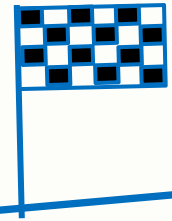
第07讲 深度神经网络的革新

欧 新 宇

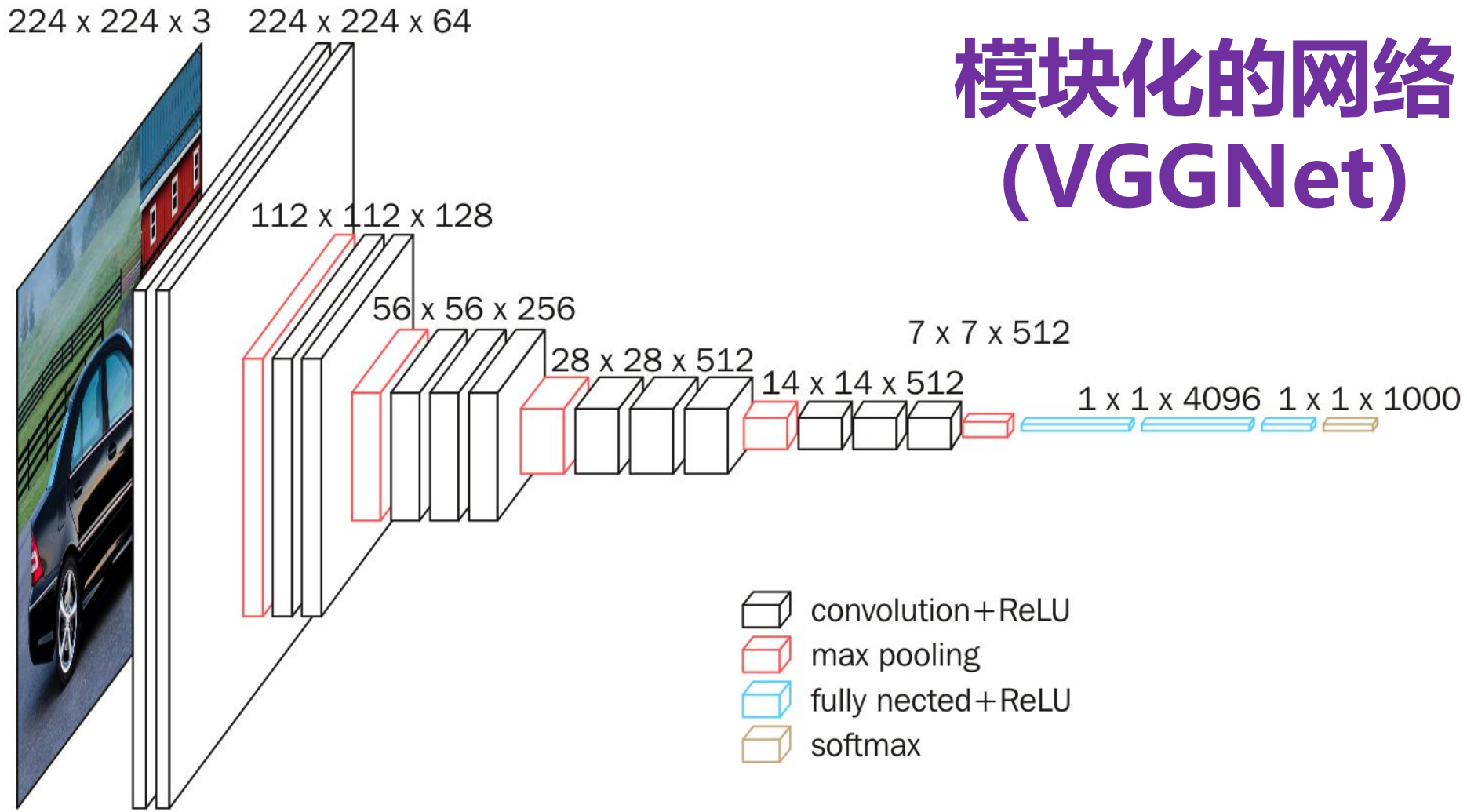
Thanks to LiMu CS239p

计算机视觉概述

- 网络的模块化：VGGNet
- 网络中的网络：Network in Network
- 多分支网络：GoogLeNet
- 残差网络：ResNet

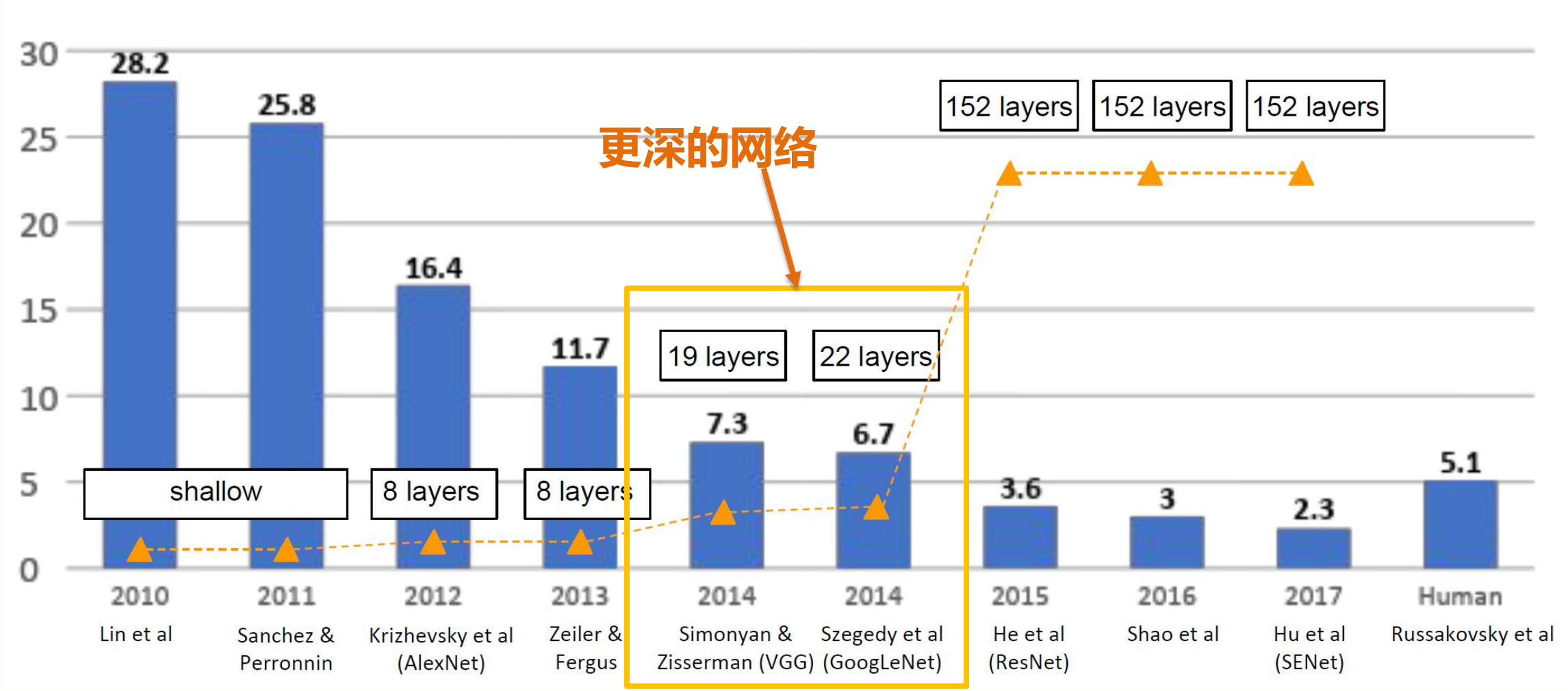


模块化的网络(VGGNet)



模块化的网络（VGGNet）

ImageNet 大规模视觉识别挑战赛(ILSVRC)



VGG的特点

VGG的特点

简单就是王道 - 基于模块的设计

AlexNet被认为是更大、更深的LeNet，通过深度化实现了更好的分类精度。

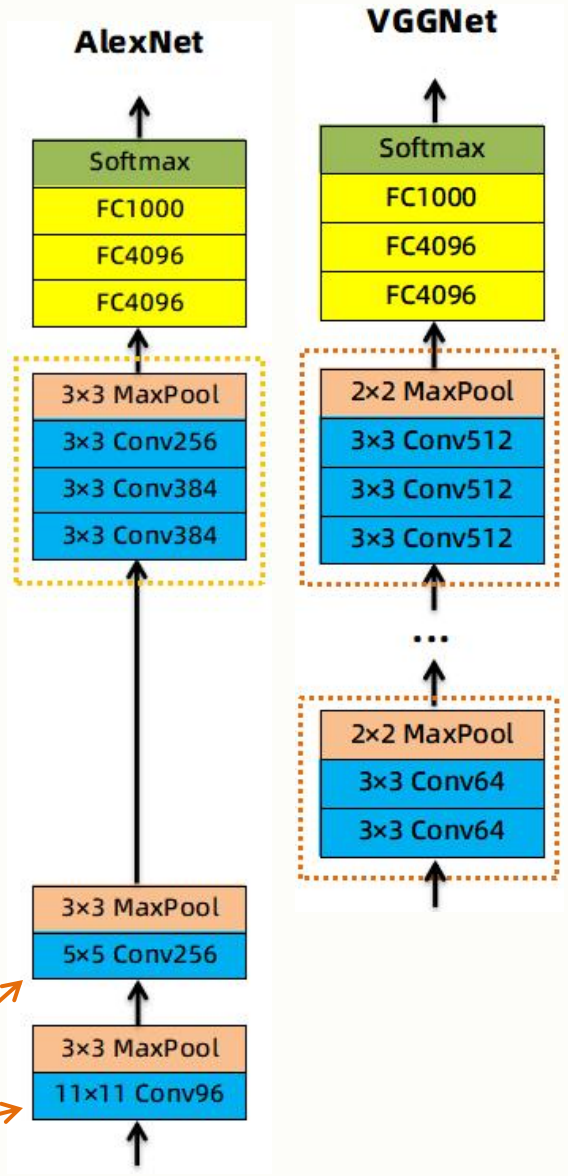
Q：是否还能通过继续加深模型，实现精度的进一步提升？

A1：堆叠更多的全连接层 → 太昂贵

A2：堆叠更多的卷积层 → 超参数难以确定

A3：精心的网络设计方法 → 设计难度较大 → GoogLeNet

A4：模块化地进行卷积层组合 → 简单、使用 → VGG



VGG的特点

小核心也能有大视野

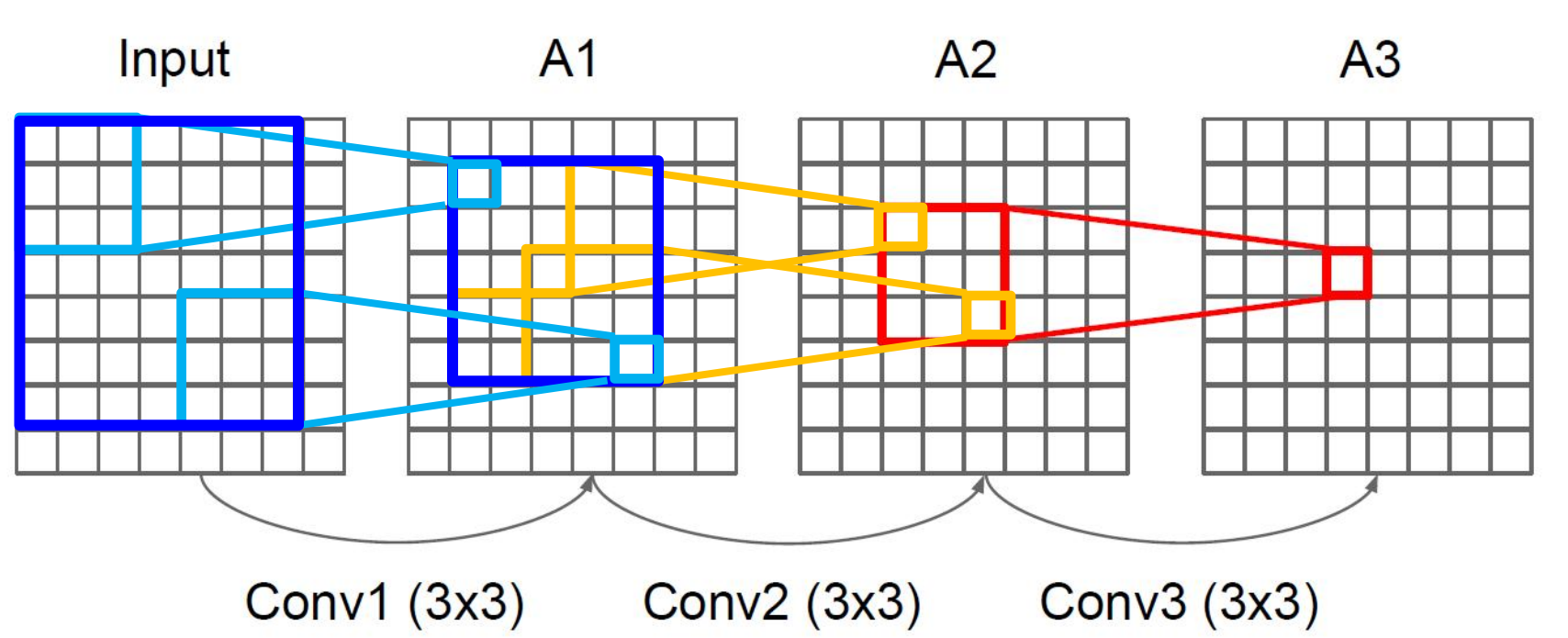
- 全部采用3×3的卷积核
  - 增加了更多的非线性，提高了网络的拟合/表达能力
  - 减少了参数的数量
- 采用堆叠卷积层的方法构建卷积层组，来保证较大的感受野（receptive filed）
- 全部使用2×2的最大池化核

Question：3×3 的小卷积核是否能和原来的7×7的卷积核具有相同有效的感知域?

VGG的特点

小核心也能有大视野

Q: 3×3的小卷积核是否能和7×7的卷积核具有相同有效的感知域?



- 两个 3×3的卷积层有和1个5×5的卷积层相同的感受野
- 三个 3×3的卷积层有和1个7×7的卷积核相同的感受野
- 四个 3×3的卷积层有和1个9×9的卷积核相同的感受野

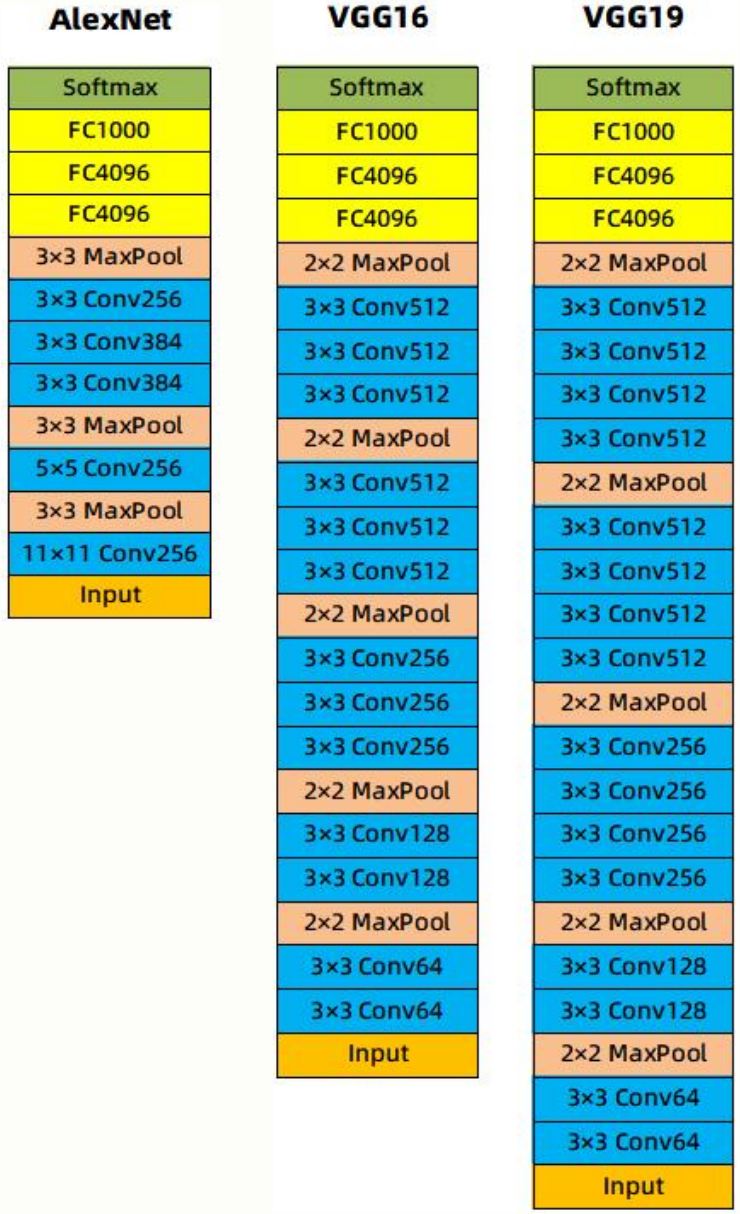
VGG16	VGG19
Softmax	Softmax
FC1000	FC1000
FC4096	FC4096
FC4096	FC4096
2×2 MaxPool	2×2 MaxPool
3×3 Conv512	3×3 Conv512
3×3 Conv512	3×3 Conv512
3×3 Conv512	3×3 Conv512
2×2 MaxPool	3×3 Conv512
3×3 Conv512	2×2 MaxPool
3×3 Conv512	3×3 Conv512
3×3 Conv512	3×3 Conv512
2×2 MaxPool	3×3 Conv512
3×3 Conv256	3×3 Conv512
3×3 Conv256	2×2 MaxPool
3×3 Conv256	3×3 Conv256
2×2 MaxPool	3×3 Conv256
3×3 Conv128	3×3 Conv256
3×3 Conv128	3×3 Conv256
2×2 MaxPool	2×2 MaxPool
3×3 Conv64	3×3 Conv128
3×3 Conv64	3×3 Conv128
Input	2×2 MaxPool
	3×3 Conv64
	3×3 Conv64
	Input

VGG的特点

更多的通道、更强的特征、更少的参数

堆叠小卷积核可以实现大卷积核相同的感受野，但是堆叠小卷积核到底有什么好处呢?

- 更强的非线性：1 to 3 -> 更深的网络
- 更少的参数: $3*(3^2C)=27C$ vs. $7^2C=49C$, C通道数
- 更多的通道数：更多的特征
- 更小的卷积核：更大的特征图，更多的信息

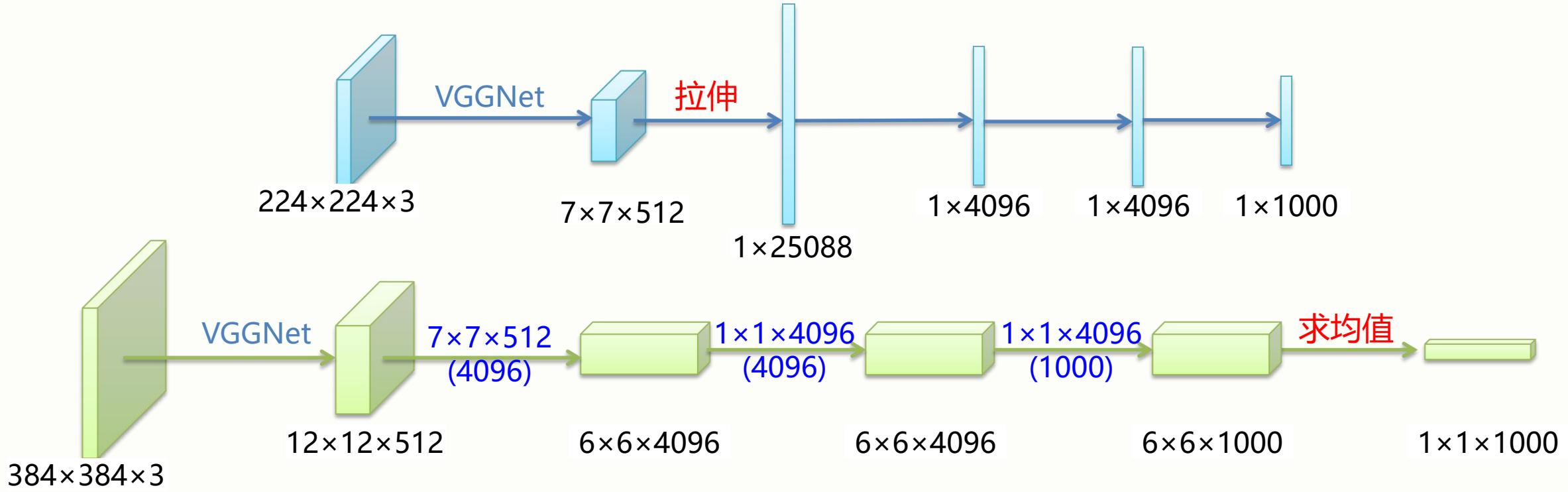


VGG的特点

支持任意尺度的测试

无论是Alexnet还是VGGNet都有一个问题，即无法实现任意尺度的训练/推理。

- 卷积层和全连接层的运算都是矩阵点乘，因此运算规则上没有发生变化
- 非标准尺度的输出是一个Scoremap，而不是一个值。需要对其求平均才能获得one-hot 标签向量



VGG的特点

简单就是王道

- 7层的AlexNet
- 11×11，5×5，3个3×3的卷积核
- 16.4% top 5 错误率 in ILSVRC12

与Alexnet一样，VGG同样采取输入层—卷积层—全连接层—输出层的结构。形式上包含5个卷积层组，3个全连接层和1个Softmax输出层，每个卷积层组后面紧跟一个Max-Pooling最大池化层，所有隐层的激活函数都是ReLU。

- 16~19层VGGNet
- 5组3×3的卷积核
- top 5 错误率 in ILSVRC14

VGG架构的详细分析

VGG架构的详细分析

VGGNet的6种网络体系结构

论文中总共设计了6种不同的网络结构，都是由**5个**卷积层组，**3个**全连接层组成。

- 所有的卷积核都是3×3卷积，且步长、填充都为1；
- 每一组卷积核内的通道数都是相同的；
- 每个卷积核组后紧跟一个maxpooling层；
- 从A到E卷积层组内堆叠卷积数量逐渐增多，从11到19，其中D为著名的**VGG16**，E为著名的**VGG19**。

VGG16 VGG19

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Conv3_1
Conv3_2
Conv3_3
Conv3_4

VGG架构的详细分析

INPUT: [224x224x3] memory: 224*224*3=150K params: 0
CONV3-64: [224x224x64] memory: 224*224*64=3.2M params: (3*3*3)*64 = 1,728
CONV3-64: [224x224x64] memory: 224*224*64=3.2M params: (3*3*64)*64 = 36,864
POOL2: [112x112x64] memory: 112*112*64=800K params: 0
CONV3-128: [112x112x128] memory: 112*112*128=1.6M params: (3*3*64)*128 = 73,728
CONV3-128: [112x112x128] memory: 112*112*128=1.6M params: (3*3*128)*128 = 147,456
POOL2: [56x56x128] memory: 56*56*128=400K params: 0
CONV3-256: [56x56x256] memory: 56*56*256=800K params: (3*3*128)*256 = 294,912
CONV3-256: [56x56x256] memory: 56*56*256=800K params: (3*3*256)*256 = 589,824
CONV3-256: [56x56x256] memory: 56*56*256=800K params: (3*3*256)*256 = 589,824
POOL2: [28x28x256] memory: 28*28*256=200K params: 0
CONV3-512: [28x28x512] memory: 28*28*512=400K params: (3*3*256)*512 = 1,179,648
CONV3-512: [28x28x512] memory: 28*28*512=400K params: (3*3*512)*512 = 2,359,296
CONV3-512: [28x28x512] memory: 28*28*512=400K params: (3*3*512)*512 = 2,359,296
POOL2: [14x14x512] memory: 14*14*512=100K params: 0
CONV3-512: [14x14x512] memory: 14*14*512=100K params: (3*3*512)*512 = 2,359,296
CONV3-512: [14x14x512] memory: 14*14*512=100K params: (3*3*512)*512 = 2,359,296
CONV3-512: [14x14x512] memory: 14*14*512=100K params: (3*3*512)*512 = 2,359,296
POOL2: [7x7x512] memory: 7*7*512=25K params: 0
FC: [1x1x4096] memory: 4096 params: 7*7*512*4096 = 102,760,448
FC: [1x1x4096] memory: 4096 params: 4096*4096 = 16,777,216
FC: [1x1x1000] memory: 1000 params: 4096*1000 = 4,096,000

