deeplearning.ai

Case Studies

Why look at case studies?

Outline

Classic networks:

- LeNet-5 ←
- AlexNet ←
- VGG ←

ResNet (152)

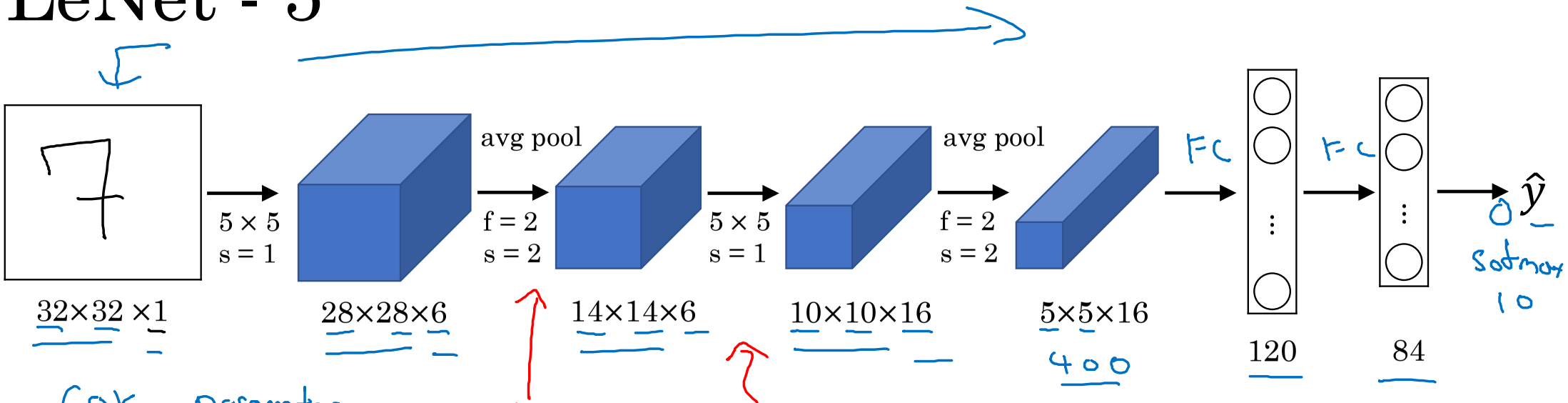
Inception

deeplearning.ai

Case Studies

Classic networks

LeNet - 5



60k parameters.

$n_H, n_w \downarrow$ $n_c \uparrow$

Non-linearity after pooling

$n_H \times n_w \times n_c$ $f \times f \times n_c$

conv pool conv pool fc fc output

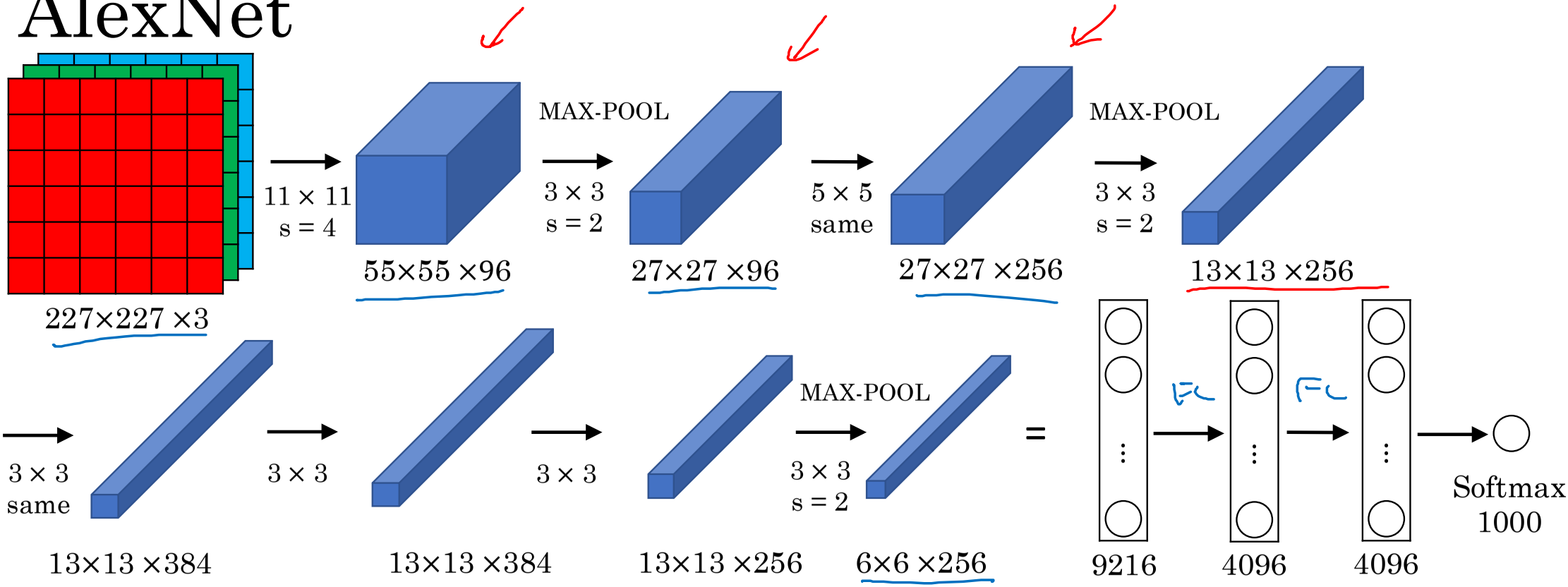
Activation: Sigmoid/tanh ReLU

II III

[LeCun et al., 1998. Gradient-based learning applied to document recognition]

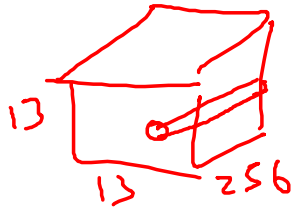
Andrew Ng

AlexNet



- Similar to LeNet, but much bigger.
- ReLU
- Multiple GPUs.
- Local Response Normalization (LRN)

9216



60M parameters

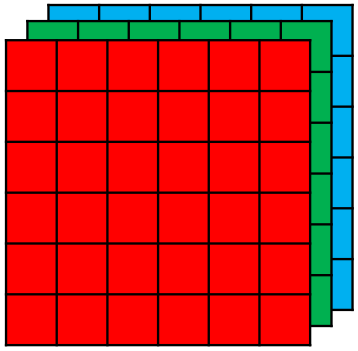
[Krizhevsky et al., 2012. ImageNet classification with deep convolutional neural networks]

Andrew Ng

VGG - 16

VGG-19

CONV = 3×3 filter, s = 1, same MAX-POOL = 2×2 , s = 2



224×224 ×3

[CONV 64] ×2

224×224×64

POOL

112×112 ×64

[CONV 128] ×2

112×112 ×128

POOL

56×56 ×128

[CONV 256] ×3

56×56 ×256

POOL

28×28 ×256

[CONV 512] ×3

28×28 ×512

POOL

14×14×512

[CONV 512] ×3

14×14 ×512

POOL

7×7×512

FC 4096

FC 4096

Softmax 1000

$n_H, n_W \downarrow$ $n_c \uparrow$ ~138M

[Simonyan & Zisserman 2015. Very deep convolutional networks for large-scale image recognition]

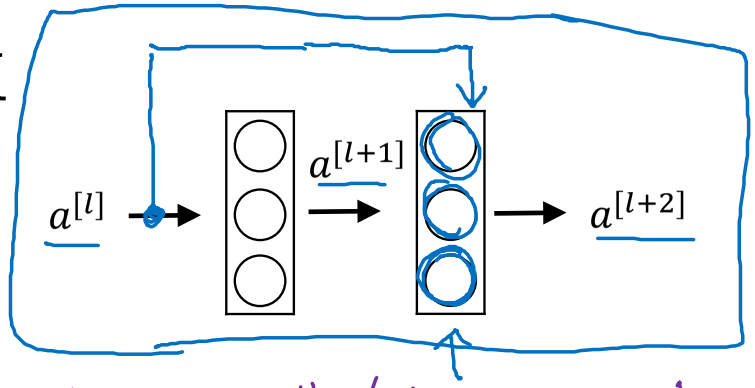
Andrew Ng

deeplearning.ai

Case Studies

Residual Networks (ResNets)

Residual block



"short cut" / skip connection

$a^{[l]}$ → Linear → ReLU $a^{[l+1]}$ → Linear → ReLU → $a^{[l+2]}$

"main pain"

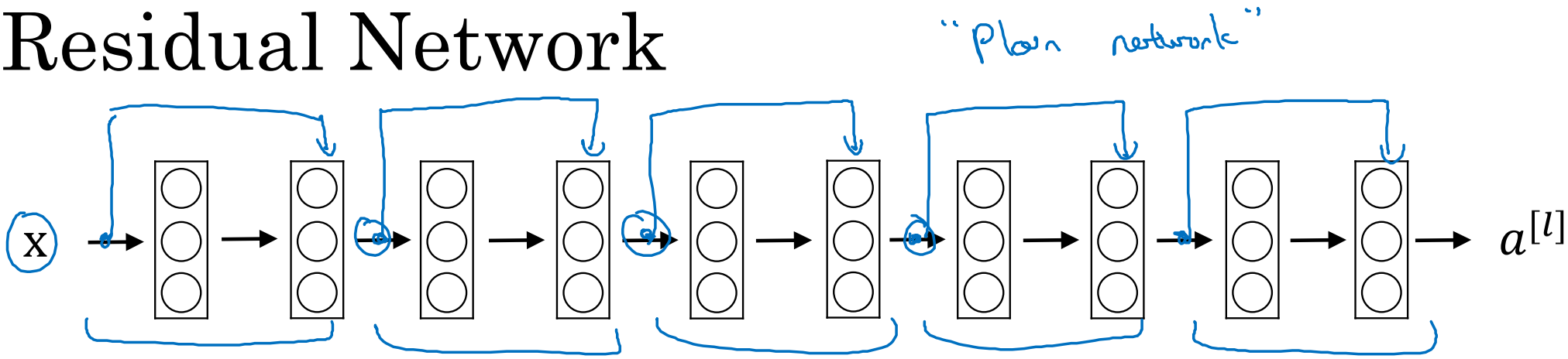
$$z^{[l+1]} = W^{[l+1]} a^{[l]} + b^{[l+1]} \qquad a^{[l+1]} = g(z^{[l+1]}) \qquad z^{[l+2]} = W^{[l+2]} a^{[l+1]} + b^{[l+2]} \qquad a^{[l+2]} = g(z^{[l+2]})$$

