Deep Learning I

7. 现代卷积神经网络

WU Xiaokun 吴晓堃

xkun.wu [at] gmail

2021/04/04

深度卷积神经网络 AlexNet

回顾：LeNet

LeNet是早期神经网络的成功案例

- 但是在更大、更真实的数据集上训练卷积神经网络?
- 90年代初到2012年之间：其他机器学习算法，例如支持向量机

回顾：LeNet

LeNet是早期神经网络的成功案例

- 但是在更大、更真实的数据集上训练卷积神经网络?
- 90年代初到2012年之间：其他机器学习算法，例如支持向量机

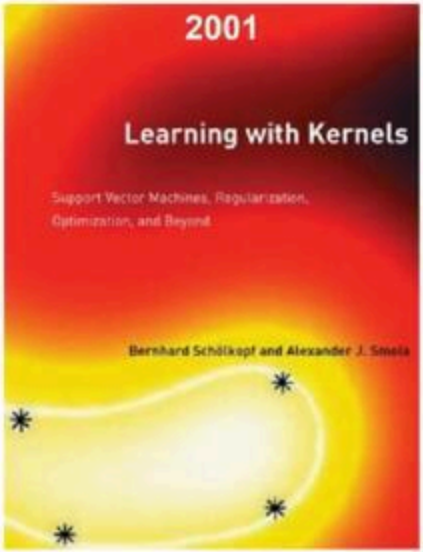
期间：卷积神经网络更像是“异类”

- 输入：原始像素值，或是简单预处理过的像素
- 经典机器学习：核心是精心设计的**特征流水线**

理论学派：机器学习

核方法：支持向量机 SVM

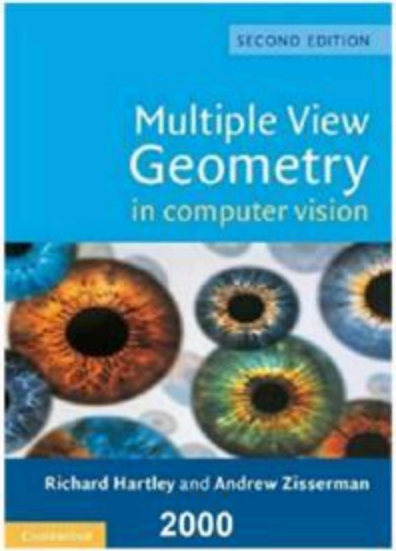
- 特征工程：构造核函数
- 凸优化：公式推导漂亮



理论学派：视觉几何

多视图几何：主要用于计算机视觉

- 描述图像内在的（三维）几何特征
- 工业实践的标准方法



早期特征工程

特征：描述目标对象的标识

- 图像：关键点附近的像素分布
  - SIFT、SURF
- 文本：词的分布
  - 词袋、词嵌入

通常这些特征来源于**偶然的发现**

- 算法往往归于事后的解释

早期特征工程

特征：描述目标对象的标识

- 图像：关键点附近的像素分布
  - SIFT、SURF
- 文本：词的分布
  - 词袋、词嵌入

通常这些特征来源于**偶然的发现**

- 算法往往归于事后的解释

人工设计好特征之后才能应用SVM等经典机器学习算法

- 当时研究的主流：设计特征、实验对比、包装成论文

理论分歧：特征工程有多重要?

经典机器学习的**工作流程**

1. 创建数据集：早期传感器非常昂贵
2. 特征工程：光学、几何学、其他知识、偶然发现
3. 特征流水线：将提取的特征转化成输入数据
4. 训练分类器：例如SVM

理论分歧：特征工程有多重要?

经典机器学习的**工作流程**

1. 创建数据集：早期传感器非常昂贵
2. 特征工程：光学、几何学、其他知识、偶然发现
3. 特征流水线：将提取的特征转化成输入数据
4. 训练分类器：例如SVM

机器学习：倾向于**理论证明**；忽略工程实践的话，数学分析的确很有趣

理论分歧：特征工程有多重要?

经典机器学习的**工作流程**

1. 创建数据集：早期传感器非常昂贵
2. 特征工程：光学、几何学、其他知识、偶然发现
3. 特征流水线：将提取的特征转化成输入数据
4. 训练分类器：例如SVM

机器学习：倾向于**理论证明**；忽略工程实践的话，数学分析的确很有趣

视觉实践：必须能够解决实际问题，就会发现算法改进不太重要

- 推动领域进步的是**数据工程**，而繁琐的数据处理非常枯燥

数据，硬件

年代	数据规模	内存	每秒浮点运算
1970	100 （鸢尾花卉）	1 KB	100 KF (Intel 8080)
1980	1 K （波士顿房价）	100 KB	1 MF (Intel 80186)
1990	10 K （光学字符识别）	10 MB	10 MF (Intel 80486)
2000	10 M （网页）	100 MB	1 GF (Intel Core)
2010	10 G （广告）	1 GB	1 TF (Nvidia C2050)
2020	1 T （社交网络）	100 GB	1 PF (Nvidia DGX-2)

ImageNet (2010)

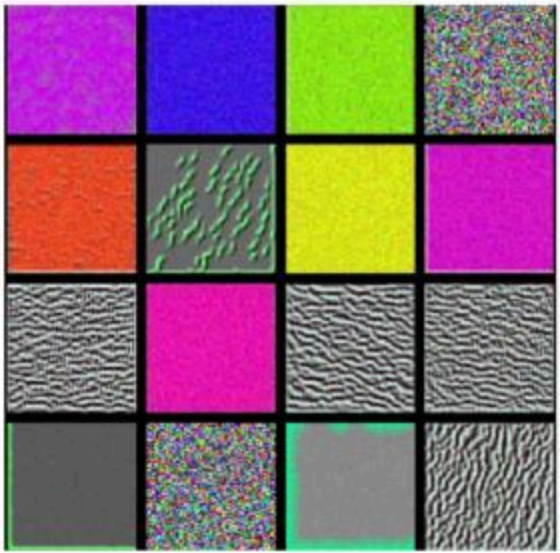
	ImageNet	MNIST
图片	自然物体、彩色	手写数字、黑白
大小	469x387	28x28
样本数	1.2M	60K
类别数	1000	10

- 所有传统方法（包括机器学习）都无法处理这个量级的数据

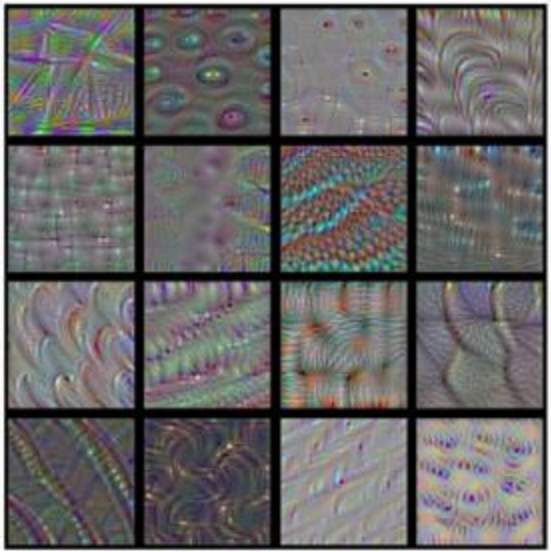
实践派的崛起

计算机视觉的新思路：**特征本身应该是能够被学习的参数**

底层：边缘、颜色和纹理



高层：复合纹理、抽象概念



AlexNet

首个**深度**卷积神经网络

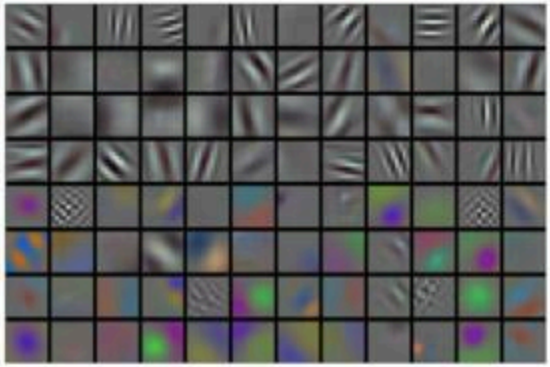
- 2012年ImageNet竞赛冠军
- 对LeNet的主要改进
  - 丢弃法
  - ReLU
  - 最大池化

AlexNet：学术贡献

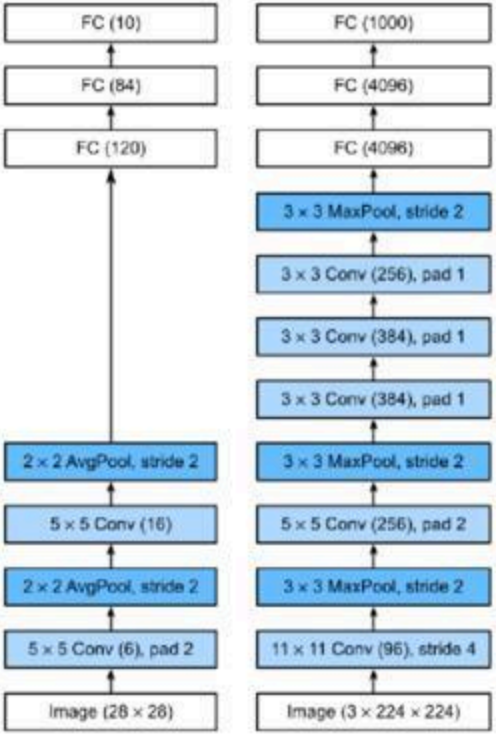
- 首个**深度**卷积神经网络
- 2012年ImageNet竞赛冠军
- 对LeNet的主要改进
  - 丢弃法
  - ReLU
  - 最大池化