总存储: 24M * 4 bytes ~= 96MB / image (only forward! ~*2 for bwd)
总参数: 138M parameters

大多数内存消耗在低层的卷积层

大多数参数在高层的FC层

VGG架构的详细分析

不同结构的性能对比

ConvNet config. (Table 1)	smallest image side		top-1 val. error (%)	top-5 val. error (%)
	train (S)	test (Q)		
A	256	256	29.6	10.4
A-LRN	256	256	29.7	10.5
B	256	256	28.7	9.9
C	256	256	28.1	9.4
	384	384	28.1	9.3
	[256;512]	384	27.3	8.8
D	256	256	27.0	8.8
	384	384	26.8	8.7
	[256;512]	384	25.6	8.1
E	256	256	27.3	9.0
	384	384	26.9	8.7
	[256;512]	384	**25.5**	**8.0**

1. LRN层无性能增益（A-LRN）

VGG作者通过网络A-LRN发现，AlexNet曾用到的LRN层并没有带来性能的提升，因此在其它组的网络中均没再出现LRN层。

2. 多尺度训练可以降低错误率（C、D、E）

3. 随着深度增加，分类性能逐渐提高（A、B、C、D、E）

从11层的A到19层的E，网络深度增加对top1和top5的错误率下降很明显。

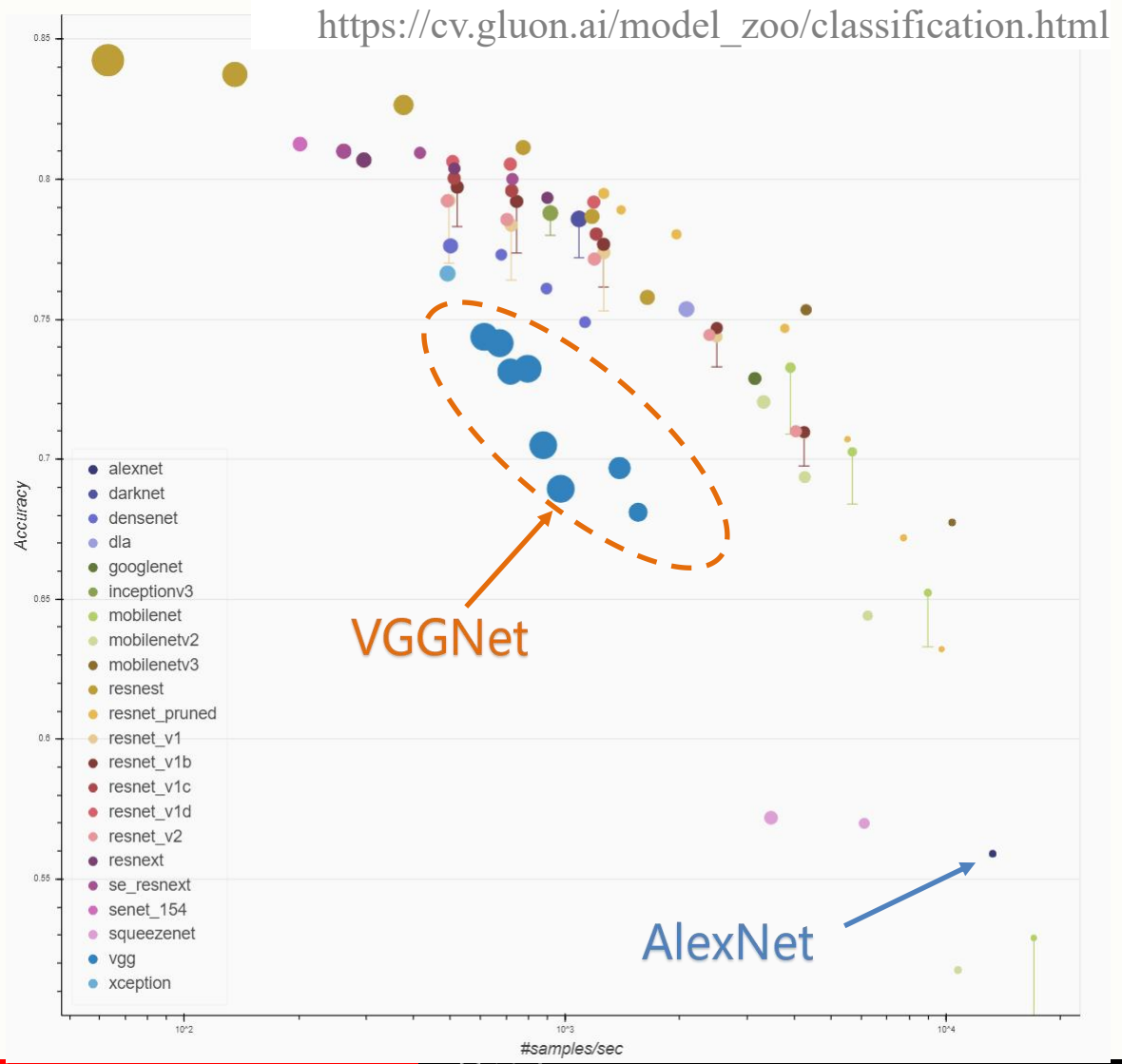
4. 多个小卷积核比单个大卷积核性能好（B）

作者用B和自己一个不在实验组里的较浅网络比较，较浅网络用conv5x5来代替B的两个conv3x3，结果显示多个小卷积核比单个大卷积核效果要好。

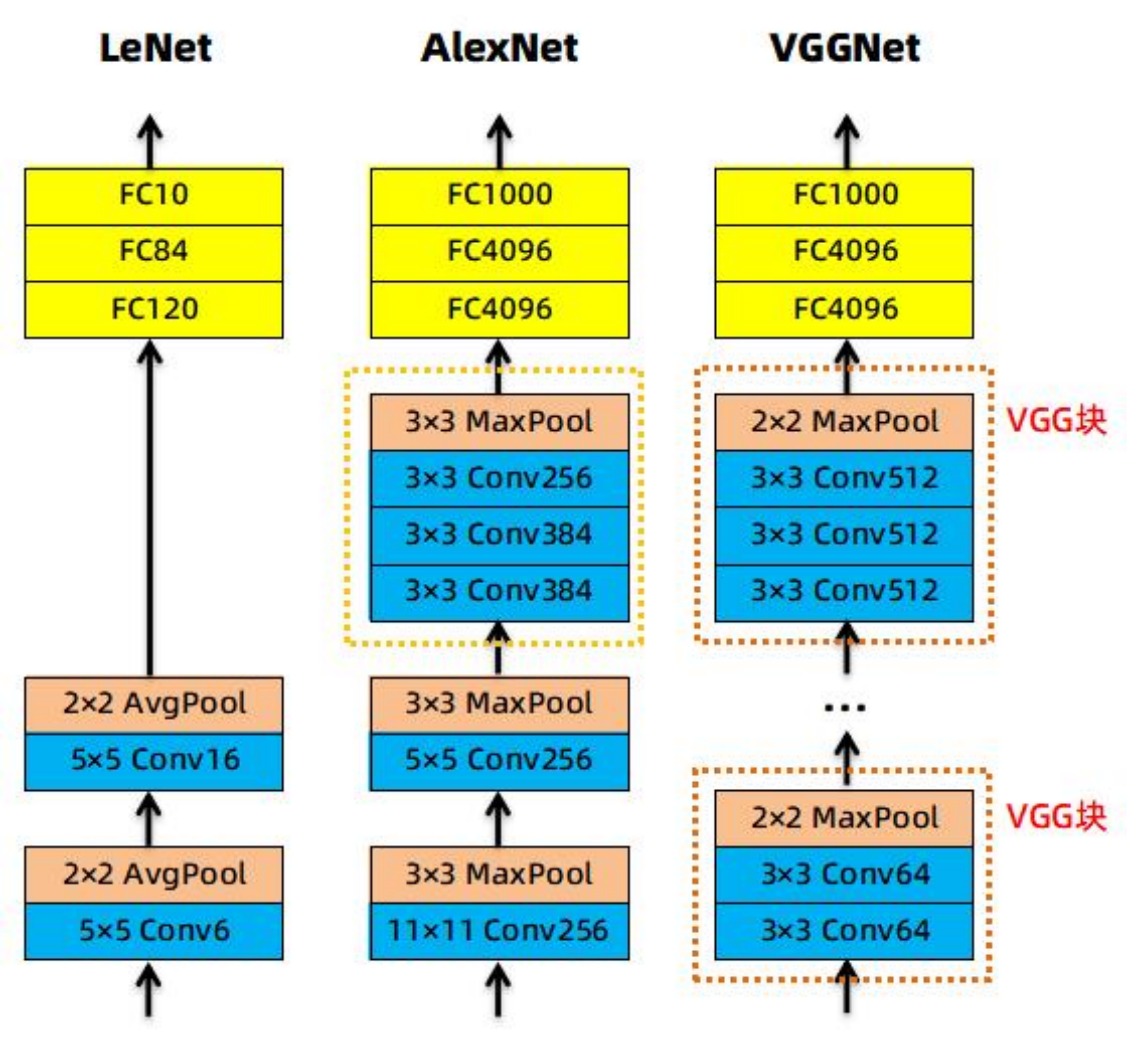
VGGNet架构的详细分析

Visualization of Inference on GluonCV

- AlexNet
  - 精度：0.559
  - 内存：202Mb
  - 速度：13270 张/秒
- VGG16
  - 精度：0.732
  - 内存：3422Mb
  - 速度：797 张/秒



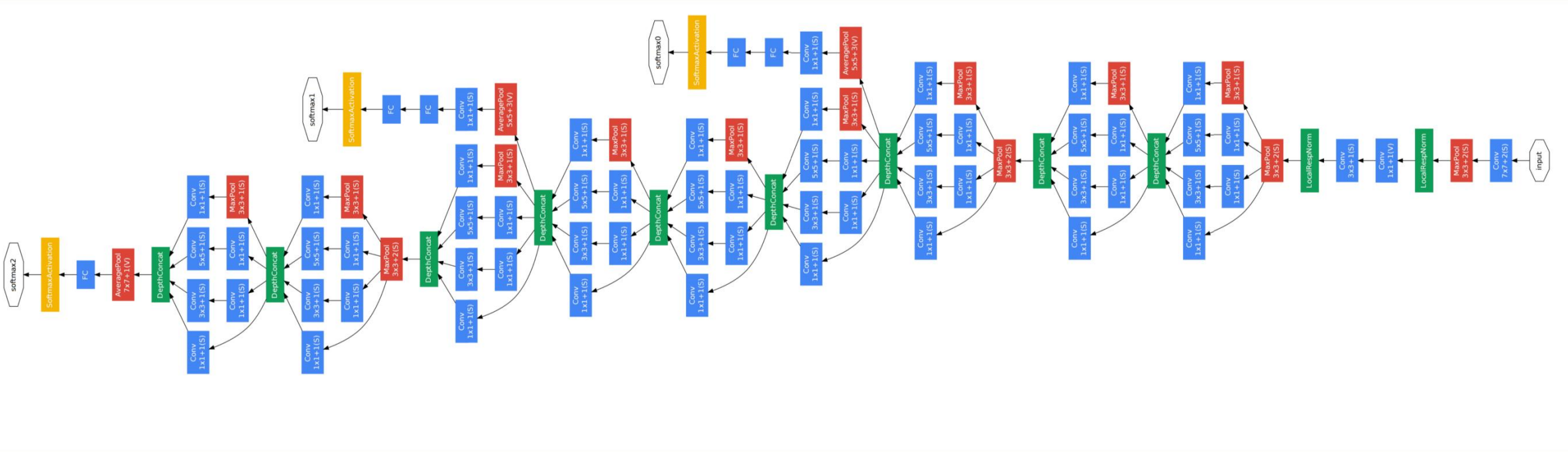
模块化的网络VGGNet



小　结

- VGG使用模块化的设计思想，通过重复堆叠相同的卷积块来构建深度神经网络；
- 不同数量卷积块的组合可以得到不同复杂度的网络变种；
- 深度的增加能够有效提高性能；
- 通过堆叠，小卷积核能够具有和大卷积核相同的感受野，并增加非线性特性；
- 使用卷积替代全连接层，可适应不同尺度的样本；
- 性价比最佳模型：VGG16，从头到尾只有3x3卷积与2x2池化，简洁优美，性能优异。

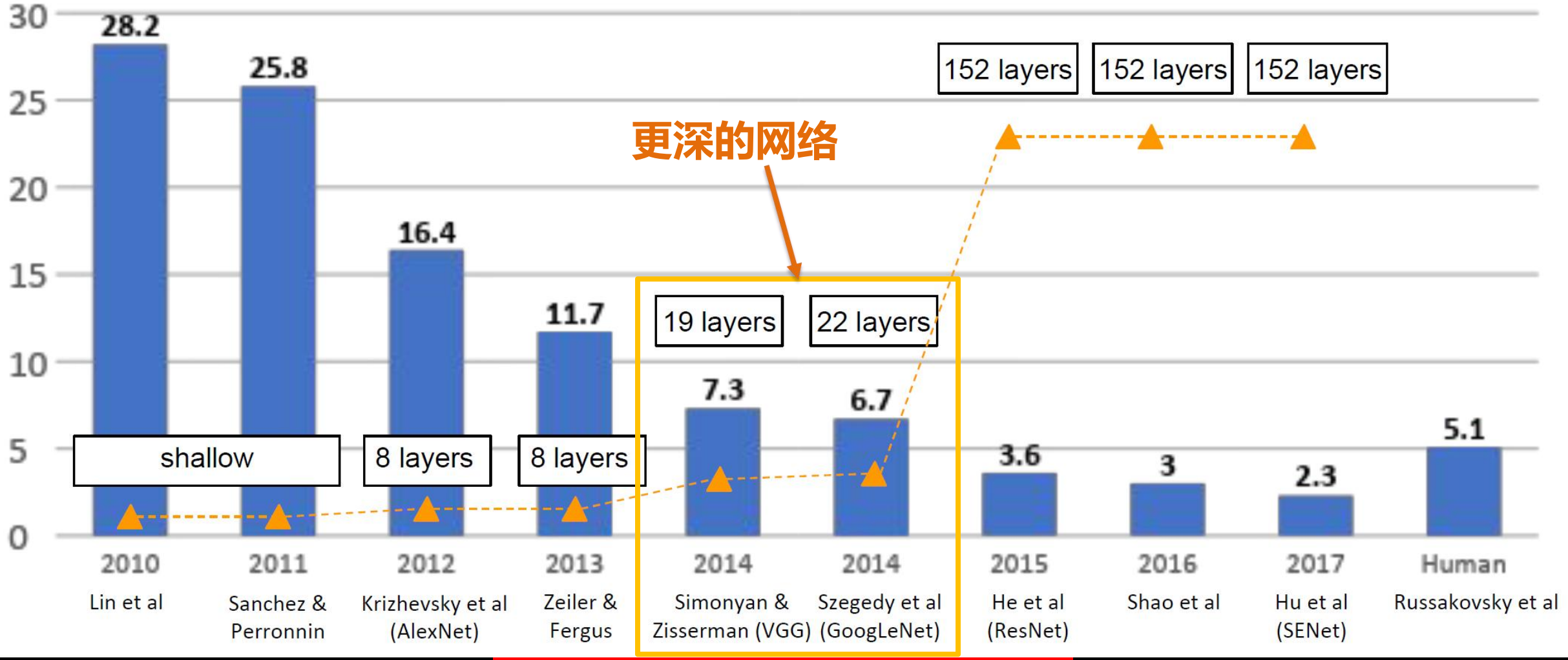
多分支网络（GoogLeNet）



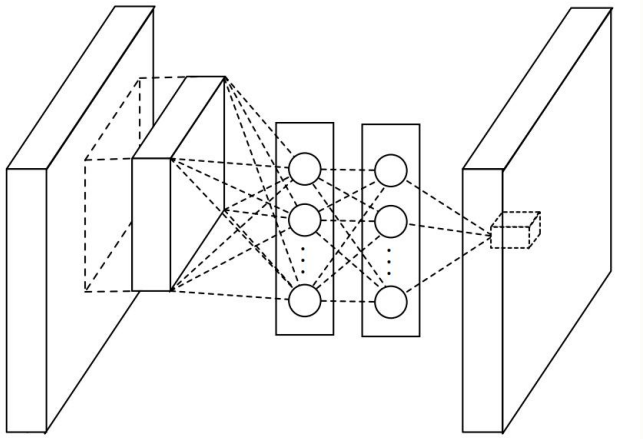
Christian Szegedy, Wei Liu, et.al. Going deeper with convolutions. *CVPR2014.*

多分支网络（GoogLeNet）

ImageNet 大规模视觉识别挑战赛(ILSVRC)



网络中的网络
Network In Network



Min Lin, Qiang Chen, Shuicheng Yan. Network in Network. *arXiv1312.*

网络中的网络（Network In Network）

全连接层的问题

- ✓ 卷积层的参数：$C_i \times C_o \times k^2$
- ✓ 全连接层的参数：$C_i \times C_o \times f^2$

} 参数大幅减少

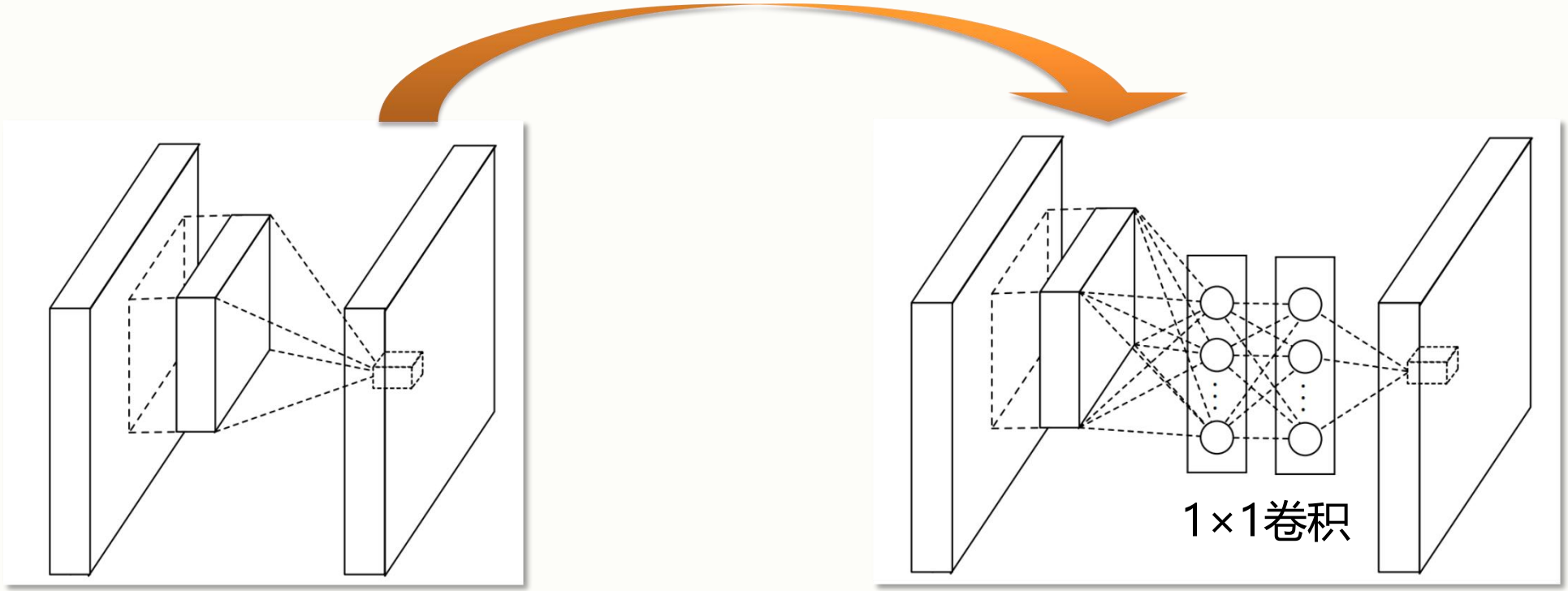
- 卷积层 与 全连接层 的连接部分？
  - LeNet：16×5×5×120 = 48k
  - AlexNet： 256×6×6×4096 = 37.7M
  - VGG：512×7×7×4096 = 102.7M
- 剩下的全连接层呢？
  - LeNet：120×84 = 10K
  - AlexNet： 4096×4096 = 16M
  - VGG：4096×4096 = 16M

