Why？

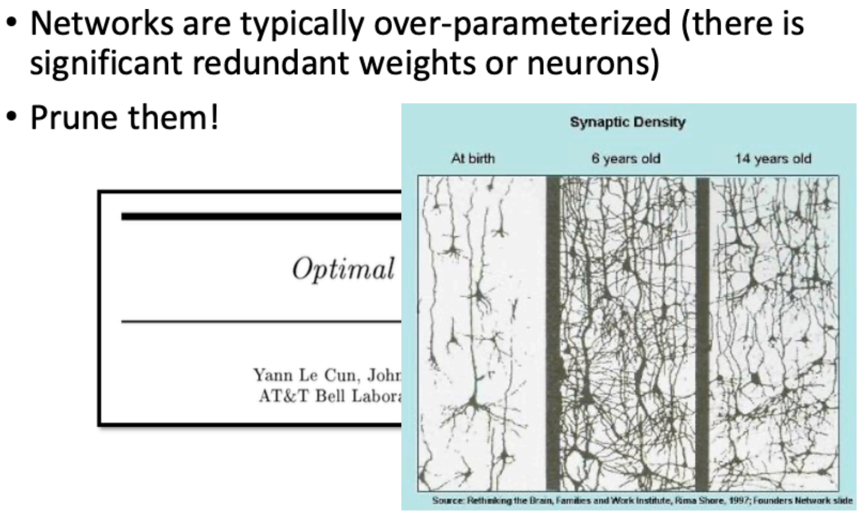
在未来我们可能需要model放到model device上面，但这些device上面的资源是有限的，包括存储控价有限和computing power有限

Outline

- Network Pruning
- Knowledge Distillation
- Parameter Quantization
- Architecture Design
- Dynamic Computation

Network Pruning

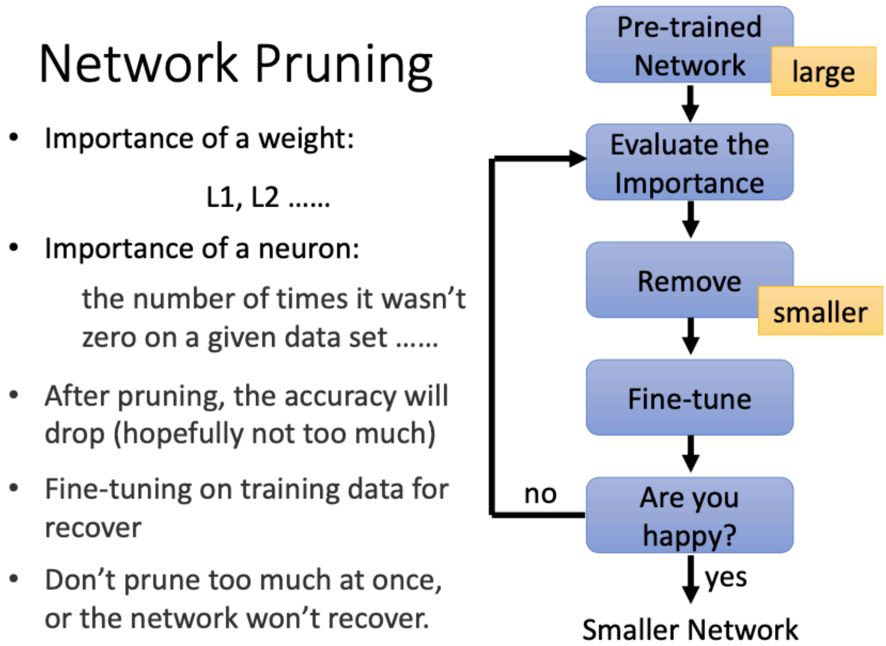
Network can be pruned



Network Pruning

对于训练好的network，我们要判断其weight和neural的重要性：

- 如果某个weight接近于0，那么我们可以认为这个neural是不那么重要的，是可以pruning的；如果是某个很正或很负的值，该weight就被认为对该network很重要；
- 如果某个neural在给定的dataset下的输出都是0，那么我们就可以认为该neural是不那么重要的



在评估出weight和neural的重要性后，再进行排序，来移除一些不那么重要的weight和neural，这样network就会变得smaller，但network的精确度也会随之降低，因此还需要进行fine-tuning

最好是每次都进行小部分的remove，再进行fine-tuing，如果一次性remove很多，network的精确度也不会再恢复

Why Pruning?

