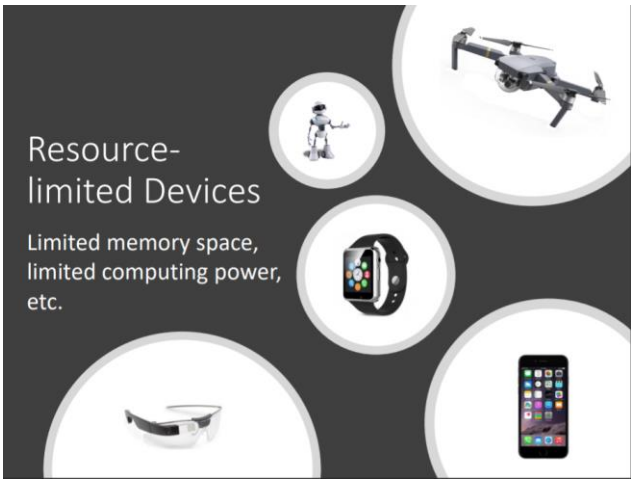
Network Compression

Why?

在未来我们可能需要 model 放到 mobile 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



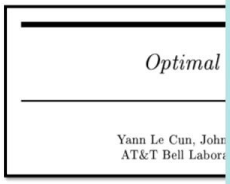
Outline

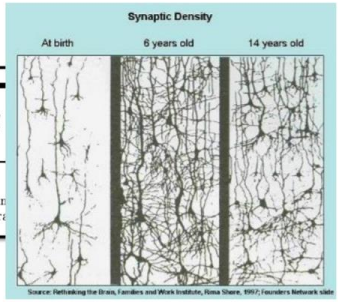
- Network Pruning
- Knowledge Distillation
- Parameter Quantization
- Architecture Design
- Dynamic Computation

We will not talk about hard-ware solution today.

Network can be pruned

- Networks are typically over-parameterized (there is significant redundant weights or neurons)
- Prune them!

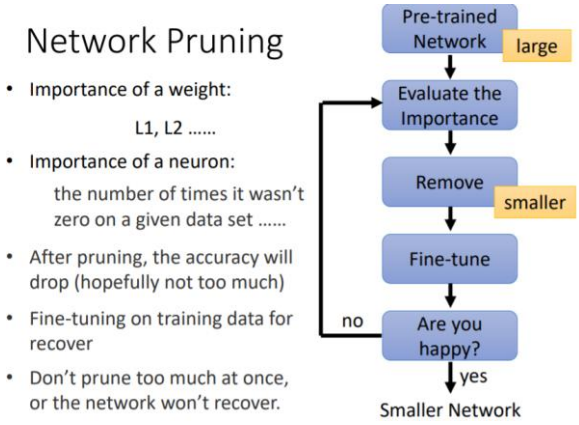




Network Pruning

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，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-tuning，如果一次性 remove 很多，network 的精确度也不会再恢复

Why Pruning?

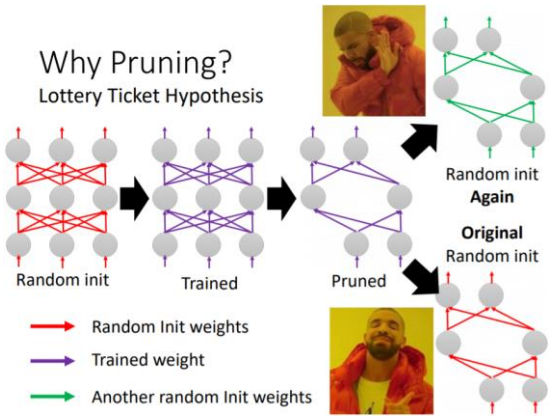
Q: 为什么不直接 train 一个小的 network 呢?

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 train 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-tuning 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**

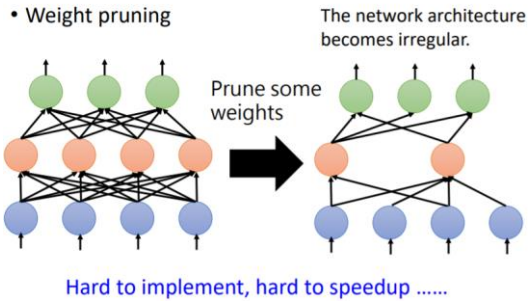
- Real random initialization, not original random initialization in "Lottery Ticket Hypothesis"
- Pruning algorithms could be seen as performing network architecture search