大多数参数来源于全连接层以及全连接层和卷积层的边界

网络中的网络（Network In Network）

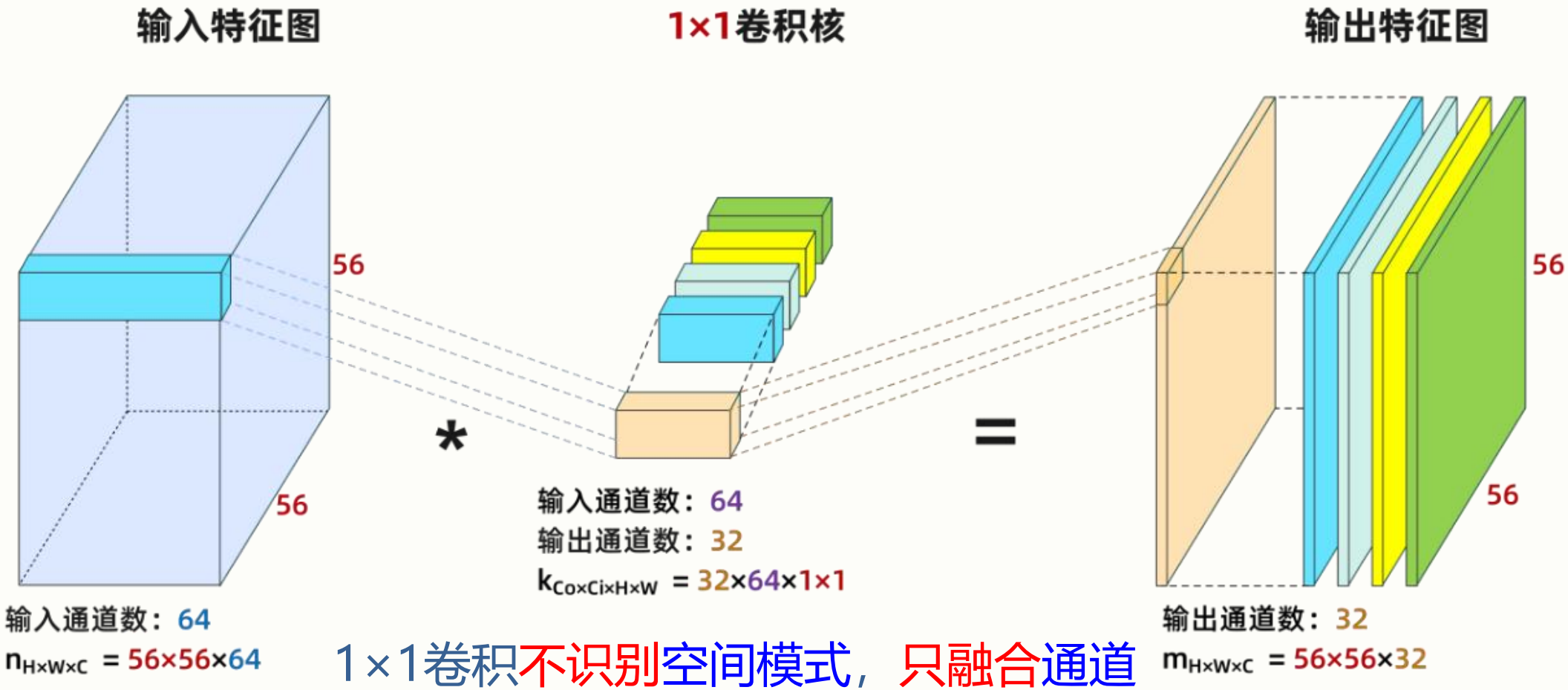
NiN模块

- **NiN模块** = 1个标准卷积 + 2个1×1卷积
  - **1×1卷积**：步长为1，无填充，不改变尺度
  - 相当于增加了模型的非线性特性



特殊的卷积层：1×1卷积

1×1卷积能够获得更完美的感知



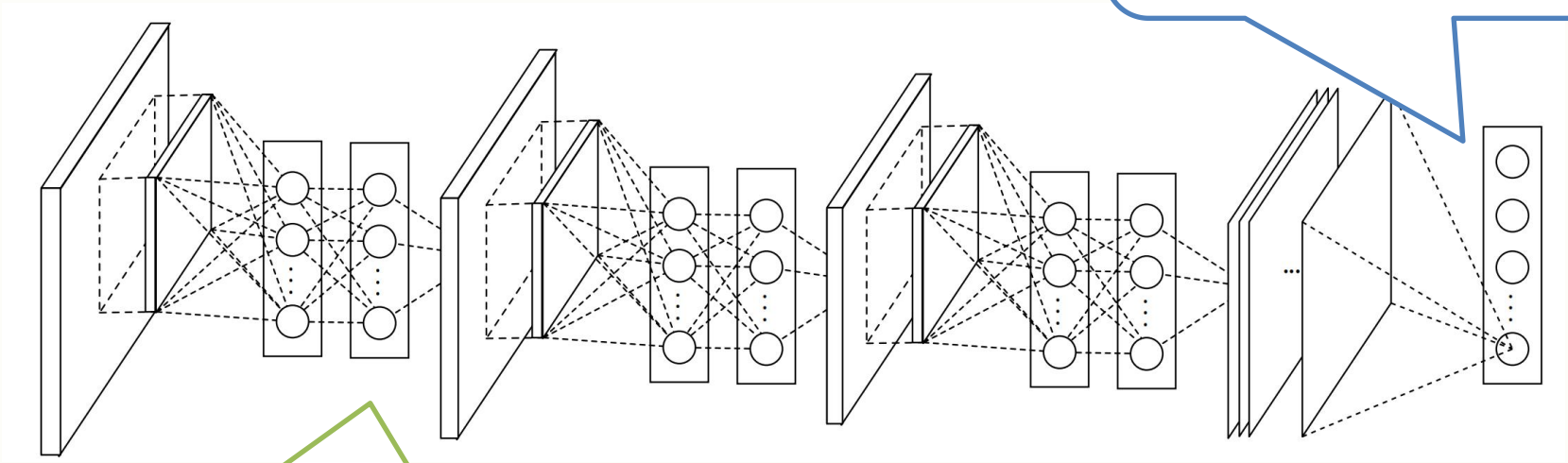
1×1卷积不识别空间模式，只融合通道

1×1卷积相当于输入形状为$n_h \times n_w \times c_i$，权重为$c_i \times c_o$的全连接层

网络中的网络（Network In Network）

NiN的体系结构

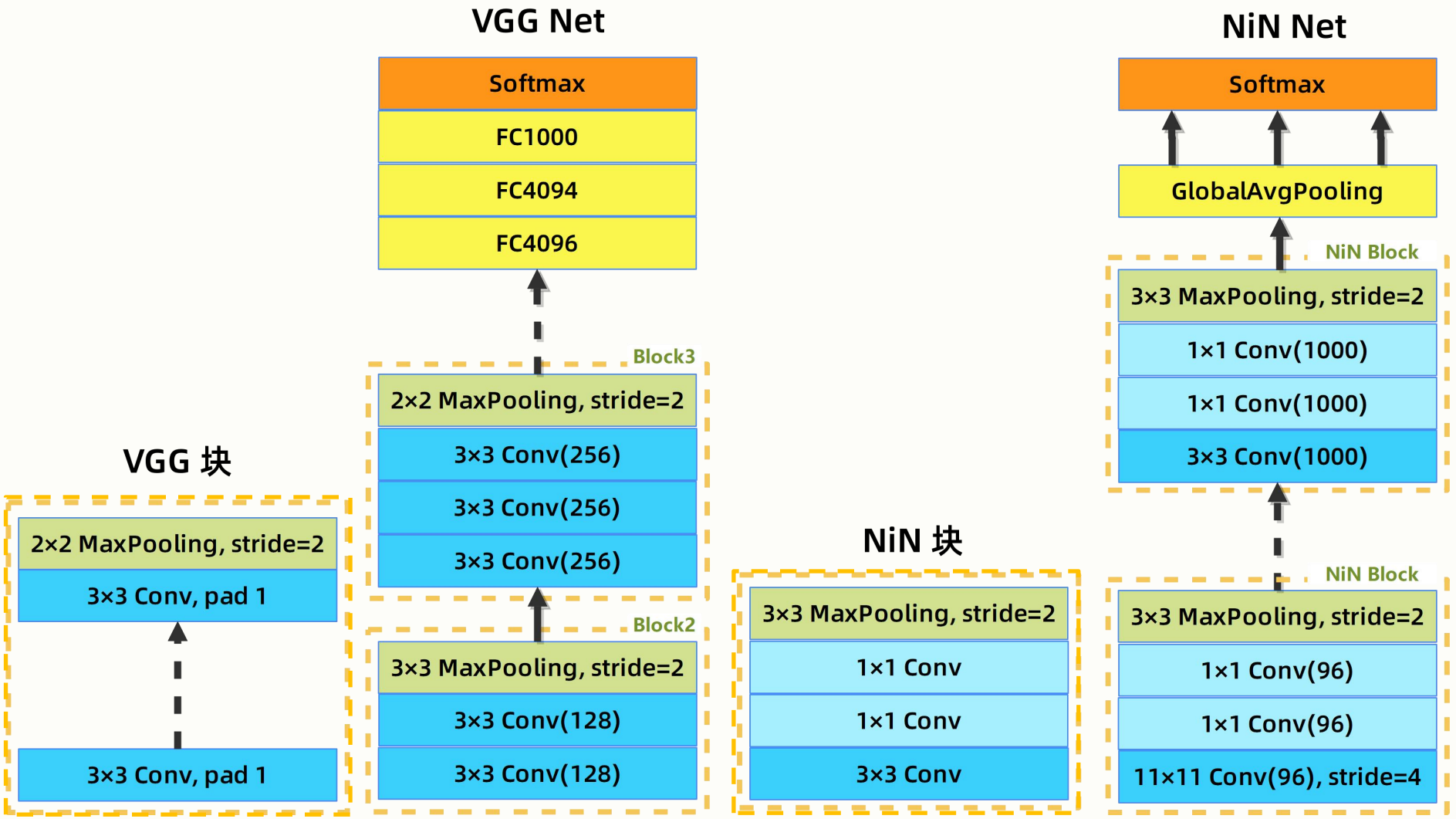
- ✓ 无全连接层
- ✓ 使用全局平均池化直接获得输出，通道数即类别数



- ✓ 交替使用NiN块和步长为2的最大池化层
- ✓ 特征图尺度逐渐缩小
- ✓ 通道数逐渐增加

网络中的网络（Network In Network）

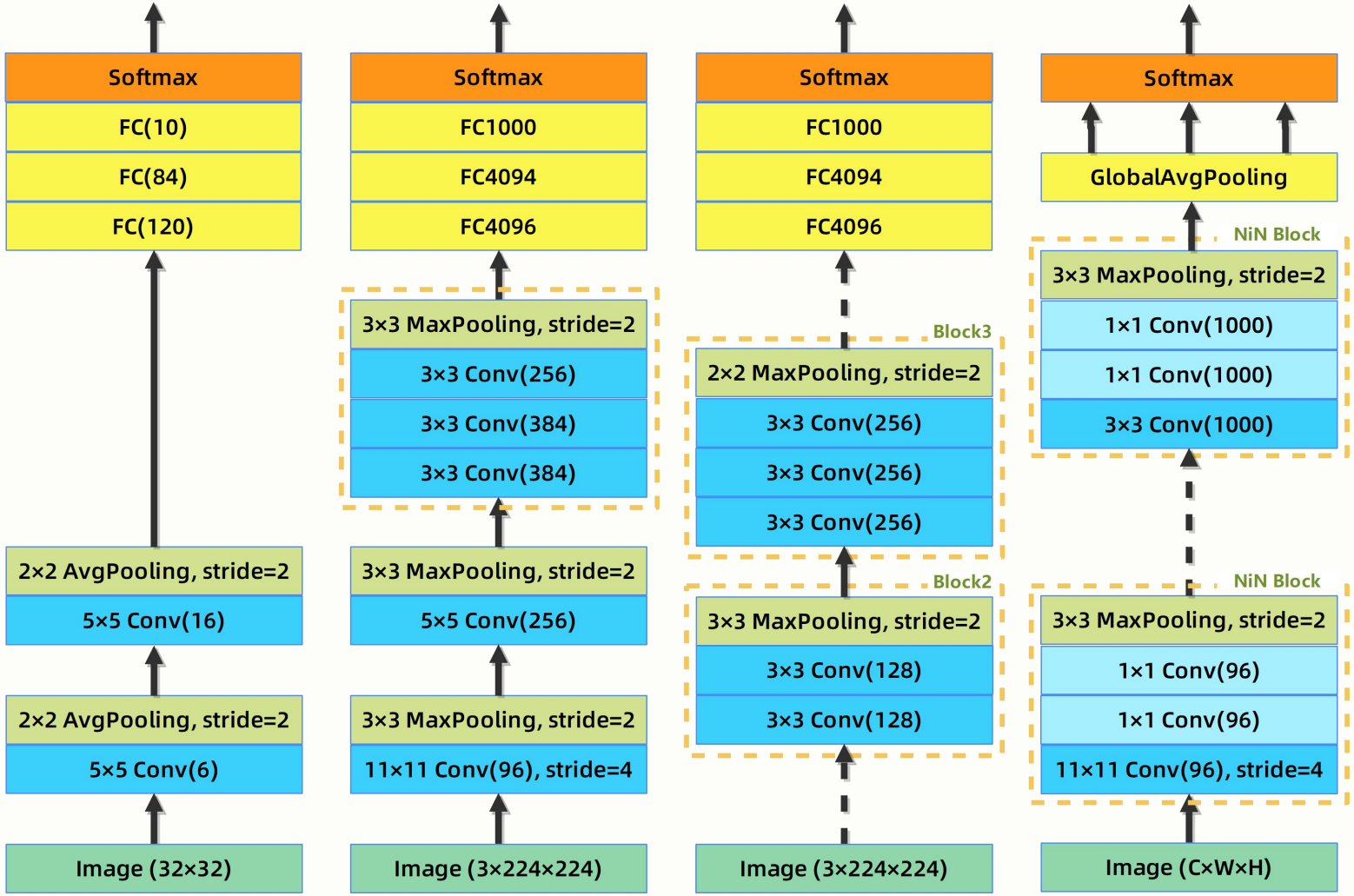
NiN的体系结构



Inception模块

Inception模块

什么样的卷积层超参数更好?



1×1　3×3

5×5　7×7

AvgPooling

MaxPooling

并行架构

串行架构

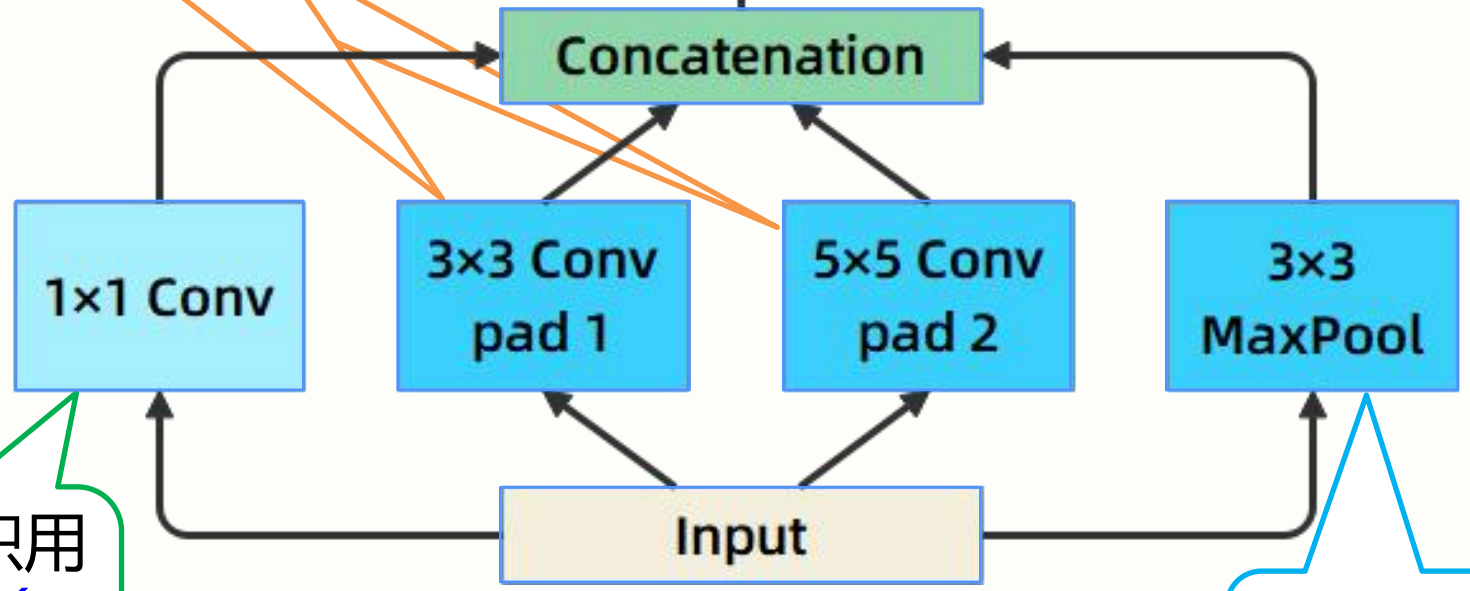
图架构

Inception模块

我不需要选择，我要兼容并蓄

不同的卷积窗口提取不同维度的特征

✓按通道将不同通道的输出进行连接
✓输出与输入的高宽尺度相同



基本Inception模块

1×1卷积用来融合多通道特征

最大池化输出局部最强响应

所贵乎君子者以能兼容并蓄
使才智者有以自见
而愚不肖者有以自全。
——明.方孝孺《复郑好义书》

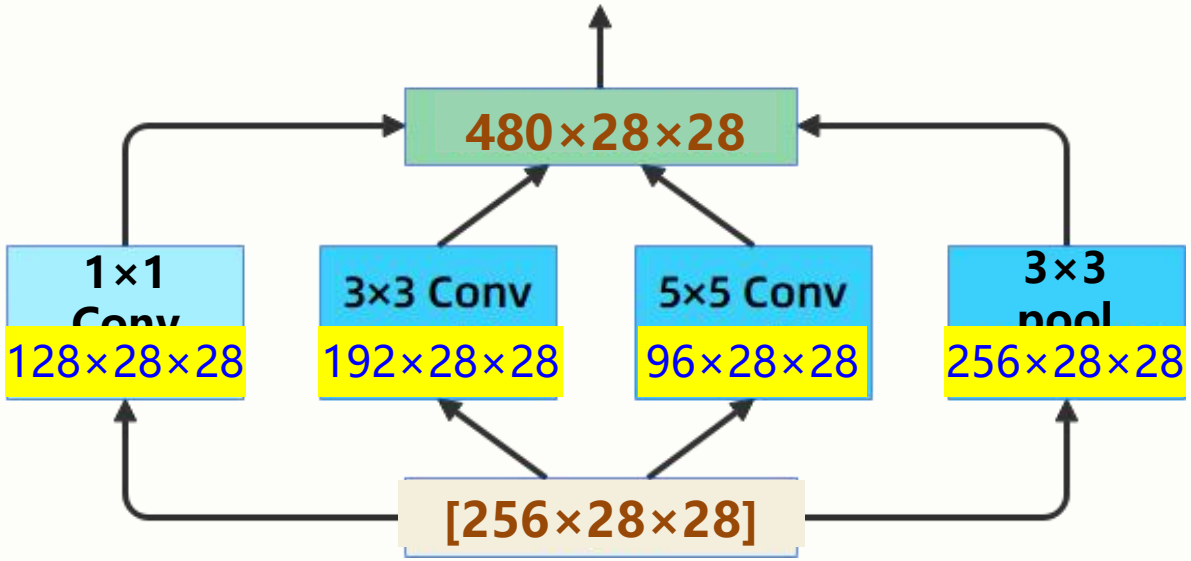
Inception模块

基本Inception模块

Q1: 不同模块的通道数是多少?

Q2: 多路输出拼接后的通道数是多少?

28×28×(128+192+96+256)=28×28×**672**



基本Inception模块

Q: 这个模块是否存在什么问题?

[Hint: 计算复杂性]

Inception中Conv的计算量:

- [1x1 conv, 128] 128×28x28x**256x1x1 = 25.69M**
- [3x3 conv, 192] 192x28x28**x256x3x3 = 346.82M**
- [5x5 conv, 96] 96x28x28**x256x5x5 = 481.69M**

Total:

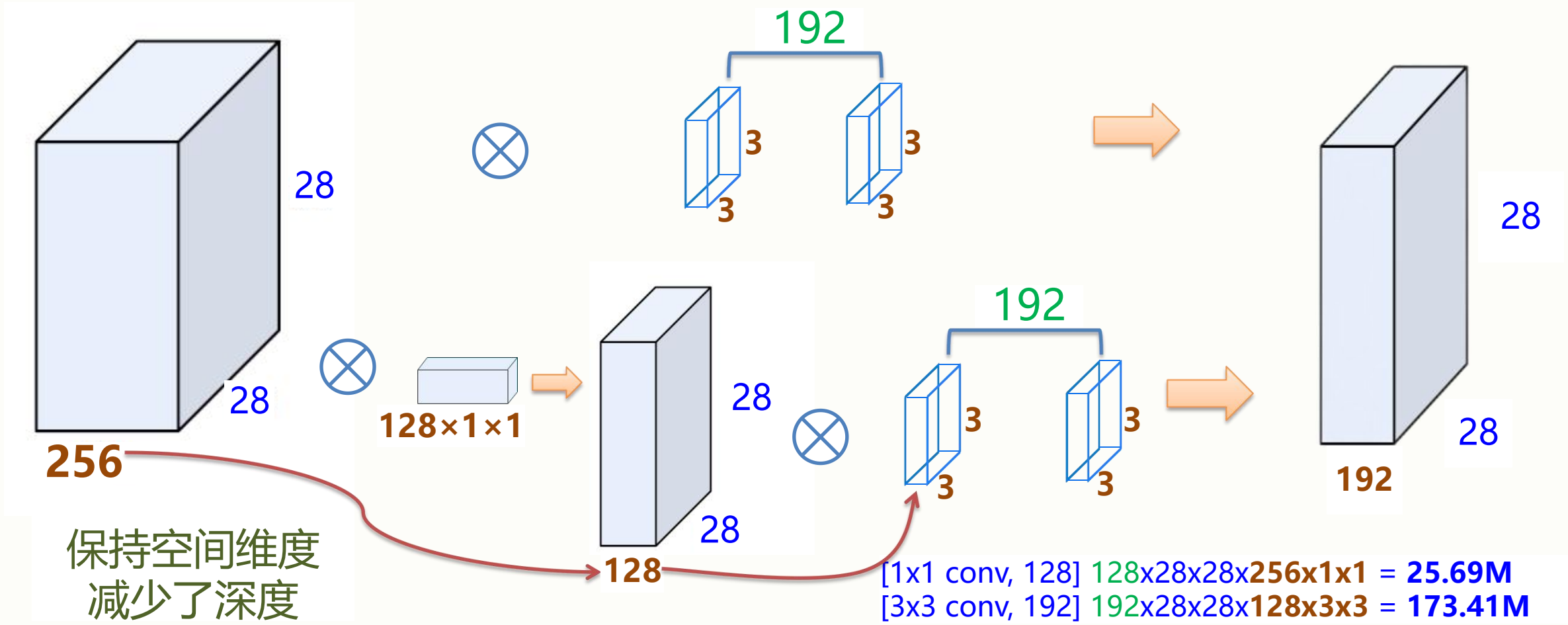
- **25.69+346.82+481.69 = 854.2M**

非常昂贵的计算代价！！！

Inception模块

基于1×1卷积的瓶颈设计

[3x3 conv, 192] 192x28x28x256x3x3 = 346.82M

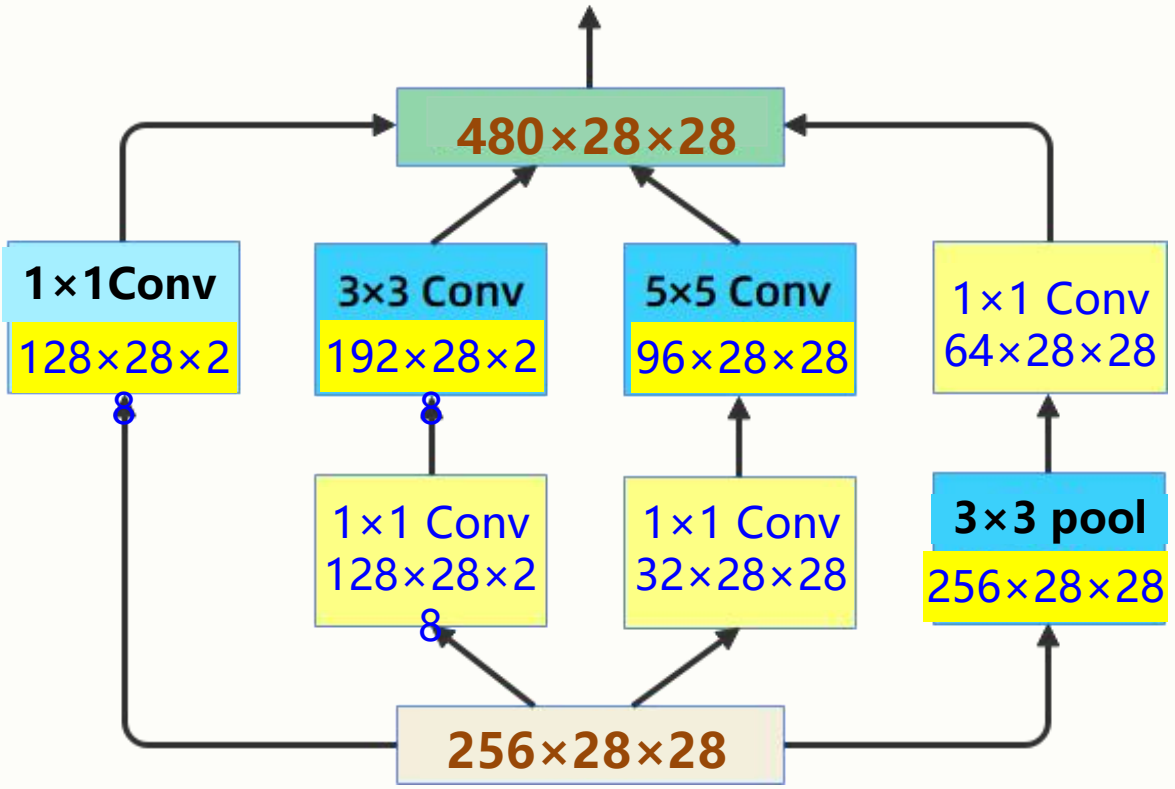


[1x1 conv, 128] 128x28x28x256x1x1 = 25.69M
[3x3 conv, 192] 192x28x28x128x3x3 = 173.41M

