卷积神经网络的压缩技术

讲师 龚轩

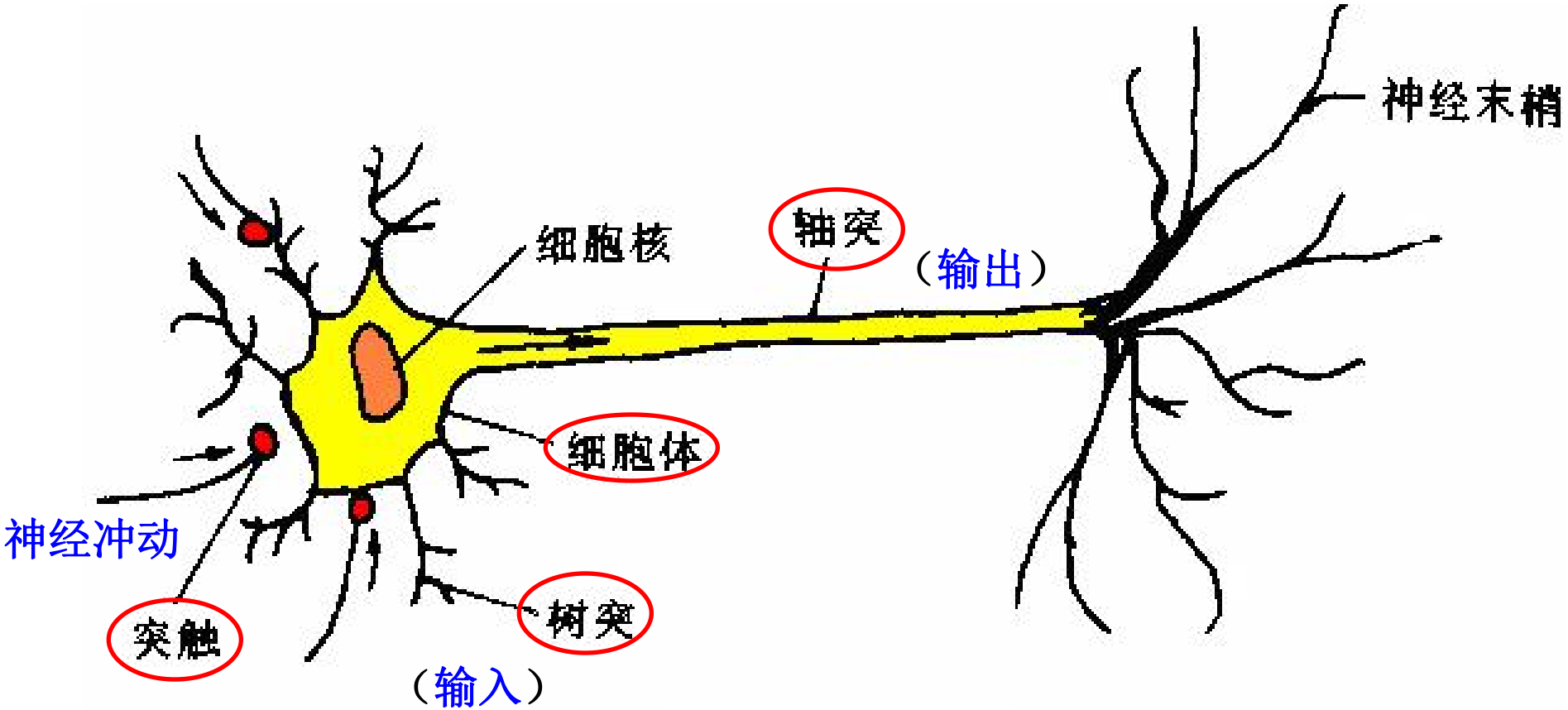
CONTENTS

目录

认识神经网络

- 生物神经元结构及数学模型
- 从BP神经网络到卷积神经网络
- 深度神经网络面临的问题
- 近几年神经网络的统计数据

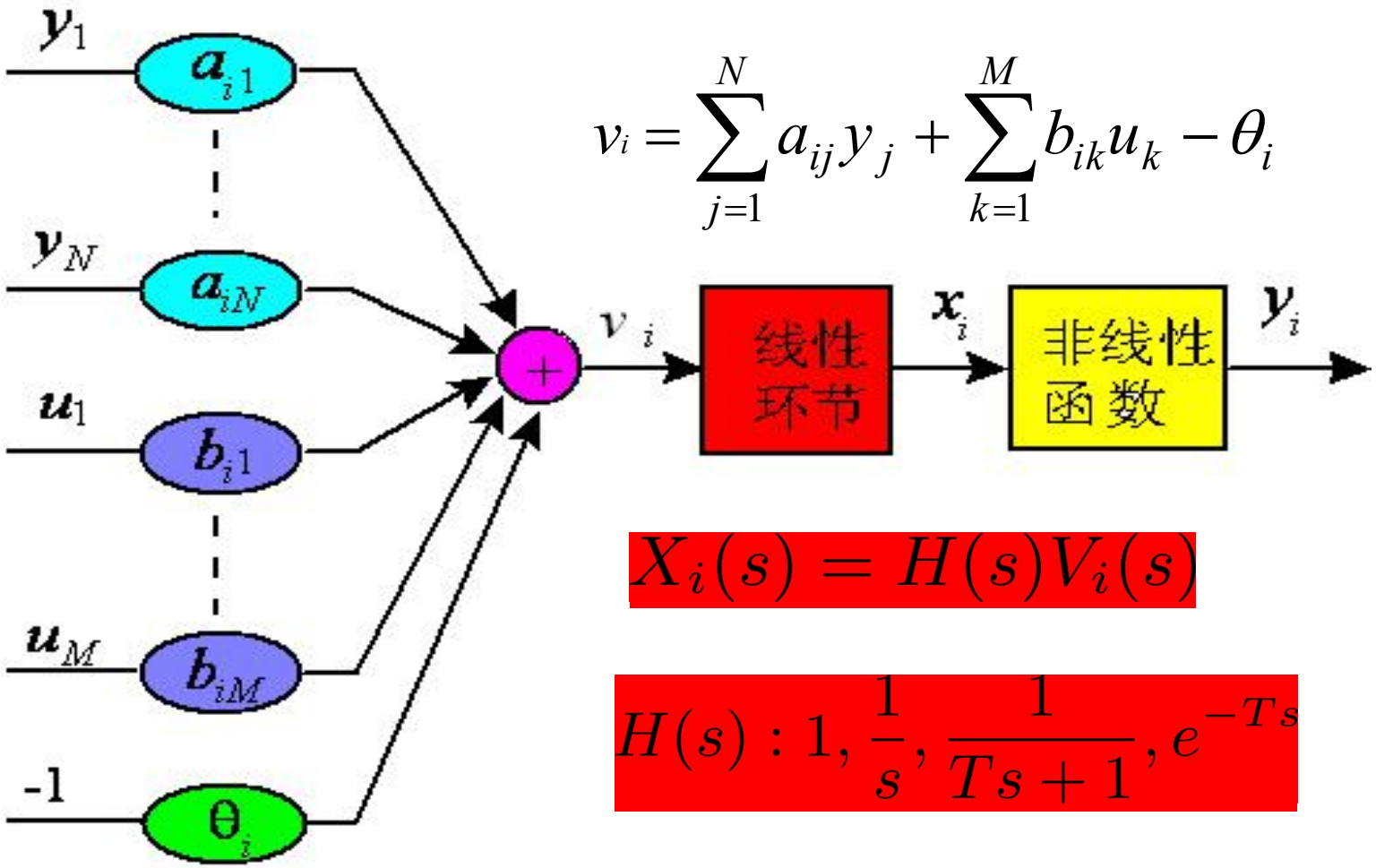
生物神经元的结构



- 神经元的状态：兴奋、抑制

神经元数学模型

- 卷积1943年，麦克洛奇和皮兹提出M—P模型。一般模型

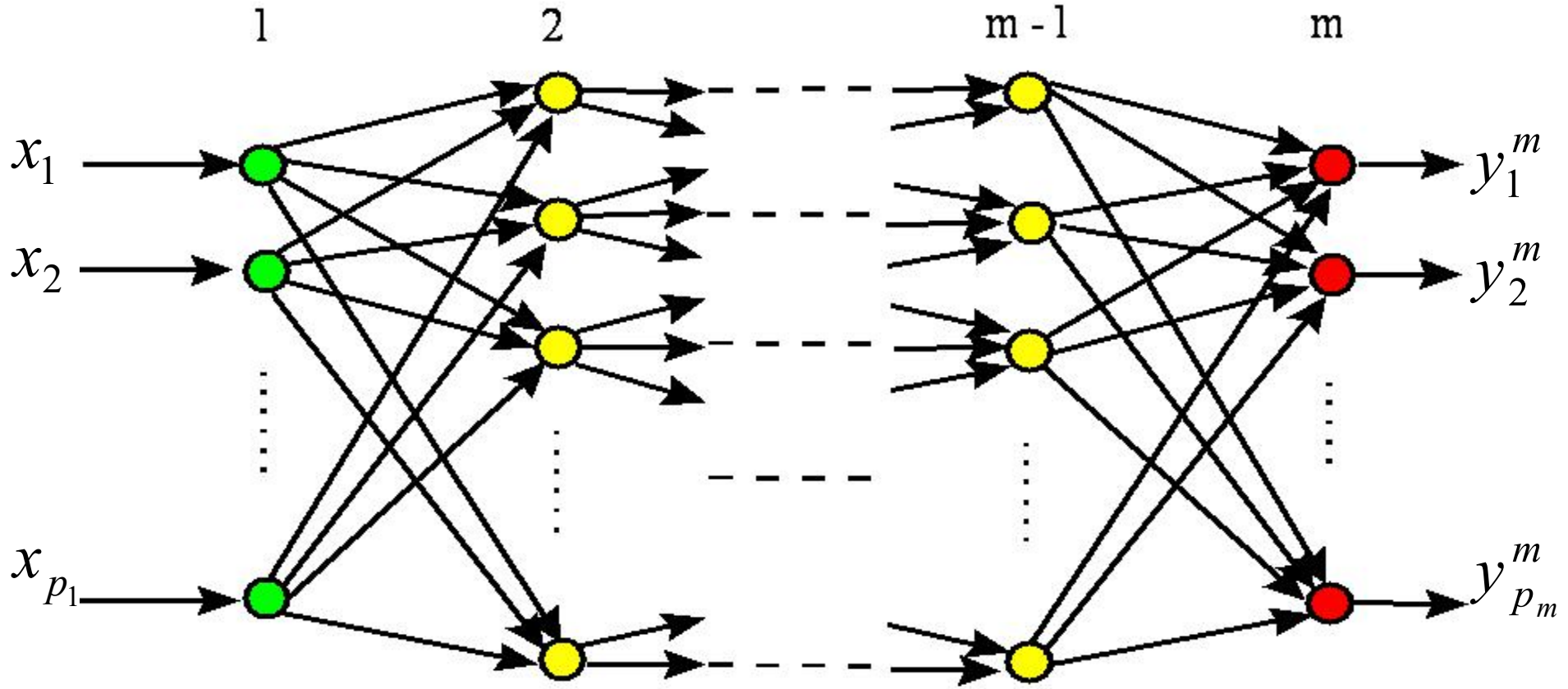


$$v_i = \sum_{j=1}^{N} a_{ij} y_j + \sum_{k=1}^{M} b_{ik} u_k - \theta_i$$

