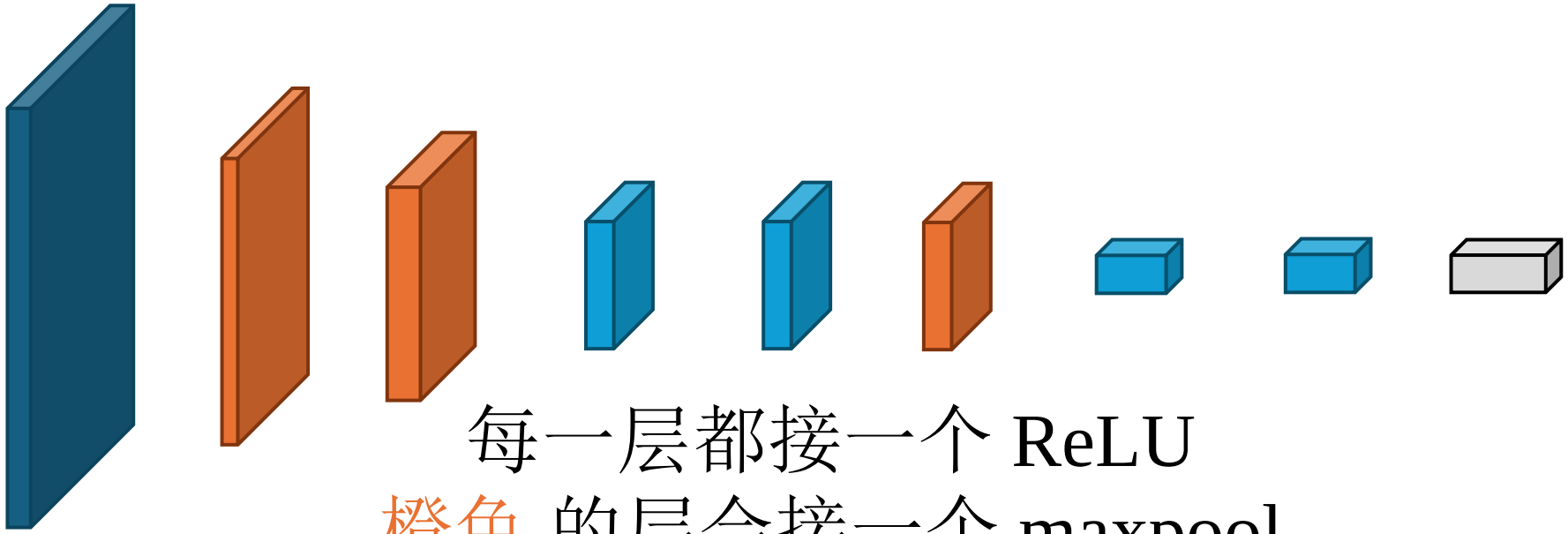
55x55
96



11x11 filter, stride of 4
(227-11)/4+1 = 55

AlexNet

Input	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	FC 6	FC 7	Output
227x227 3	55x55 96	27x27 256	13x13 384	13x13 384	13x13 256	1x1 4096	1x1 4096	1x1 1000



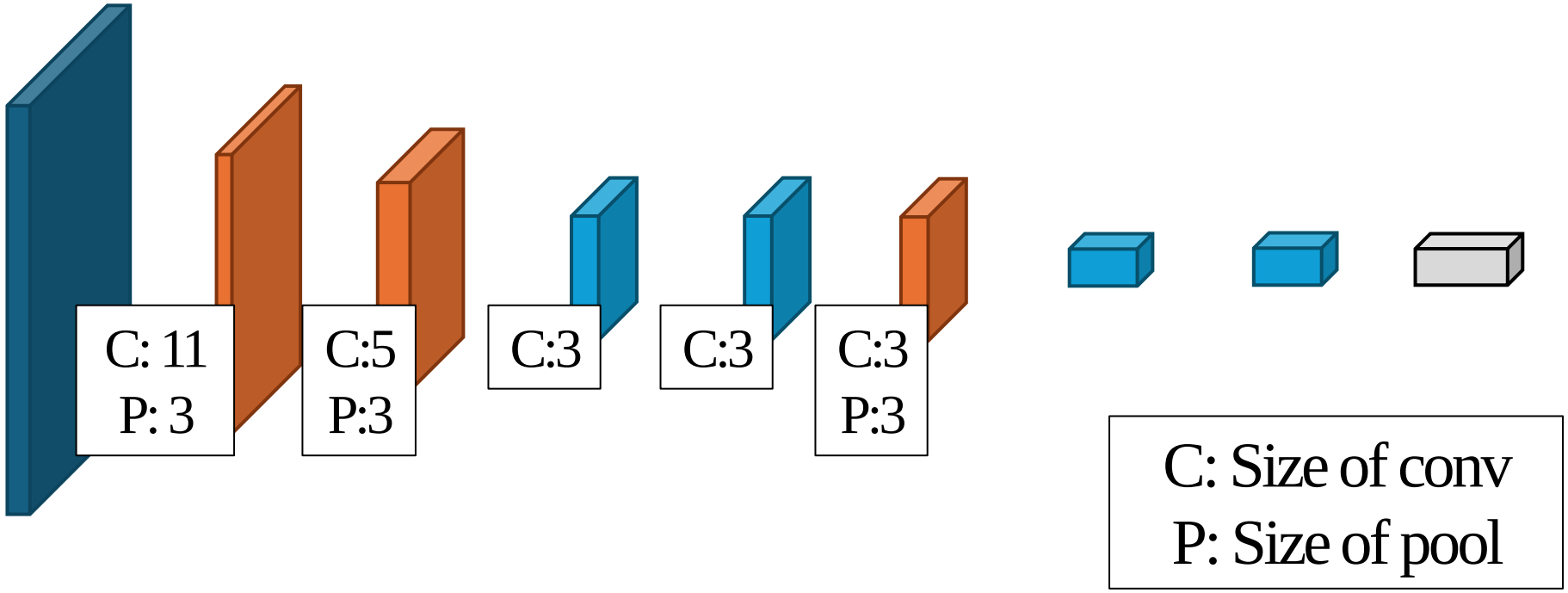
每一层都接一个 ReLU

橙色 的层会接一个 maxpool

每一层都有 “normalization” 标准化

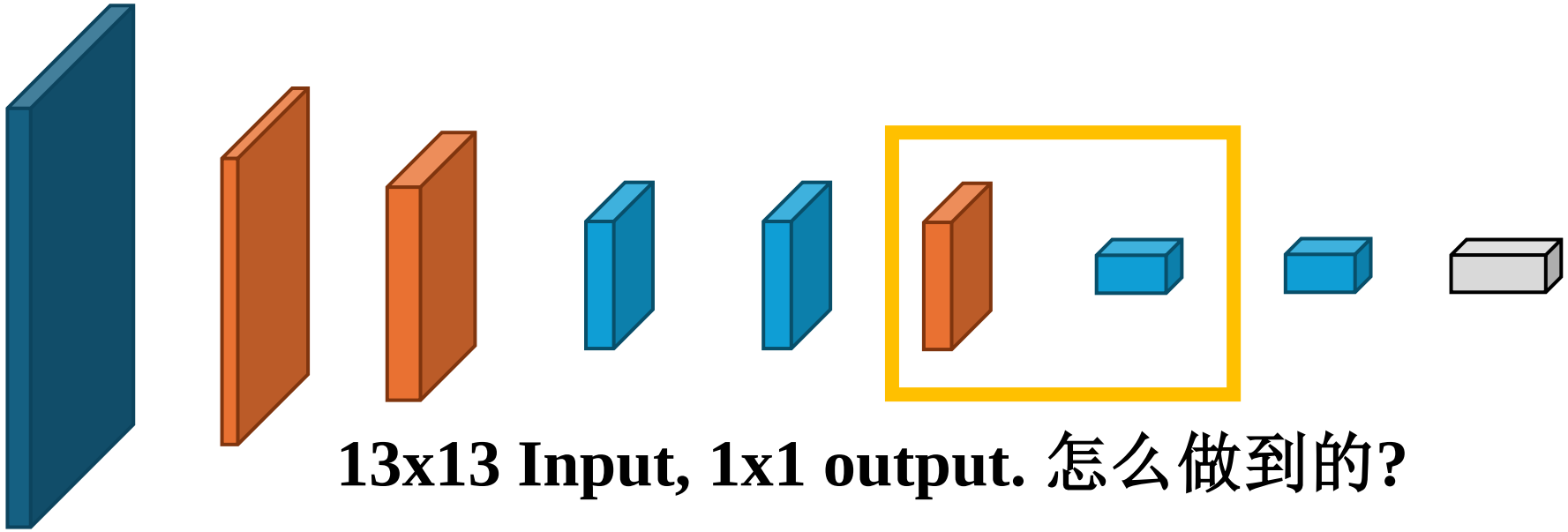
AlexNet – Details

Input	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	FC 6	FC 7	Output
227x227 3	55x55 96	27x27 256	13x13 384	13x13 384	13x13 256	1x1 4096	1x1 4096	1x1 1000



AlexNet

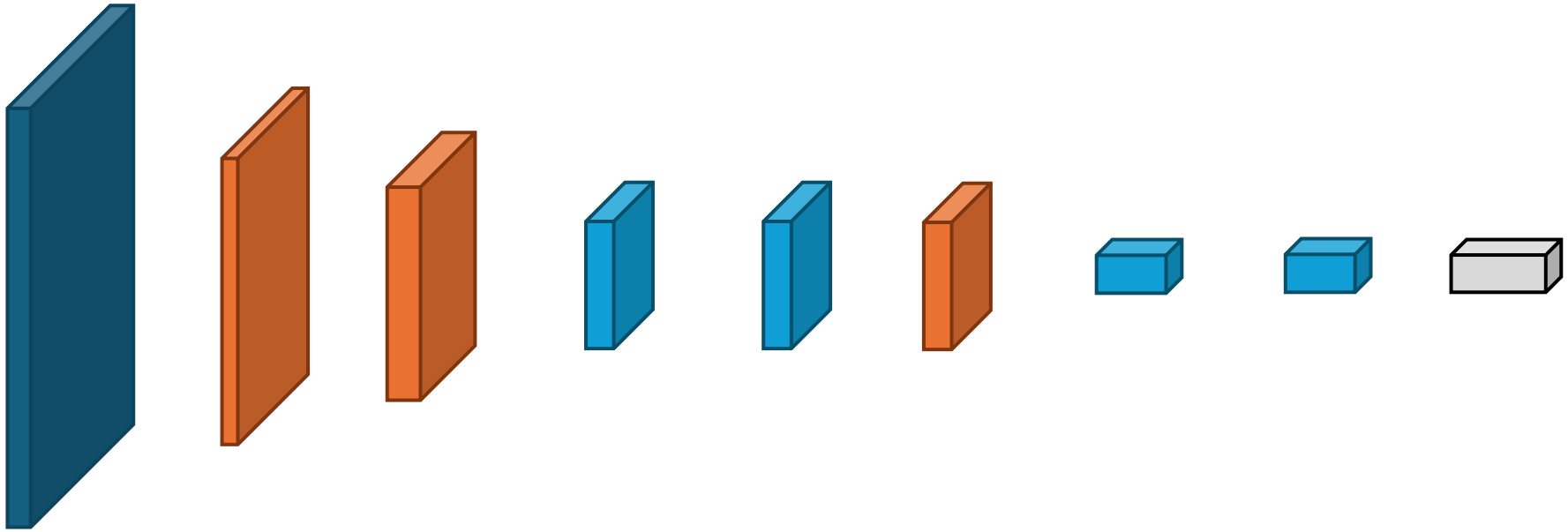
Input	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	FC 6	FC 7	Output
227x227 3	55x55 96	27x27 256	13x13 384	13x13 384	13x13 256	1x1 4096	1x1 4096	1x1 1000



13x13 Input, 1x1 output. 怎么做到的?

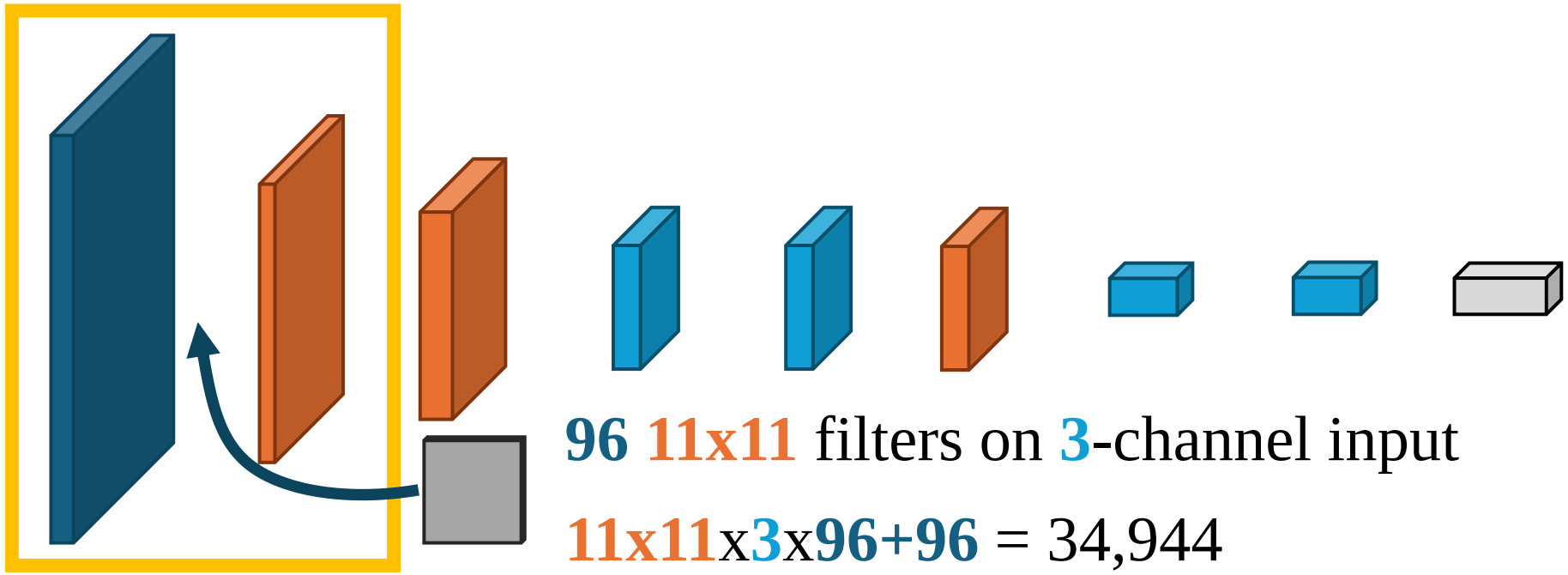
Alexnet – 有多少参数?

Input	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	FC 6	FC 7	Output
227x227 3	55x55 96	27x27 256	13x13 384	13x13 384	13x13 256	1x1 4096	1x1 4096	1x1 1000



Alexnet –有多少参数?

Input	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	FC 6	FC 7	Output
227x227 3	55x55 96	27x27 256	13x13 384	13x13 384	13x13 256	1x1 4096	1x1 4096	1x1 1000

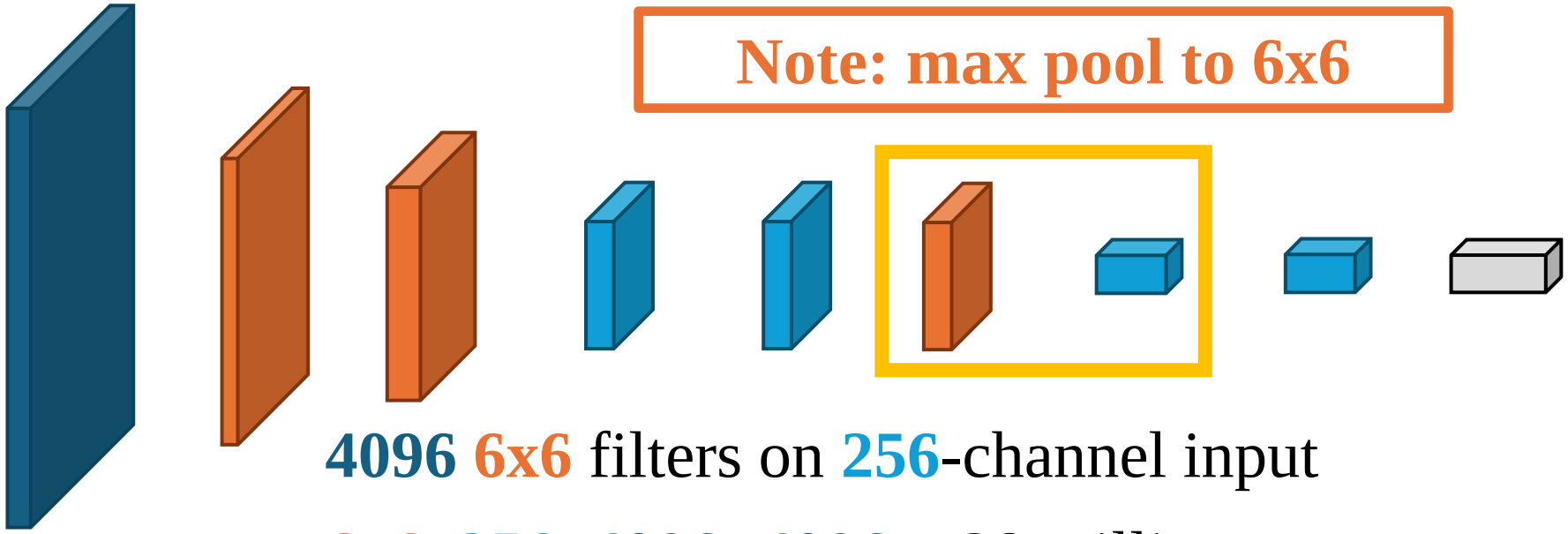


96 11x11 filters on **3**-channel input

11x11x**3**x**96+96** = 34,944

Alexnet –有多少参数?

Input	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	FC 6	FC 7	Output
227x227 3	55x55 96	27x27 256	13x13 384	13x13 384	13x13 256	1x1 4096	1x1 4096	1x1 1000

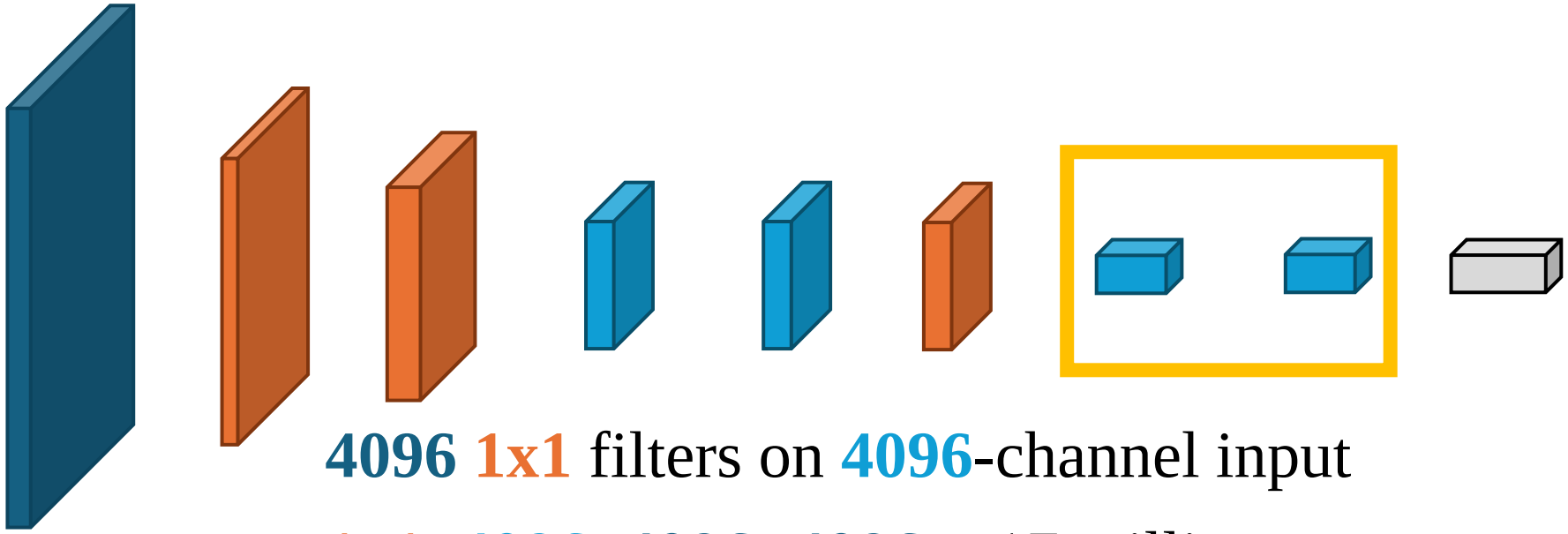


4096 6x6 filters on **256**-channel input

6x6x**256**x**4096+4096** = 38 million

Alexnet –有多少参数?

Input	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	FC 6	FC 7	Output
227x227 3	55x55 96	27x27 256	13x13 384	13x13 384	13x13 256	1x1 4096	1x1 4096	1x1 1000



4096 1x1 filters on **4096**-channel input

1x1x**4096**x**4096+4096** = 17 million

Alexnet –有多少参数

如果以每个参数4秒的速度列出AlexNet所有参数，会需要多长时间?

1年?　　4年?　　8年?　　16年?

- AlexNet有62.4百万个参数
- *绝大多数参数都在全连接层中*
- 论文中指出去掉卷积层会对性能产生灾难性的影响

数据集 – ILSVRC

- Imagenet Largescale Visual Recognition Challenge
- 1.4M张图像
- 1000个类别，类别通常非常精确

数据集 – ILSVRC

Figure Credit: O. Russakovsky

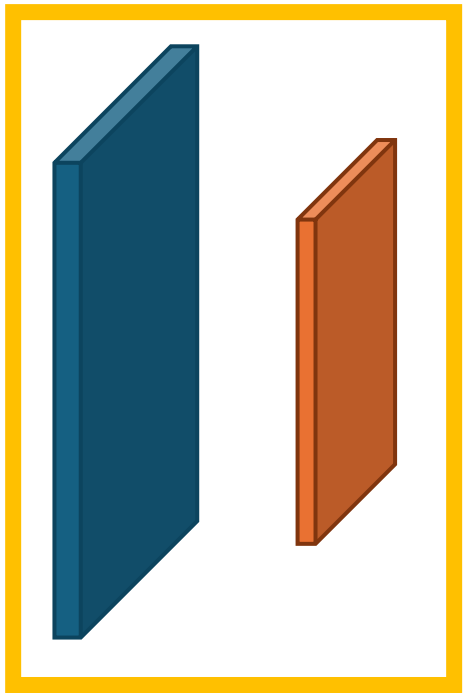
可视化卷积核

Input

227x227
3

Conv
1

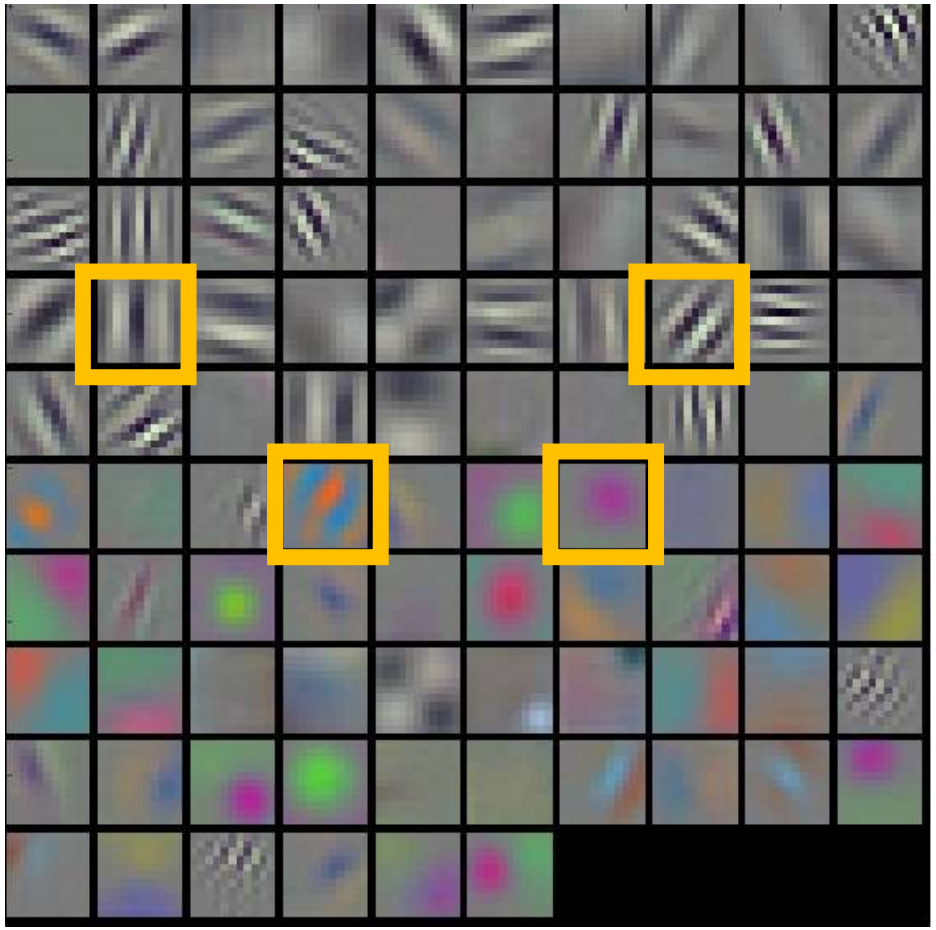
55x559
6



Conv 1 Filters

- **Q. 有多少输入通道?**
  - A: 3
- **每个输入通道代表什么?**
  - R, G, B.

