经典神经网络模型二

新的结构

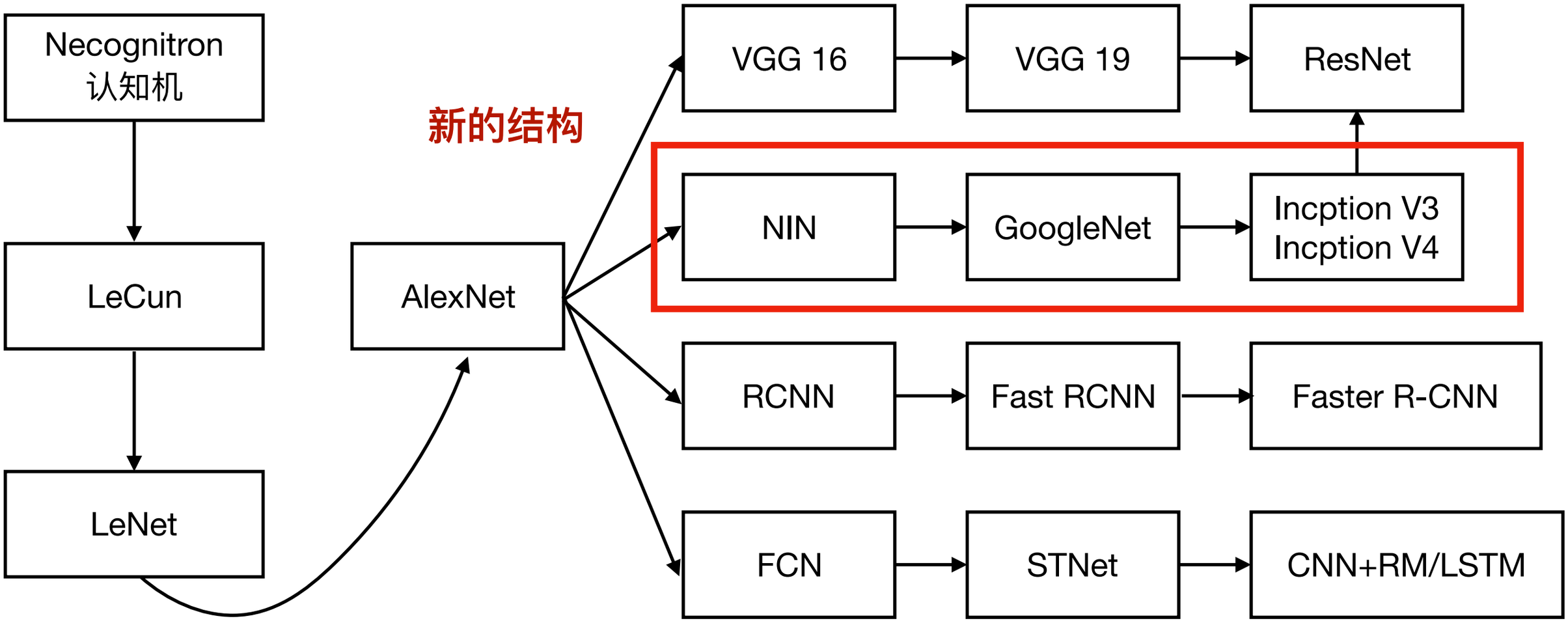
Author:WangQi
Email:wangqikaixin@gmail.com

概览

1. NIN
2. GoogLeNet
3. InceptionNet
4. ResNet

Author:WangQi
Email:wangqikaixin@gmail.com

卷积神经网络结构演化史



Author:WangQi
Email:wangqikaixin@gmail.com

1. NIN

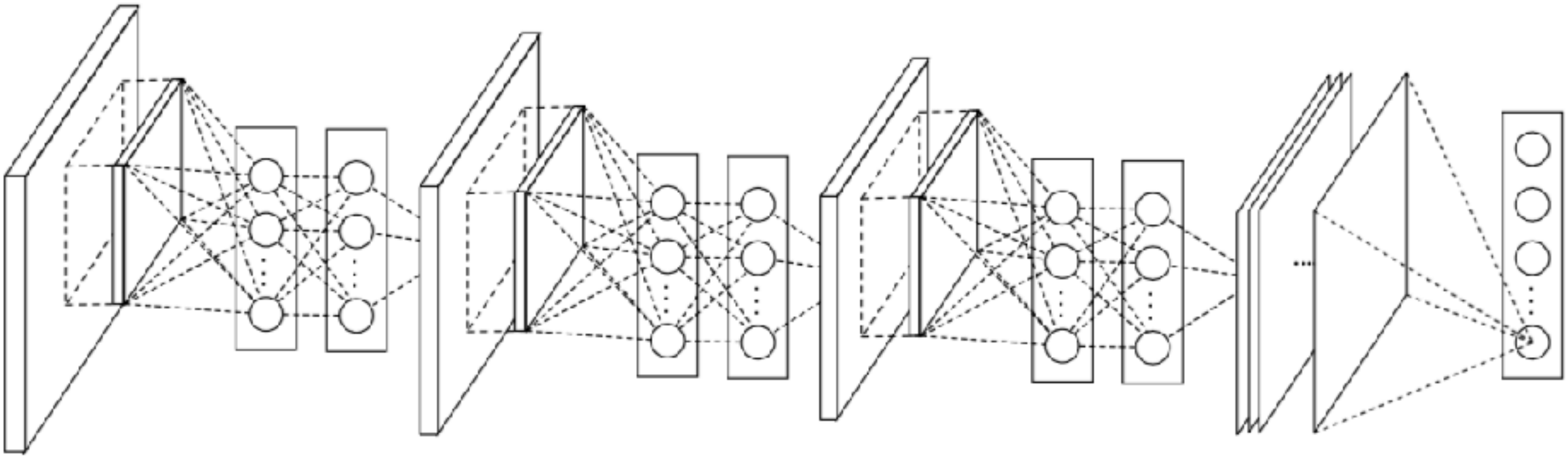
Author:WangQi
Email:wangqikaixin@gmail.com

1.1 简介与背景

Author:WangQi
Email:wangqikaixin@gmail.com

简介

Network In Network（NIN）发表与2013年，入选国际学习表征会议ICLR 2014 优秀论文，其通过少量参数的深度学习模型轻松击败了AlexNet。

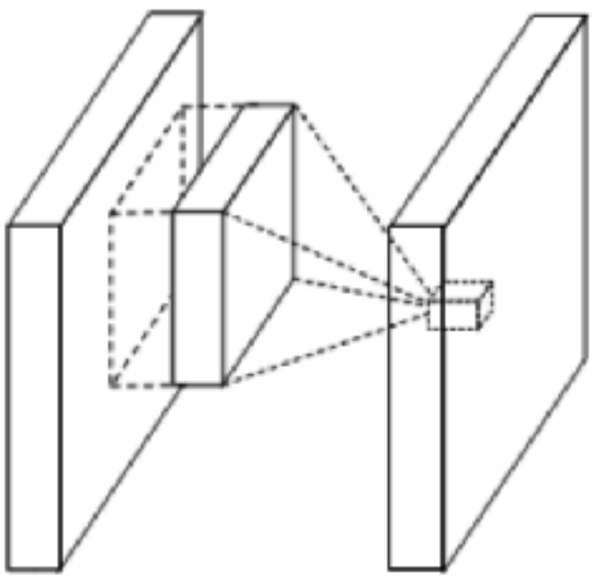


Author:WangQi
Email:wangqikaixin@gmail.com

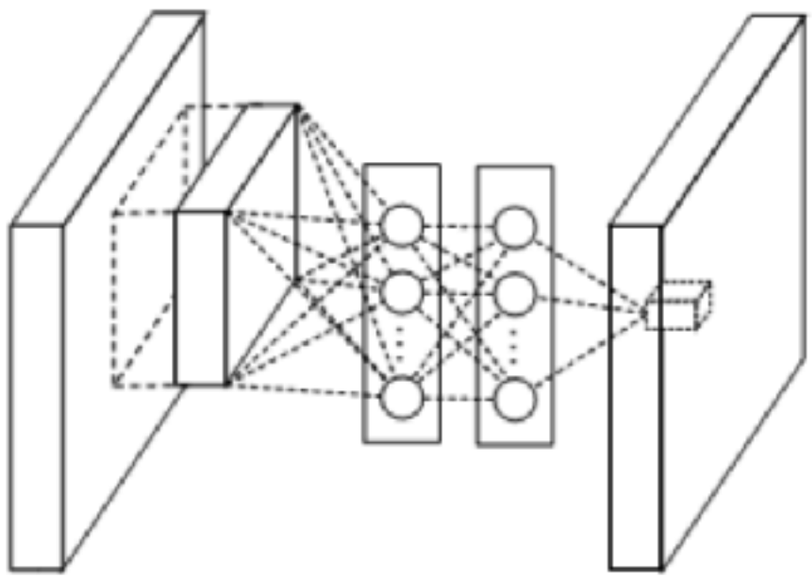
背景

CNN中的卷积核执行卷积的过程是一个广义线性模型（Generalized Liner Model, GLM），能够较好的提取线性可分的流行特征，然而特征一般都存在与非线性流行中，应该使用非线性能力更强的模型进行特征提取。

作者提出使用MLP卷积代替传统卷积：



(a) Linear convolution layer



(b) Mlpconv layer

Author:WangQi
Email:wangqikaixin@gmail.com

1.2 模型结构与主要贡献

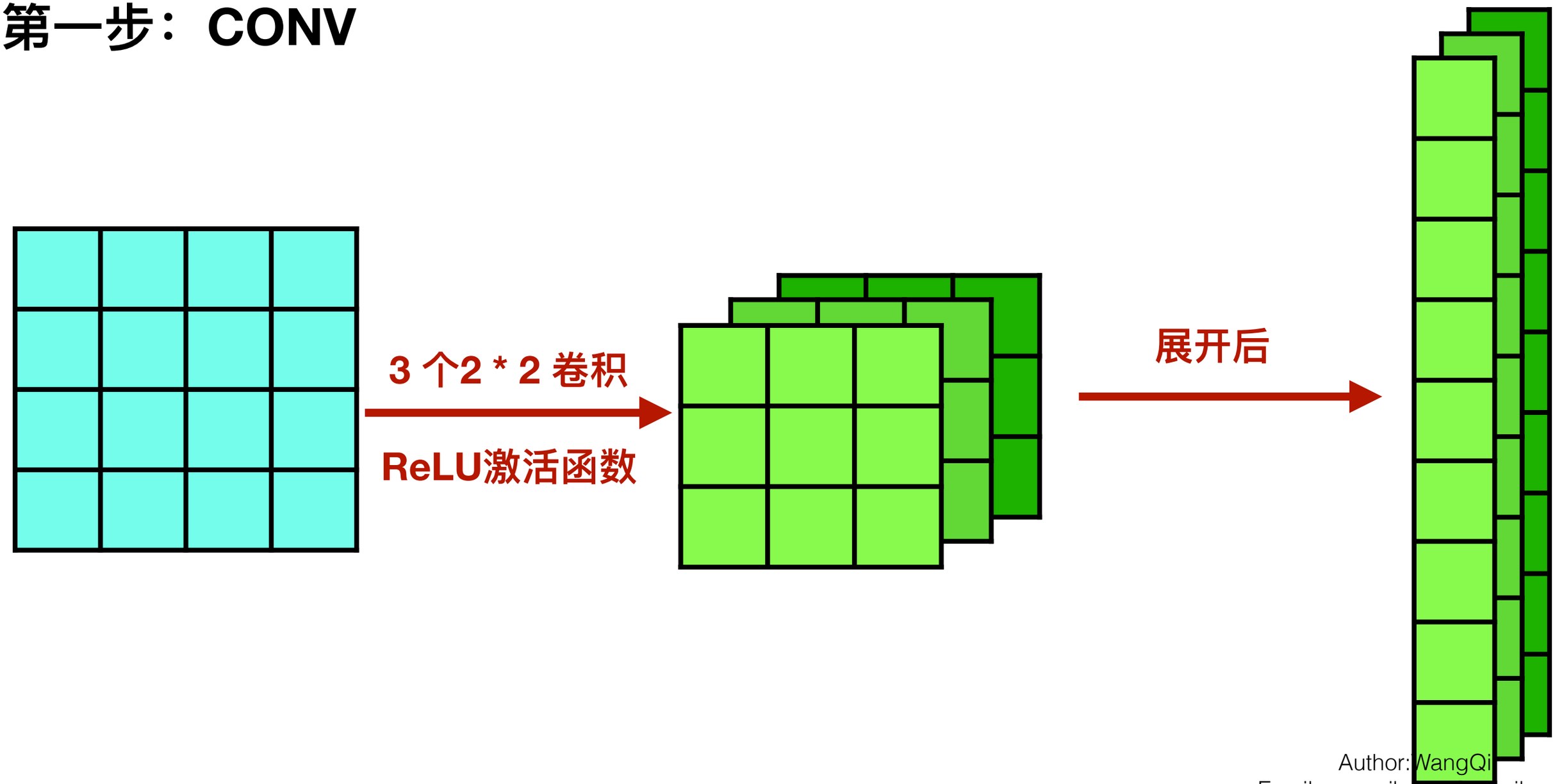
Author:WangQi
Email:wangqikaixin@gmail.com

贡献一：MlpConv

Author:WangQi
Email:wangqikaixin@gmail.com

MlpConv 结构

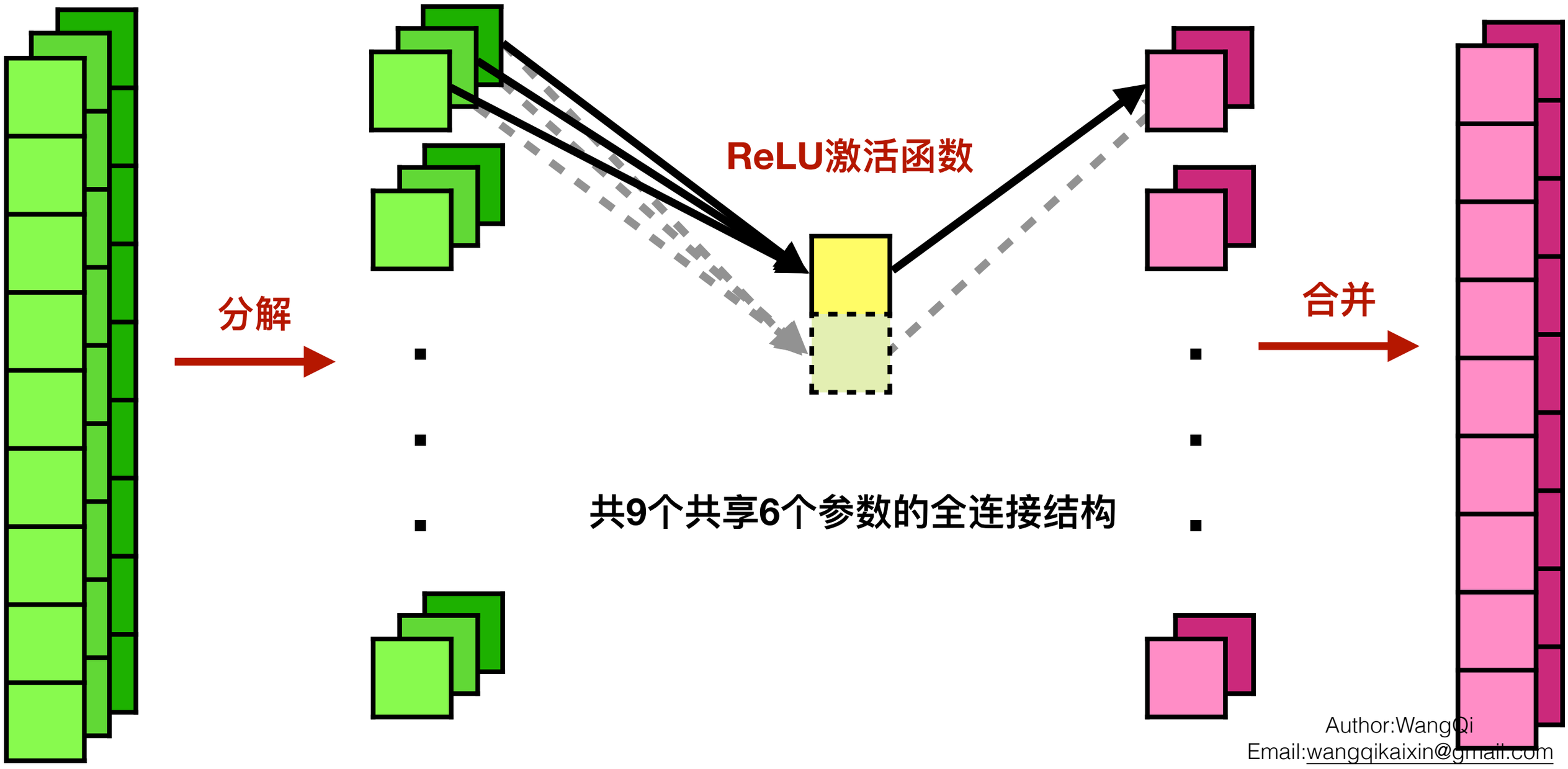
第一步：CONV



Author: WangQi
Email:wangqikaixin@gmail.com

MlpConv 结构

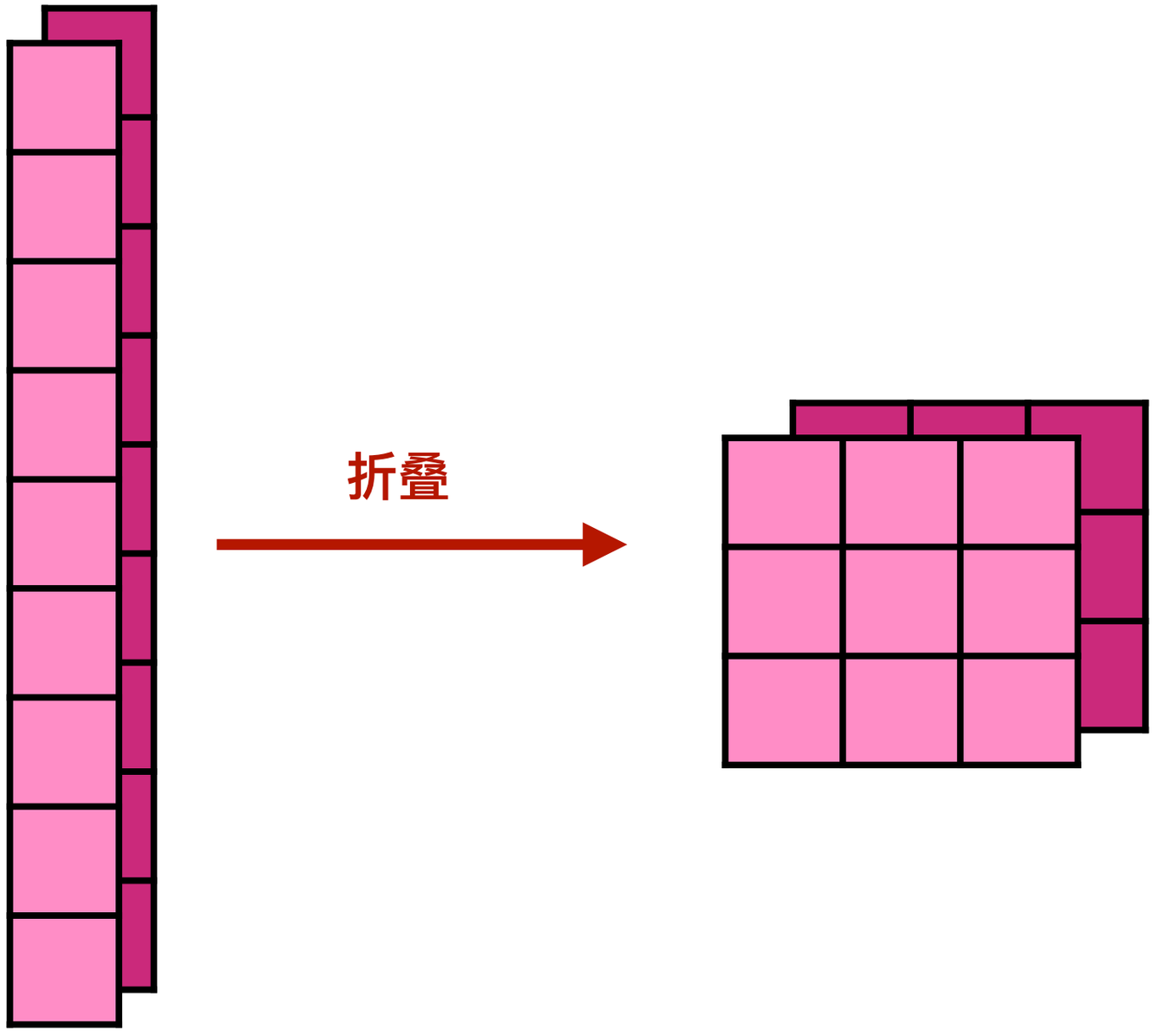
第二步：MLP



Author:WangQi
Email:wangqikaixin@gmail.com

MlpConv 结构

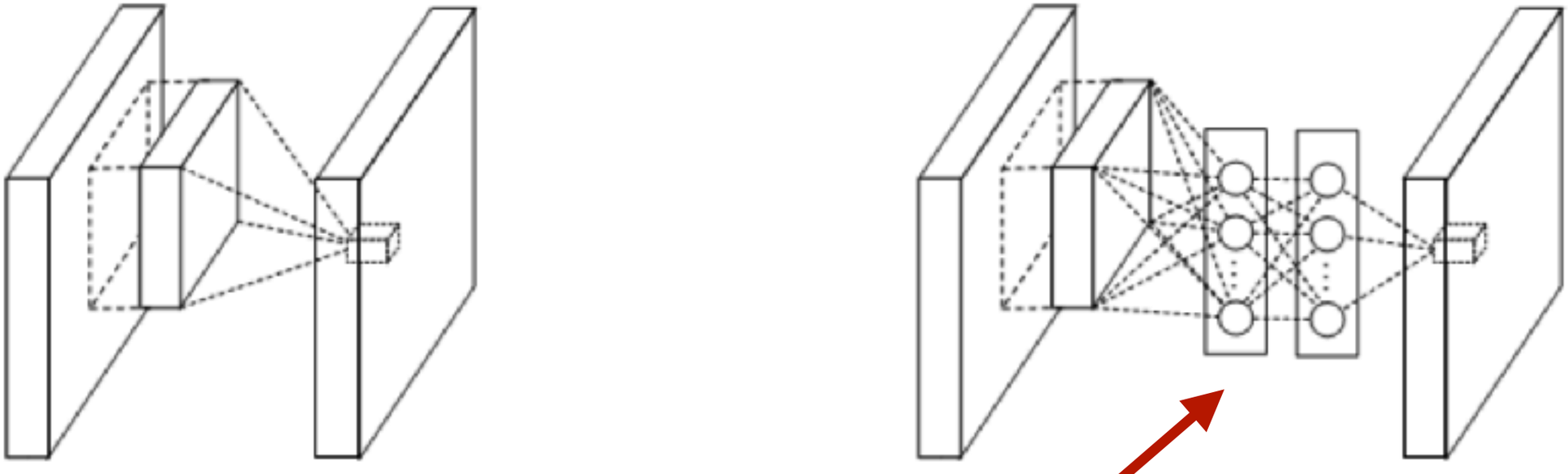
第二步：MLP



Author:WangQi
Email:wangqikaixin@gmail.com

MlpConv 结构

卷积后接一层共享参数的全连接层等价于卷积后接一层1 * 1卷积层。



(a) Linear convolution layer

(b) Mlpconv layer

接两层MLP等价于接两层1 * 1卷积层，论文中建议接入两层以保证其拥有拟合任意非线性流形的能力。

Author:WangQi
Email:wangqikaixin@gmail.com

MlpConv 优点

- MlpConv可提取非线性特征，减少了卷积层与卷积核的数量；

- 使用1 * 1卷积大大降低了参数规模。

Author:WangQi
Email:wangqikaixin@gmail.com

贡献二：全局平均池化

Author:WangQi
Email:wangqikaixin@gmail.com

全局平均池化

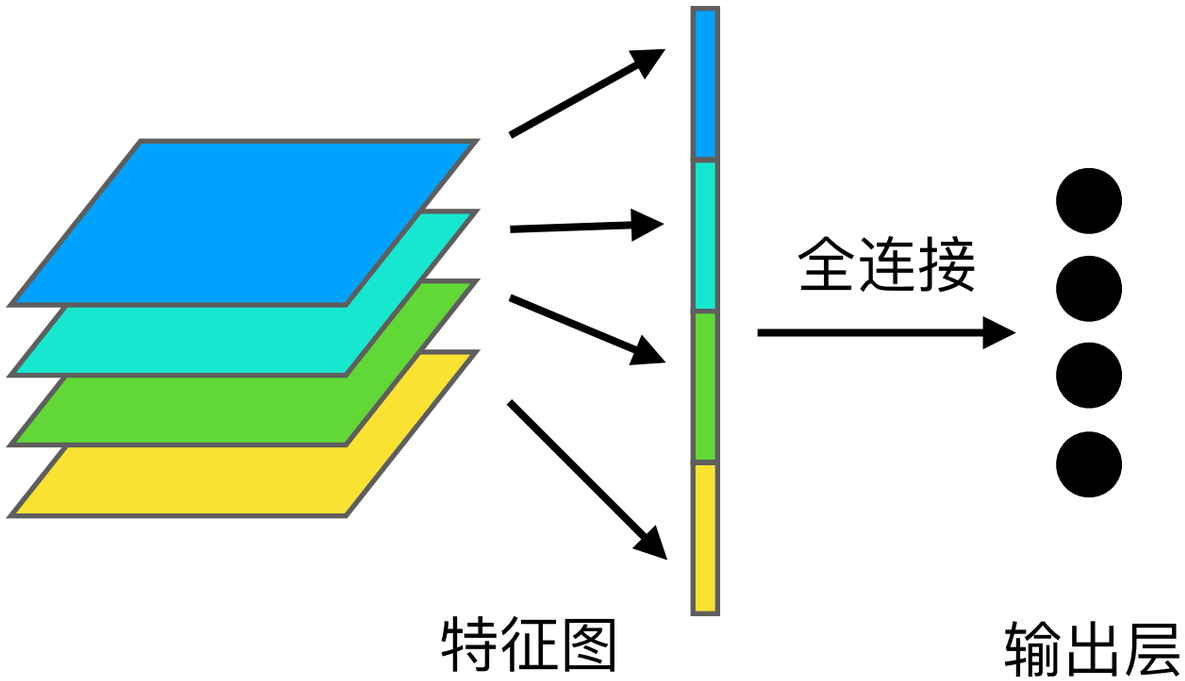
全局平均池化（Global Average Pooling，GAP）将卷积后的特征图使用与特征图等大的窗口执行平均池化，使得每一张特征图均转化为一个标量值。是NIN中重要的贡献之一，通过使用全局平均池化替代全连接层，极大的降低了模型参数规模，起到了正则化的作用，提高了模型性能。