$$X_i(s) = H(s) V_i(s)$$

$$H(s) : 1, \frac{1}{s}, \frac{1}{Ts+1}, e^{-Ts}$$

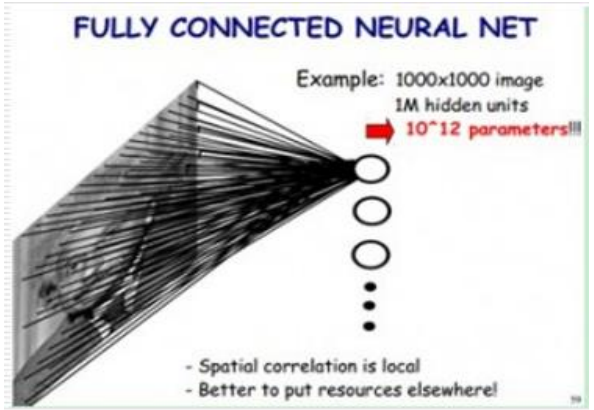
从BP神经网络到卷积神经网络 – BP网络结构



$$X = [x_1 \quad x_2 \quad \cdots \quad x_{p_1}]^T$$

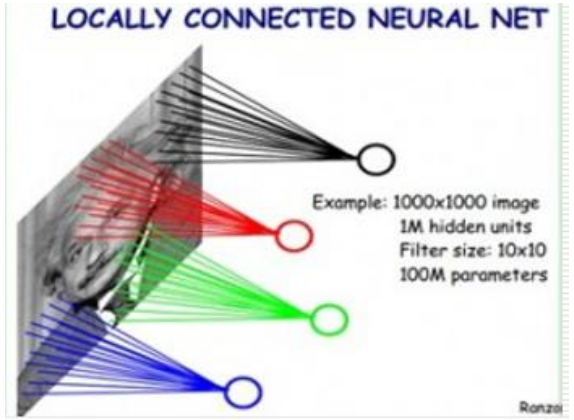
$$Y = [y_1^m \quad y_2^m \quad \cdots \quad y_{p_m}^m]^T$$