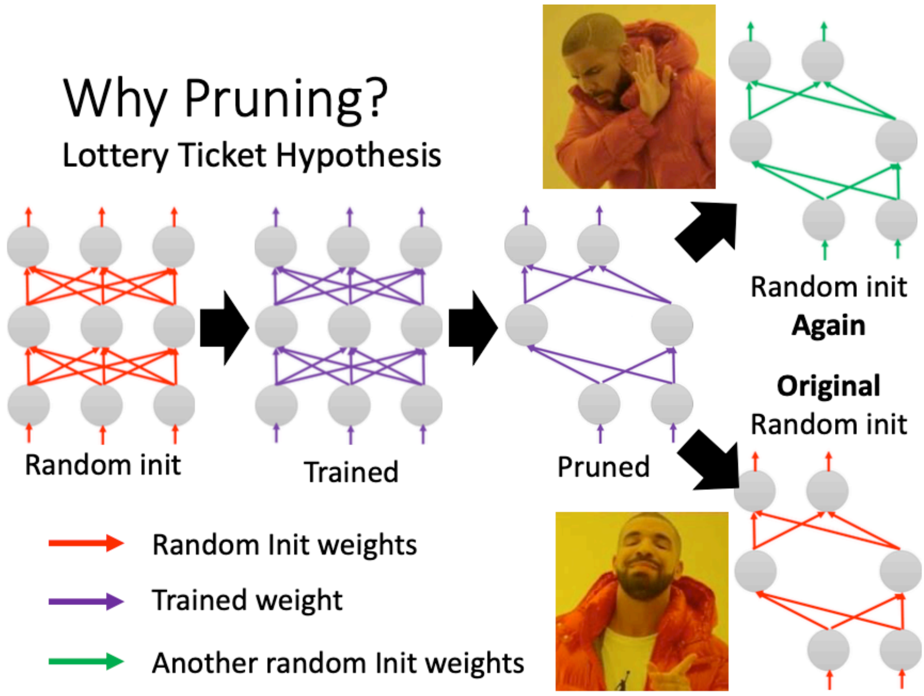
Q：为什么不直接train一个小的network呢?

A：小的network比较难train，[大的network更容易optimize]

Lottery Ticket Hypothesis

我们先对一个network进行初始化（红色的weight），再得到训练好的network（紫色的weight），再进行pruned，得到一个pruned network

- 如果我们使用pruned network的结构，再进行random init（绿色的weight），会发现这个network不能train下去
- 如果我们使用pruned network的结构，再使用original random init（红色的weight），会发现network可以得到很好的结果



作者就说train这个network就像买大乐透一样，有的random可以tranin起来，有的不可以

Rethinking the Value of Network Pruning

Scratch-E/B表示使用real random initialization，并不是使用original random initialization，也可以得到比fine-tuing之后更好的结果

- Rethinking the Value of Network Pruning
  - https://arxiv.org/abs/1810.05270

Dataset	Model	Unpruned	Pruned Model	Fine-tuned	Scratch-E	Scratch-B
CIFAR-10	VGG-16	93.63 (±0.16)	VGG-16-A	93.41 (±0.12)	93.62 (±0.11)	**93.78** (±0.15)
	ResNet-56	93.14 (±0.12)	ResNet-56-A	92.97 (±0.17)	92.96 (±0.26)	**93.09** (±0.14)
			ResNet-56-B	92.67 (±0.14)	92.54 (±0.19)	**93.05** (±0.18)
	ResNet-110	93.14 (±0.24)	ResNet-110-A	93.14 (±0.16)	**93.25** (±0.29)	93.22 (±0.22)
			ResNet-110-B	92.69 (±0.09)	92.89 (±0.43)	**93.60** (±0.25)
ImageNet	ResNet-34	73.31	ResNet-34-A	72.56	72.77	**73.03**
			ResNet-34-B	72.29	72.55	**72.91**