Author:WangQi
Email:wangqikaixin@gmail.com

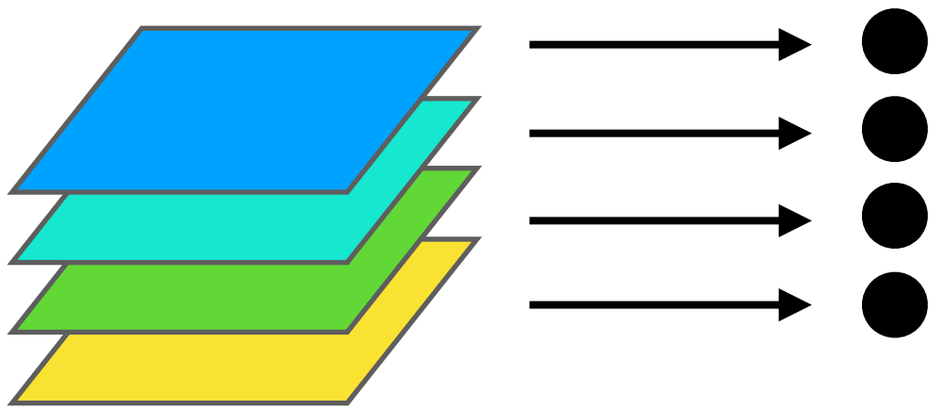
全局平均池化

AlexNet中3层全连接层的参数数量占全部参数数量的95%；

Vgg19中3层全连接层的参数数量占全部参数数量的86%；



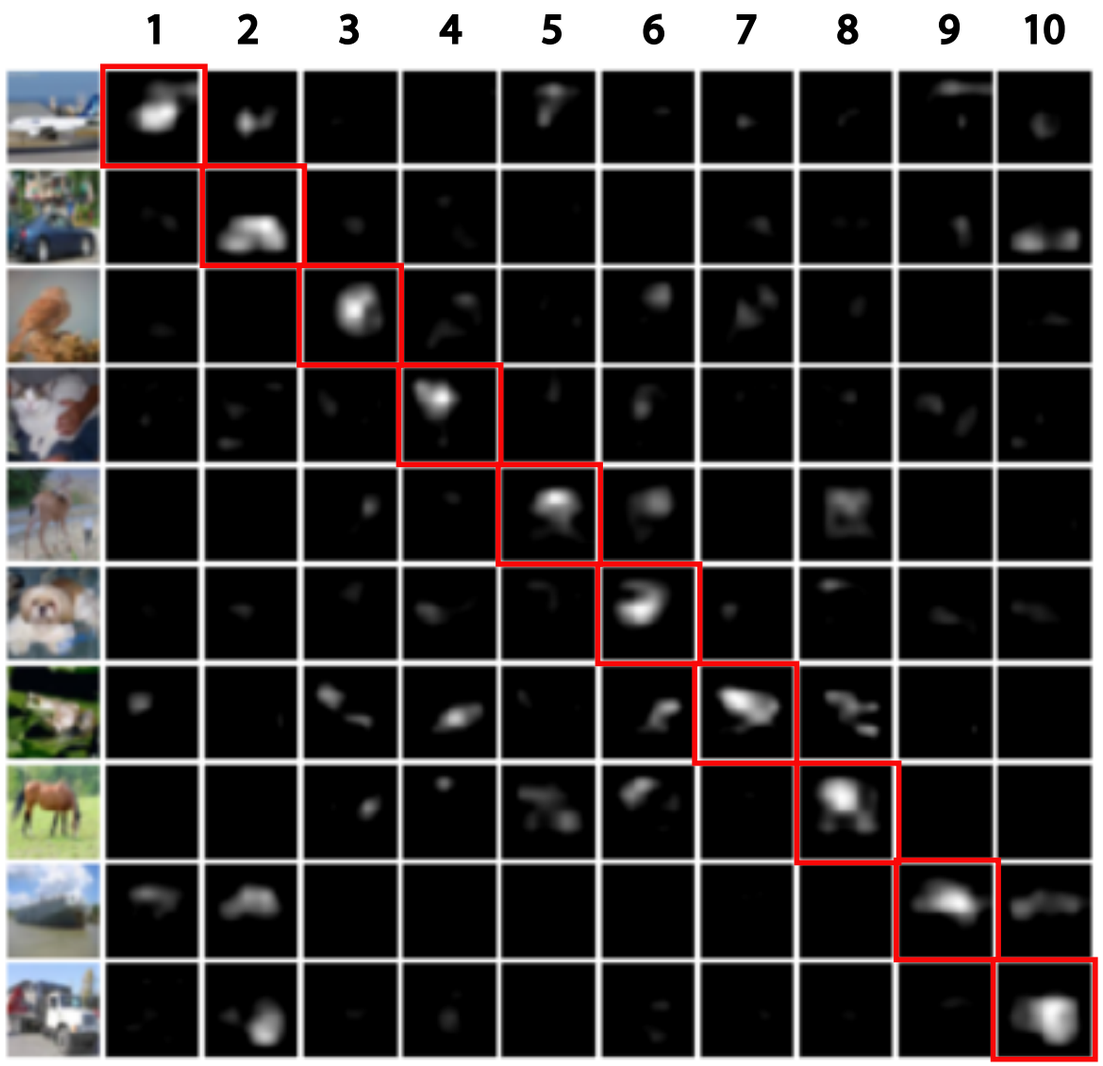
a) 全连接层

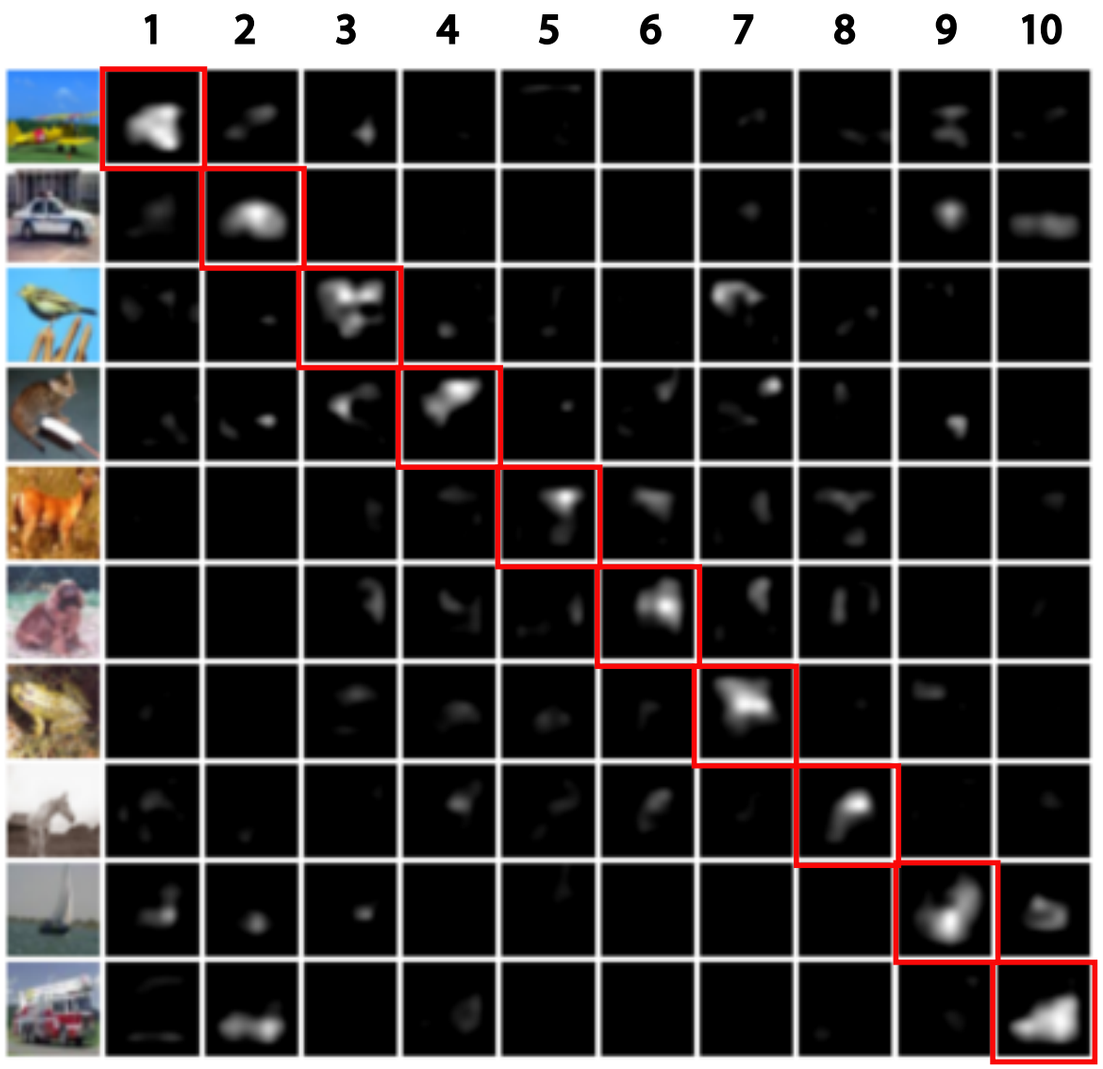


a) 全局平均池化

Author:WangQi
Email:wangqikaixin@gmail.com

全局平均池化





全局平均池化使得最后一层的每个特征图的激活区域都对应了相应的对象，可用来解释模型输出结果的原因。

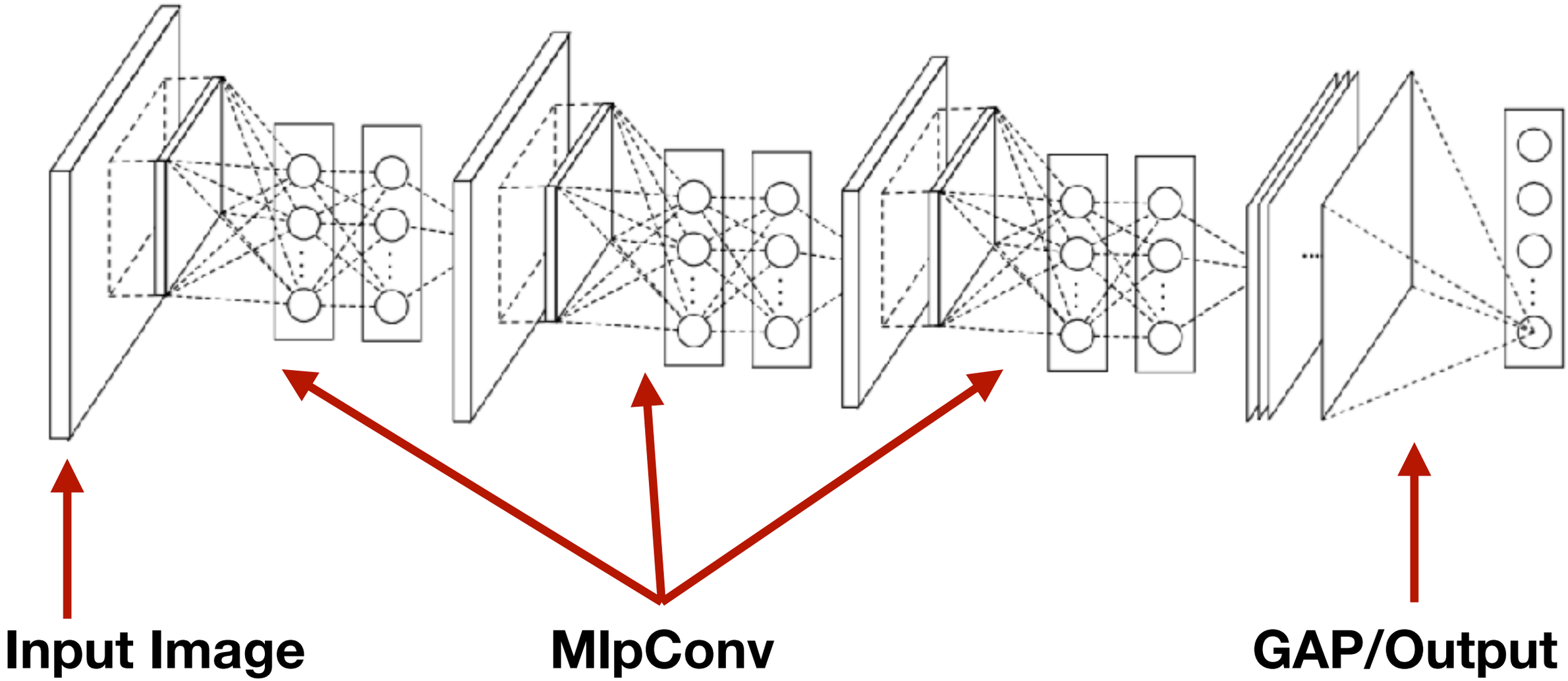
Author:WangQi
Email:wangqikaixin@gmail.com

全局平均池化的优点

- 相比全连接层的“黑箱操作”，其可解释性更强；
- 极大降低了模型参数量，起到了正则化的作用；
- 极大降低了模型计算量。

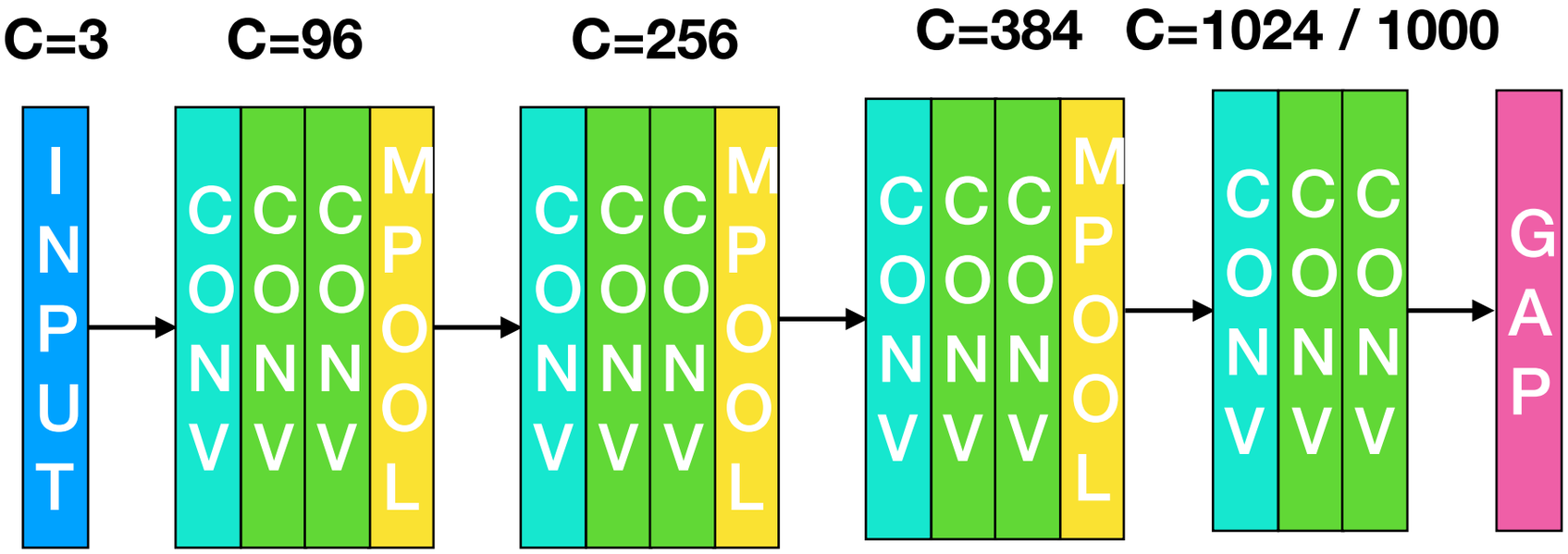
Author:WangQi
Email:wangqikaixin@gmail.com

NIN Net 总体结构



Author:WangQi
Email:wangqikaixin@gmail.com

NIN Net 总体结构



3/2重叠最大池化

普通卷积

1*1卷积

第一次卷积、第一次、第三次最大池化以及GAP采用VALID边界处理方式，其它均为SAME。

Author:WangQi
Email:wangqikaixin@gmail.com

计算：NIN模型参数数量。

Author:WangQi
Email:wangqikaixin@gmail.com

NIN 总结

- 提出MlpConv，可用于提取非线性特征；
- 使用了较少的MlpConv层即可完成特征提取，极大的降低了模型参数数量；
- MlpConv中的多层感知机实现等价于 1 * 1 卷积；
- 1 * 1 卷积本质上是在做通道信息的交流；
- 1 * 1 卷积可以用于提升或降低通道数；
- 全局平均池化

Author:WangQi
Email:wangqikaixin@gmail.com

1.3 使用TensorFlow实现NIN模型

Author:WangQi
Email:wangqikaixin@gmail.com

思考

思考：使用全局平均池化是否能够使得神经网络的输入兼容不同大小的图片？使用NIN模型，17 flower数据集进行验证。

Author:WangQi
Email:wangqikaixin@gmail.com

2. GoogLeNet

Author:WangQi
Email:wangqikaixin@gmail.com

2.1 简介与背景

Author:WangQi
Email:wangqikaixin@gmail.com

GoogLeNet简介

DeepCNet	17.5	-	○	-	-	-	University of Warwick
DeepInsight	-	-	-	-	-	40.5	NLPR†, HKUST‡
FengjunLv	17.4	-	○	-	-	-	Fengjun Lv Consulting
GoogLeNet	**6.7**	-	26.4	-	-	**43.9**	Google
HKUST	-	-	-	-	28.9	-	Hong Kong U. of Science and Tech.†, Chinese U. of H. K.‡, Stanford U.∓
libccv	16.0	-	○	-	-	-	libccv.org
MIL	18.3	-	33.7	-	-	30.4	The University of Tokyo†, IIT

注：此得分名单不完整。

GoogLeNet在 ILSVRC 2014 的比赛中，将 Top-5 error 降低至6.7%，获得了分类任务第一名的成绩。

Author:WangQi
Email:wangqikaixin@gmail.com

GoogLeNet 背景

- 提升NN性能最直接的办法是增加模型深度与宽度，但缺点明显：参数量大、计算量大、过拟合严重；

- 参数量大本质上是由密集连接造成的；

- 生物神经系统的连接是稀疏的；

- ANN的统计特征表明大规模稀疏网络可以在不损失性能的条件下被简化；

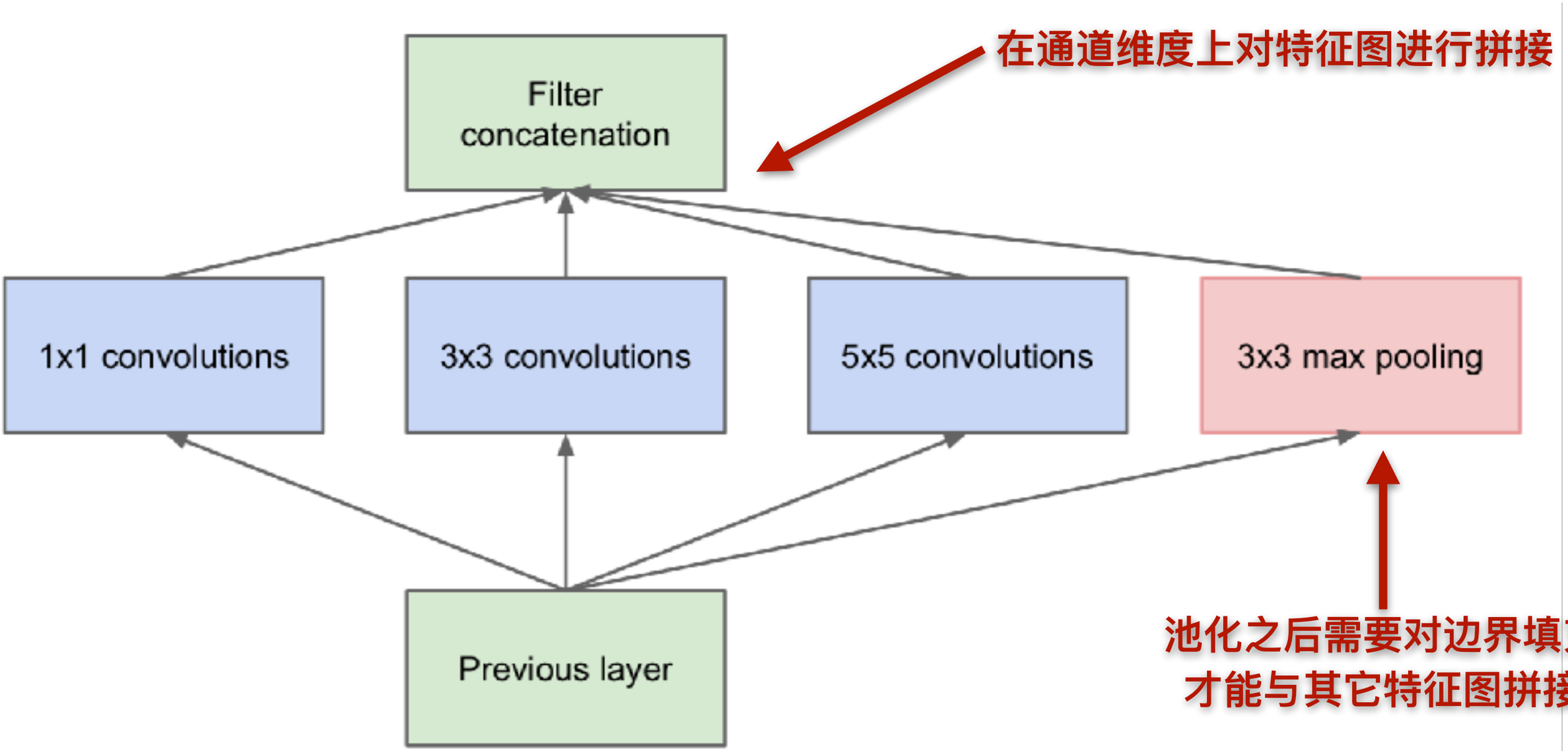
- 计算机软硬件对非均匀稀疏数据的计算效率差。

Author:WangQi
Email:wangqikaixin@gmail.com

2.1 Inception Module

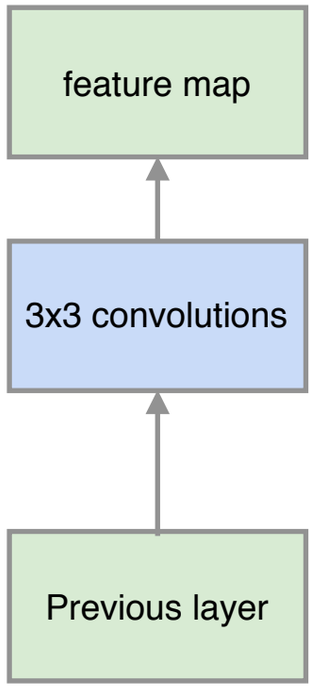
Author:WangQi
Email:wangqikaixin@gmail.com