从BP神经网络到卷积神经网络 – BP神经网络的改进



全局感知

全连接结构，参数过多

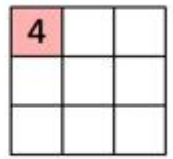


局部感知

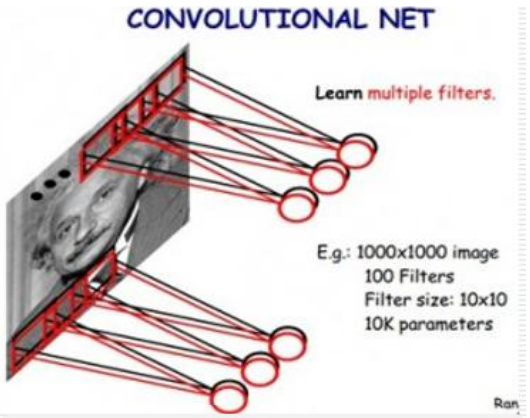
特征提取不充分

权值共享

Image



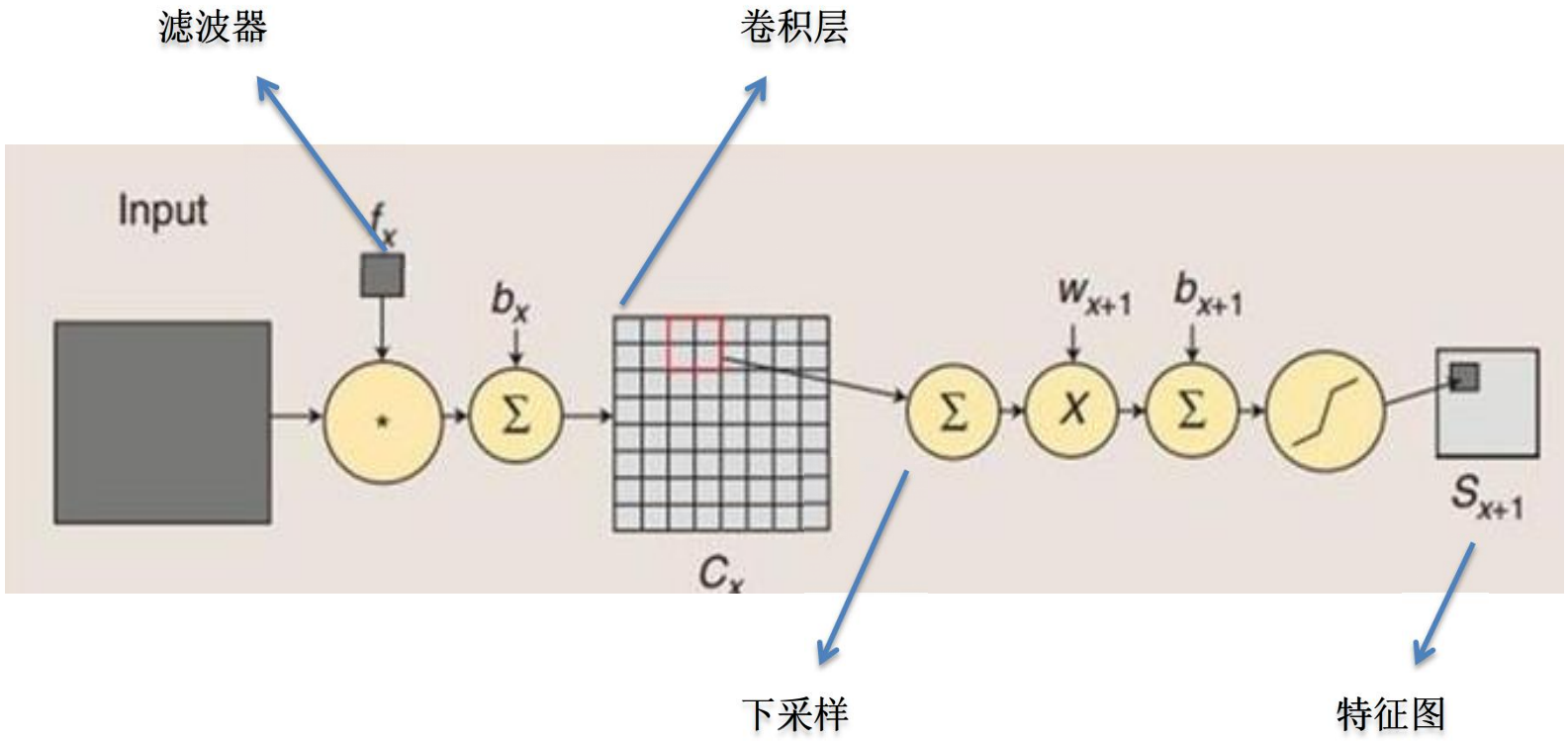
Convolved Feature



多卷积核

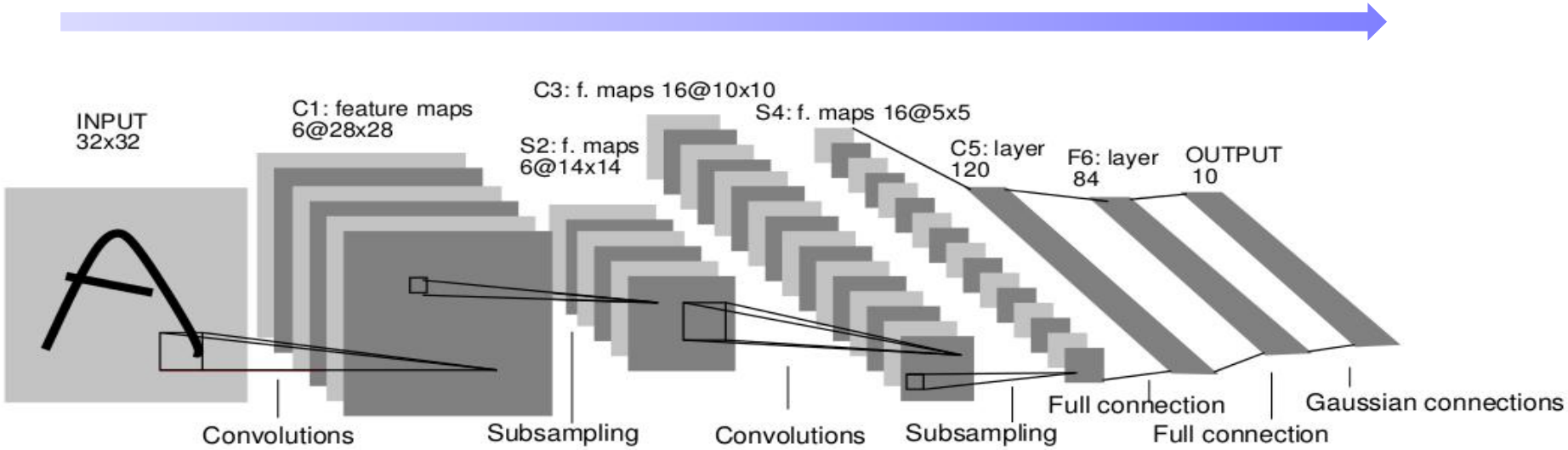
从BP神经网络到卷积神经网络 – 卷积和子采样

- 通常卷积和子采样的过程如图



从BP神经网络到卷积神经网络 - LeNet-5

原始低级特征 抽象高级特征

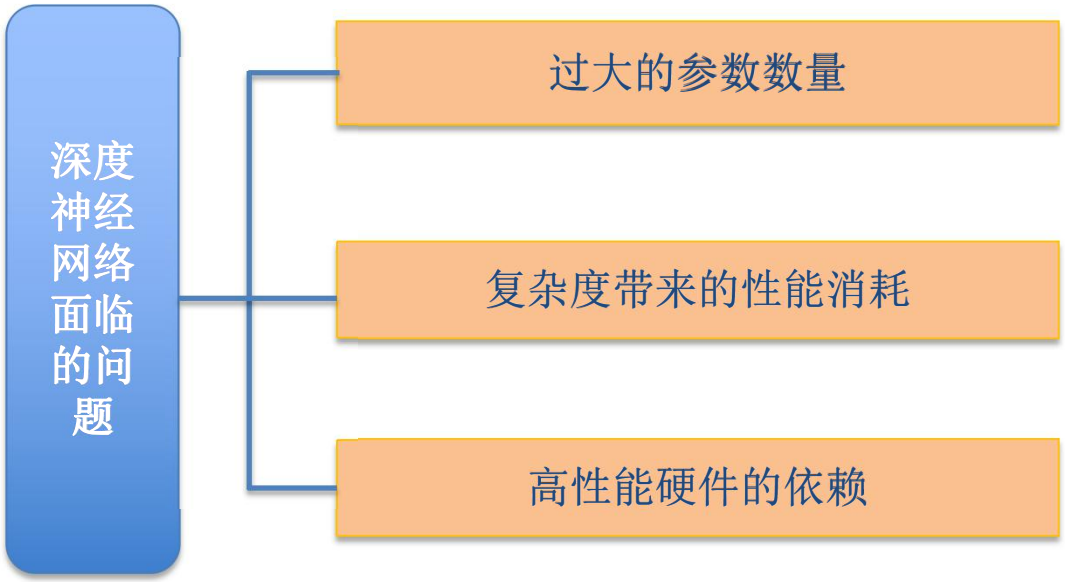


从BP神经网络到卷积神经网络 - LeNet-5训练参数

Layer	Input	Filter size	Feature map size	Number of unit	Training parameters	Remark
C1	32 x 32	5 x 5 x 1 ->6	28 x 28	28 x 28 x 6	156	(5 x 5 +1) x 6
S2	28 x 28	2 x 2(sampling zero)	14 x 14	14 x 14 x 6	12	2 x 6
C3	14 x 14	5 x 5x1->16	10 x 10	10 x 10 x 16	1516	6*（3*5x5+1）+6*（4*5x5+1）+3*（4*5x5+1）+（5x5*6+1）
S4	10 x 10	2 x 2(sampling zero)	5 x 5	5 x 5 x16	32	2*16
C5	5x5	5x5x1->120	1 x 1	1 x 1x120	48120	120*（16*5*5+1）
F6	84 node				10164	7x12*(120+1)

深度神经网络所面临的问题

- 深度神经网络面临的问题



- 深度神经网络面临的这些问题，同时也制约着神经网络在嵌入式系统中的应用。

近几年神经网络的统计数据

- 下表展示了近年来神经网络的统计数据

年份	网络模型	网络层数	参数大小
2012	AlexNet	5+3	60M
2013	Clarifai	5+3	60M
2014	MSRA	5+3	200M
2014	VGG-19	16+3	143M
2014	GoogleNet	22	6.8M
2015	ResNet	152	19.4M