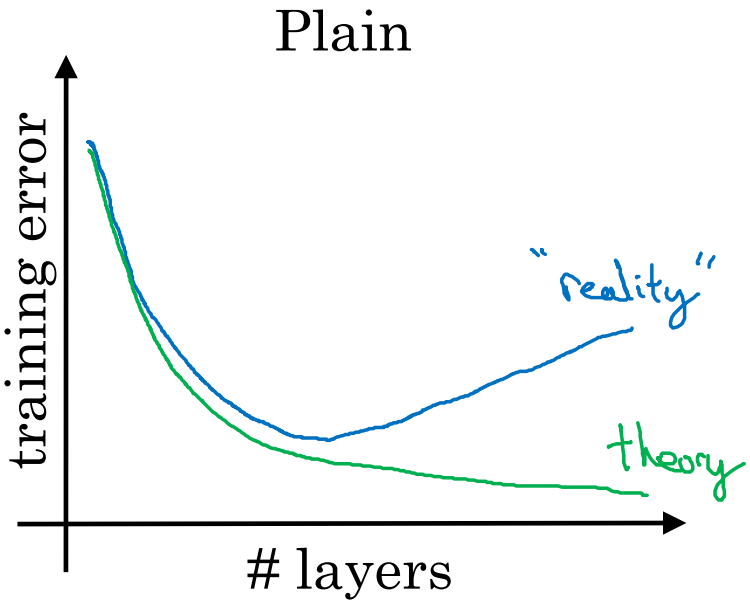
$$a^{[l+2]} = g(z^{[l+2]} + a^{[l]})$$

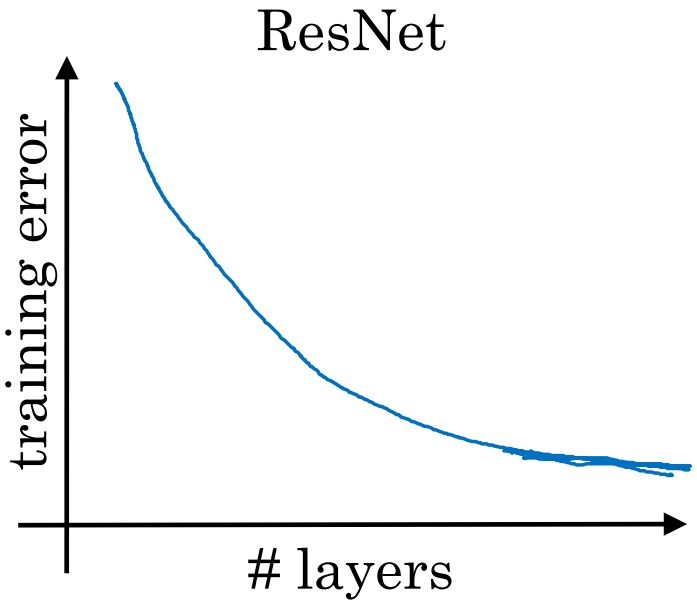
[He et al., 2015. Deep residual networks for image recognition]

Andrew Ng

Residual Network







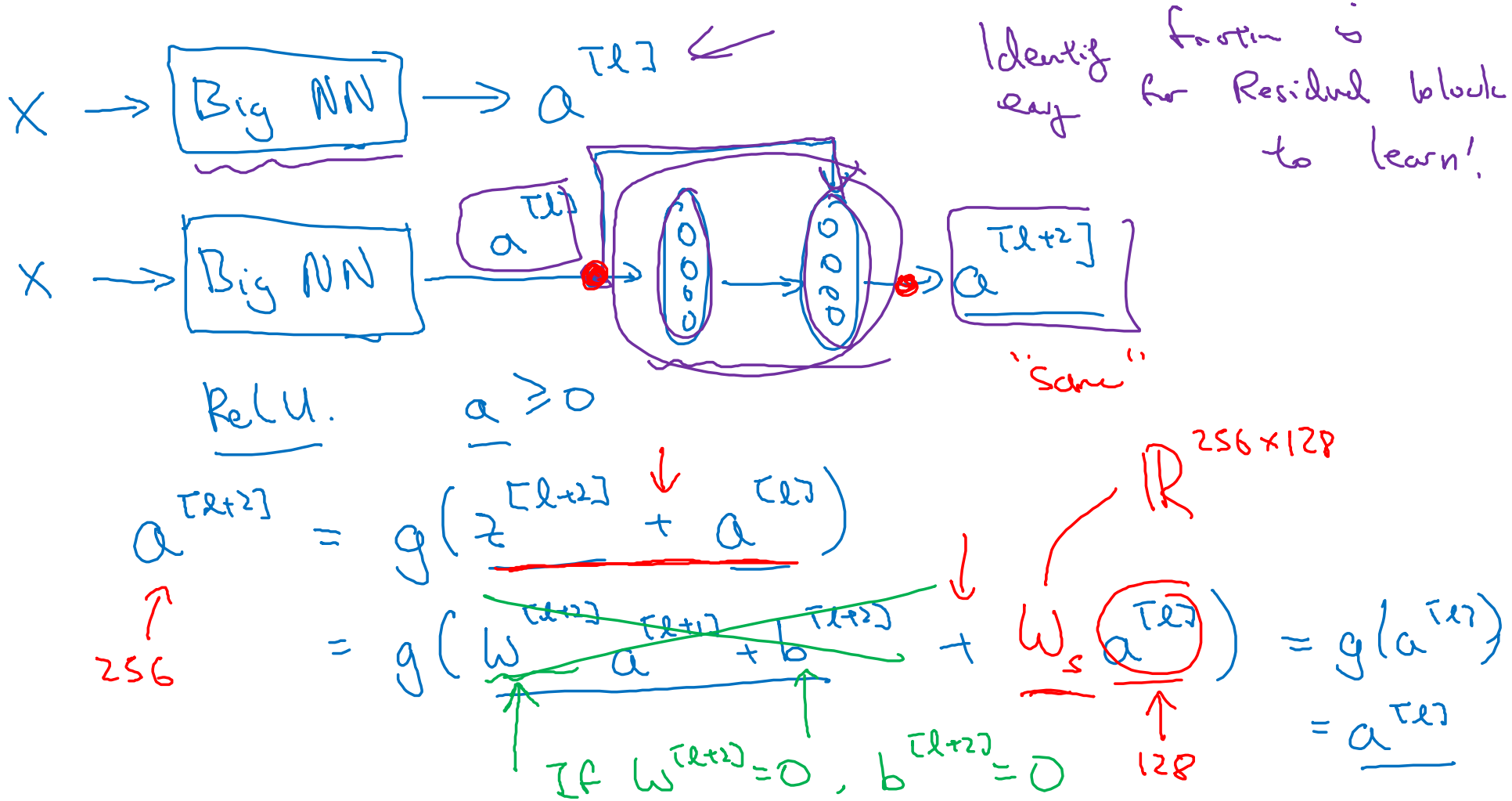
[He et al., 2015. Deep residual networks for image recognition]

deeplearning.ai

Case Studies

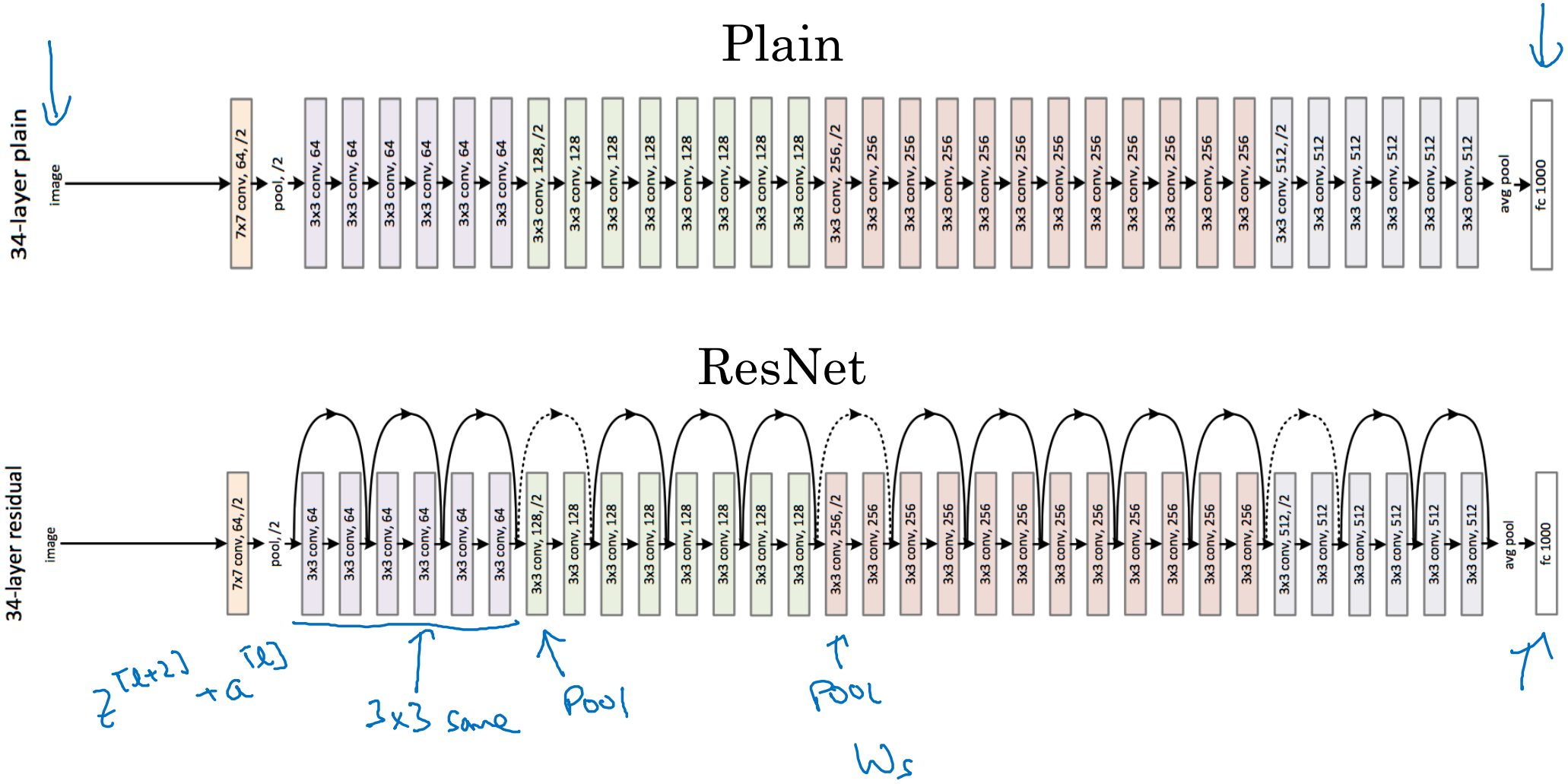
Why ResNets work

Why do residual networks work?