Inception（naive）模块

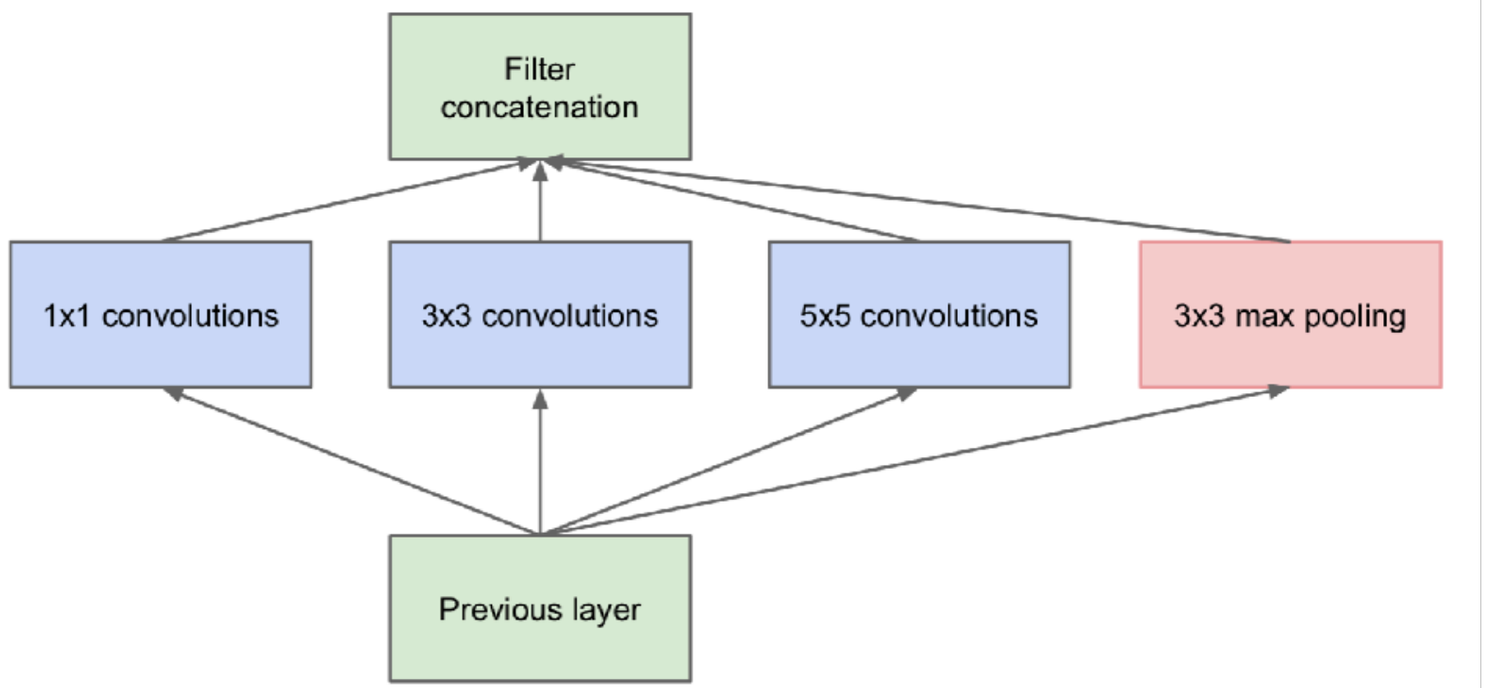


Author:WangQi
Email:wangqikaixin@gmail.com

结构对比



“传统” 卷积



Inception （naive）卷积模块

之前同一层的卷积中均使用较多单一型的核执行卷积，Inception卷积模块使用多种核进行卷积与池化，每种卷积核的数量与单一型的核相比均要少很多。

Author:WangQi
Email:wangqikaixin@gmail.com

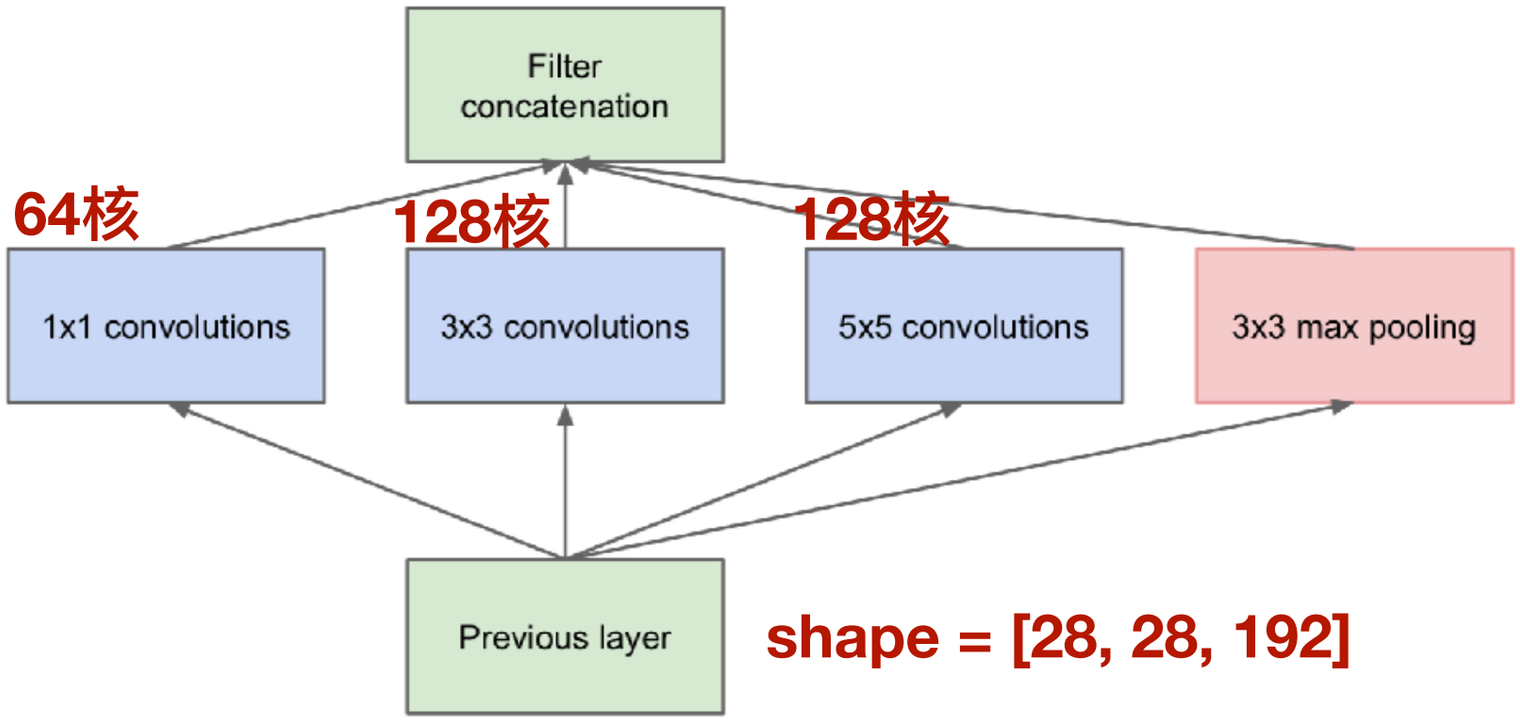
Inception（naive）模块特点

- 并联执行小规模的卷积池化，使得卷积结构变得更加稀疏；
- 卷积采用不同大小的卷积核，可提取不同尺度上的特征；
- 并联的卷积与池化得到的特征图再通道上进行拼接，使得后面的层能够利用前面提取到的多尺度特征；
- 合并通道带来维度灾难问题，尤其是池化层。

Author:WangQi
Email:wangqikaixin@gmail.com

实例

问题：使用TensorFlow实现下图中所展示的卷积过程，并计算输出通道数，观察通道维度扩大了多少?



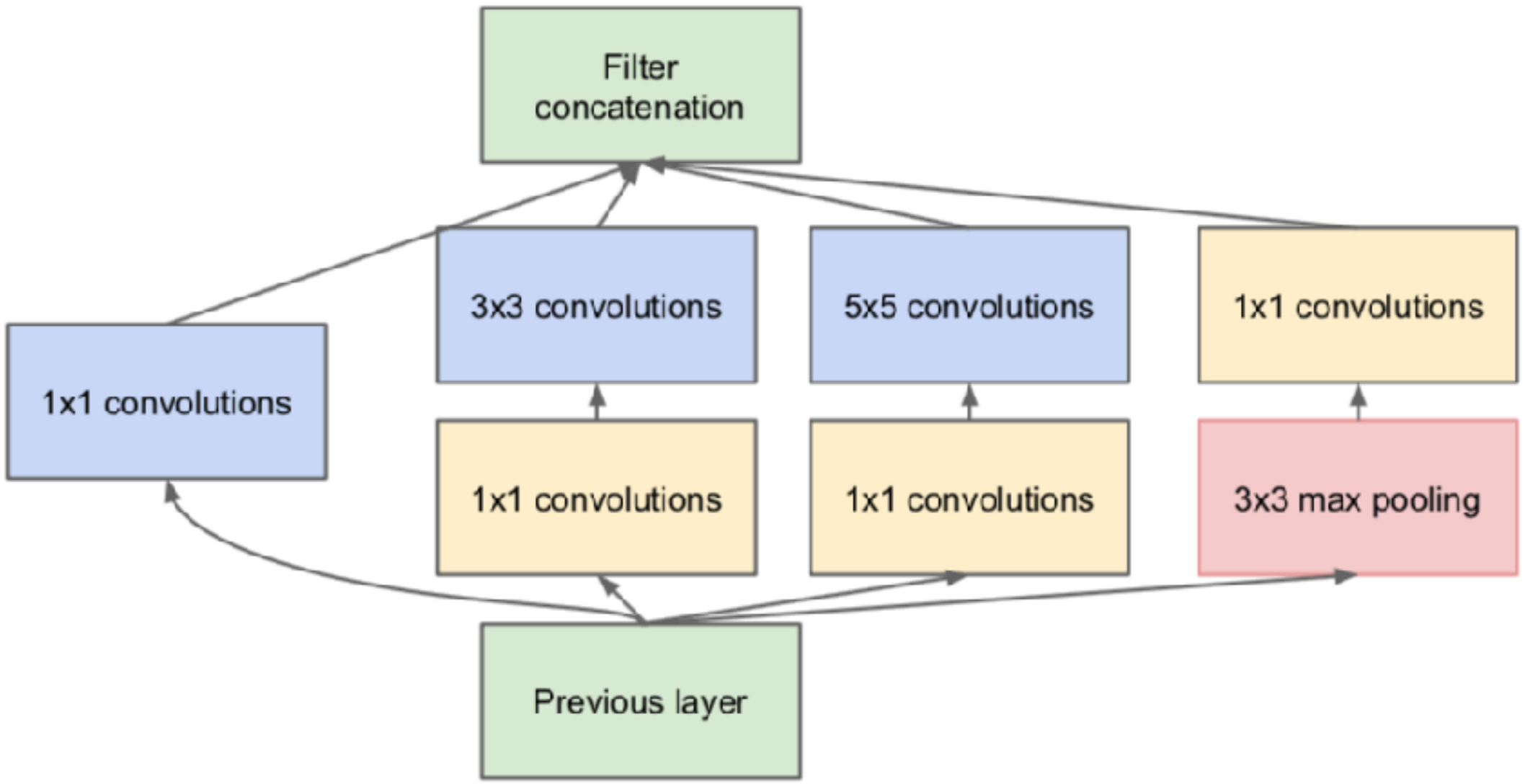
padding = SAME

pool_stride = 2

2.2 改进 Inception Module

Author:WangQi
Email:wangqikaixin@gmail.com

Inception模块



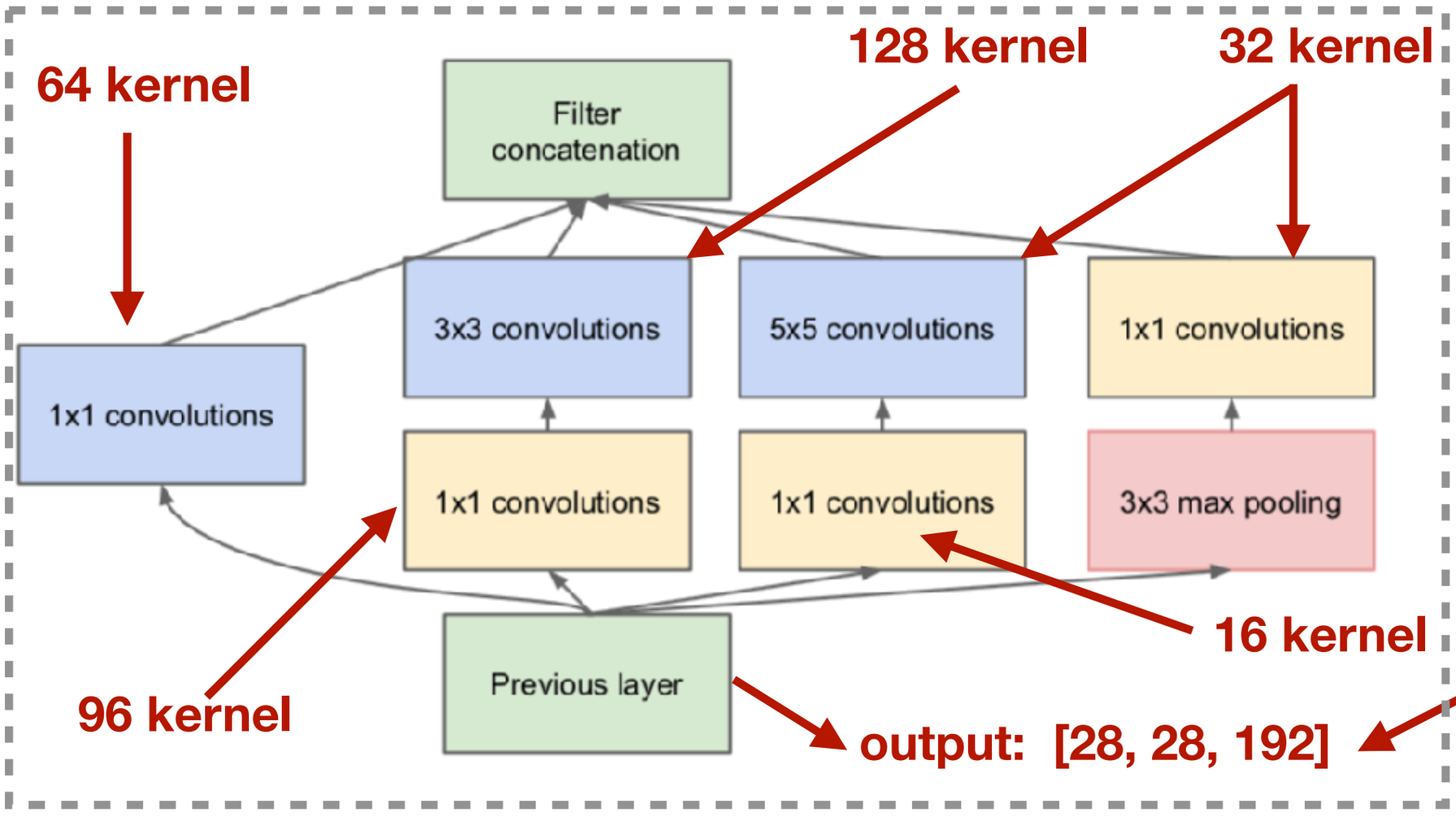
Author:WangQi
Email:wangqikaixin@gmail.com

改进Inception模块的优点

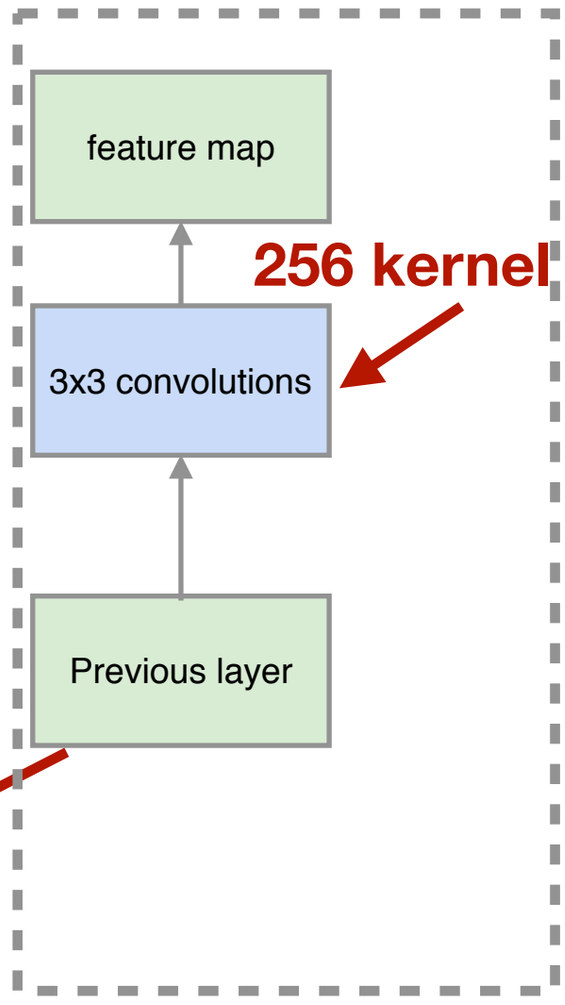
- 使用 1 * 1 卷积降低了模型参数规模；
- 使用 1 * 1 卷积降低了通道规模；
- 两次卷积提高了Inception模块的非线性拟合能力。

Author:WangQi
Email:wangqikaixin@gmail.com

实验



Inception 3(a)



普通卷积

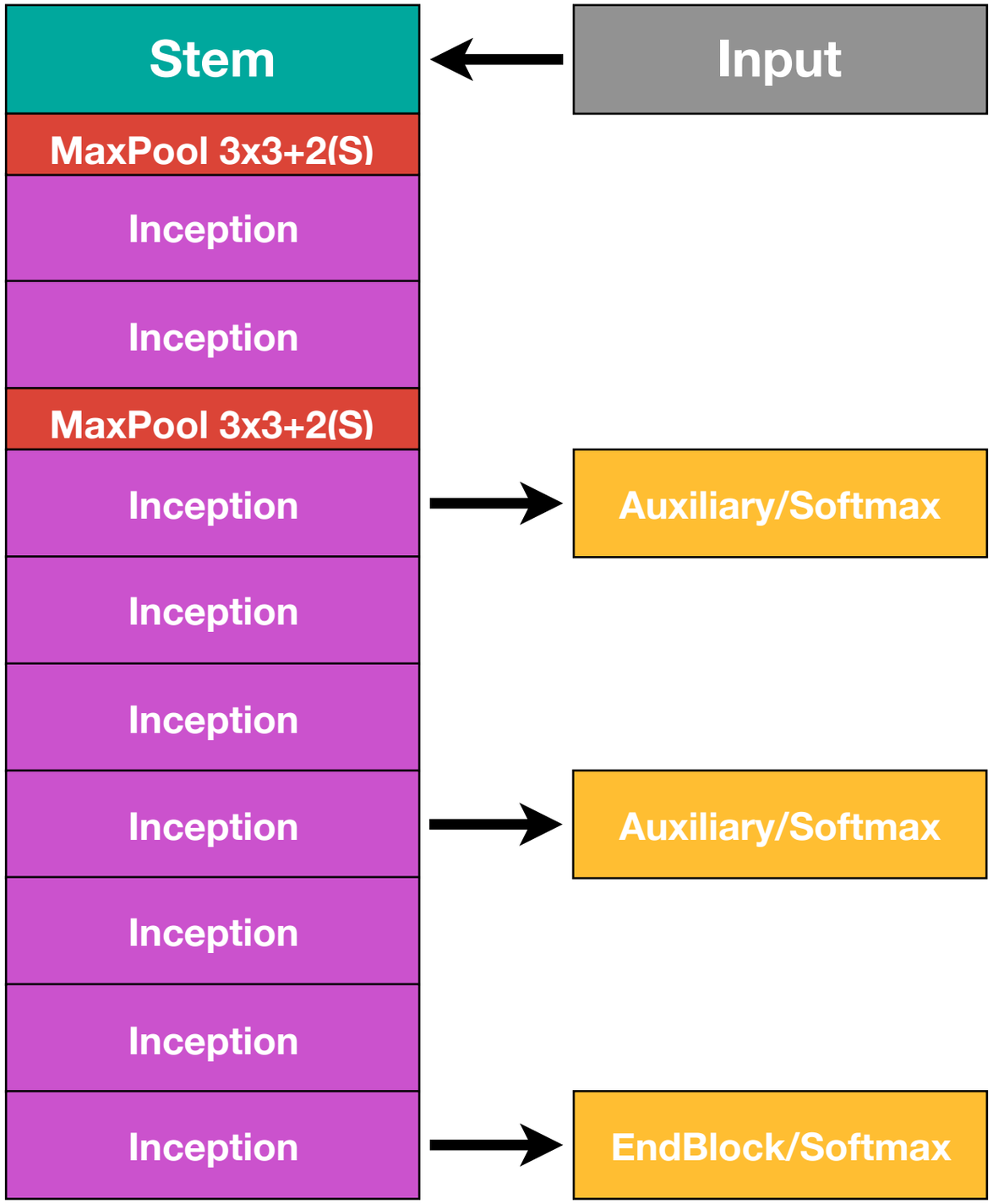
1. 计算：模块参数数量，以及对应的“传统”卷积的参数数量。2. 使用TensorFlow实现模块。

Author:WangQi
Email:wangqikaixin@gmail.com

2.3 GoogLeNet模型结构

Author:WangQi
Email:wangqikaixin@gmail.com

整体结构（简化）



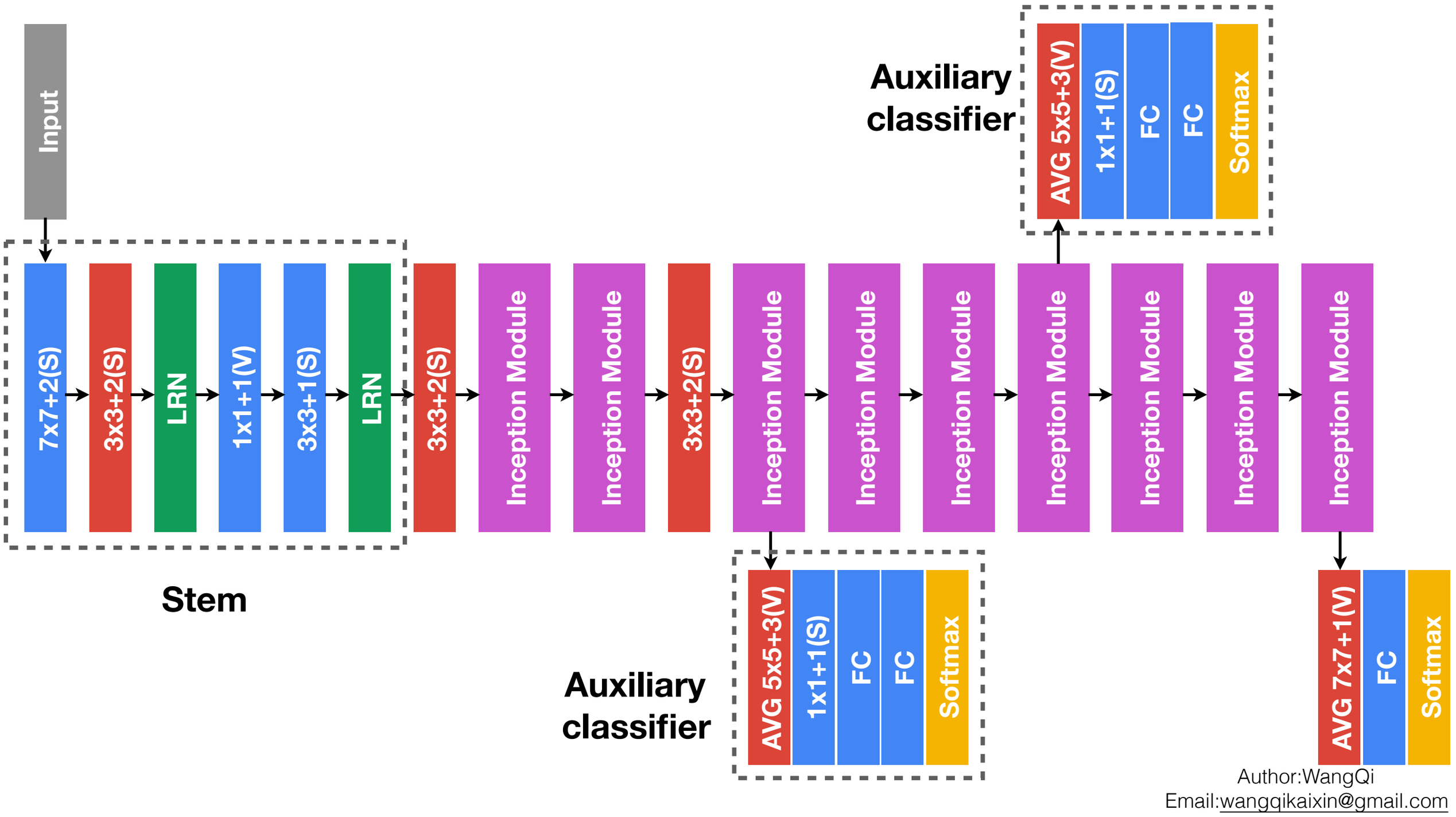
初始的几层（stem）主要提取局部特征，无需使用Inception模块。

加入辅助分类器，降低梯度消失的影响，模型训练完毕之后，将不再需要辅助分类器。

取消全连接层，使用全局平均池化代替，降低模型参数数量。（注：为方便模型迁移，在GAP层之后接入一个全连接层）

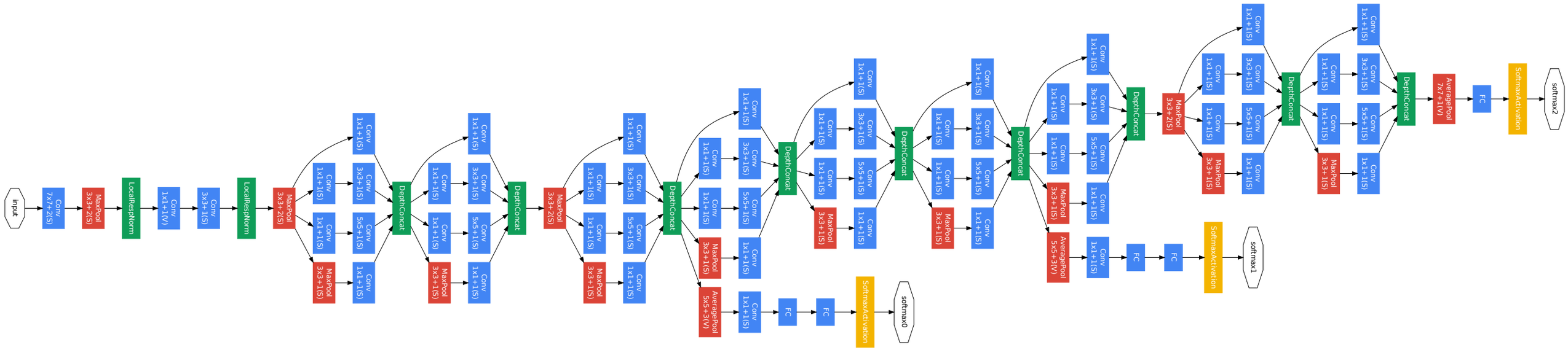
Author:WangQi
Email:wangqikaixin@gmail.com