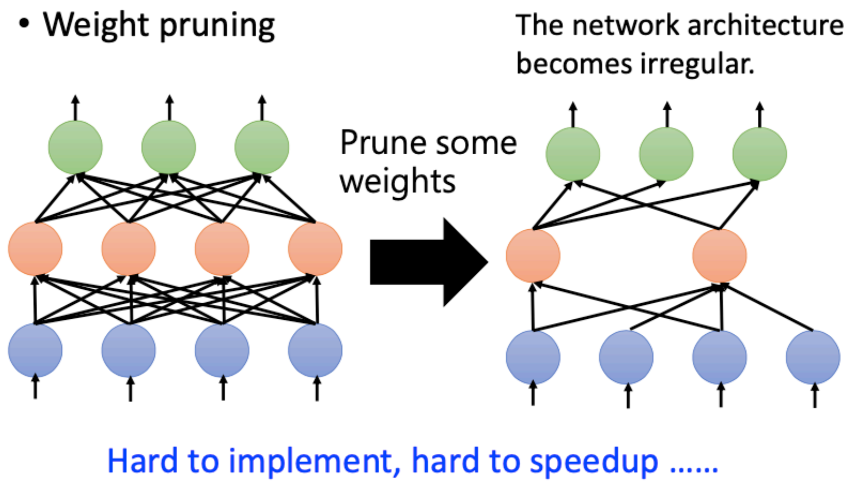
- Real random initialization, not original random initialization in “Lottery Ticket Hypothesis”
- Pruning algorithms could be seen as performing network architecture search

Pratical Issue

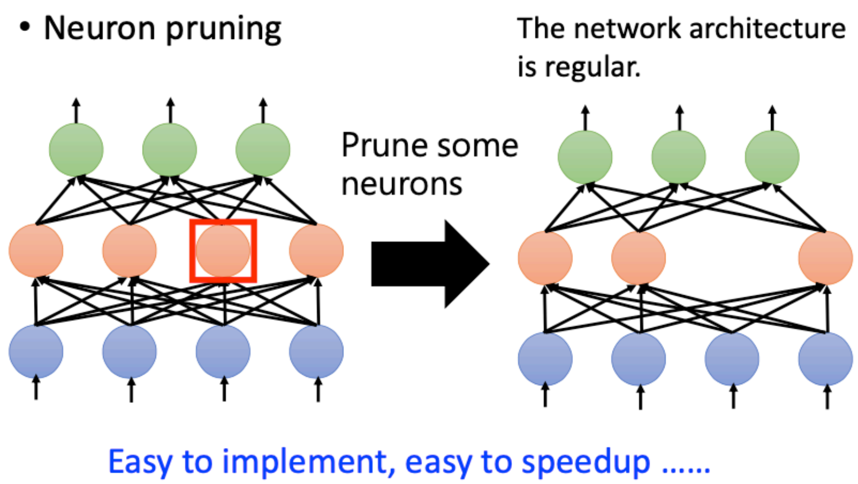
如果我们现在进行weight pruning，进行weight pruning之后的network会变得不规则，有些neural有2个weight，有些neural有4个weight，这样的network是不好implement出来的；

GPU对矩阵运算进行加速，但现在我们的weight是不规则的，并不能使用GPU加速；

实做的方法是将pruning的weight写成0，仍然在做矩阵运算，仍然可以使用GPU进行加速；但这样也会带来一个新的问题，我们并没有将这些weight给pruning掉，只是将它写成0了而已



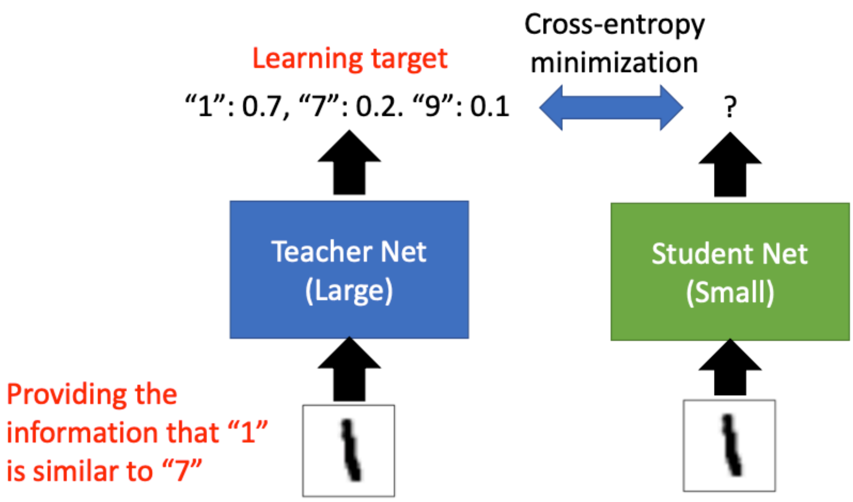
实际上做weight pruning是很麻烦的，通常我们都进行neuron pruning，可以更好地进行implement，也很容易进行speedup