Inception模块

使用瓶颈网络之后的Inception1×1卷积

◆ 使用基本Inception模块，但增加“1×1卷积” BottleNeck瓶颈设计。



Inception中Conv的计算量:

- [1x1 conv, 128] 128x28x28x**256x1x1** = **25.69M**
- [1x1 conv, 128] 128x28x28x**256x1x1** = **25.69M**
- [1x1 conv, 32] 32x28x28x**256x1x1** = **6.42M**
- [3x3 conv, 192] 192x28x28x**128x3x3** = **173.41M**
- [5x5 conv, 96] 96x28x28x**32x5x5** = **60.21M**
- [1x1 conv, 64] 64x28x28x**256x1x1** = **12.85M**

Total:

- **25.69+25.69+6.42+172.41+60.21+12.85=358M**

与需要854M计算量的基本Inception相比，

Bottleneck设计还能减少池化层的深度。

Inception模块

Inception模块的参数和计算复杂性

inception(3b)	参数数量	FLOPs
3×3卷积	1.11M	867M
5×5卷积	3.07M	2408M
Inception (基本)	1.06M	854M
Inception (Bottleneck)	0.39M	358M

参数数量 = $K_W \times K_H \times in\_channel \times out\_channel$

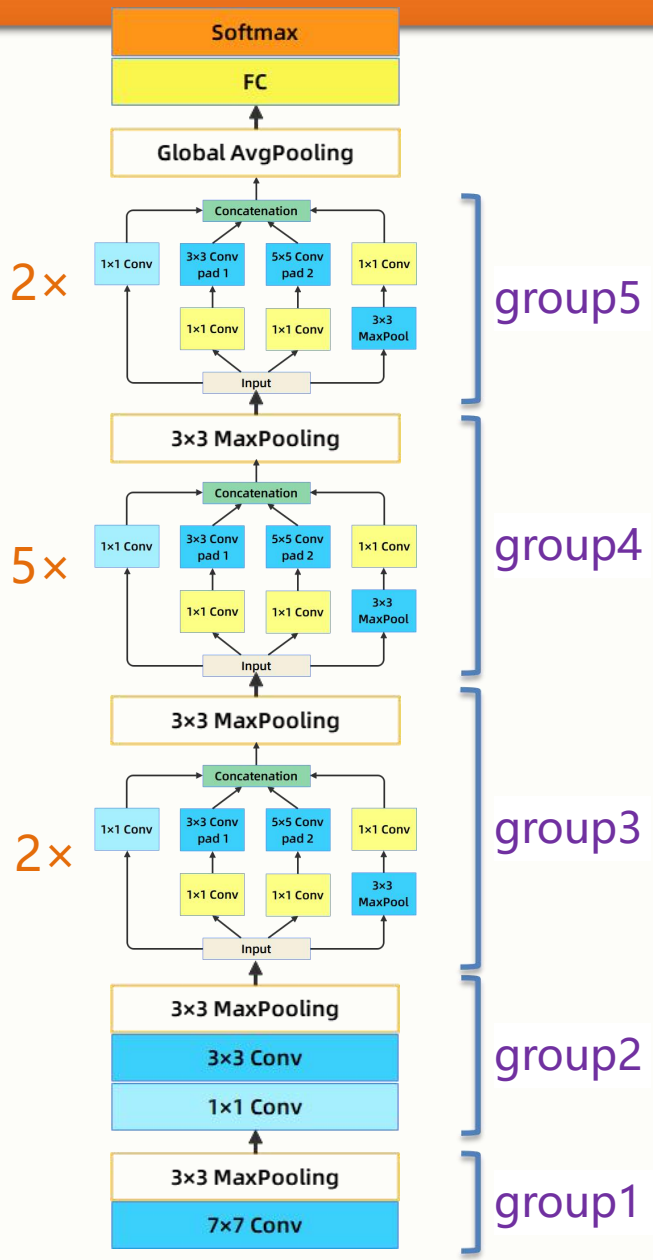
FLOPs = $(K_W \times K_H \times K_D) \times Kernel_{num} \times F_W \times F_H$

GoogLeNet体系结构详解

GoogLeNet体系结构详解

更深的网络，更高的计算效率

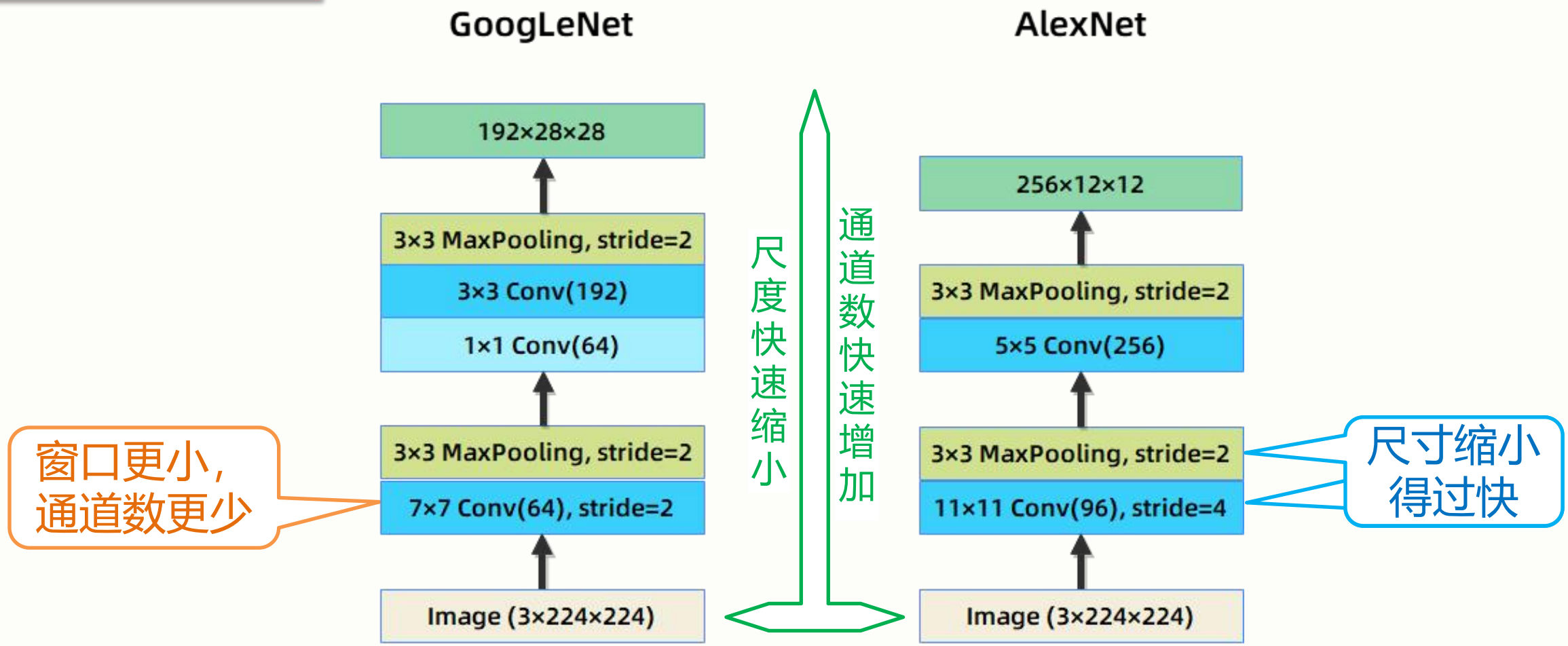
- Inception模块是一种精心设计的网络拓扑结构
- 将这些Inception模块按顺序进行堆叠，就可以构成GoogLeNet的主体结构
- GoogLeNet由5组，共9个Inception块组成，深度22层
- 堆叠具有降维功能的Inception模块可以构建计算量更低的GoogLeNet，只有 5百万的参数
  - ✓ 比AlexNet少12倍
  - ✓ 比VGG16少16倍



GoogLeNet体系结构详解

GoogLeNet的卷积层组

第1组和第2组

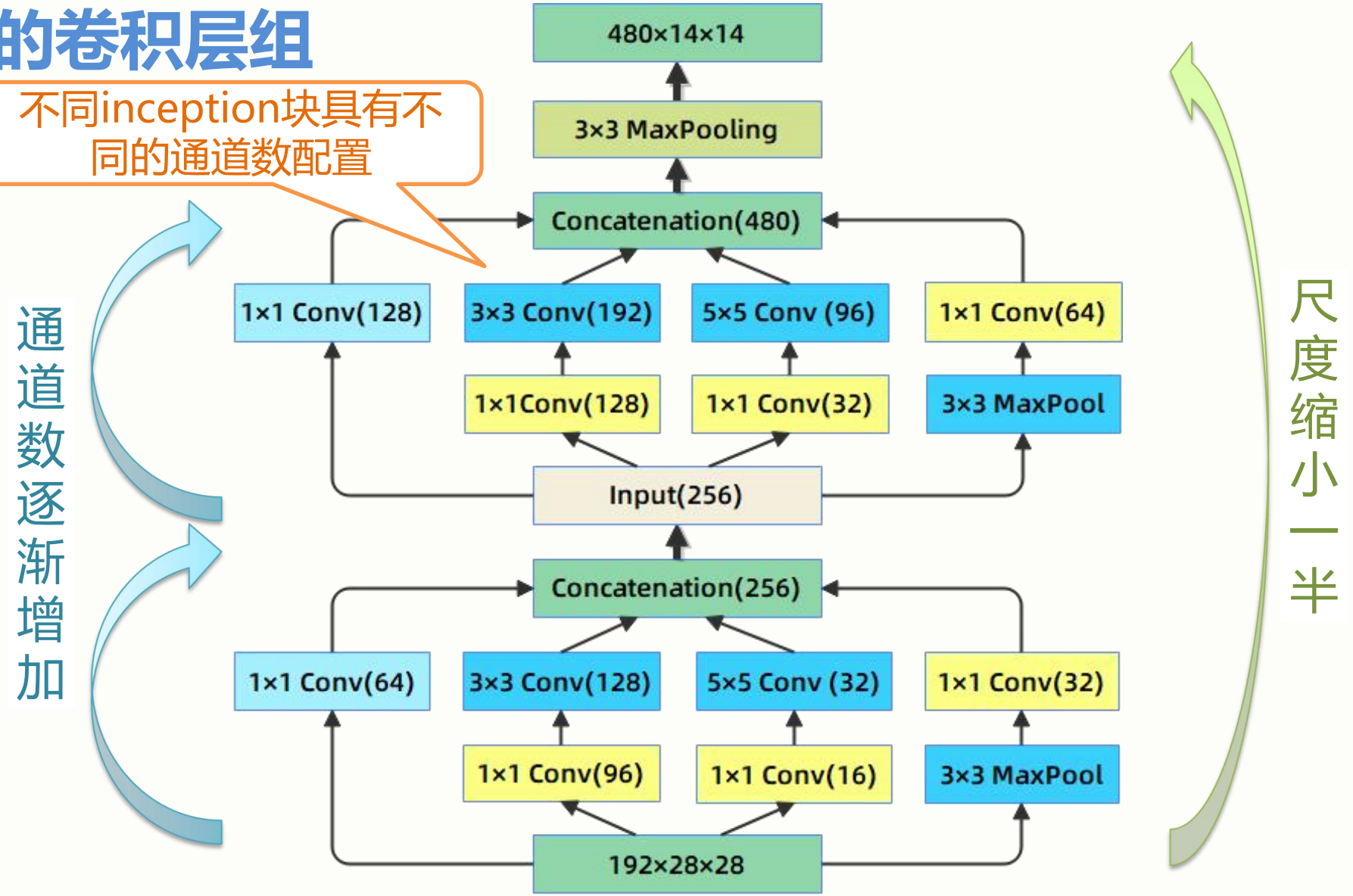


GoogLeNet体系结构详解

GoogLeNet的卷积层组

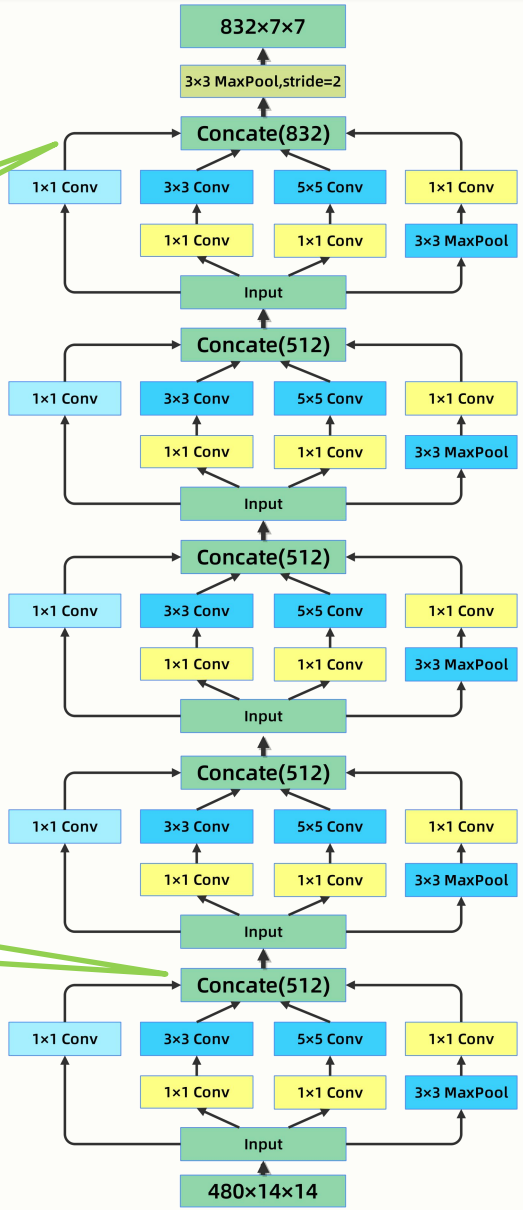
第3组

不同inception块具有不同的通道数配置



GoogLeNet体系结构详解

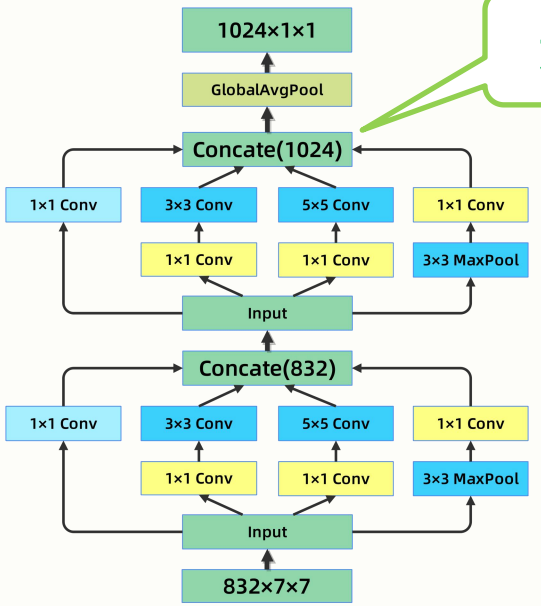
第4组和第5组



通道数增加

通道数增加

GoogLeNet的卷积层组



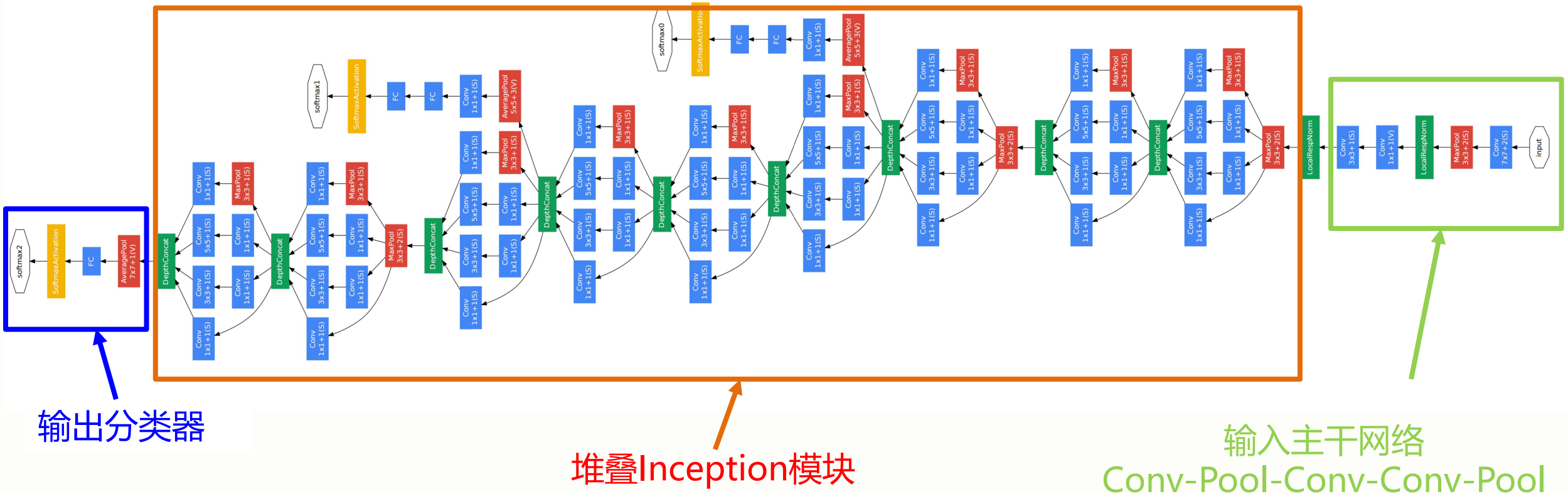
输出1024维的特征向量

通道数增加

GoogLeNet体系结构详解

GoogLeNet的整体设计

总共包含22个权重层（并行层只算一层，每个Inception算2层）



输出分类器

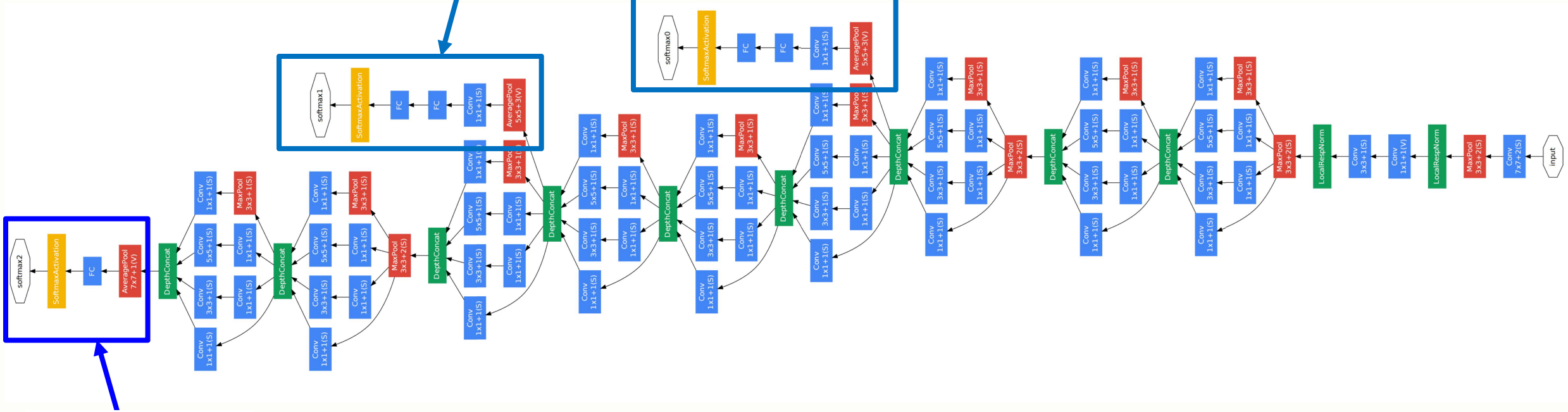
堆叠Inception模块

输入主干网络
Conv-Pool-Conv-Conv-Pool

GoogLeNet体系结构详解

GoogLeNet的整体设计

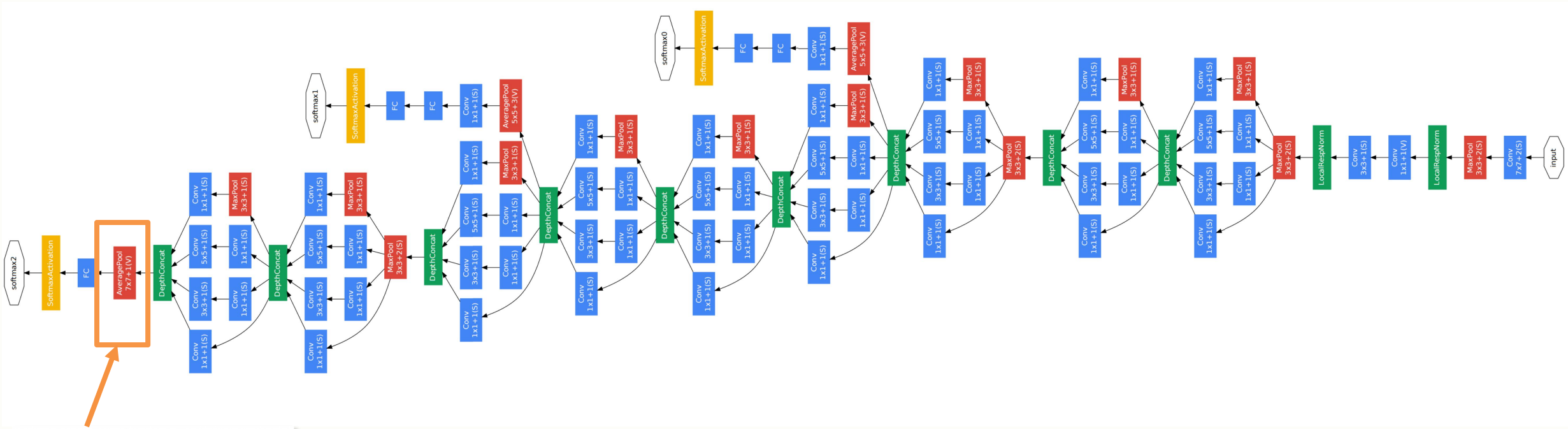
辅助分类器基于低层梯度输出额外的分类信息
(AvgPool - 1×1卷积 - FC - FC - Softmax)



输出分类器

GoogLeNet体系结构详解

GoogLeNet的整体设计



在最后一个卷积层之后，最终的FC层之前，使用一个全局平均池化层对每个Feature map进行空间平均，而不再使用昂贵的FC层

GoogLeNet的变种模型

GoogLeNet的变种模型

InceptionBN(V2)

使用Batch normalization对每个层的批数据进行正则化约束。

v2

Sergey Ioffe, Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv1502*.

InceptionV3

将5×5卷积替换为多个3×3卷积

将5×5卷积替换为1×7和7×1卷积

将3×3卷积替换为1×3和3×1卷积

更大的输入尺寸299×299，更深的深度

v3

Christian Szegedy, Vincent Vanhoucke, sergey Ioffe, etc.al. Rethinking the Inception Architecture for Computer Vision. *arXiv1512*.