整体结构



Author:WangQi
Email:wangqikaixin@gmail.com

整体结构



Author:WangQi
Email:wangqikaixin@gmail.com

整体结构

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

注意：3*3 reduce代表3*3卷积前1*1滤波器的个数，5*5 reduce代表5*5卷积前1*1滤波器的个数。

Author:WangQi
Email:wangqikaixin@gmail.com

GoogLeNet 总结

- 提出 Inception Module，即使用多个并行且较小的稠密卷积与池化近似稀疏连接；

- 引入1 * 1卷积改进Inception Module，降低了模型参数数量与特征图大小；

- 加入辅助分类器解决梯度消失问题；

- 取消全连接层，使用GAP代替。

Author:WangQi
Email:wangqikaixin@gmail.com

实验

- 根据前面的“整体结构”，使用TensorFlow构建GoogLeNet模型。

3. InceptionNet

Author:WangQi
Email:wangqikaixin@gmail.com

3.1 简介

Inception Net V2、V3出自于论文《**Rethinking the Inception Architecture for Computer Vision**》，其思考了不同结构的卷积神经网络应用于计算机视觉时的表现，并通过实验证实了这些结构的作用与利弊。为我们设计计算机视觉模型等提供了即为重要的参考。

Author:WangQi
Email:wangqikaixin@gmail.com

3.2 通用设计原则

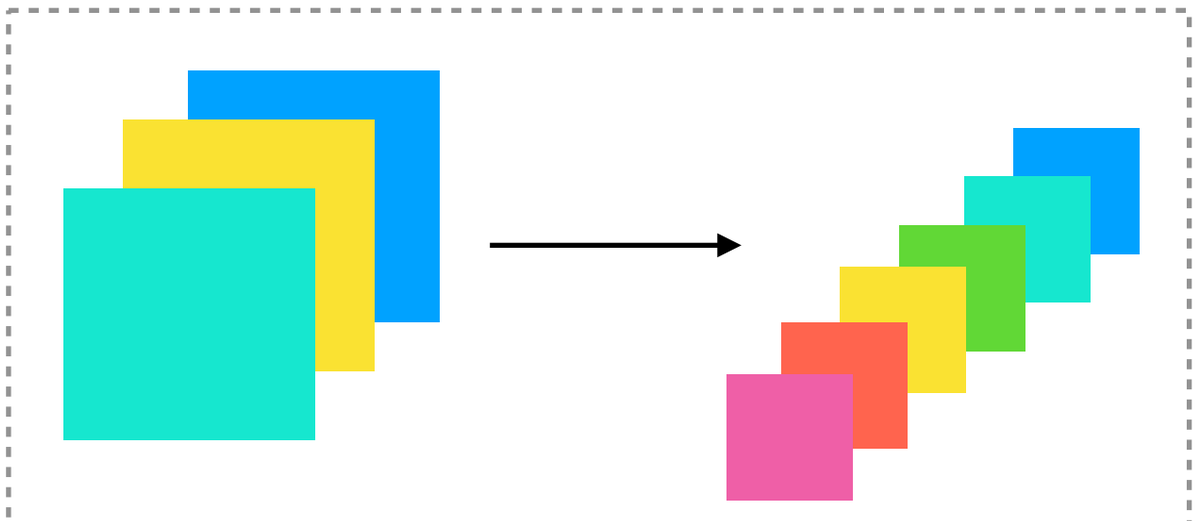
Author:WangQi
Email:wangqikaixin@gmail.com

通用设计原则

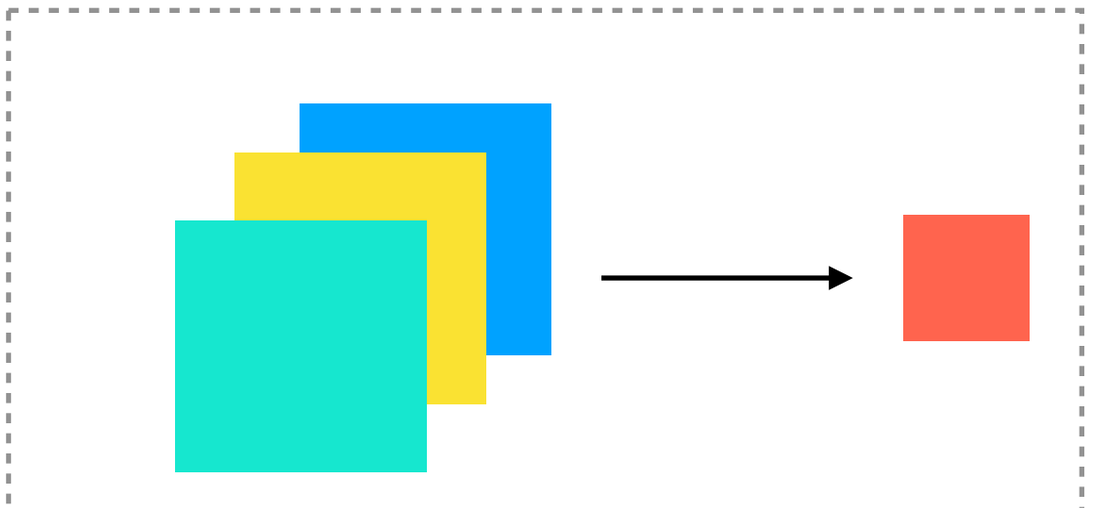
- 避免表示瓶颈；
- 在网络局部中，更高维度的表示更容易去处理；
- 低维嵌入空间上进行空间聚集，损失并不是很大。
- 平衡网络的宽度和深度。

Author:WangQi
Email:wangqikaixin@gmail.com

避免表示瓶颈



步长为2，卷积核数量增加一倍。较好的表示

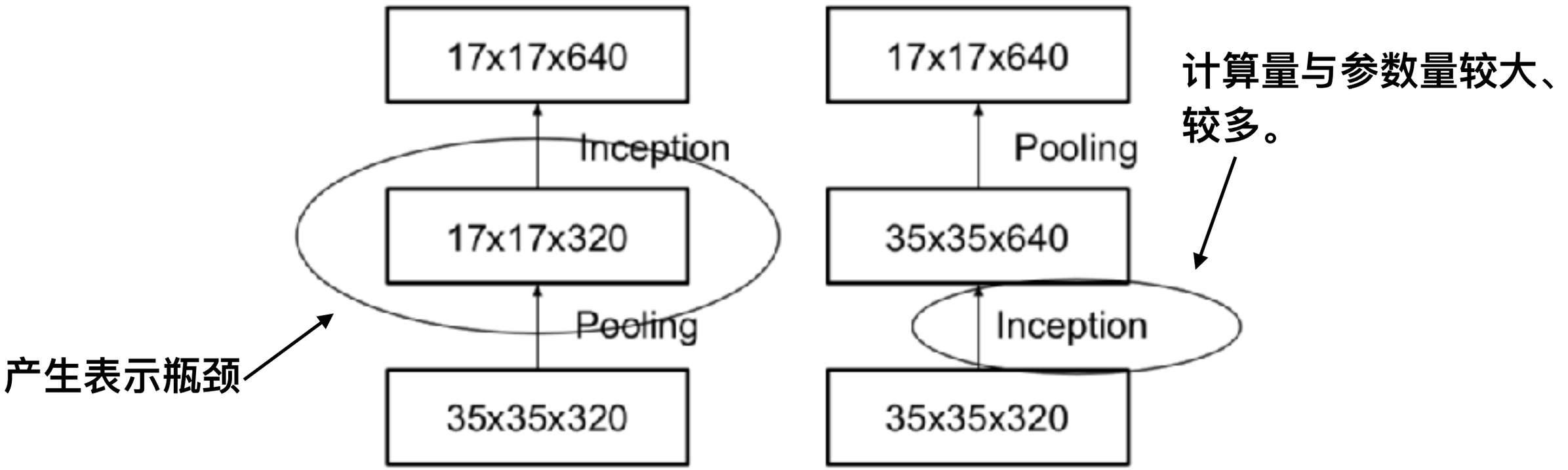


步长为2，卷积核数量减少。较差的表示

我们可以使用网络中每一层的维度近似每一层的信息容量（只能近似表示，因为丢失了结构信息）。一般的在到达当前任务的最终表示之前，表示的大小应该由输入层到输出层缓慢的减少，所以应该避免极端的瓶颈层表示。

Author:WangQi
Email:wangqikaixin@gmail.com

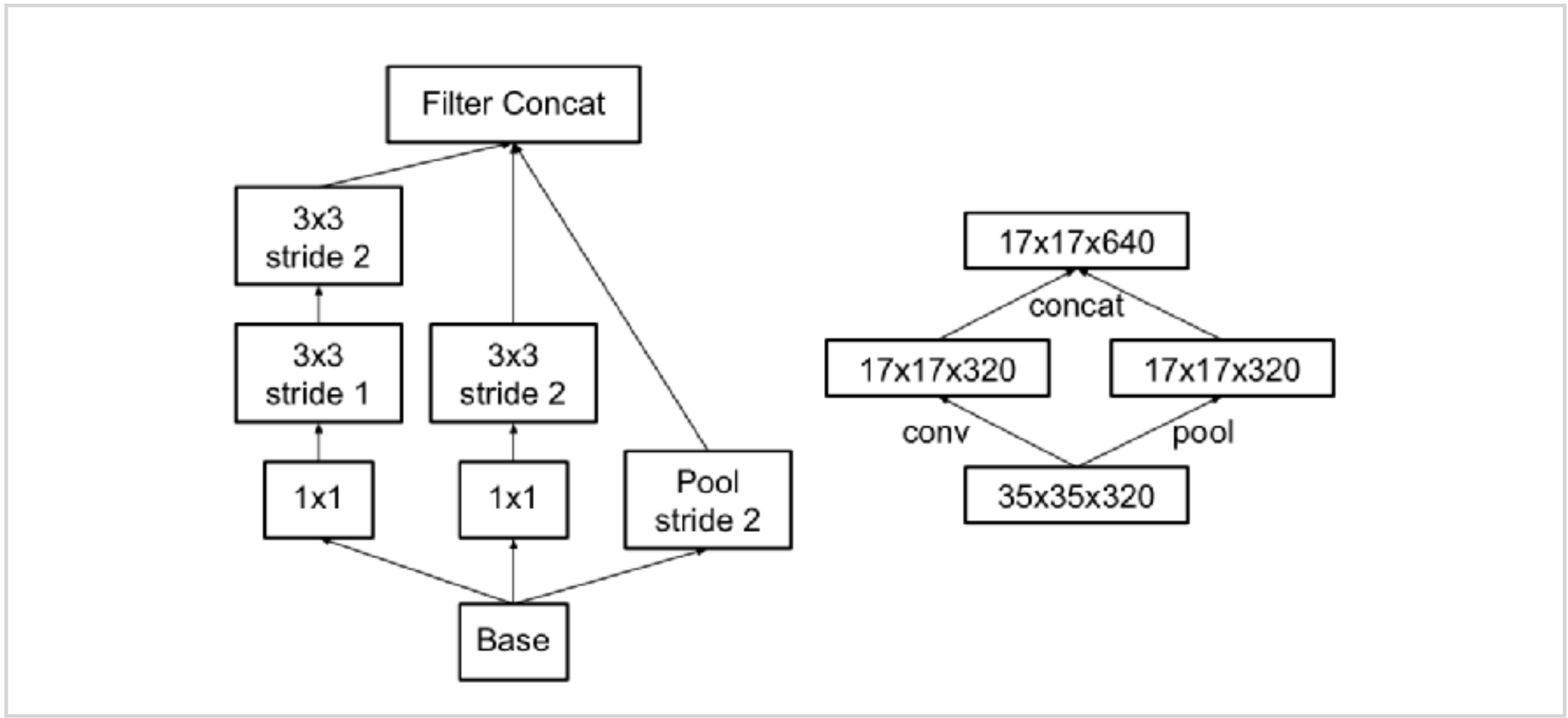
欠佳用法



传统上，使用池化操作来缩减特征图尺寸，为避免表示瓶颈，在池化之前需要扩展通道维度。上图中左边的方案违反了表示瓶颈原则，右边的方案参数量增加1倍，计算量增加3倍。

Author:WangQi
Email:wangqikaixin@gmail.com

正确用法

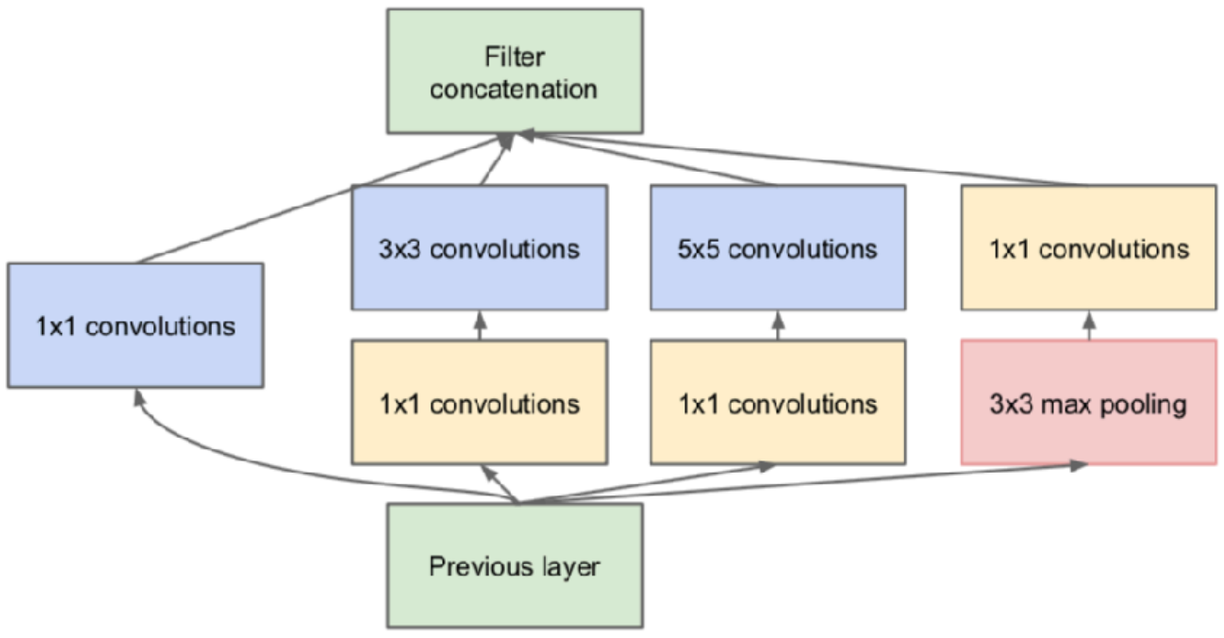


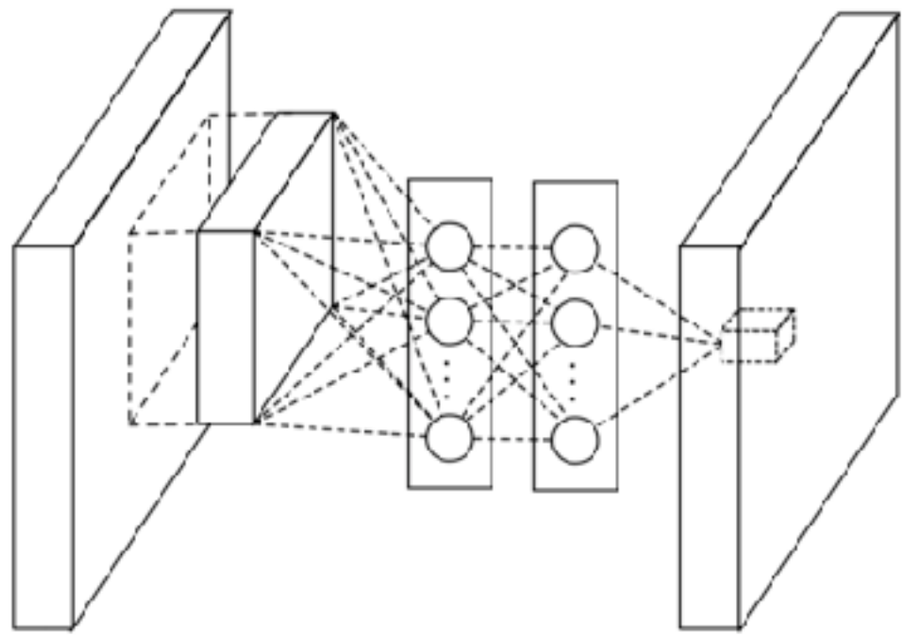
缩减特征图尺寸的同时扩展通道维度。它不仅计算量少而且避免了表示瓶颈。右图为从特征图大小的角度看的相同解决方案。

Author:WangQi
Email:wangqikaixin@gmail.com

更高维的表示更易处理

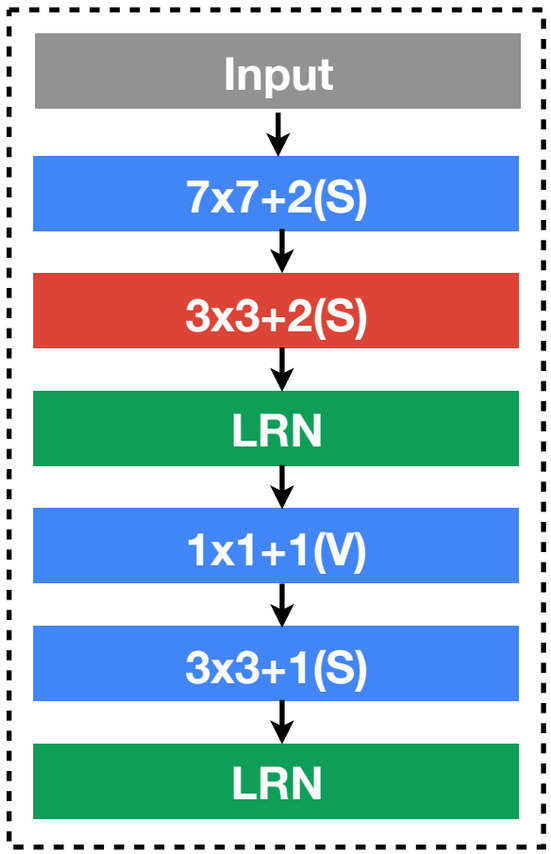
卷积之后增加非线性激活函数可以更快速的提取到特征，训练速度也更快。
例如Inception Module与MlpConv。



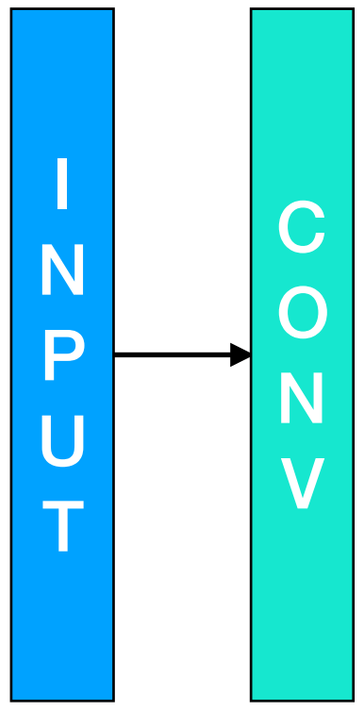


Author:WangQi
Email:wangqikaixin@gmail.com

空间聚集



GoogLeNet stem

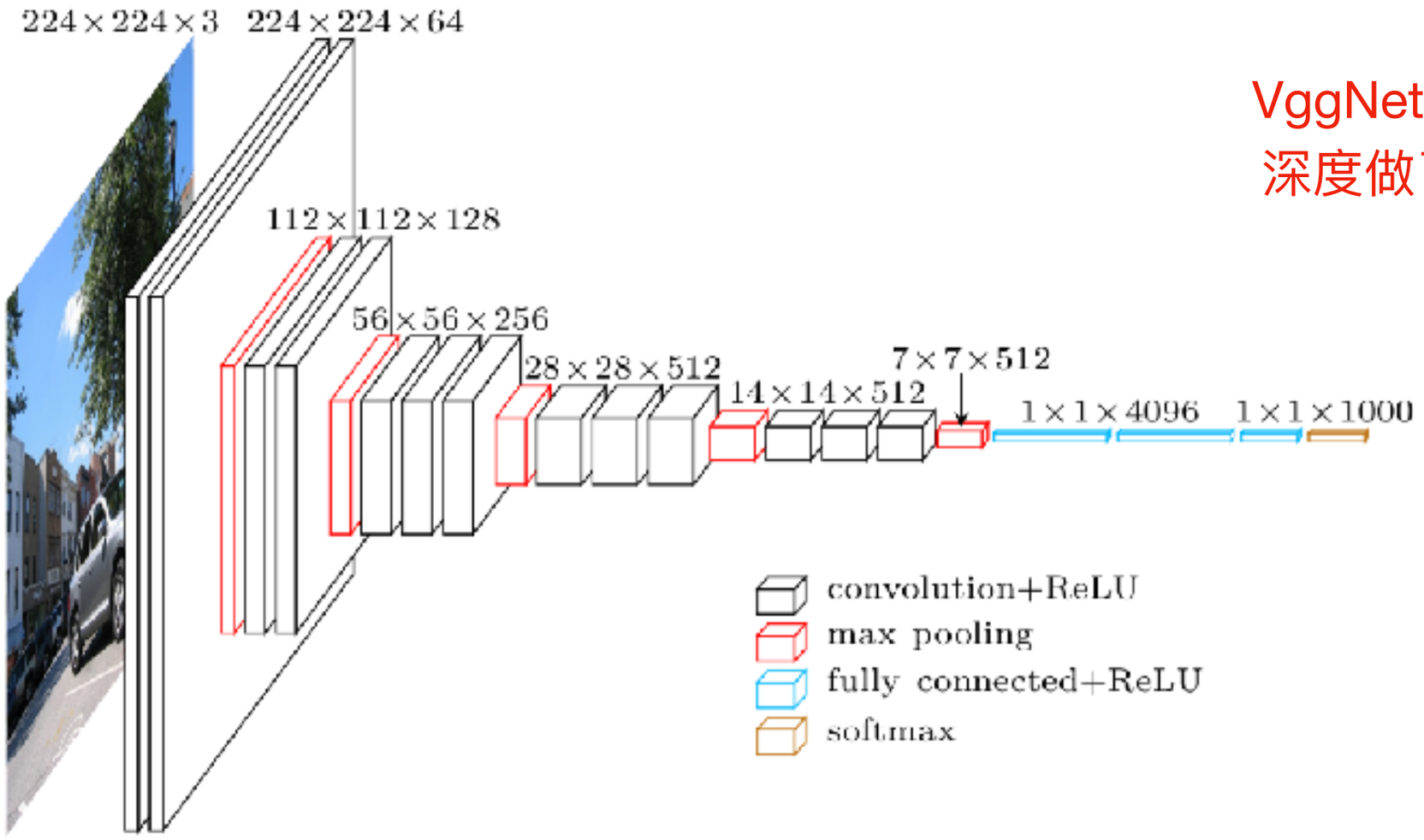


NIN 部分结构

相邻神经元之间由很强的相关性，即信息具有冗余。

Author:WangQi
Email:wangqikaixin@gmail.com

平衡网络宽度与深度



VggNet对网络宽度与深度做了很好的平衡

增加网络的深度与宽度可以提升网络性能，如果两者通过合理的方式并行增加，则可以达到恒定计算量。

Author:WangQi
Email:wangqikaixin@gmail.com

3.3 batch normal

batch normal

Batch Normal（Batch Normalization，BN）是对每一个小批次训练数据在每一层神经网络中进行标准化的技术。其通过简便的操作，减缓了梯度不稳定带来的影响，极大的加快了模型训练的速度（应用在当时最好的神经网络中，训练次数减少为原来的1/14）。

思考：标准化的意义?