Andrew Ng

ResNet



[He et al., 2015. Deep residual networks for image recognition]

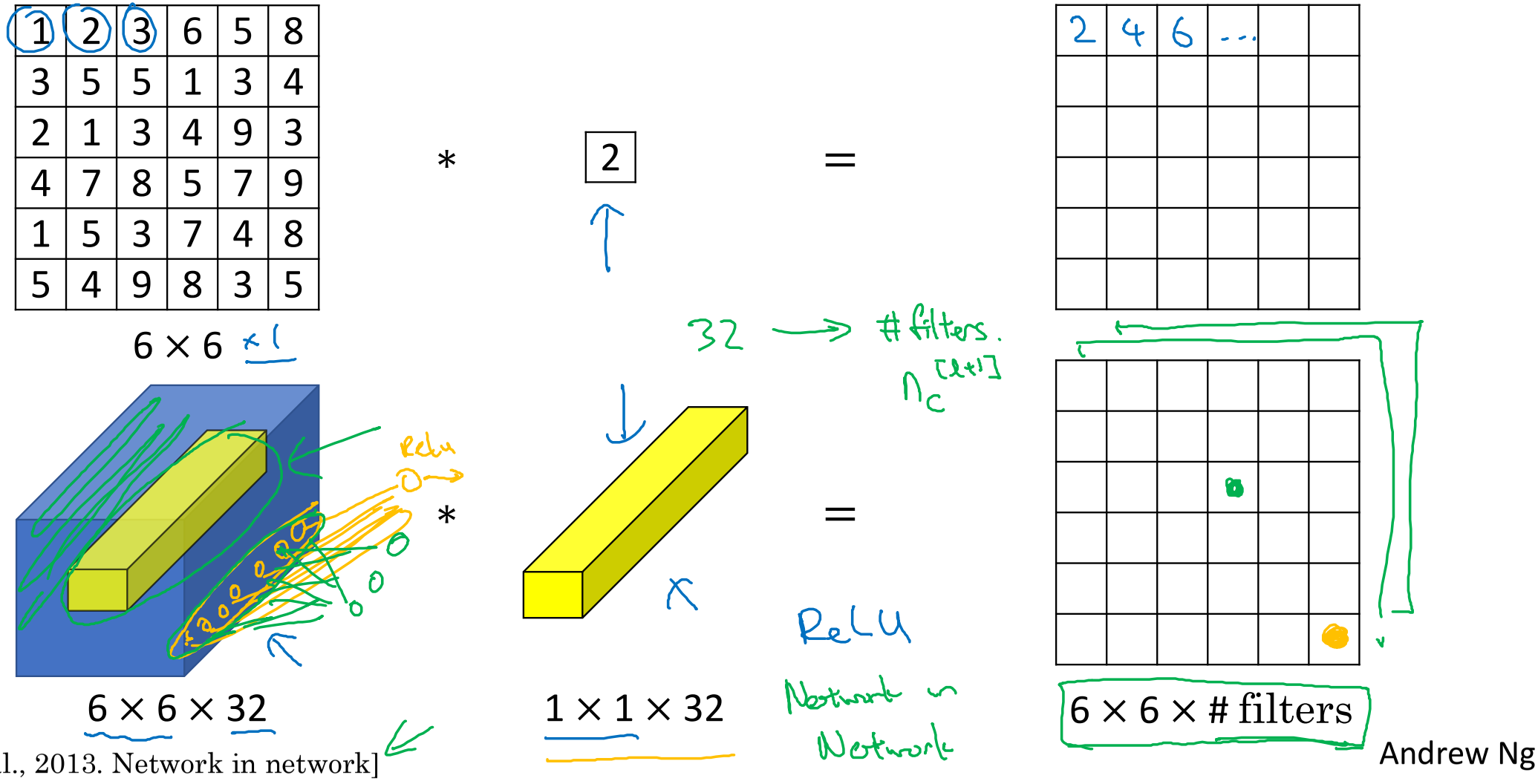
Andrew Ng

deeplearning.ai

Case Studies

Network in Network and 1×1 convolutions

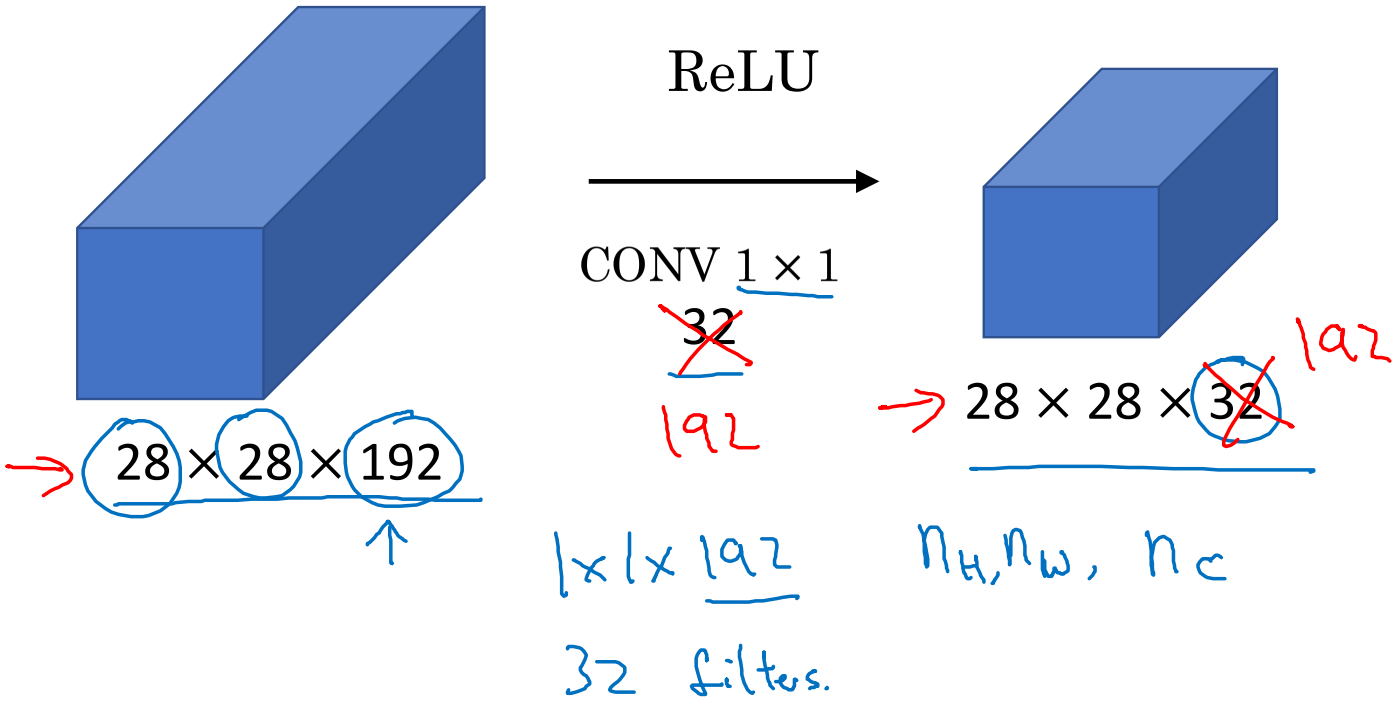
Why does a 1 × 1 convolution do?



[Lin et al., 2013. Network in network]

Andrew Ng

Using 1×1 convolutions



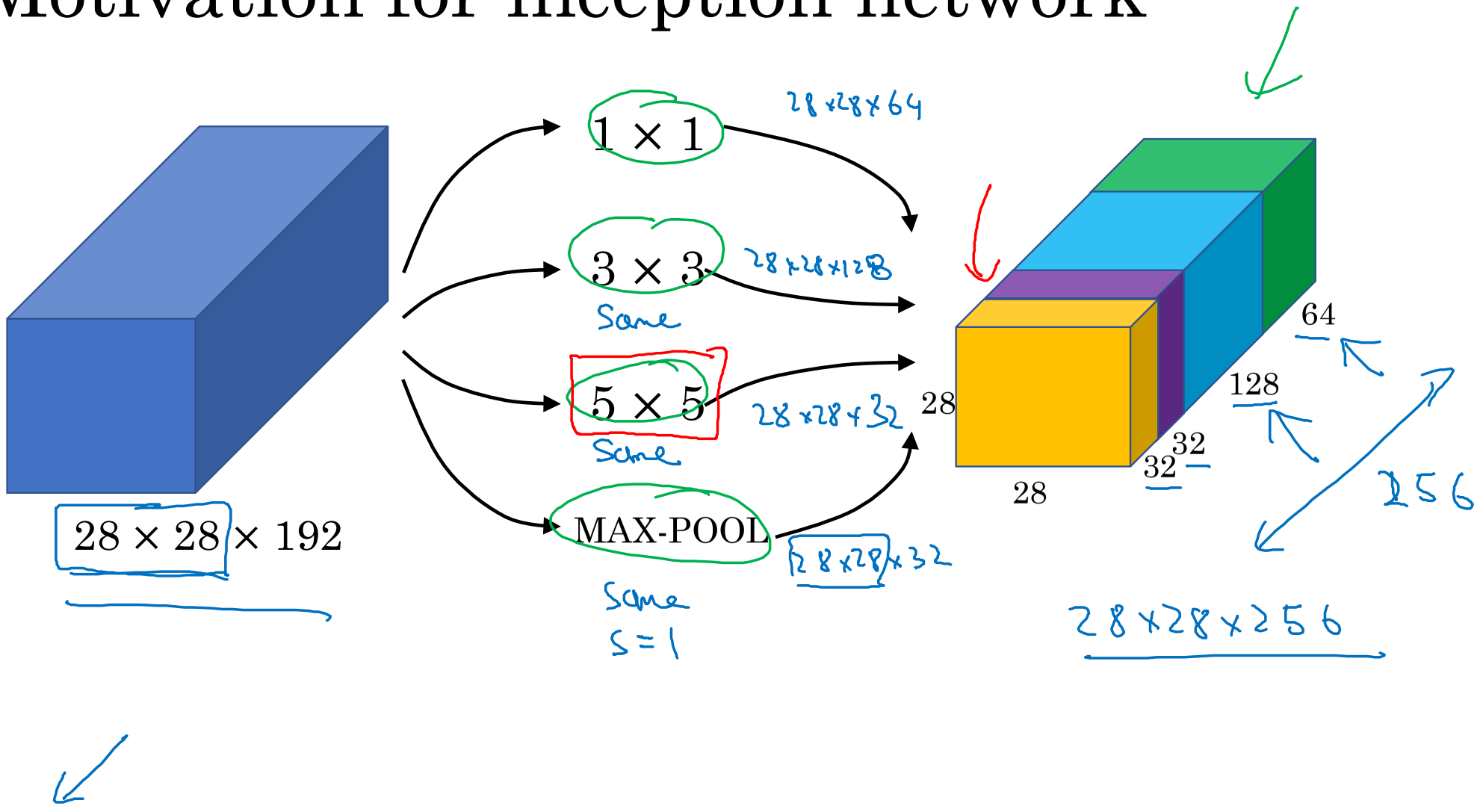
[Lin et al., 2013. Network in network]