引发计算机视觉**方法论的改变**

- 之前：人工特征工程
  - SVM判定类别
- 之后：CNN自动提取特征
  - softmax回归分类

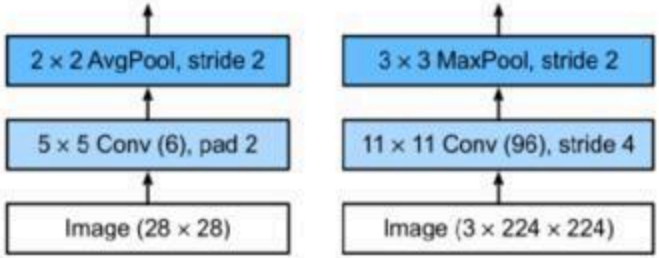


AlexNet 架构：对比 LeNet



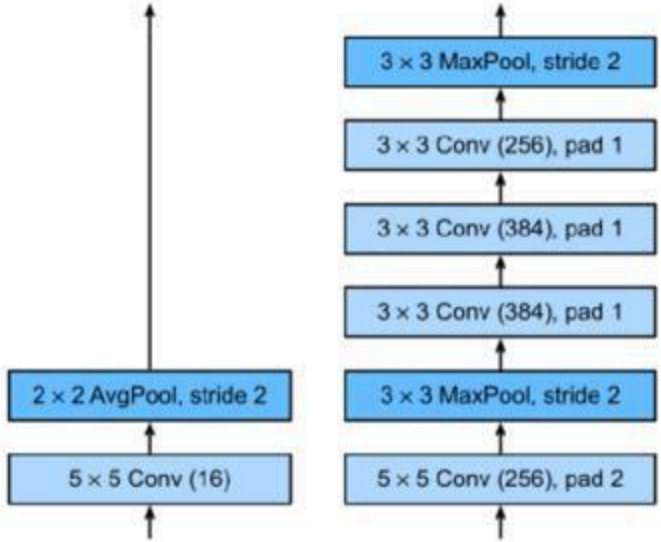
AlexNet 架构：卷积I

- 更大的池化窗口
  - 最大池化
- 更大的核、步长
  - 11x11
- 更大的图片
  - 3通道（彩色）



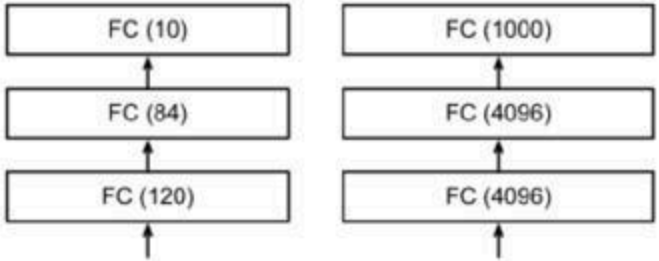
AlexNet 架构：卷积II

- 新加3个卷积层、1个池化层
- 更多的输出通道：增多，再减少



AlexNet 架构：全连接

- 问题更复杂：输出1000个类
- 展平成更长的向量



更多技术细节

激活函数：从sigmoid改成ReLU

- 缓解梯度消失问题

更多技术细节

激活函数：从sigmoid改成ReLU

- 缓解梯度消失问题

全连接模块中加入了丢弃层

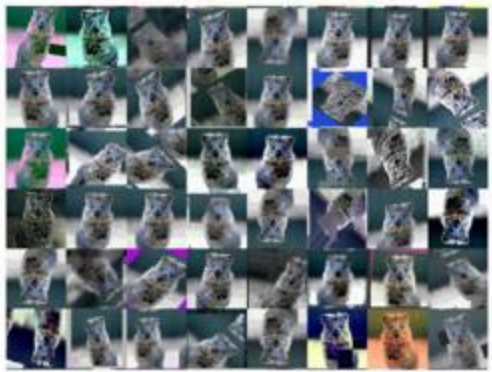
更多技术细节

激活函数：从sigmoid改成ReLU

- 缓解梯度消失问题

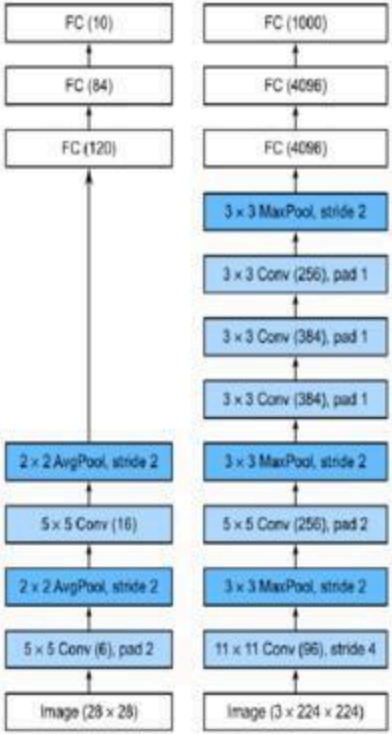
全连接模块中加入了丢弃层

数据增广：增强数据多样性



复杂度

	参数量		FLOPS	
Conv1	35K	150	101M	1.2M
Conv2	614K	2.4K	415M	2.4M
Conv3-5	3M	N/A	445M	N/A
Dense1	26M	0.48M	26M	0.48M
Dense2	16M	0.1M	16M	0.1M
Total	46M	0.6M	1G	4M
Increase	80x	1x	250x	1x
	AlexNet	**LeNet**	**AlexNet**	**LeNet**



实验：AlexNet

小结：AlexNet

- AlexNet是更大、更深的LeNet
  - 80x参数量，250x计算量
- 新加入：丢弃法、ReLU、最大池化、数据增广
- 标志着新一轮人工智能热潮的兴起
  - ImageNet 2012 竞赛冠军

使用块的网络 VGG

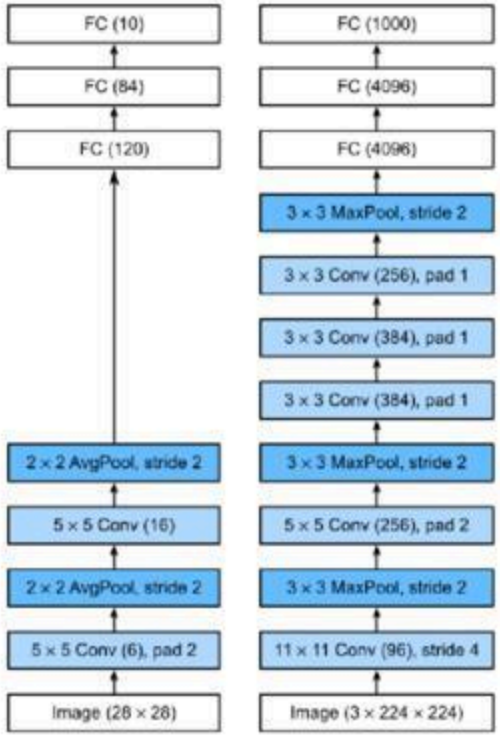
AlexNet 的启示

AlexNet是更大、更深的LeNet

- 80x参数量，250x计算量
- 模型**容量提升**：解决更复杂的问题

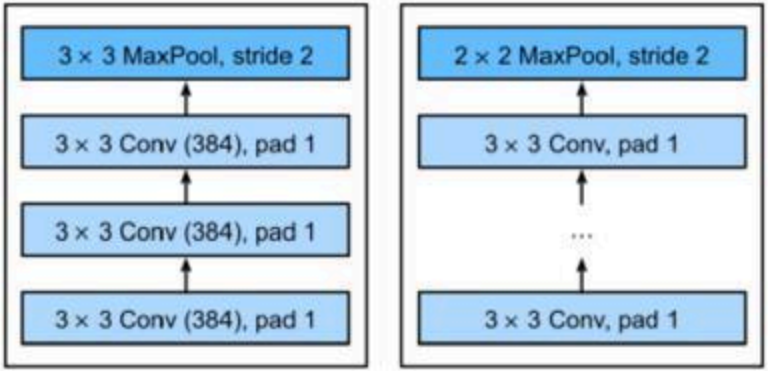
假如将AlexNet做得更大、更深呢?

- 全连接层？得不偿失（回顾参数表）
  - 参数量大，丢失空间信息
- 卷积层：如何组织?



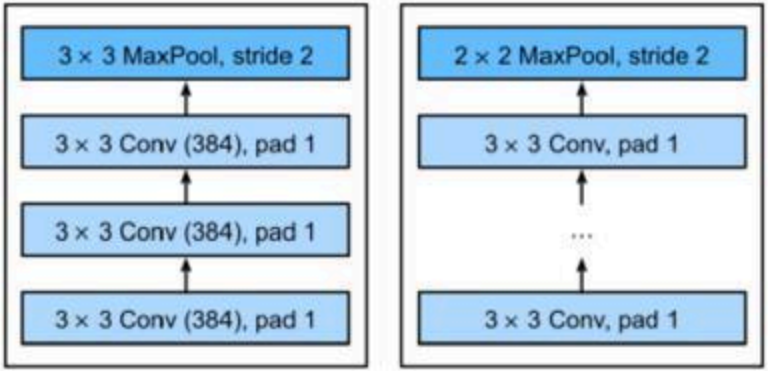
VGG：模块单元

VGG：以**模块**作为基本单元，仿照AlexNet卷积序列



VGG：模块单元

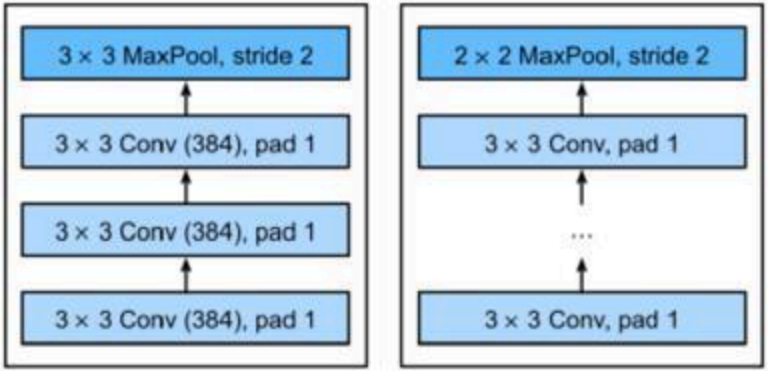
VGG：以**模块**作为基本单元，仿照AlexNet卷积序列



- **深，但窗口窄**的网络性能更佳：
  - 3x3卷积核、填充1：保持宽、高
  - 2x2池化窗口、步幅2：特征图分辨率减半

VGG：模块单元

VGG：以**模块**作为基本单元，仿照AlexNet卷积序列



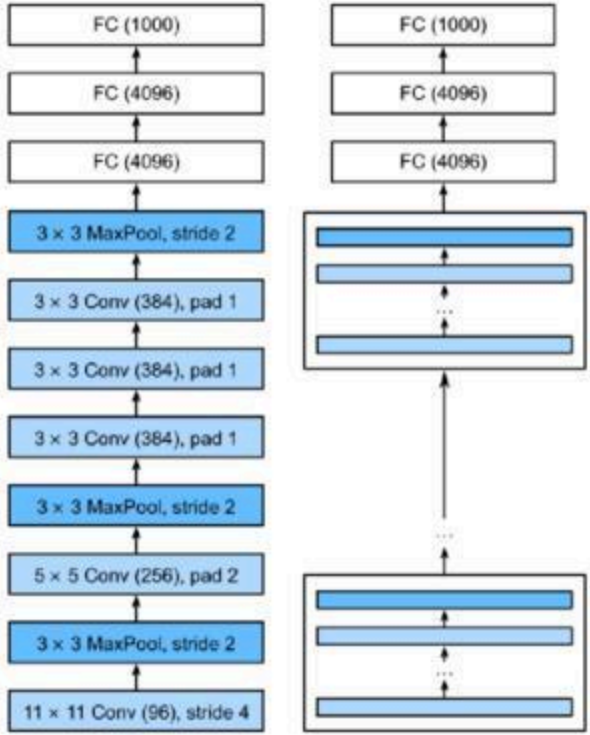
- **深，但窗口窄**的网络性能更佳：
  - 3x3卷积核、填充1：保持宽、高
  - 2x2池化窗口、步幅2：特征图分辨率减半
- 牛津大学的视觉几何组（Visual Geometry Group）