min-max normalization：

$$x^* = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

特点：

将数据映射到了区间[0, 1]中；

去量纲化；

改变了原始数据分布。

zero-mean normalization：

$$x^* = \frac{x - \mu}{\sigma}$$

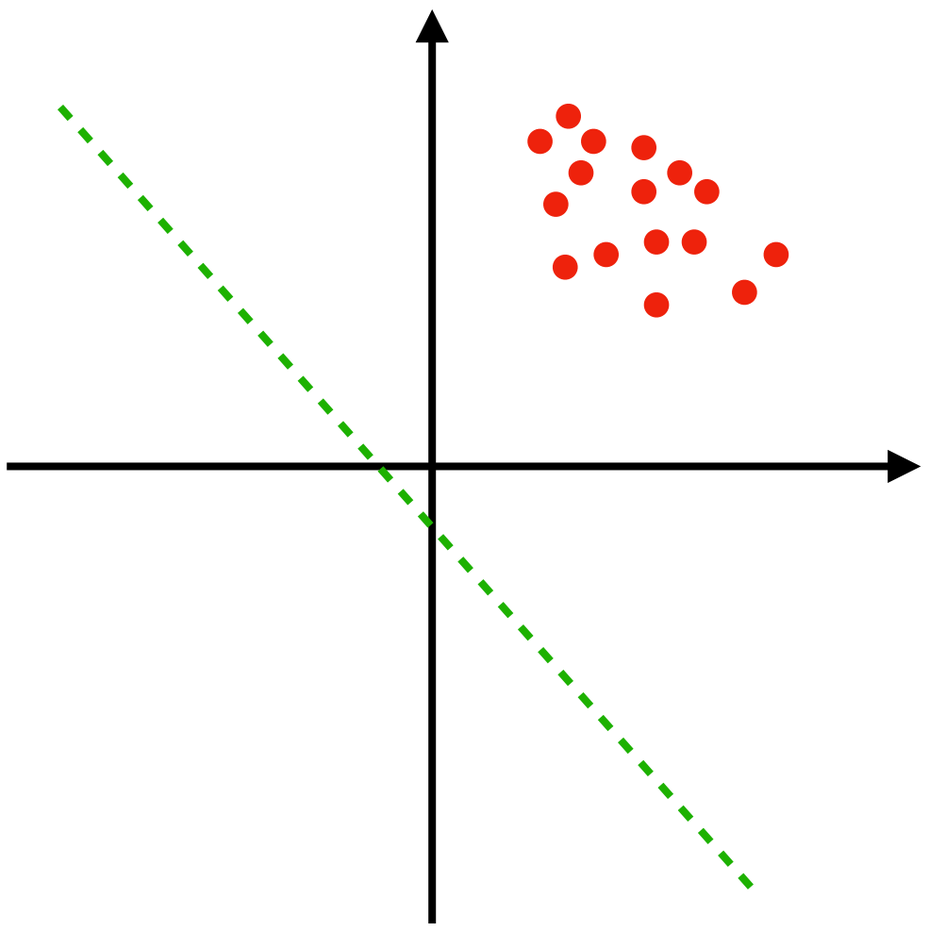
特点：

数据服从0均值，1标准化差的标准正态分布；

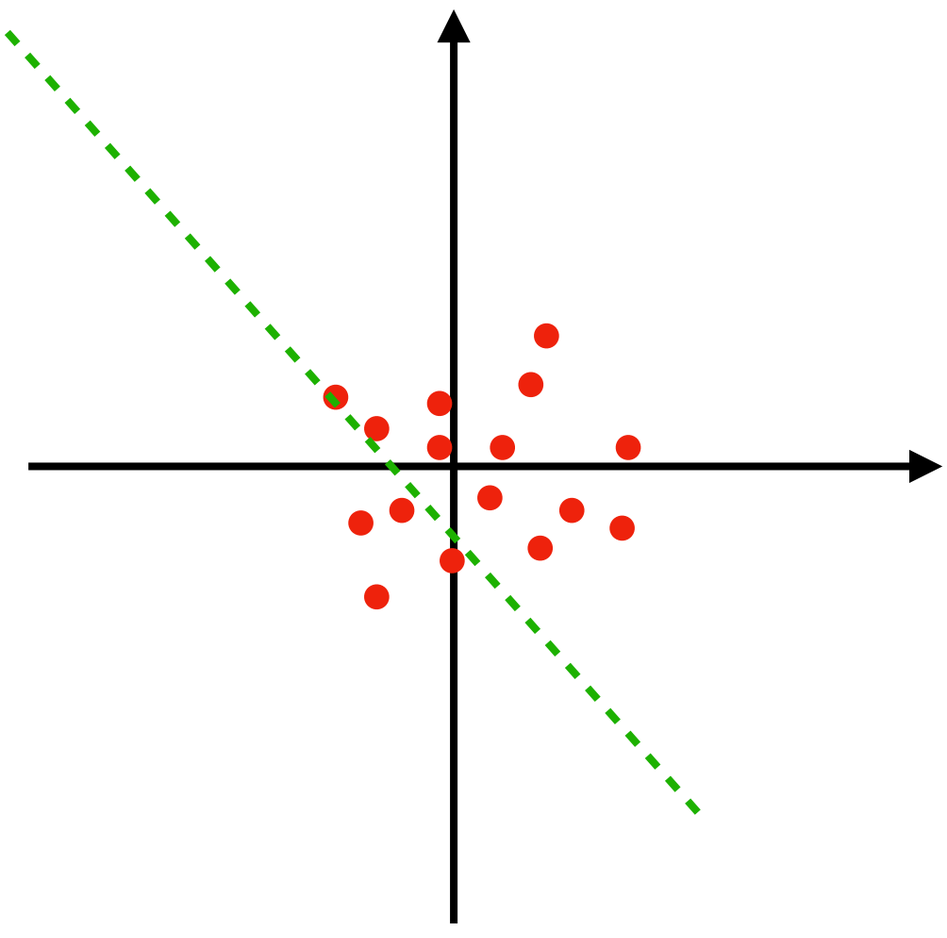
大部分数据均在区间[-1, 1]之间。

z-score标准化

标准化之前：



标准化之后：



使用相同的初始化方法，标准化之后模型与数据之间的“距离”更近，优化所需步数更少。

BN诞生的背景与动机

- 在训练模型之前，对数据进行标准化、白化等操作使得输入数据分布一致，加快了模型训练；
- 神经网络的某一层的输入分布会在模型参数更新之后发生剧烈变化，使得当前层不得不重新学习新的数据分布，这样使得DNN训练变得很复杂。这种问题在其文献中称之为“Internal Covariate Shift”。

猜想：如果使得DNN每一层的输入分布变得稳定的，那么就可以加速训练模型。

BN的核心要义

1. 使用z-score方法（原论文中考虑使用白化，但其计算量大，且非处处可微，所以没有使用），标准化每一层神经网络中的输入，使得在进入激活函数前，输入数据呈现标准正态分布；

2. 为保证DNN的表达能力，给每个标准化后的输入加入缩放与平移参数。

BN在训练时

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

小批量输入

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// 小批量输入的均值}$$

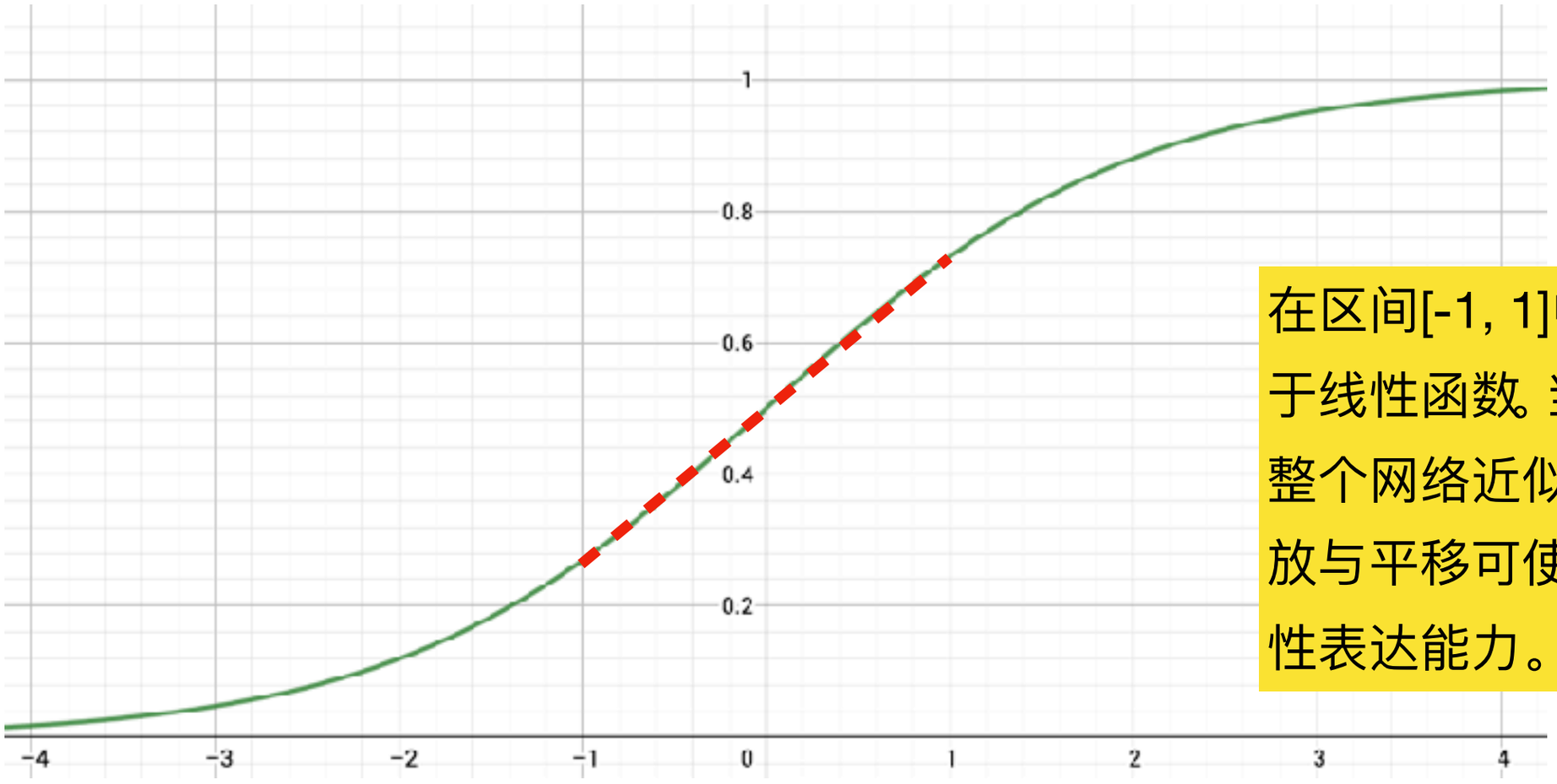
$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// 小批量输入的方差}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// 标准化}$$

小常数，用于避免分母为0

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// 缩放与平移}$$

缩放与平移的必要性



在区间[-1, 1]中logistic函数近似等于线性函数。当没有缩放与平移时，整个网络近似于线性模型。引入缩放与平移可使模型获得更强的非线性表达能力。

绿色曲线表示logistic函数图像，红色线段表示在区间[-1, 1]之间的近似于logistic的图像部分。

思考：当训练完毕模型时，便不存在min-batch了，此时，如何计算BN中的均值、方差呢?

BN在测试时

在测试时，使用训练阶段产生的均值与方差进行求得最终固定的均值与方差：

$$\mathrm{E}[x] \leftarrow \mathrm{E}_{\mathcal{B}}[\mu_{\mathcal{B}}] \qquad \text{// 所有批次中得到的均值的平均数}$$

$$\mathrm{Var}[x] \leftarrow \frac{m}{m-1}\mathrm{E}_{\mathcal{B}}[\sigma^2_{\mathcal{B}}] \qquad \text{// 所有批次中得到的方差的无偏估计}$$

$$y = \frac{\gamma}{\sqrt{\mathrm{Var}[x]+\epsilon}} \cdot x + \left(\beta - \frac{\gamma\,\mathrm{E}[x]}{\sqrt{\mathrm{Var}[x]+\epsilon}}\right)$$

BN注意事项

- 某一层神经网络使用了BN，代表这一层在数据进入激活函数前进行了BN操作（亦有实验说明在激活函数之后使用BN，效果更好，但通常不这样做）。

- BN是对每一个神经元的输入进行标准化，每一个神经元都拥有对应的重构变换（缩放、平移）超参数。在卷积中，为避免超参数过多，也采用类似与权重共享的策略，即以特征图为单位执行BN。

- 通常的缩放参数初始化为1，平移参数初始化为0，这样默认的所有的输入都进行了标准化；

- 全连接神经网络层之后接BN层，计算均值、方差时是在第0个维度上；卷积神经网络层之后接BN层，计算均值、方差时是在第0、1、2个维度上。

- 为了测试时可以使用训练时的均值与方差，需要将训练过程中所有的均值与方差保留下来，这样做的代价比较高，通常我们使用滑动平均进行代替。

实例

使用TensorFlow实现BN层，主要使用3个方法来完成：

```
tf.nn.moments                                  #   计算均值与方差
tf.train.ExponentialMovingAverage              #    执行滑动平均
tf.nn.batch_normalization                      #   计算BN输出
```

执行过程：

① 计算均值方差；

② 执行滑动平均更新积累的均值与方差；

③ 根据更新之后的均值与方差执行BN计算。

实例

计算均值与方差：

```
# 全连接神经网络中某一层的输入 inputs.shape=[32, 256]
mean, variance = tf.nn.moments(inputs, axes=0)
```

全连接神经网络中计算所有小批量样本对同一个神经元输入的均值、方差。

```
# 卷积神经网络中某一层的输入 inputs.shape=[32, 64, 64, 3]
mean, variance = tf.nn.moments(conv_inputs, axes=[0, 1, 2])
```

卷积神经网络中相当于计算所有小批量样本中对应同一张特征图上所有值的均值、方差。

实例

使用滑动平均更新积累均值与方差：

```
mean, var = tf.nn.moments(inputs, axes=0)

ema = tf.train.ExponentialMovingAverage(decay=0.99)
ema_apply_op = ema.apply([mean, var])
```

这里会为均值与方差创建对应的影子变量保存当前的均值与方差。

实例

执行BN计算：

```
mean, var = tf.nn.moments(inputs, axes=0)

ema = tf.train.ExponentialMovingAverage(decay=0.99)
ema_apply_op = ema.apply([mean, var])

with tf.control_dependencies([ema_apply_op]):
    outputs= tf.nn.batch_normalization(inputs, ema.average(mean),
                                       ema.average(var), offset=0,
                                       scale=1, variance_epsilon=0.001)
```

实验

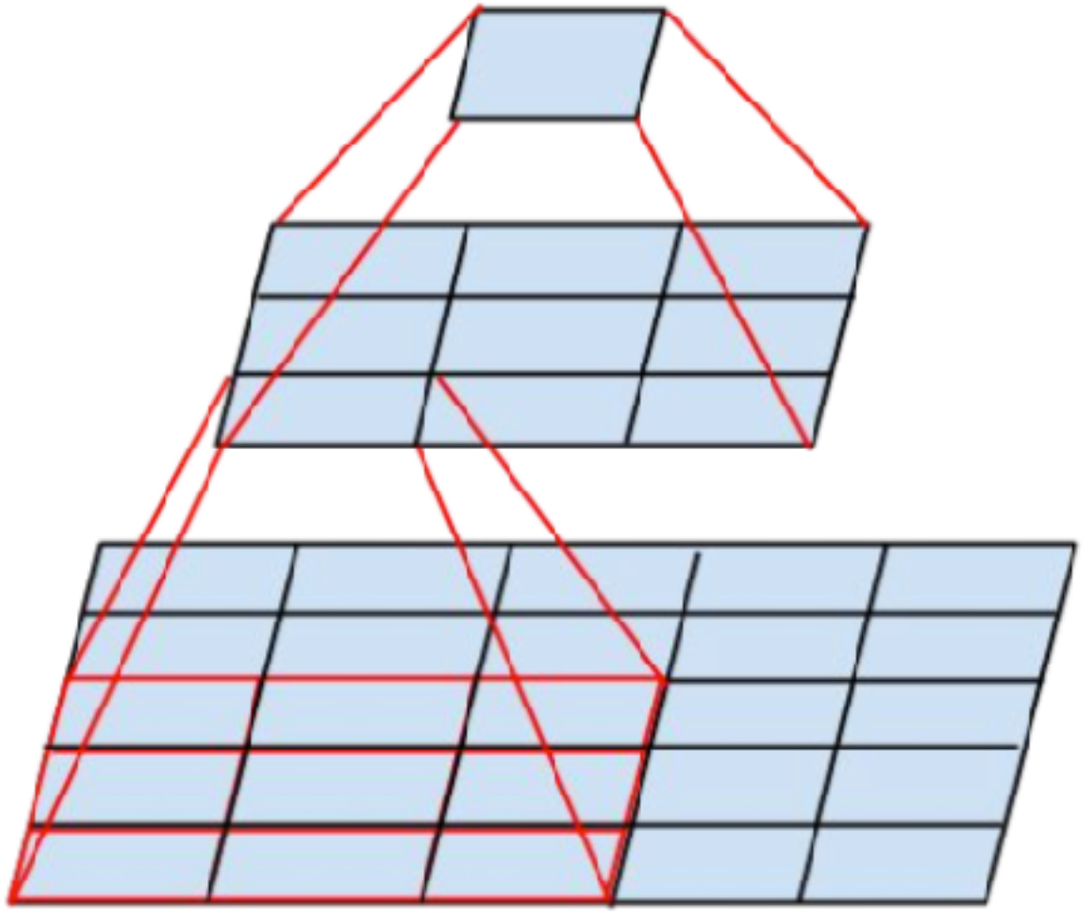
- 使用TensorFlow构建3层卷积神经网络完成MNIST识别任务（或任意一个任务），要求每层中间均使用BN层，记录模型收敛步数。