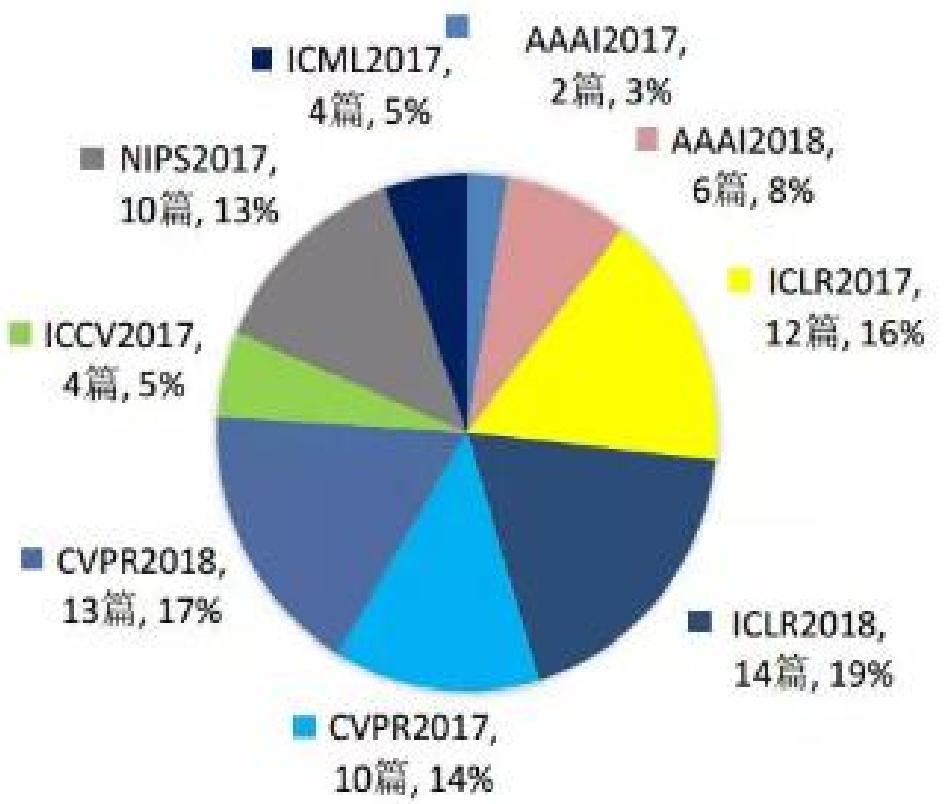
神经网络压缩技术发展状况

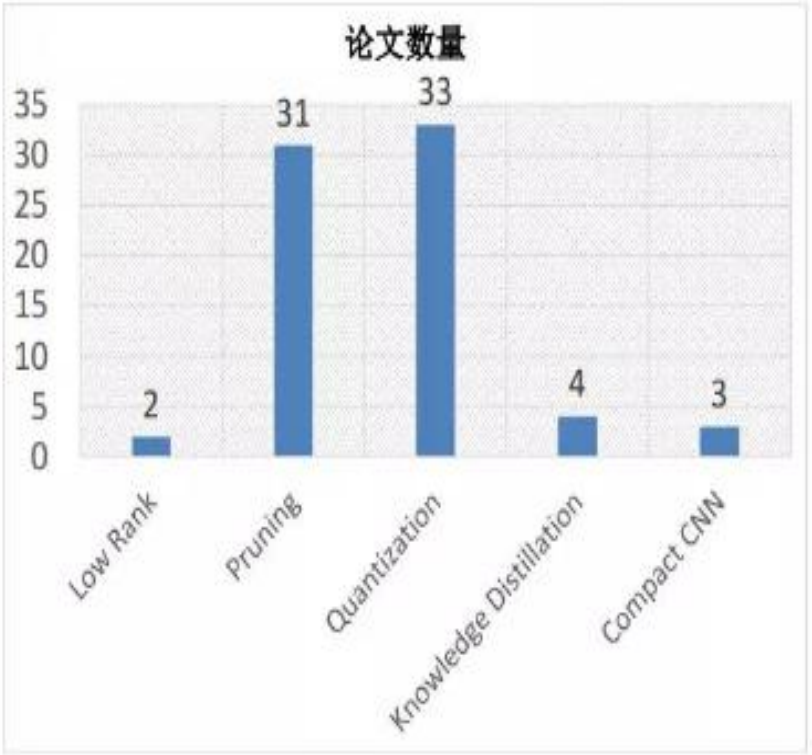
- 模型加速与压缩分类
- 网络加速和压缩的论文统计

神经网络压缩方法分类

- Low-Rank
- Pruning
- Quantization
- Knowledge Distillation
- Compact Network Design

网络加速和压缩的论文统计





- Quantization becomes popular
  - *efficient training using quantization*
  - *low-bit representation*
  - *binary convolutional neural networks*
- Pruning is still a hot topic
  - *small accuracy drop*
  - *efficient structured pruning*
- Few low-rank based method
  - *tensor decomposition is not efficient for current network structure*

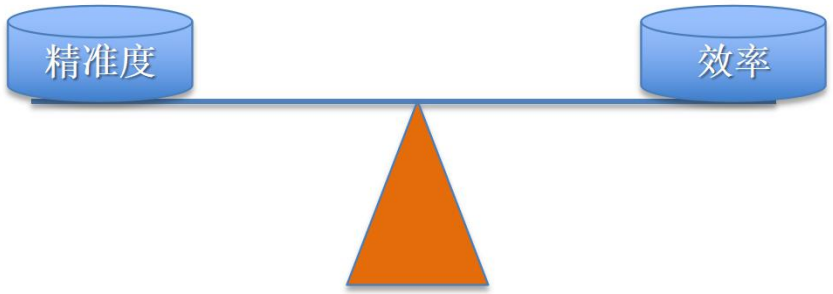
- 引自程健博士《让机器“删繁就简”：深度神经网络加速与压缩》

神经网络压缩技术

- 神经网络压缩遵循的基本原则
- 紧凑网络模型举例(compact net)
- 网络剪枝(pruning)
- 参数量化(quantization)

神经网络压缩遵循的基本原则

- 神经网络压缩技术力求遵循一个基本的原则，即精准度和效率的平衡；效率问题主要是模型的存储问题和模型进行预测的速度问题。



- 可从下面几个角度衡量压缩效率

- 精准度
- 计算资源和存储资源占用
- 模型体量
- 硬件或平台依赖性

几种轻量化网络模型

- 轻量化模型设计主要思想在于设计更高效的网络计算方式（主要针对卷积方式），从而使网络参数减少的同时，不损失网络性能。.
- 仅几年来诞生的轻量化网络模型如下：

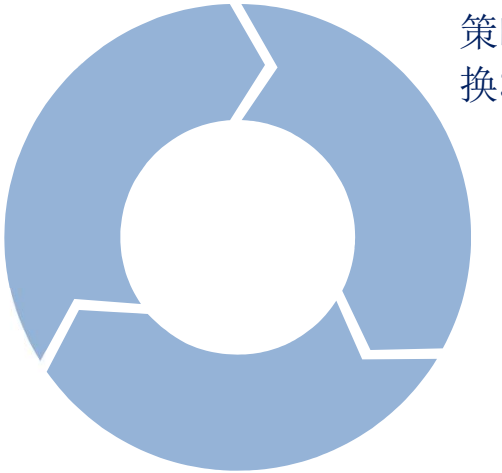
Network model	Publication	Auther	Key technology
SqueezeNet	ICLR-2017	Han.etc.(Berkeley&Stanford)	引入fire module, 用1*1的卷积核压缩特征图(feature maps)数量
MobileNet	CVPR-2017	Google	Depth-wise convolution (depth-wise convolution + pointwise convolution)
ShuffleNet	CVPR-2017	Face++	Depth-wise convolution （Group convolution + channel shuffle）
Xception	N/A	Google	总体基于Depth-wise convolution的思想

轻量化网络模型举例-SqueezeNet（基本策略）

- SqueezeNet(Iandola等人设计)　SqueezeNet设计目标不是为了得到最佳的CNN识别精度，而是希望简化网络复杂度，同时达到public网络的识别精度。所以SqueezeNet主要是为了降低CNN模型参数数量而设计的

- 为达到以上目的而遵循的三个基本策略：

策略三：延后降采样操作以便卷积层有大的激活图（更大的激活图保留了更多的信息，可以提供更高的分类准确率）



策略一：用1 x 1的卷积和替换3x3的卷积核；参数减少为原来的1/9

一个采用3x3卷积核的卷积层，该层所有卷积参数的数量（不考虑偏置）为：P=（输入通道数）*（滤波器数）*3*3

策略二：减少输入通道数量：（这一部分使用squeeze layers来实现）

策略一和策略二主要是在保持精准度不变的情况下减少网络参数；而策略三主要是在参数数量受限的情况下提高准确率。

紧凑网络模型举例-SqueezeNet（Fire module）

- Fire module:将原来简单的一层conv层变成两层：squeeze层+expand层，各自带上Relu激活层。
- 使用Fire module的过程中，令s1x1 < e1x1 + e3x3，这样squeeze layer可以限制输入通道数量（也就是前面提到的策略二）

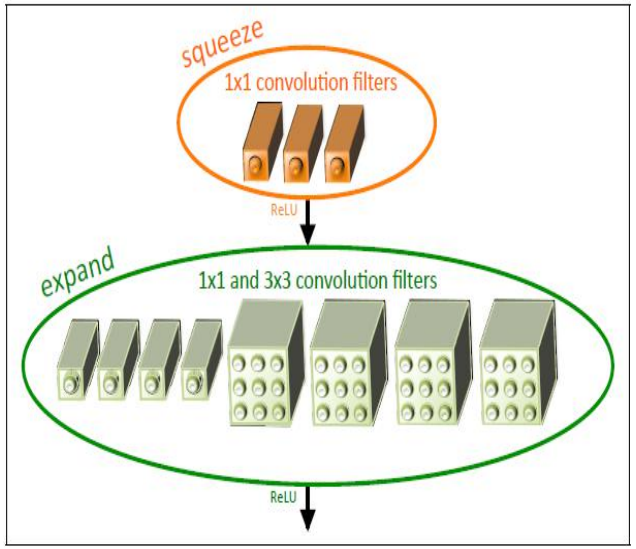


Figure 1: Microarchitectural view: Organization of convolution filters in the **Fire module**. In this example, $s_{1x1} = 3$, $e_{1x1} = 4$, and $e_{3x3} = 4$. We illustrate the convolution filters but not the activations.