VGG：架构

架构：**串联**重复的模块单元

- VGG-16、VGG-19

最后仍然接全连接层



架构演化

LeNet：最早的卷积神经网络

- 卷积 + 池化层、全连接层

AlexNet：更大、更深的LeNet

- ReLU、Dropout、最大池化、数据增广

VGG：更大、更深的AlexNet

- 串联重复的模块单元

架构演化

LeNet：最早的卷积神经网络

- 卷积 + 池化层、全连接层

AlexNet：更大、更深的LeNet

- ReLU、Dropout、最大池化、数据增广

VGG：更大、更深的AlexNet

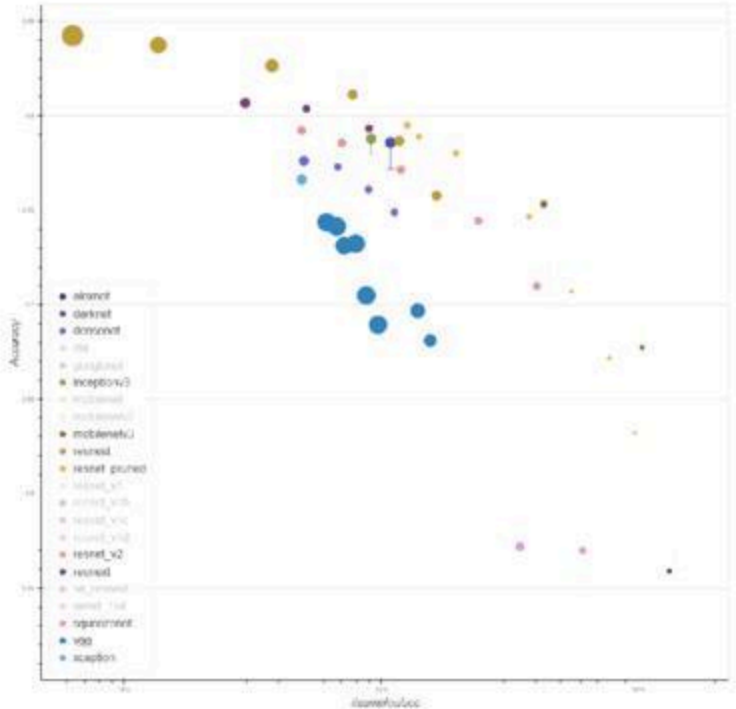
- 串联重复的模块单元

下一步：更大、更深的VGG?

- 的确有做深网络的尝试，但遇到了**瓶颈**
- 各种**模块开发**如百花齐放

GluonCV Model Zoo：分类

cv.gluon.ai/model_zoo/classification



实验：VGG

小结：VGG

- VGG架构：串联重复的模块单元来构造深度卷积网络
- 超参数：卷积块的个数、（超）参数

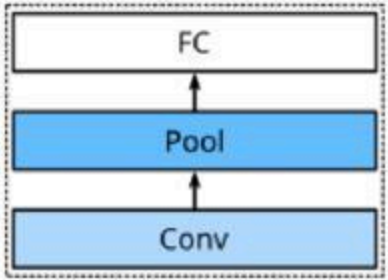
网络中的网络 NiN

设计模式

LeNet、AlexNet和VGG

- 卷积层 + 池化层：提取空间结构信息
- 全连接层：特征空间变换，信息投影

AlexNet和VGG：**加大**、**加深**以上两个模块

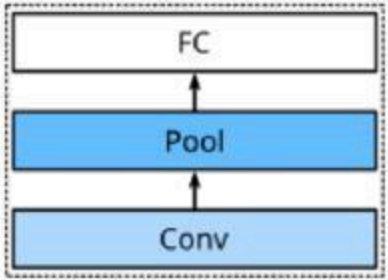


设计模式

LeNet、AlexNet和VGG

- 卷积层 + 池化层：提取空间结构信息
- 全连接层：特征空间变换，信息投影

AlexNet和VGG：**加大、加深**以上两个模块



NiN将串联的每个模块看成**完整网络**

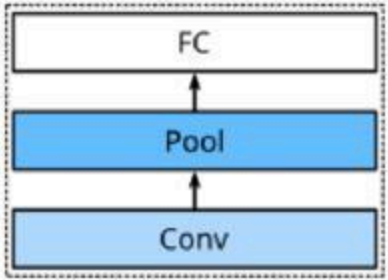
- 架构设计：递归构造

设计模式

LeNet、AlexNet和VGG

- 卷积层 + 池化层：提取空间结构信息
- 全连接层：特征空间变换，信息投影

AlexNet和VGG：**加大、加深**以上两个模块



NiN将串联的每个模块看成**完整网络**

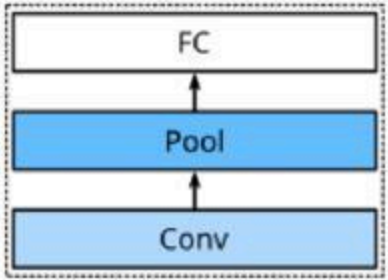
- 架构设计：递归构造
- 简单方案：中间模块使用全连接
  - 问题：特征的**空间结构会丢失**

设计模式

LeNet、AlexNet和VGG

- 卷积层 + 池化层：提取空间结构信息
- 全连接层：特征空间变换，信息投影

AlexNet和VGG：**加大、加深**以上两个模块



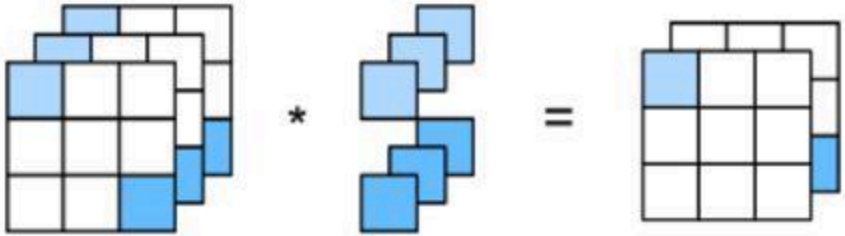
NiN将串联的每个模块看成**完整网络**

- 架构设计：递归构造
- 简单方案：中间模块使用全连接
  - 问题：特征的**空间结构会丢失**
- 创新方案：使用**1x1卷积层**

NiN：模块

回顾：1x1卷积等价于全连接

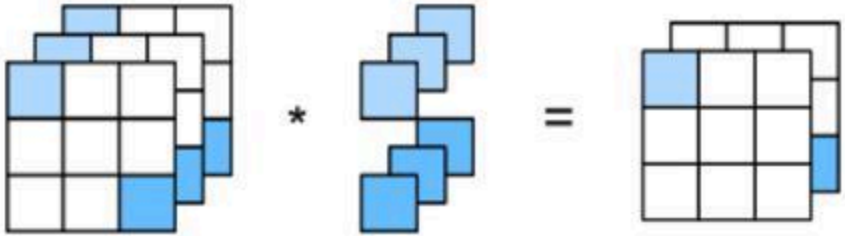
- 每个像素的**通道维度**上可以看成感知机



NiN：模块

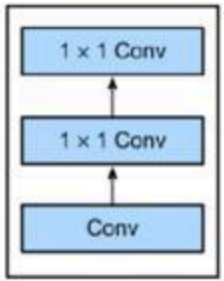
回顾：1x1卷积等价于全连接

- 每个像素的**通道维度**上可以看成感知机



卷积层 + 两个1x1卷积层

- 两个ReLU激活函数：**加强非线性**
- 步幅1，无填充：**形状不变**



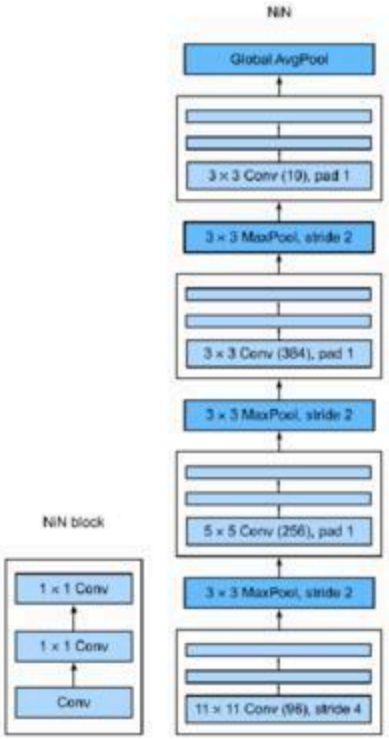
NiN：架构

交替使用：NiN模块、步幅2的最大池化层

- 每个模块等价于完整网络
- 逐步缩小特征图、增多通道

无全连接层

- 实际上每个模块都有全连接：1x1卷积
- 最后输出：使用全局平均池化层
  - 输出通道数 = 类别数



全连接：计算之源

卷积层的参数量（$\mathbf{W}$）：$c_o \times c_i \times k^2$

- k：卷积核大小；c_i, c_o：输入、输出通道数

全连接：计算之源

卷积层的参数量（$\mathbf{W}$）：$c_o \times c_i \times k^2$

- k：卷积核大小；c_i, c_o：输入、输出通道数

卷积层之后的第一个全连接层

- LeNet：$16 \times 5 \times 5 \times 120 = 48$ k
- AlexNet：$256 \times 5 \times 5 \times 4096 = 26$ M
- VGG：$512 \times 7 \times 7 \times 4096 = 102$ M