InceptionV4

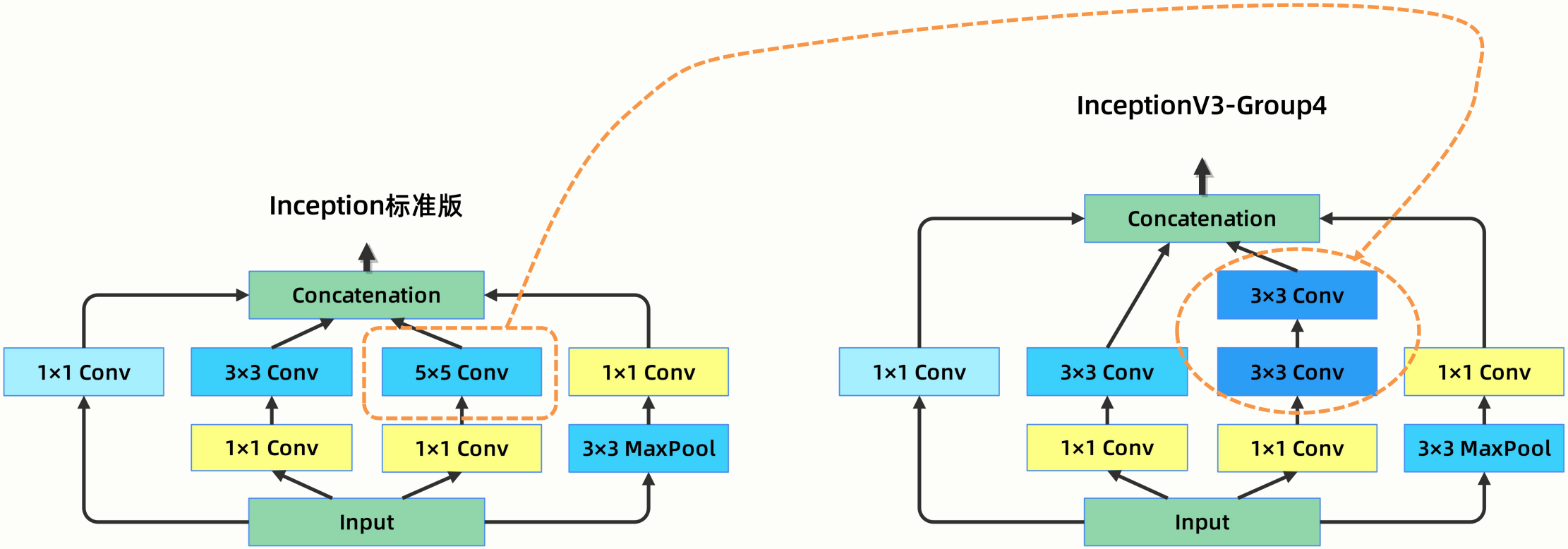
将残差连接融入inception，构建Inception-ResNet块

v4

Christian Szegedy, sergey Ioffe, Vincent Vanhoucke. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *arXiv1512*.

GoogLeNet的变种模型

Inception V3, 第3组

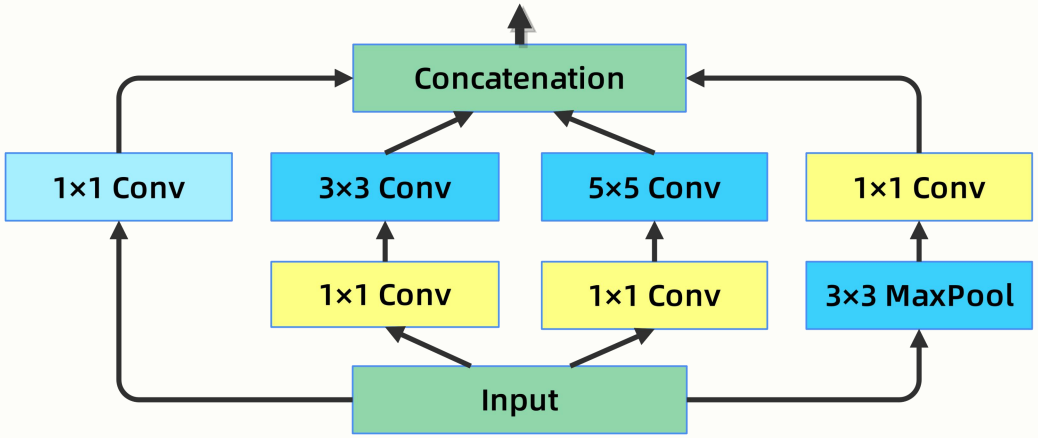


Christian Szegedy, Vincent Vanhoucke, sergey Ioffe, etc.al. Rethinking the Inception Architecture for Computer Vision. *arXiv1512.*

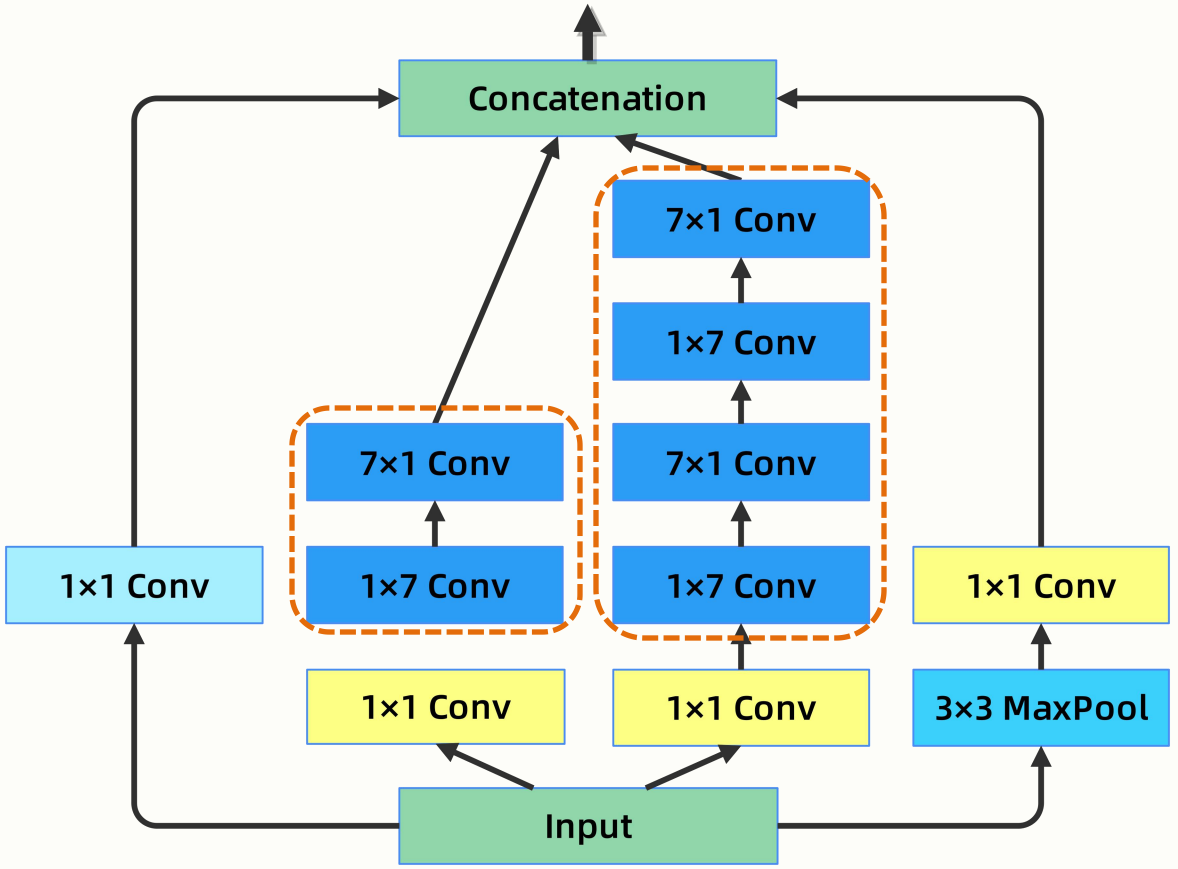
GoogLeNet的变种模型

Inception V3, 第4组

Inception标准版



InceptionV3-Group4



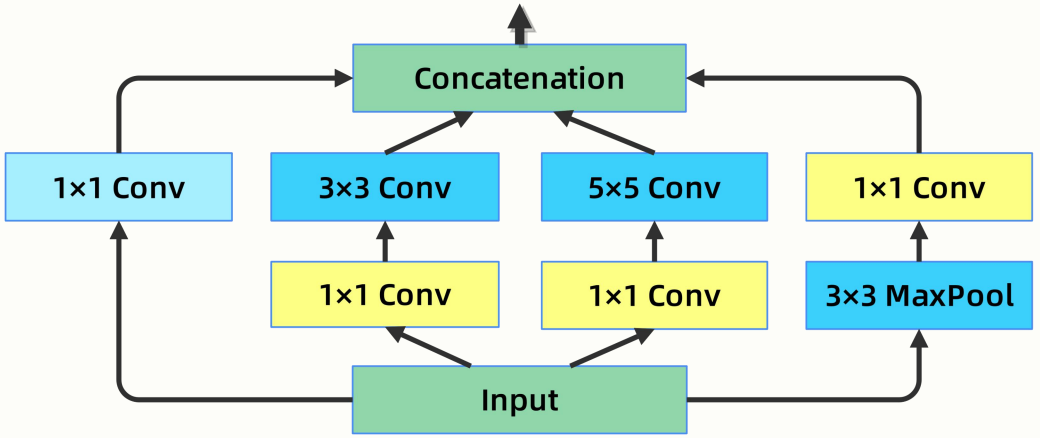
Christian Szegedy, Vincent Vanhoucke, sergey Ioffe, etc.al. Rethinking the Inception Architecture for Computer Vision. *arXiv1512.*

GoogLeNet的变种模型

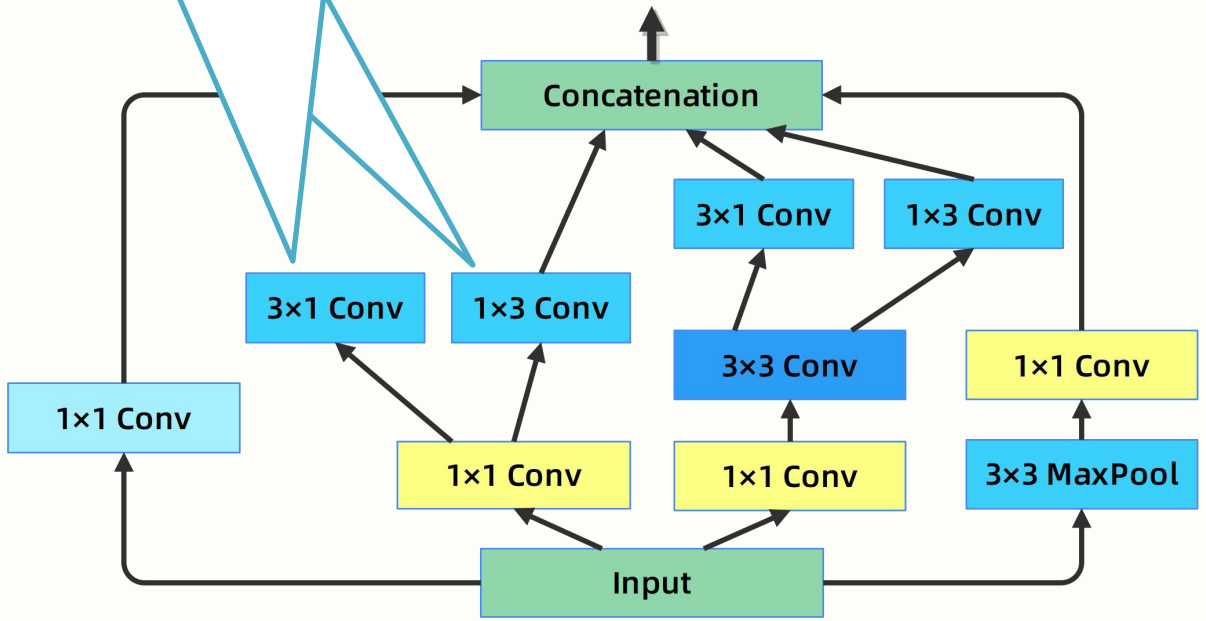
Inception V3, 第5组

宽度的增加有利于缓解
表征性瓶颈

Inception标准版



InceptionV3-Group5



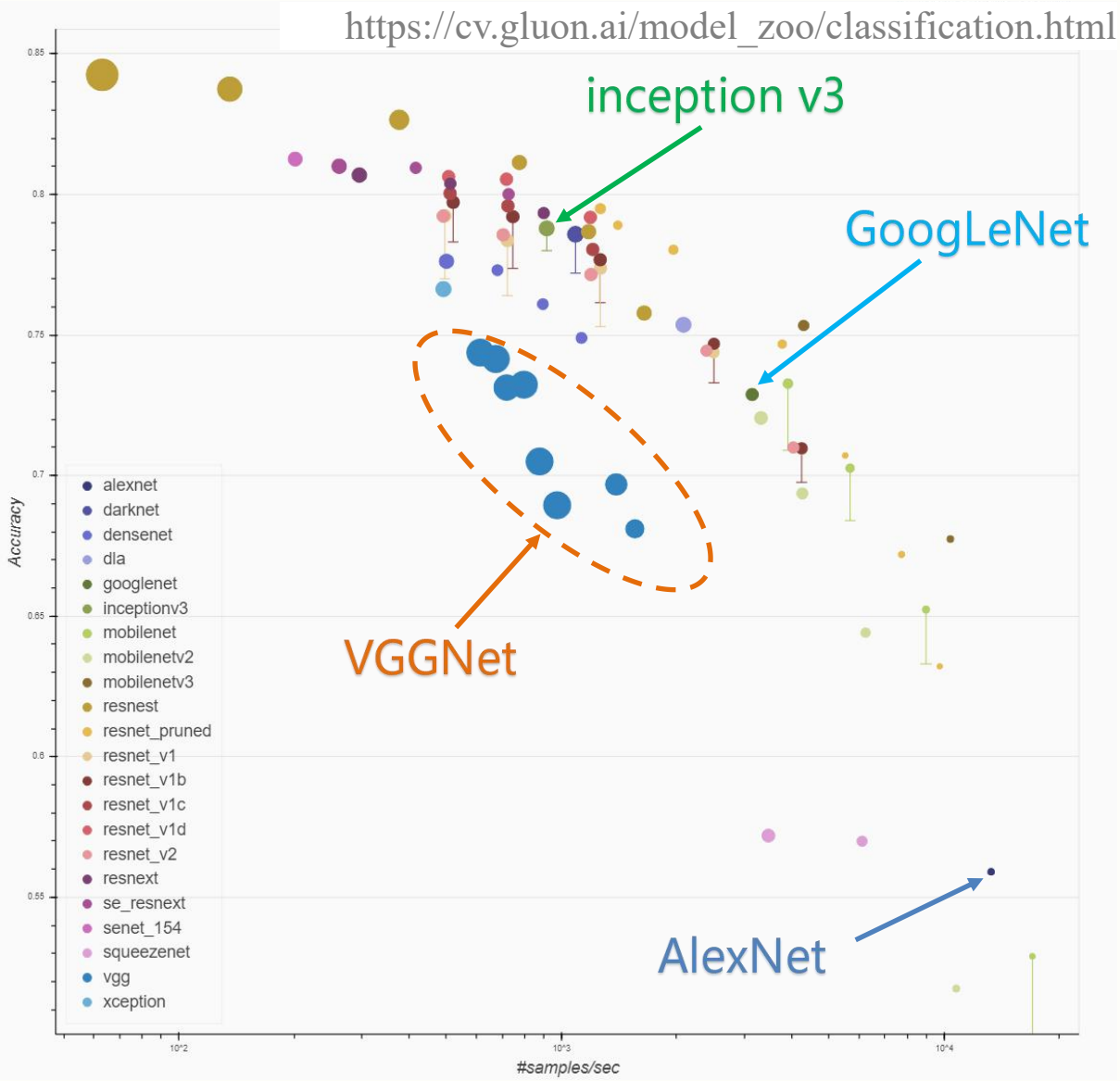
Christian Szegedy, Vincent Vanhoucke, sergey Ioffe, etc.al. Rethinking the Inception Architecture for Computer Vision. *arXiv1512.*

GoogLeNet的性能可视化

Visualization of Inference on GluonCV

- AlexNet
  - 精度：0.559
  - 内存：202Mb
  - 速度：13270 张/秒
- GoogLeNet
  - 精度：0.729
  - 内存：696Mb
  - 速度：3150 张/秒
- VGG16
  - 精度：0.732
  - 内存：3422Mb
  - 速度：797 张/秒
- inceptionV3
  - 精度：0.788
  - 内存：1052Mb
  - 速度：914 张/秒

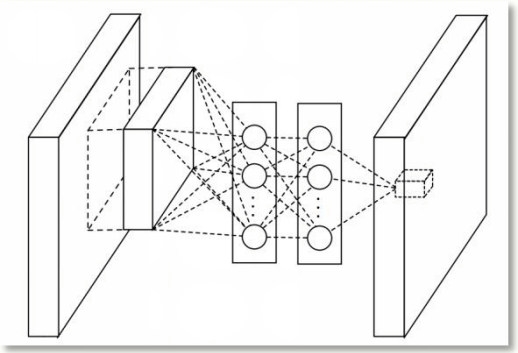
https://cv.gluon.ai/model_zoo/classification.html



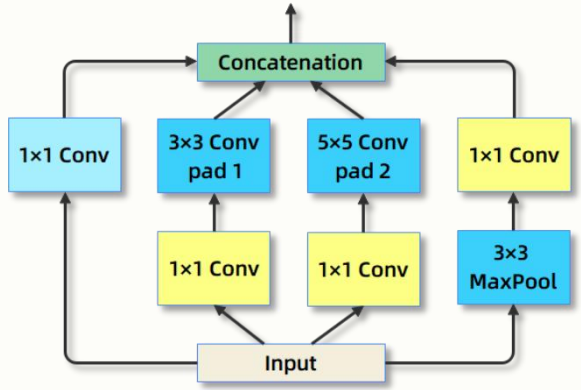
多分支网络（GoogLeNet）

小　结

- **NiN网络**是一个NiN块和最大池化层交替组合的模型。NiN模块等价于1个标准卷积加上2个1×1卷积；通过全局平均池化，NiN模型大幅缩减了参数的数量，不容易产生过拟合。
- **Inception块**使用**4条**不同超参数、不同结构的卷积层来抽取不同类型的信息。Inception块具有深度更深、参数更少、计算复杂性更低的特点。
- **GoogLeNet**使用了9个Inception模块，构建了一个**22层**的深度模型，它是第一个达到百层的网络。它获得了ILSVRC14分类任务冠军 (6.7% top 5 error)。**GoogLeNet**一系列后续的改进版，使得模型的性能和效率都得到了提升。

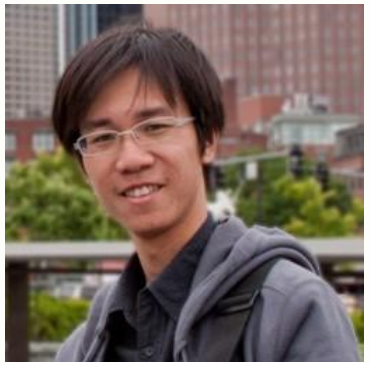


NiN 模块

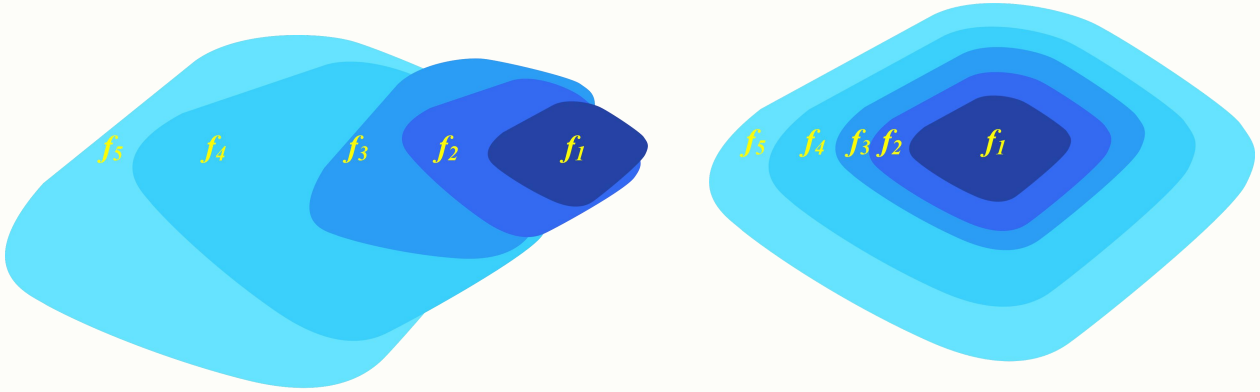


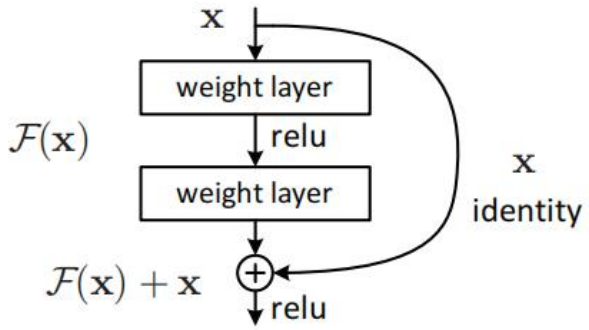
Inception 模块

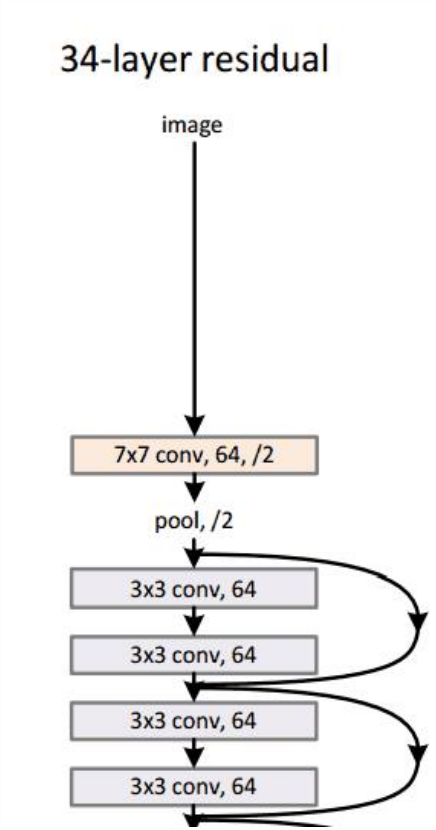
深度残差网络（ResNet）



Kaiming He

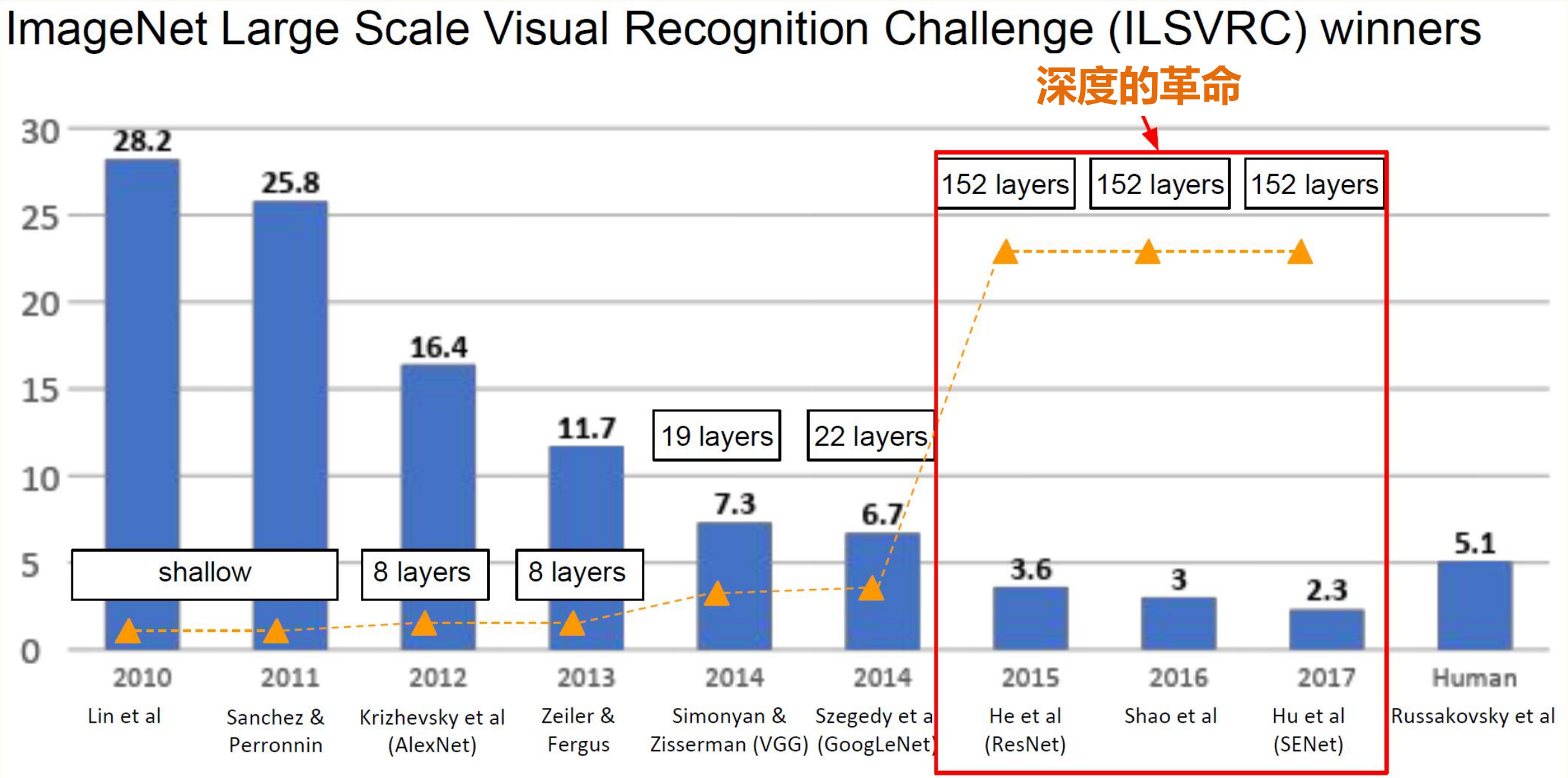






ResNet网络概述

ResNet网络概述


ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners
深度的革命
30
25
20
15
10
5
0
28.2
25.8
16.4
11.7
7.3
6.7
3.6
3
2.3
5.1
shallow
8 layers
8 layers
19 layers
22 layers
152 layers
152 layers
152 layers
2010
2011
2012
2013
2014
2014
2015
2016
2017
Human
Lin et al
Sanchez & Perronnin
Krizhevsky et al (AlexNet)
Zeiler & Fergus
Simonyan & Zisserman (VGG)
Szegedy et a (GoogLeNet)
He et al (ResNet)
Shao et al
Hu et al (SENet)
Russakovsky et al