卷积滤波学到了什么？



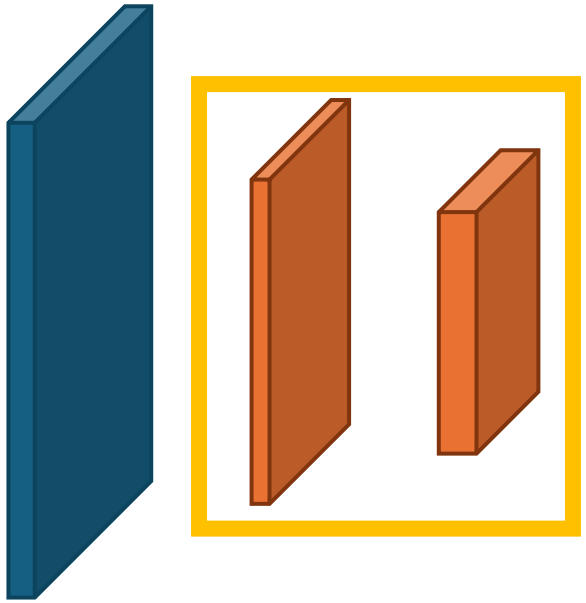
第一层卷积滤波器学到的特征，这些滤波器被训练用于识别1000个不同的对象类别。

这些滤波器是在颜色上应用的，可以捕捉到基于颜色的特征

Figure Credit: Karpathy and Fei-Fei

更深层次的滤波器的可视化

Input	Conv 1	Conv 2
227x227 3	55x559 6	27x272 56



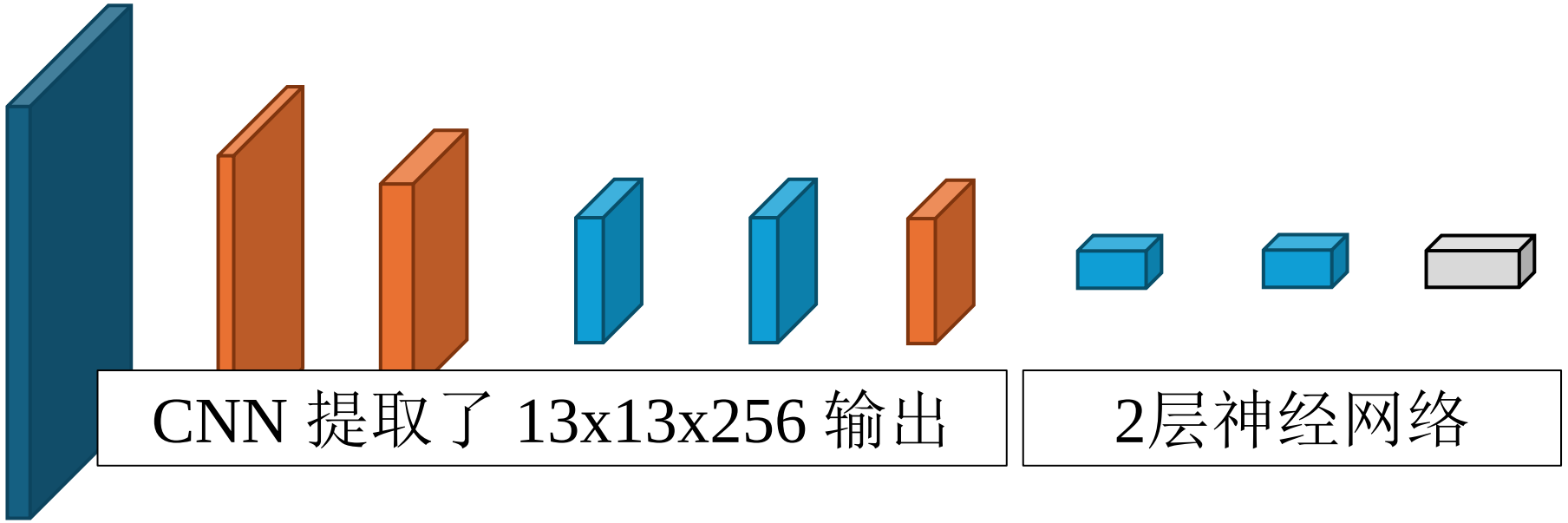
- 理解更深层卷积层的滤波器的具体值是非常困难的，甚至是不可能的，因为输入维度太多，并且输入的具体含义不明确。这是因为随着网络层数的加深，每个滤波器所代表的特征变得更加抽象和复杂，不再像初级滤波器那样容易通过视觉上直接解释。

Conv 2 Filters

- **Q. 有多少通道?**
  - A: 96….
- **每个输入通道代表?**
- 大概是基于颜色的特征。。。

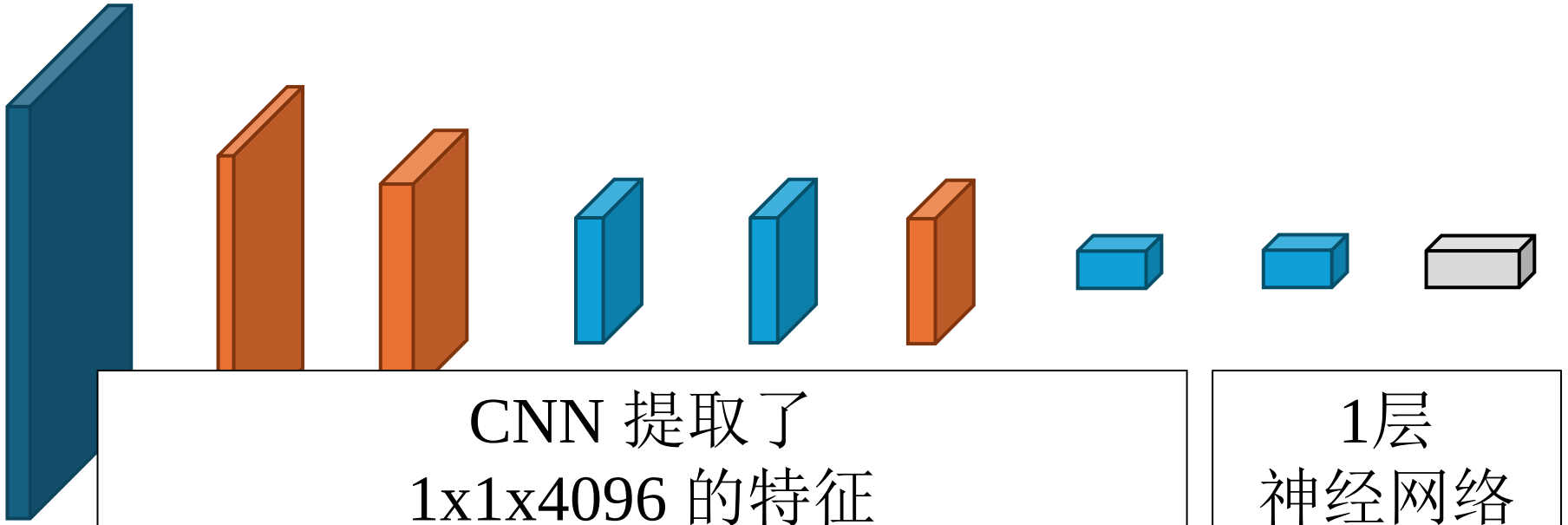
理解深层网络

Input	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	FC 6	FC 7	Output
227x227 3	55x55 96	27x27 256	13x13 384	13x13 384	13x13 256	1x1 4096	1x1 4096	1x1 1000



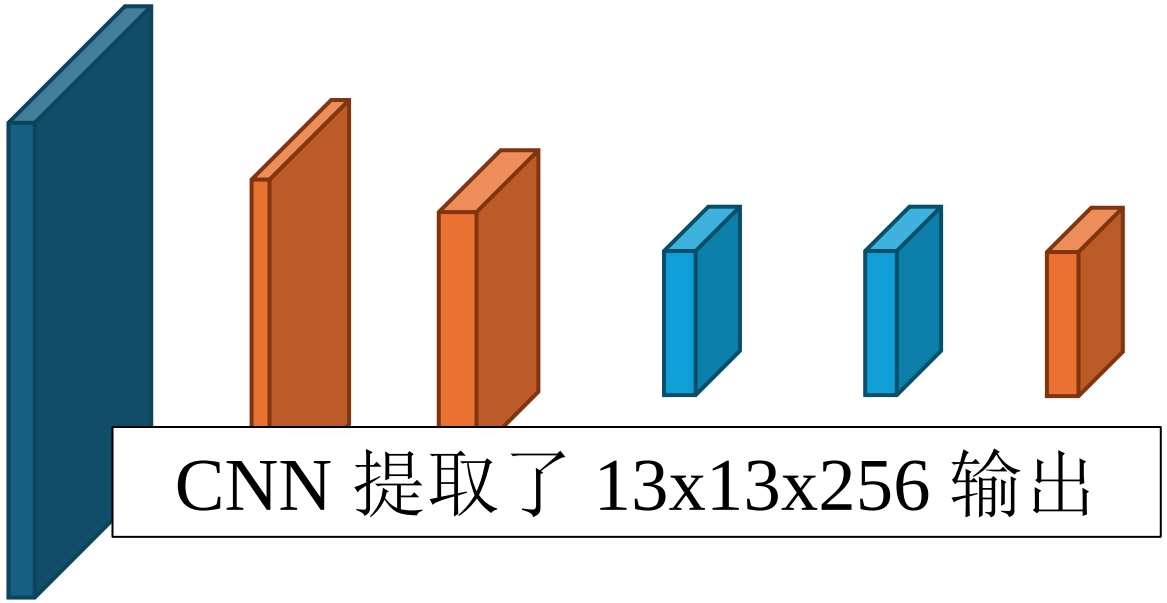
理解深层网络

Input	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	FC 6	FC 7	Output
227x227 3	55x55 96	27x27 256	13x13 384	13x13 384	13x13 256	1x1 4096	1x1 4096	1x1 1000



理解深层网络

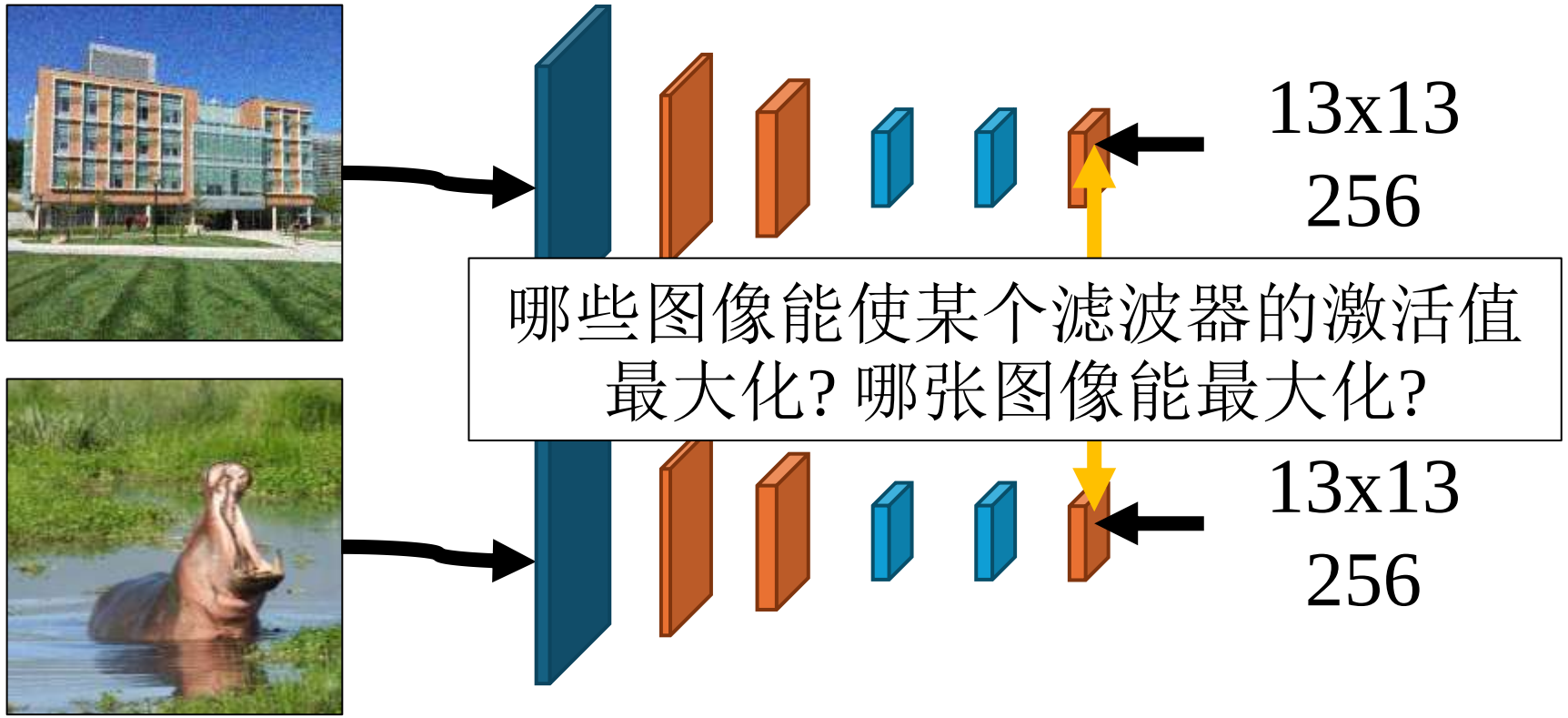
Input	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5
227x227 3	55x55 96	27x27 256	13x13 384	13x13 384	13x13 256



CNN 提取了 13x13x256 输出

理解深层网络

给CNN输入不同的图像，观察不同的滤波器如何对这些图像做出反应。这可以帮助理解特定滤波器的激活模式，即通过看哪些图像能使某个滤波器的激活值最大化，我们可以得到关于该滤波器所捕捉特征的线索。



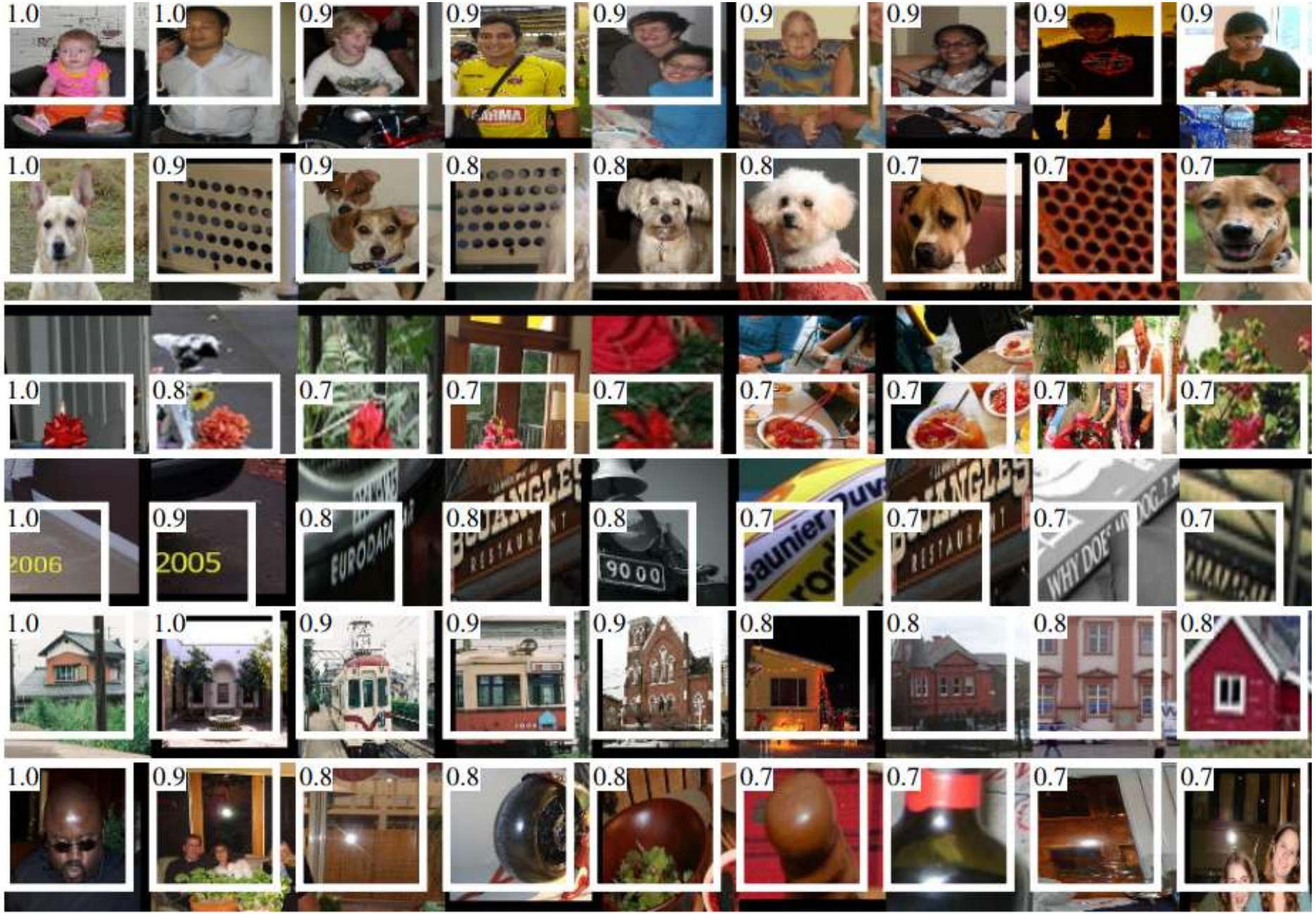
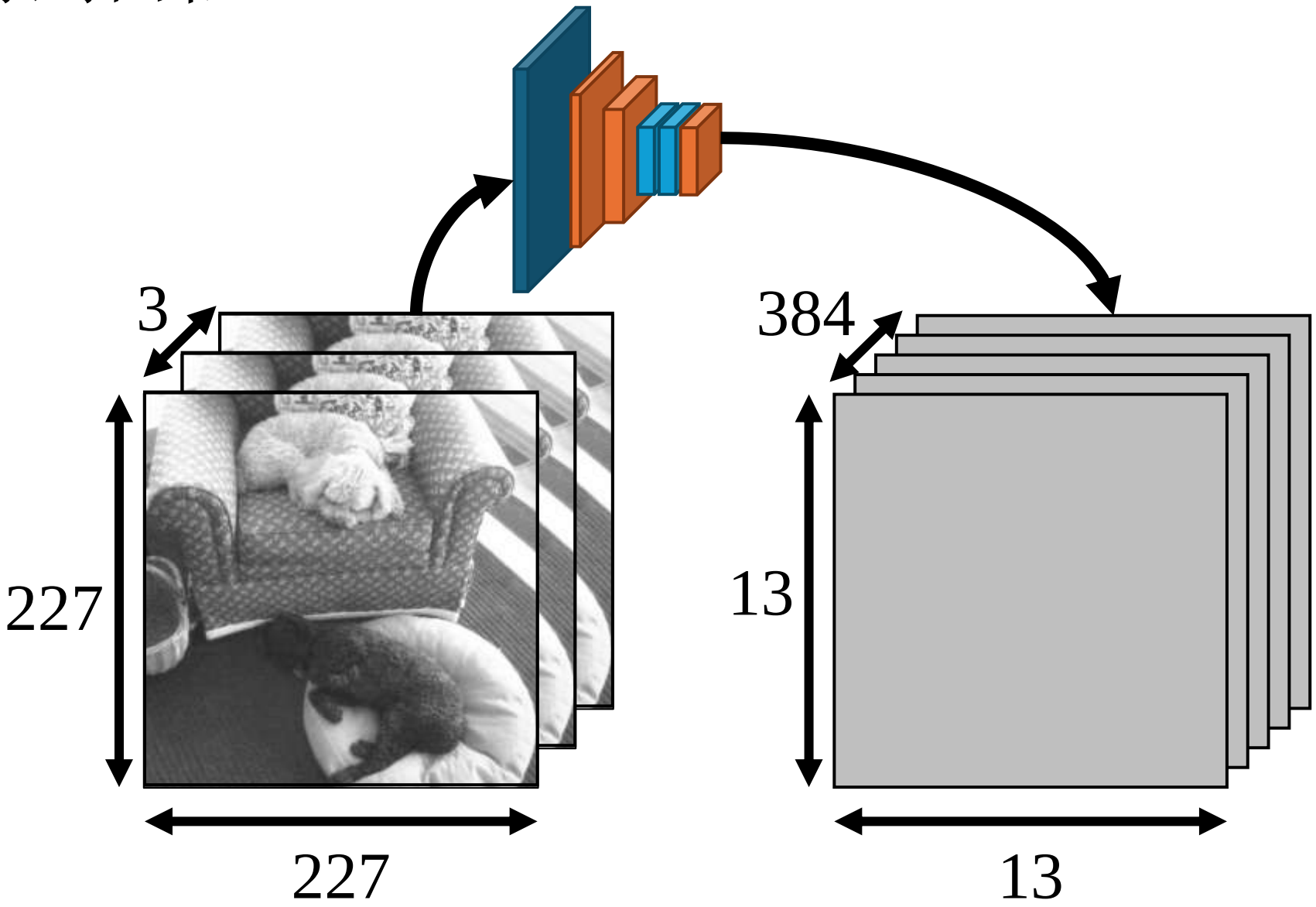
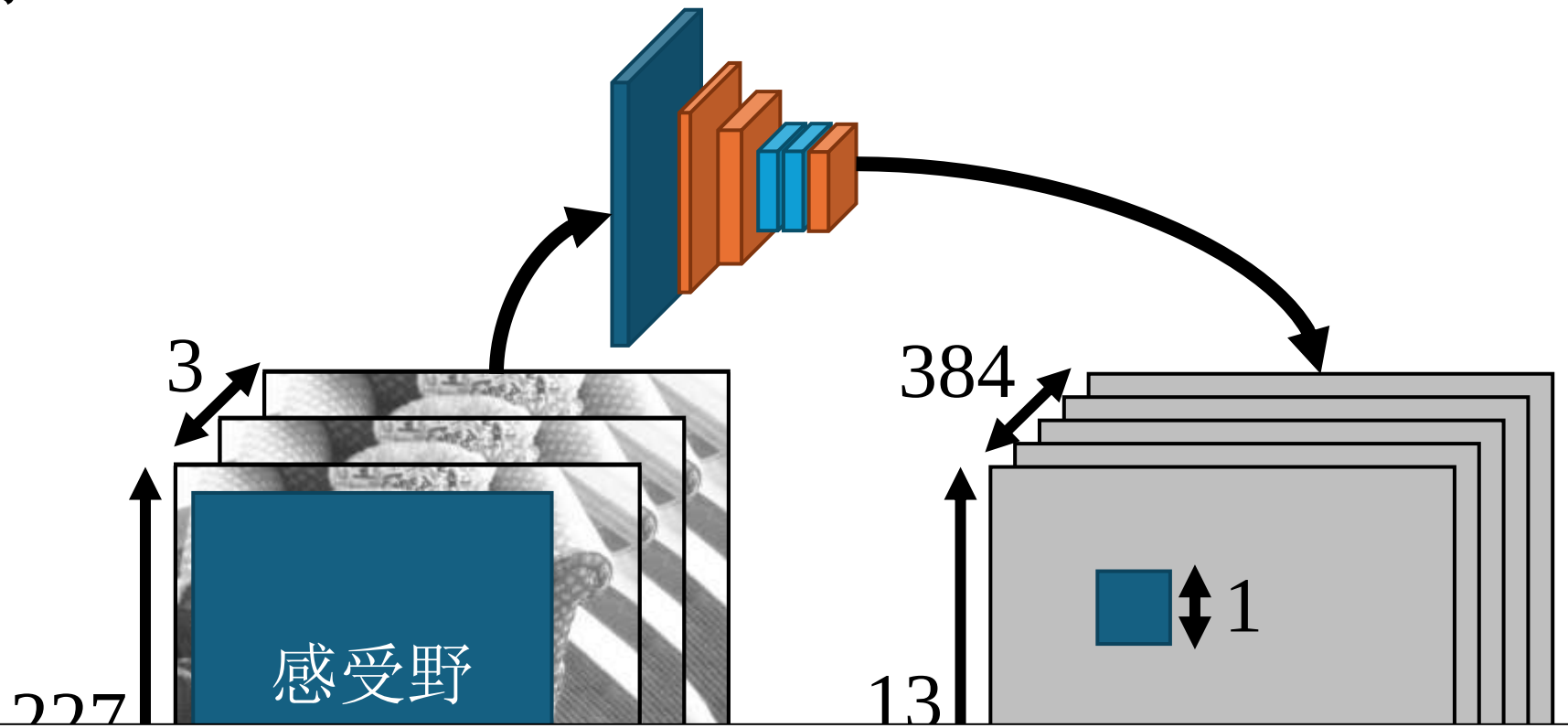


Figure Credit: Girschick et al. CVPR 2014.