Knowledge Distillation

Student and Teacher

我们可以使用一个small network（student）来学习teacher net的输出分布（1:0.7...），并计算两者之间的cross-entropy，使其最小化，从而可以使两者的输出分布相近

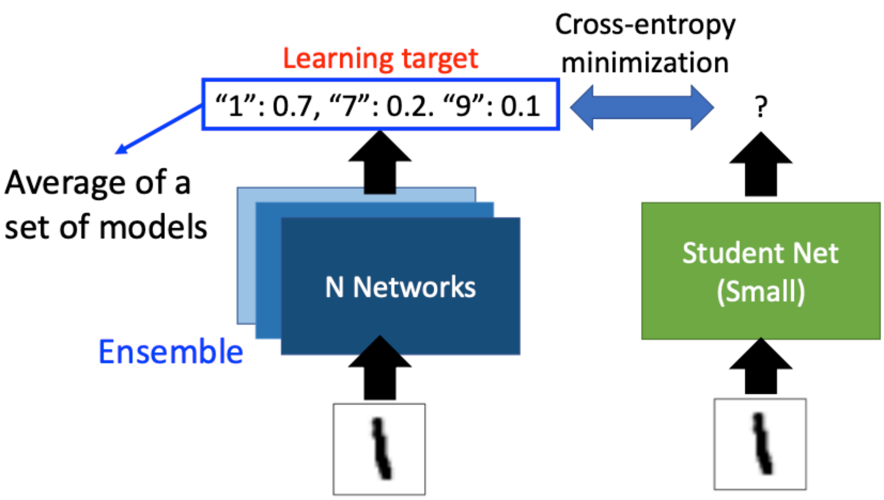


Q：那么我们为什么要让student跟着teacher去学习呢?

A：teacher提供了比label data更丰富的资料，比如teacher net不仅给出了输入图片和1很像的结果，还说明了1和7长得很像，1和9长得很像；因此，student跟着teacher net学习，是可以得到更多的information的

Ensemble

在kaggle上打比赛，很多人的做法是将多个model进行ensemble，通常可以得到更好的精度。但在实际生活中，设备往往放不下这么多的model，这时我们就可以使用Knowledge Distillation的思想，使用student net来对teacher进行学习，在实际的应用中，我们只需要student net的model就好



Temperature

$$y_i = \frac{exp(x_i)}{\sum_j exp(x_j)} \xrightarrow[T=100]{} y_i = \frac{exp(x_i/T)}{\sum_j exp(x_j/T)}$$

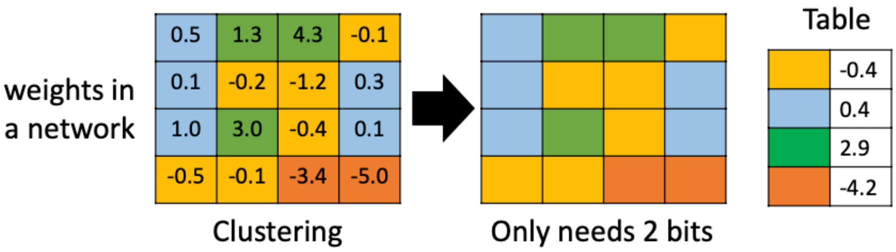
$$x_1 = 100 \quad y_1 = 1 \qquad x_1/T = 1 \quad y_1 = 0.56$$