紧凑网络模型举例-SqueezeNet（网络结构和规模）

- 网络结构和规模

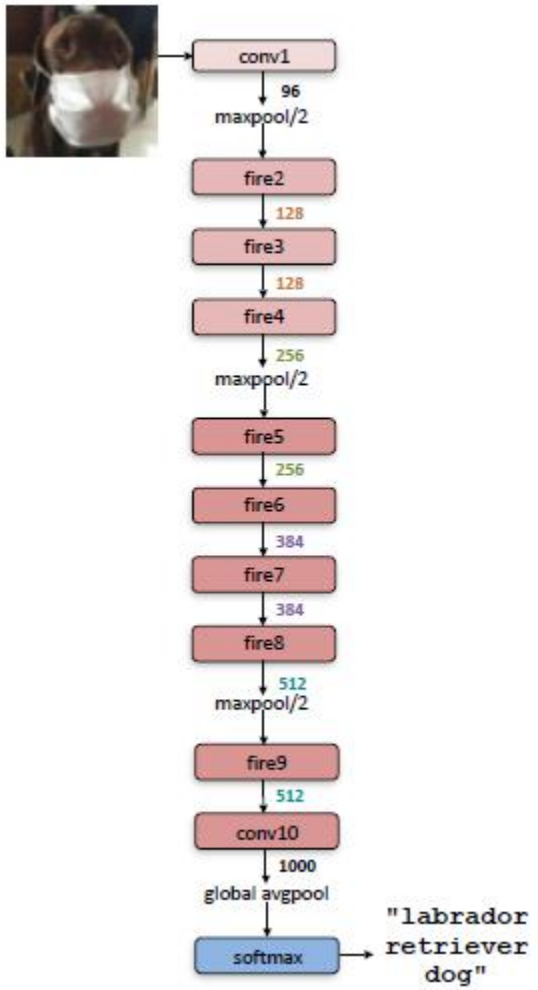


Table 1: SqueezeNet architectural dimensions. (The formatting of this table was inspired by the Inception2 paper (Ioffe & Szegedy, 2015).)

layer name/type	output size	filter size / stride (if not a fire layer)	depth	s_{1x1} (#1x1 squeeze)	e_{1x1} (#1x1 expand)	e_{3x3} (#3x3 expand)	s_{1x1} sparsity	e_{1x1} sparsity	e_{3x3} sparsity	# bits	#parameter before pruning	#parameter after pruning
input image	224x224x3										-	-
conv1	111x111x96	7x7/2 (x96)	1				100% (7x7)			6bit	14,208	14,208
maxpool1	55x55x96	3x3/2	0									
fire2	55x55x128		2	16	64	64	100%	100%	**33%**	6bit	11,920	5,746
fire3	55x55x128		2	16	64	64	100%	100%	**33%**	6bit	12,432	6,258
fire4	55x55x256		2	32	128	128	100%	100%	**33%**	6bit	45,344	20,646
maxpool4	27x27x256	3x3/2	0									
fire5	27x27x256		2	32	128	128	100%	100%	**33%**	6bit	49,440	24,742
fire6	27x27x384		2	48	192	192	100%	**50%**	**33%**	6bit	104,880	44,700
fire7	27x27x384		2	48	192	192	**50%**	100%	**33%**	6bit	111,024	46,236
fire8	27x27x512		2	64	256	256	100%	**50%**	**33%**	6bit	188,992	77,581
maxpool8	13x12x512	3x3/2	0									
fire9	13x13x512		2	64	256	256	**50%**	100%	**30%**	6bit	197,184	77,581
conv10	13x13x1000	1x1/1 (x1000)	1					**20%** (3x3)		6bit	513,000	103,400
avgpool10	1x1x1000	13x13/1	0								1,248,424 (total)	**421,098** (total)