- 对比不使用BN层的模型，观察模型训练速度是否加快。

3.4 大卷积核分解为小卷积核

Author:WangQi
Email:wangqikaixin@gmail.com

使用 3 * 3卷积核代替大卷积核



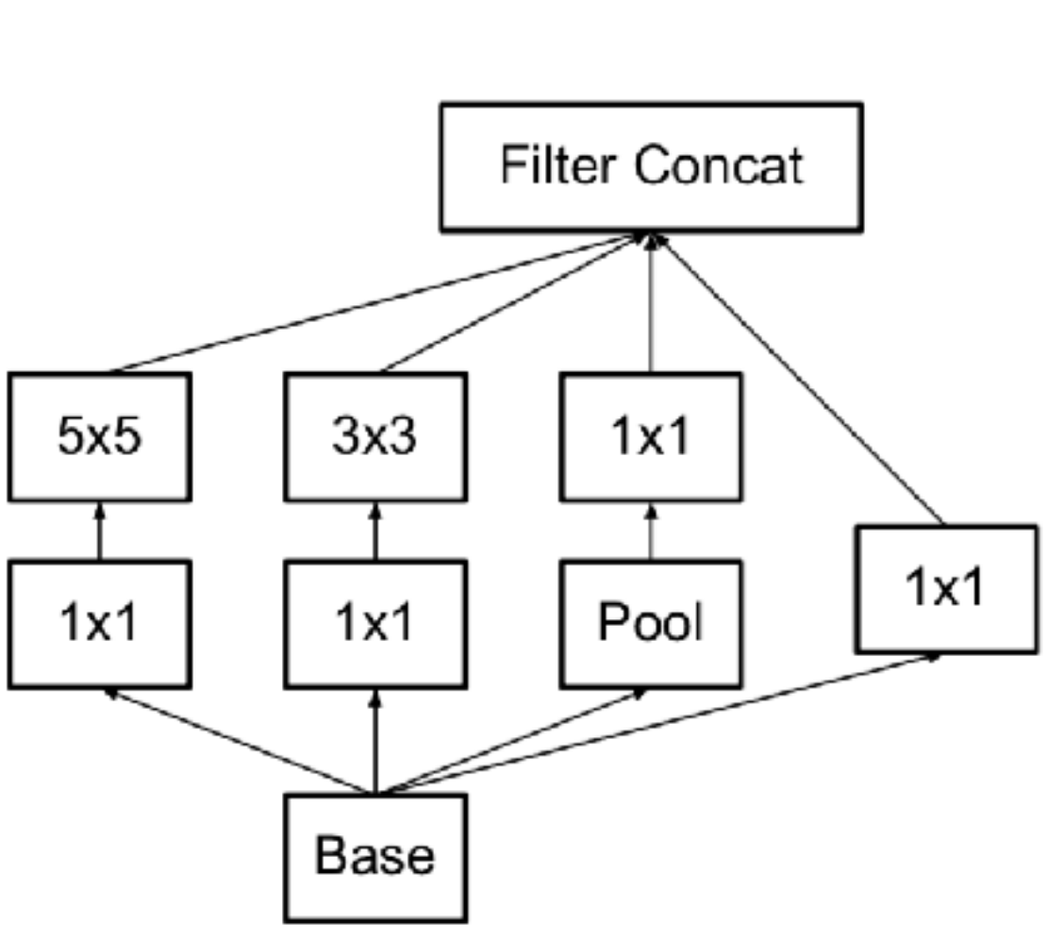
使用 3 * 3 卷积核代替 5 * 5 卷积核时：
参数量**减少28%**

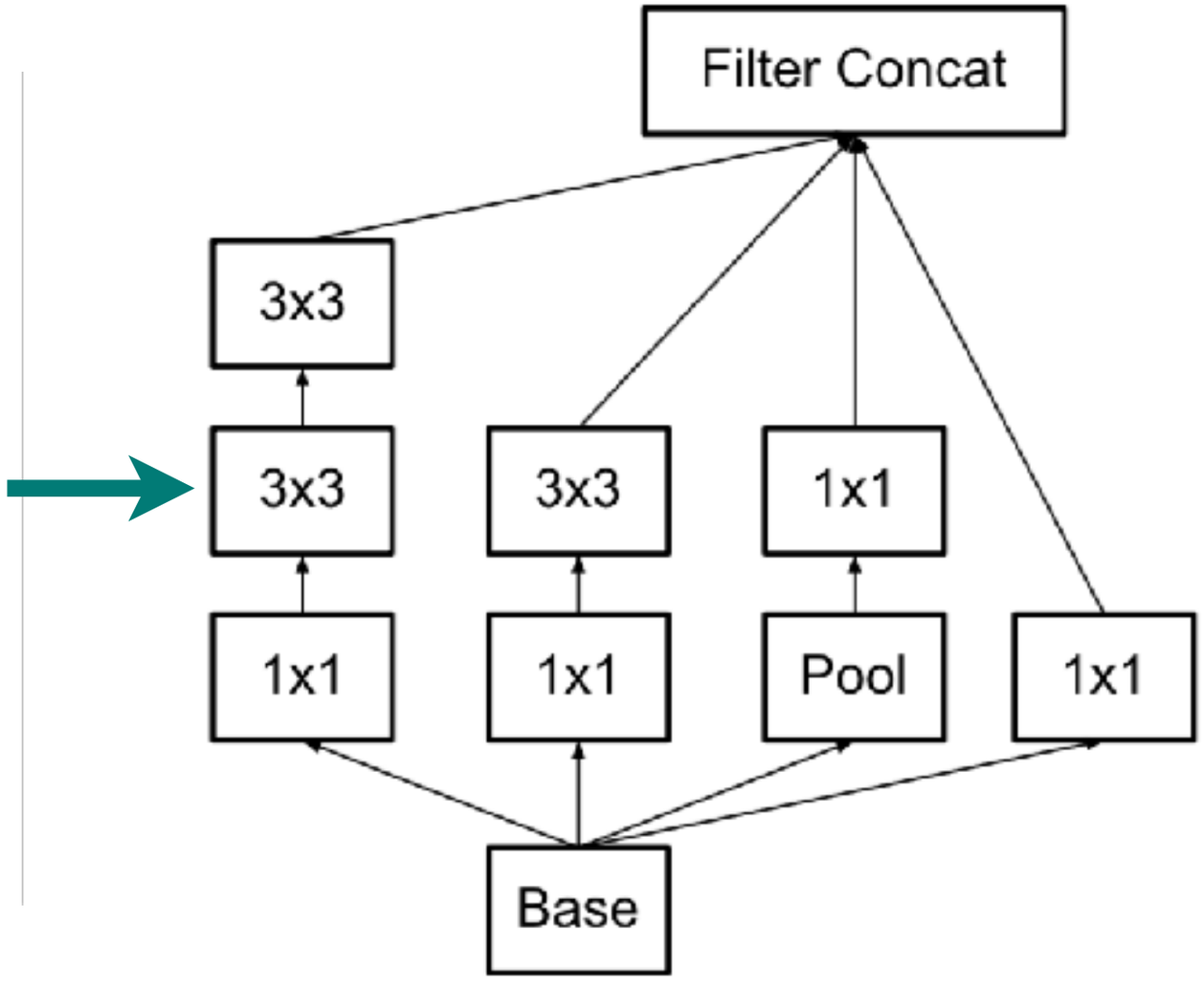
使用 3 * 3 卷积核代替 7 * 7 卷积核时：
参数量**减少45%**

这样做会有两个问题，① 会不会降低表达能力？② 3 * 3卷积之后还需要再加激活函数吗？

Author:WangQi
Email:wangqikaixin@gmail.com

使用 3 * 3卷积核代替大卷积核

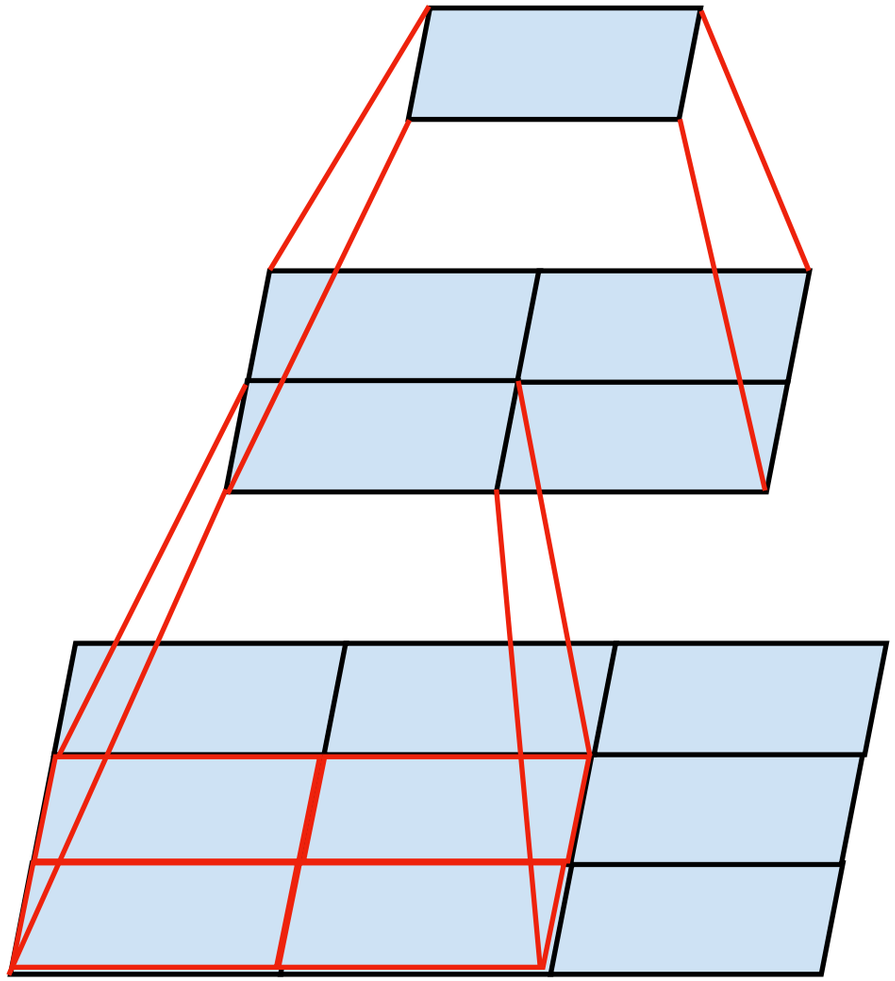




实验表明，实验3*3卷积核代替5*5卷积核不会造成性能损失，使用激活函数效果更好。

Author:WangQi
Email:wangqikaixin@gmail.com

使用2 * 2卷积核代替大卷积核

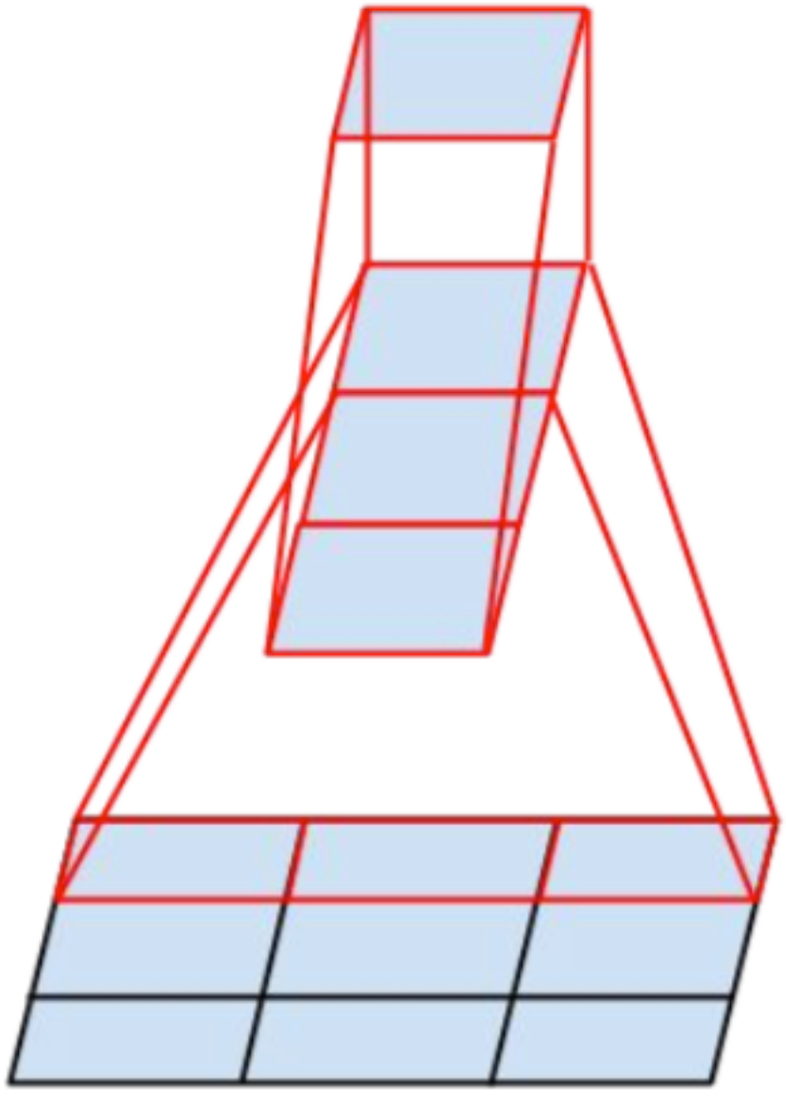


使用2个 2 * 2 卷积代替3 * 3卷积：
参数数量**减少11%**

思考：能否使用更小的卷积核代替大卷积核?

Author:WangQi
Email:wangqikaixin@gmail.com

非对称卷积



使用1个 1 * 3 卷积核1个 3 * 1 卷积代替 3 * 3 卷积:

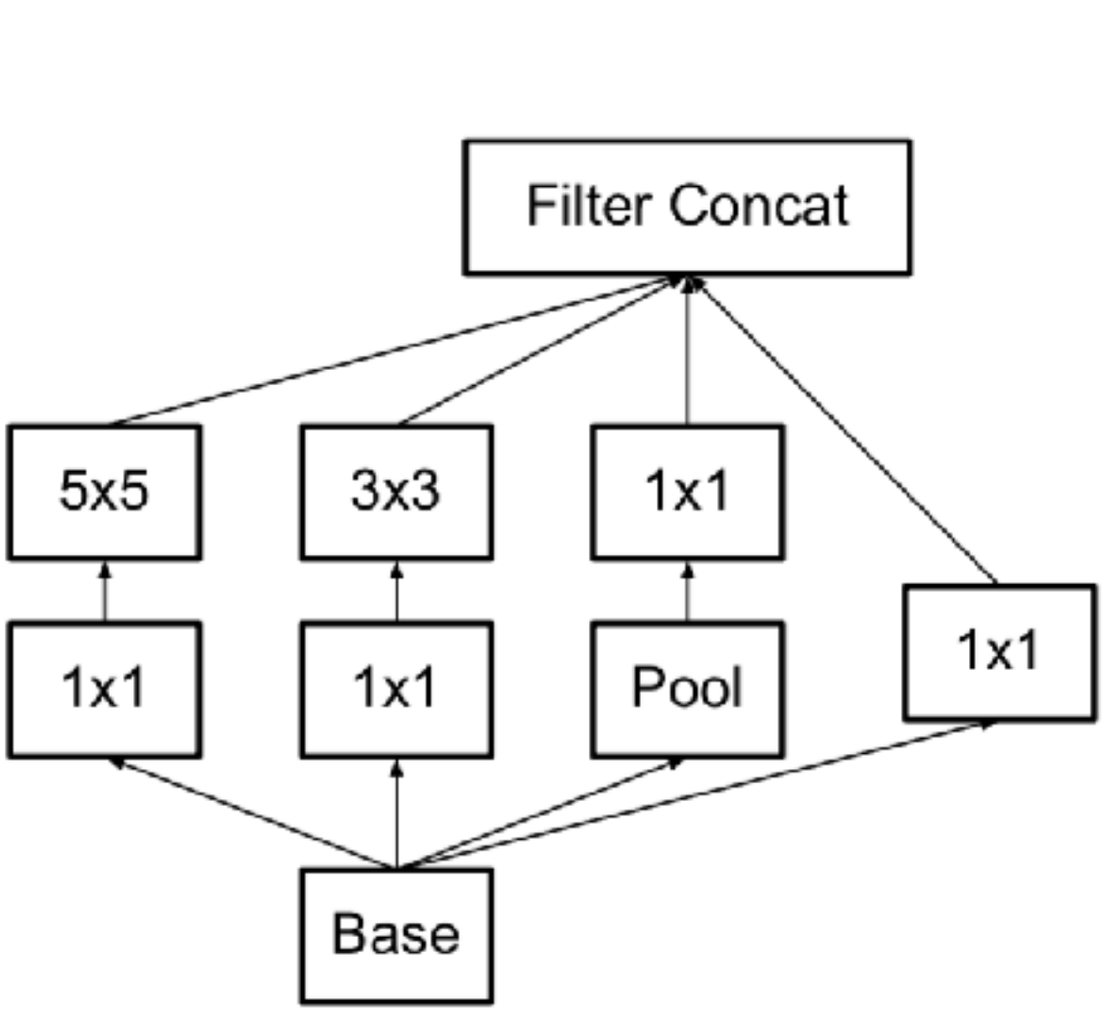
参数数量减少33%

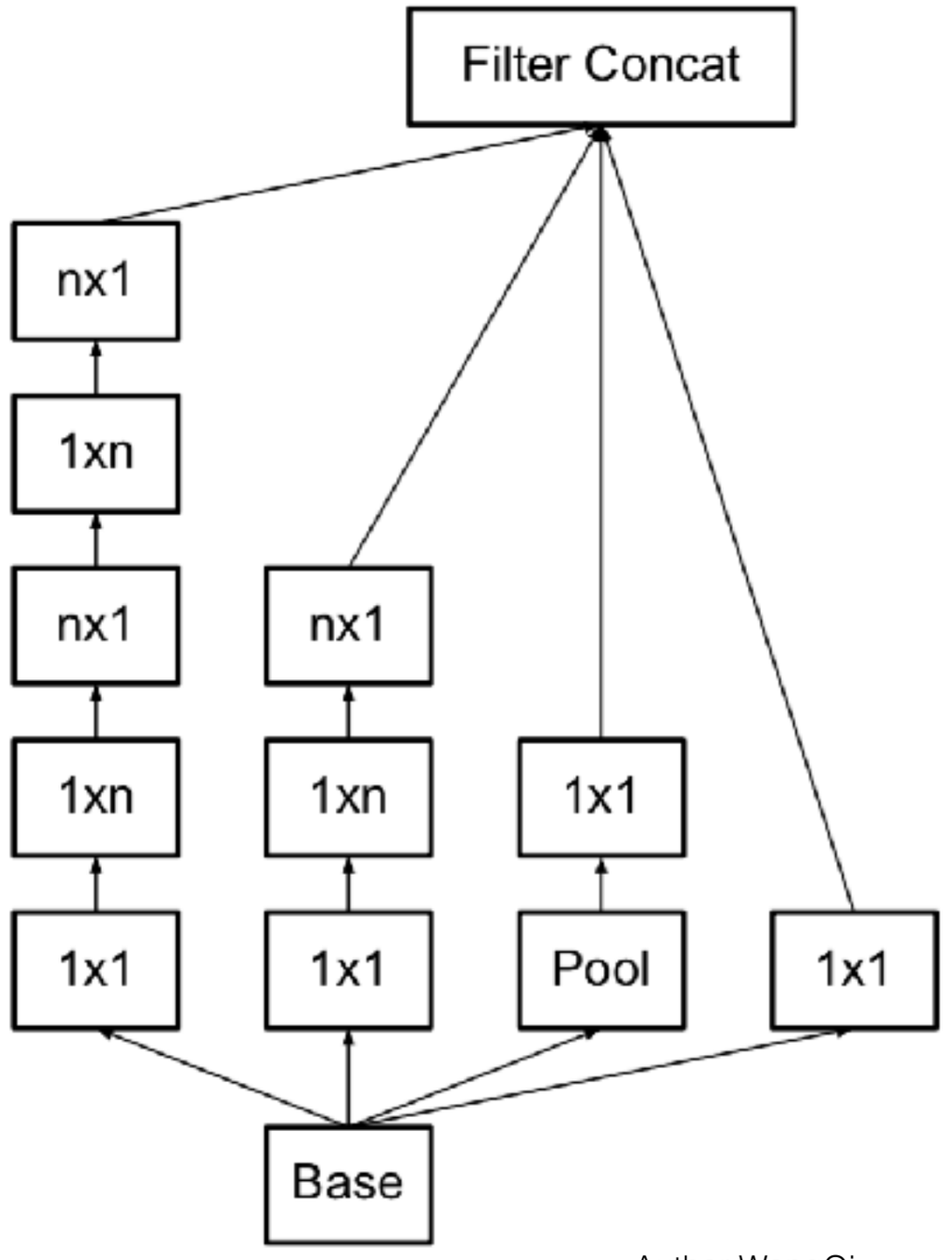
使用1个 1 * 7 卷积核1个 7 * 1 卷积代替 7 * 7 卷积:

参数数量减少71%

Author:WangQi
Email:wangqikaixin@gmail.com

非对称卷积

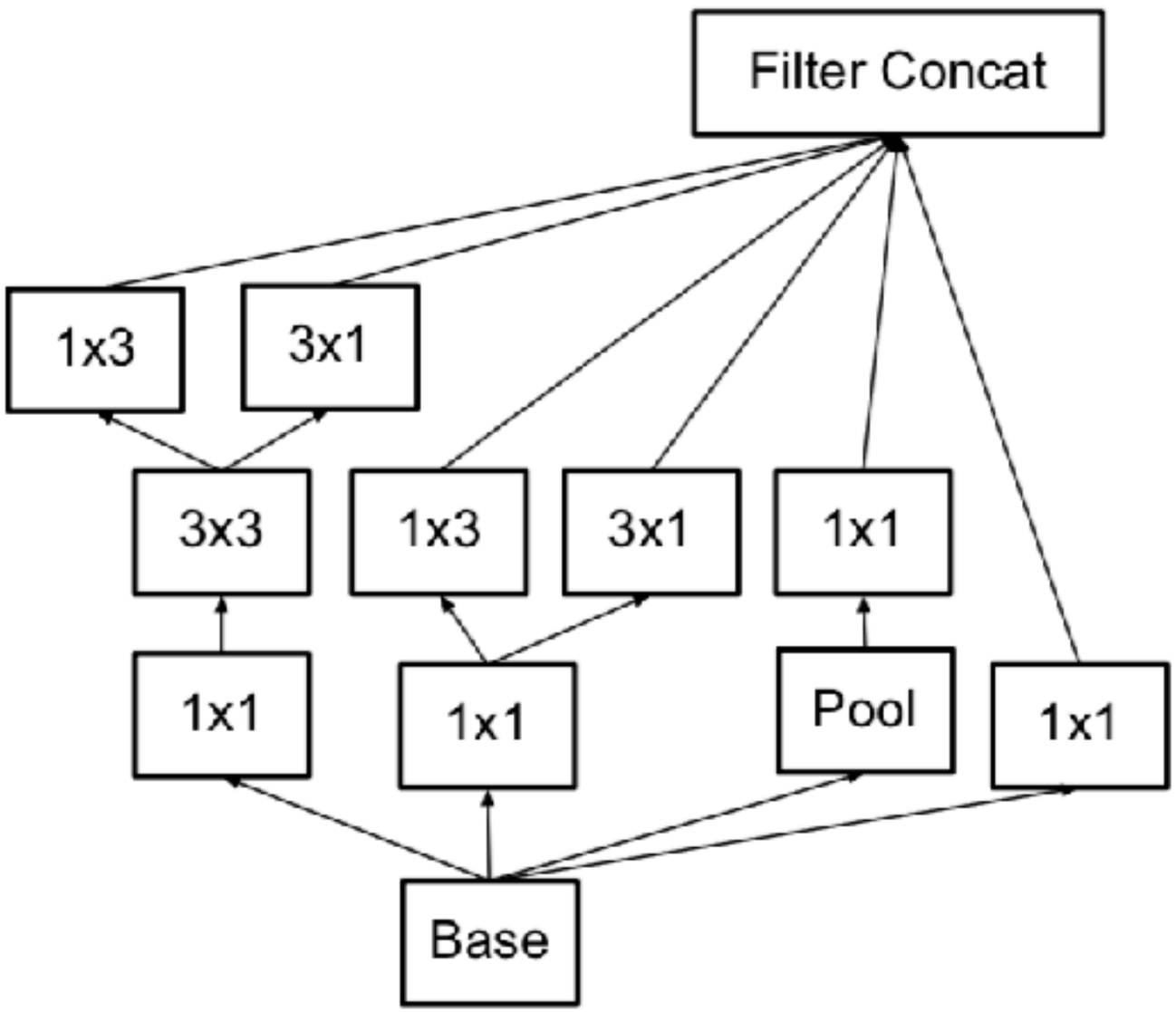




实验发现：非对称卷积在网络前面层并不能很好的发挥作用，通常在中间的12到20大小的特征图使用非对称卷积效果最好。

Author:WangQi
Email:wangqikaixin@gmail.com

使用非对称卷积提升维度



Author:WangQi
Email:wangqikaixin@gmail.com

3.5 Inception 模型结构

Author:WangQi
Email:wangqikaixin@gmail.com

Inception V2 / V3

type	patch size/stride or remarks	input size
conv	$3\times3/2$	$299\times299\times3$
conv	$3\times3/1$	$149\times149\times32$
conv padded	$3\times3/1$	$147\times147\times32$
pool	$3\times3/2$	$147\times147\times64$
conv	$3\times3/1$	$73\times73\times64$
conv	$3\times3/2$	$71\times71\times80$
conv	$3\times3/1$	$35\times35\times192$
$3\times$Inception	As in figure 5	$35\times35\times288$
$5\times$Inception	As in figure 6	$17\times17\times768$
$2\times$Inception	As in figure 7	$8\times8\times1280$
pool	8×8	$8\times8\times2048$
linear	logits	$1\times1\times2048$
softmax	classifier	$1\times1\times1000$

Figure 5 表示原始Inception，
Figure 6 表示小卷积核Inception，
Figure 7 表示非对称卷积Inception。

Author:WangQi
Email:wangqikaixin@gmail.com

InceptionNet 的其它探索

- 分析BN层的使用对性能的影响；
- 使用标签平滑降低过拟合；
- 分析高分辨率输入对模型性能的影响；
- 辅助分类器对训练的影响；
- 使用RMSPop代替SGD训练模型；
- 其它。

Author:WangQi
Email:wangqikaixin@gmail.com

InceptionNet总结

- 总结了面向计算机视觉的卷积神经网络设计原则；
- 提出非对称卷积以减少模型参数与计算量；
- 通过大量实验总结了不同结构对模型性能的贡献。

Author:WangQi
Email:wangqikaixin@gmail.com

4. ResNet

Author:WangQi
Email:wangqikaixin@gmail.com

4.1 ResNet 简介与背景

Author:WangQi
Email:wangqikaixin@gmail.com

ResNet 简介

ResNet 在ILSVRC 2015 分类任务中取得了第一名的好成绩，Top-5 error 降低至 3.57%，同时也取得了COCO 2015 检测与分割任务的第一名。ResNet提出了**深度残差学习（Deep Residual Learning）**框架，不仅极大的提高了模型训练速度，而且使得的模型深度大大增加（如100-1000层），模型性能极大提高。

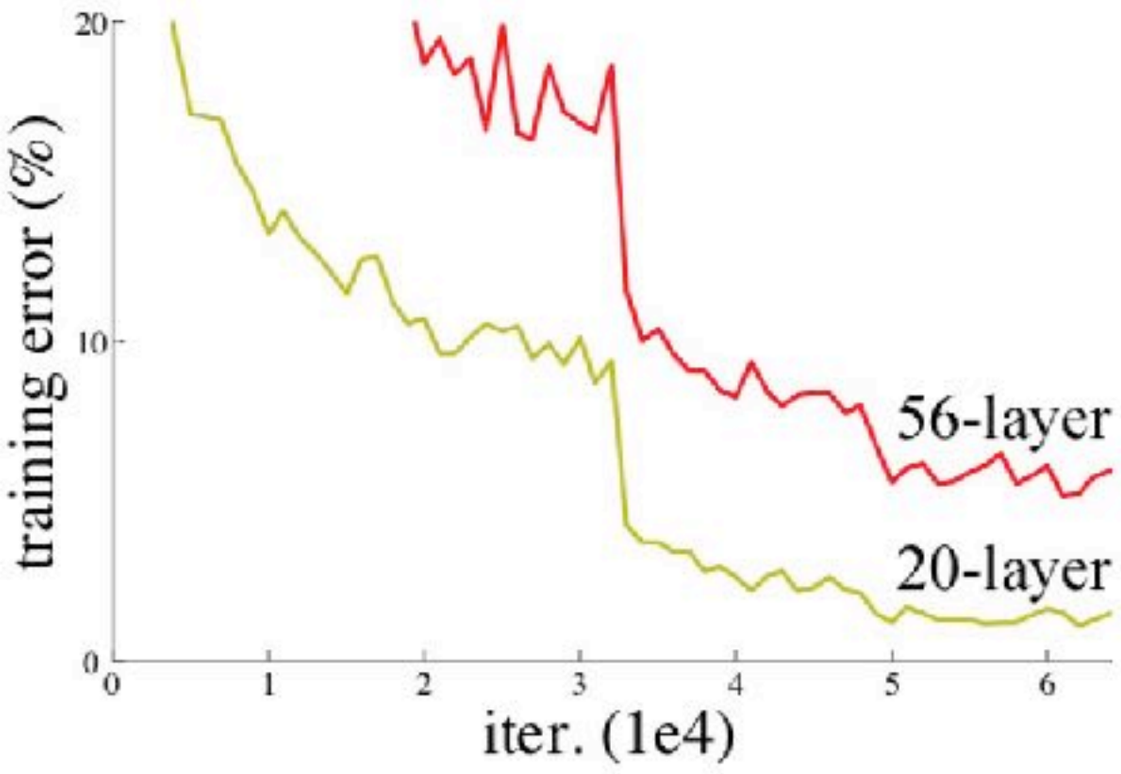
ResNet 背景

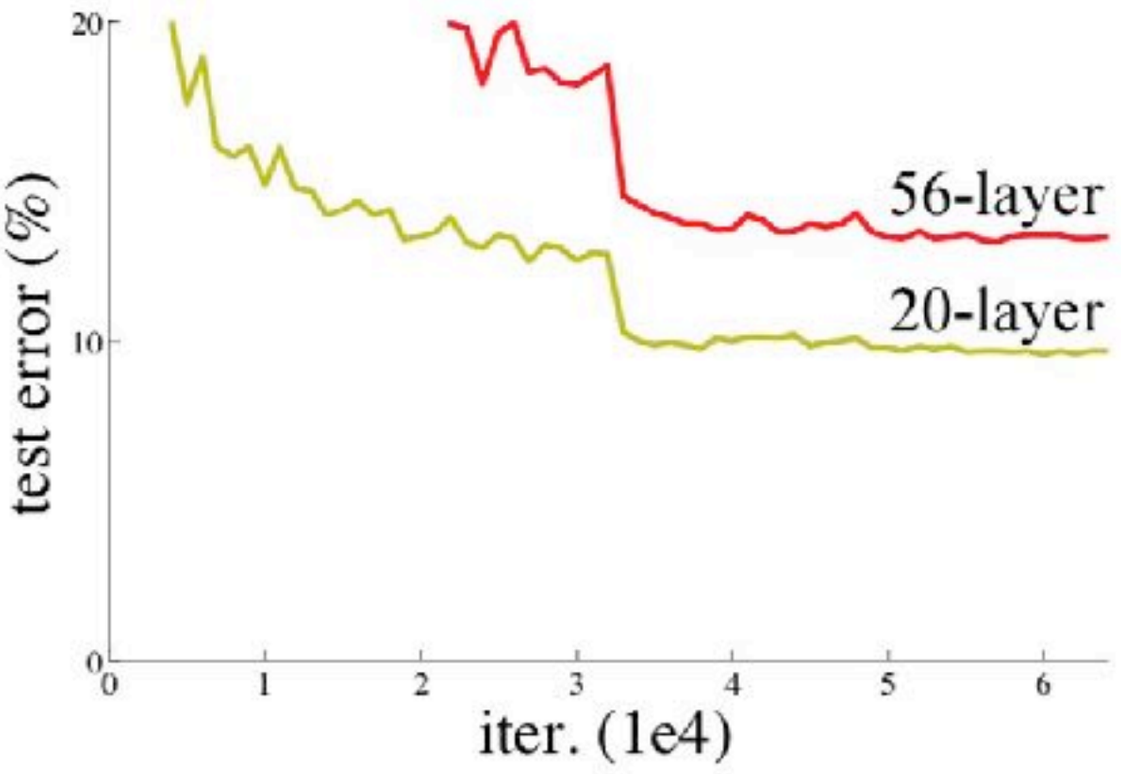
VggNet的实验表明，随着模型深度的加深，模型的性能得到了很大的提升，那么VggNet为何仅仅使用了19层的模型?