ResNet网络概述

真正让深度学习变得很深的ResNet

深度学习最重要的特征是“深”，通过加深网络的层次实现图像、语音等样本识别率的提高。因此，不难想到的是，较深的网络应该**比**较浅的网络效果更好。如果要进一步提升模型的准确率，最直接的办法就是把网络设计得很深。

在神经网络中，第一个真正将网络做得**很深**的模型是**深度残差网络(ResNet)**，它由当时在MSRA的何凯明(Kaiming He) 团队设计完成。

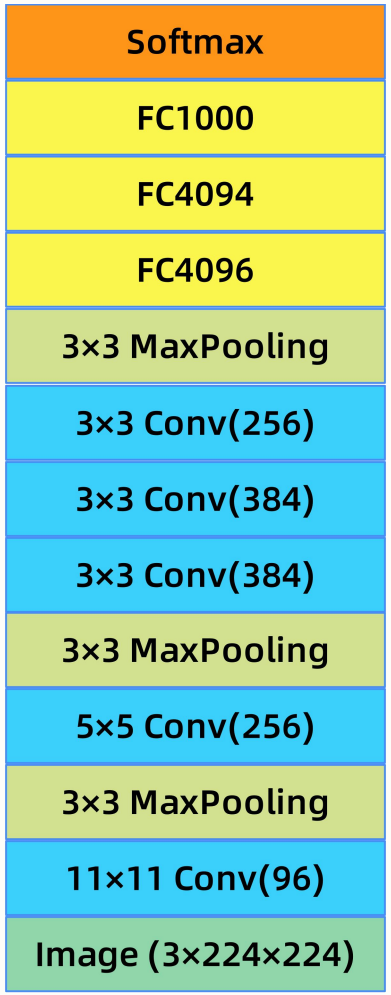
1st Places @ILSVRC & COCO 2015 竞赛

- ✓ ImageNet classification：**152-layer** nets
- ✓ ImageNet Detection：**16%** better than 2nd
- ✓ ImageNet Localization：**27%** better than 2nd
- ✓ COCO Detection：**11%** better than 2nd
- ✓ COCO Segmentation：**12%** better than 2nd

ResNet网络概述

深度网络的革命（从浅层到越来越深的层级）

AlexNet，7层
(ILSVRC 2012)

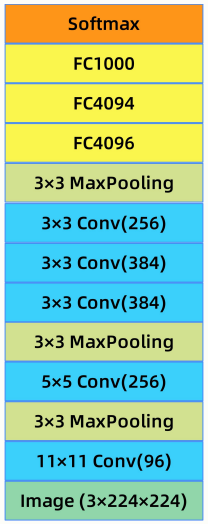


7层的AlexNet是基于卷积神经网络的深度学习第一次正式站上历史舞台，并从此让深度学习风靡了世界。

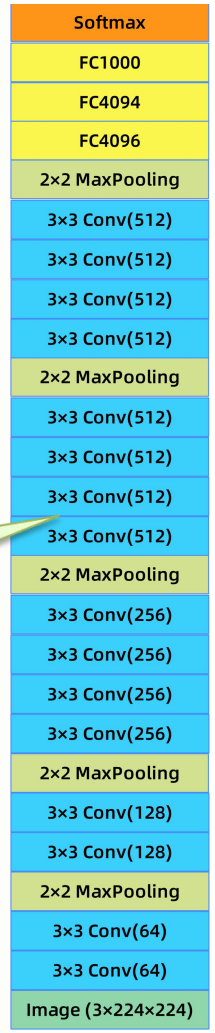
ResNet网络概述

深度网络的革命（从浅层到越来越深的层级）

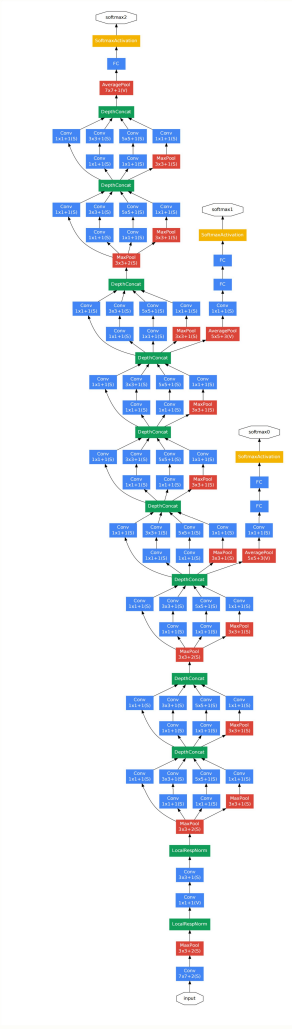
AlexNet，7层
(ILSVRC 2012)



VGG19,19层
(ILSVRC 2014)



GoogLeNet，22层
(ILSVRC 2014)

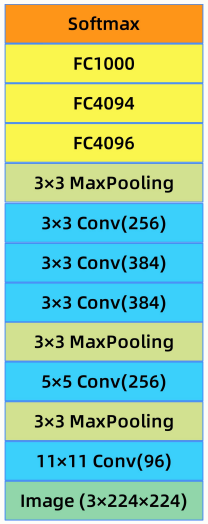


越来越深的网络，
越来越好的性能，让我们
看到了CNN的潜力。

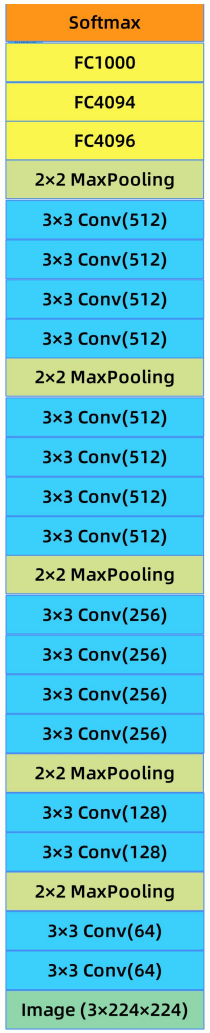
ResNet网络概述

深度网络的革命（ResNet之深）

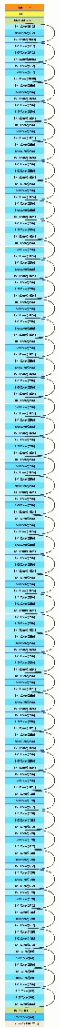
AlexNet，7层
(ILSVRC 2012)



VGG19,19层
(ILSVRC 2014)



ResNet152,152层
(ILSVRC 2015)

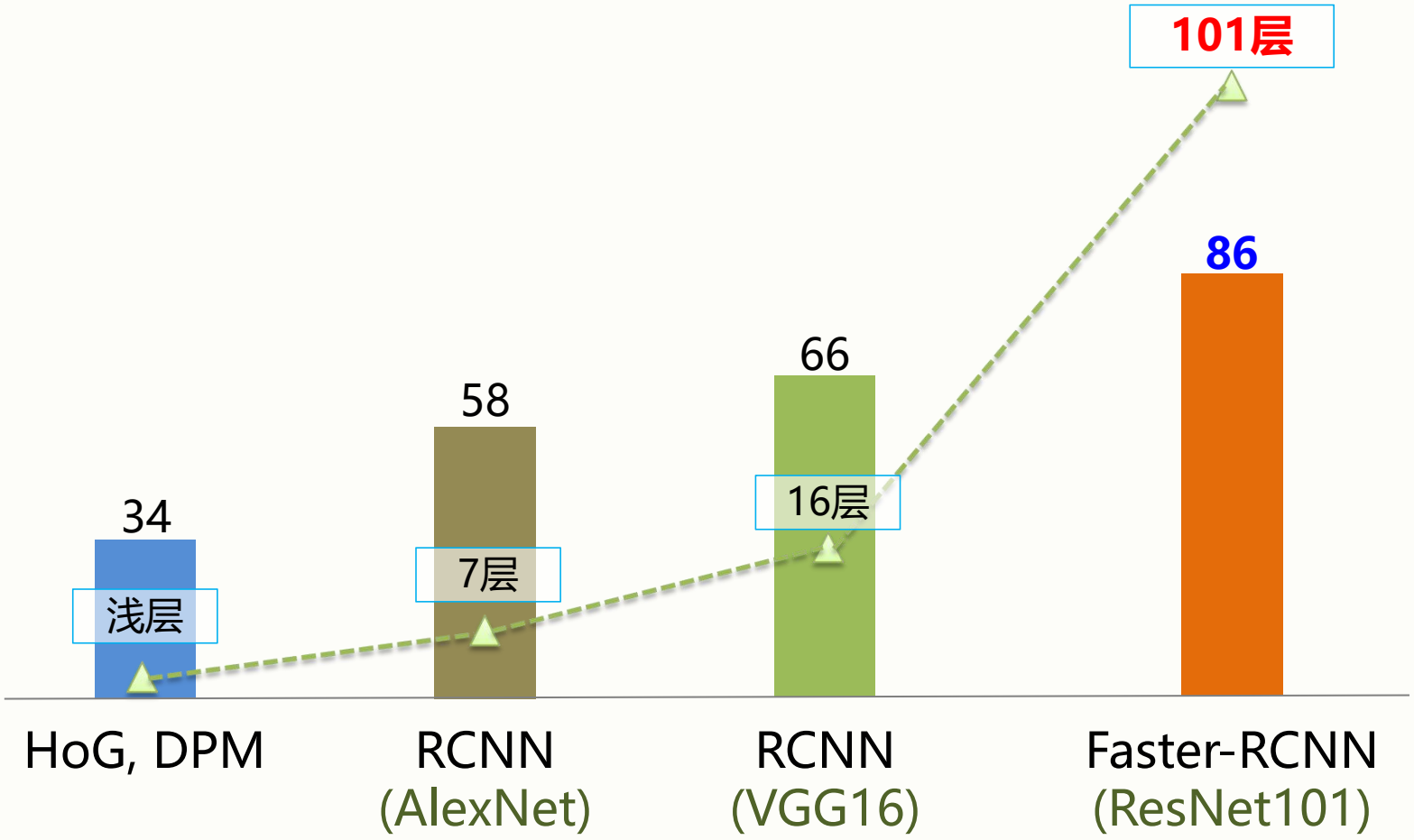


何凯明带领我们探索了CNN的极限，ResNet论文中尝试了超过1000层的网络。

ResNet网络概述

深度网络的革命（ResNet之深）

Pascal VOC2007 Detection(目标检测) mAP(%)



ResNet网络概述

VGG、GoogLeNet是否能更深?

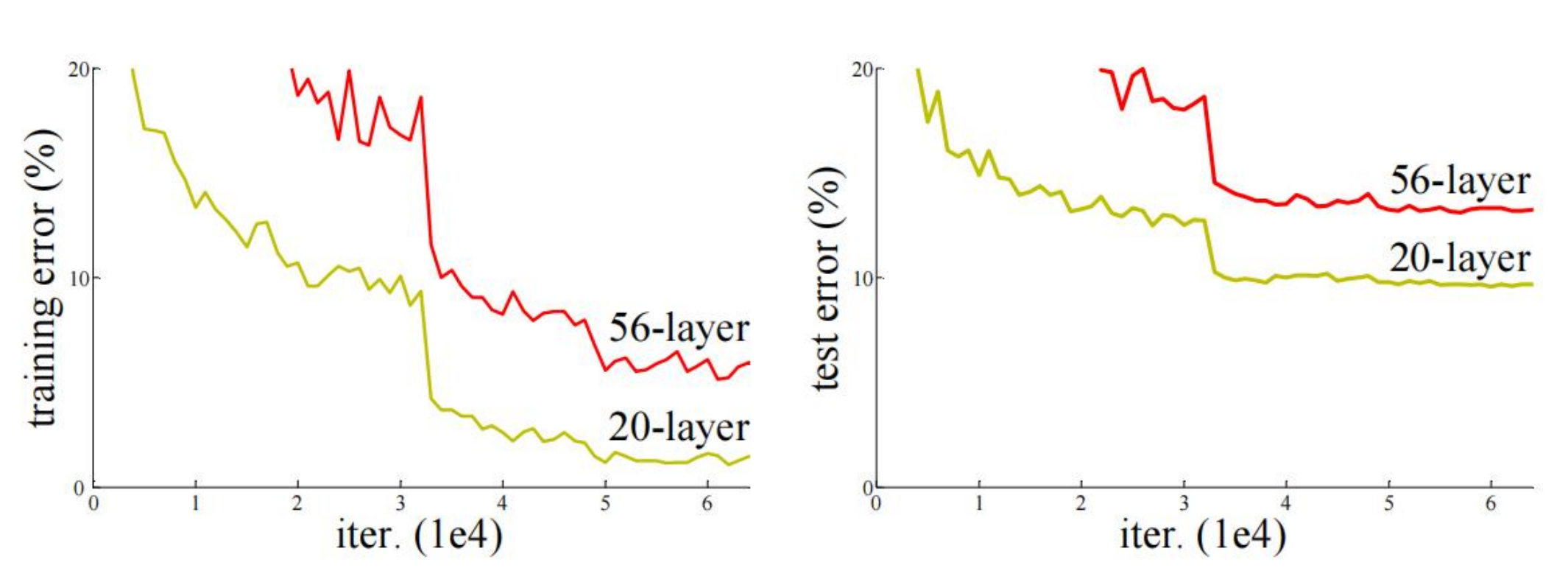
100层甚至1000层?

No!

ResNet网络概述

失效的“深度”

当我们在VGG等模型的基础上继续在对 plain CNN进行深度堆叠，会发生什么?



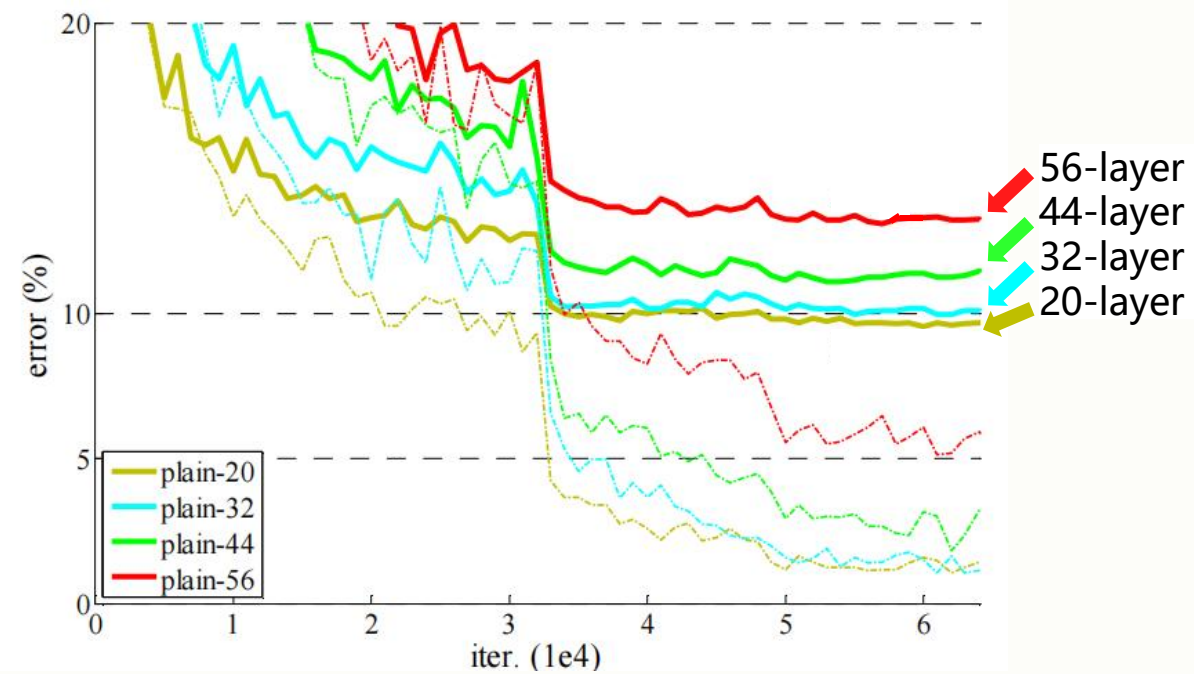
56层的网络比20层的网络具有更大的训练误差

深度模型性能更差，但这不是过拟合引起的

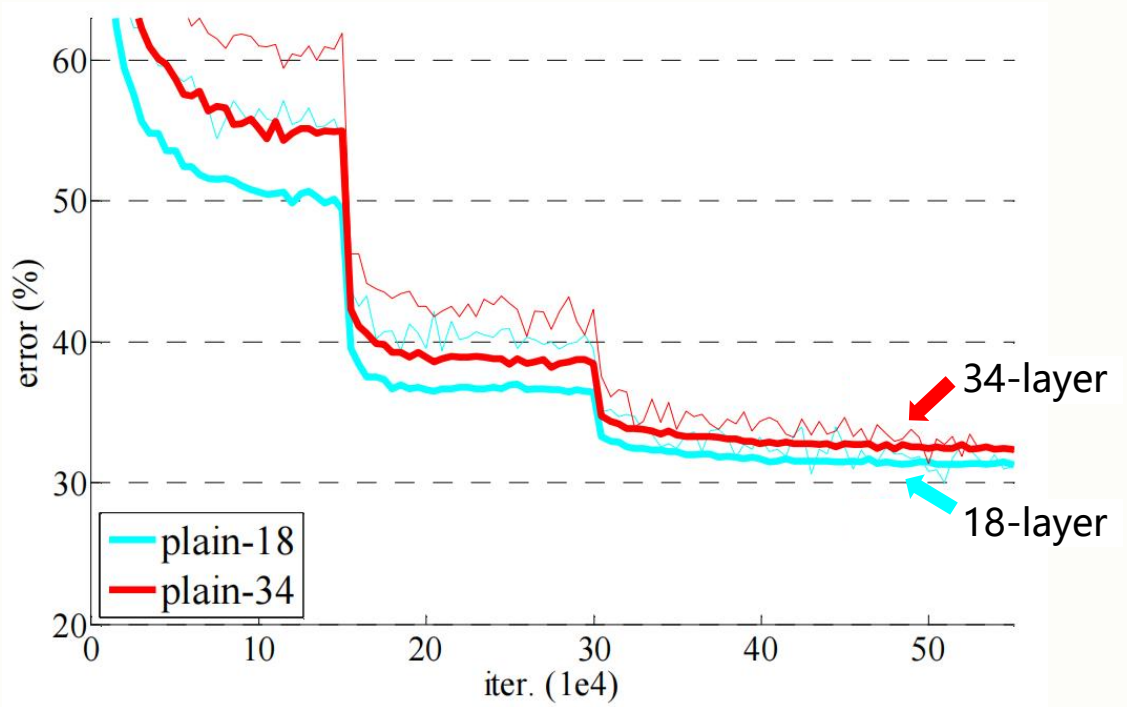
ResNet网络概述

失效的“深度”

CIFAR10



ImageNet



层数过深的普通网络(plain nets)具有更高的训练/验证误差

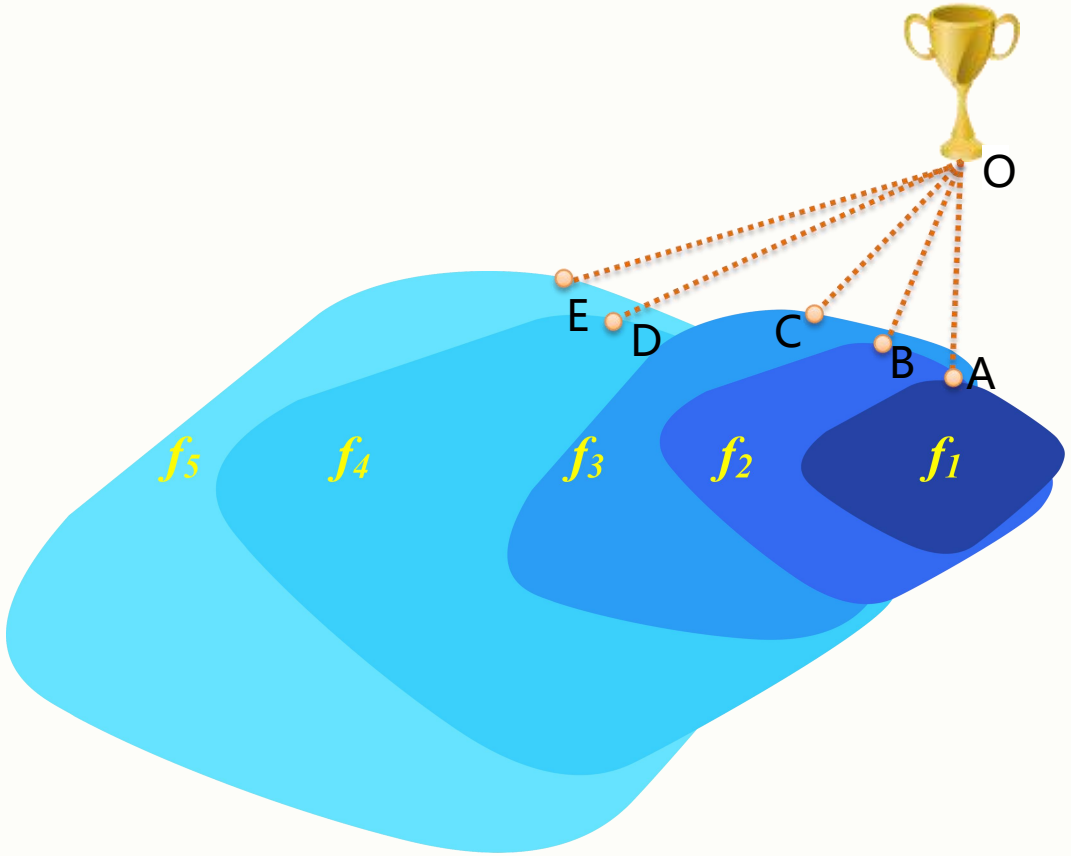
ResNet网络概述

深度学习失效了？

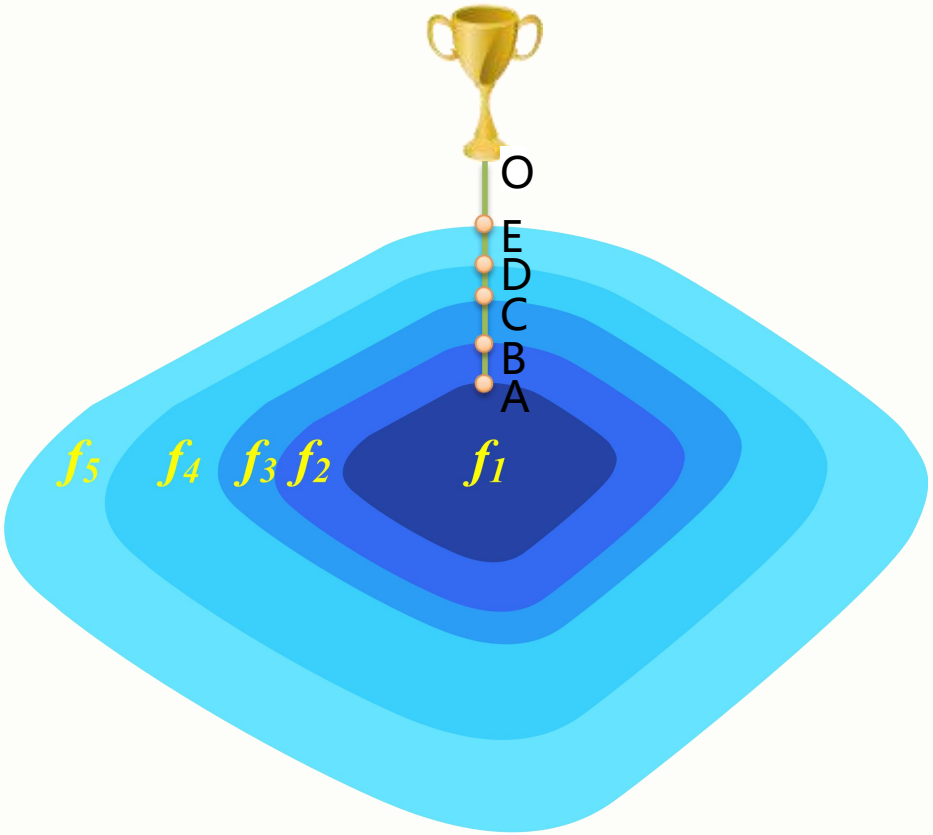
残差块的基本设计思路

残差块的基本设计思路

更复杂的模型一定能改进精度么?