activations　parameters　compression info

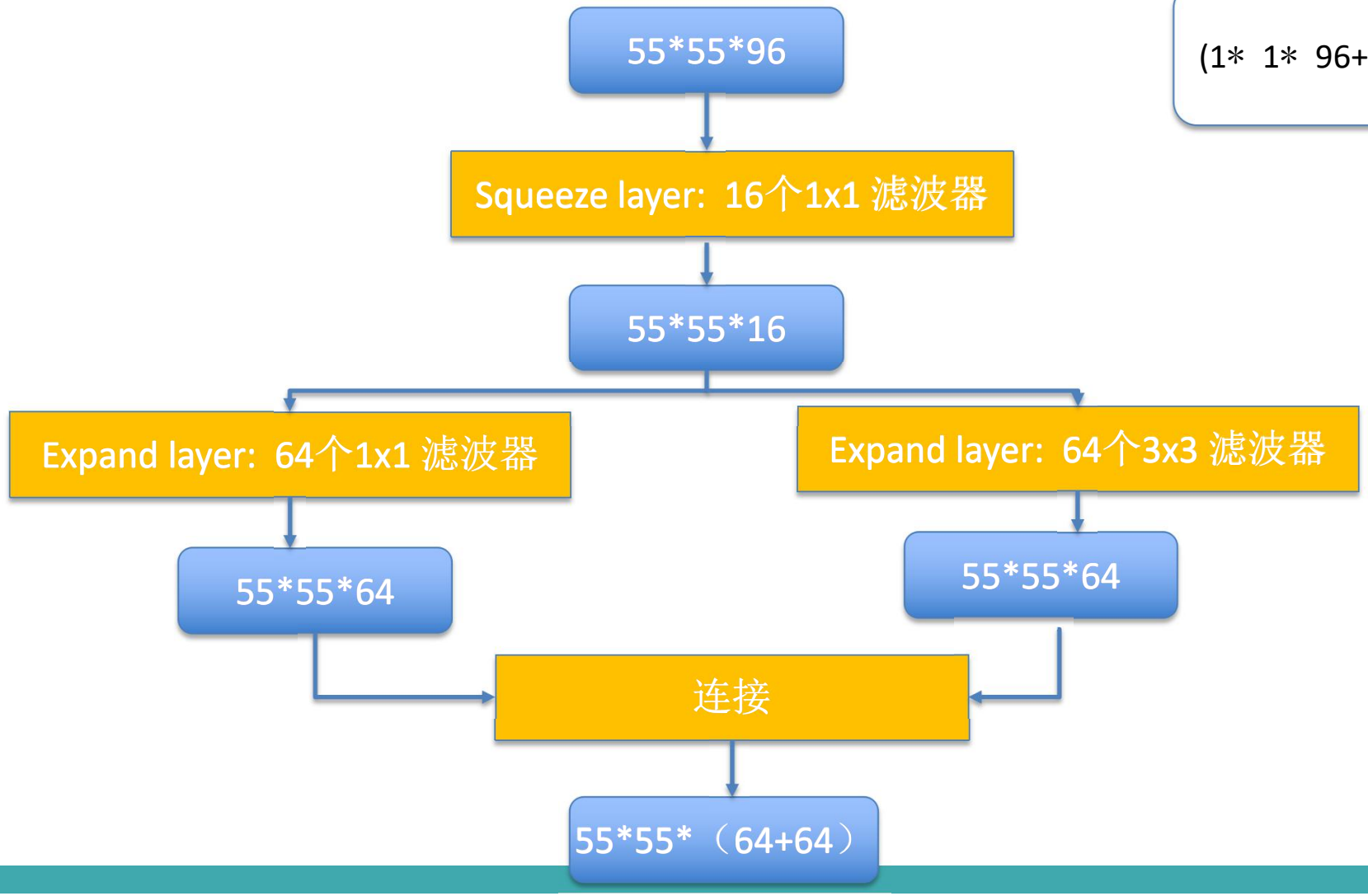
紧凑网络模型举例-SqueezeNet（参数计算举例）

- 以前一节表中的fire2为例，计算这 一层的参数规模。

参数总数

(1* 1* 96+1)* 16+(1* 1* 16+1)* 64+(3* 3* 16+1)* 64

＝(1552+1088+9280)=11920



紧凑网络模型举例-SqueezeNet（验证对比）

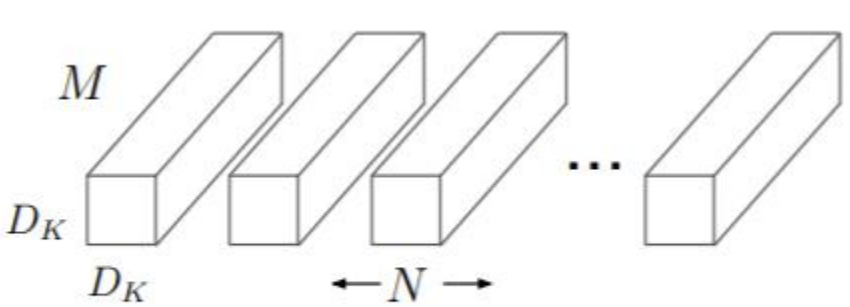
- SqueezeNet与AlexNet在精度和模型大小方面做了对比，如下：

Table 2: Comparing SqueezeNet to model compression approaches. By *model size*, we mean the number of bytes required to store all of the parameters in the trained model.

CNN architecture	Compression Approach	Data Type	Original → Compressed Model Size	Reduction in Model Size vs. AlexNet	Top-1 ImageNet Accuracy	Top-5 ImageNet Accuracy
AlexNet	None (baseline)	32 bit	240MB	1x	57.2%	80.3%
AlexNet	SVD (Denton et al., 2014)	32 bit	240MB → 48MB	5x	56.0%	79.4%
AlexNet	Network Pruning (Han et al., 2015b)	32 bit	240MB → 27MB	9x	57.2%	80.3%
AlexNet	Deep Compression (Han et al., 2015a)	5-8 bit	240MB → 6.9MB	35x	57.2%	80.3%
SqueezeNet (ours)	None	32 bit	4.8MB	**50x**	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	8 bit	4.8MB → 0.66MB	**363x**	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	6 bit	4.8MB → 0.47MB	**510x**	57.5%	80.3%

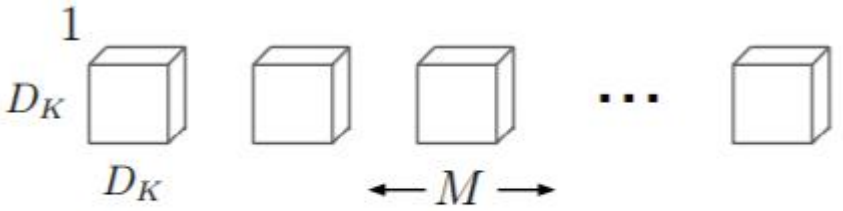
几种轻量化网络模型-深度可分离卷积（Deepwise Separable Convolution）

- 下图显示了传统的卷积滤波器能够被深度可分离卷积所替换：



(a) Standard Convolution Filters

计算代价： $D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F$



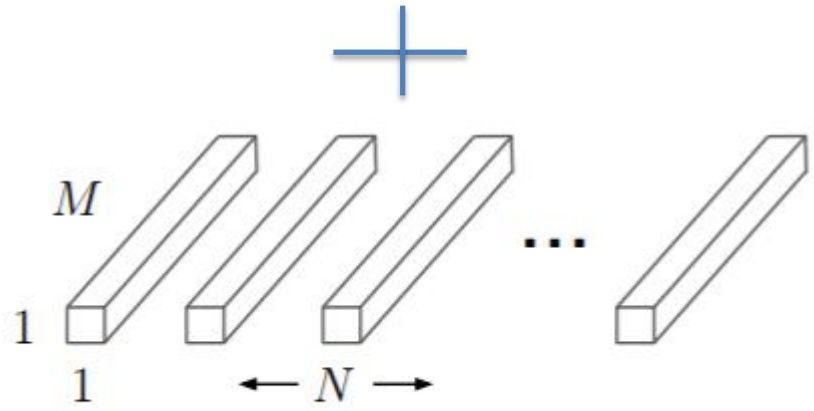
(b) Depthwise Convolutional Filters

计算代价：

$D_K \cdot D_K \cdot M \cdot D_F \cdot D_F$

深度可分离卷积相对于传统的卷积操作的计算代价：

$$\frac{D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F}{D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F} = \frac{1}{N} + \frac{1}{D_K^2}$$



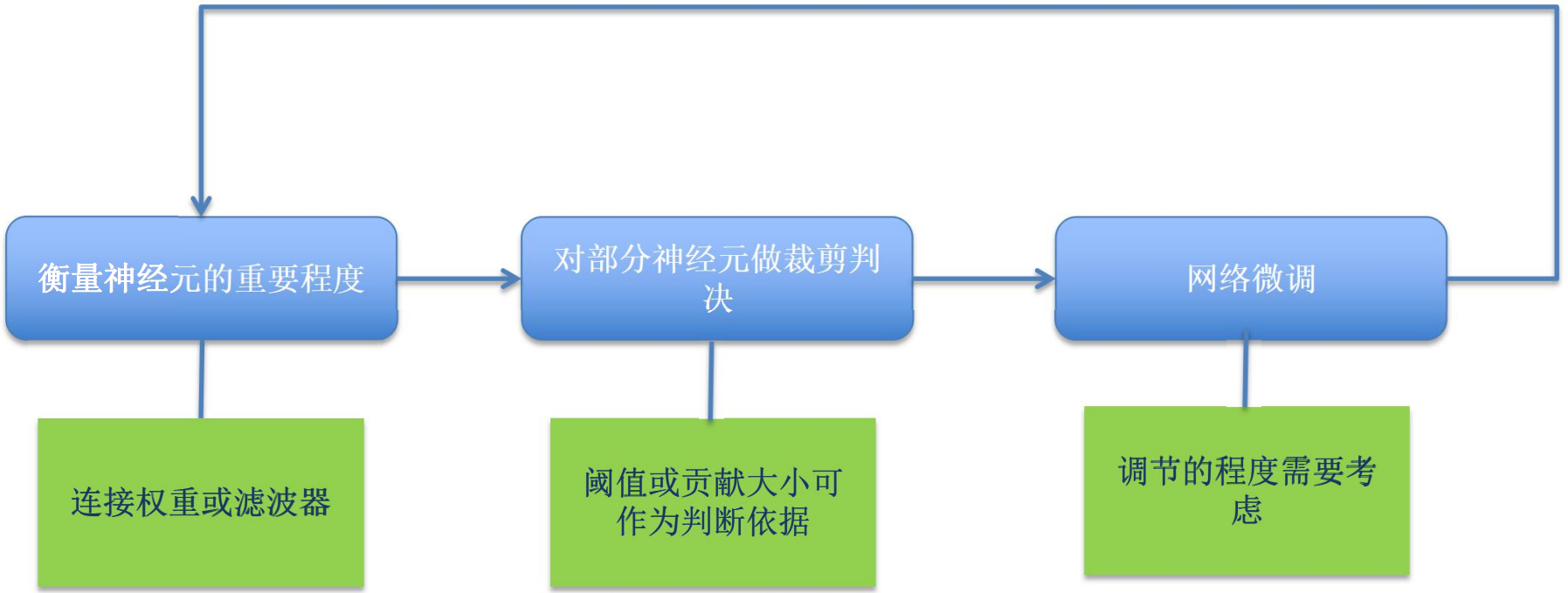
(c) 1×1 Convolutional Filters called Pointwise Convolution in the context of Depthwise Separable Convolution