Practical issue

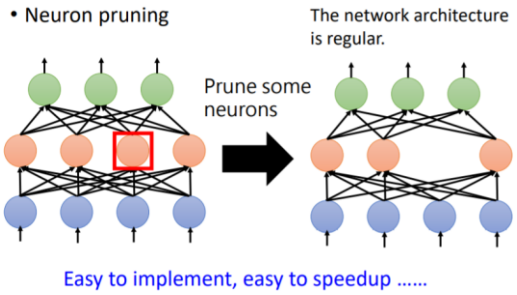
如果我们现在进行 weight pruning, 进行 weight pruning 之后的 network 会变得不规则, 有些 neural 有两个 weight,

有些 neural 有四个 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 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

Network Pruning - Practical Issue



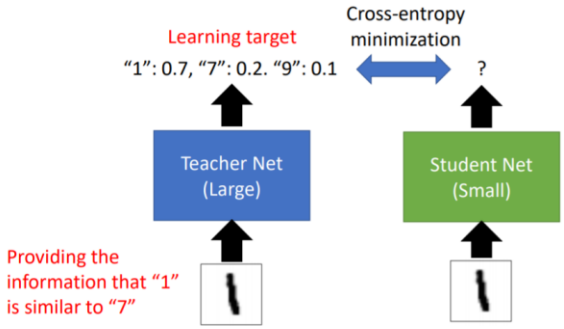
实际是做 weight pruning 是很麻烦的，通常我们都进行 neuron pruning，可以更好地进行 implement，也很容易进行 speedup



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

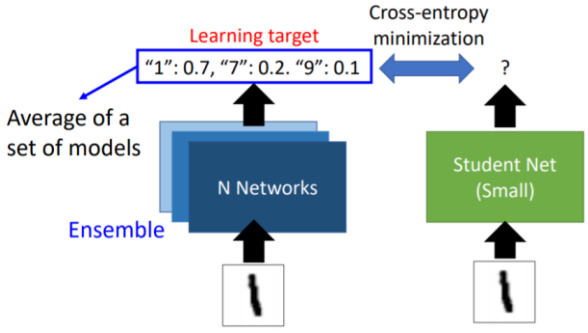


Q: 那么我们为什么要让 student 跟着 teacher 去学习呢?

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



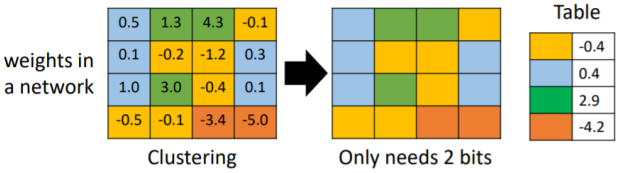
Temperature

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

$x_1 = 100$	$y_1 = 1$	$x_1/T = 1$	$y_1 = 0.56$
$x_2 = 10$	$y_2 \approx 0$	$x_2/T = 0.1$	$y_2 = 0.23$
$x_3 = 1$	$y_3 \approx 0$	$x_3/T = 0.01$	$y_3 = 0.21$

Parameter Quantization

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

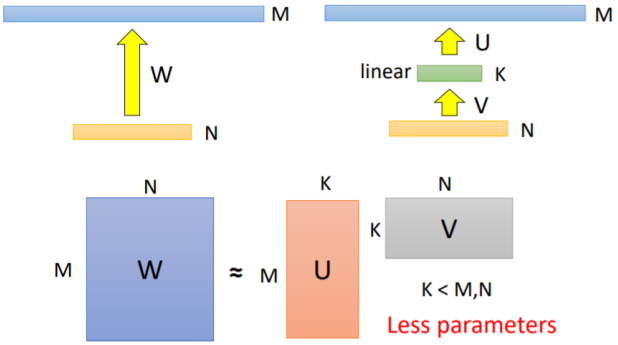


- 3. Represent frequent clusters by less bits, represent rare clusters by more bits
  - e.g. Huffman encoding

Architecture Design(most)