deeplearning.ai

Case Studies

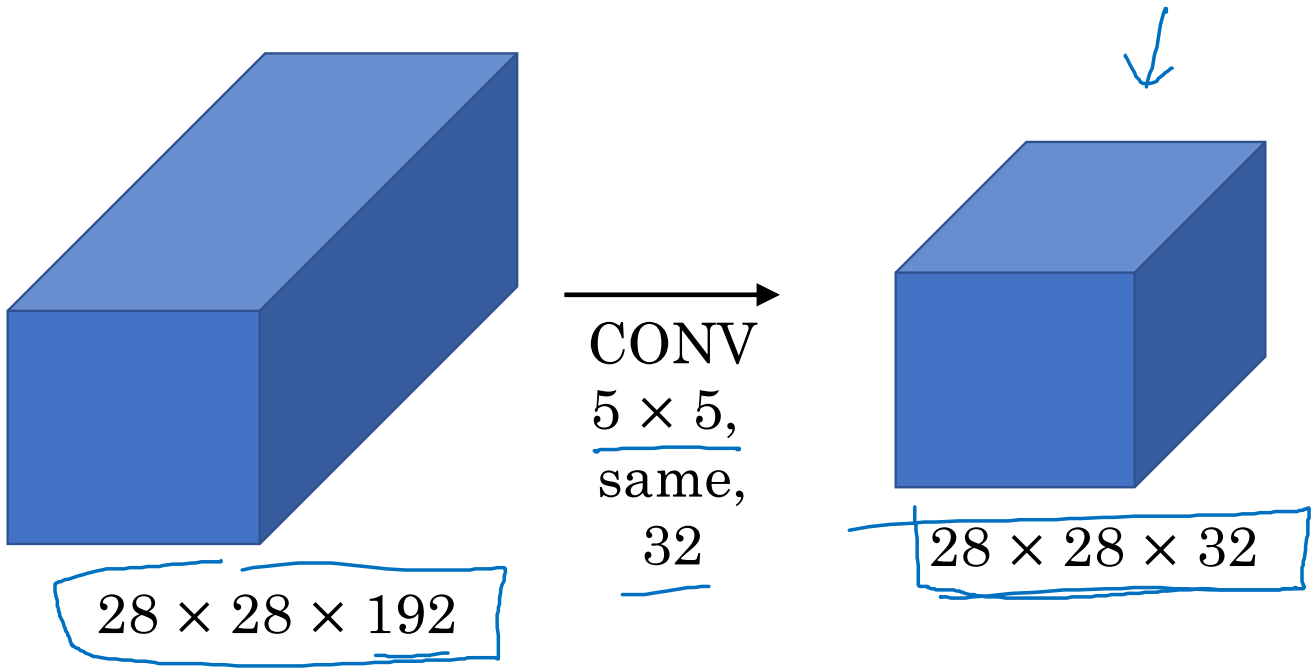
Inception network motivation

Motivation for inception network



[Szegedy et al. 2014. Going deeper with convolutions]

The problem of computational cost

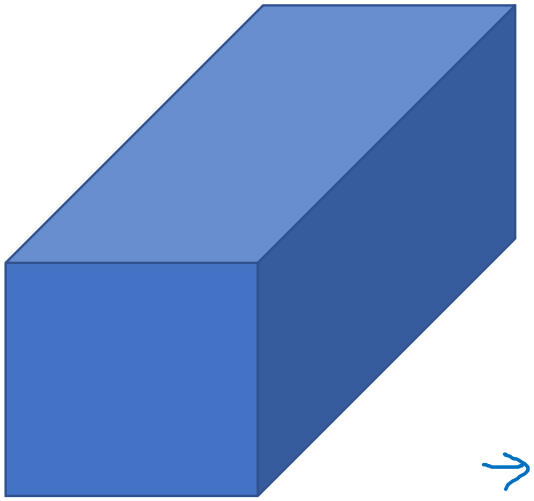


32 filters. filter one 5x5x192.

28x28x32 x 5x5x192 = 120M.

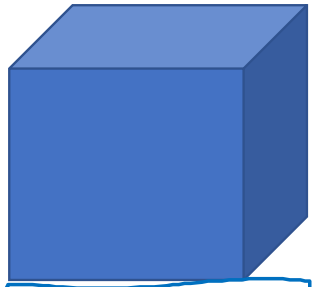
Using 1×1 convolution

"bottleneck layer"



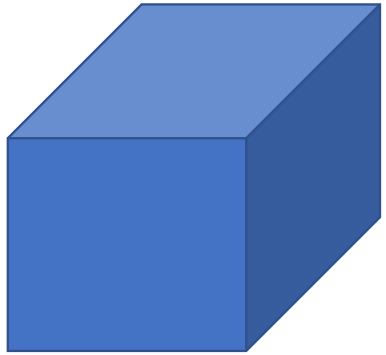
28 × 28 × 192

CONV
1 × 1,
16,
1 × 1 × 192



28 × 28 × 16

CONV
5 × 5,
32,
5 × 5 × 16



28 × 28 × 32

28×28×16 × 192 = 2.4M

28×28×32 × 5×5×16 = 10.0M

12.4M

120M

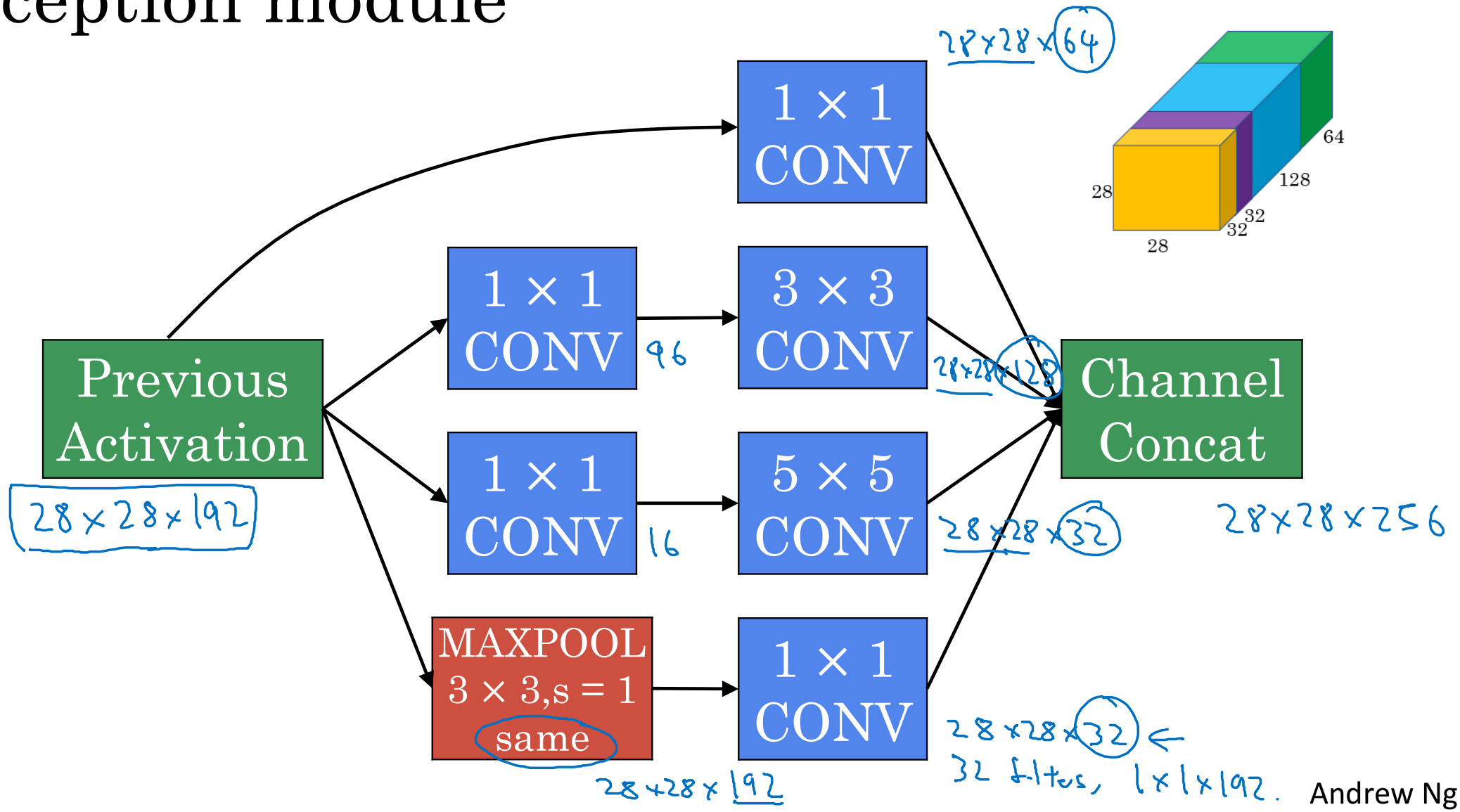
Andrew Ng

deeplearning.ai

Case Studies

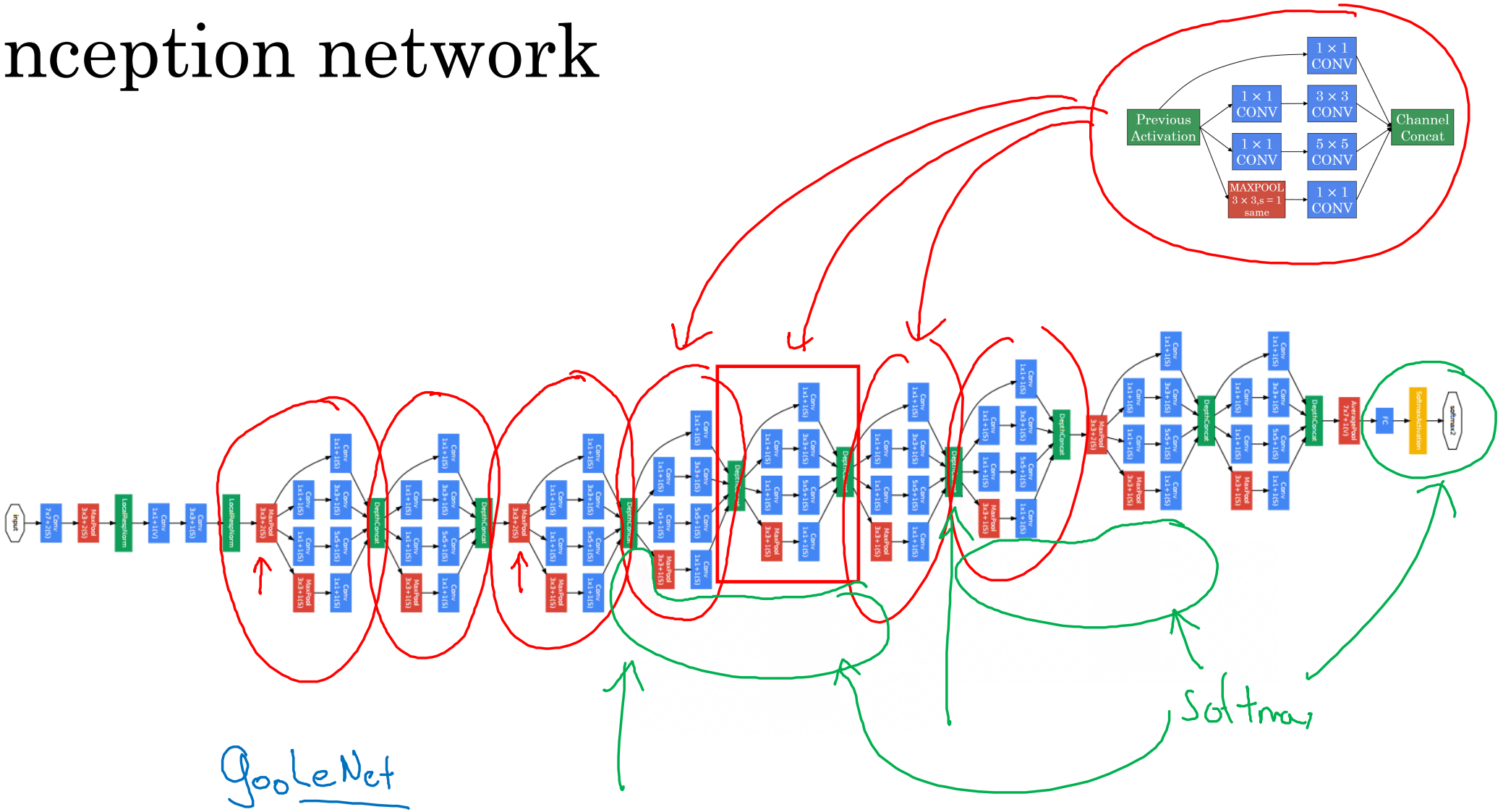
Inception network

Inception module



Andrew Ng

Inception network



[Szegedy et al., 2014, Going Deeper with Convolutions]