计算代价：

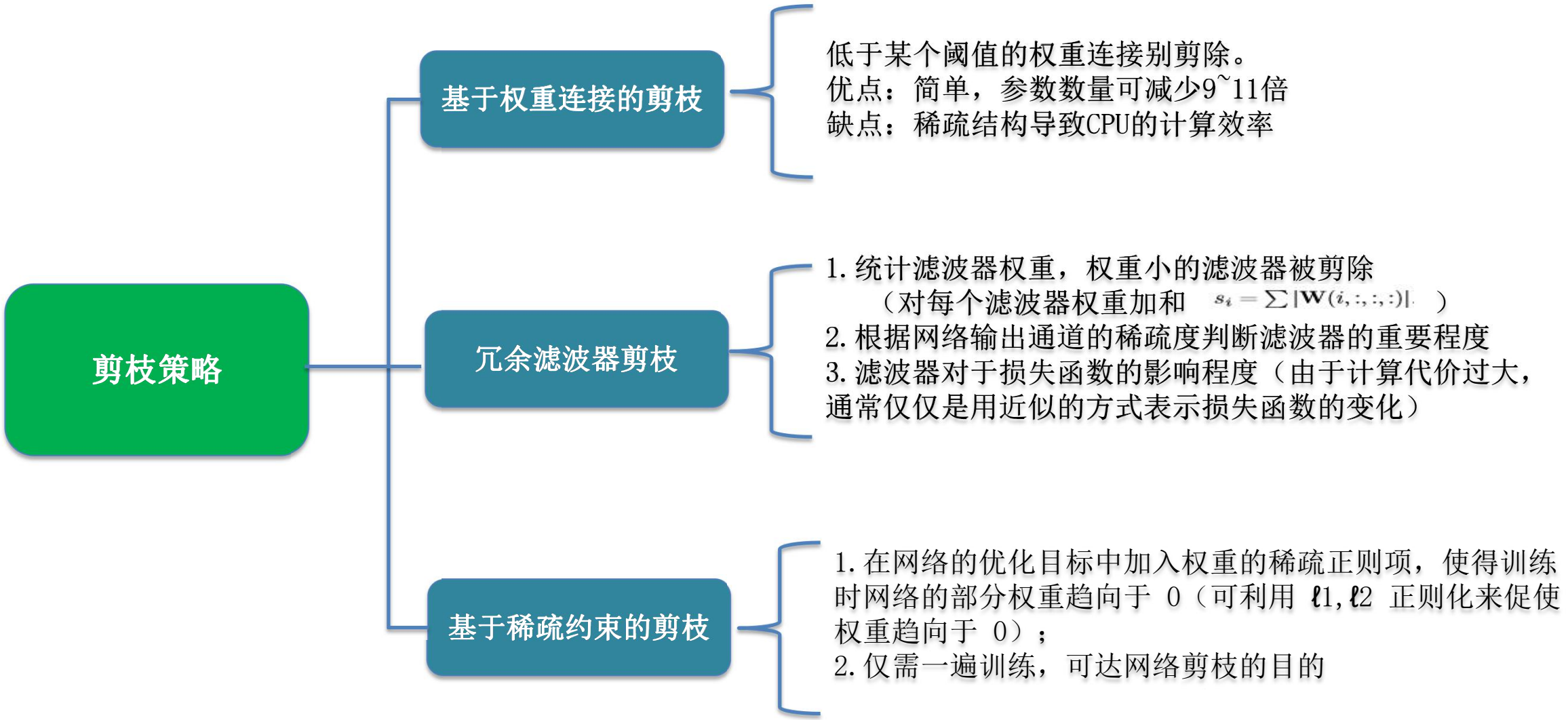
$M \cdot N \cdot D_F \cdot D_F$

网络剪枝（pruning）

- 通过剪枝处理，在减小模型复杂度，还能有效防止过拟合，提升模型泛化性。
- 进入到深度学习时代后，如何对大型深度神经网络进行高效的剪枝，成为了 一个重要的研究课题。尽管各种剪纸算法的具 体细节不尽相同，但所采用的基本框架却是相似的。给定一个预训练好的网络 模型，常用的剪枝算法一般都遵从如下的操作流程：



网络剪枝(pruning)-剪枝策略



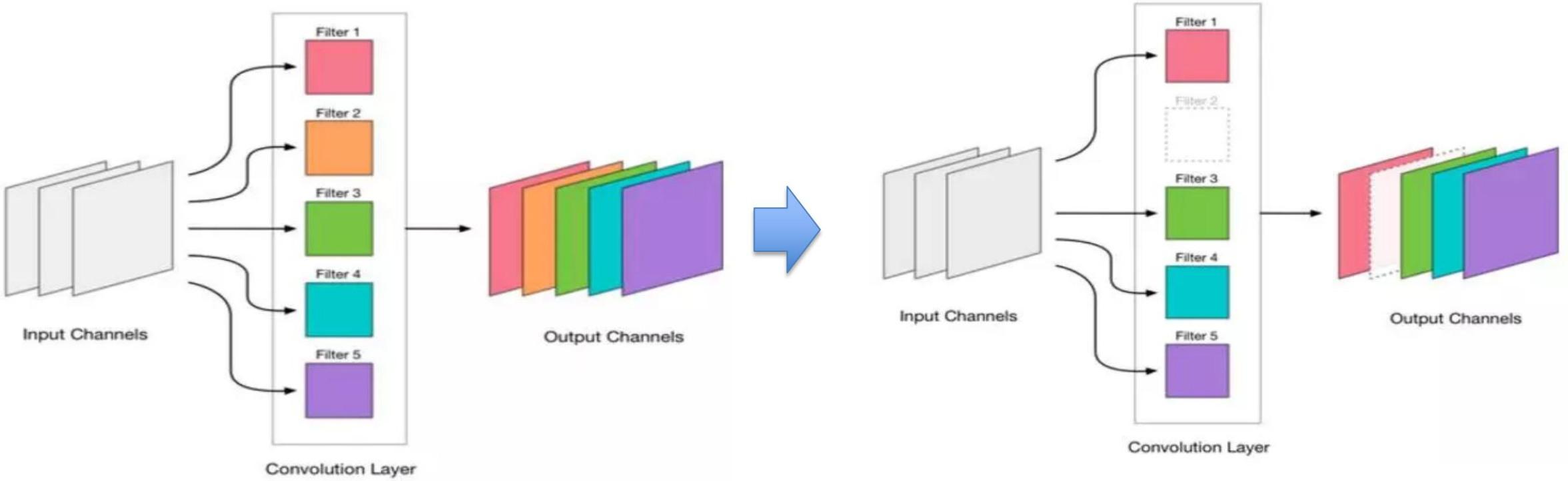
网络剪枝(pruning)-验证对比

- 下表显示了剪枝前后的压缩统计数据（来自Han等人的deep compression)

The compression pipeline can save 35× to 49× parameter storage with no loss of accuracy.