通用模型

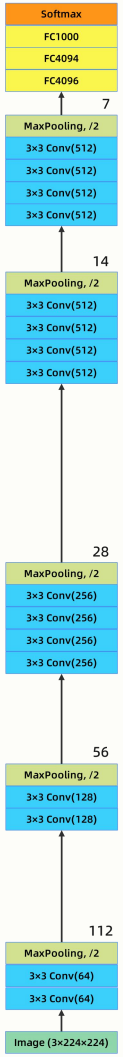


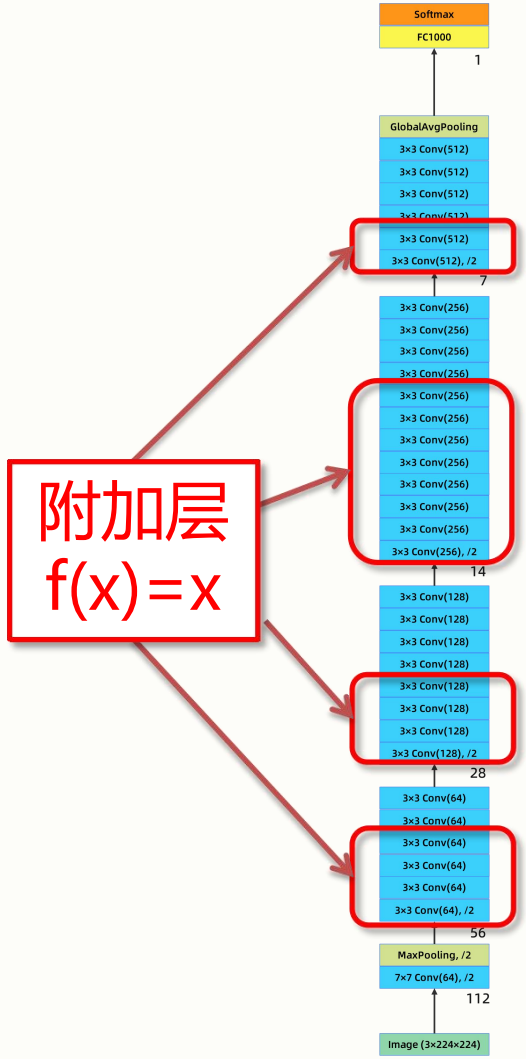
嵌套模型

残差块的基本设计思路

较深的网络模型不应该具有较高的的训练误差

相对较浅的模型
VGG19（19层）





相对较深的模型
VGG19扩展版
（34层）

考虑如下问题：

原始层： 已经训练好的较浅模型

附加层： 设置成“恒等”，即f(x)=x

目标： 至少获得与较浅模型相同的训练误差

问题： 如何实现优化

难点： 随着网络的不断加深，优化器无法找到实现目标的解决方法。

残差块的基本设计思路

使用恒等层增加网络深度

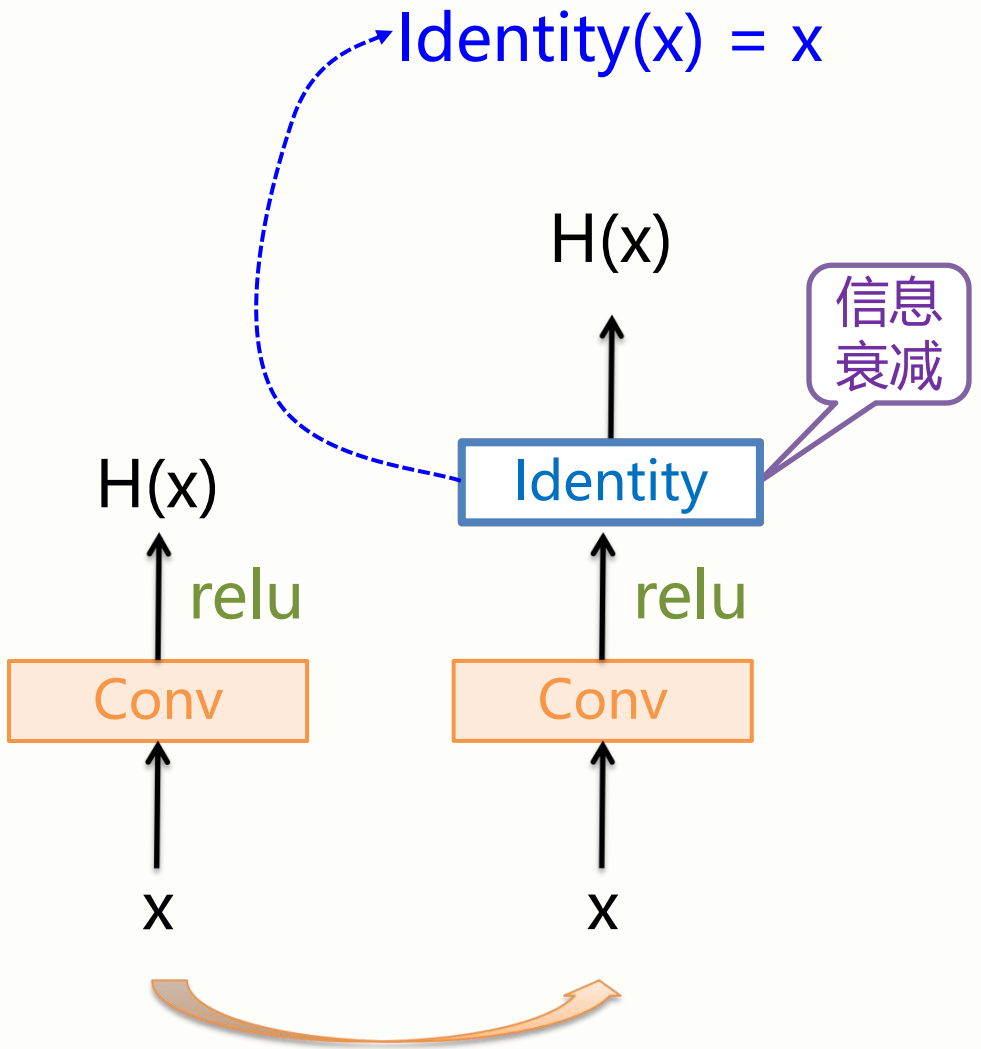
事实：深度学习因为有更多的参数，因此比浅层模型有更强的表达能力。

假设：**深度学习失效**的问题是优化问题，即深度模型更难被优化。

提问：更深的模型应该学到什么知识才能获得比它浅一些的模型**至少相同**的性能?

解决方案：在已经学到特征的浅层模型后面附加一个恒等映射层。

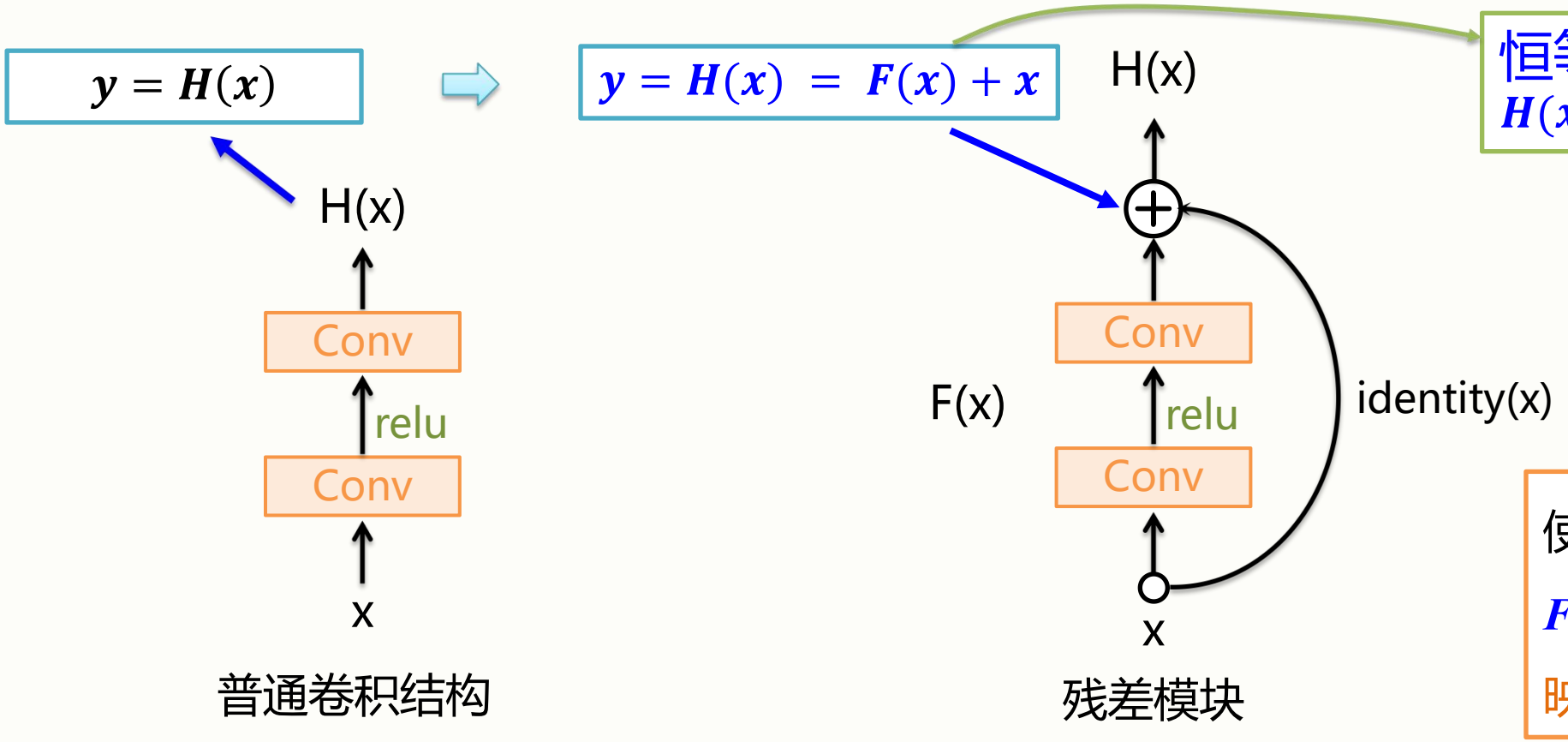
=》深度增加了，但性能并没有下降。



残差块的基本设计思路

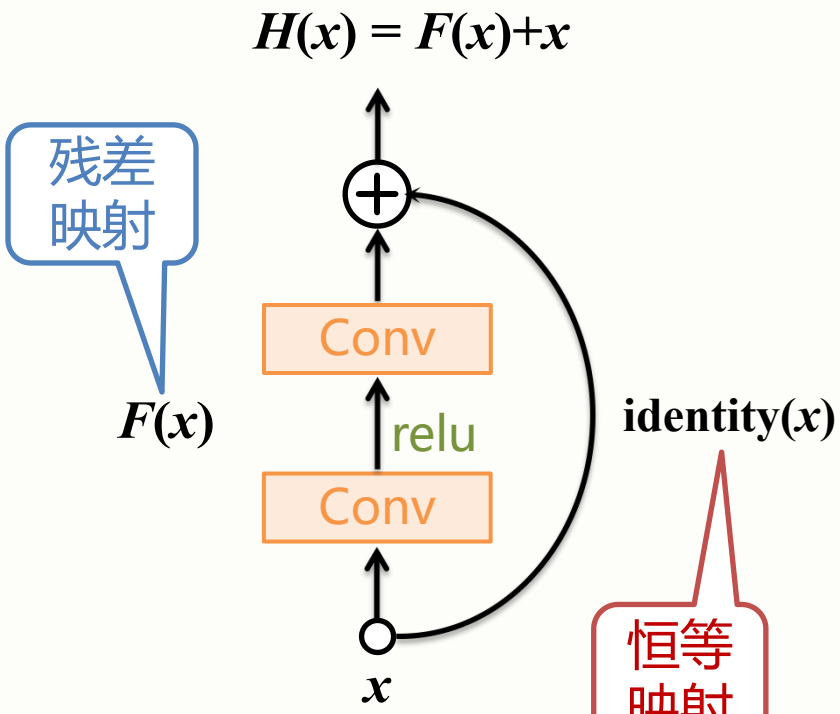
残差模块设计思想

假设存在任意一种网络中的任意一种理想映射$H(x)$，
该映射表示从输入x向输出 $y=H(x)$ 的转换。



使用层去适配残差，用$F(x)=H(x)-x$替代直接映射$H(x)$。

残差块的基本设计思路



残差模块

残差结构的数学解释

- 前向传输

$$x_{l+1} = H(x_l) = F(x_l) + x_l$$

$$x_{l+2} = H(x_{l+1}) = F(x_{l+1}) + x_{l+1} = F(x_{l+1}) + F(x_l) + x_l$$

$$\boldsymbol{x_L} = x_l + \sum_{i=l}^{L-1} F(x_i)$$

*l*层与*L*层之间的累计残差

- 反向传输

$$\frac{\partial loss}{\partial x_l} = \frac{\partial loss}{\partial x_L}\frac{\partial x_L}{\partial x_l} = \frac{\partial loss}{\partial x_L}\left(1 + \frac{\partial}{\partial x_l}\sum_{i=l}^{L-1} F(x_i)\right)$$

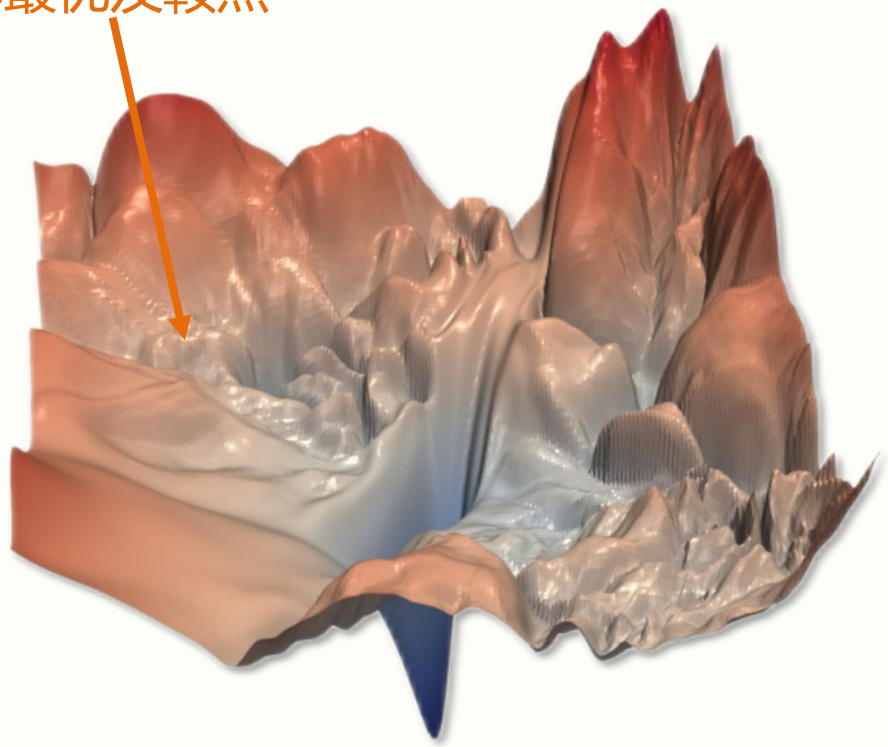
$$= \frac{\partial loss}{\partial x_L} + \frac{\partial loss}{\partial x_L}\frac{\partial}{\partial x_l}\sum_{i=l}^{L-1} F(x_i)$$

不会永远等于-1

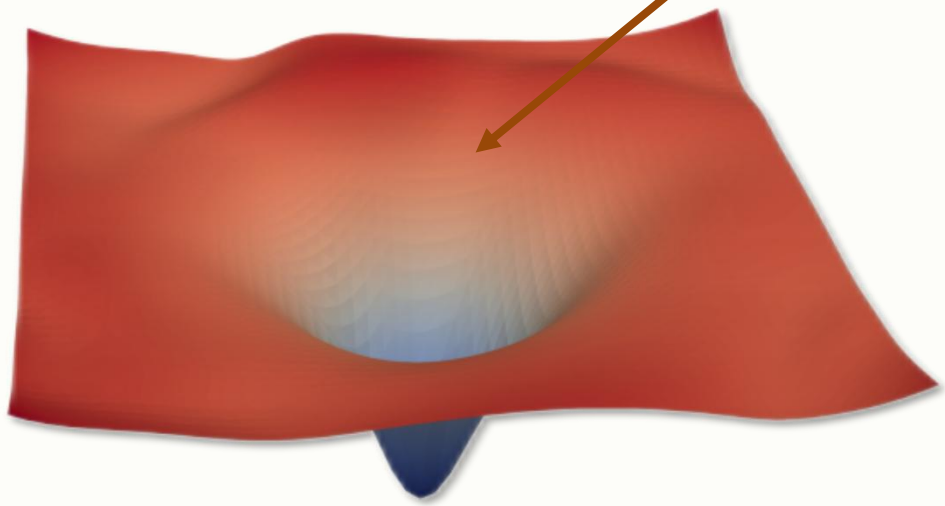
残差块的基本设计思路

残差结构的可视化对比

充斥着大量的局部最优及鞍点



曲线更平滑
极值点更明显



(a) 未使用SkipConnections的ResNet56

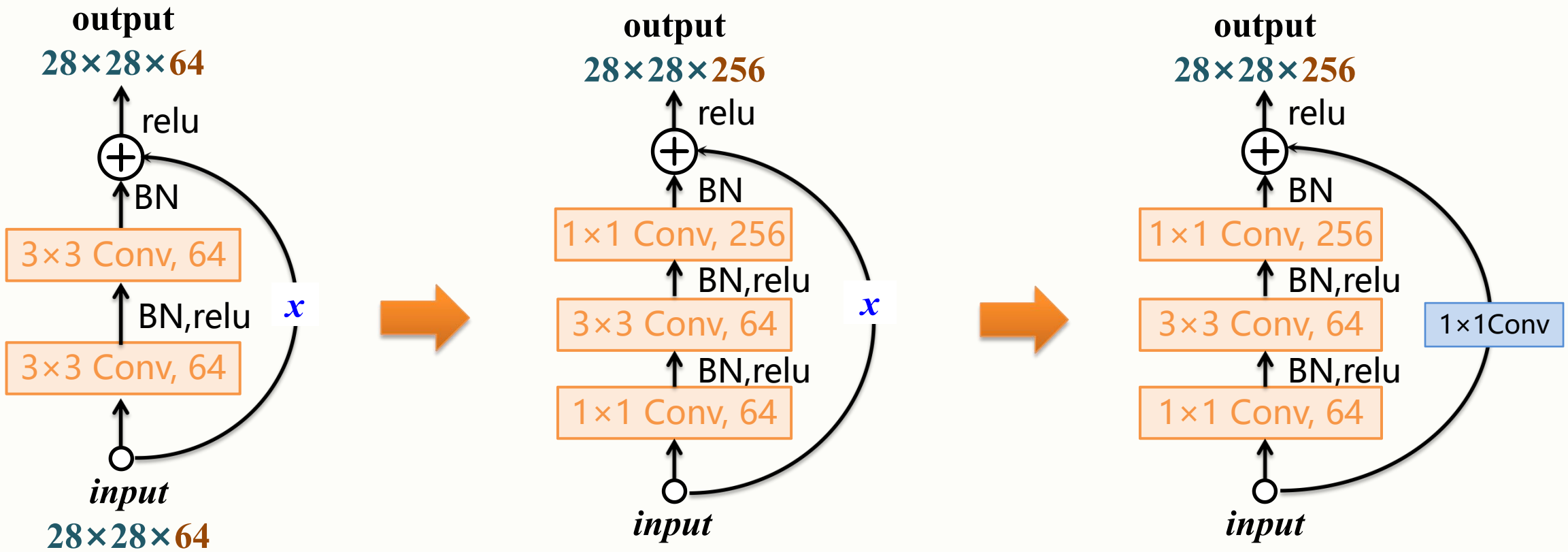
(b) 使用SkipConnections的ResNet56

Figure 1: The loss surfaces of ResNet-56 with/without skip connections. The proposed filter normalization scheme is used to enable comparisons of sharpness/flatness between the two figures.

Hao Li, Zheng Xu, et.al. Visualizing the Loss Landscape of Neural Nets. *NIPS2018*.