http://knowyourmeme.com/memes/we-need-to-go-deeper

Andrew Ng

Convolutional Neural Networks

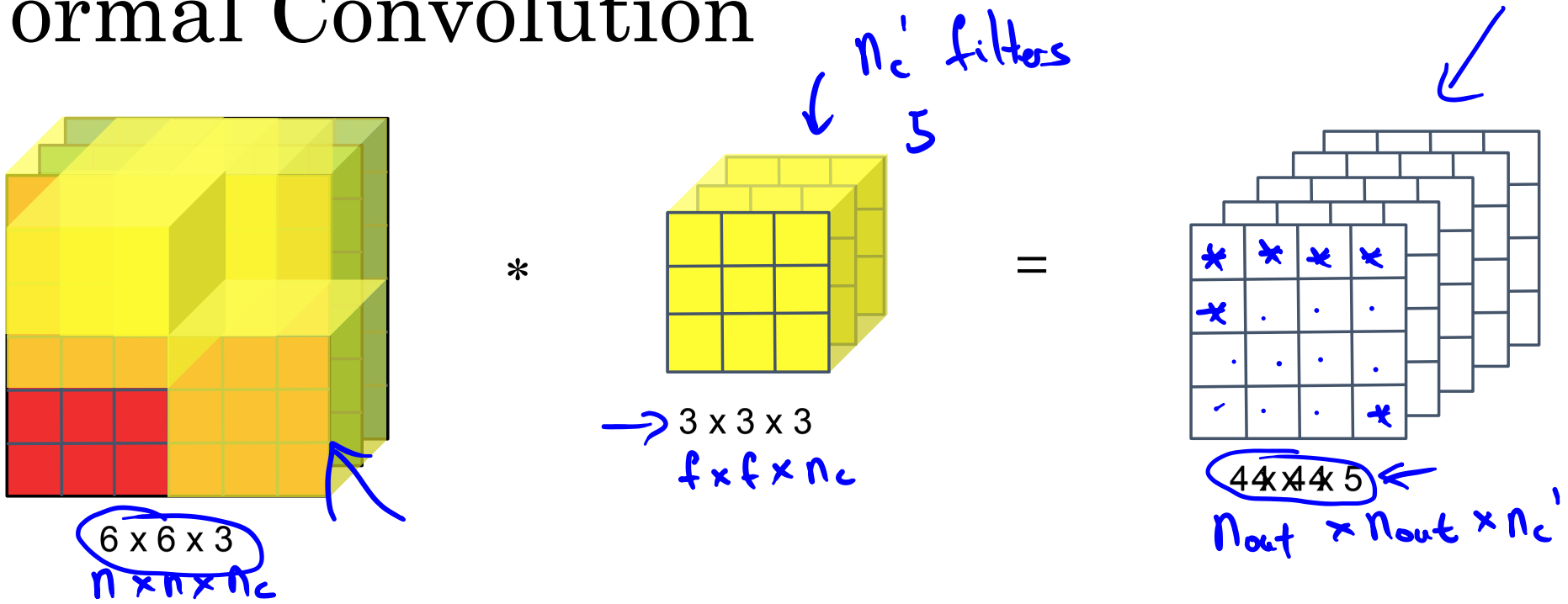
MobileNet

Motivation for MobileNets

- Low computational cost at deployment
- Useful for mobile and embedded vision applications
- Key idea: Normal vs. depthwise-separable convolutions

[Howard et al. 2017, MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications]

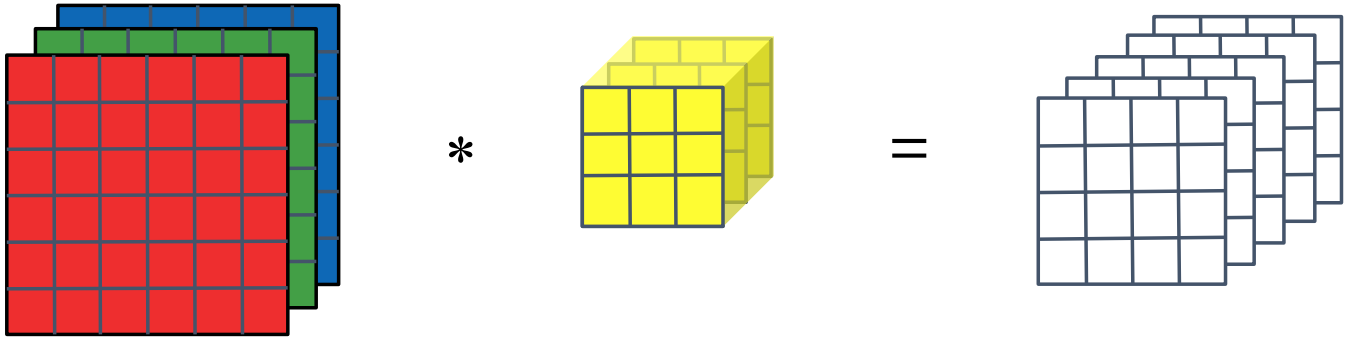
Normal Convolution



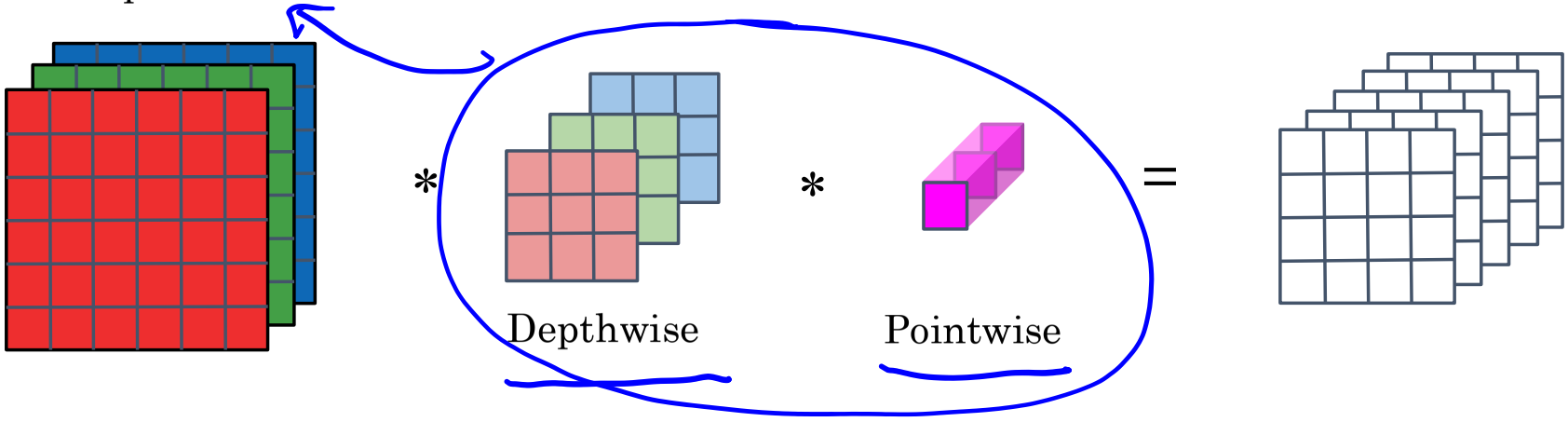
Computational cost	=	#filter params	x	# filter positions	x	# of filters
2160	=	3x3x3	x	4x4	x	5

Depthwise Separable Convolution

Normal Convolution

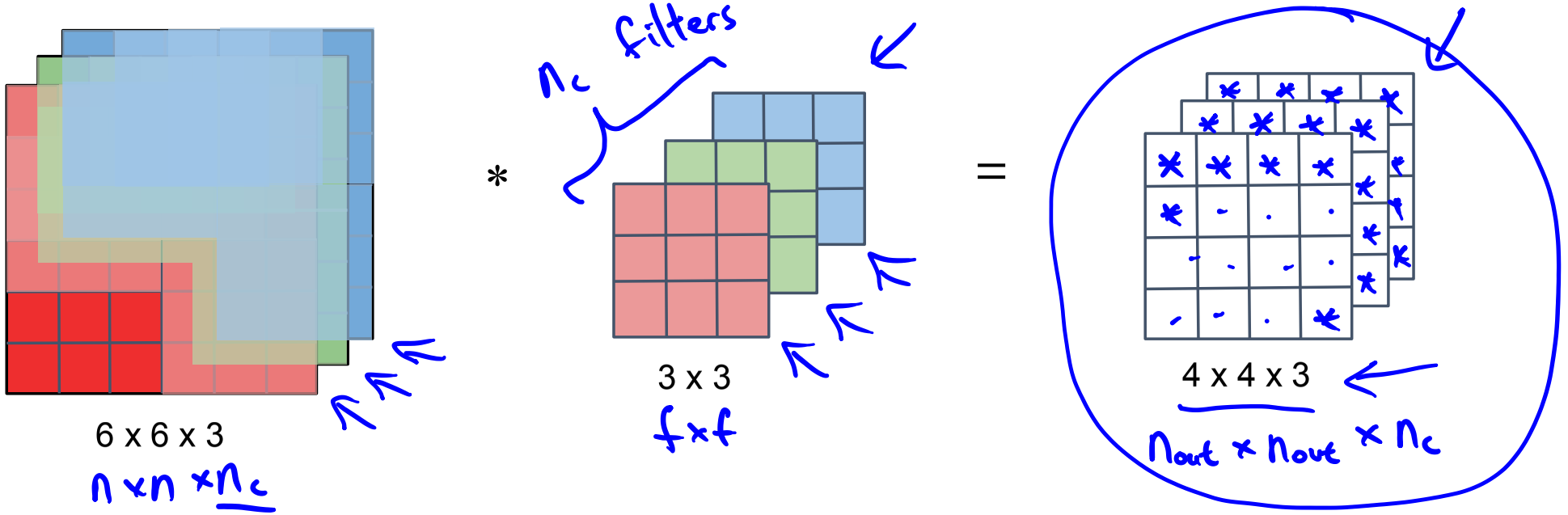


Depthwise Separable Convolution



Andrew Ng

Depthwise Convolution

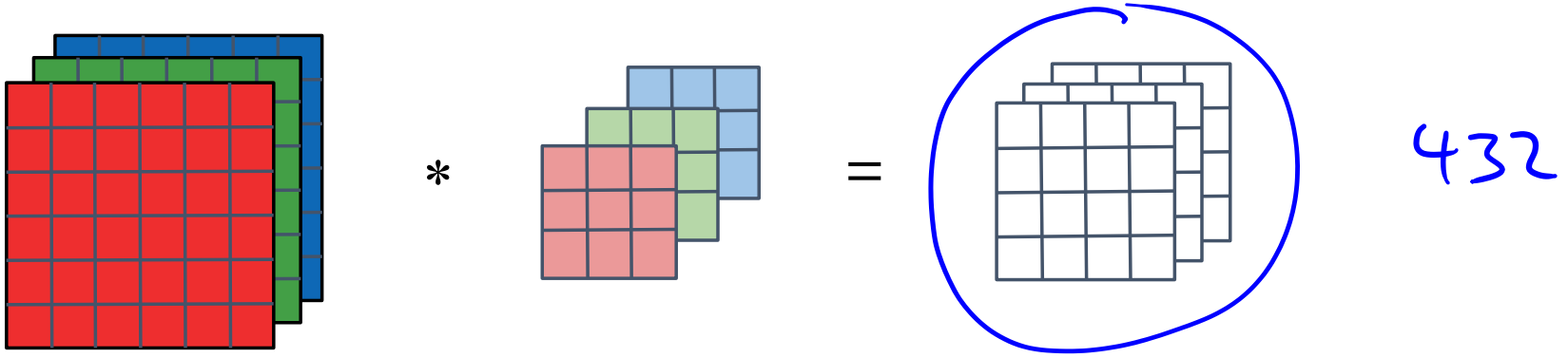


Computational cost	=	#filter params	x	# filter positions	x	# of filters
432	=	3x3	x	4x4	x	3