怎么得到白框?



感受野



每一个输出特征图的单个像素实际上是由输入图像中的一个小区域转换而来的。卷积层不会一次性看到整个图像，而是通过滤波器窗口看到图像的一小部分，这个窗口逐步移动覆盖整个图像，从而在特征图上生成每一个单独的像素点。

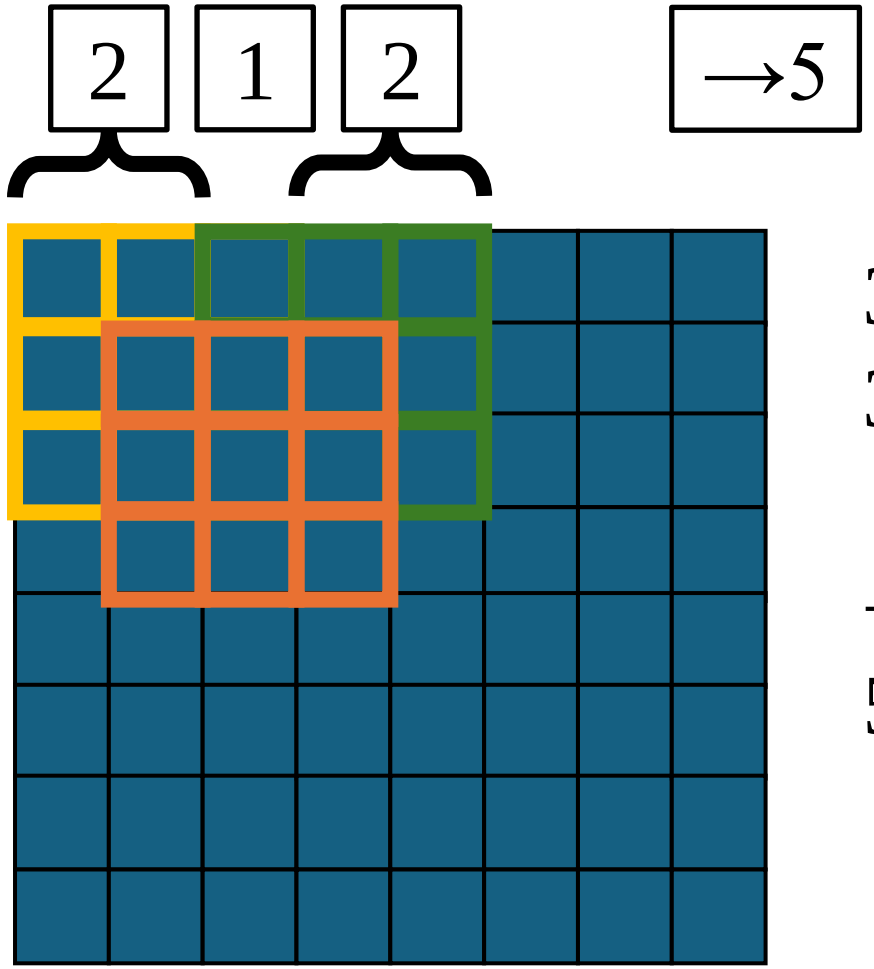
我们可以用感受野来看CNN在做决策时关注哪些区域

B. Zhou et al. Learning Deep Features for Discriminative Localization. CVPR 2016.

3 个训练小技巧

- 3x3 Filters 3x3卷积核
- Batch Normalization 批量归一化
- Residual Learning 残差学习

3x3 滤波

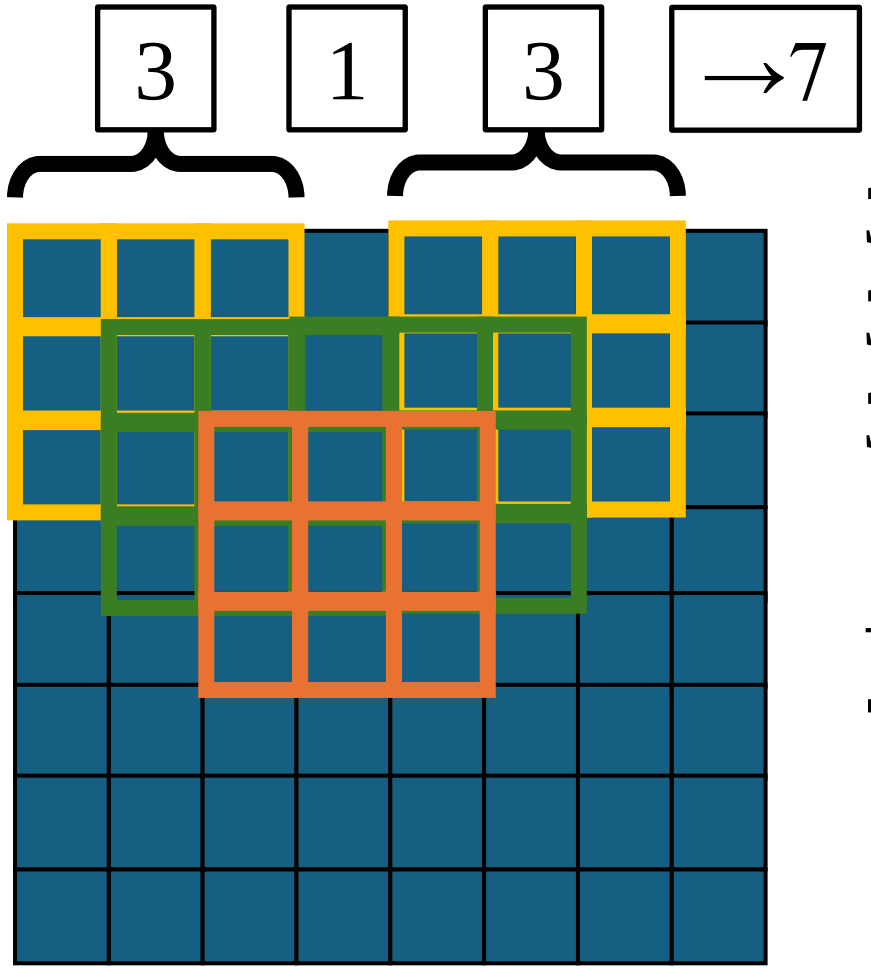


3x3 滤波 接一个
3x3 滤波

→

5x5 感受野的滤波

3x3 滤波

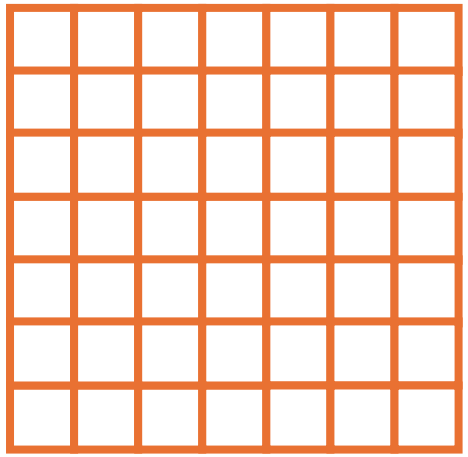


3x3 滤波 接一个
3x3 滤波 接一个
3x3 滤波

→

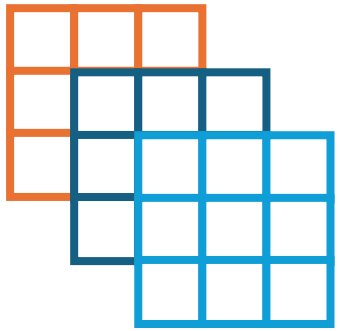
7x7 感受野的滤波

思考——与7x7卷积的区别？

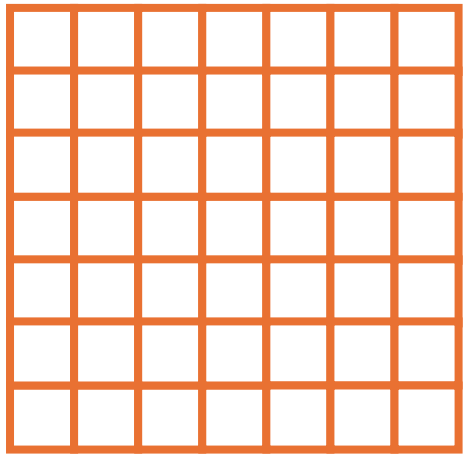


重复三次3x3 filters 一般会比直接做 a 7x7 filter 好

为什么？



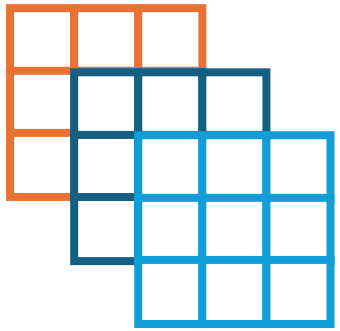
3x3 滤波



感受野: 7x7 pixels

参数量: 49

ReLUs: 1



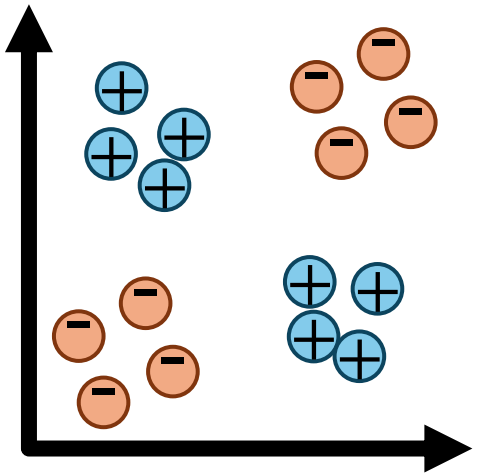
感受野: 7x7 pixels

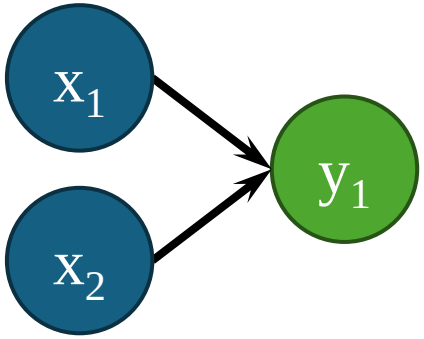
参数量: 3x3x3=**27**

ReLUs: **3**

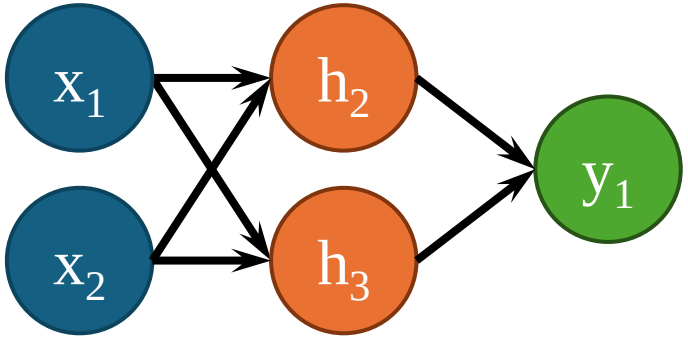
我们需要做更多的非线性层!

一个7x7可以做XOR吗?





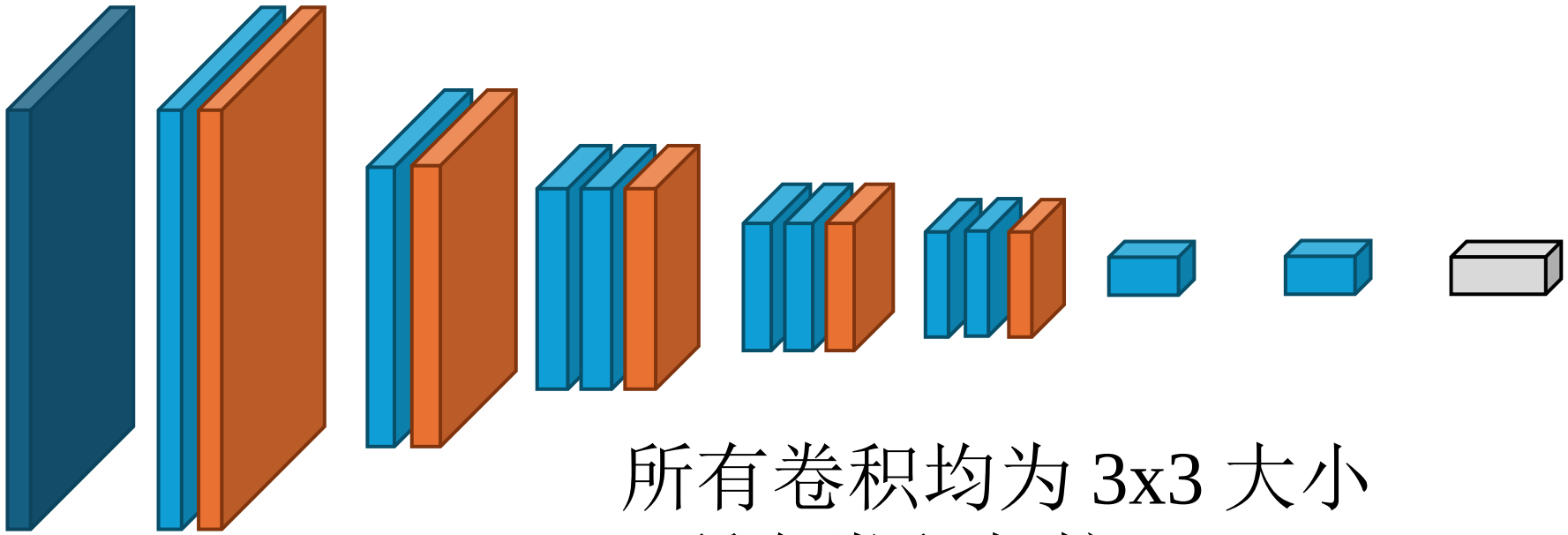
No



Yes

VGG16

Input	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	FC 6	FC 7	Output
224x224 3	224x224 64	112x112 128	56x56 256	28x28 512	14x14 512	1x1 4096	1x1 4096	1x1 1000

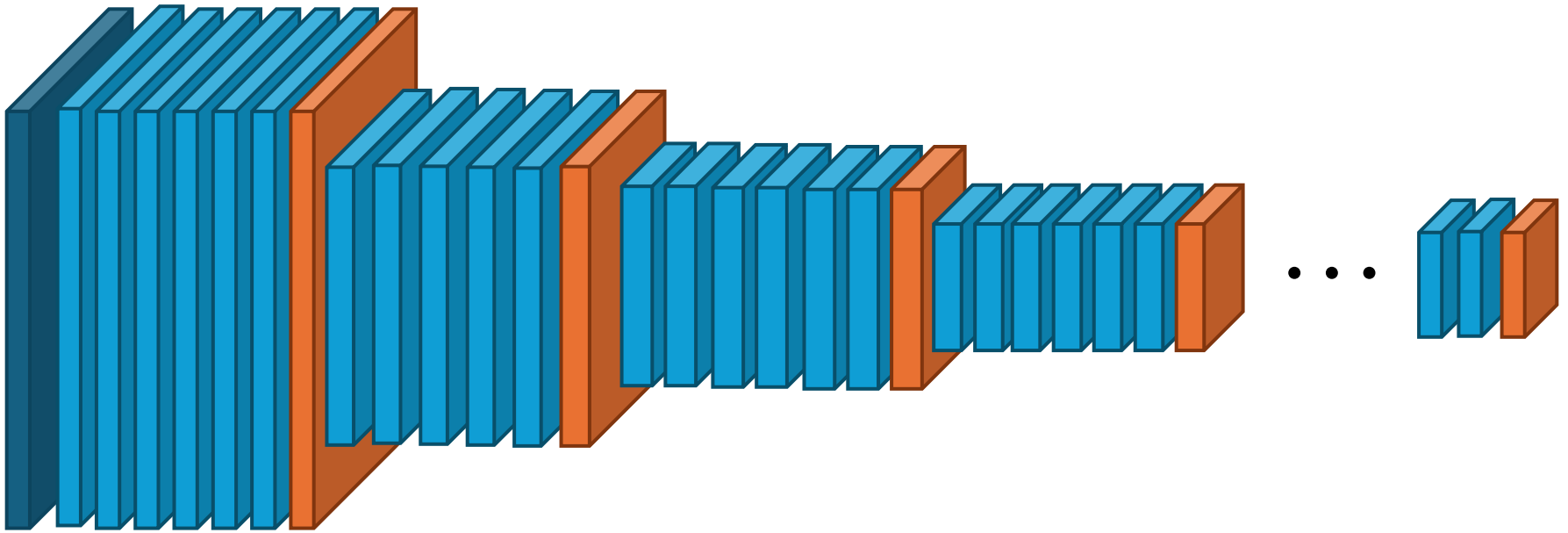


所有卷积均为 3x3 大小

所有卷积都接 ReLU

训练更深的网络

梯度反传在深层网络会出现什么问题?



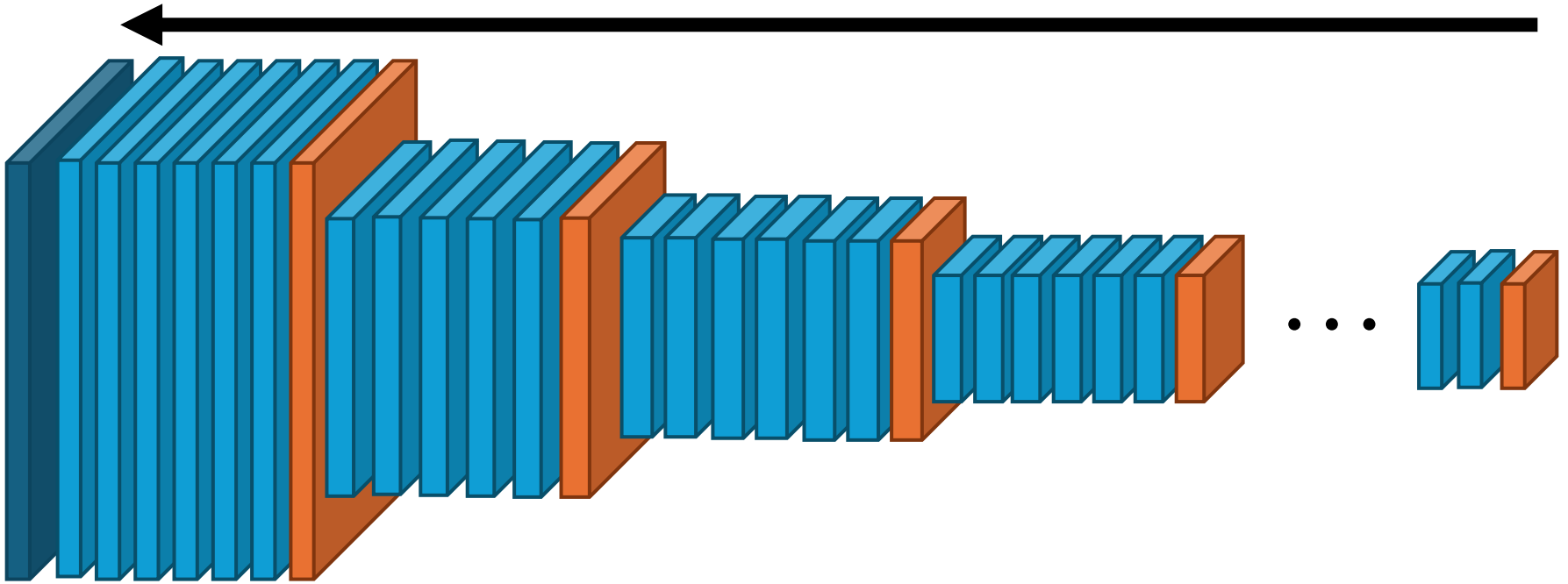
反向传播

每个反向传播操作会把局部的梯度乘上
回传的梯度

$1 * d * d * d \ldots * d = d^{n-1}$

如果 d << 1, n 很大?

Vanishing Gradients 梯度消失



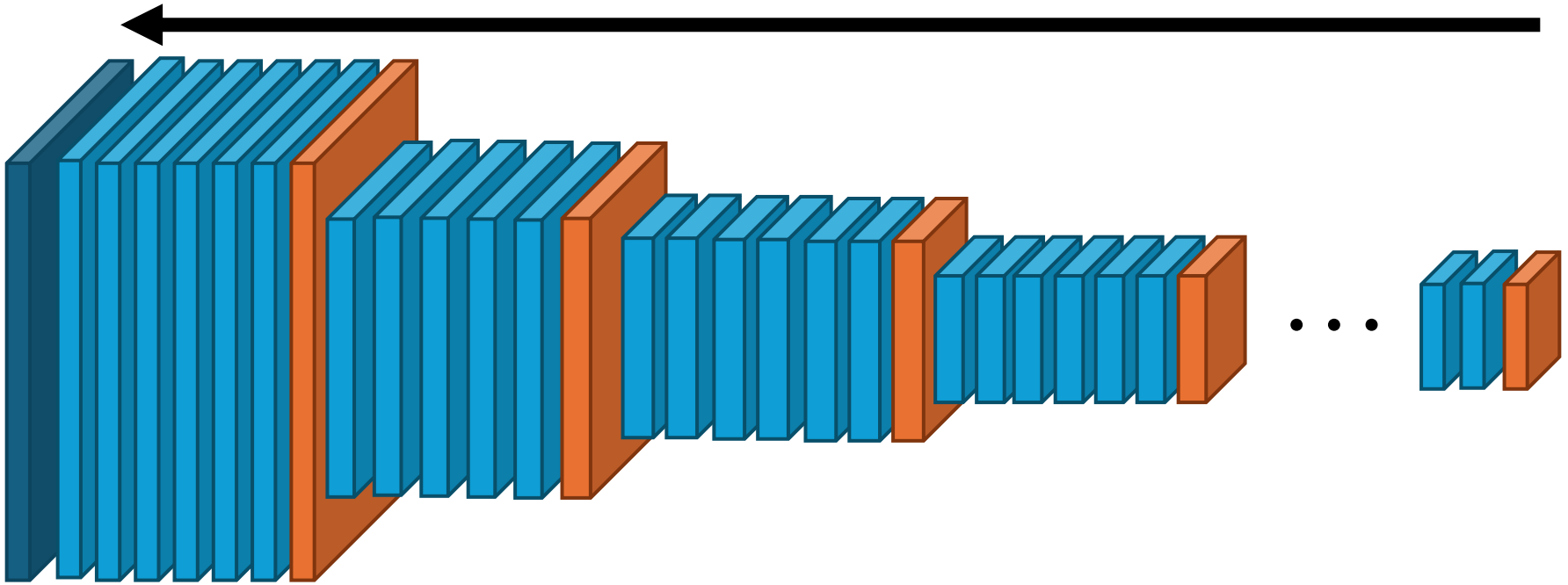
反向传播

每个反向传播操作会把局部的梯度乘上
回传的梯度

$1 * d * d * d \ldots * d = d^{n-1}$

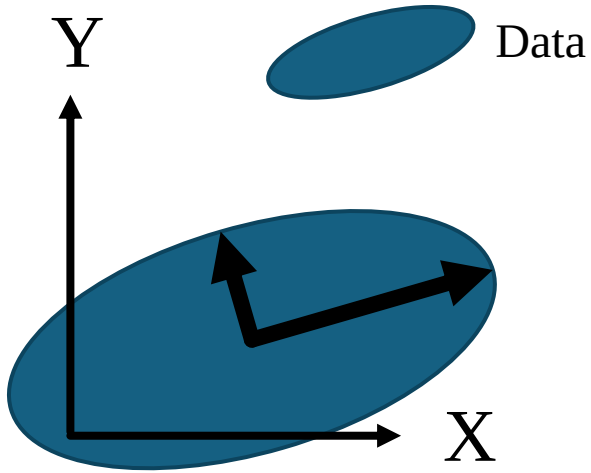
如果 d >> 1, n 很大?