全连接：计算之源

卷积层的参数量（$\mathbf{W}$）：$c_o \times c_i \times k^2$

- k：卷积核大小；c_i, c_o：输入、输出通道数

卷积层之后的第一个全连接层

- LeNet：$16 \times 5 \times 5 \times 120 = 48$ k
- AlexNet：$256 \times 5 \times 5 \times 4096 = 26$ M
- VGG：$512 \times 7 \times 7 \times 4096 = 102$ M

参数量决定计算、存储：限制网络深度

- NiN标志着网络**架构调优竞赛**的开始
  - 给定硬件平台：参数量恒定，只能优化架构设计

实验：NiN

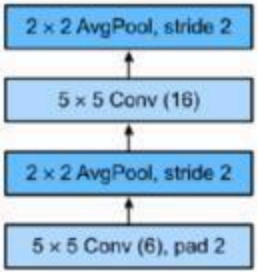
小结：NiN

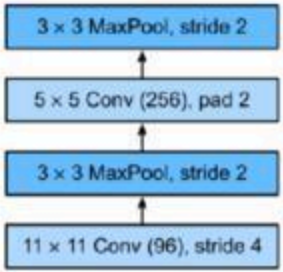
- NiN模块：等价于完整网络
  - 卷积层 + 两个1x1卷积层
- 使用1x1卷积替代全连接
  - 模块参数更少：单个模块计算量可控；并且能够把网络做深

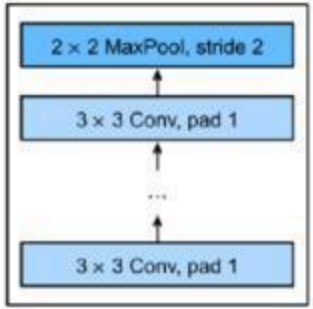
含并行连结的网络 GoogLeNet

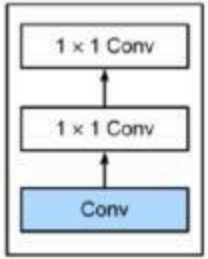
卷积层超参数：选择困难症

LeNet、AlexNet、VGG、NiN





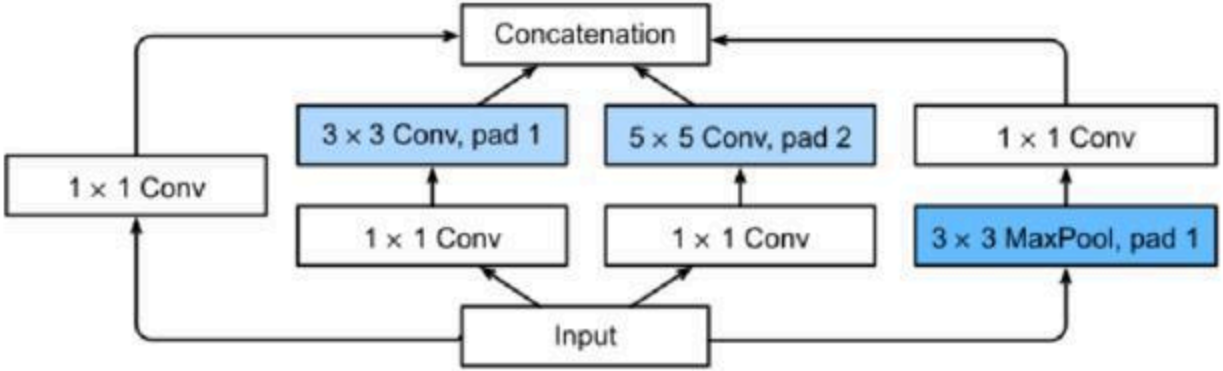




- 1x1、3x3、5x5、11x11?
- 平均、最大池化?
- 多少个1x1?

Inception 模块：“我全都要”

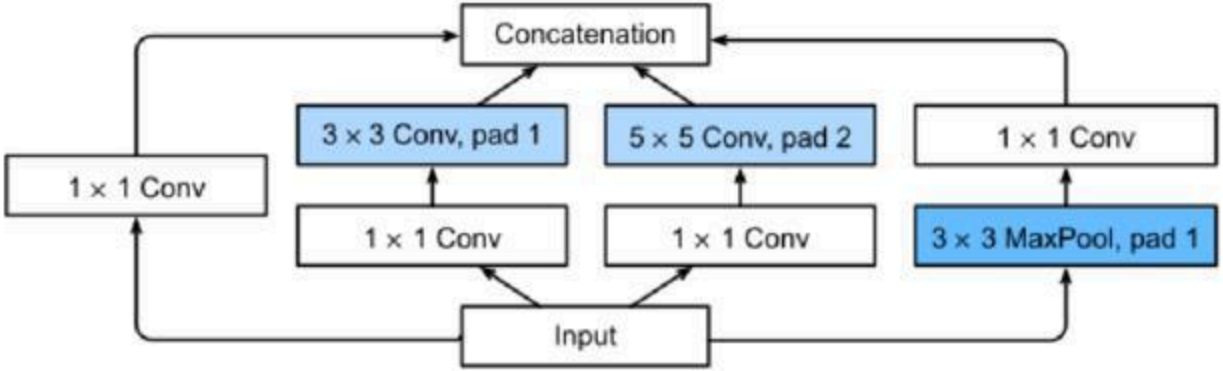
工程美学的典范：使用不同大小卷积核的**组合**



- 4条路径：从**不同视角**提取信息
  - 不同卷积核大小：识别不同尺寸的细节模式
  - 每条路径输出：形状不变；通道数是超参数

Inception 模块：“我全都要”

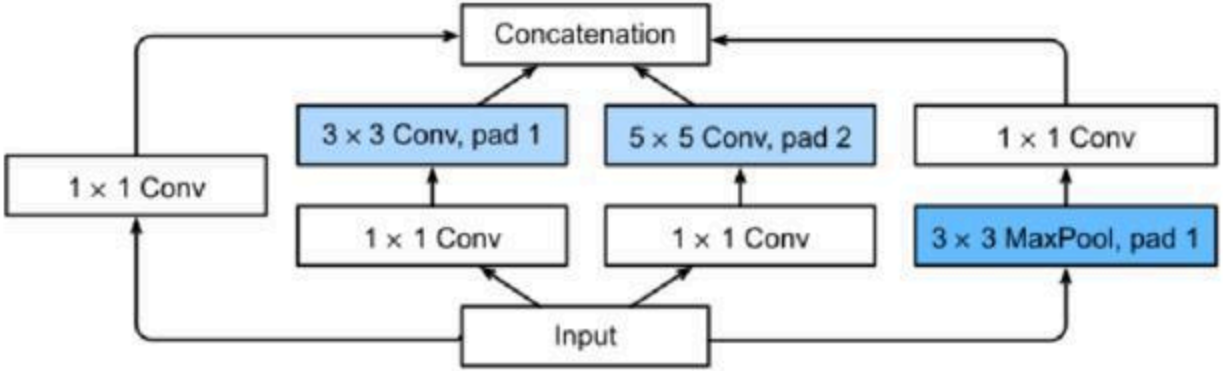
工程美学的典范：使用不同大小卷积核的**组合**



- 4条路径：从**不同视角**提取信息
  - 不同卷积核大小：识别不同尺寸的细节模式
  - 每条路径输出：形状不变；通道数是超参数
- 合并所有路径的输出通道：（不同视角的）**模式组合**

Inception 模块：“我全都要”

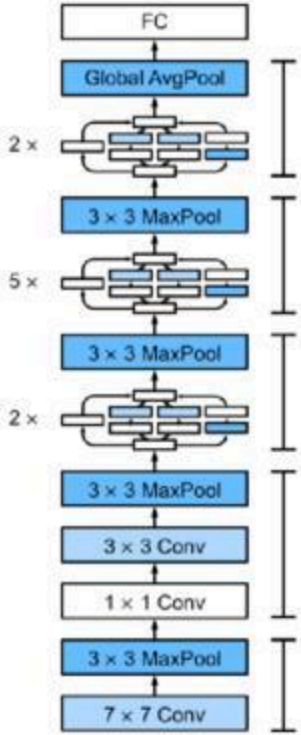
工程美学的典范：使用不同大小卷积核的**组合**



- 4条路径：从**不同视角**提取信息
  - 不同卷积核大小：识别不同尺寸的细节模式
  - 每条路径输出：形状不变；通道数是超参数
- 合并所有路径的输出通道：（不同视角的）**模式组合**