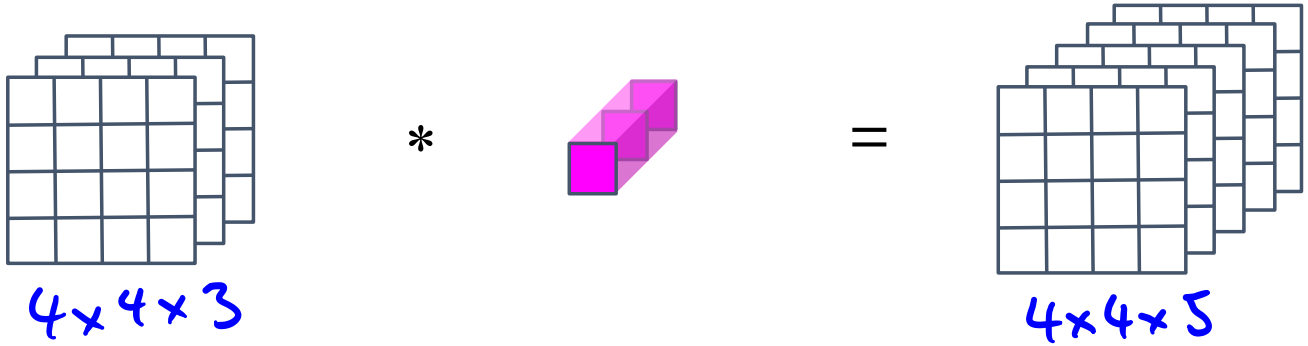
Andrew Ng

Depthwise Separable Convolution

Depthwise Convolution

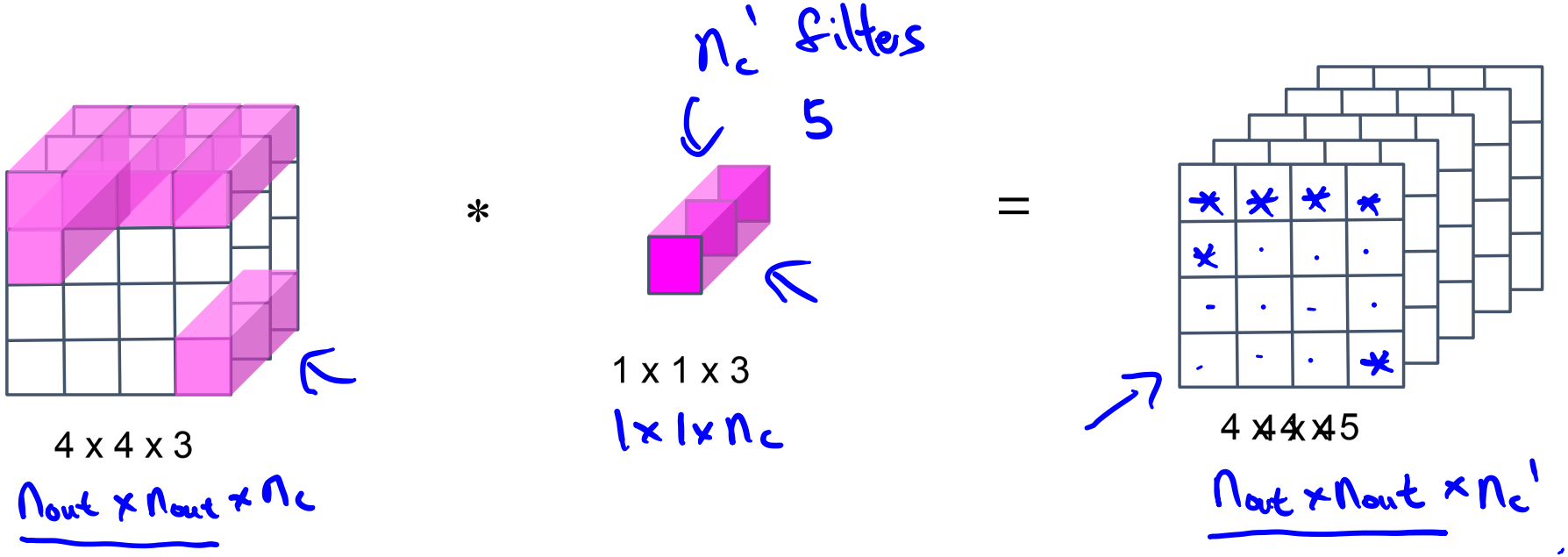


Pointwise Convolution



Andrew Ng

Pointwise Convolution

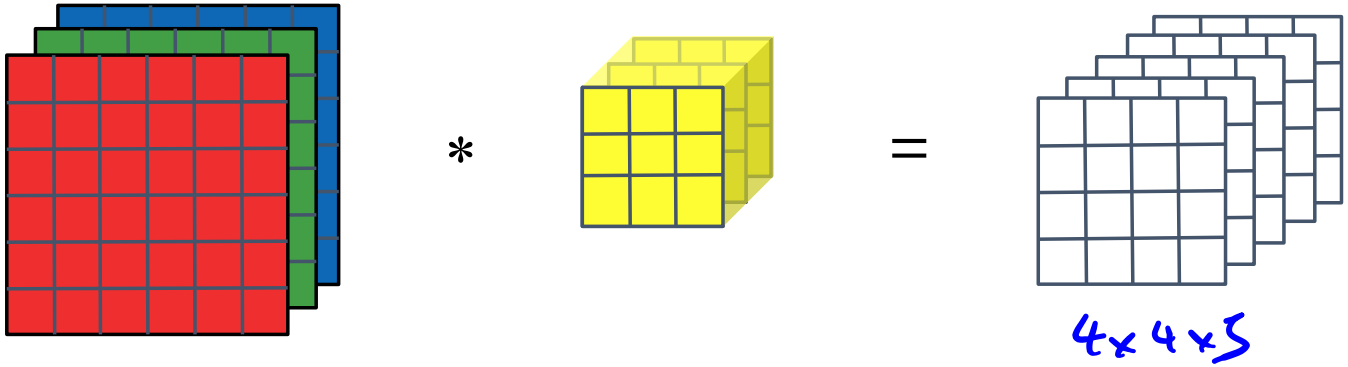


Computational cost	=	#filter params	x	# filter positions	x	# of filters
240	=	$1 \times 1 \times 3$	x	4×4	x	5