$$x_2 = 10 \quad y_2 \approx 0 \qquad x_2/T = 0.1 \quad y_2 = 0.23$$

$$x_3 = 1 \quad y_3 \approx 0 \qquad x_3/T = 0.01 \quad y_3 = 0.21$$

Parameter Quantization

- 1. Using less bits to represent a value
- 2. Weight clustering

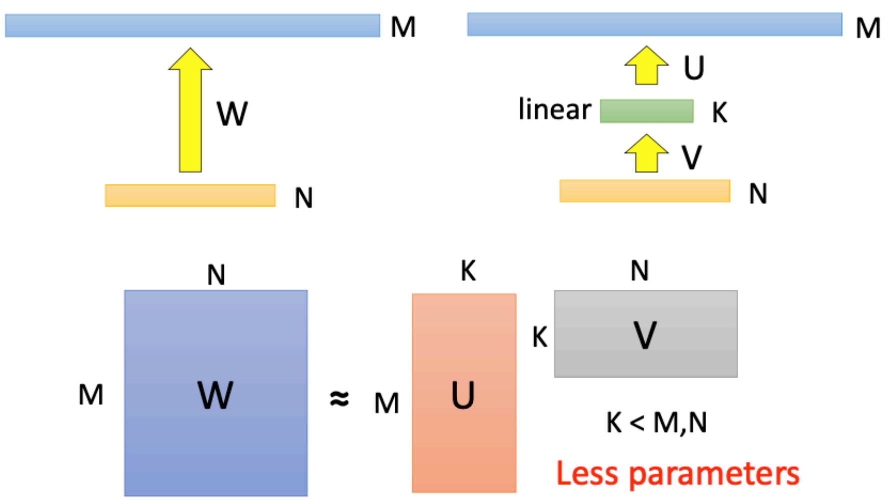


- 3. Represent frequent clusters by less bits, represent rare clusters by more bits
  - e.g. Huffman encoding