GoogLeNet

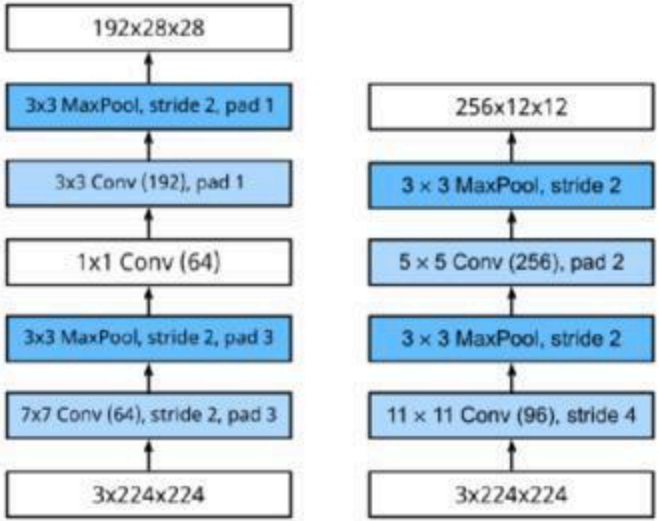
- 5个“阶段”
- 9个Inception模块



阶段 1，2

与AlexNet（右）对比

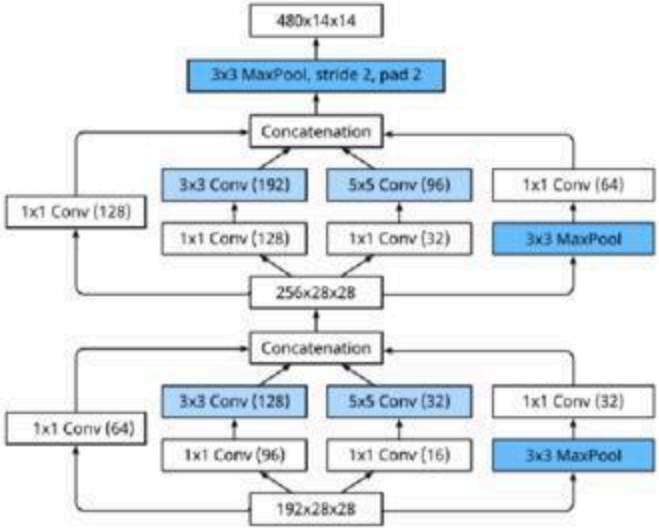
- 更小窗口：减少计算量
- 通道数增长速度更平稳



阶段 3

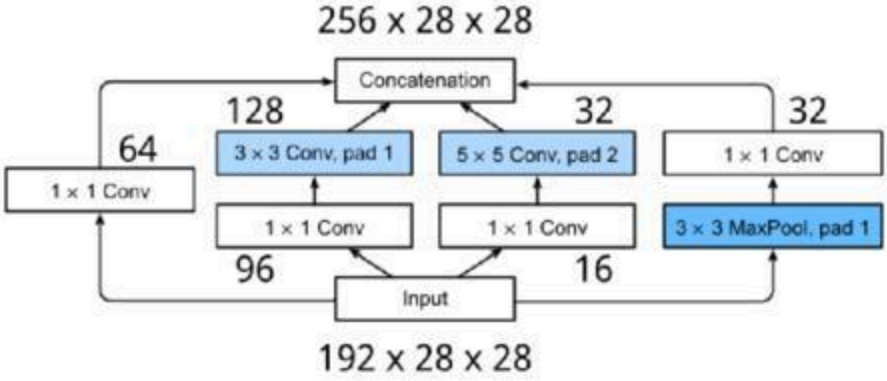
核心：2x Inception 模块

- 通道数增加：递进的复杂度



Inception 模块：S3B1

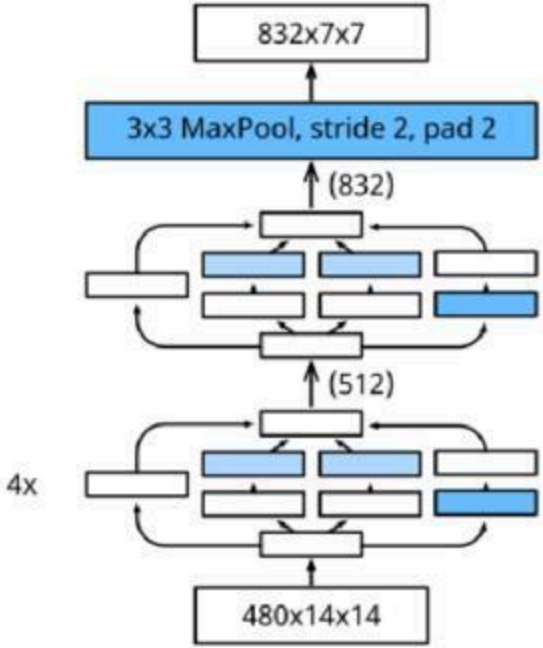
第一个Inception 模块的通道数图示

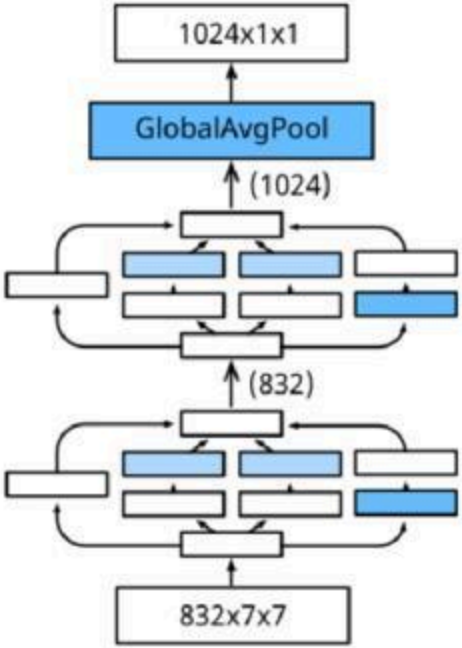


- 每条路径输出：形状不变；**通道数是超参数**
  - 1x1卷积：减少分支通道数、控制复杂度
  - 通道数：与卷积核大小成反比

	#参数	FLOPS
Inception	0.16 M	128 M
3x3 Conv	0.44 M	346 M
5x5 Conv	1.22 M	963 M

阶段 4，5


832x7x7
3x3 MaxPool, stride 2, pad 2
(832)
(512)
4x
480x14x14


1024x1x1
GlobalAvgPool
(1024)
(832)
832x7x7

Inception：变种

Inception-BN (V2)：使用batch normalization

Inception：变种

Inception-BN (V2)：使用batch normalization

Inception-V3：模块调优

- 替换5x5为两个3x3
- 替换5x5为1x7和7x1
- 替换3x3为1x3和3x1

Inception：变种

Inception-BN (V2)：使用batch normalization

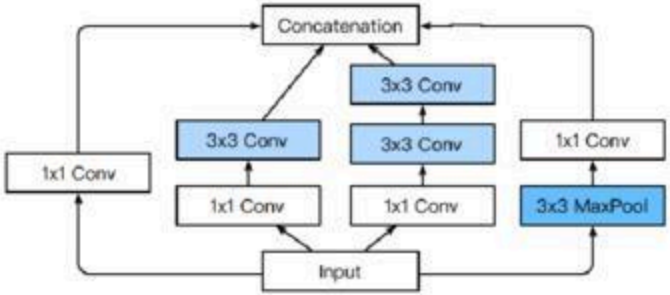
Inception-V3：模块调优

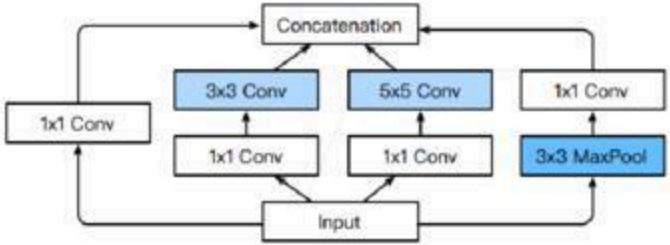
- 替换5x5为两个3x3
- 替换5x5为1x7和7x1
- 替换3x3为1x3和3x1

Inception-V4：使用残差连接

Inception-V3: S3

替换5x5为两个3x3

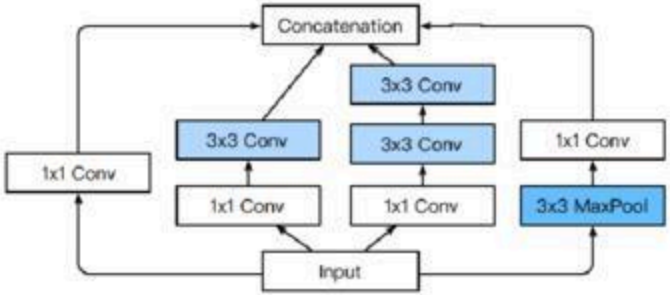


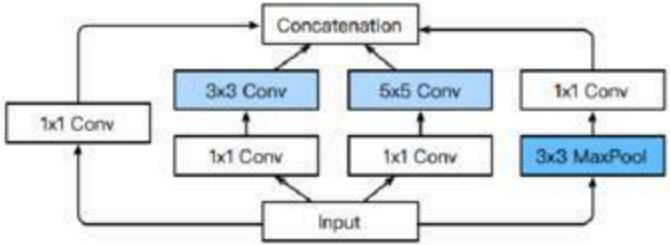


- 参数：$25 \rightarrow 2 \times 9 = 18$

Inception-V3: S3

替换5x5为两个3x3

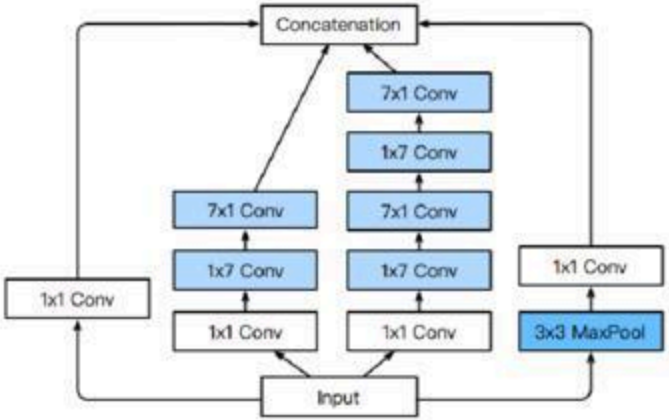


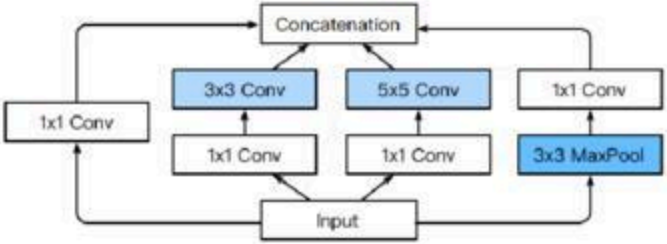


- 参数：$25 \rightarrow 2 \times 9 = 18$
- **感受野不变**：两个3x3相邻近似于5x5
  - 提示：从两层计算依赖关系来看，第二个3x3相当于扩充一圈边界