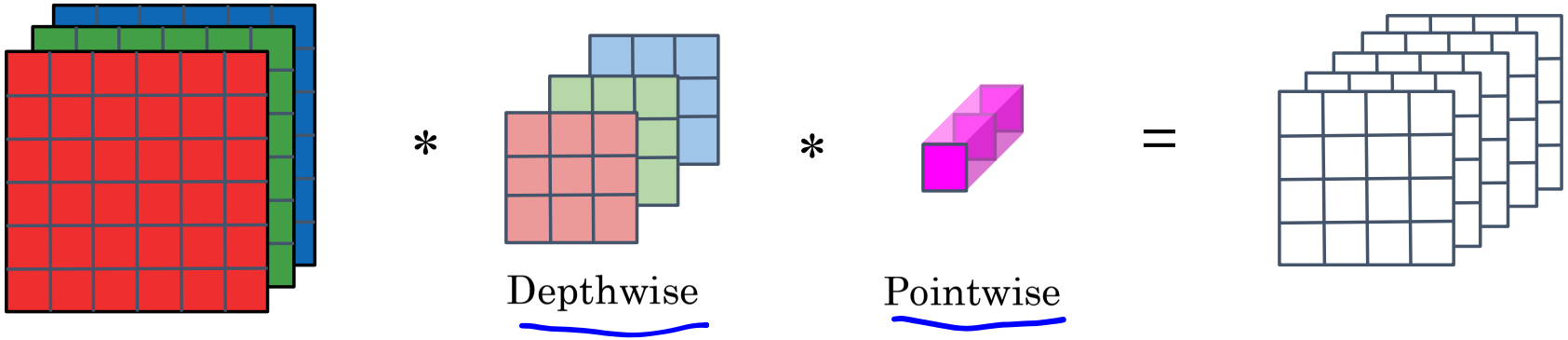
Andrew Ng

Depthwise Separable Convolution

Normal Convolution



Depthwise Separable Convolution



Andrew Ng

Cost Summary

Cost of normal convolution 2160

Cost of depthwise separable convolution

depthwise + pointwise

432 + 240 = 672

$$\frac{672}{2160} = 0.31$$

$$= \frac{1}{n_c} + \frac{1}{f^2}$$

$$\frac{1}{5} + \frac{1}{9}$$

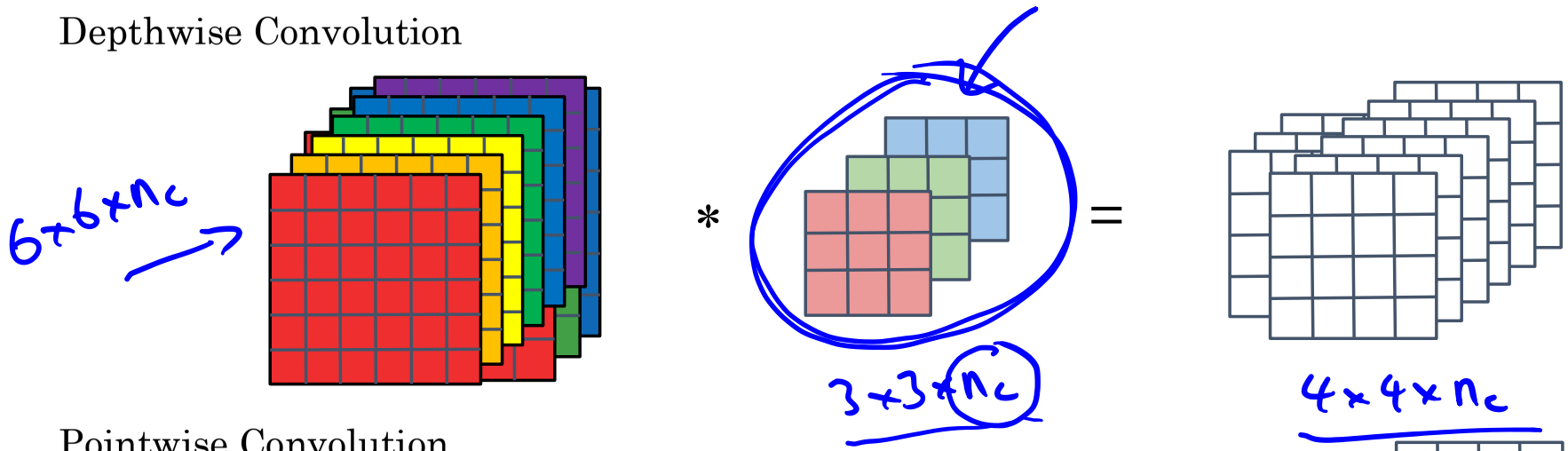
$$= \frac{1}{512} + \frac{1}{3^2}$$

≈ 10 times cheaper

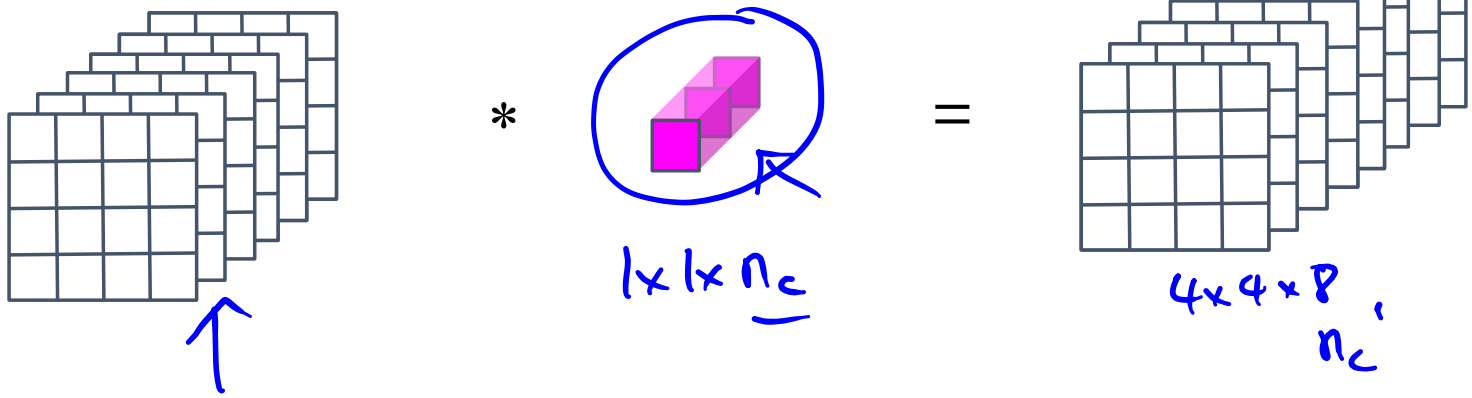
[Howard et al. 2017, MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications]

Depthwise Separable Convolution

Depthwise Convolution



Pointwise Convolution



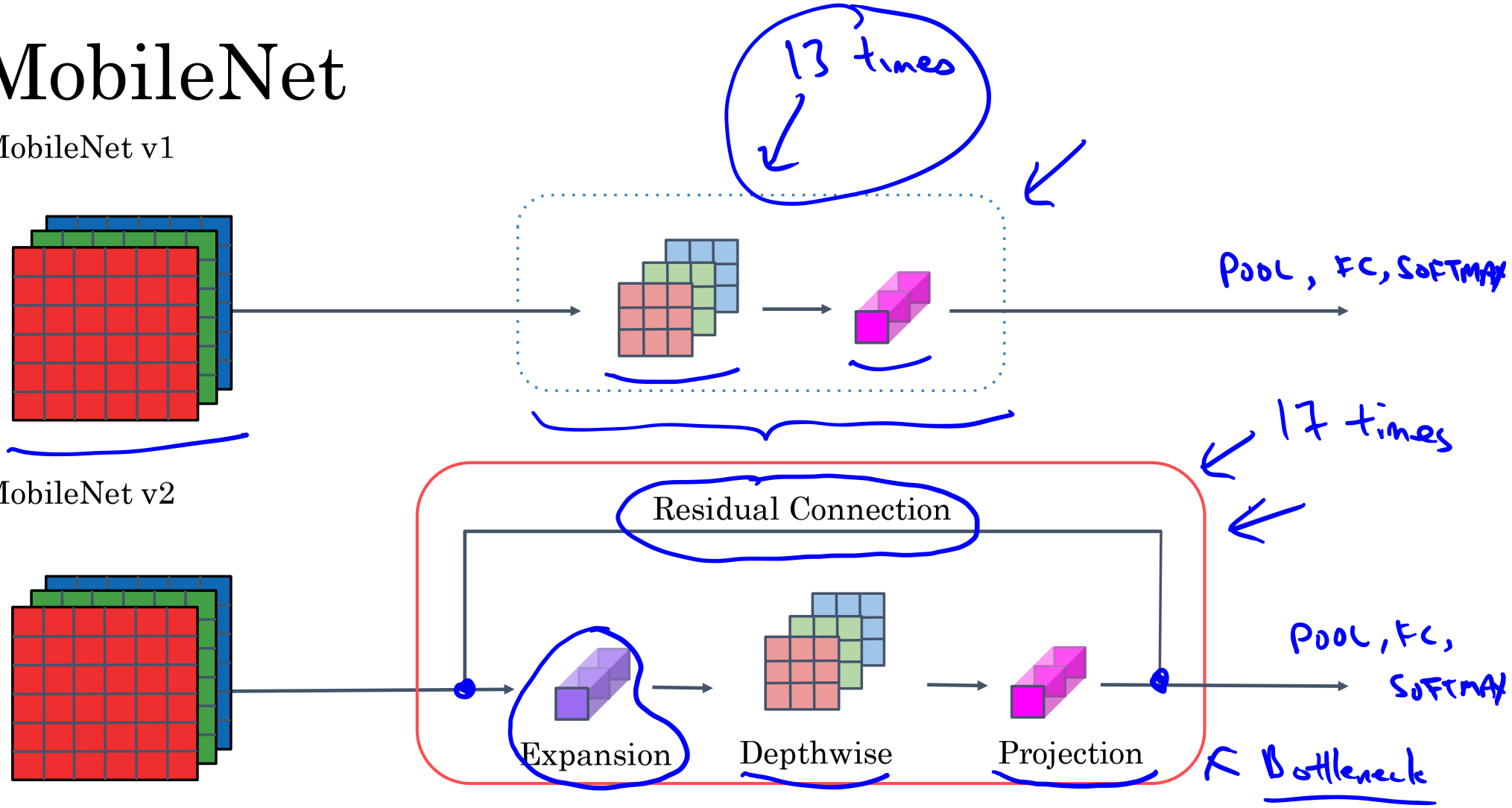
Andrew Ng

Convolutional Neural Networks

MobileNet Architecture

MobileNet

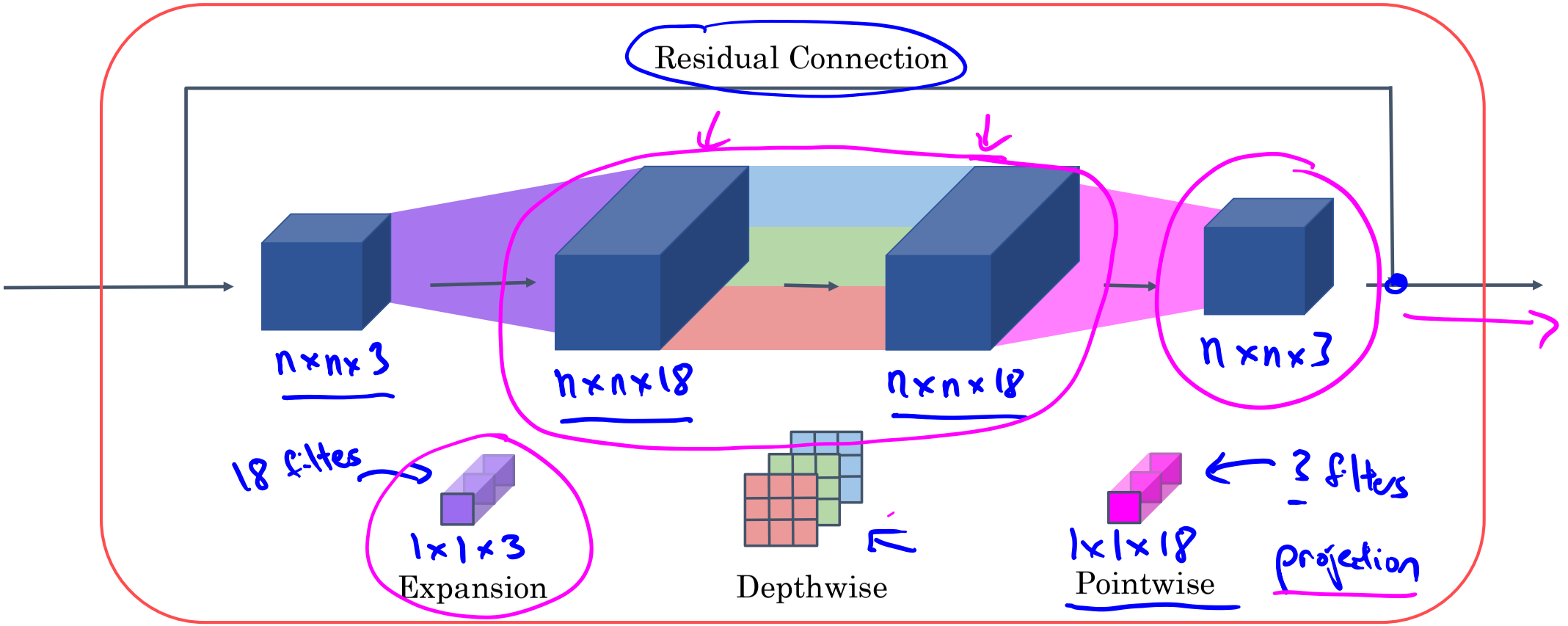
MobileNet v1



MobileNet v2

[Sandler et al. 2019, MobileNetV2: Inverted Residuals and Linear Bottlenecks]