Exploding Gradients 梯度爆炸

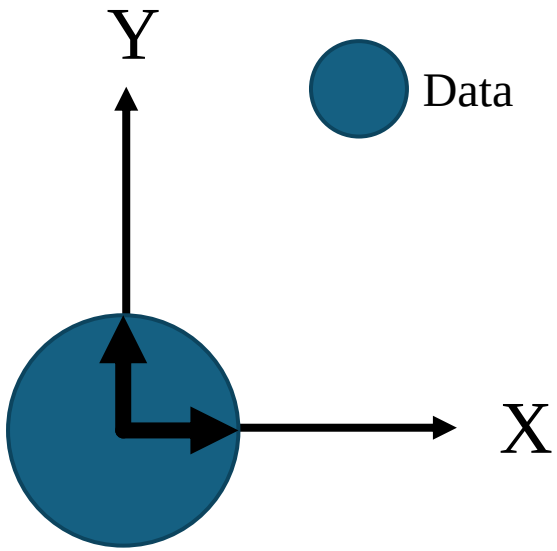


Batch Normalization

右边的情况一般会比左边的情况容易学习

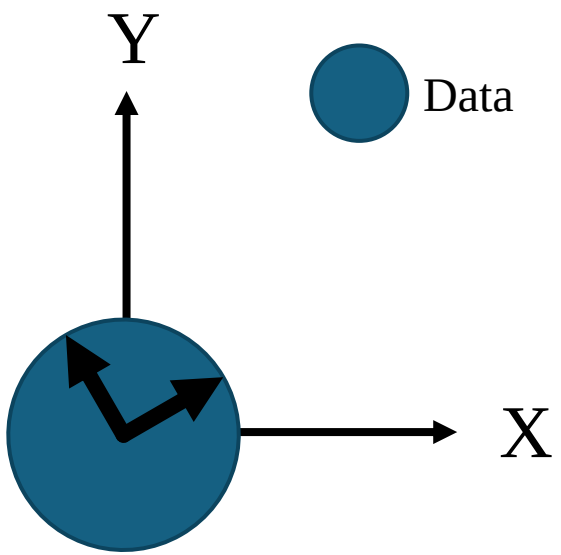


Mean(x) != Mean(Y) != 0
Var(x) != Var(y) != 0
Cov(x,y) != 0



Mean(x) = Mean(Y) = 0
Var(x) = Var(y) = 1
Cov(x,y) = 0

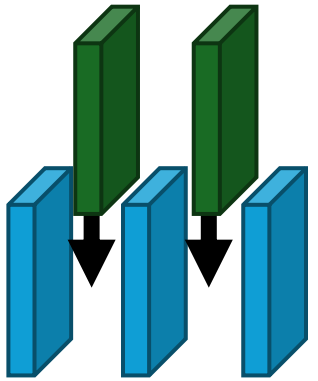
Batch Normalization



Mean(x) = Mean(Y) = 0

Var(x) = Var(y) = 1

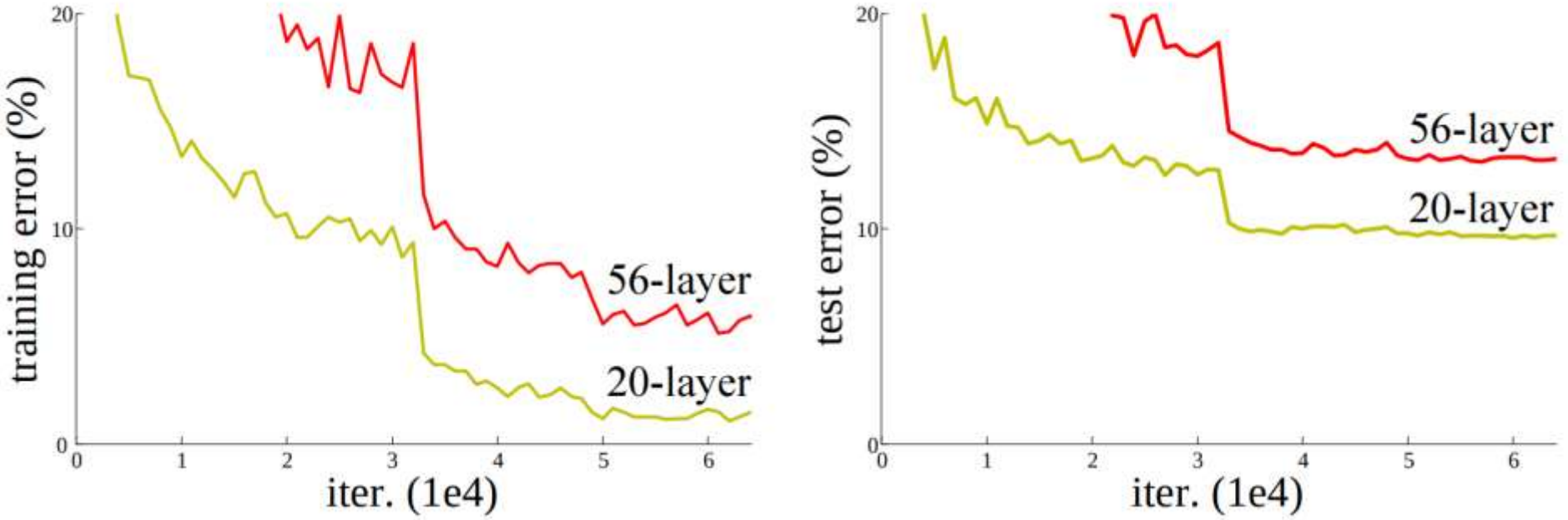
Idea: (**Batch Norm**)将批量归一化作为单独层插入到网络中的想法，这个层会对经过它的数据进行归一化，基于每个批次中数据的方差估计值.



S. Ioffe and C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.

现象

- 更深的模型 在训练集上的拟合能力比 更浅的模型 更差.
- **深度网络可能更难训练。虽然存在更深的模型理论上能够拟合数据，但实际上我们可能找不到合适的参数来训练它。**

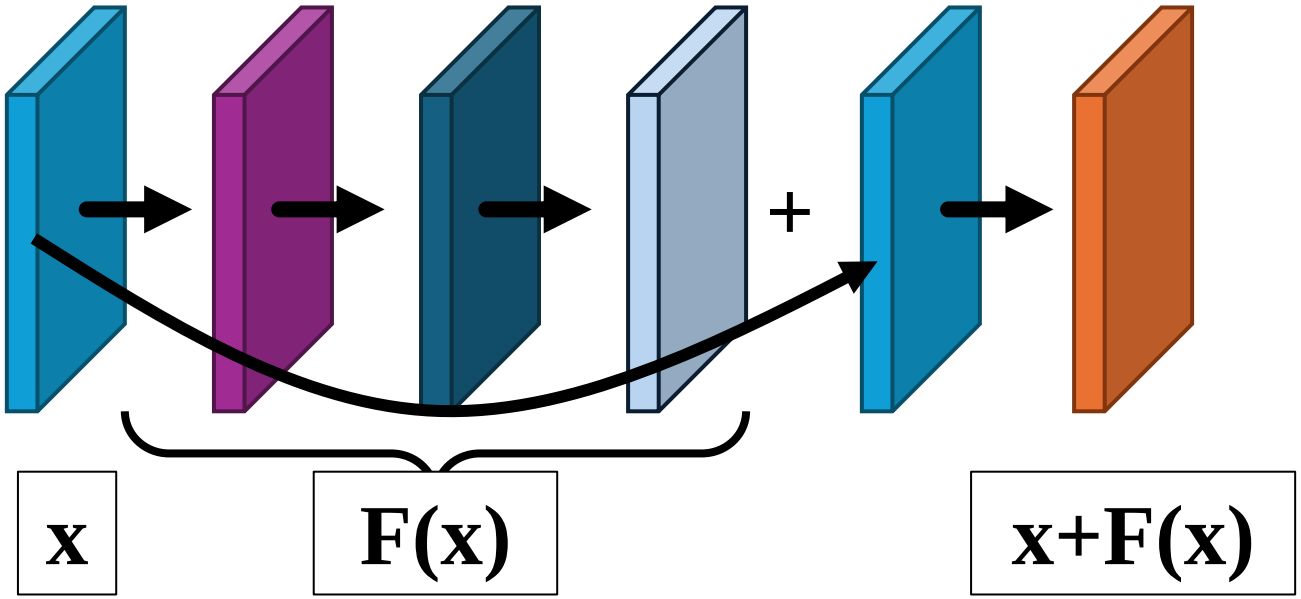


K. He et al. *Deep Residual Learning for Image Recognition.* CVPR 2016

Residual Learning 残差学习

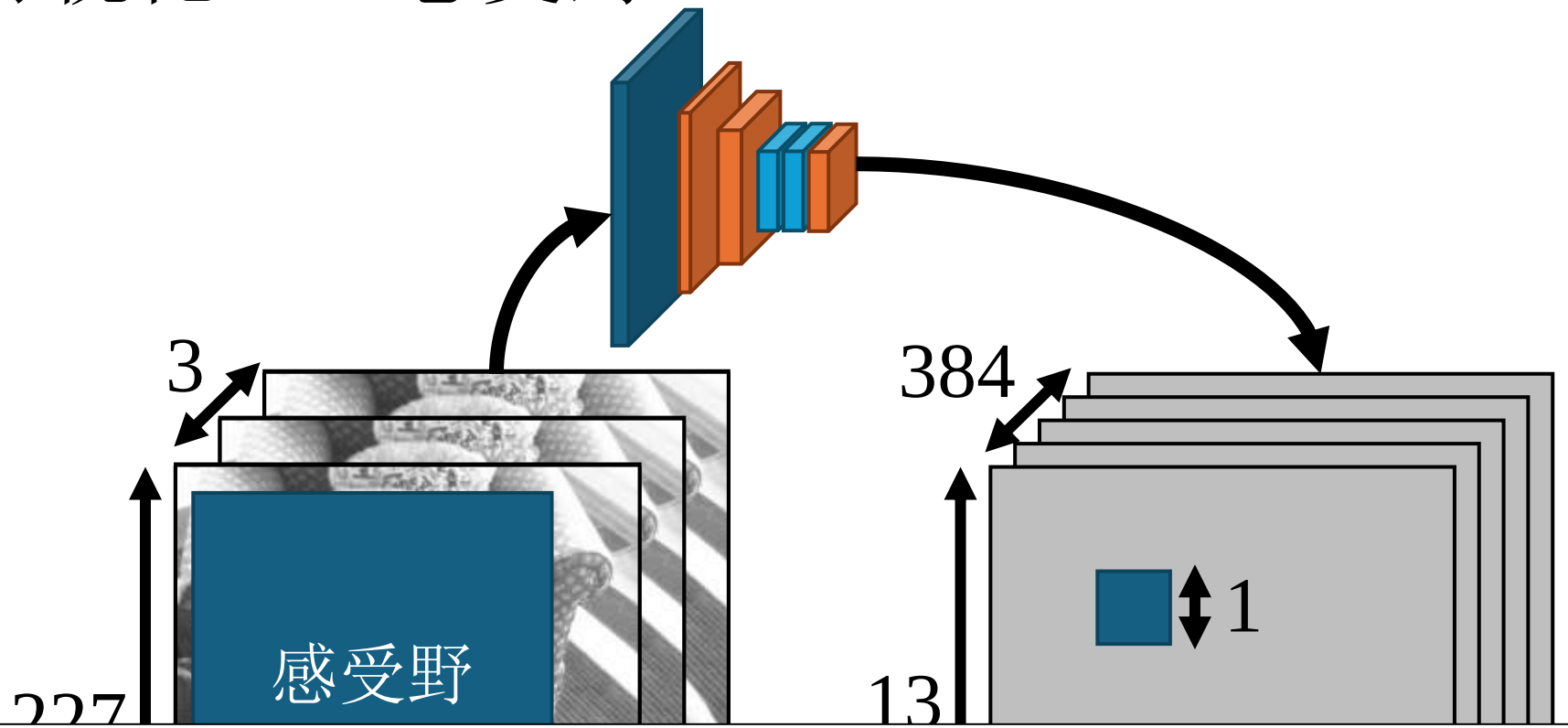
新的模块: $\boldsymbol{x} + F(\boldsymbol{x})$

让你轻松训练100层的网络.



回顾——卷积神经网络

CNN可视化——感受野

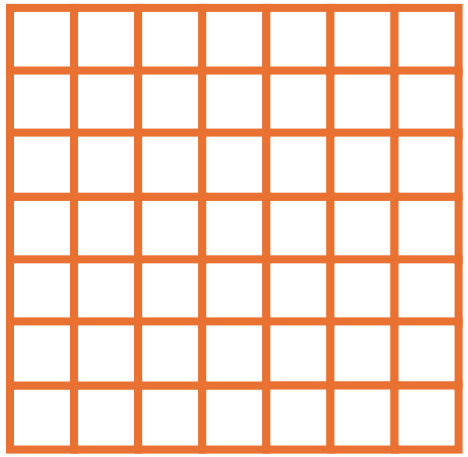


每一个输出特征图的单个像素实际上是由输入图像中的一个小区域转换而来的。卷积层不会一次性看到整个图像，而是通过滤波器窗口看到图像的一小部分，这个窗口逐步移动覆盖整个图像，从而在特征图上生成每一个单独的像素点。

我们可以用感受野来看CNN在做决策时关注哪些区域

B. Zhou et al. Learning Deep Features for Discriminative Localization. CVPR 2016.

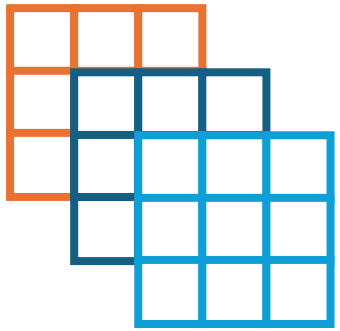
CNN设计——3x3 滤波



感受野: 7x7 pixels

参数量: 49

ReLUs: 1



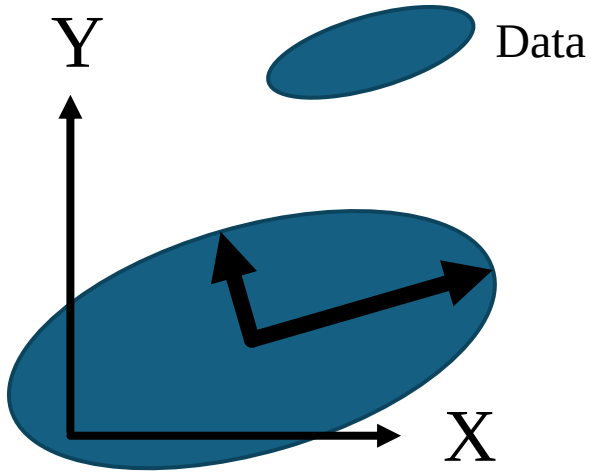
感受野: 7x7 pixels

参数量: 3x3x3=**27**

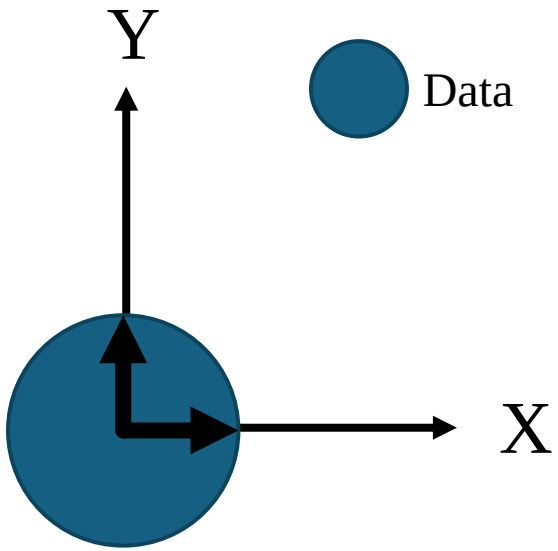
ReLUs: **3**

CNN设计—— Batch Normalization

右边的情况一般会比左边的情况容易学习



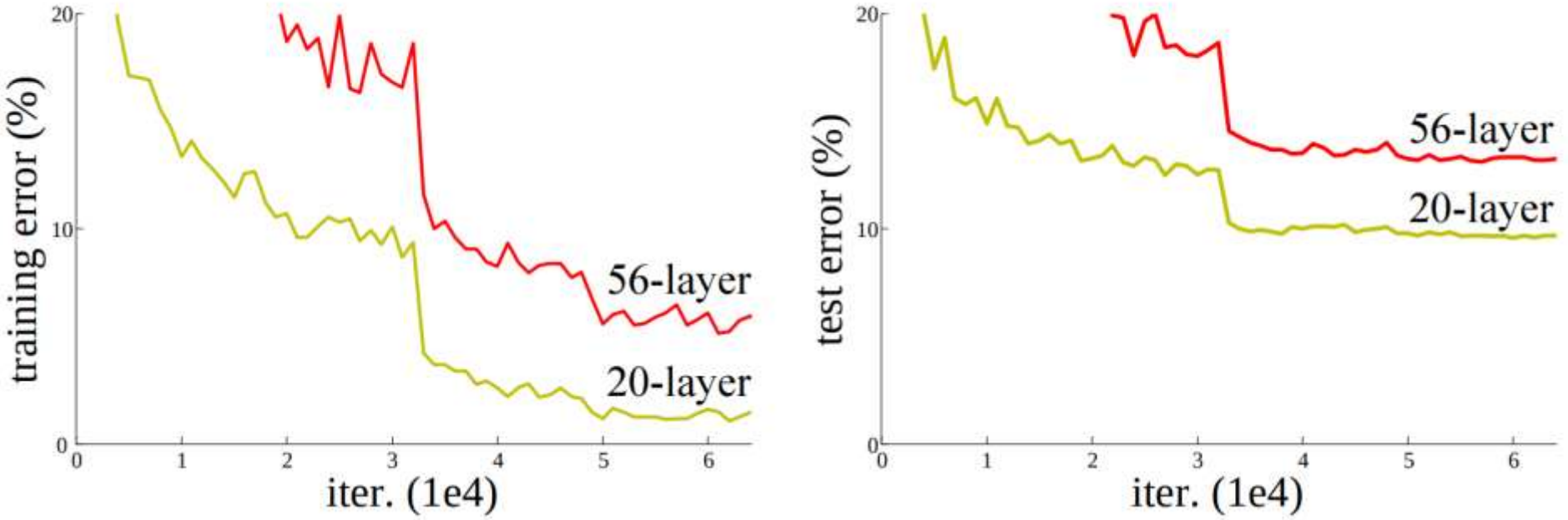
Mean(x) != Mean(Y) != 0
Var(x) != Var(y) != 0
Cov(x,y) != 0



Mean(x) = Mean(Y) = 0
Var(x) = Var(y) = 1
Cov(x,y) = 0

CNN设计——残差

- 更深的模型 在训练集上的拟合能力比 更浅的模型 更差.
- **深度网络可能更难训练。虽然存在更深的模型理论上能够拟合数据，但实际上我们可能找不到合适的参数来训练它。**

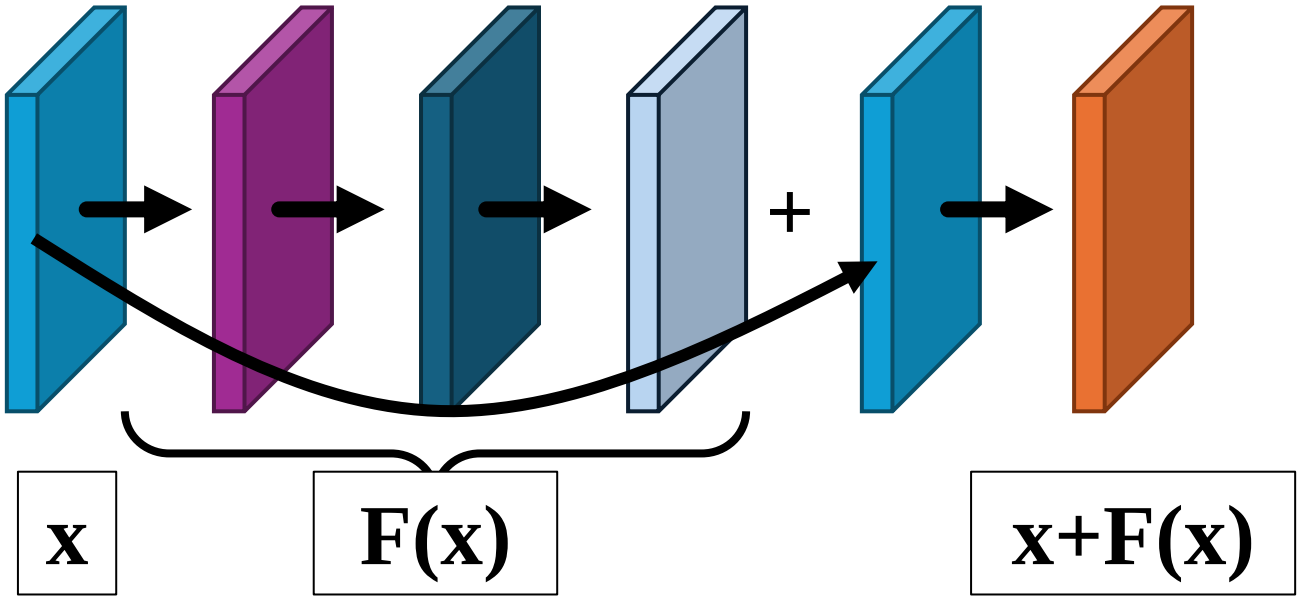


K. He et al. *Deep Residual Learning for Image Recognition.* CVPR 2016

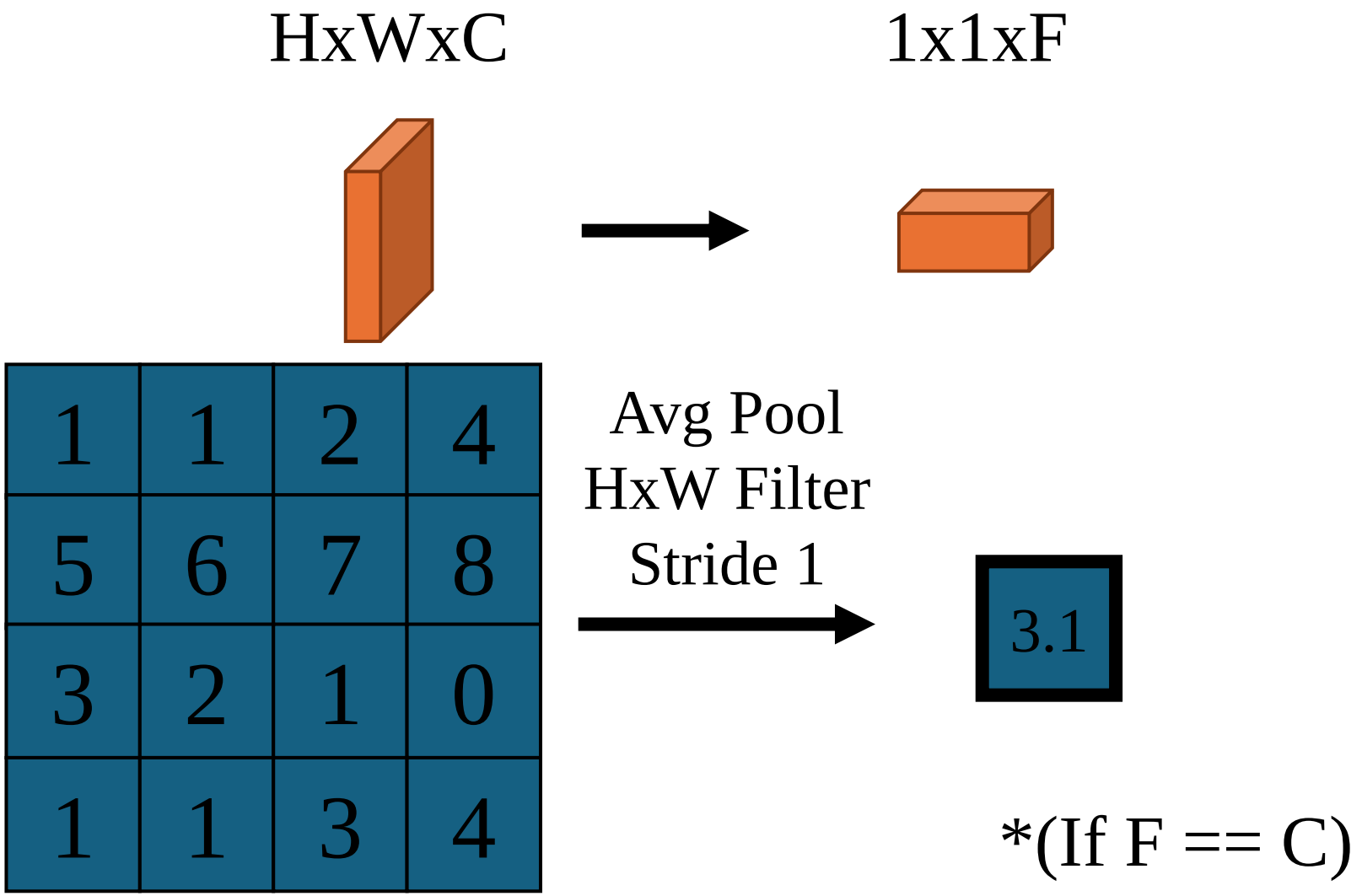
CNN设计——残差学习

新的模块: $\boldsymbol{x} + F(\boldsymbol{x})$

让你轻松训练100层的网络.