可能的原因:

- 梯度消失问题使得更深的模型难以甚至无法训练;
- 需要极大的计算资源支撑;
- 其它问题所致。

ResNet 背景





上图分别为20层与56层“plain”模型使用了Batch Norm之后，在 CIFAR-10 数据集上的表现，左图为在训练集上的表现，右图为在测试集上的表现。观测到模型出现了“退化问题”，即层数达到一定数量之后，越多的层表现越差。

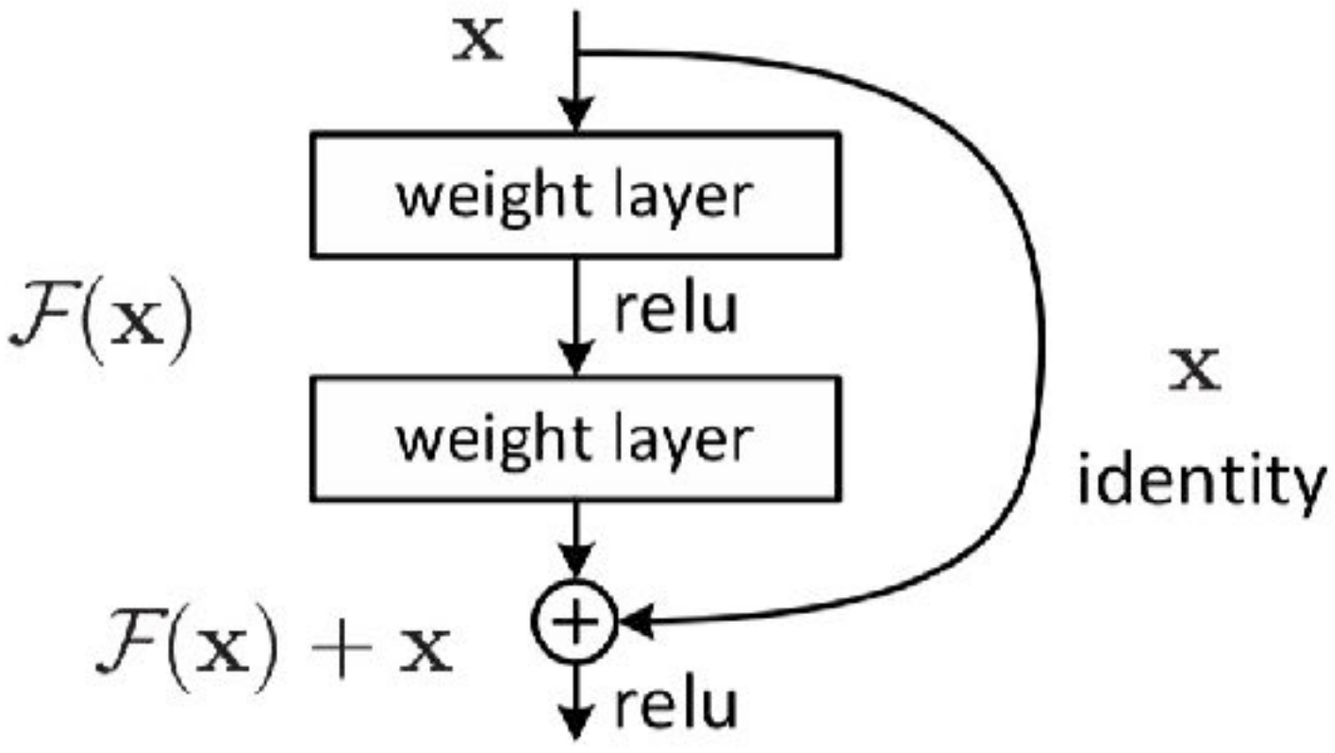
模型退化问题

- 模型退化问题并不是由梯度不稳定造成的。
- 模型退化表明并不是所有的系统都很容易优化；
- 堆叠的非线性层对拟合线性变换是有困难的；

4.2 ResNet 模型结构与主要贡献

Author:WangQi
Email:wangqikaixin@gmail.com

残差块

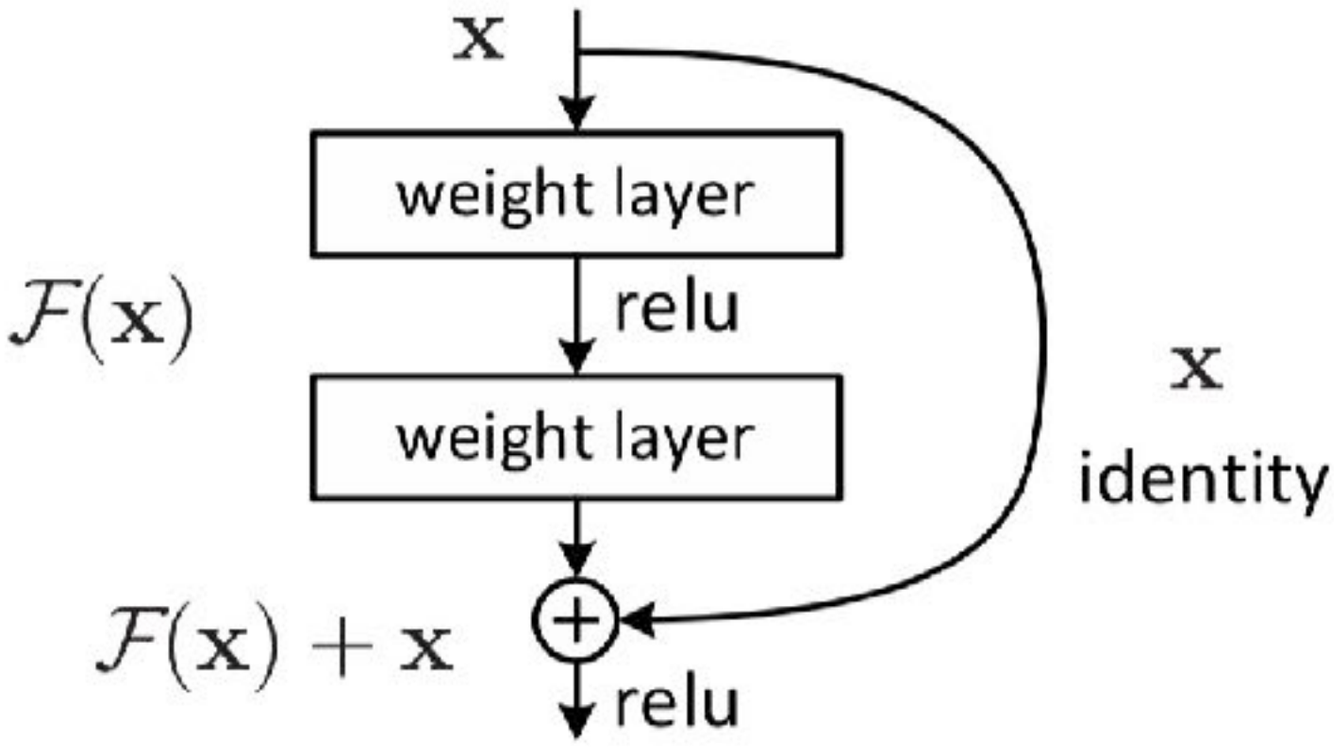


残差块增加了快捷连接，但并没有引入新的参数。

在“标准”残差结构中数据有两条通路进行流动，一条为线性通路，一条为非线性通路，在一些情况下，如果恒等映射是最优的，则非线性连接权重会被优化为接近0，此时线性连接部分发挥主要作用，这比使用堆叠的非线性层拟合identity函数要更容易的多。

Author:WangQi
Email:wangqikaixin@gmail.com

残差块



残差块中通常有大于1个的卷积层，避免了两条路线均为线性变换；

实验表明，残差块中快捷连接使用identity函数就足够了。

Author:WangQi
Email:wangqikaixin@gmail.com

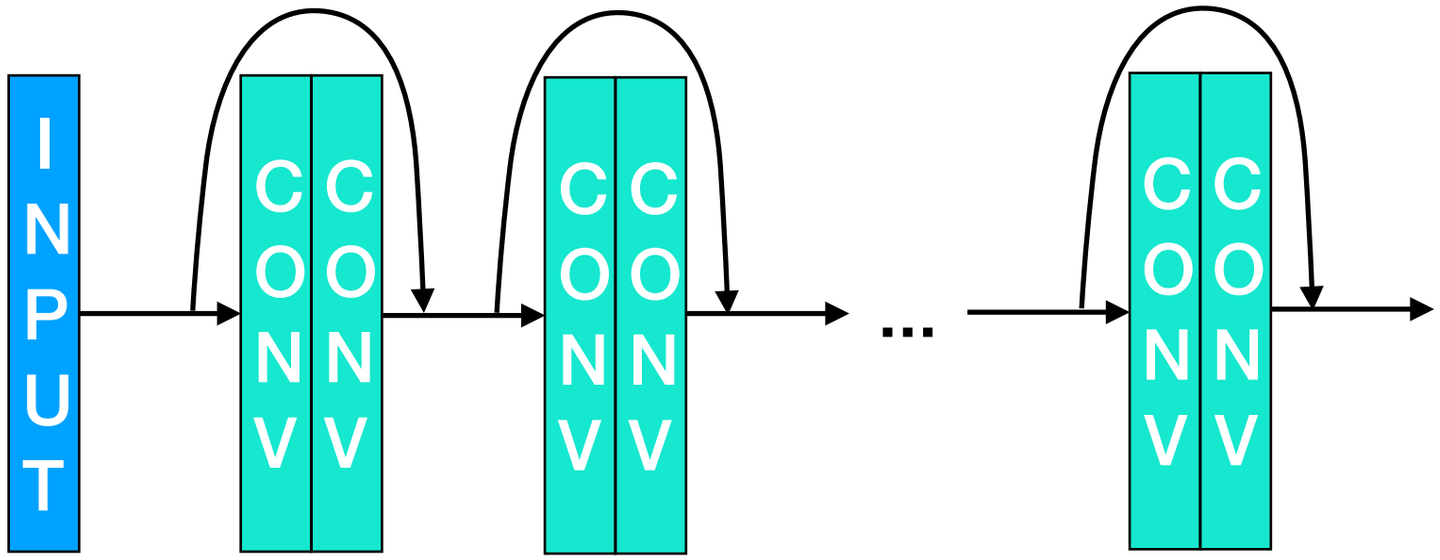
示例

使用TensorFlow实现一个残差块。

```
def res_unit(inputs, kernel_num):
    net = slim.conv2d(inputs, kernel_num, [3, 3], padding='SAME')
    net = slim.conv2d(net, kernel_num, [3, 3], padding='SAME')

    net = tf.add(net, inputs)
    return net
```

ResNet 模型基本结构



模型结构：

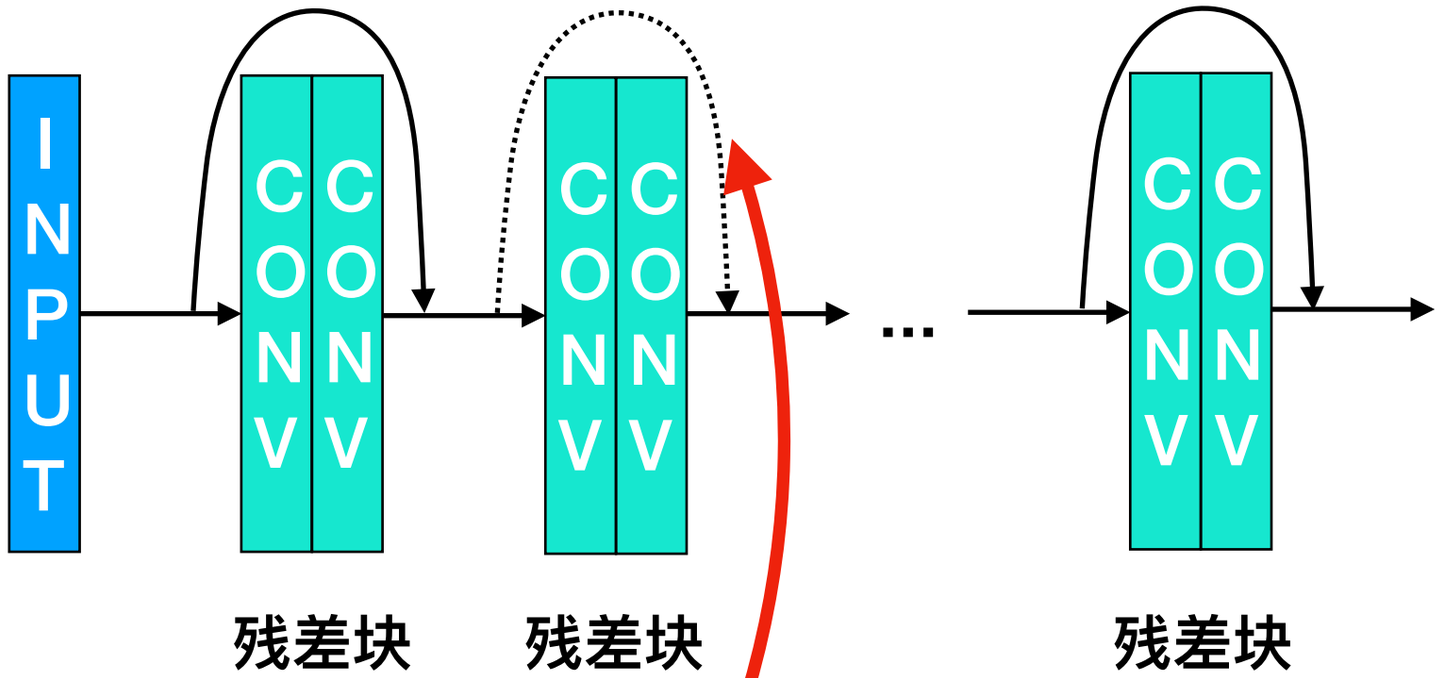
受VggNet启发，网络主要使用 3 * 3 卷积核，每两层卷积为一个残差块；
为减少模型参数，降低过拟合，ResNet去除了全连接层。

设计哲学：

对于相同输出特征图尺寸的卷积，使用相同数量的滤波器；
如果特征图尺寸减半，则卷积核数量加倍，以保证时间复杂度。

Author:WangQi
Email:wangqikaixin@gmail.com

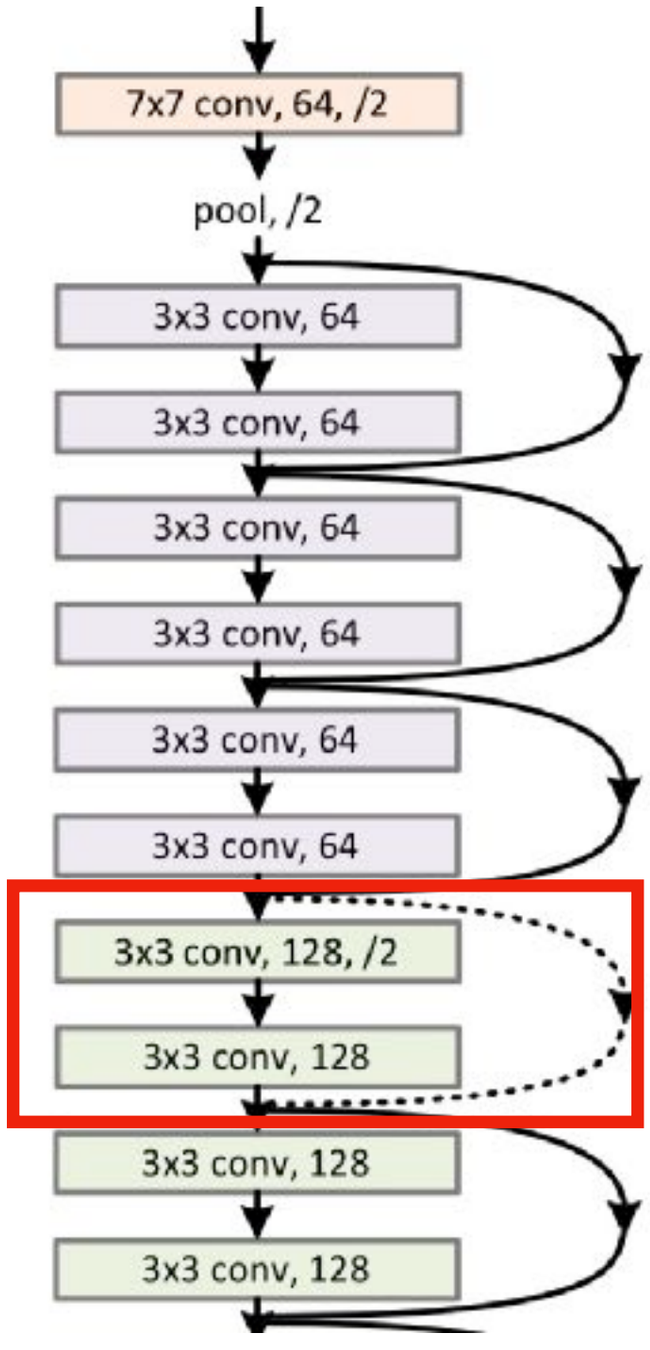
ResNet 模型基本结构



当特征图大小减小时，可使用投影映射（1*1卷积）对快捷连接进行变换。

Author:WangQi
Email:wangqikaixin@gmail.com

ResNet 模型基本结构



快捷连接需要将输入数据转换成与输出数据形状一致的数据，这里使用：

① 128个1*1卷积，2的步长对输入进行卷积即可；

② 也可使用池化与边界填充的方法对输入进行变换。

输入： [56, 56, 64]

输出： [28, 28, 128]

此处无法直接进行快捷连接。

Author:WangQi
Email:wangqikaixin@gmail.com

示例

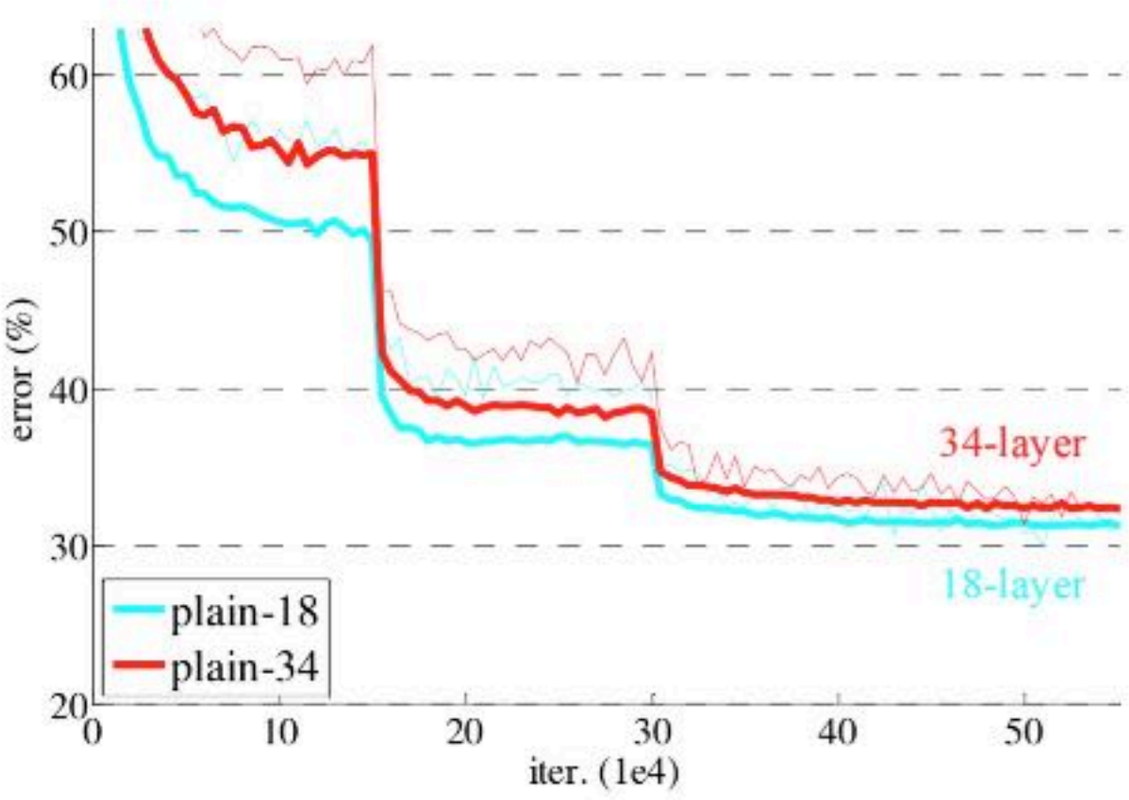
使用TensorFlow实现一个输入输出shape不等的残差块。

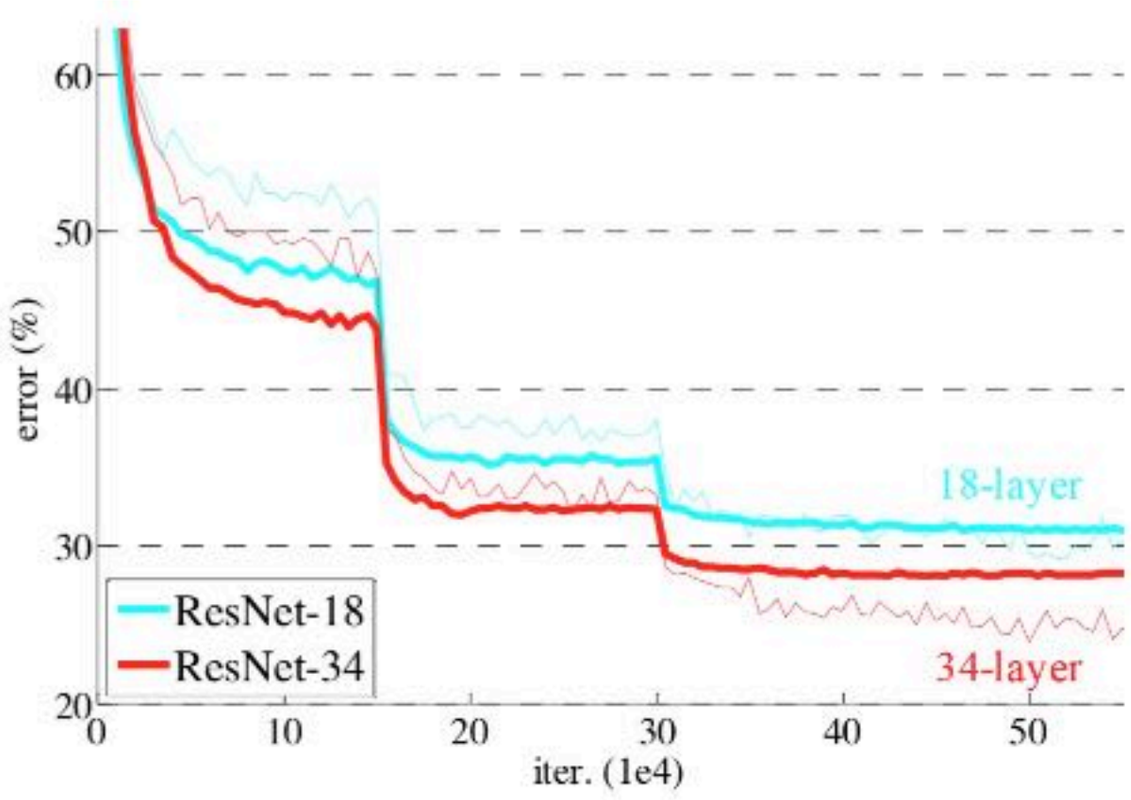
```
def res_unit(inputs, last_layer_kernel_num, kernel_num, stride=1):
    net = slim.conv2d(inputs, kernel_num, [3, 3], stride=stride,
                      padding='SAME')
    net = slim.conv2d(net, kernel_num, [3, 3], padding='SAME')

    need_trans = False
    if last_layer_kernel_num != kernel_num:
        need_trans = True
    if stride != 1:
        need_trans = True
    if need_trans is True:
        inputs = slim.conv2d(inputs, kernel_num, [1, 1],
                             padding='SAME', activation_fn=None)

    net = tf.add(net, inputs)
    return net
```

ResNet 训练结果

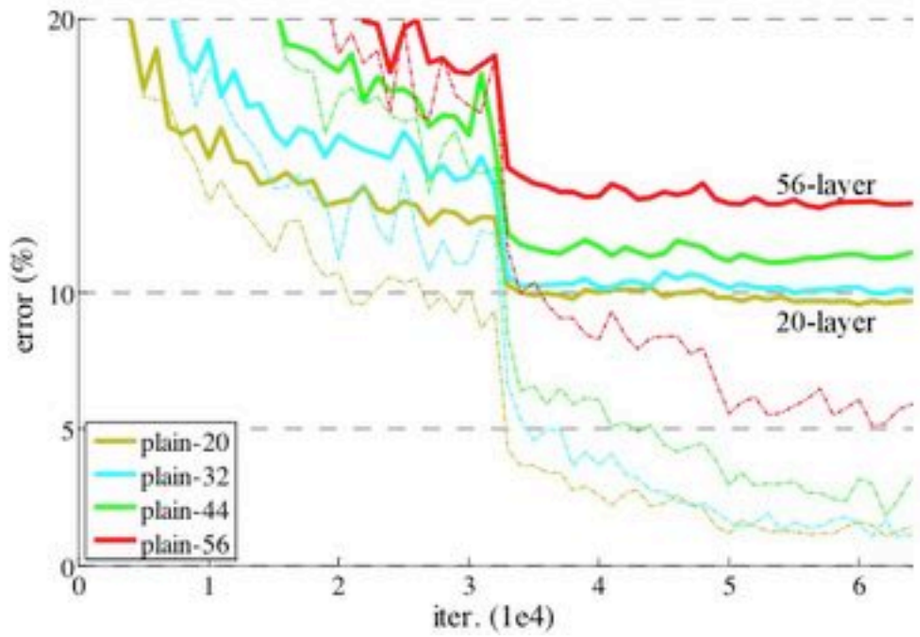


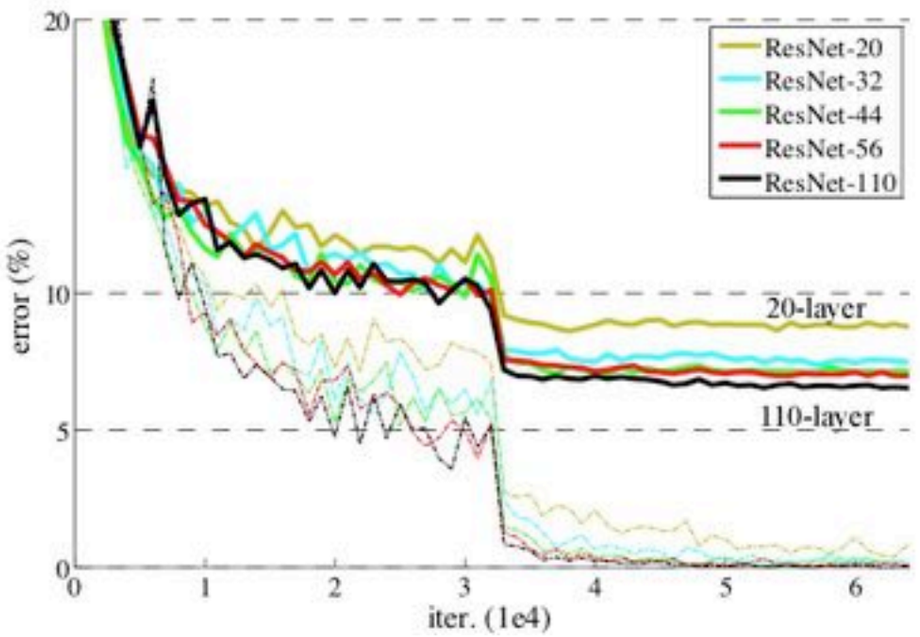


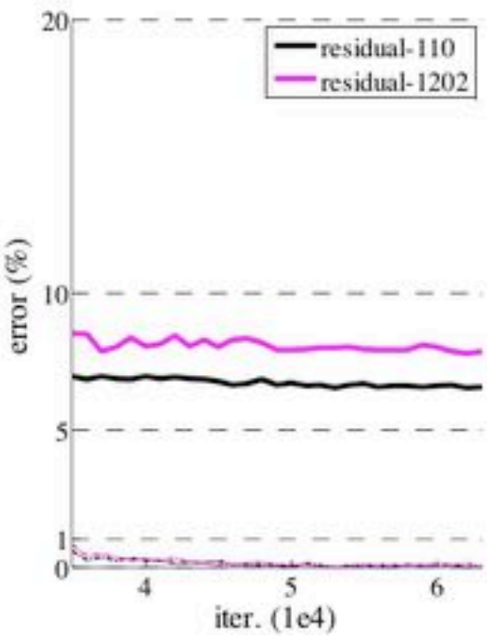
上图中细曲线表示训练误差，粗曲线表示中心裁剪图像的验证误差，数据集为ImageNet。左图为plain网络，表现出了模型退化，右图为ResNet，更深的网络性能更好，没有出现模型退化。