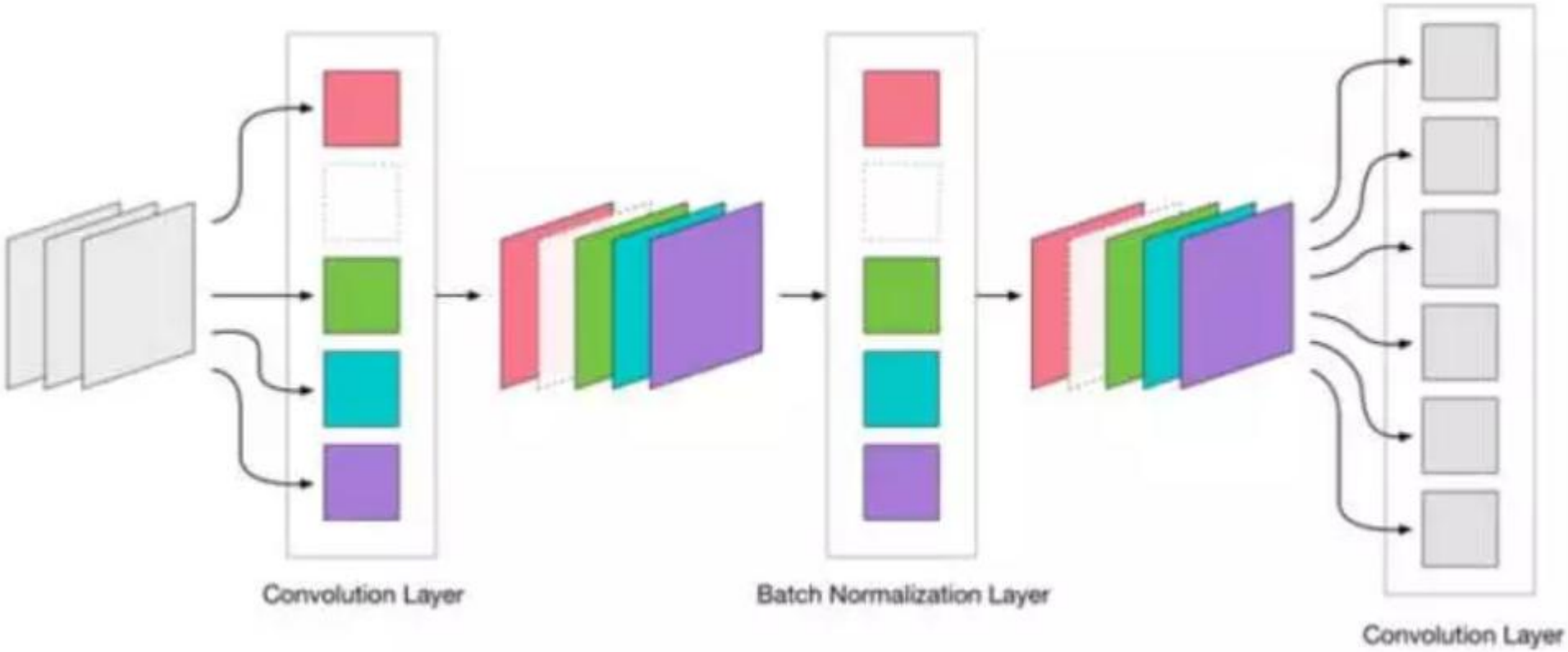
Network	Top-1 Error	Top-5 Error	Parameters	Compress Rate
LeNet-300-100 Ref	1.64%	-	1070 KB	
LeNet-300-100 Compressed	1.58%	-	**27 KB**	**40×**
LeNet-5 Ref	0.80%	-	1720 KB	
LeNet-5 Compressed	0.74%	-	**44 KB**	**39×**
AlexNet Ref	42.78%	19.73%	240 MB	
AlexNet Compressed	42.78%	19.70%	**6.9 MB**	**35×**
VGG-16 Ref	31.50%	11.32%	552 MB	
VGG-16 Compressed	31.17%	10.91%	**11.3 MB**	**49×**

网络剪枝(pruning)-以滤波器级别的剪指为例（一）



可使用不同方式来估计滤波器的相关性，例如可通过计算权重的方式计算滤波器权重的 L1 范数，即所有滤波器权重的绝对值之和。

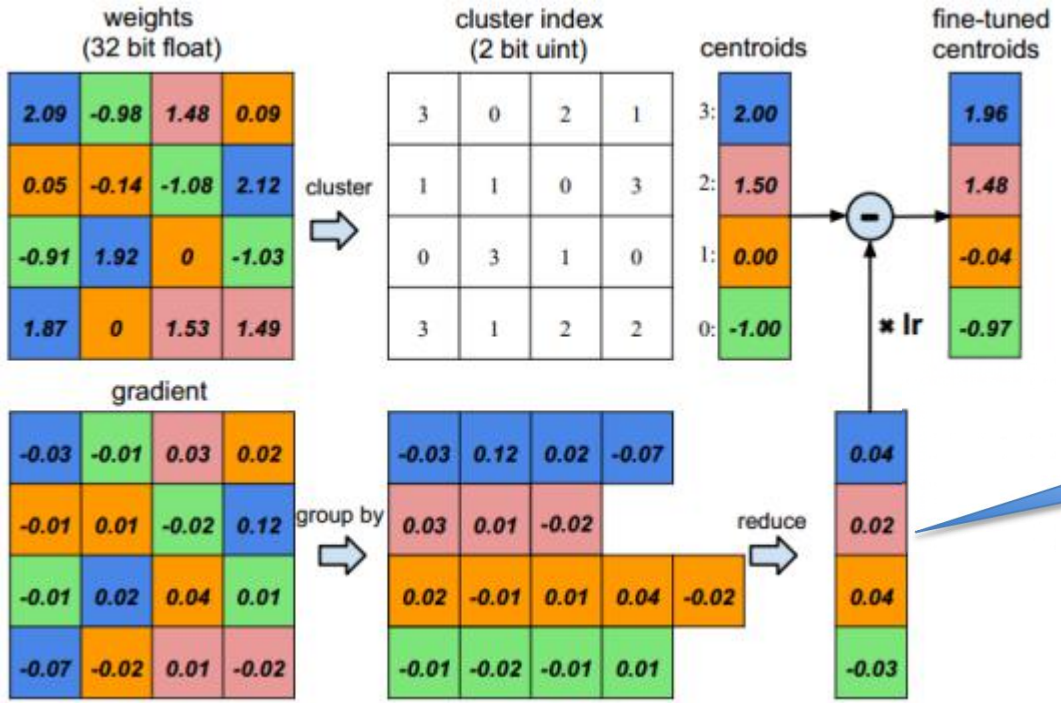
网络剪枝(pruning)-以滤波器级别的剪指为例（二）



当卷积之后是批量归一化（BN）时，必须从批量归一化参数中去除这些通道

参数量化(quantization)

- 权重量化与共享　使用 k-means聚类来标识训练网络中的共享权值。
- 将连续分布的权值离散化，从而减小需要存储的权值数量。
- 让许多连接共享同一权重，使原始存储整个网络权重变为只需要存储码本(有效的权重)和索引；

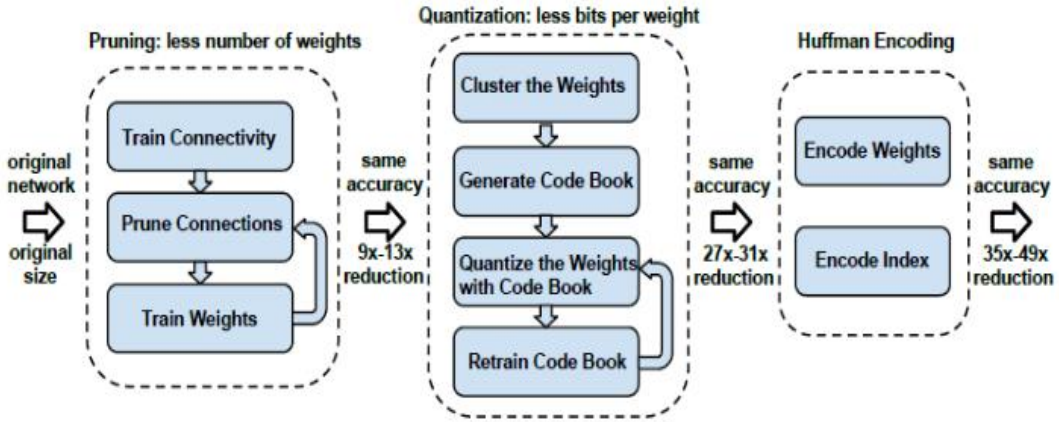


K-means聚类：

$$\arg\min_{C} \sum_{i=1}^{k} \sum_{w \in c_i} |w - c_i|^2$$

剪枝、量化和哈夫曼编码

- 通过将剪枝，量化和哈夫曼编码运用在不同阶段以达到有效的压缩率。
（来自Han等人的deep compression）



Compression statistics for AlexNet. P: pruning, Q: quantization, H:Huffman coding.

Layer	#Weights	Weights% (P)	Weight bits (P+Q)	Weight bits (P+Q+H)	Index bits (P+Q)	Index bits (P+Q+H)	Compress rate (P+Q)	Compress rate (P+Q+H)
conv1	35K	84%	8	6.3	4	1.2	32.6%	20.53%
conv2	307K	38%	8	5.5	4	2.3	14.5%	9.43%
conv3	885K	35%	8	5.1	4	2.6	13.1%	8.44%
conv4	663K	37%	8	5.2	4	2.5	14.1%	9.11%
conv5	442K	37%	8	5.6	4	2.5	14.0%	9.43%
fc6	38M	9%	5	3.9	4	3.2	3.0%	2.39%
fc7	17M	9%	5	3.6	4	3.7	3.0%	2.46%
fc8	4M	25%	5	4	4	3.2	7.3%	5.85%
Total	61M	11%(9×)	5.4	4	4	3.2	3.7% (**27**×)	2.88% (**35**×)

- 基于ImageNet ILSVRC-2012数据集的统计结果。

Thanks!

Q & A