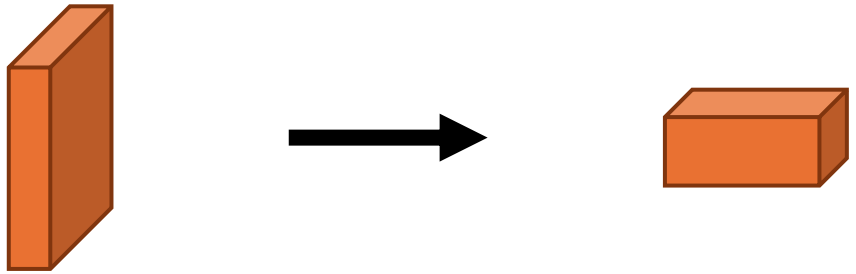
特征转化为向量 —— Pooling



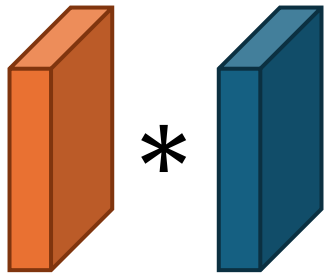
*(If F == C)

特征转化为向量—— 卷积

HxWxC　　1x1xF



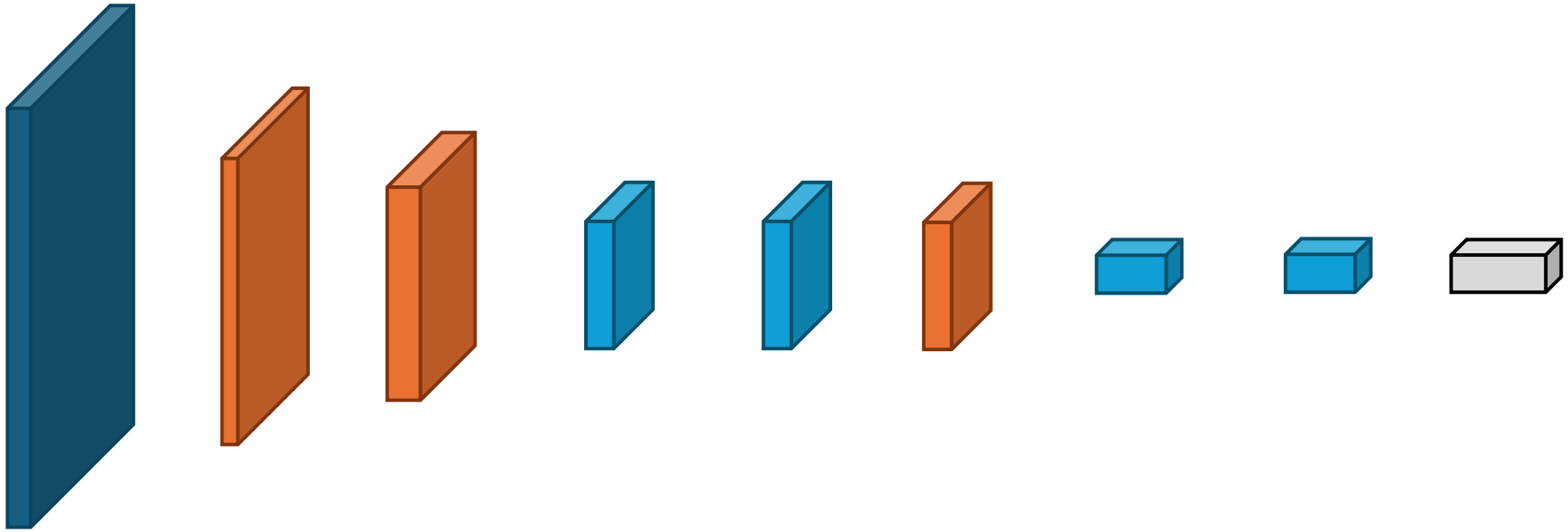
HxW F 个滤波的卷积



每个滤波只
输出一个值

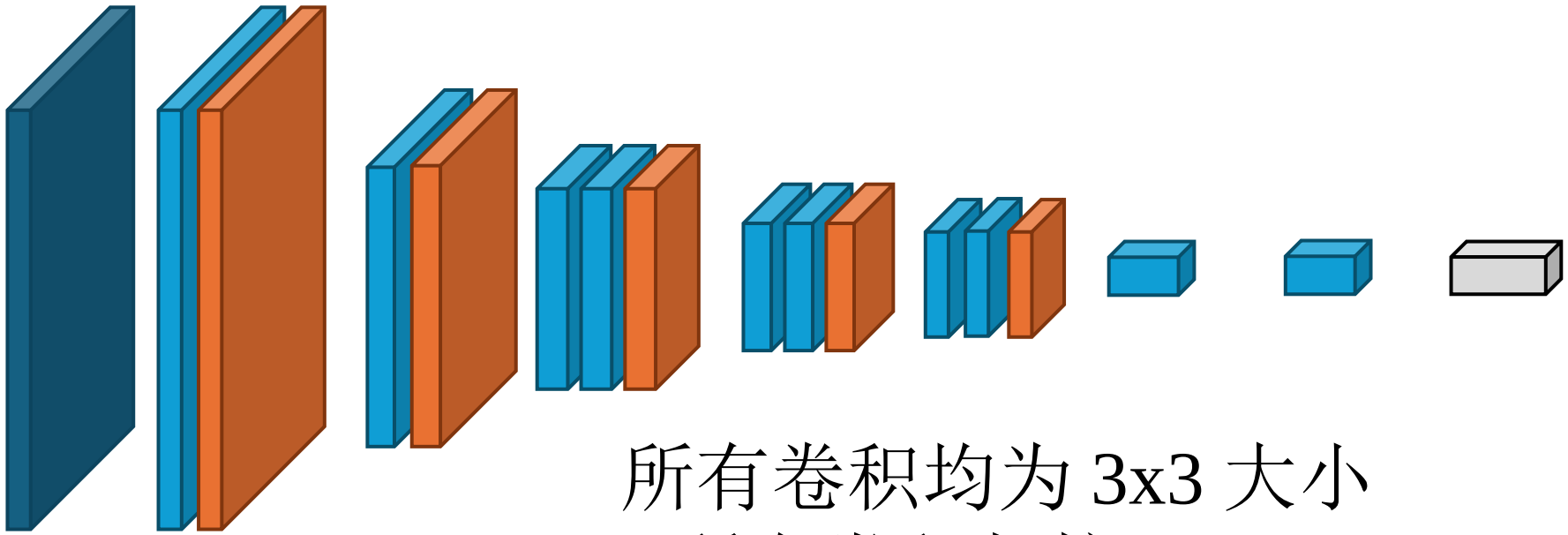
AlexNet

Input	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	FC 6	FC 7	Output
227x227 3	55x55 96	27x27 256	13x13 384	13x13 384	13x13 256	1x1 4096	1x1 4096	1x1 1000



VGG16

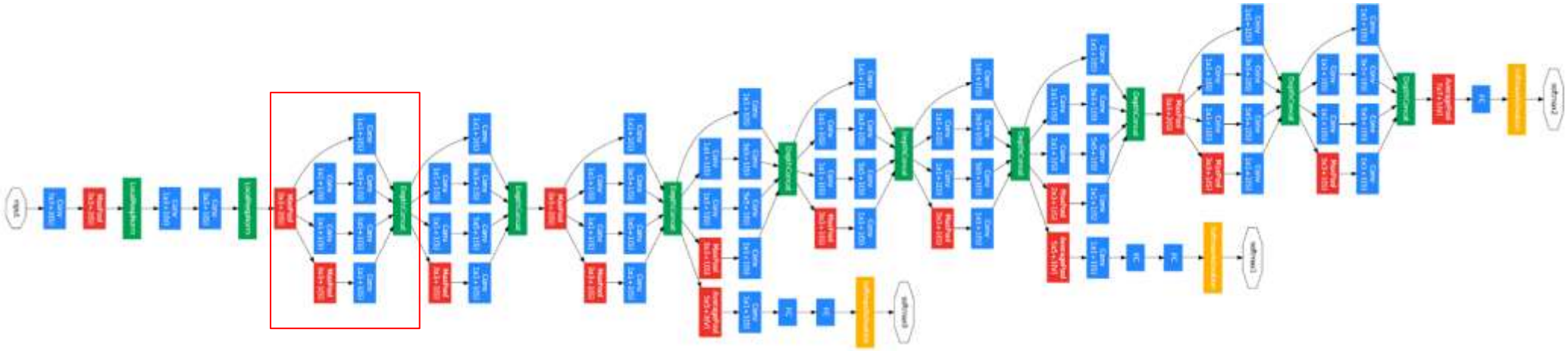
Input	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	FC 6	FC 7	Output
224x224 3	224x224 64	112x112 128	56x56 256	28x28 512	14x14 512	1x1 4096	1x1 4096	1x1 1000

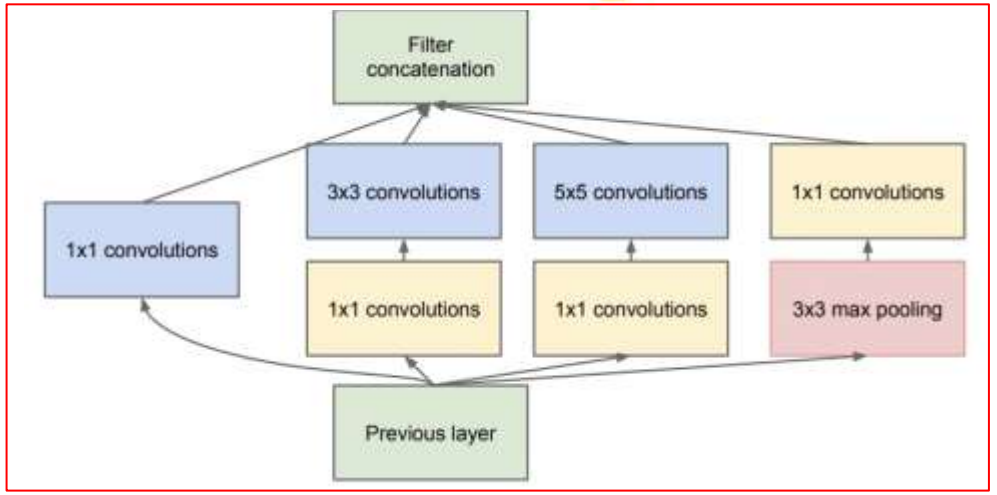


所有卷积均为 3x3 大小

所有卷积都接 ReLU

Inception Nets (2014+)

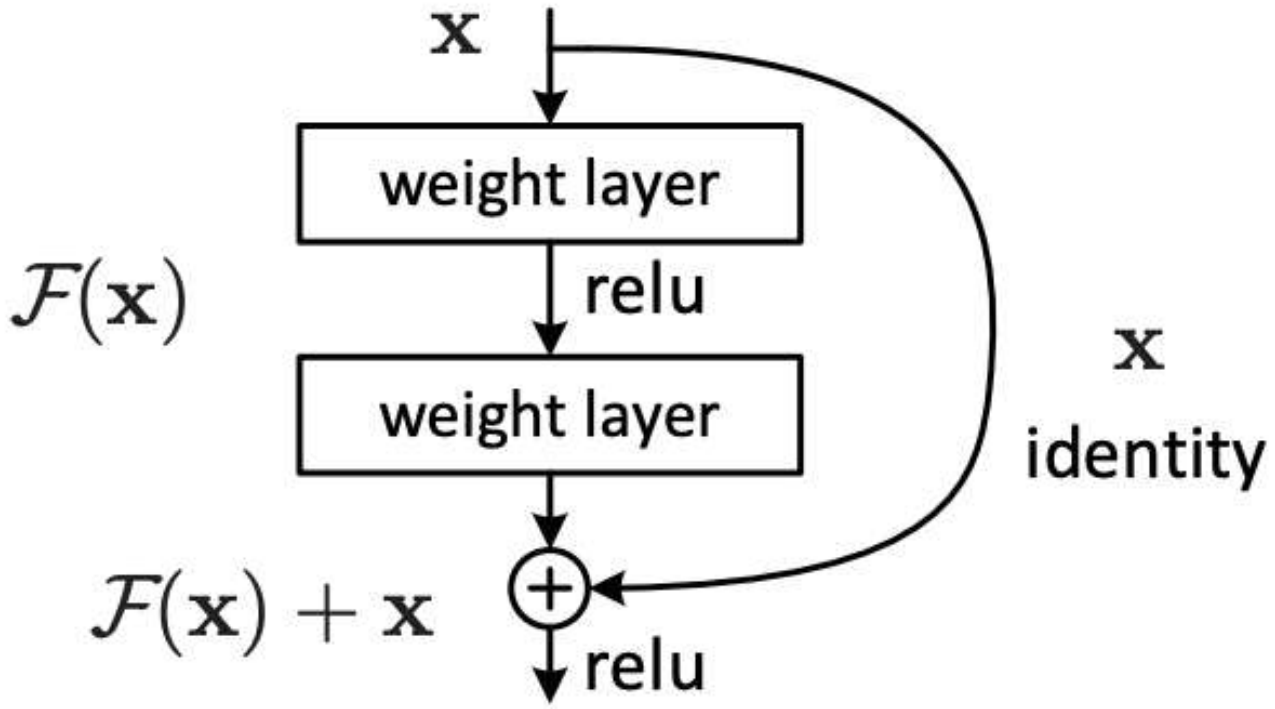




各种卷积组合

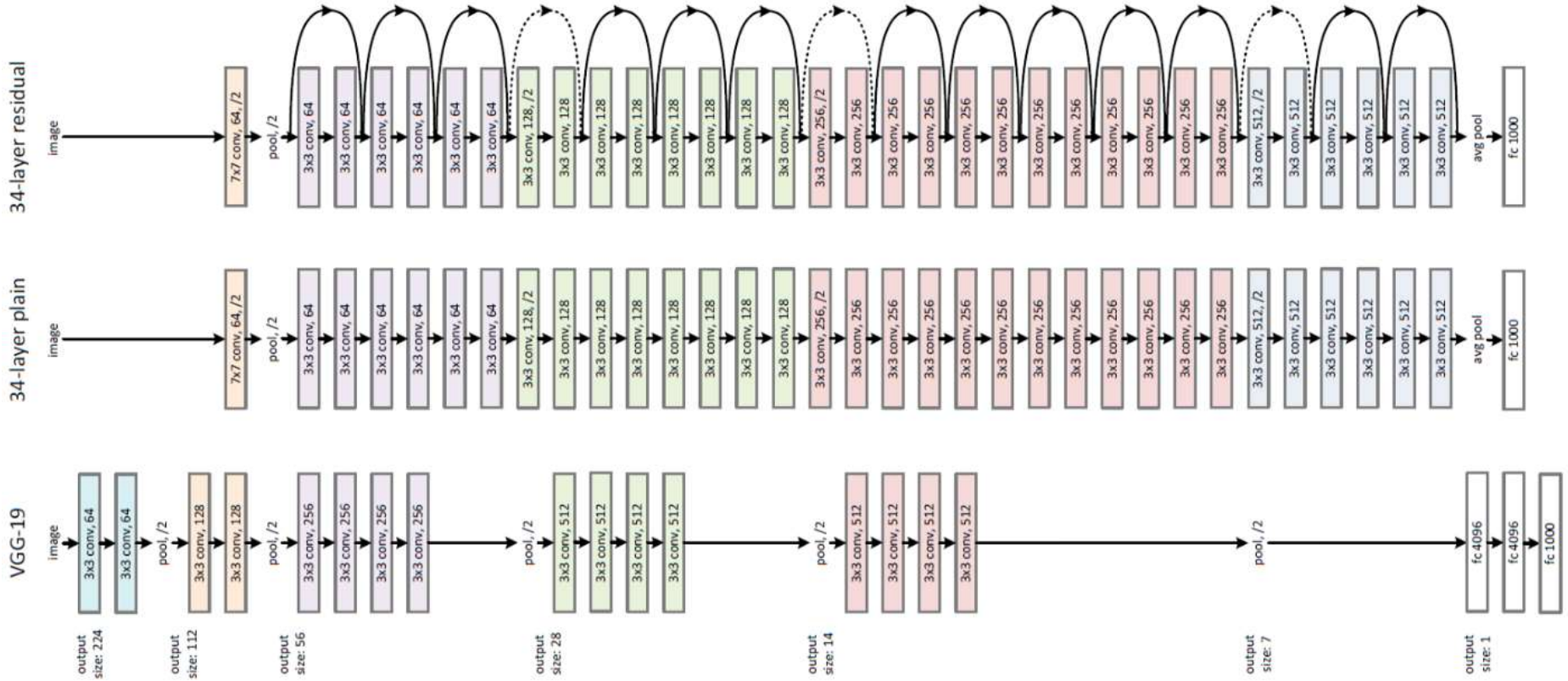
https://arxiv.org/pdf/1409.4842.pdf

ResNet (2015)



https://arxiv.org/pdf/1512.03385.pdf

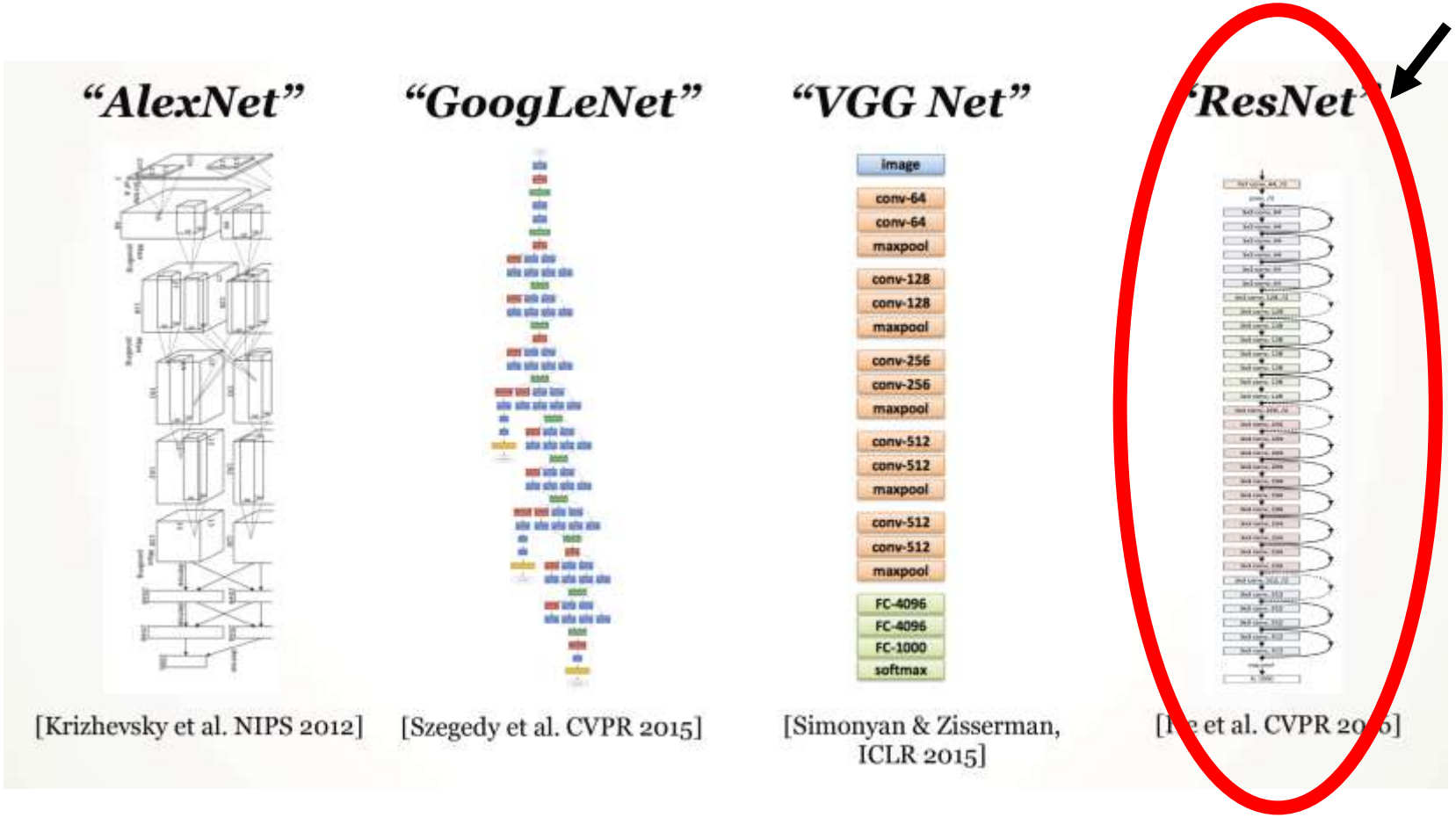
ResNet (2015)



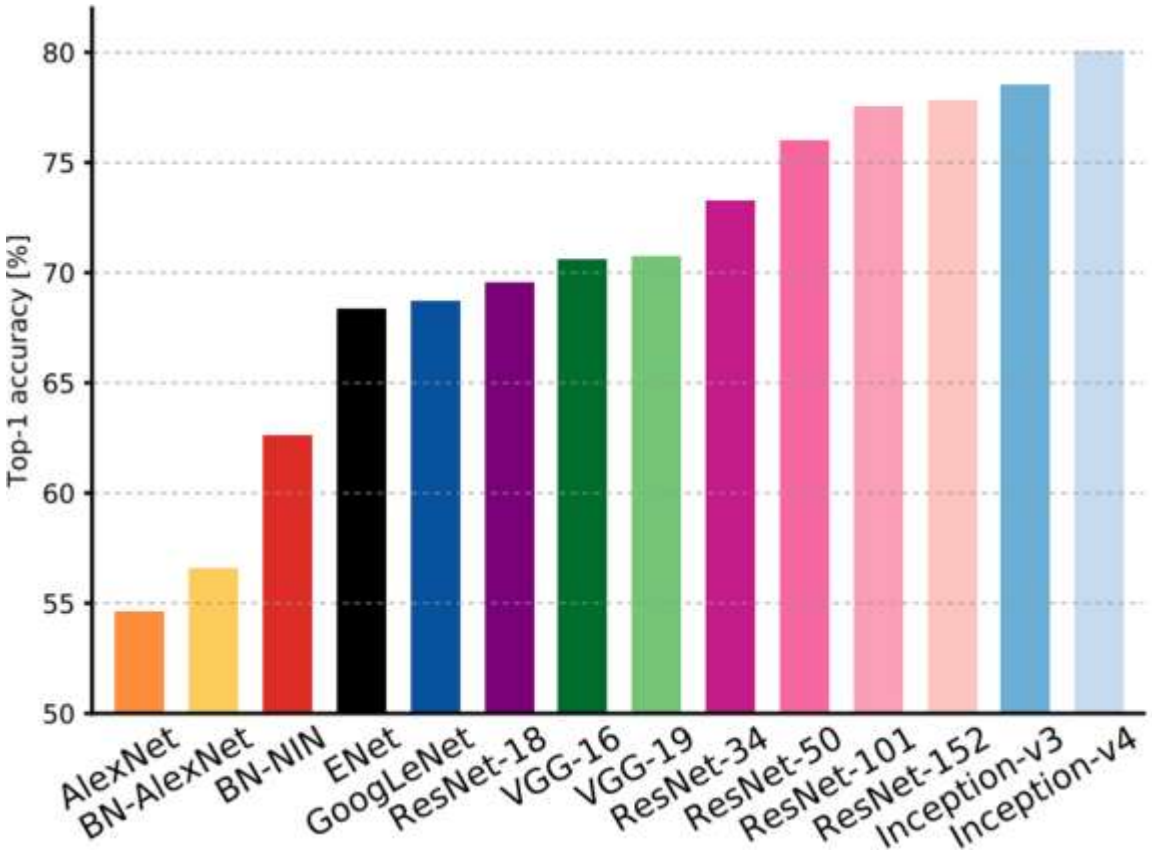
https://arxiv.org/pdf/1512.03385.pdf

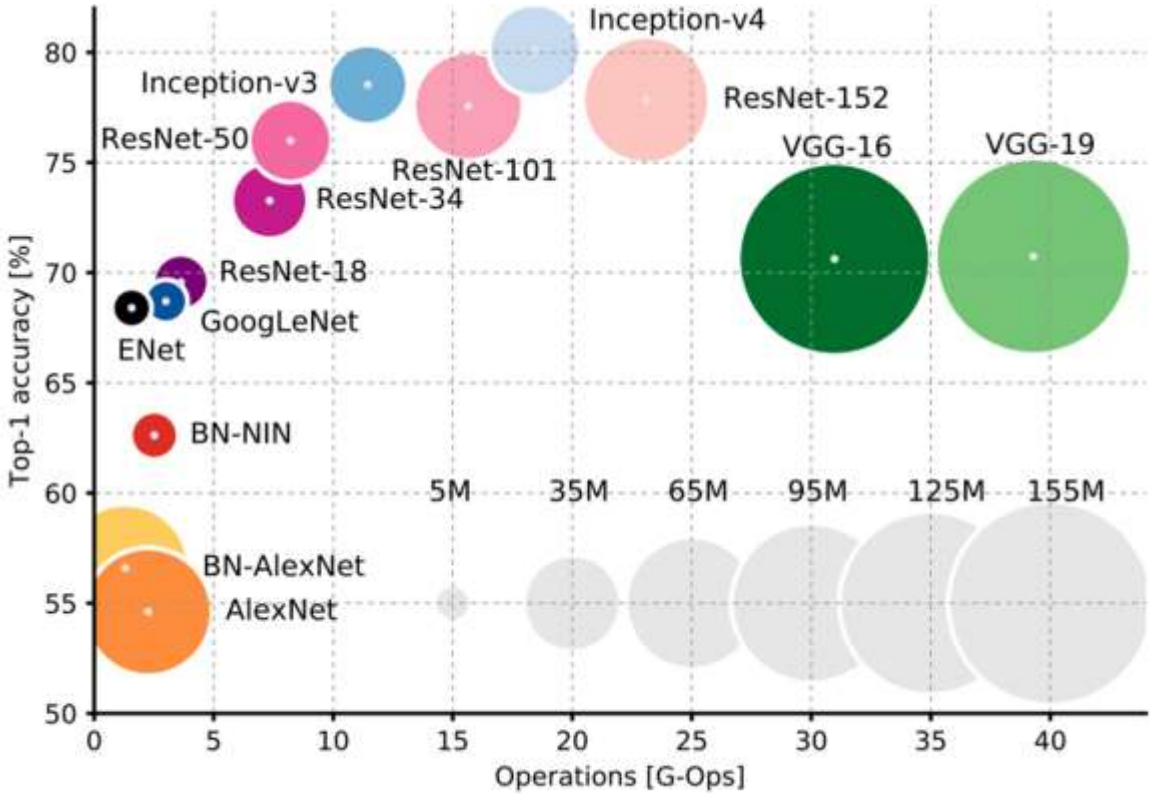
CNN网络结构

最常用的选择

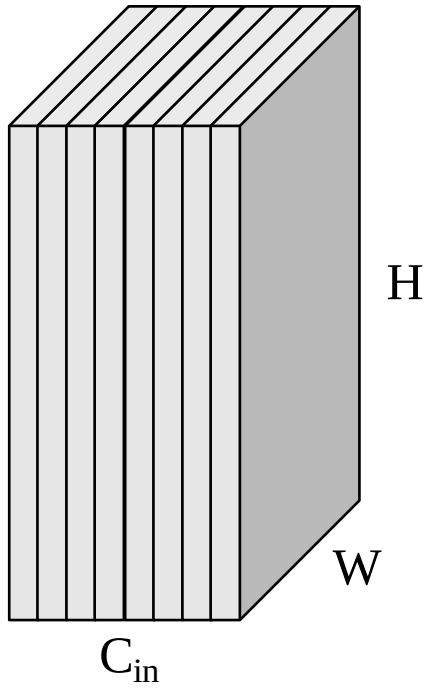


模型复杂度

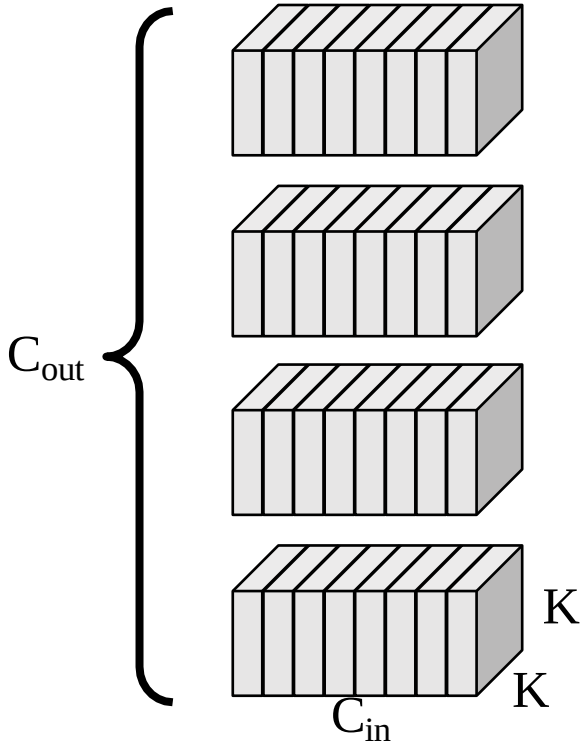

Top-1 accuracy [%]
80
75
70
65
60
55
50
AlexNet
BN-AlexNet
BN-NIN
ENet
GoogLeNet
ResNet-18
VGG-16
VGG-19
ResNet-34
ResNet-50
ResNet-101
ResNet-152
Inception-v3
Inception-v4