Inception-V3: S4

替换3x3为1x7和7x1

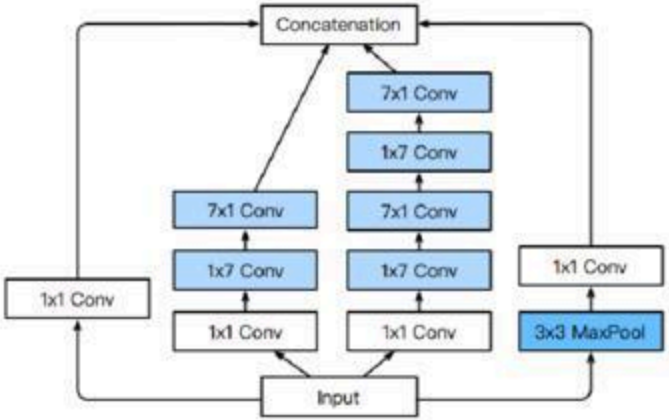


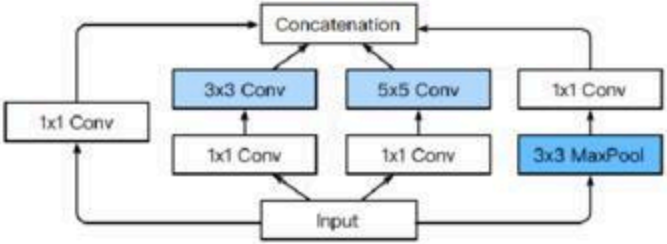


- 参数：$9 \rightarrow 2 \times 7 = 14$

Inception-V3: S4

替换3x3为1x7和7x1

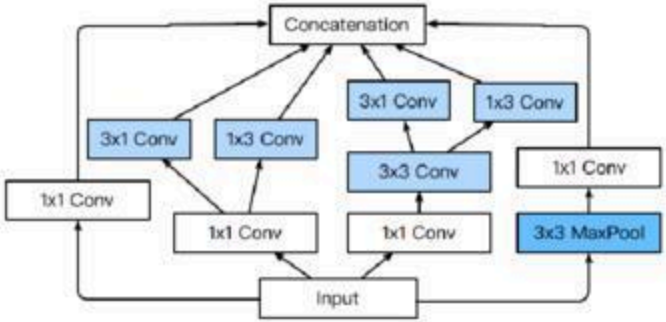


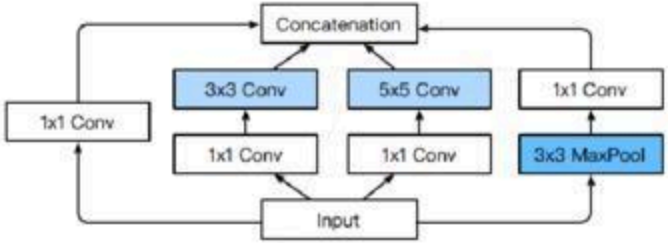


- 参数：$9 \rightarrow 2 \times 7 = 14$
- **感受野变大**：1x7和7x1相邻近似于7x7
- **感受野变形**：水平、垂直的特征

Inception-V3: S5

替换3x3为1x3和3x1（也有替换5x5为两个3x3）

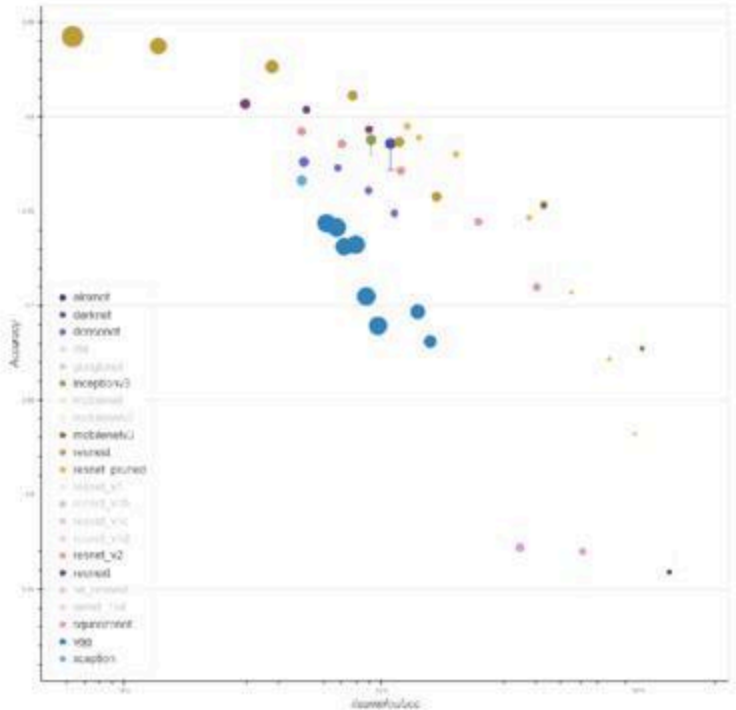




- 参数：$9 \rightarrow 2 \times 3 = 6$

GluonCV Model Zoo：分类

cv.gluon.ai/model_zoo/classification



实验：GoogLeNet

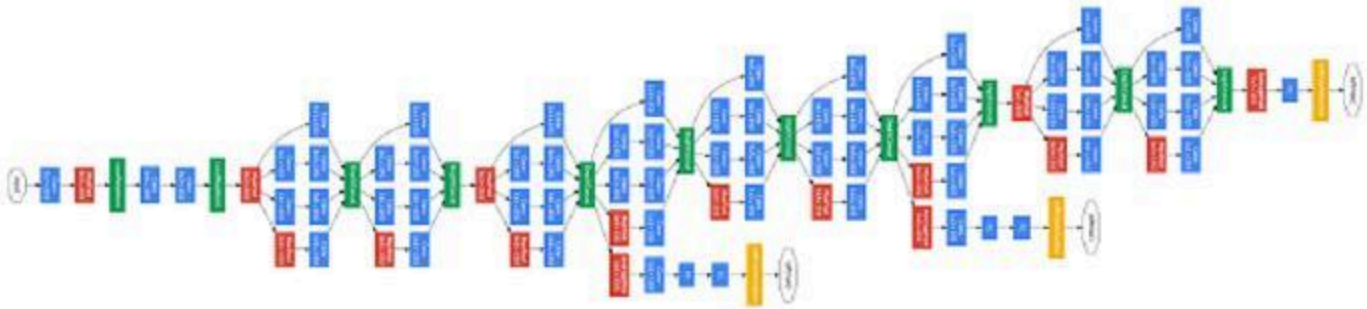
小结：GoogLeNet

Inception模块：4条不同路径

- 从**不同视角**提取信息
- 模型参数少、计算复杂度低

GoogLeNet：9个Inception模块

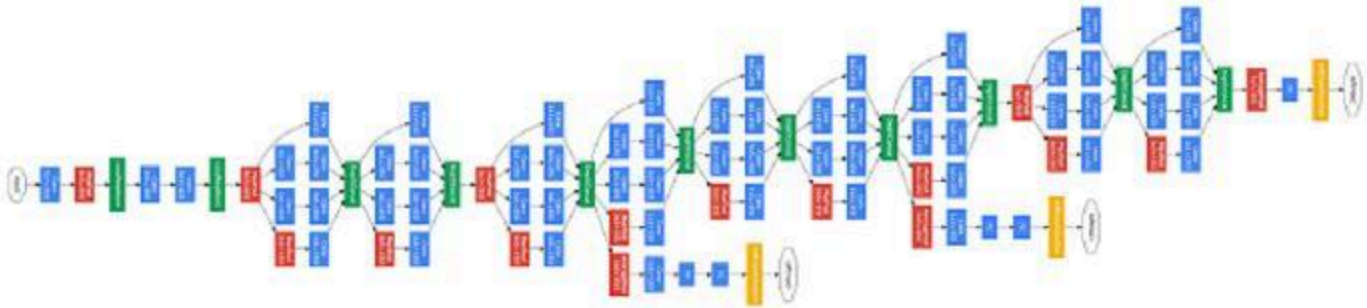
- 第一个达到上百层的网络
- 后续一系列版本改进



批量规范化

深度带来的问题

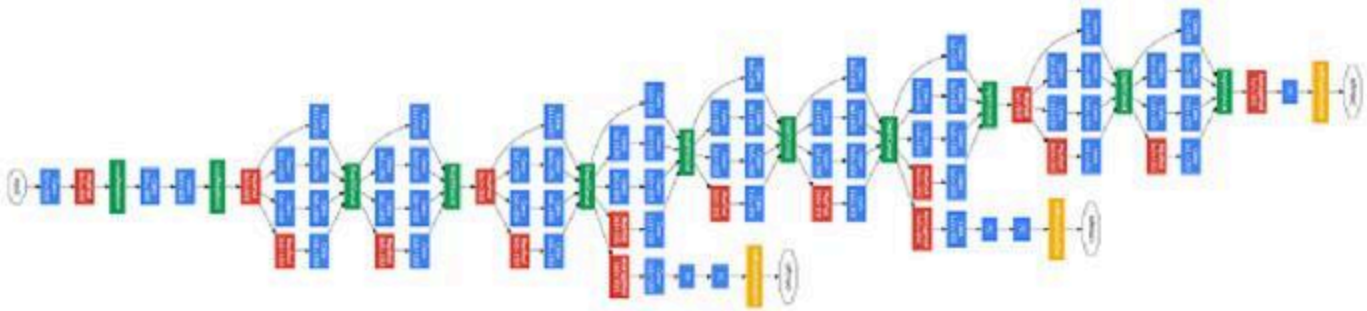
网络非常深：计算**依赖链过长**，层间信息传递变难，导致收敛变慢。想象用桶烧水