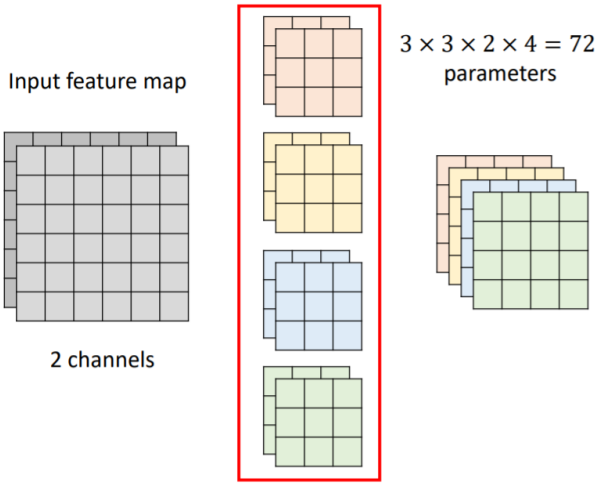
Low rank approximation

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

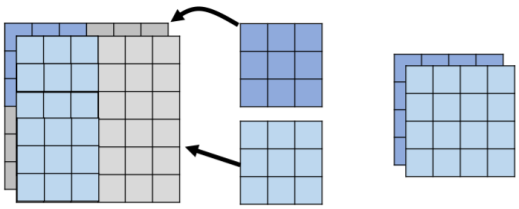
每个 filter 要处理所有的 channel



Depthwise Separable Convolution

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

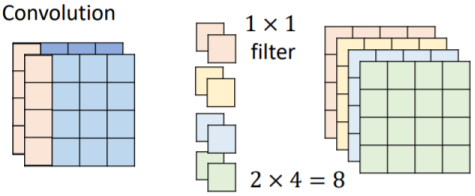
1. Depthwise Convolution



- Filter number = Input channel number
- Each filter only considers one channel.
- The filters are $k \times k$ matrices
- There is no interaction between channels.

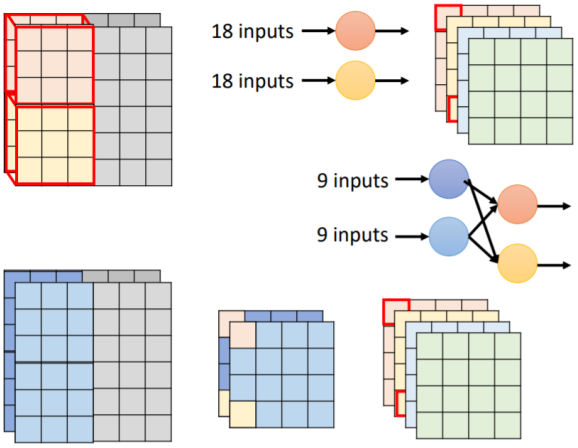
和一般的 convolution 是一样的，有 4 个 filter，就有 4 个不同的 matmix

2. Pointwise Convolution

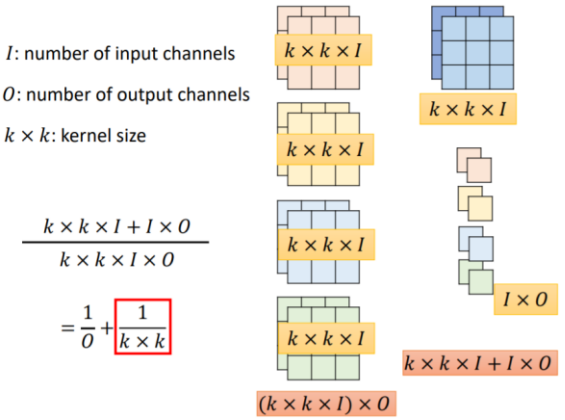


第一步用到的参数量为 3*3*2=18，第二步用到的参数为 2*4=8，一共有 26 个参数

Standard CNN vs Depthwise Separable Convolution

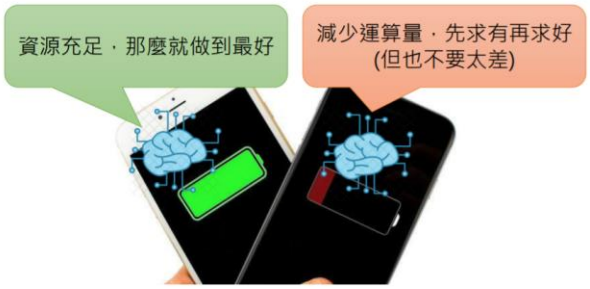


对于普通的卷积，需要的参数量为$(k * k * I) * O$;对于 Depthwise Separable Convolution，需要的参数量为$k * k * I + I * O$

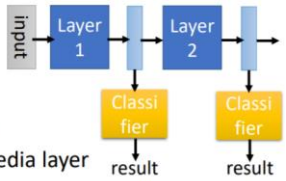


Dynamic Computation

• Can network adjust the computation power it need?



Possible Solutions



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