Top-1 accuracy [%]
80
75
70
65
60
55
50
0
5
10
15
20
25
30
35
40
Operations [G-Ops]
Inception-v4
Inception-v3
ResNet-152
ResNet-50
VGG-16
VGG-19
ResNet-101
ResNet-34
ResNet-18
GoogLeNet
ENet
BN-NIN
5M
35M
65M
95M
125M
155M
BN-AlexNet
AlexNet

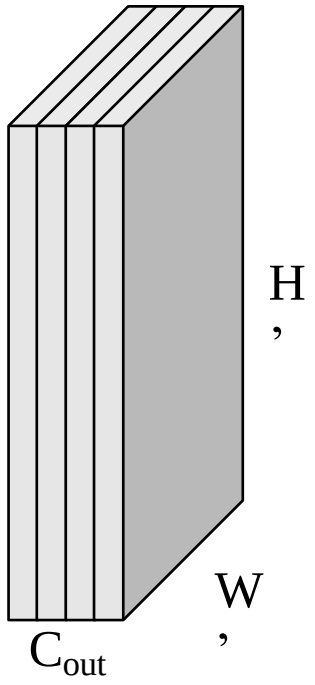
卷积层



输入: C_{in} x H x W

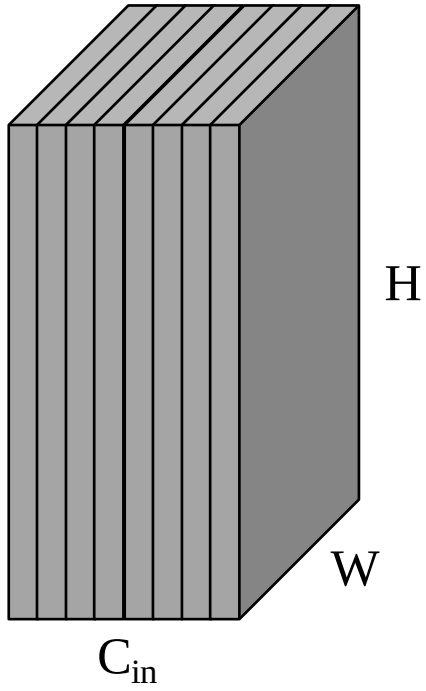


权重: C_{out} x C_{in} x H x W

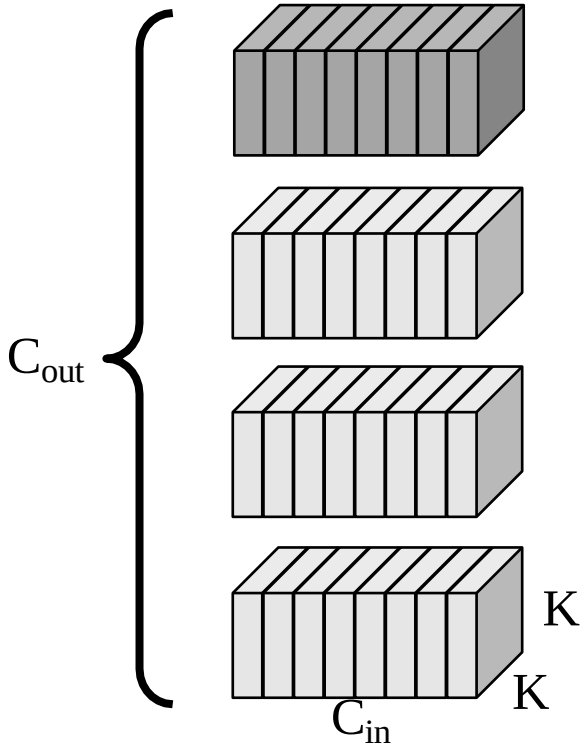


输出: C_{out} x H’ x W’

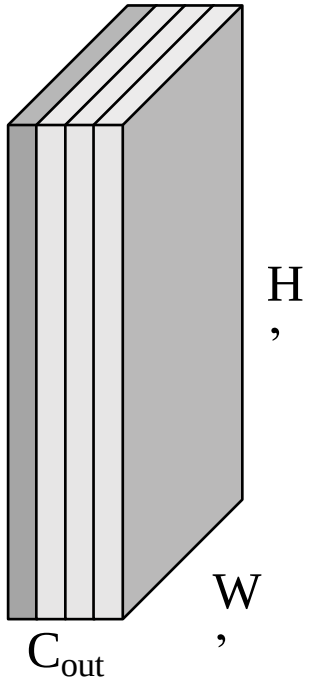
卷积层



输入: C_{in} x H x W

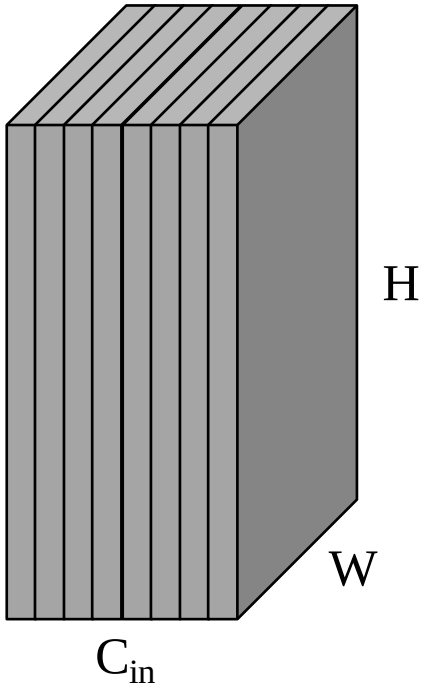


权重: C_{out} x C_{in} x H x W

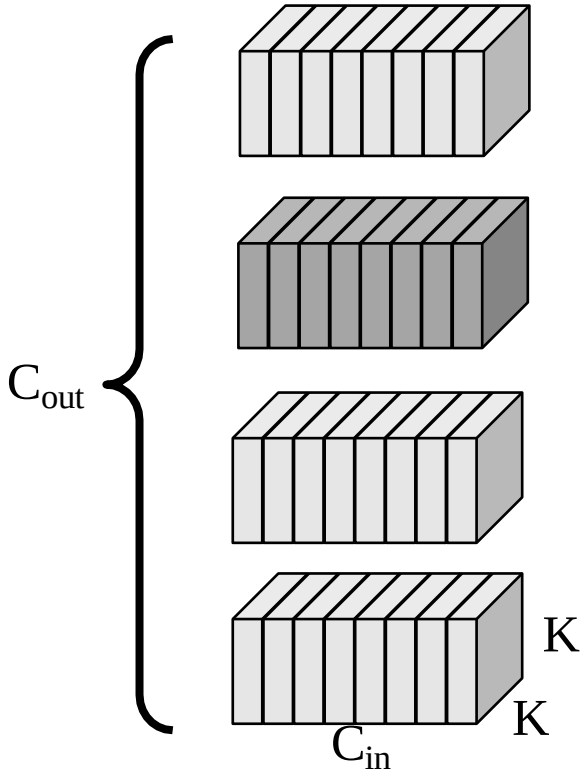


输出: C_{out} x H’ x W’

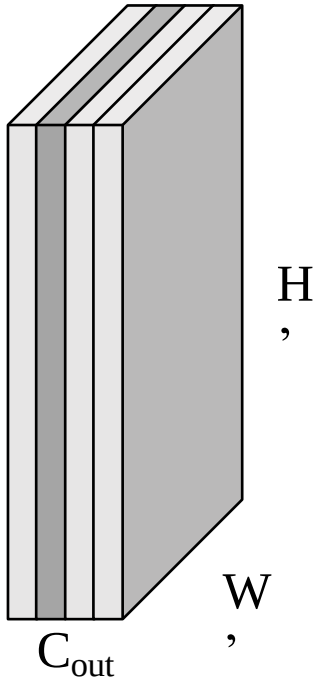
卷积层



输入: C_{in} x H x W



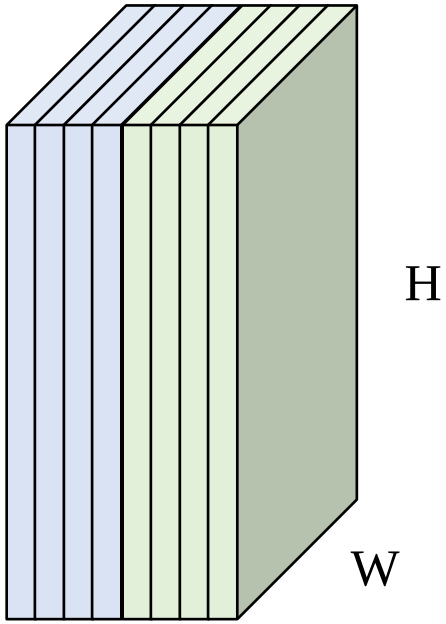
权重: C_{out} x C_{in} x H x W



输出: C_{out} x H x W

Group Convolution 组卷积层

把输入通道分为 G 组
每组 (C_{in}/G) 个通道



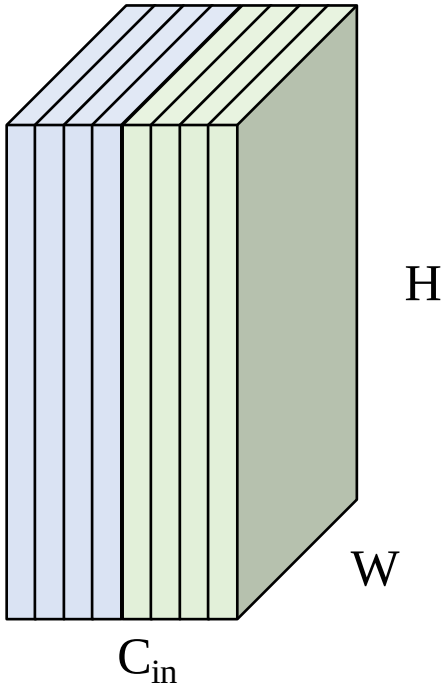
输入: C_{in} x H x W

权重: C_{out} x (C_{in} / G) x H x W

输出: C_{out} x H’ x W’

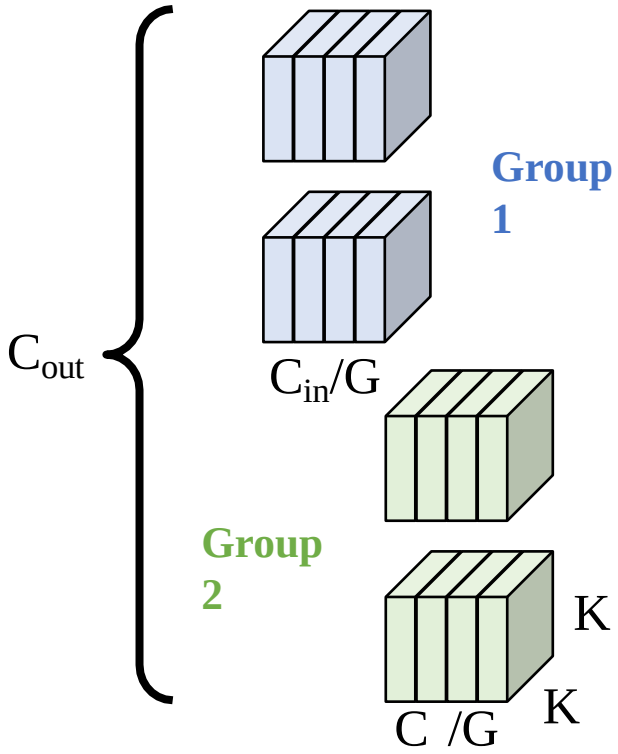
Group Convolution 组卷积层

把输入通道分为 G 组
每组 (C_{in}/G) 个通道

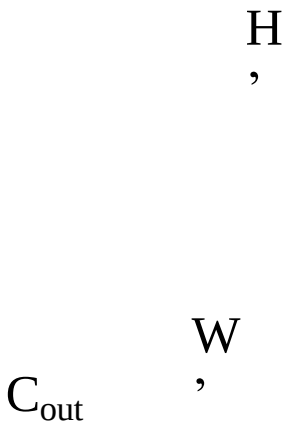


输入: C_{in} x H x W

滤波同样分为 G 组， 每组只
看自己组内的输入通道



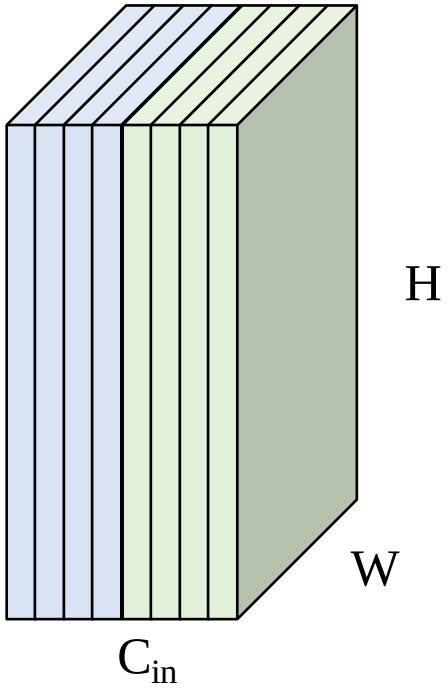
权重: C_{out} x (C_{in} / G) x H x W



输出: C_{out} x H' x W'

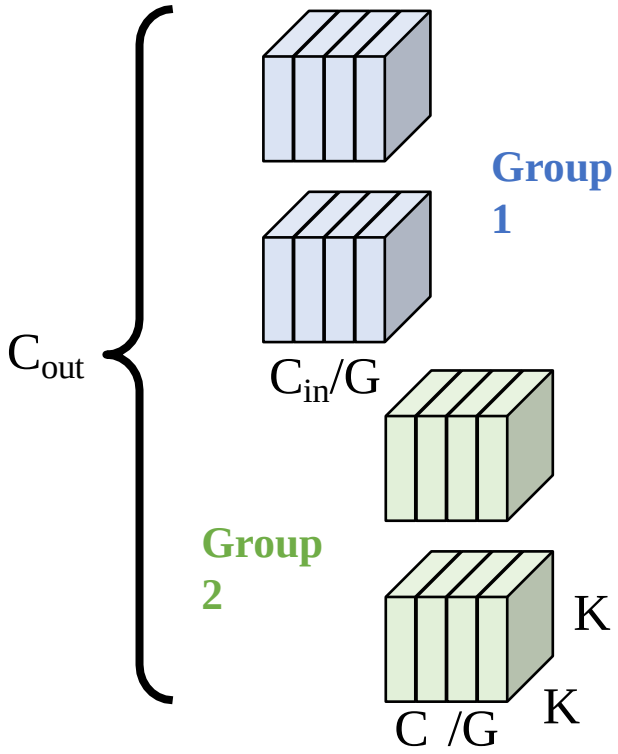
Group Convolution 组卷积层

把输入通道分为 G 组
每组 (C_{in}/G) 个通道



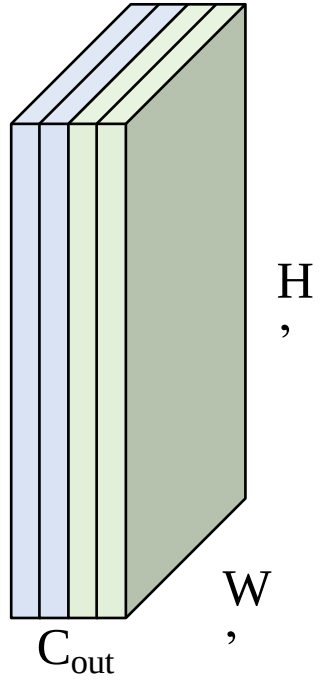
输入: C_{in} x H x W

滤波同样分为 G 组，每组只看自己组内的输入通道



权重: C_{out} x (C_{in} / G) x H x W

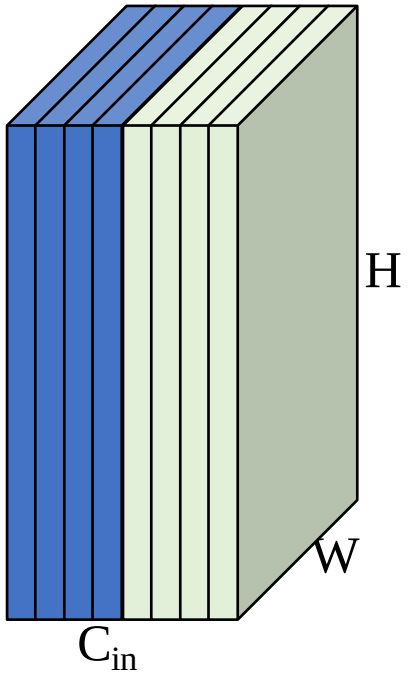
输出依照卷积滤波，只看某组内的输入通道



输出: C_{out} x H' x W'

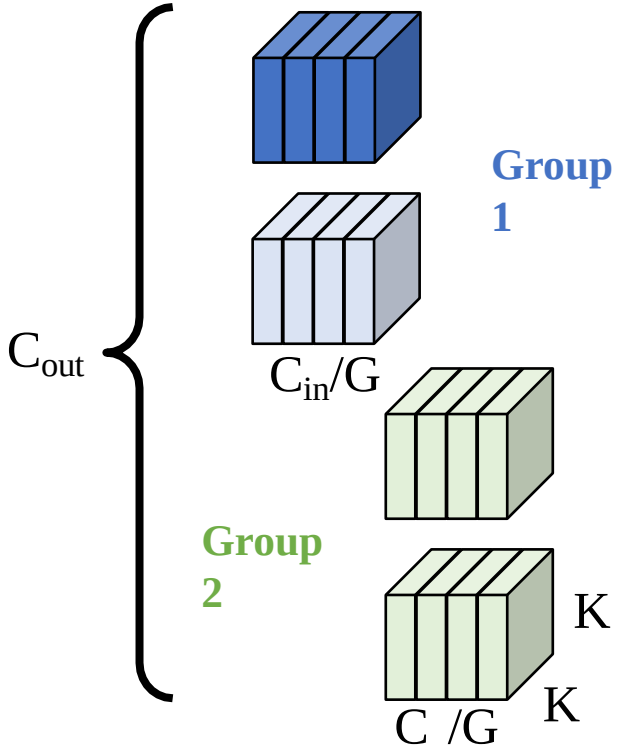
Group Convolution 组卷积层

把输入通道分为 G 组
每组 (C_{in}/G) 个通道



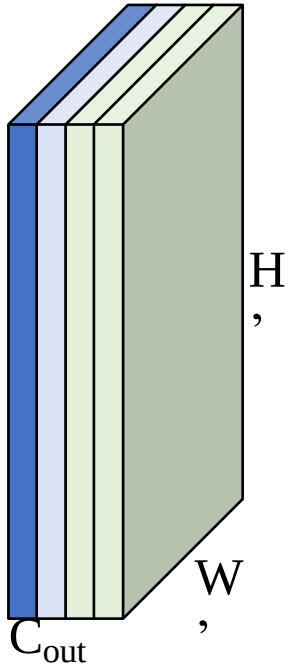
输入: C_{in} x H x W

滤波同样分为 G 组，每组只看自己组内的输入通道



权重: C_{out} x (C_{in} / G) x H x W

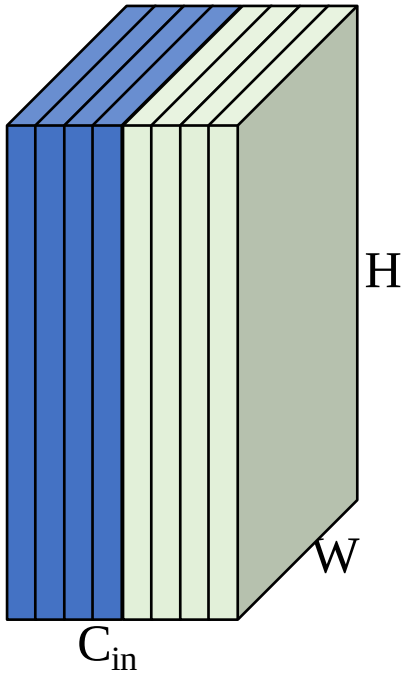
输出依照卷积滤波，只看某组内的输入通道



输出: C_{out} x H' x W'