残差块的基本设计思路

使用瓶颈(bottleneck)改进残差结构



标准残差模块
(ResNet-18/34)

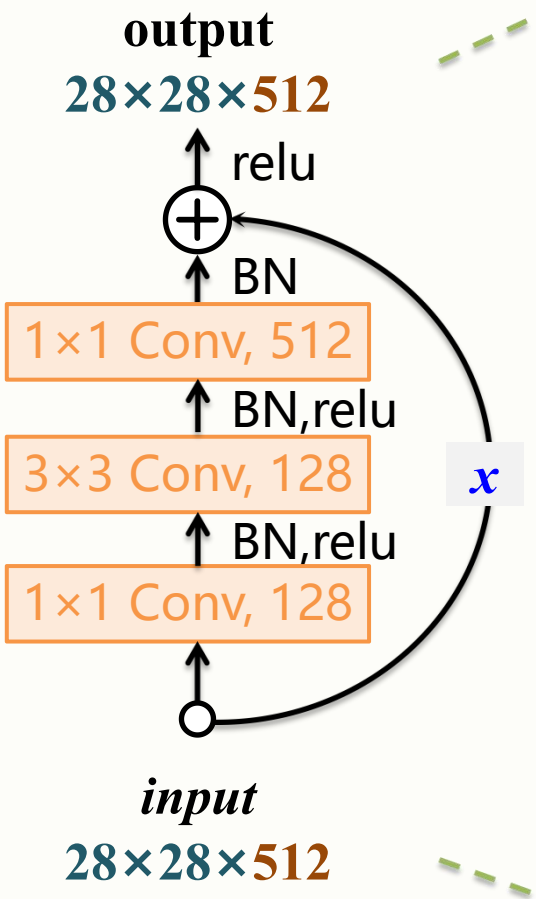
瓶颈残差模块
(ResNet-50/101/152)

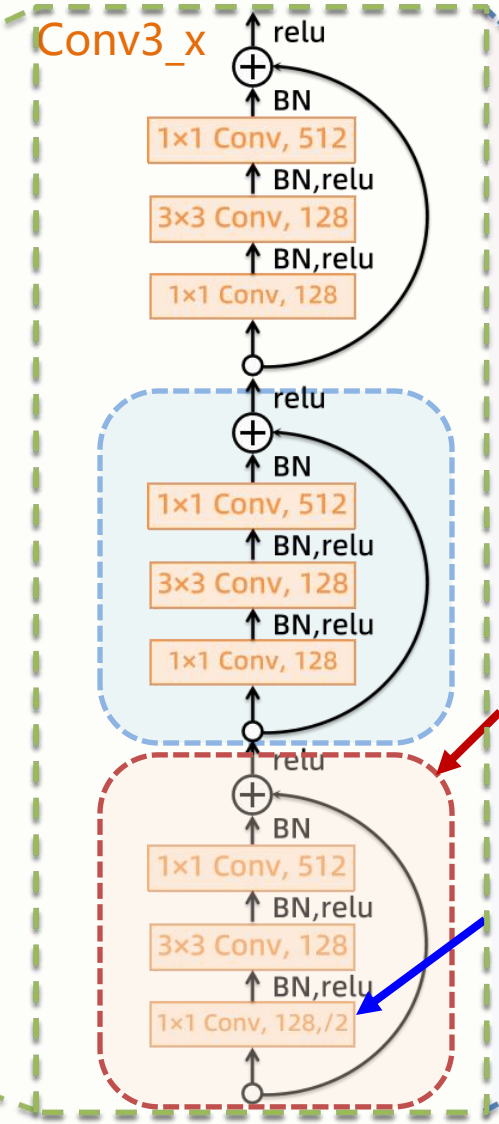
通道可控的瓶颈残差模块
(ResNet-50/101/152)

Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. Identity Mapping in Deep Residual Networks. *arXiv1603.*

ResNet网络体系结构详解

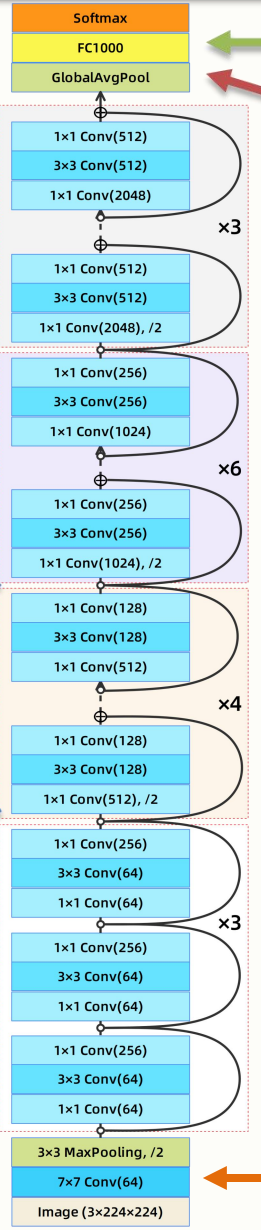
ResNet网络体系结构详解





高宽减半模块被作为默认被添加到所有卷积组中

stride=2 实现空间尺度减半



分类FC层

全局平均池化

四个残差卷积组

开始的卷积层

ResNet网络体系结构详解

经典ResNet结构表

特征图尺度缩小8倍

普通残差模块

瓶颈残差模块

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		[3×3, 64; 3×3, 64] ×2	[3×3, 64; 3×3, 64] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3
conv3_x	28×28	[3×3, 128; 3×3, 128] ×2	[3×3, 128; 3×3, 128] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×8
conv4_x	14×14	[3×3, 256; 3×3, 256] ×2	[3×3, 256; 3×3, 256] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×23	[1×1, 256; 3×3, 256; 1×1, 1024] ×36
conv5_x	7×7	[3×3, 512; 3×3, 512] ×2	[3×3, 512; 3×3, 512] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

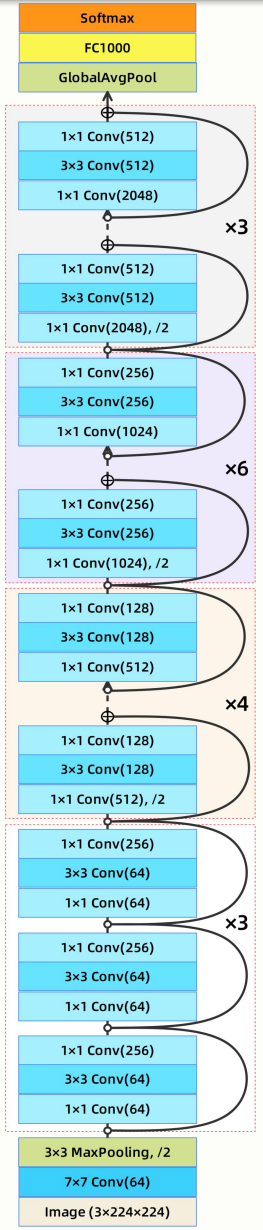
conv 3×3：padding=1;
conv 1×1：padding= 0
stride(conv3_1, conv4_1, conv5_1)=2,　stride_others=1

瓶颈设计使得深度增加50%，但计算量增加不大

ResNet网络体系结构详解

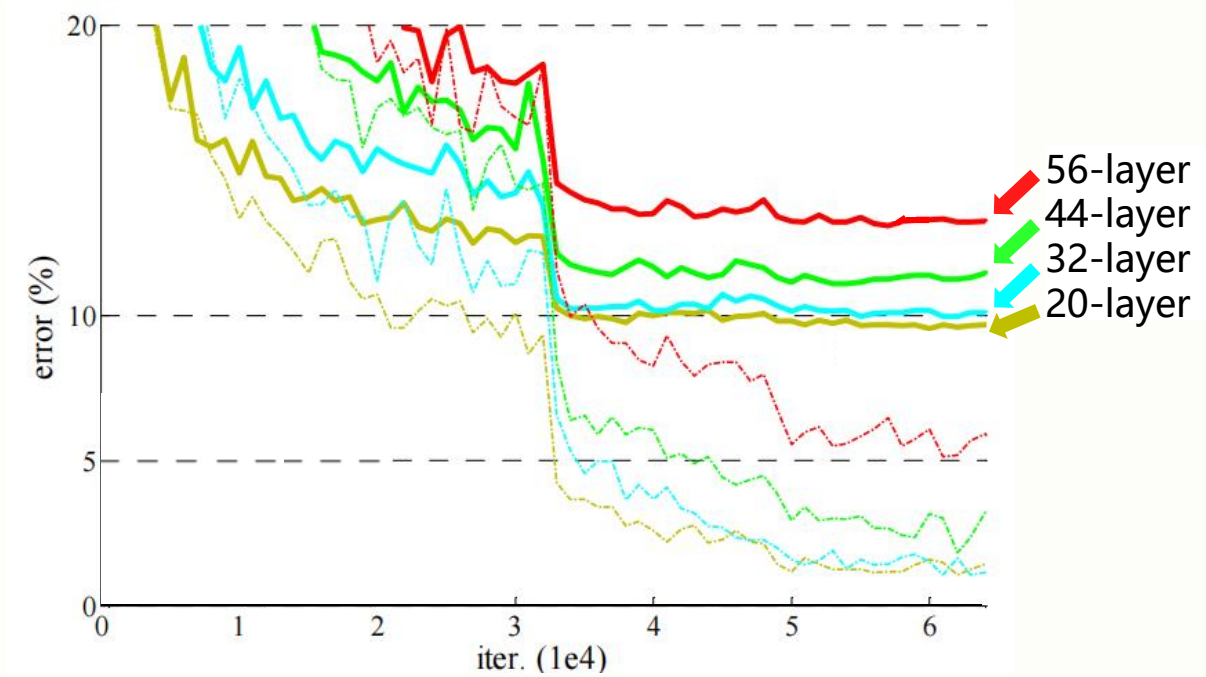
更多细节:

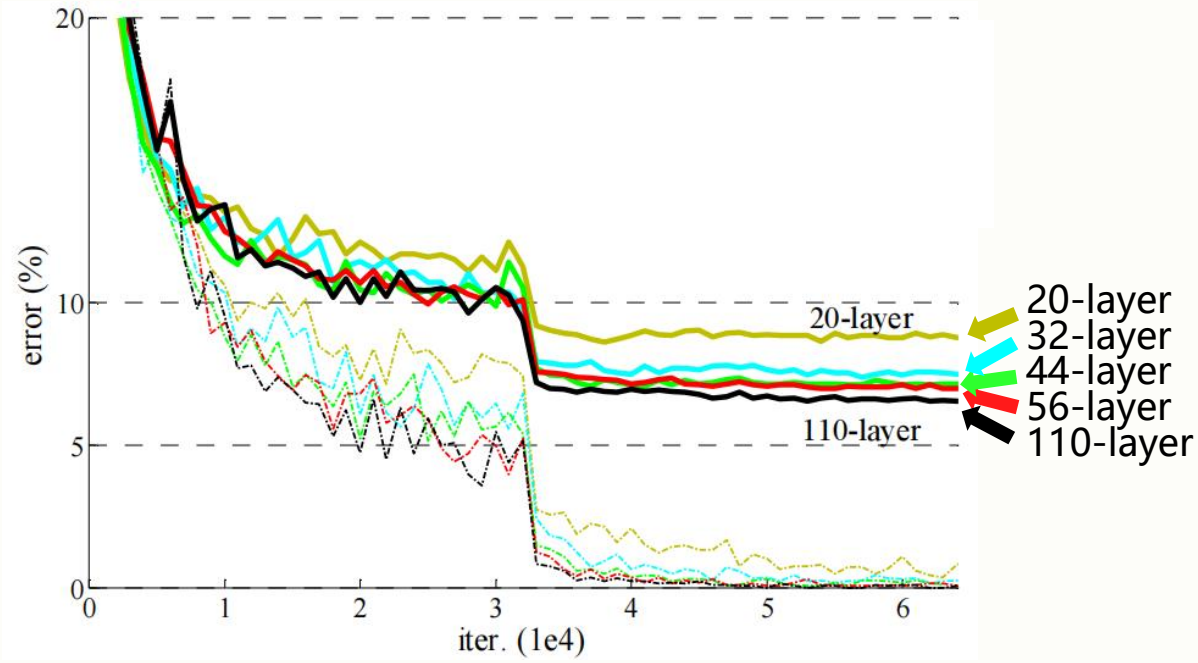
- 每个卷积层后都使用Batch Normalization批正则化
- 使用 Xavier(He et al.) 替换随机初始化
- 优化方法：SGD + Momentum (0.9)
- 学习率: **0.1**起步(worm up), 验证误差不再改变后缩小10倍
- Mini-batch：256, L2权重衰减系数：1e-5
- 最后一个卷积层后面紧跟一个**全局平均池化层**
- 没有使用dropout（除分类器，没有额外的FC层）
- 所有模型都从头开始训练（from scratch），未使用微调训练
- 标准化的超参数(Hyper-parameters)和数据增广(Augmentation)



ResNet网络体系结构详解

CIFAR-10 实验

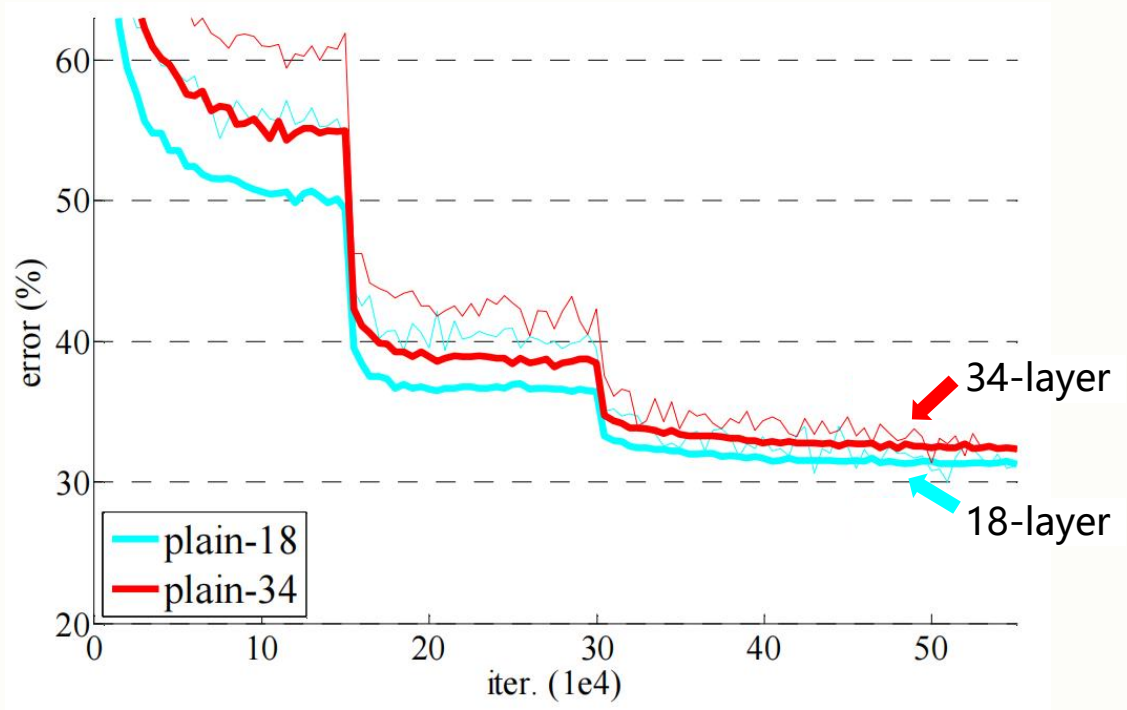


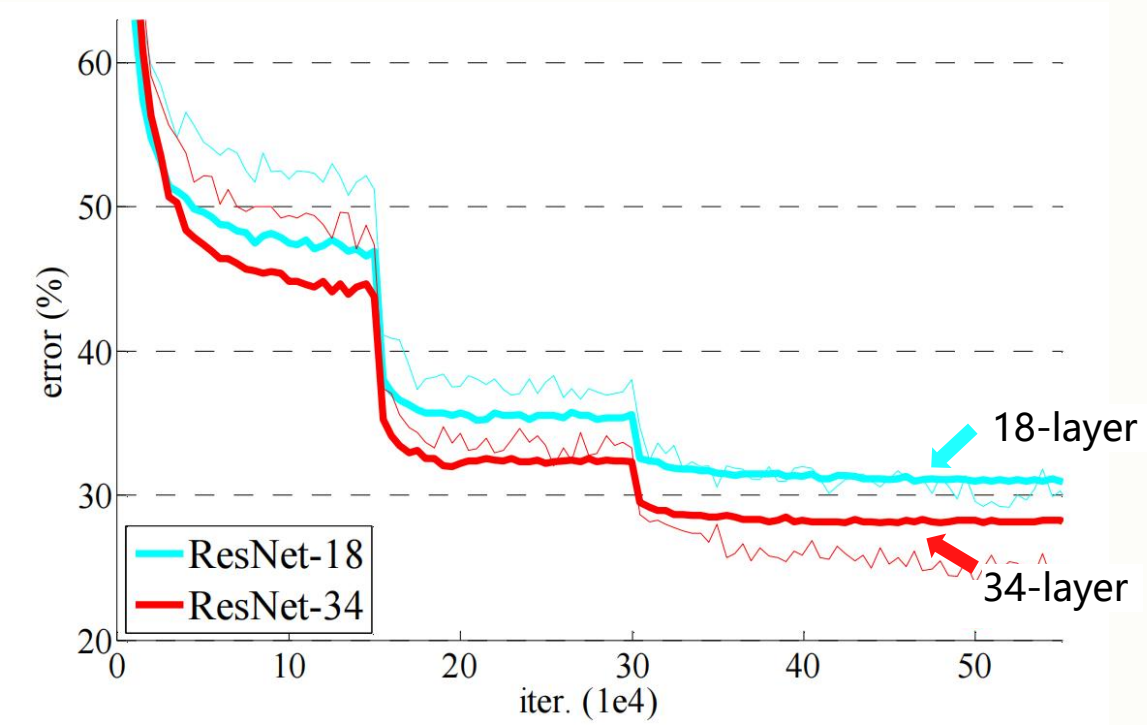


深度残差网络(ResNets)具有较低的训练误差和测试误差

ResNet网络体系结构详解

ImageNet 实验





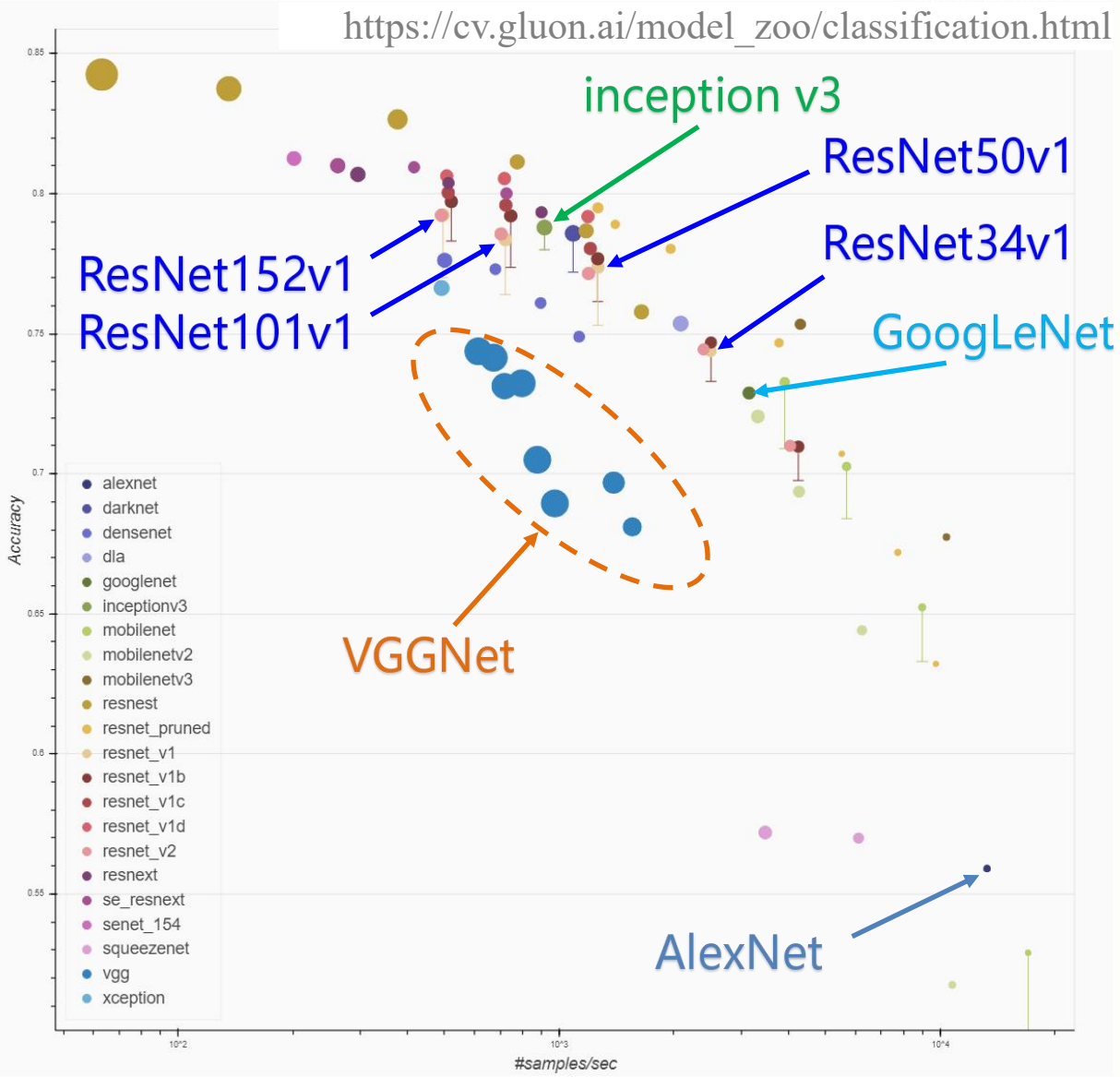
残差结构有效地改进了神经网络在较深的设置下的梯度弥散问题，从而使得通过构建更深的神经网络来提高性能变成可能。

ResNet的性能可视化

Visualization of Inference on GluonCV

- AlexNet
  - 精度：0.559
  - 内存：202Mb
  - 速度：13270 张/秒
- VGG16
  - 精度：0.732
  - 内存：3422Mb
  - 速度：797 张/秒
- ResNet的精度
  - ResNet34：0.744
  - ResNet50：0.774
  - ResNet101：0.784
  - ResNet152：0.792
- 推理速度（张/秒）
  - ResNet34：2497
  - ResNet50：1263
  - ResNet101：722
  - ResNet152：495

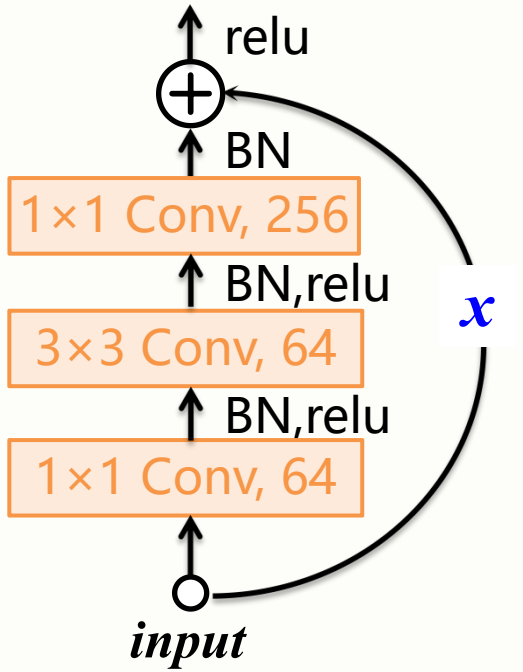
https://cv.gluon.ai/model_zoo/classification.html



ResNet模型小结

Summary

1. 残差块有效缓解了梯度弥散带来的训练困难的问题很深的网络更加容易训练。在***Imagenet***上实现总深度18, 34, 50, 101, 152的模型，在***CIFAR***上实现了超过1000层的训练。
2. 瓶颈结构进一步提高了残差结构的性能
3. 间歇性（组）地实现卷积特征图数量×2以及尺度减半（stride=2）
4. 残差网络保持了网络“设计”的简约性，使用VGG-Style，所有卷积组都是3×3和1×1卷积，结构简单，无全连接层、无Dropout
5. ResNet通过堆叠残差模块构建网络，性能优异，横扫了ILSVRC15和COCO15竞赛中所有的分类和检测任务，并超越了人的识别能力。
6. 残差网络对随后的深层神经网络设计产生了深远影响



参考文献：
[1] He Kaiming, Zhang Xiangyu, Ren Shaoqing, JianSun. Deep Residual Learning for Image Recognition. *CVPR2016*.
[2] He Kaiming, Zhang Xiangyu, Ren Shaoqing, JianSun. Identity Mappings in Deep Residual Networks. arXiv1603.

欧老师的联系方式

读万卷书 行万里路 只为最好的修炼

QQ：14777591 (宇宙骑士)

Email：ouxinyu@alumni.hust.edu.cn

Website: http://ouxinyu.cn

Tel：18687840023

地址：安宁校区 诚远楼201

南院 智能应用研究院A306-2