Architecture Design（most）

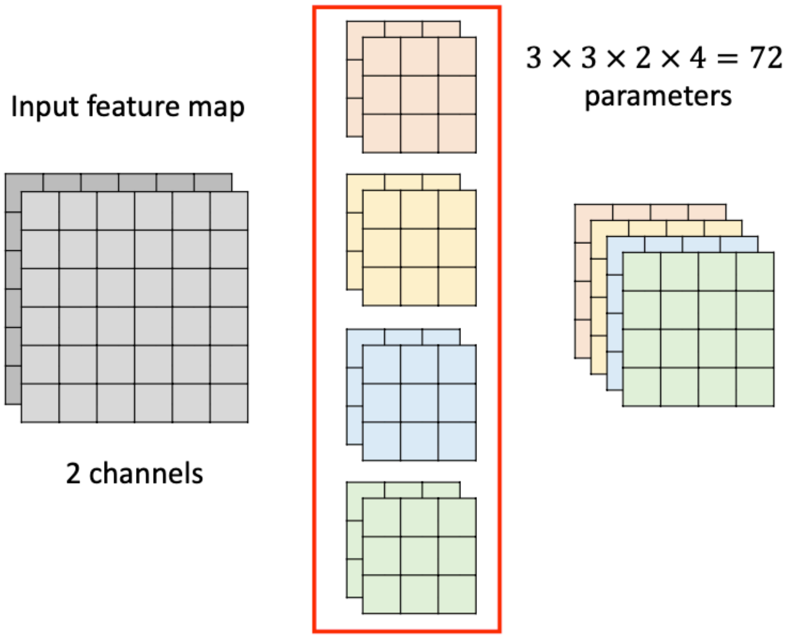
Low rank approximation

中间插入一个linear层，大小为K，那么也可以减少需要训练的参数



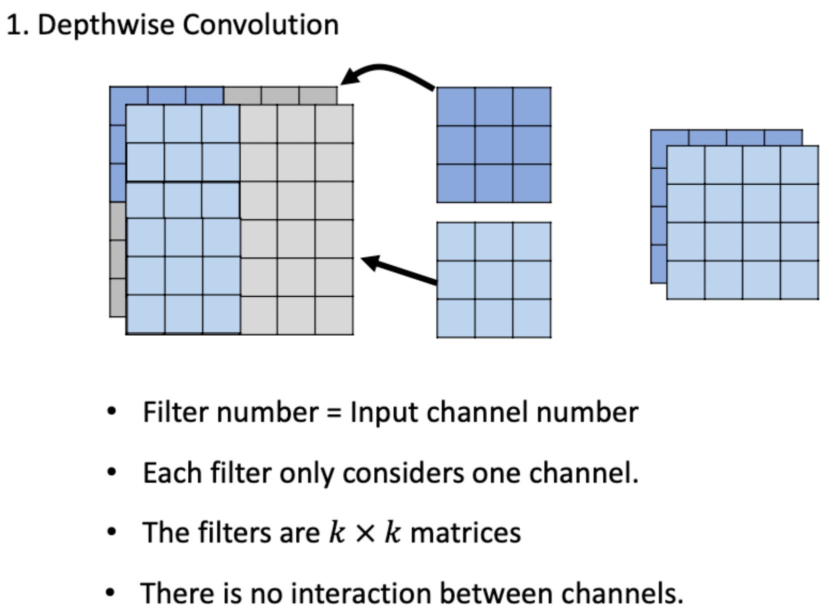
Review: Standard CNN

每个filter要处理所有的channel

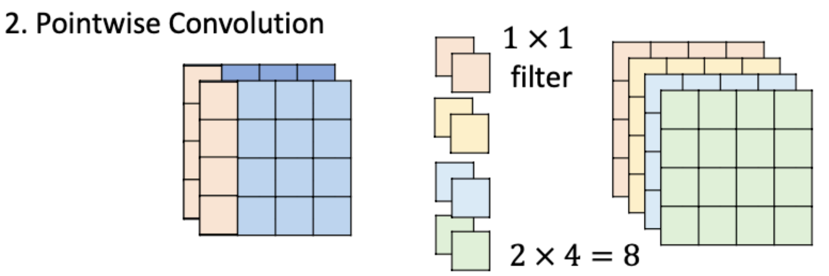


Depthwise Separable Convolution

每个filter只处理一个channel，不同channel之间不会相互影响

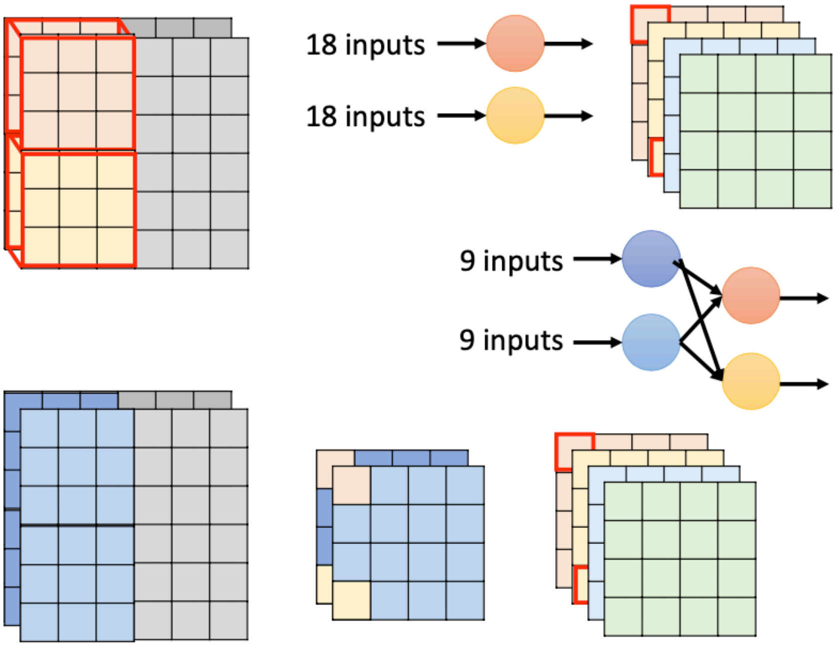


和一般的convolution是一样的，有4个filter，就有4个不同的matrix

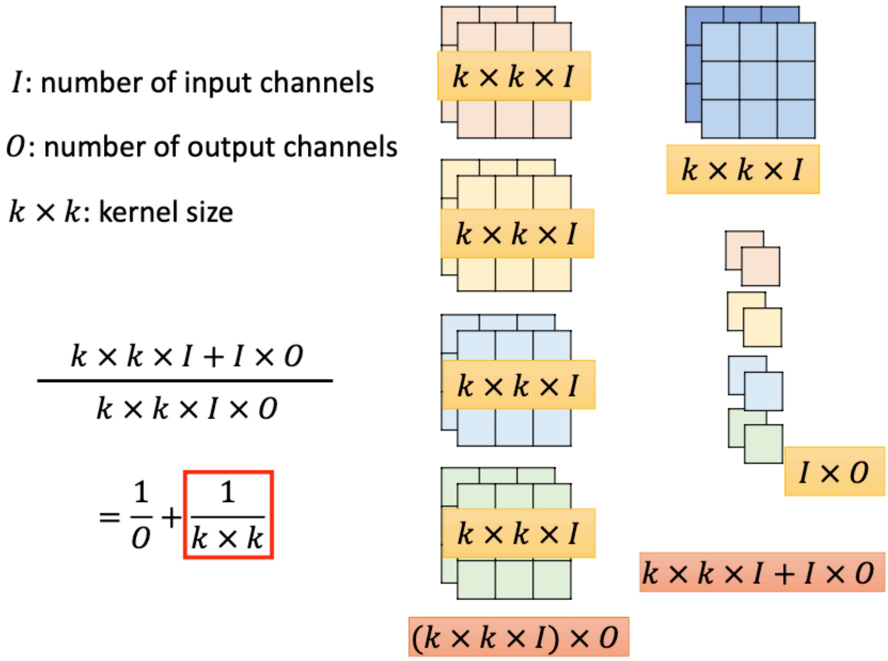


第一步用到的参数量为$3 \times 3 \times 2 = 18$，第二步用到的参数量为$2 \times 4 = 8$，一共有26个参数