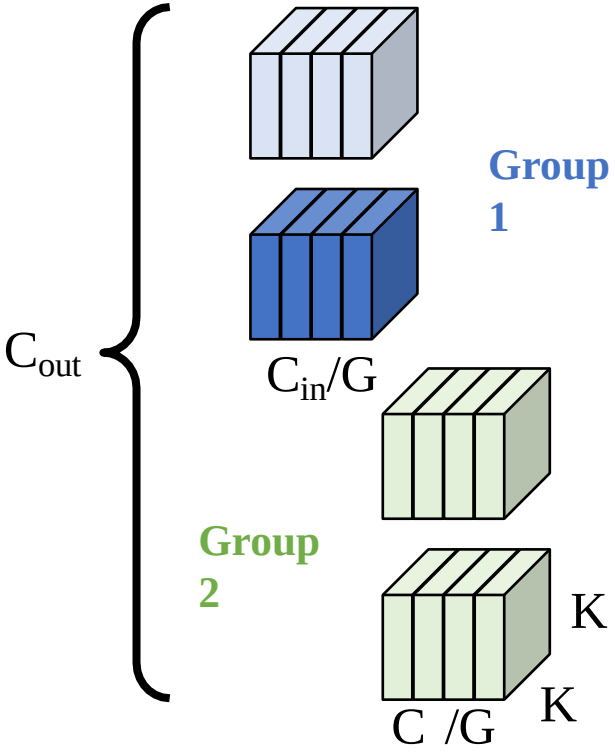
Group Convolution 组卷积层

把输入通道分为 G 组
每组 (C_{in}/G) 个通道



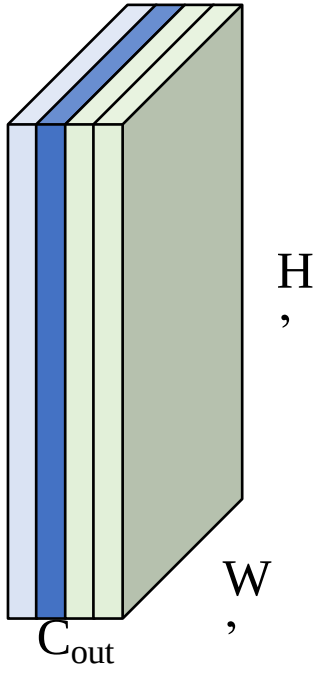
输入: C_{in} x H x W

滤波同样分为 G 组，每组只看自己组内的输入通道



权重: C_{out} x (C_{in} / G) x H x W

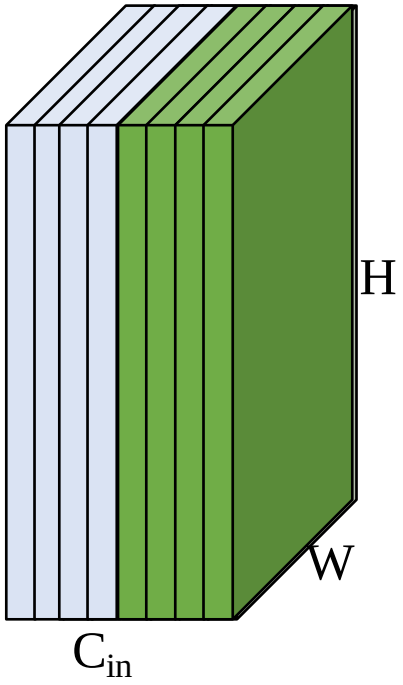
输出依照卷积滤波，只看某组内的输入通道



输出: C_{out} x H' x W'

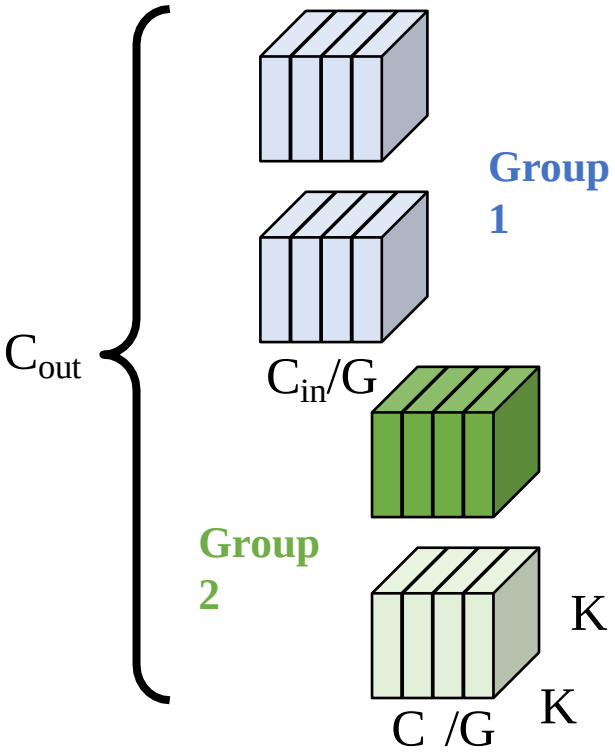
Group Convolution 组卷积层

把输入通道分为 G 组
每组 (C_{in}/G) 个通道



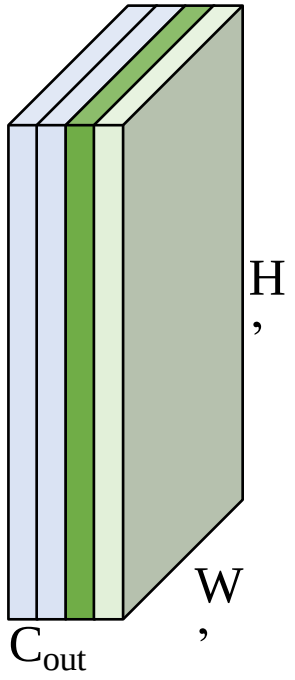
输入: C_{in} x H x W

滤波同样分为 G 组，每组只看自己组内的输入通道



权重: C_{out} x (C_{in} / G) x H x W

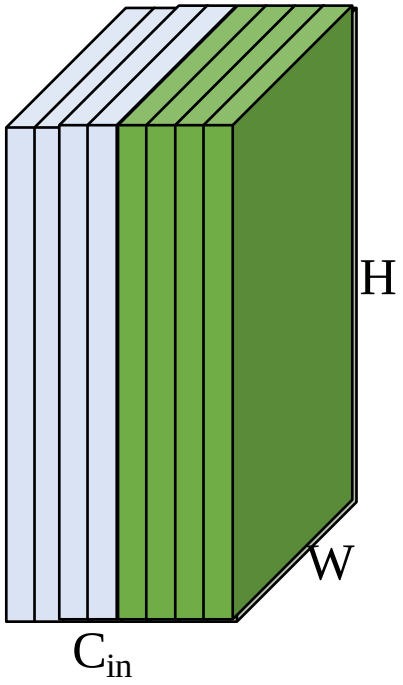
输出依照卷积滤波，只看某组内的输入通道



输出: C_{out} x H' x W'

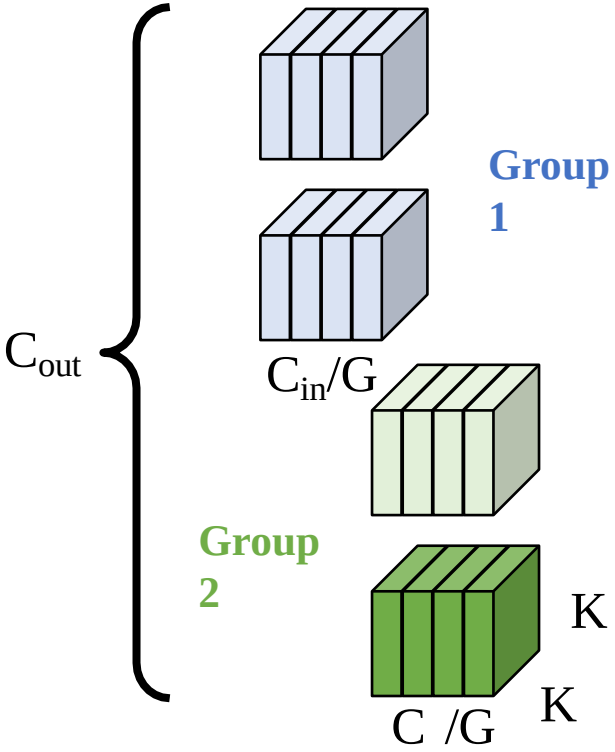
Group Convolution 组卷积层

把输入通道分为 G 组
每组 (C_{in}/G) 个通道



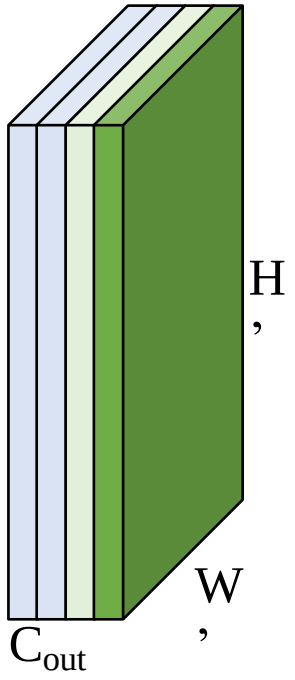
输入: C_{in} x H x W

滤波同样分为 G 组，每组只看自己组内的输入通道



权重: C_{out} x (C_{in} / G) x H x W

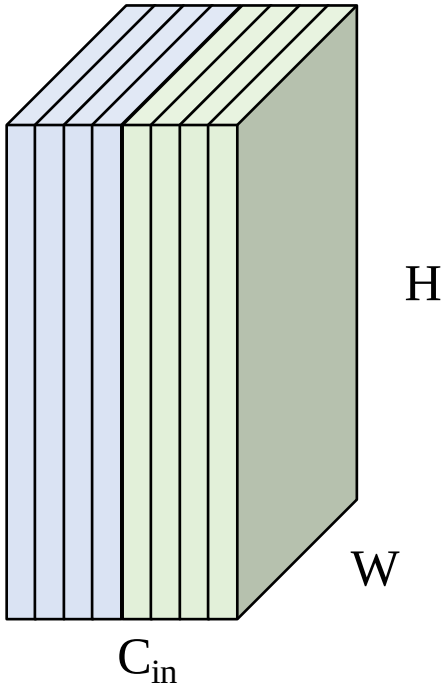
输出依照卷积滤波，只看某组内的输入通道



输出: C_{out} x H' x W'

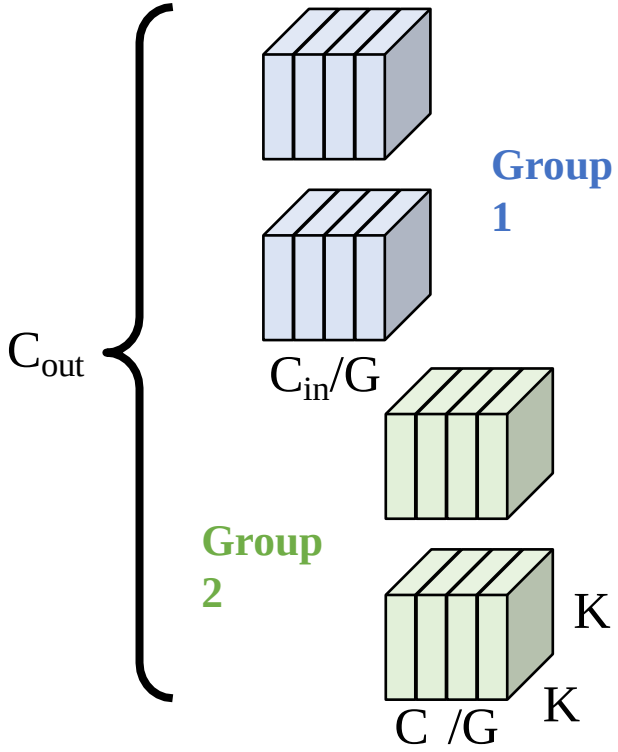
Group Convolution 组卷积层

把输入通道分为 G 组
每组 (C_{in}/G) 个通道



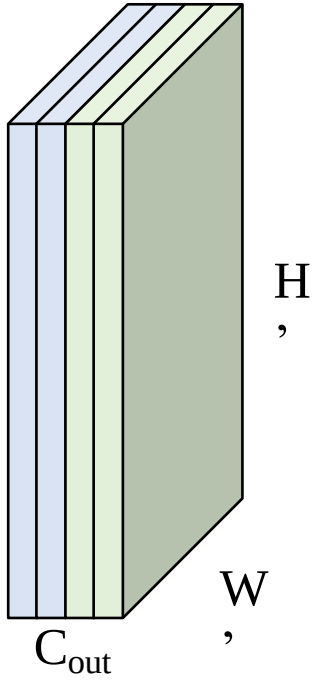
输入: C_{in} x H x W

滤波同样分为 G 组，每组只看自己组内的输入通道



权重: C_{out} x (C_{in} / G) x H x W

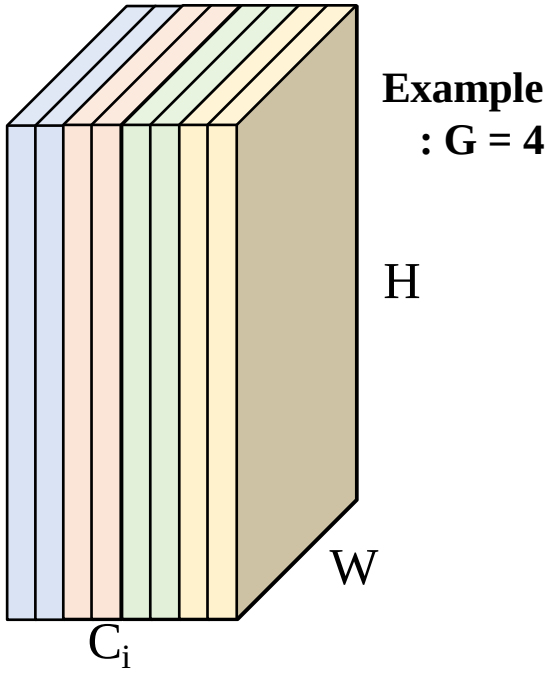
输出依照卷积滤波，只看某组内的输入通道



输出: C_{out} x H' x W'

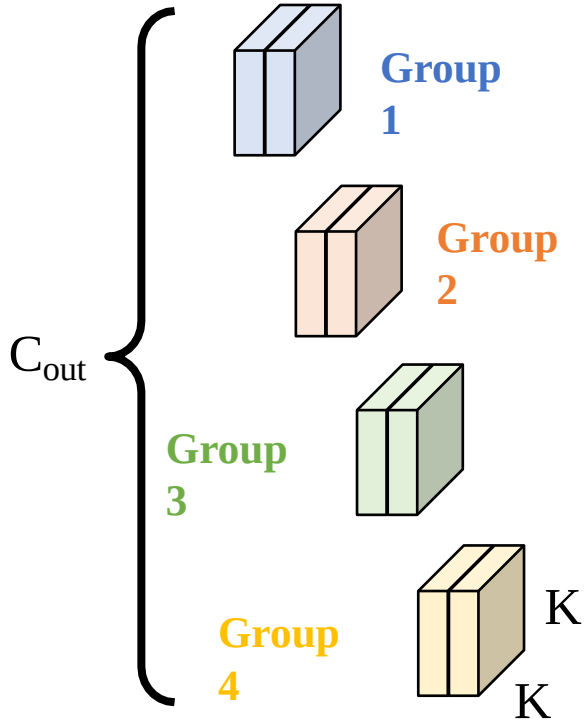
Group Convolution 组卷积层

把输入通道分为 G 组
每组 (C_{in}/G) 个通道



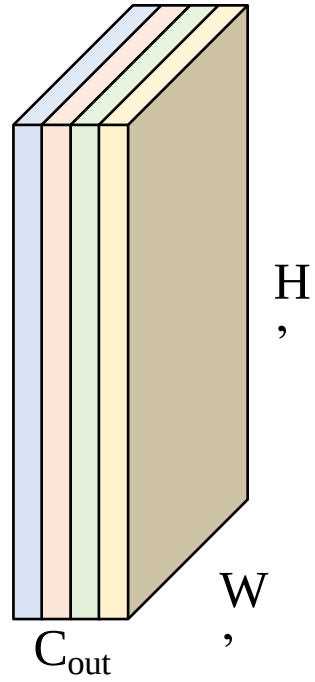
输入: $^{n}C_{in}$ x H x W

滤波同样分为 G 组，每组只
看自己组内的输入通道



权重: C_{out} x (C_{in} / G) x H x W

输出依照卷积滤波，只看某
组内的输入通道

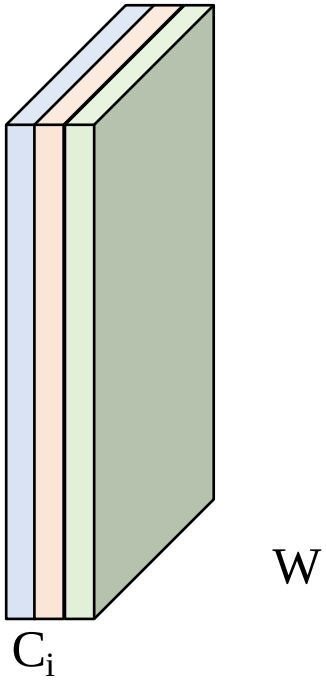


输出: C_{out} x H' x W'

Depthwise Convolution 逐深度卷积层

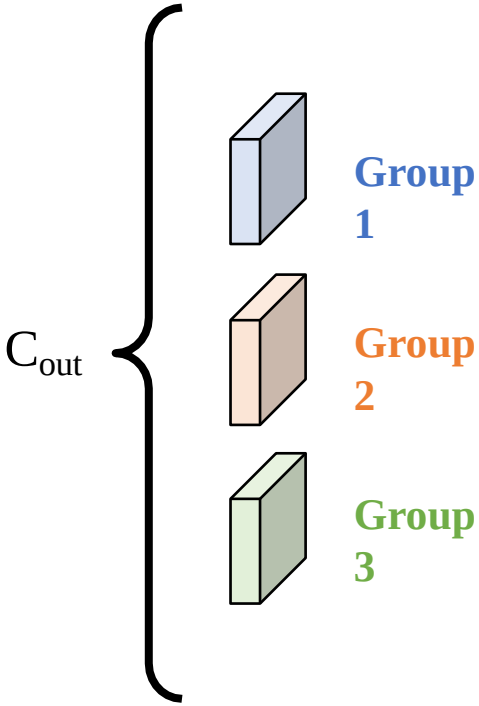
把输入通道各自分为一组，

$G = C_{in}$



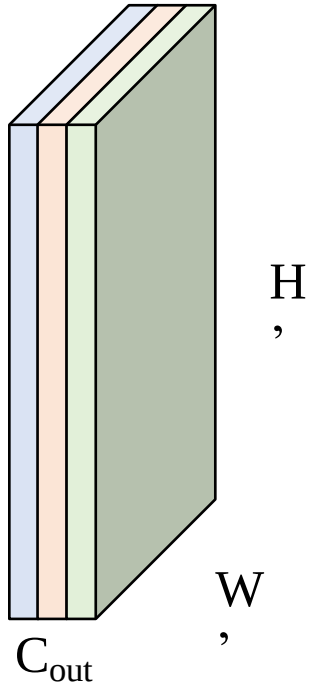
输入: $^{n}C_{in}$ x H x W

滤波同样分为 G **组**，每组只看自己组内的输入通道



权重: C_{out} x 1x H x W

输出依照卷积滤波，只看某组内的输入通道

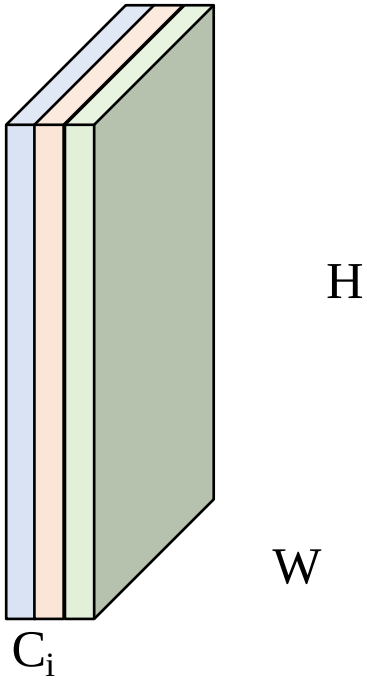


输出: C_{out} x H' x W'

Depthwise Convolution逐深度卷积层

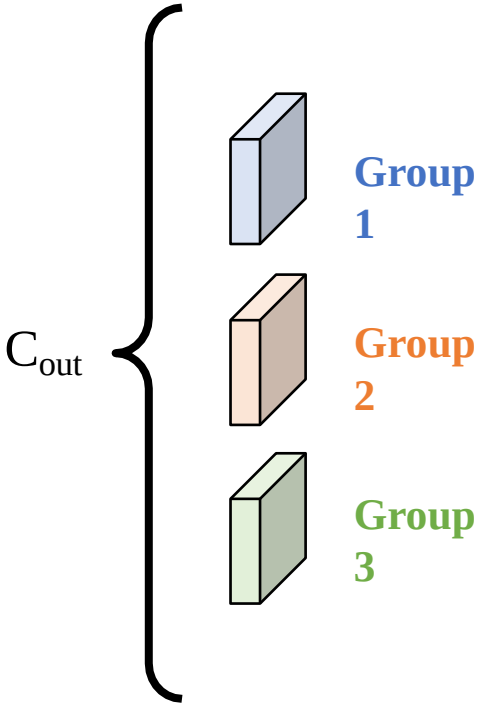
把输入通道各自分为一组，

$G = C_{in}$



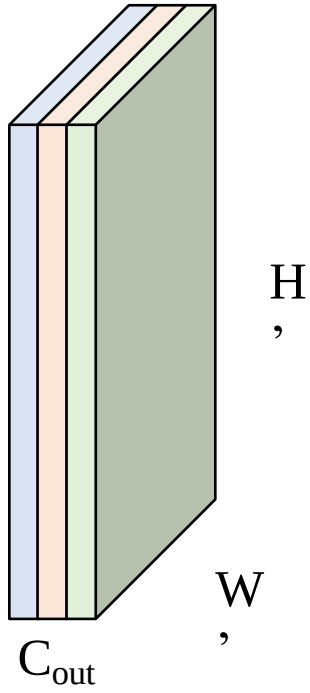
输入: $^{n}C_{in}$ x H x W

滤波同样分为 G **组**，每组只看自己组内的输入通道



权重: C_{out} x 1x H x W

输出与输入通道对应，
卷积只做空间信息混合

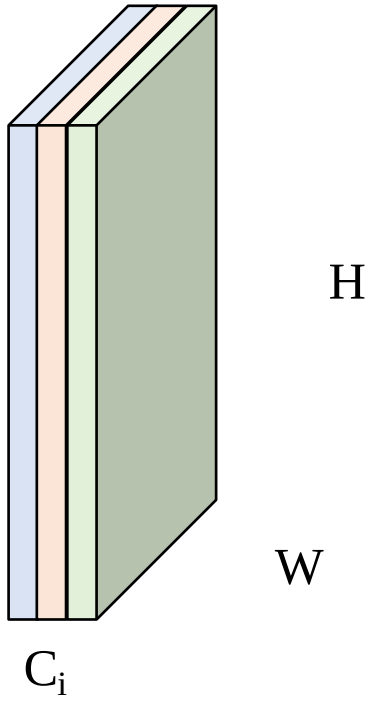


输出: C_{out} x H' x W'

Depthwise Convolution逐深度卷积层

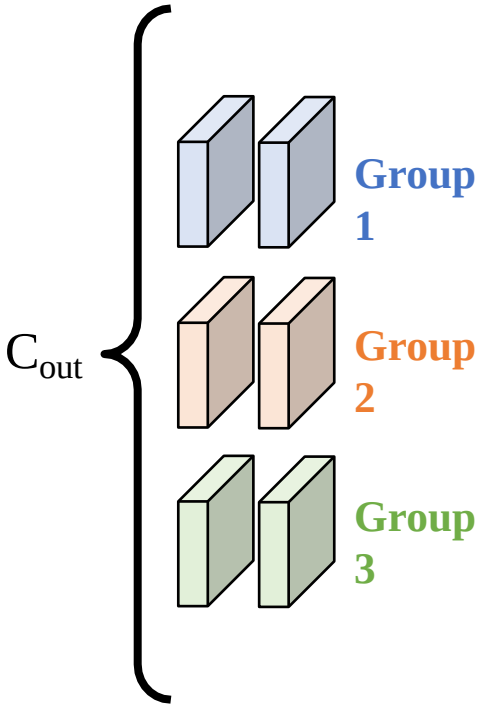
把输入通道各自分为一组，

$G = C_{in}$



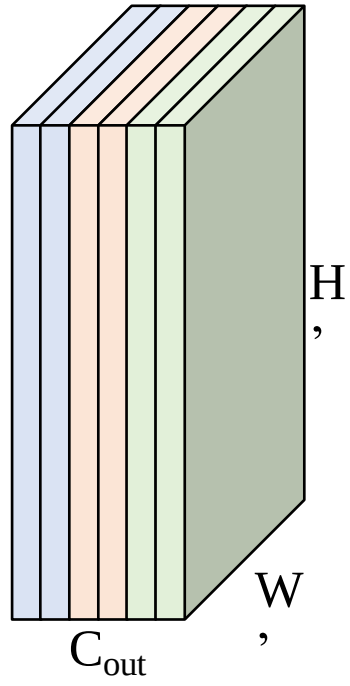
输入: $^{n}C_{in}$ x H x W

滤波同样分为 G **组**，每组只看自己组内的输入通道



权重: C_{out} x 1x H x W

输出与输入通道对应，
卷积只做空间信息混合

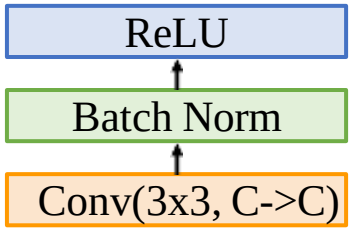


输出: C_{out} x H' x W'

MobileNet (2017)

- 目标: 减少网络参数
- 解决方案: Depthwise separable convolutions
  “分离卷积”

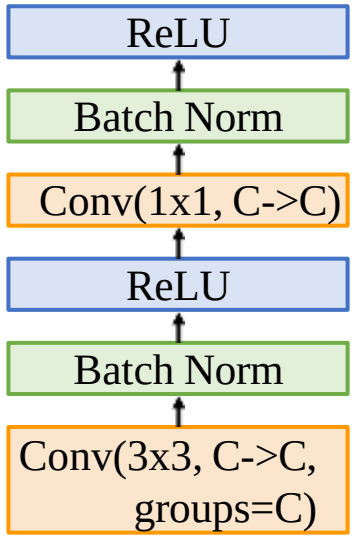
Standard Convolution Block
Total cost: $9C^2HW$



$9C^2HW$

Speedup = $9C^2/(9C+C^2)$
= $9C/(9+C)$
=> 9 (as C->inf)

Depthwise Separable Convolution
Total cost: $(9C + C^2)HW$



C^2HW

“Pointwise Convolution”

$9CHW$

“Depthwise Convolution”

Howard et al, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications”, arXiv 2017
Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions”, CVPR 2017

MobileNet (2017)

- 与 InceptionV3 接近的准确度
- 十倍的速度，十分之一的参数

Layer/Modification	Million Mult-Adds	Million Parameters
Convolution	462	2.36
Depthwise Separable Conv	52.3	0.27

Table 8. MobileNet Comparison to Popular Models

Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
1.0 MobileNet-224	70.6%	569	4.2
GoogleNet	69.8%	1550	6.8
VGG 16	71.5%	15300	138

Table 9. Smaller MobileNet Comparison to Popular Models

Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
0.50 MobileNet-160	60.2%	76	1.32
Squeezenet	57.5%	1700	1.25
AlexNet	57.2%	720	60

Table 10. MobileNet for Stanford Dogs

Model	Top-1 Accuracy	Million Mult-Adds	Million Parameters
Inception V3 [18]	84%	5000	23.2
1.0 MobileNet-224	83.3%	569	3.3
0.75 MobileNet-224	81.9%	325	1.9
1.0 MobileNet-192	81.9%	418	3.3
0.75 MobileNet-192	80.5%	239	1.9

https://arxiv.org/pdf/1704.04861.pdf

训练与测试

训练时最小化: $-\log\left(\frac{\exp((Wx)_{y_i}}{\sum_k \exp((Wx)_k))}\right)$

测试时，对预测类别 $\hat{y}_i$:

Accuracy 准确率: $\frac{1}{n}\sum_{i=1}^{n} 1(y_i = \hat{y}_i)$

多类别测试

这样图分类为猫还是狗?