损失函数在最后：后面层**参数关联**直接、训练较快

深度带来的问题

网络非常深：计算**依赖链过长**，层间信息传递变难，导致收敛变慢。想象用桶烧水



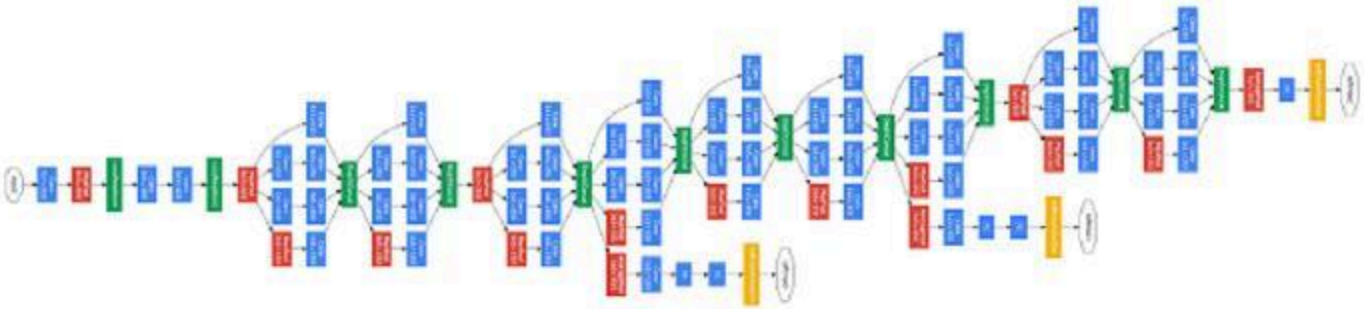
损失函数在最后：后面层**参数关联**直接、训练较快

数据在最前：前面层易受**数据影响**、训练较慢

- 前面层的变化导致全都跟着变，例如异常数据

深度带来的问题

网络非常深：计算**依赖链过长**，层间信息传递变难，导致收敛变慢。想象用桶烧水



损失函数在最后：后面层**参数关联**直接、训练较快

数据在最前：前面层易受**数据影响**、训练较慢

- 前面层的变化导致全都跟着变，例如异常数据

能否避免层间**学习难度**剧烈变化？

- 同一套优化算法：同一种优化速率

标准化中间层

回忆：输出、梯度可以看成**随机变量**

- 中间层变量分布偏移：阻碍网络收敛

只要**控制统计量**，就能避免数据分布的剧烈变化

标准化中间层

回忆：输出、梯度可以看成**随机变量**

- 中间层变量分布偏移：阻碍网络收敛

只要**控制统计量**，就能避免数据分布的剧烈变化

批量标准化：γ, β是可学习参数

$$x_i = \gamma \frac{x_i - \mu}{\sigma} + \beta$$

- γ：拉伸参数；β：偏移参数

标准化中间层

回忆：输出、梯度可以看成**随机变量**

- 中间层变量分布偏移：阻碍网络收敛

只要**控制统计量**，就能避免数据分布的剧烈变化

批量标准化：γ, β是可学习参数

$$x_i = \gamma \frac{x_i - \mu}{\sigma} + \beta$$

- γ：拉伸参数；β：偏移参数
- 想象：按生产线批次控制零件质量（符号）标准，便于组装产品

批量规范化

估计批量的均值、方差：

$$\hat{\mu}_B = \frac{1}{|B|}\sum_i x_i$$

$$\hat{\sigma}_B^2 = \frac{1}{|B|}\sum_i (x_i - \hat{\mu}_B)^2 + \epsilon$$

- 注意：小常数$\epsilon > 0$保证分母不为0

批量规范化

估计批量的均值、方差：

$$\hat{\mu}_B = \frac{1}{|B|}\sum_i x_i$$

$$\hat{\sigma}_B^2 = \frac{1}{|B|}\sum_i (x_i - \hat{\mu}_B)^2 + \epsilon$$

- 注意：小常数$\epsilon > 0$保证分母不为0
- 优化中各种噪声源通常会导致训练更快、较少过拟合
  - 可以看成**正则化**的一种形式

批量规范化

估计批量的均值、方差：

$$\hat{\mu}_B = \frac{1}{|B|}\sum_i x_i$$

$$\hat{\sigma}_B^2 = \frac{1}{|B|}\sum_i (x_i - \hat{\mu}_B)^2 + \epsilon$$

- 注意：小常数$\epsilon > 0$保证分母不为0
- 优化中各种噪声源通常会导致训练更快、较少过拟合
  - 可以看成**正则化**的一种形式
- 批量选择变得更重要
  - 批量不能为1：减去均值后，每个隐藏单元输出0
  - 批量规范化最适用 50 ~ 100 范围的**中等批量大小**

批量规范化

估计批量的均值、方差：

$$\hat{\mu}_B = \frac{1}{|B|}\sum_i x_i$$

$$\hat{\sigma}_B^2 = \frac{1}{|B|}\sum_i (x_i - \hat{\mu}_B)^2 + \epsilon$$

- 注意：小常数$\epsilon > 0$保证分母不为0
- 优化中各种噪声源通常会导致训练更快、较少过拟合
  - 可以看成**正则化**的一种形式
- 批量选择变得更重要
  - 批量不能为1：减去均值后，每个隐藏单元输出0
  - 批量规范化最适用 50 ~ 100 范围的**中等批量大小**

≡ 测试模式：使用训练阶段的**移动平均**估算（整个数据集的均值、方差）

批量规范化层

可学习参数：γ, β

$$x_i = \gamma \frac{x_i - \hat{\mu}_B}{\hat{\sigma}_B} + \beta$$

- 作用在：**全连接、卷积层**的输出，激活函数之前

批量规范化层

可学习参数：γ, β

$$x_i = \gamma \frac{x_i - \hat{\mu}_B}{\hat{\sigma}_B} + \beta$$

- 作用在：**全连接、卷积层**的输出，激活函数之前

- 全连接层（2维张量）：作用在**特征维**
- 卷积层（4维张量）：作用在**通道维**

批量规范化在做什么?

最初论文**猜测**：减少内部协方差的偏移

- 但其实与协变量偏移的严格定义不同

批量规范化在做什么?

最初论文**猜测**：减少内部协方差的偏移

- 但其实与协变量偏移的严格定义不同

后续工作认为：给批量加入噪音，进而增强模型**鲁棒性**

$$x_i = \gamma \frac{x_i - \hat{\mu}_B}{\hat{\sigma}_B} + \beta$$

- $\hat{\mu}_B, \hat{\sigma}_B$：随机偏移、缩放
- 因此没必要与丢弃法混合使用

批量规范化在做什么?

最初论文**猜测**：减少内部协方差的偏移

- 但其实与协变量偏移的严格定义不同

后续工作认为：给批量加入噪音，进而增强模型**鲁棒性**

$$x_i = \gamma \frac{x_i - \hat{\mu}_B}{\hat{\sigma}_B} + \beta$$

- $\hat{\mu}_B, \hat{\sigma}_B$：随机偏移、缩放
- 因此没必要与丢弃法混合使用

深度学习论文的特点：理论基本靠猜

- 先有实践发现，然后**倒推**、**猜测**可能的原因

实验：批量规范化

小结：批量规范化

- 批量规范化：估计批量的均值、方差
  - 学习出合适的偏移、缩放
- 可以加快收敛速度，但一般不改变模型精度

残差网络 ResNet

网络可以无限加深吗?

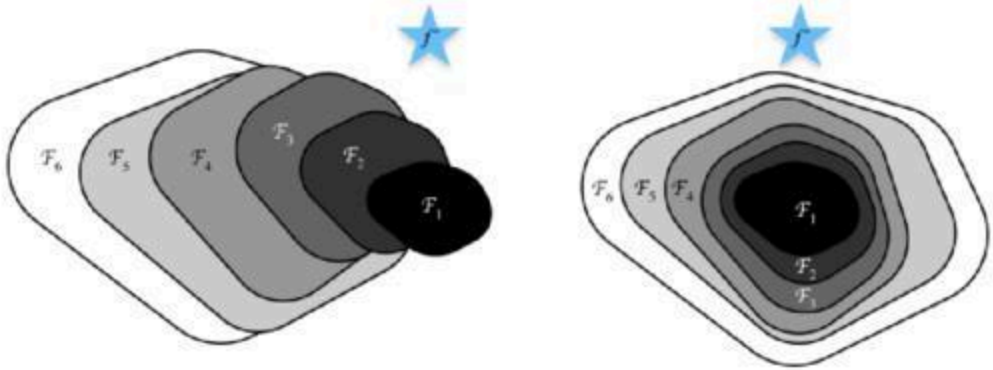
回顾：似乎将网络做深就能提升效能。那么有极限吗?

- 越深的网络：可表示出**更多函数种类**

网络可以无限加深吗?

回顾：似乎将网络做深就能提升效能。那么有极限吗?

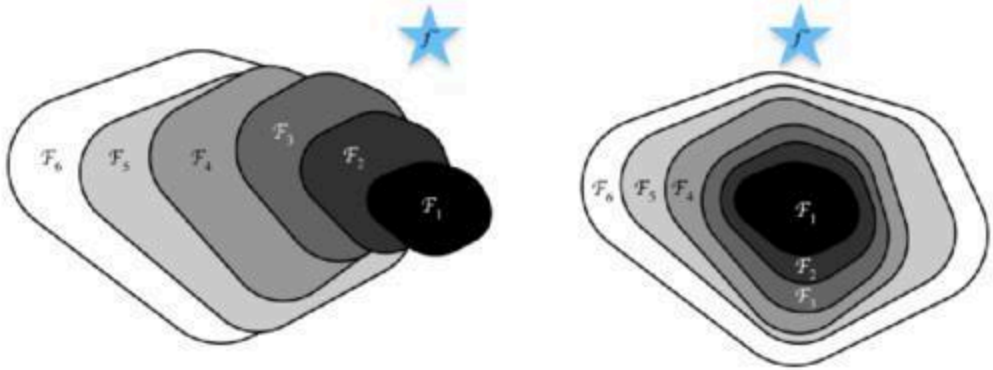
- 越深的网络：可表示出**更多函数种类**
- 将所有可表示的函数种类看成集合：最优函数不一定能通过**扩充函数类**涵盖



网络可以无限加深吗?

回顾：似乎将网络做深就能提升效能。那么有极限吗?

- 越深的网络：可表示出**更多函数种类**
- 将所有可表示的函数种类看成集合：最优函数不一定能通过**扩充函数类**涵盖



- 扩充得到的函数类：应该**严格包含**原函数类
  - 将新添加的层训练成**恒等映射**，新、原模型将同等有效
  - $\forall t, f_2(f_1(t)) = f_1(t) \Rightarrow f_2(x) = x$