Author:WangQi
Email:wangqikaixin@gmail.com

ResNet 训练结果



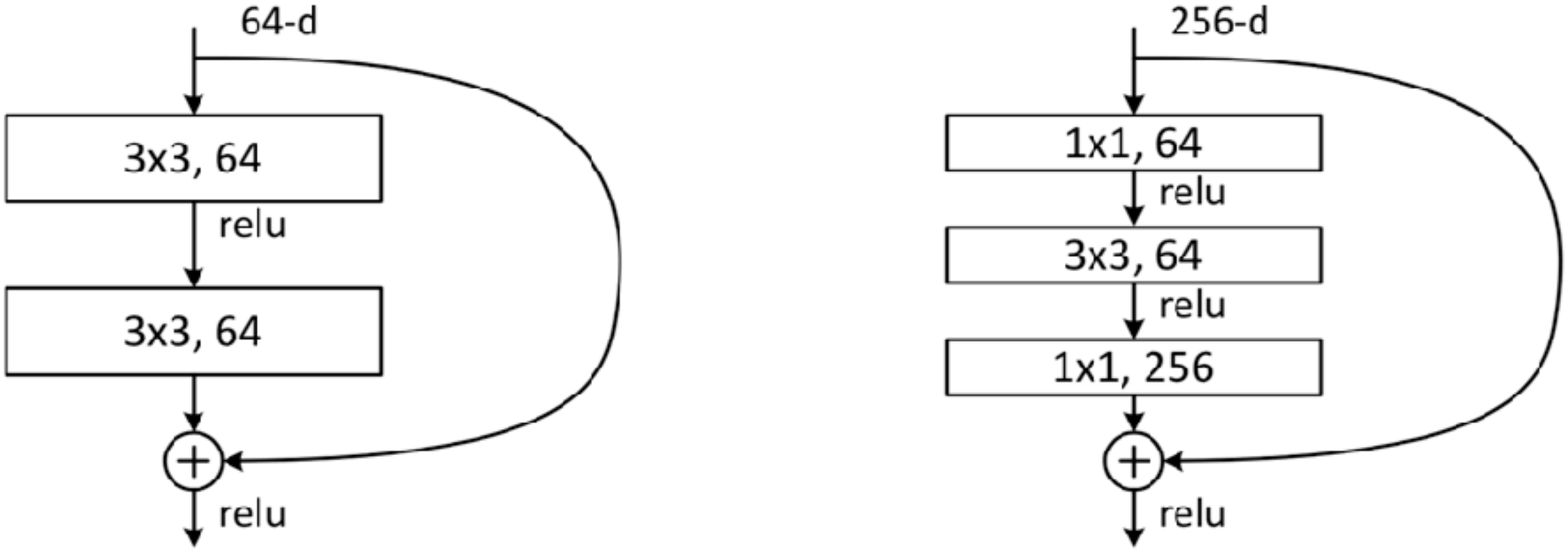




进一步的实验表明，ResNet不存在模型退化问题，可以被很好的优化。

Author:WangQi
Email:wangqikaixin@gmail.com

改进残差块



左图为普通残差块，右图为带瓶颈结构的残差块，两种残差块参数规模相等，但右图维度要比左图高很多。ResNet-50/101/152使用的是带瓶颈结构的残差块。

Author:WangQi
Email:wangqikaixin@gmail.com

示例

```
def res_unit(inputs, last_layer_kernel_num, kernel_num, stride=1):
    net = slim.conv2d(inputs, kernel_num, [1, 1], stride=stride,
                      padding='SAME')
    net = slim.conv2d(net, kernel_num, [3, 3], padding='SAME')
    net = slim.conv2d(net, kernel_num, [1, 1], padding='SAME')

    need_trans = False
    if last_layer_kernel_num != kernel_num:
        need_trans = True
    if stride != 1:
        need_trans = True
    if need_trans is True:
        inputs = slim.conv2d(inputs, kernel_num, [1, 1],
                             padding='SAME', activation_fn=None)

    net = tf.add(net, inputs)
    return net
```

带瓶颈的残差块优点

- 快捷连接连接着两个高维端，避免违反了表示瓶颈原则；
- 参数量与计算量比无瓶颈结构更少；
- 引入更多的非线性层，提高了模型的表示能力。

Author:WangQi
Email:wangqikaixin@gmail.com

ResNet 模型结构

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				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	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

Author:WangQi
Email:wangqikaixin@gmail.com

实验

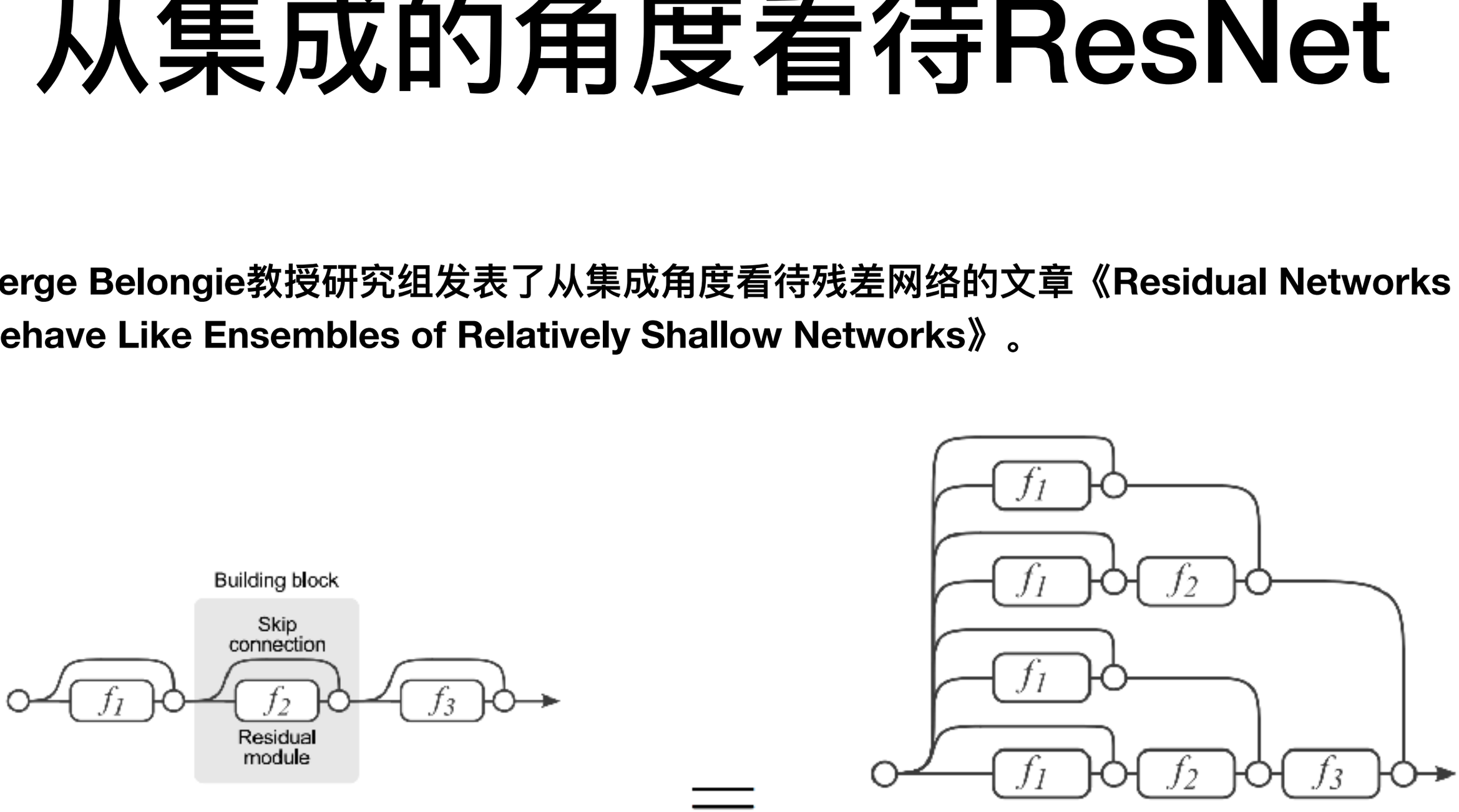
根据上述模型结构，使用TensorFlow实现ResNet-18，要求在卷积之后添加BatchNormal层。

4.3 ResNet新的解释

Author:WangQi
Email:wangqikaixin@gmail.com

从集成的角度看待ResNet

Serge Belongie教授研究组发表了从集成角度看待残差网络的文章《Residual Networks Behave Like Ensembles of Relatively Shallow Networks》。



(a) Conventional 3-block residual network　　(b) Unraveled view of (a)

上图是把一个三个残差单元串联的网络展开的示意图，每个单元都有两条路径，所以网络展开后有8条路径。

Author:WangQi
Email:wangqikaixin@gmail.com

测试ResNet与集成关系的方法

- 随机去掉某一层。模型集成中随着模型数量的减少性能平滑的降低。验证关键：

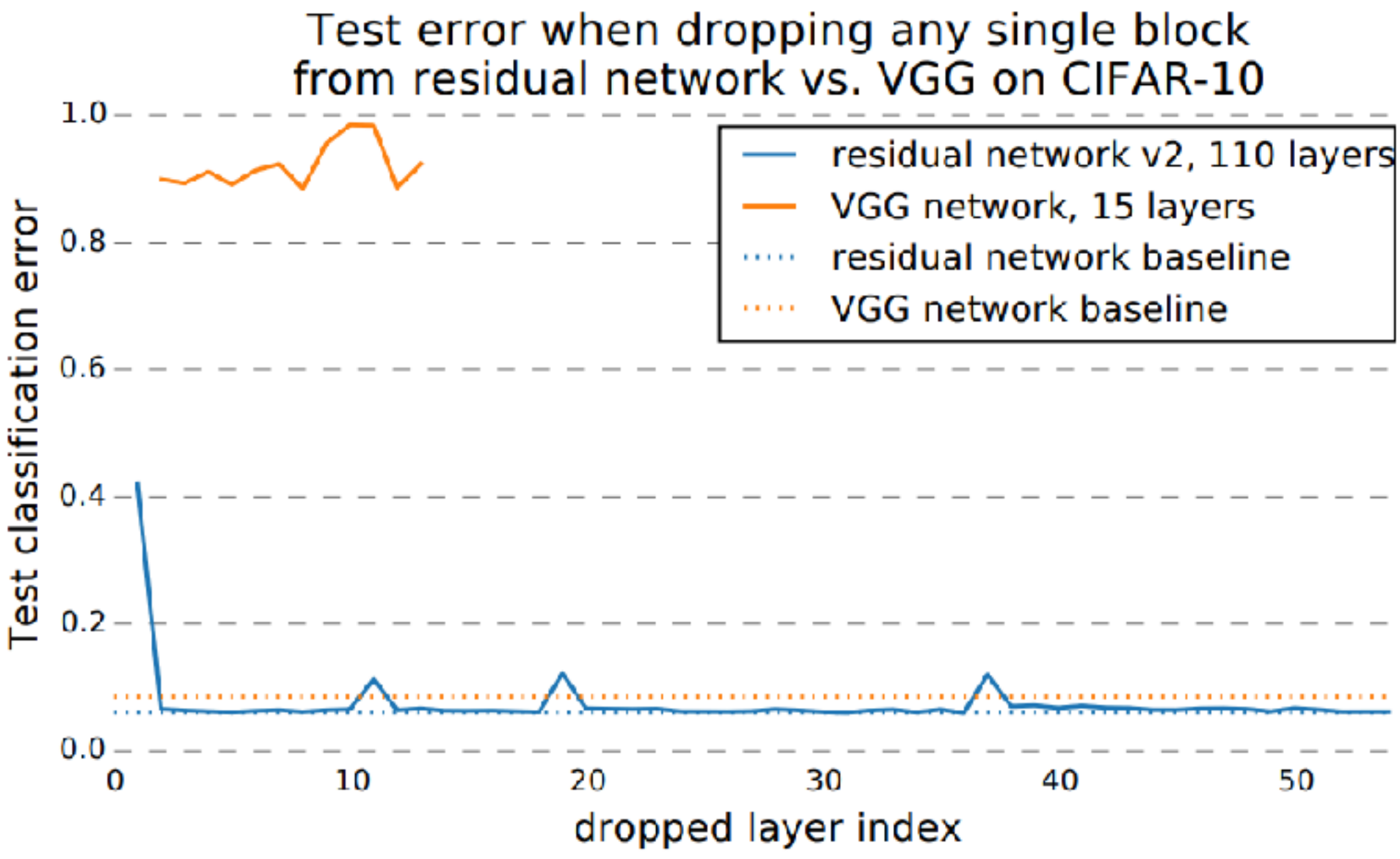
  ① 随机去掉一层，对性能影响不大；

  ② 随着层的减少性能逐渐减低。

- 随机交换某些层。交换层等价于破坏了部分子模型的输入。验证关键：随着重组程度增加，性能平滑下降。

Author:WangQi
Email:wangqikaixin@gmail.com

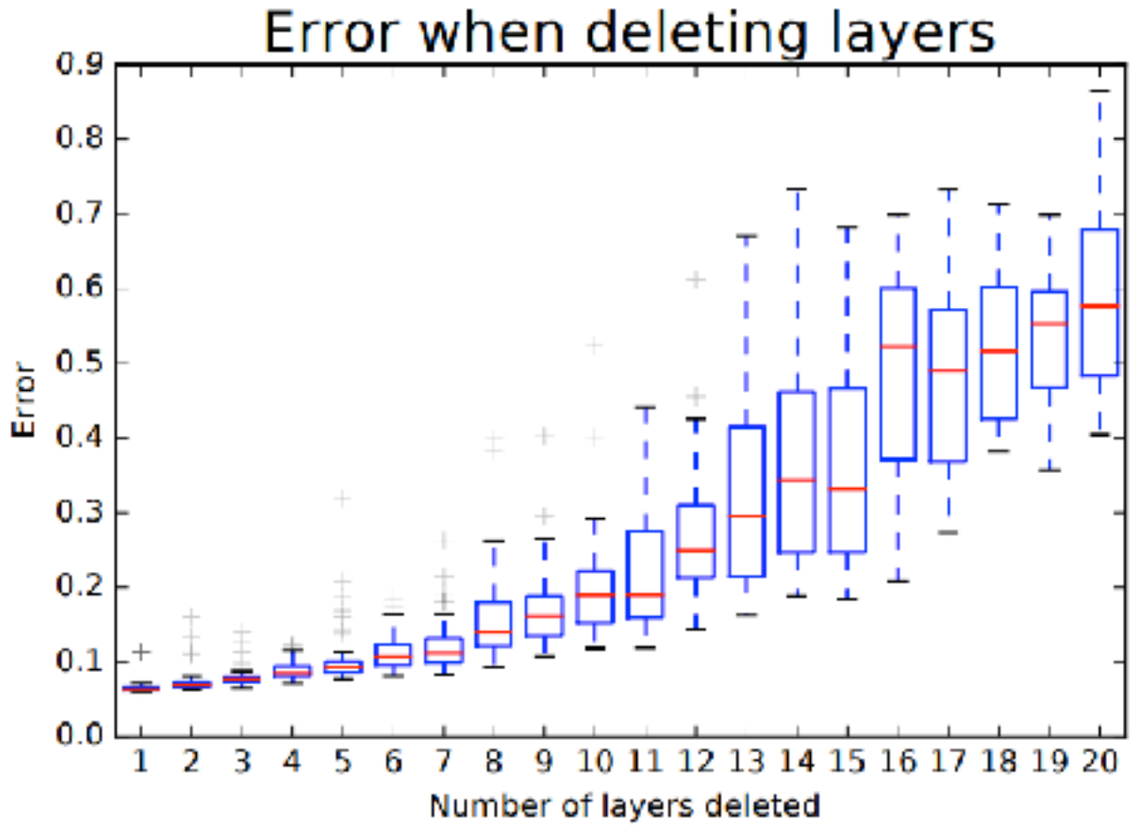
随机去掉一层



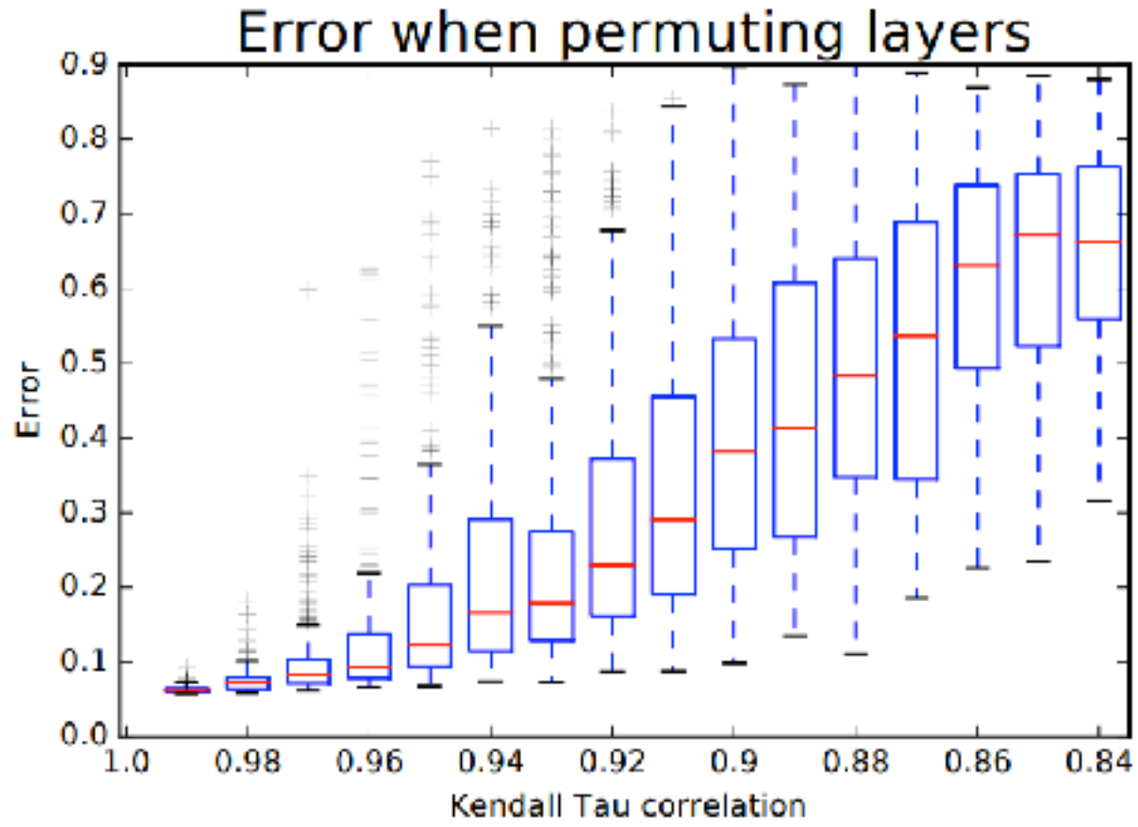
VggNet中随机去掉任何一层其均表现出随机结果，约10%正确率，而ResNet中除了去掉池化层影响较大以外，对模型性能几乎不影响。

Author:WangQi
Email:wangqikaixin@gmail.com

去掉或重组某些层



(a)



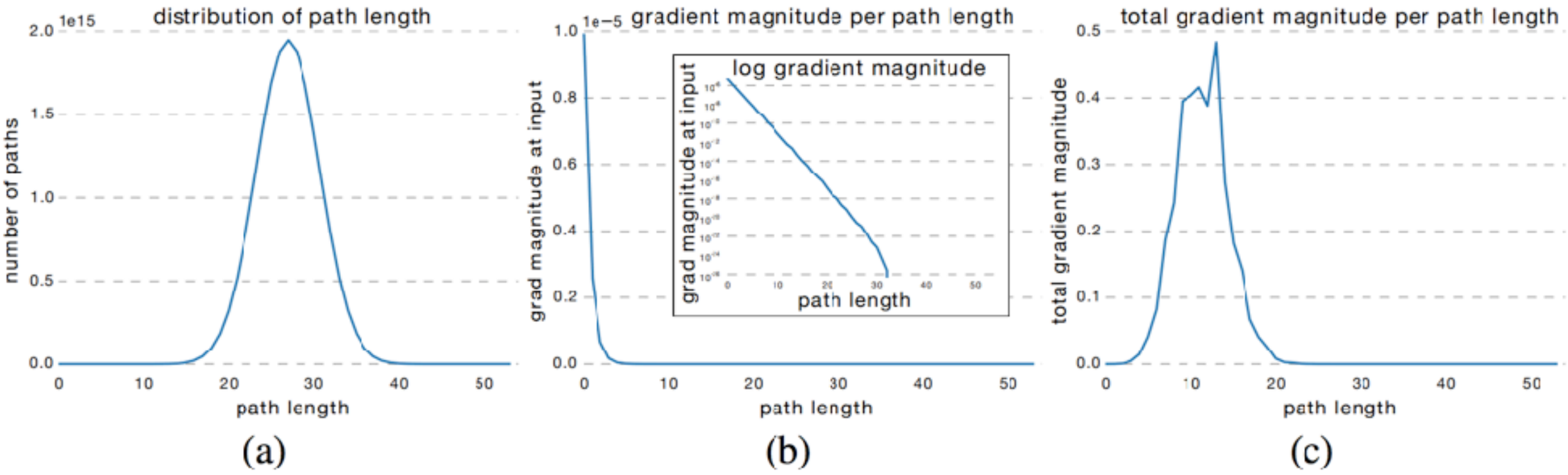
(b)

随着去掉或重组的层增多，模型性能平滑下降。左图为去掉层的实验结果，右图为重组层的实验结果。

Author:WangQi
Email:wangqikaixin@gmail.com

实验结果表明：ResNet与模型集成的效果类似，层数越多，集成的模型越多。但仍然没有证明ResNet是否解决了深层网络反向传播的问题，即没有证明在“集成”的模型中“短路径”与“长路径”模型谁对梯度贡献较大。

短路径对梯度的贡献更大



上图为54个残差块构成的模型。其中（a）为不同长短路径的数量分布；（b）为不同长度路径所对应的第一层中梯度大小；（c）表示路径长度与对梯度大小贡献的关系。可以看到：对于较长的路径对梯度的贡献并不大。

ResNet与模型集成的关系

- ResNet可以看做是模型集成的结果；
- ResNet并没有直接解决模型退化问题；
- ResNet与模型集成的关系表明不一定更深的网络就是更好的网络，为此诞生了一些更宽的网络，其表现比ResNet更优。

4.4 ResNet 总结

Author:WangQi
Email:wangqikaixin@gmail.com

ResNet 总结

- 残差块相比普通卷积增加了跨层线性连接，为线性变换提供了路径；
- 带瓶颈的残差块优化了残差块的结构；
- ResNet同样去掉了全连接层，使用全局平均池化替代；
- ResNet模型结构简单，扩展方便，性能很好，是构建模型、迁移学习的常用模型。

Author:WangQi
Email:wangqikaixin@gmail.com

实验

- 使用TensorFlow实现ResNet-50与ResNet-101，并尝试使用ResNet-50或其改进模型完成cifar-10分类任务。

- 计算两个模型参数数量，看看101层的模型比50层的模型增加了多少参数？与AlexNet对比如何?

Author:WangQi
Email:wangqikaixin@gmail.com

小结

- NIN 模型提出了MlpConv与GAP，在保证模型性能的同时使得参数数量与计算量极大的降低。

- GoogLeNet提出并改进了Inception模块，使得卷积层进一步的变得稀疏。

- InceptionNet总结了神经网络模型在计算机视觉上的设计原则并设计了非对称卷积。

- ResNet提出了深度残差学习框架，间接避免了模型退化问题。

- 1*1卷积贯穿了上述所有模型，其特点是带来的参数量少，可以交换通道信息，对通道进行升降维度。

Author:WangQi
Email:wangqikaixin@gmail.com

THANKS

Author:WangQi
Email:wangqikaixin@gmail.com