为了避免有歧义的图像影响测试，一般给出五个预测结果 (top-5 accuracy):

只要五个预测有一个对就算正确.

Image credit: Coco dataset

准确率

	Top 1 Error	Top 5 Error
Best Pre-Deep (~2012)	-	26.2%
Alexnet, 2012	43.5%	20.9%
VGG-16, 2014	28.4%	9.6%
ResNet-50, 2015	24.7%	7.8%
ResNet-152, 2015	21.7%	5.9%
ResNet-50 done better, 2018	20.7%	5.4%
Swin Transf., 2021	15.5%	-
ConvNeXt, 2022	14.5%	-
CoAtNet-7* 2021 (2B params!)	9.1%	-
Human*	-	5.1%

Many results from https://paperswithcode.com/sota/image-classification-on-imagenet . I am missing loads of great papers, and the numbers depend on tons of practical details. *Human – this number is from Andrej Karpathy and isn't really human performance with training but a ballpark. Resnet-50 one better = "Bag of Tricks for Image Classification with Convolutional Neural Networks", He et al.

训练CNN

- 下载数据集，准备GPU
- 初始化网络参数
- for epoch in range(epochs):
  - 打乱数据集
  - for each minibatch in datsaet.:
    - 加载数据
    - 计算梯度
    - 优化网络

从头训练CNN

需要找到一个初始的网络权重 **w**

- AlexNet: weights ~ Normal(0,0.01), bias = 1
- “Xavier” 初始化: Uniform($\frac{-1}{\sqrt{n}}, \frac{1}{\sqrt{n}}$) n为神经元个数
- “Kaiming” 初始化: Normal(0,$\sqrt{2/n}$)

Quick Quiz

哪一张图里有河马？

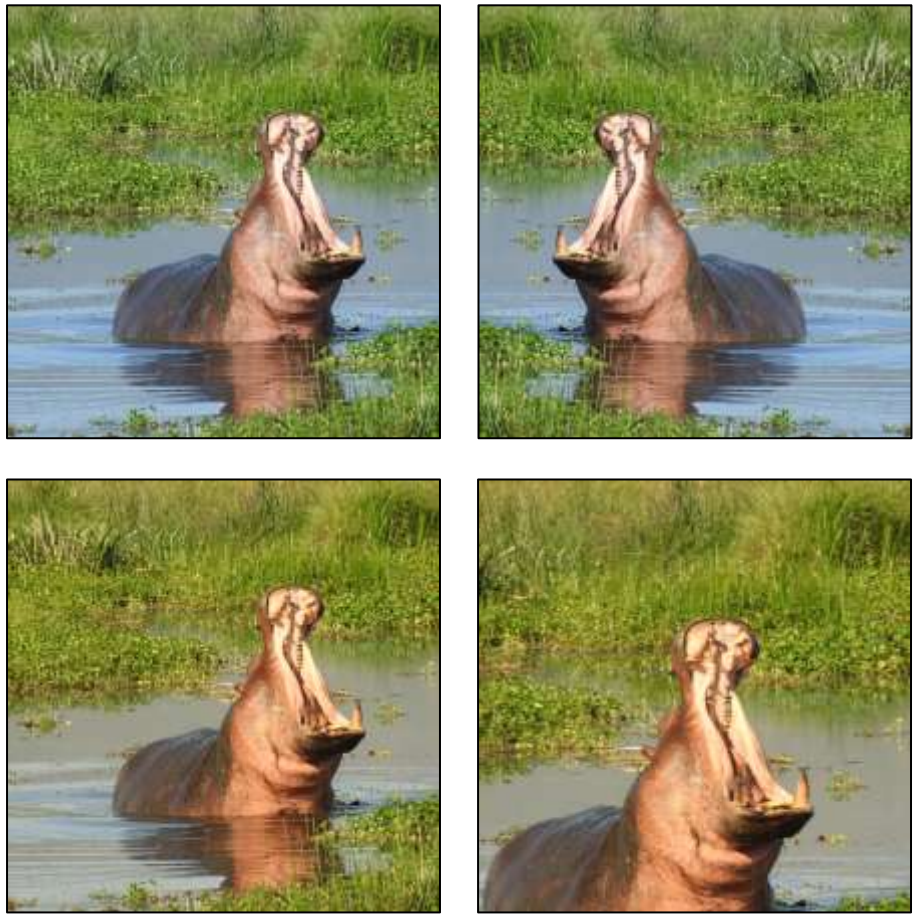
Horizontal Flip

Color Jitter

Image Cropping

训练CNN –Augmentation 图像增广

- 应用一些不会影响图像分类结果的变换
- 请注意，一定要确保不会影响图像的表达语义



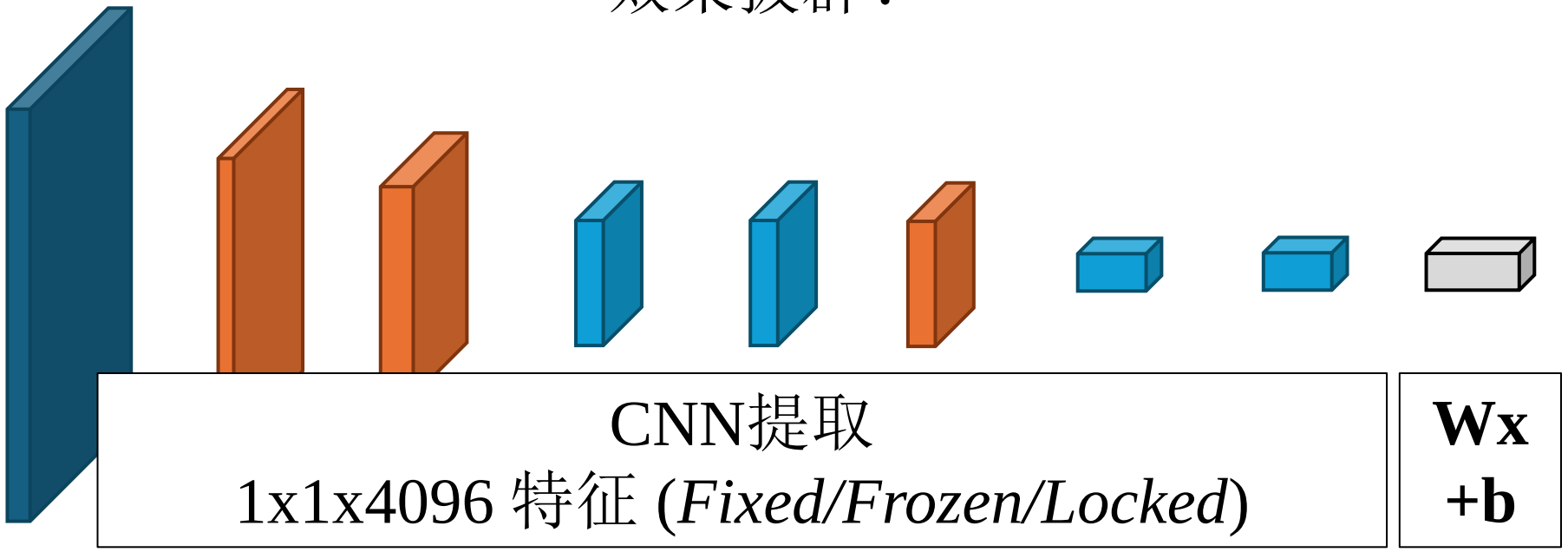
训练CNN – Fine-tuning 微调

- 如果你没有（没有大量的）训练数据呢?

预训练模型与特征

1. 从已有网络的某一层提取特征
2. 当作你的初始特征.
3. 训练线性模型.

效果拔群!

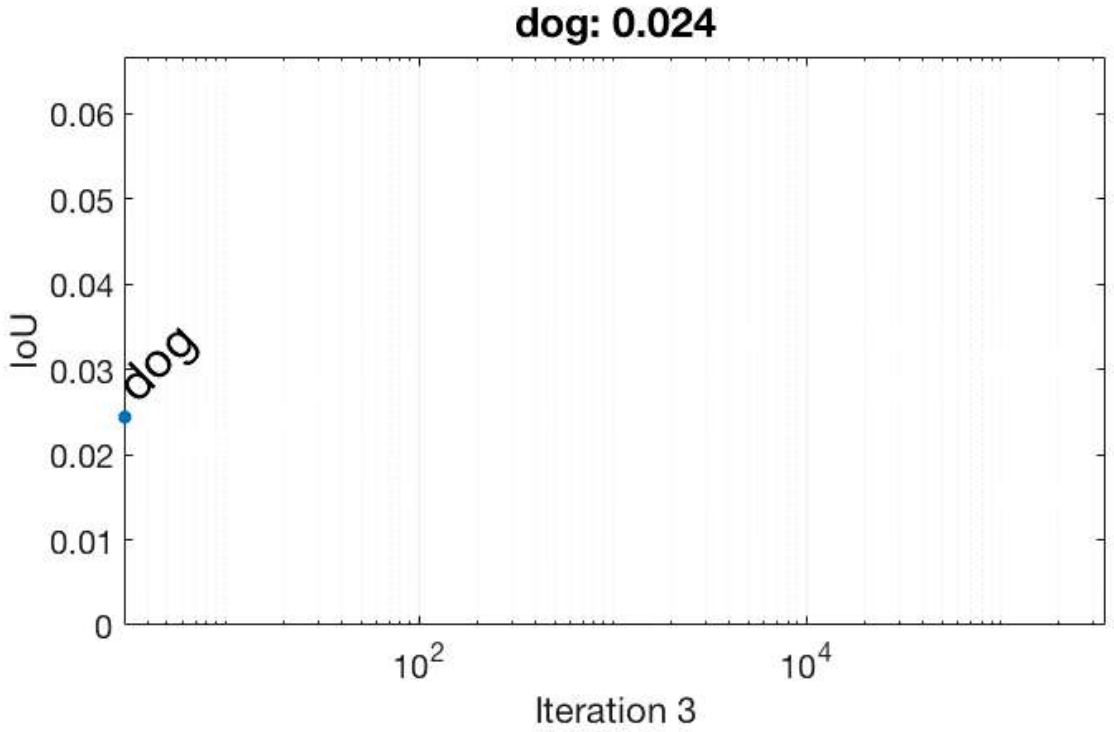


微调：迁移学习

- 与其初始化随机参数，用已经训练好的网络参数.
- 最常用的是在ImageNet训练的网络.
- 也有其他的特定任务用特定的数据集，目前也有更新更大的数据集出现

微调：迁移学习

为什么能实现?
从 物体(dog) 迁移到 场景(waterfall)



Bau and Zhou et al. Network Dissection: Quantifying Interpretability of Deep Visual Representations. CVPR 2017.

一般性建议

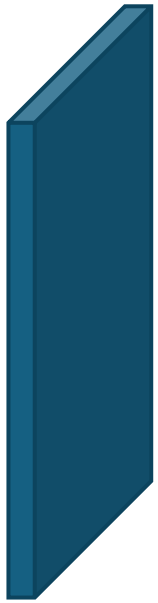
- <10K images: 用预训练特征
- **永远都要尝试微调 finetune**
- >100K images: 从头训练

- 思考：为什么？

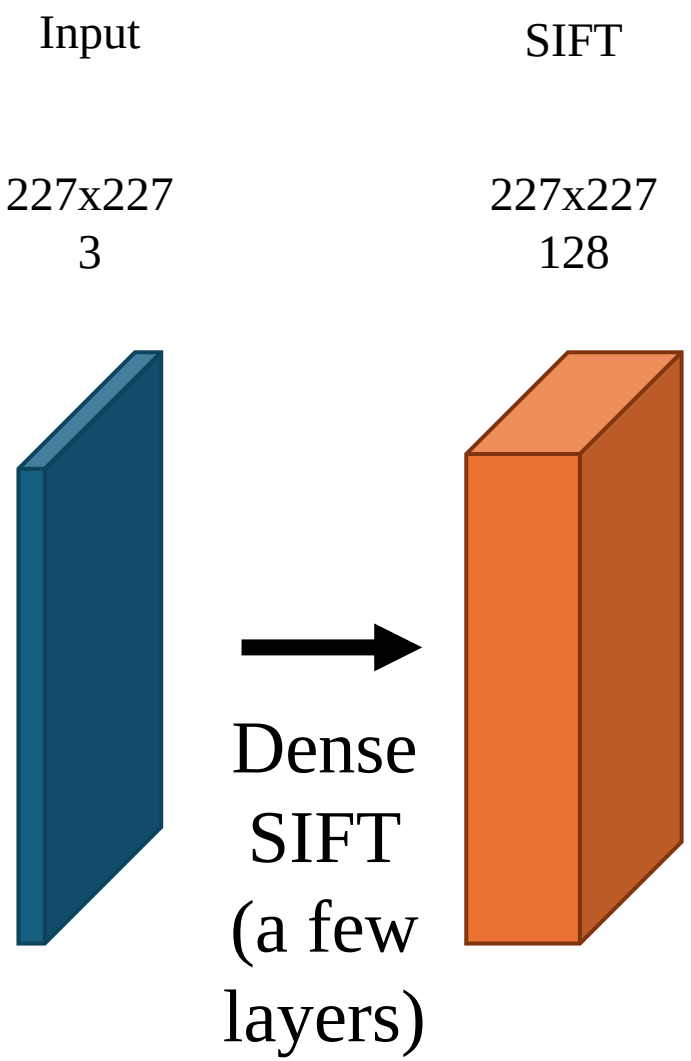
回顾：分类识别 传统模型 与 深度学习

Input

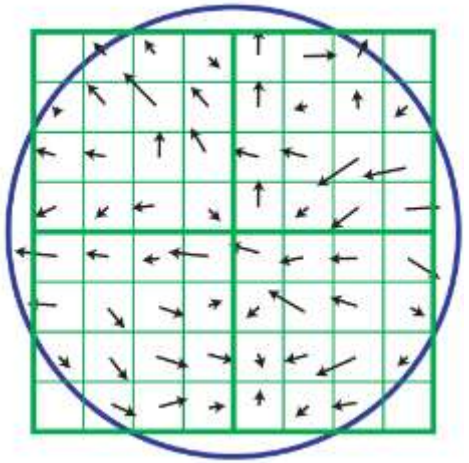
227x227
3



回顾：分类识别 传统模型 与 深度学习



回顾：可以基于图像梯度直方图计算描述子。在每个像素处密集地执行。



回顾：分类识别 传统模型 与 深度学习

Input

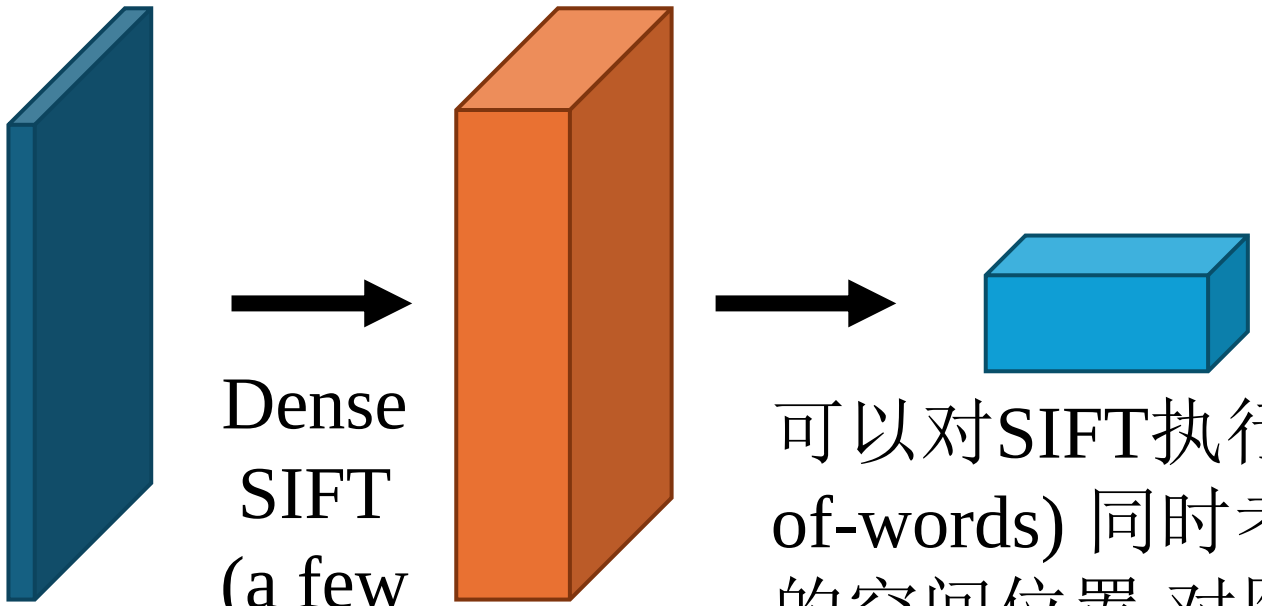
227x227
3

SIFT

227x227
128

Bag of Words

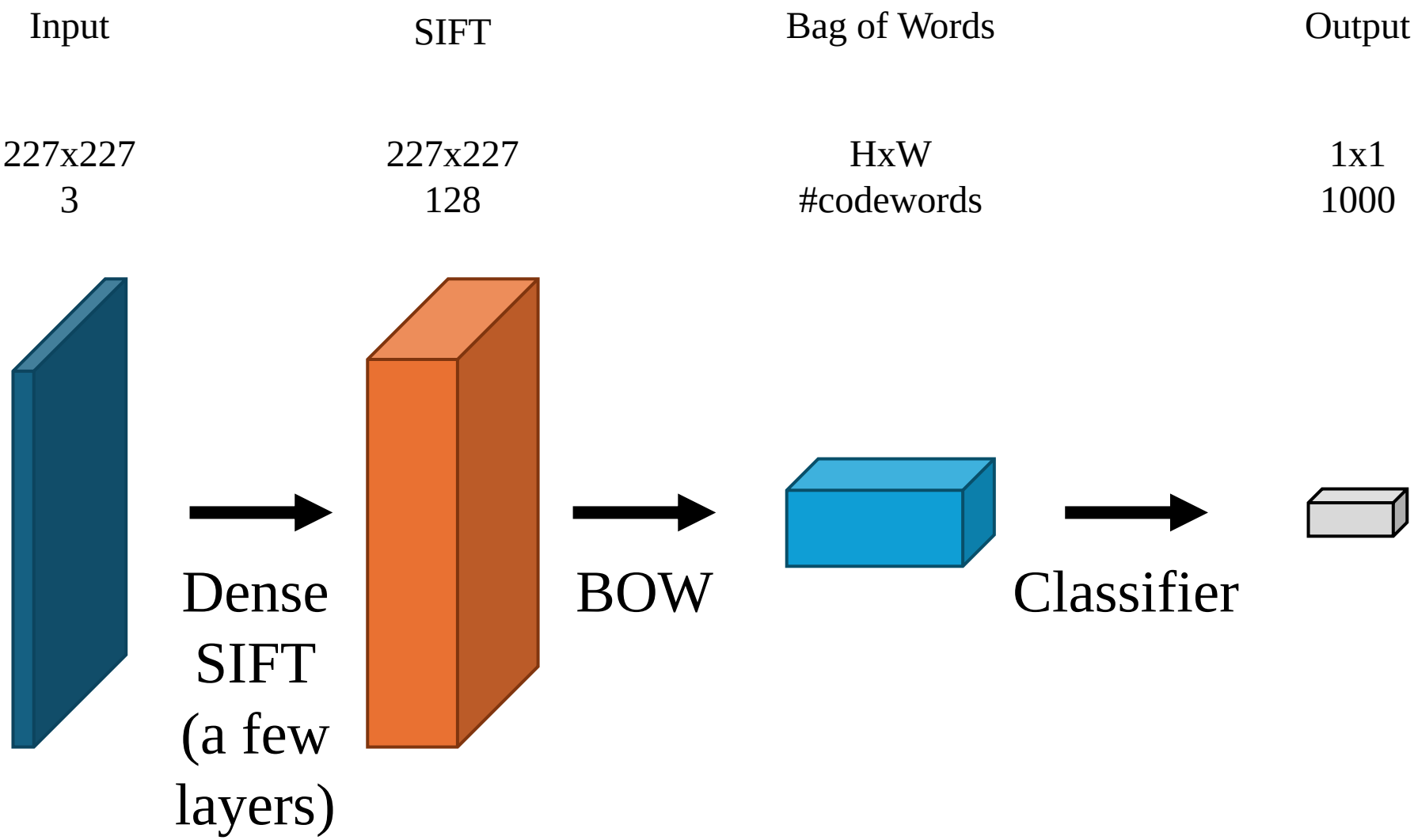
HxW
#codewords



Dense SIFT (a few layers)

可以对SIFT执行类似词袋的技术(bag-of-words) 同时考虑到描述子在图像中的空间位置,对图像的局部区域进行编码，并改进图像分类或检索的性能。

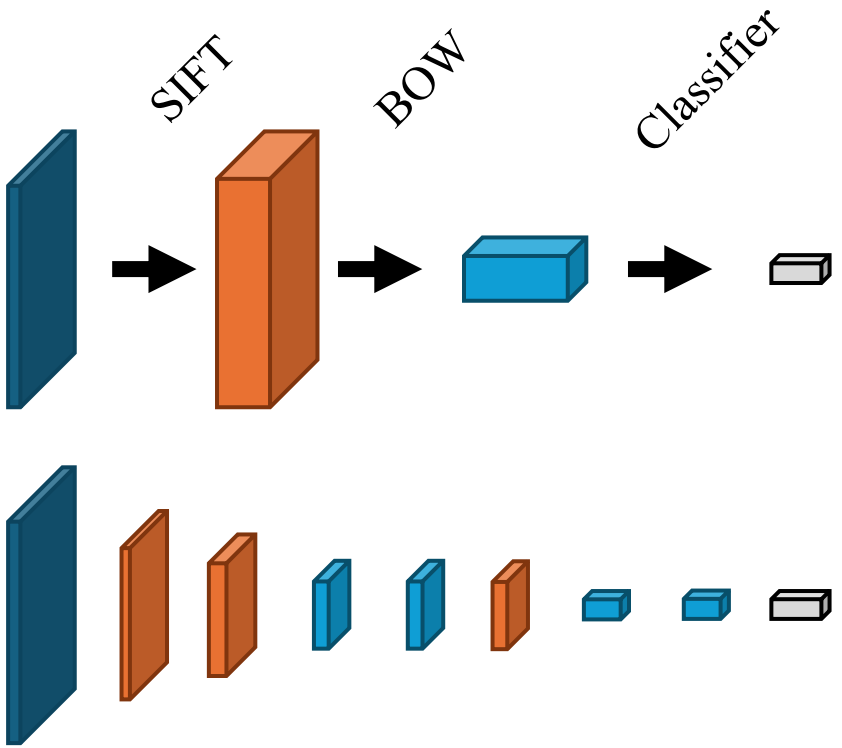
回顾：分类识别 传统模型 与 深度学习



回顾：分类识别 传统模型 与 深度学习

传统方法　手工设计的流水线作业

深度方法　可学习的卷积、池化、归一化的组合



有什么区别?

传统步骤中的各个组件通常是独立设计和优化的，不会“相互交流”或从数据中学习大量参数。相比之下，深度学习方法中的层是通过学习数据自动优化参数的，各层之间会有信息的交流。