Standard CNN vs Depthwise Separable Convolution

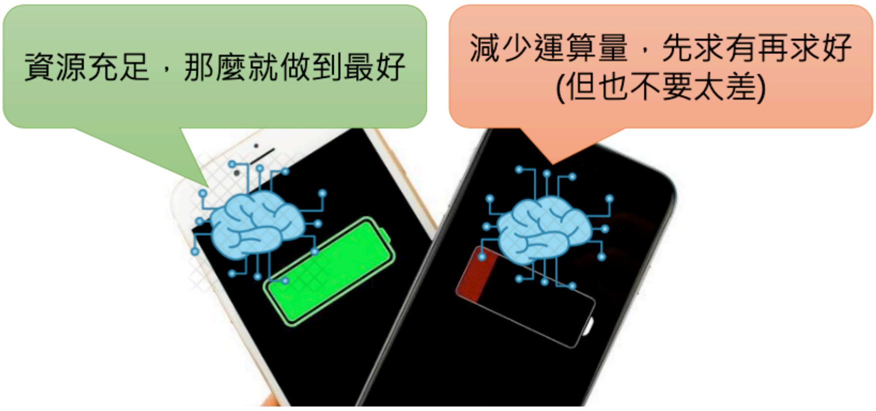


对于普通的卷积，需要的参数量为$(k \times k \times I) \times O$；对于Depthwise Separable Convolution，需要的参数量为$k \times k \times I + I \times O$

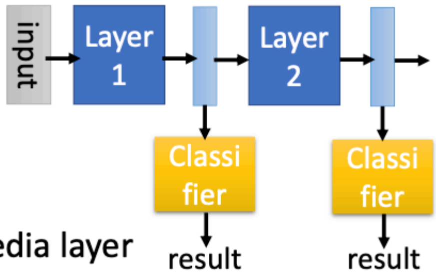


Dynamic Computation

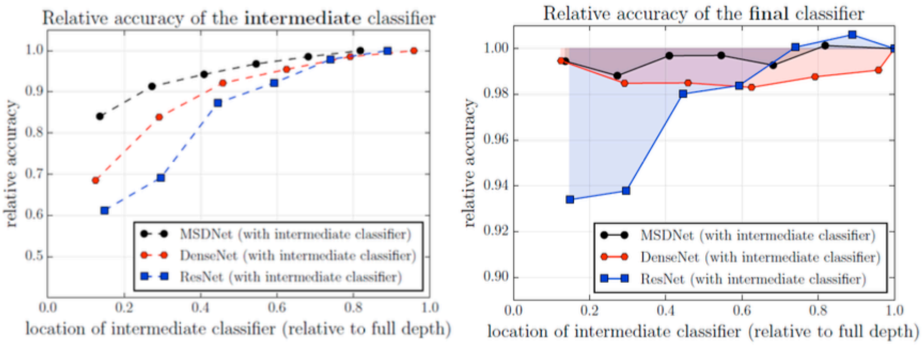
- Can network adjust the computation power it need?



Possible Solutions



- 1. Train multiple classifiers
- 2. Classifiers at the intermedia layer



https://arxiv.org/abs/1703.09844