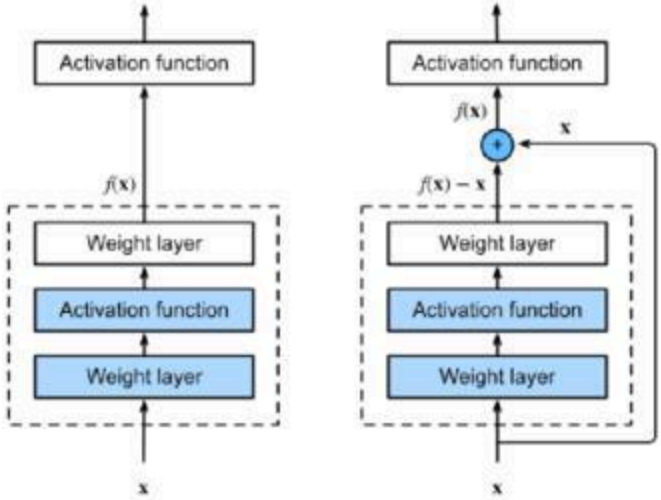
残差

虚线：串联一个模块

- 左边：假设理想映射是$f(x)$
- 右边：拟合目标转换成$f(x)-x$

残差：只需学习增量$\Delta = f(x)-x$

- 由函数类讨论：$f(x)=x$
  - 故$\Delta \approx 0$
  - 数值计算比较稳定



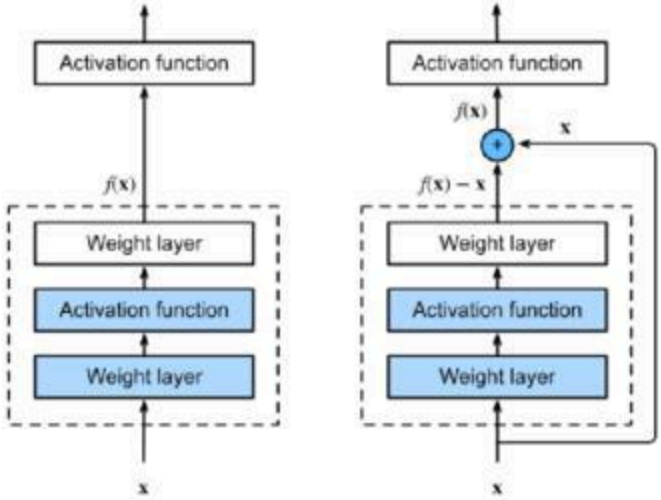
残差

虚线：串联一个模块

- 左边：假设理想映射是$f(x)$
- 右边：拟合目标转换成$f(x)-x$

残差：只需学习增量$\Delta = f(x)-x$

- 由函数类讨论：$f(x)=x$
  - 故$\Delta \approx 0$
  - 数值计算比较稳定



实线旁路：数据可以**跨层直接传播**

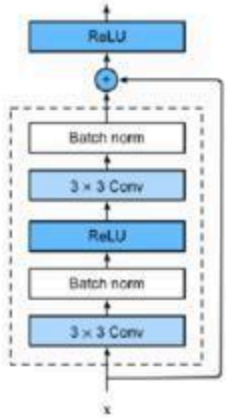
- 避免因层数过多而丢失信息

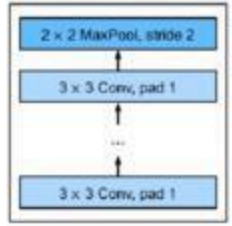
ResNet：模块

类似VGG的3x3卷积层设计

- 3x3卷积、BN、ReLU

数据旁路：接在最后激活之前



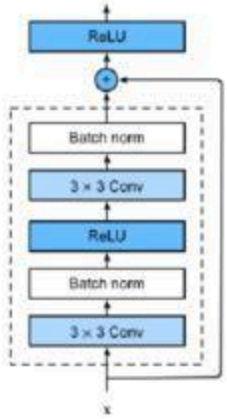


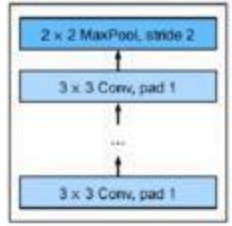
ResNet：模块

类似VGG的3x3卷积层设计

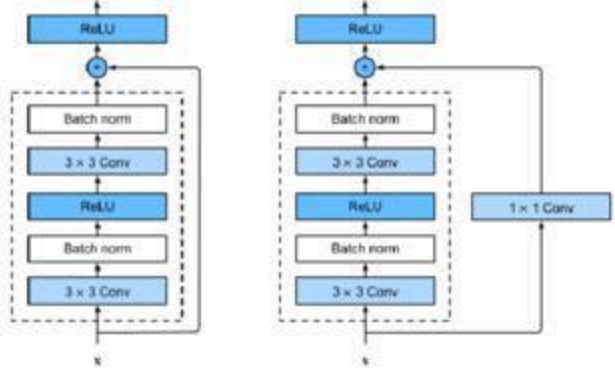
- 3x3卷积、BN、ReLU

数据旁路：接在最后激活之前

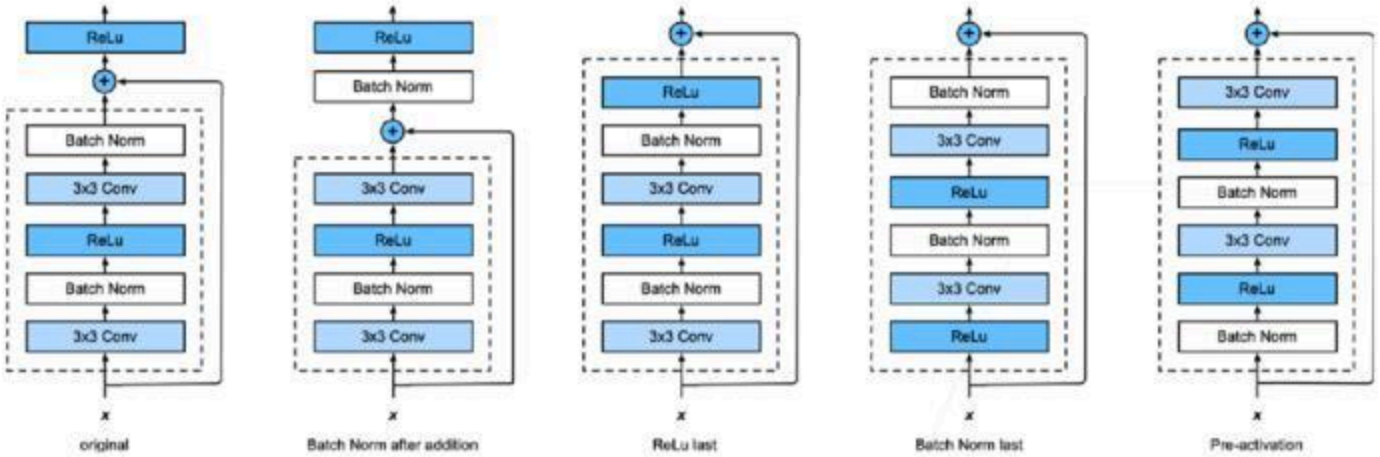




- 1x1卷积：调整成相同的通道数


ReLU
Batch norm
3 × 3 Conv
ReLU
Batch norm
3 × 3 Conv
x
ReLU
Batch norm
3 × 3 Conv
ReLU
Batch norm
3 × 3 Conv
1 × 1 Conv
x

ResNet：模块变种


ReLu
Batch Norm
3x3 Conv
ReLu
Batch Norm
3x3 Conv
x
original
ReLu
Batch Norm
3x3 Conv
ReLu
Batch Norm
3x3 Conv
x
Batch Norm after addition
ReLu
Batch Norm
3x3 Conv
ReLu
Batch Norm
3x3 Conv
x
ReLu last
Batch Norm
3x3 Conv
ReLu
Batch Norm
3x3 Conv
ReLu
x
Batch Norm last
3x3 Conv
ReLu
Batch Norm
3x3 Conv
ReLu
Batch Norm
x
Pre-activation

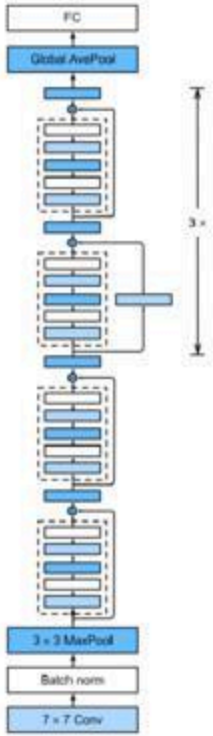
ResNet：架构

类似GoogLeNet架构

- 共18层：ResNet-18

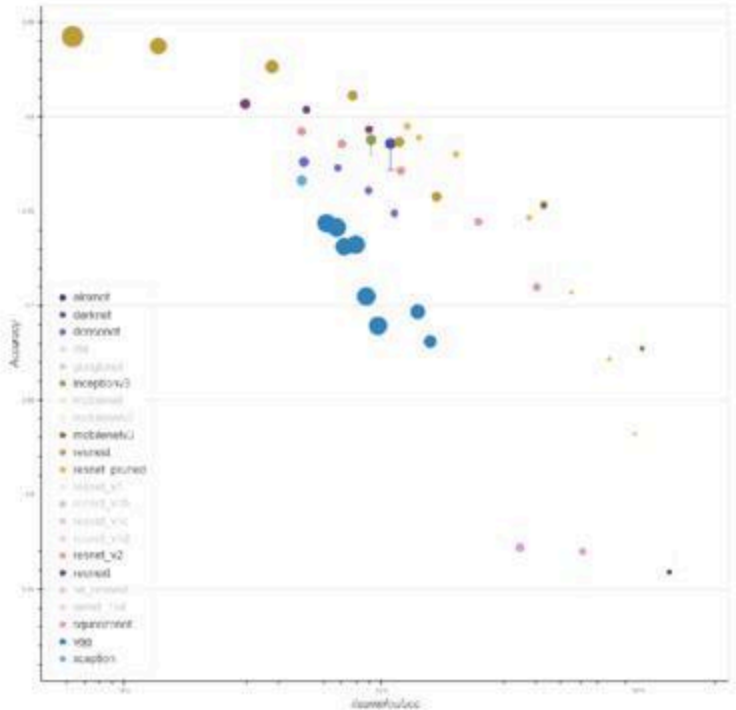
ResNet架构更简单，修改也更方便

- ResNet-152：ImageNet'15识别冠军



GluonCV Model Zoo：分类

cv.gluon.ai/model_zoo/classification



实验：ResNet

小结：ResNet

- ResNet 模块：训练更稳定，跨层直接传递信息
  - 甚至可以训练一千层的网络
- 目前最为流行的架构之一
  - 对之后的架构设计产生了深远影响

Review

本章内容

深度卷积神经网络 AlexNet。使用块的网络 VGG。网络中的网络 NiN。含并行连结的网络 GoogLeNet。批量规范化。残差网络 ResNet。

重点：AlexNet；VGG；NiN；Inception模块；批量规范化；残差网络 ResNet。

难点：Inception变种。

学习目标

- 理解深度卷积神经网络对数据特征的观点：可被学习
- 理解AlexNet的设计特点（丢弃法、ReLU、最大池化、数据增广）、学术贡献（特征工程自动化）
- 理解VGG的设计特点（模块化设计）
- 理解NiN的设计特点（模块等价于完整网络）、创新点（1x1卷积替代全连接）
- 理解Inception模块的设计特点（分支处理、模式组合），及其变种的设计特点
- 理解批量规范化的动机、原理、方法
- 理解ResNet的原理，及其模块的设计特点

问题

简述AlexNet的设计特点、学术贡献。

简述VGG的设计特点。

简述NiN的设计特点、创新点。

简述Inception模块的设计特点，及其变种的设计特点。

简述批量规范化的动机、原理、方法。

简述ResNet的原理，及其模块的设计特点。