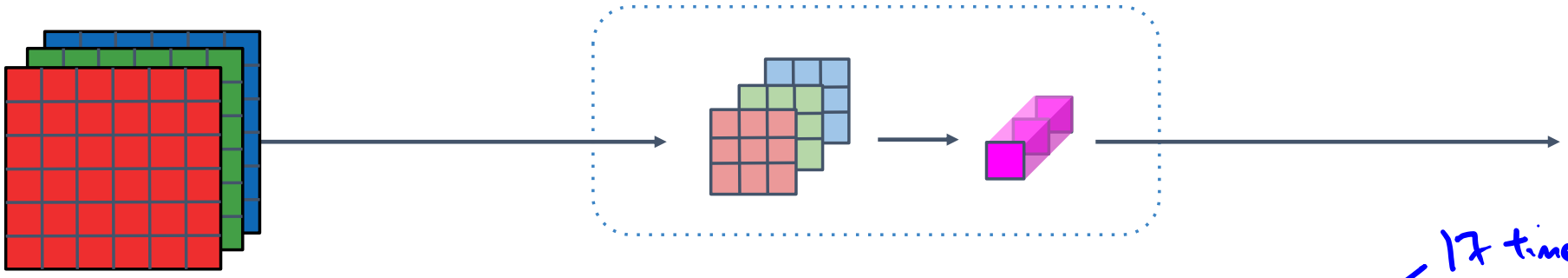
MobileNet v2 Bottleneck



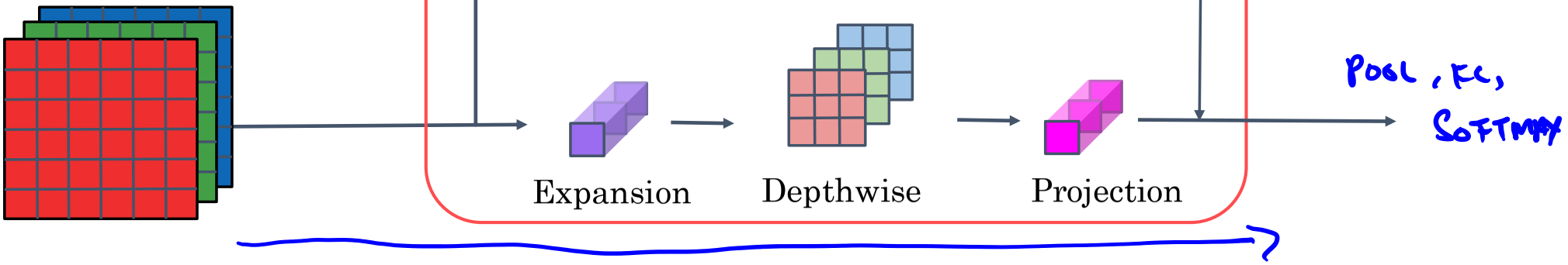
[Sandler et al. 2019, MobileNetV2: Inverted Residuals and Linear Bottlenecks]

MobileNet

MobileNet v1



MobileNet v2

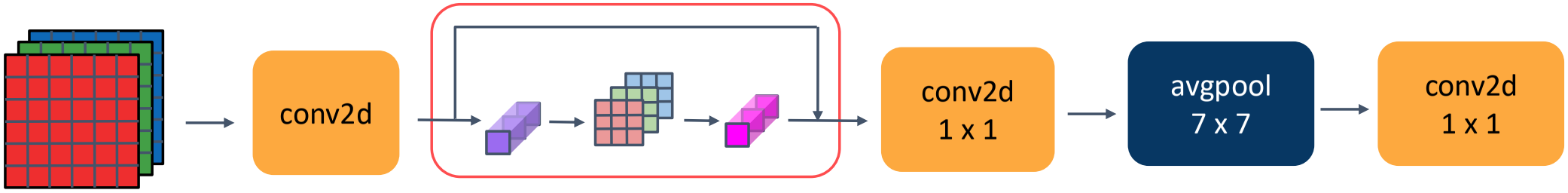


17 times

Pool, FC, Softmax

[Sandler et al. 2019, MobileNetV2: Inverted Residuals and Linear Bottlenecks]

MobileNet v2 Full Architecture

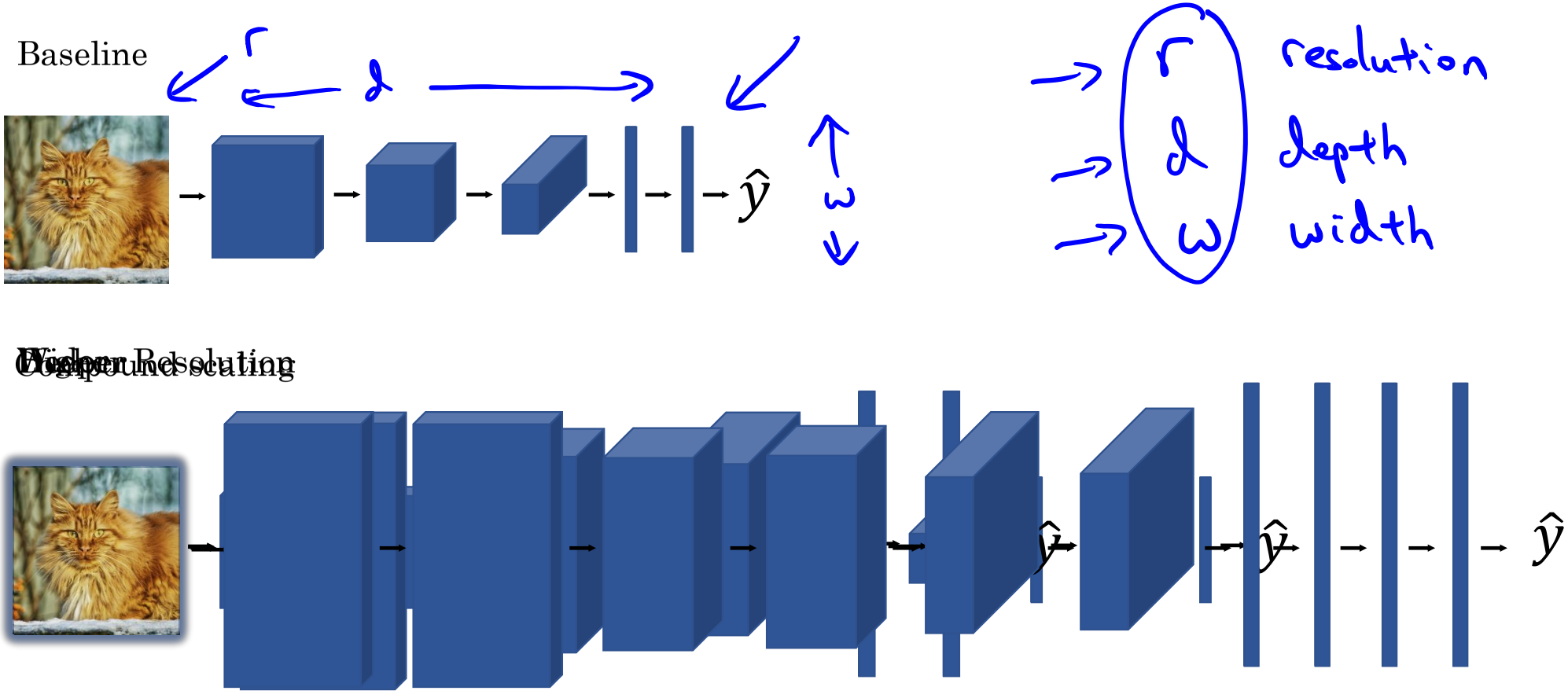


[Sandler et al. 2019, MobileNetV2: Inverted Residuals and Linear Bottlenecks]

Convolutional Neural Networks

EfficientNet

EfficientNet



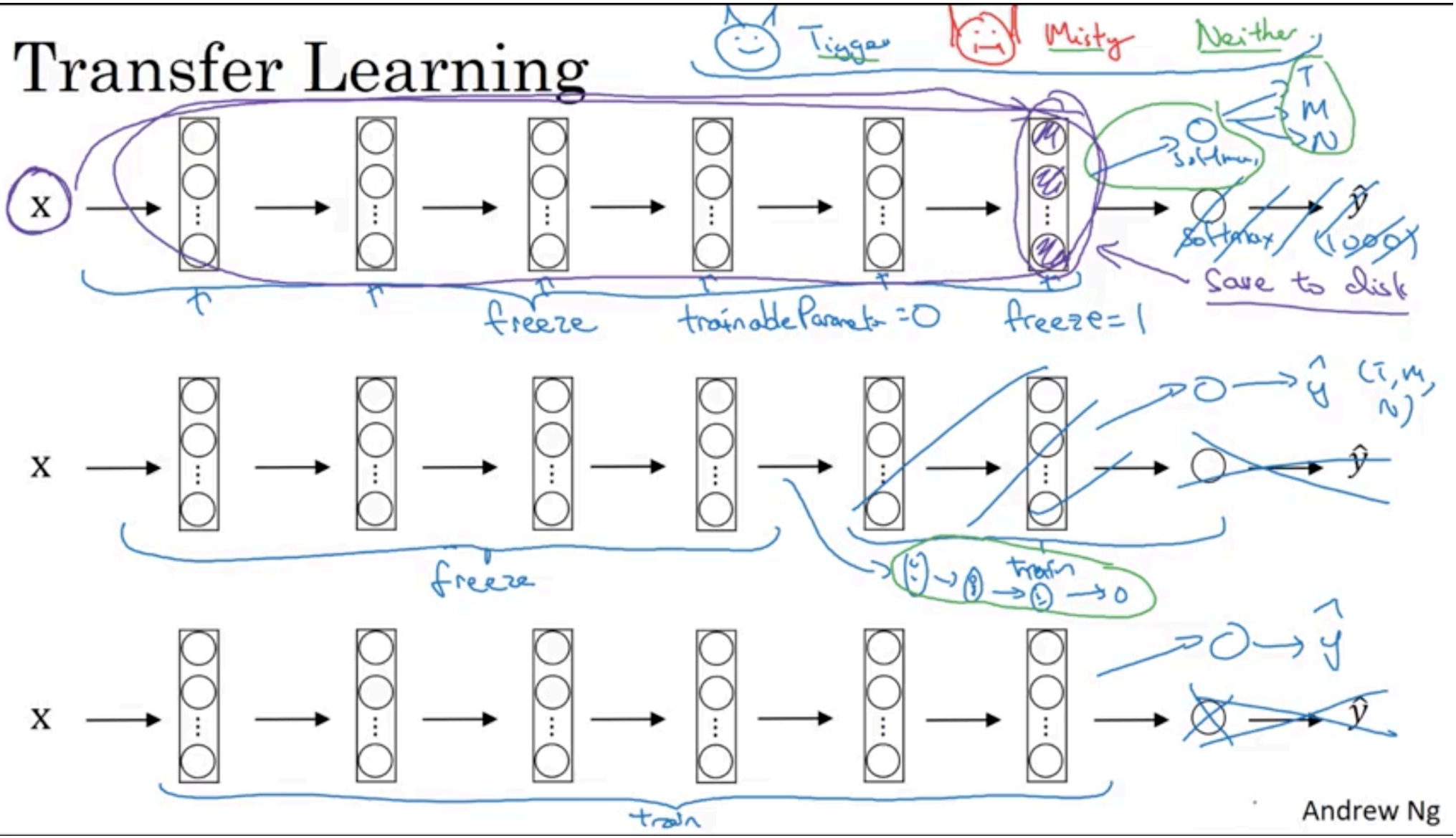
[Tan and Le, 2019, EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks]

deeplearning.ai

Practical advice for using ConvNets

Transfer Learning

Transfer Learning



Andrew Ng

deeplearning.ai

Practical advice for using ConvNets

Data augmentation

Common augmentation method

Mirroring

Random Cropping

- Rotation
- Shearing
- Local warping
- ...

Andrew Ng

Color shifting

R G B

+20,-20,+20

-20,+20,+20

+5,0,+50

Advanced:

PCA

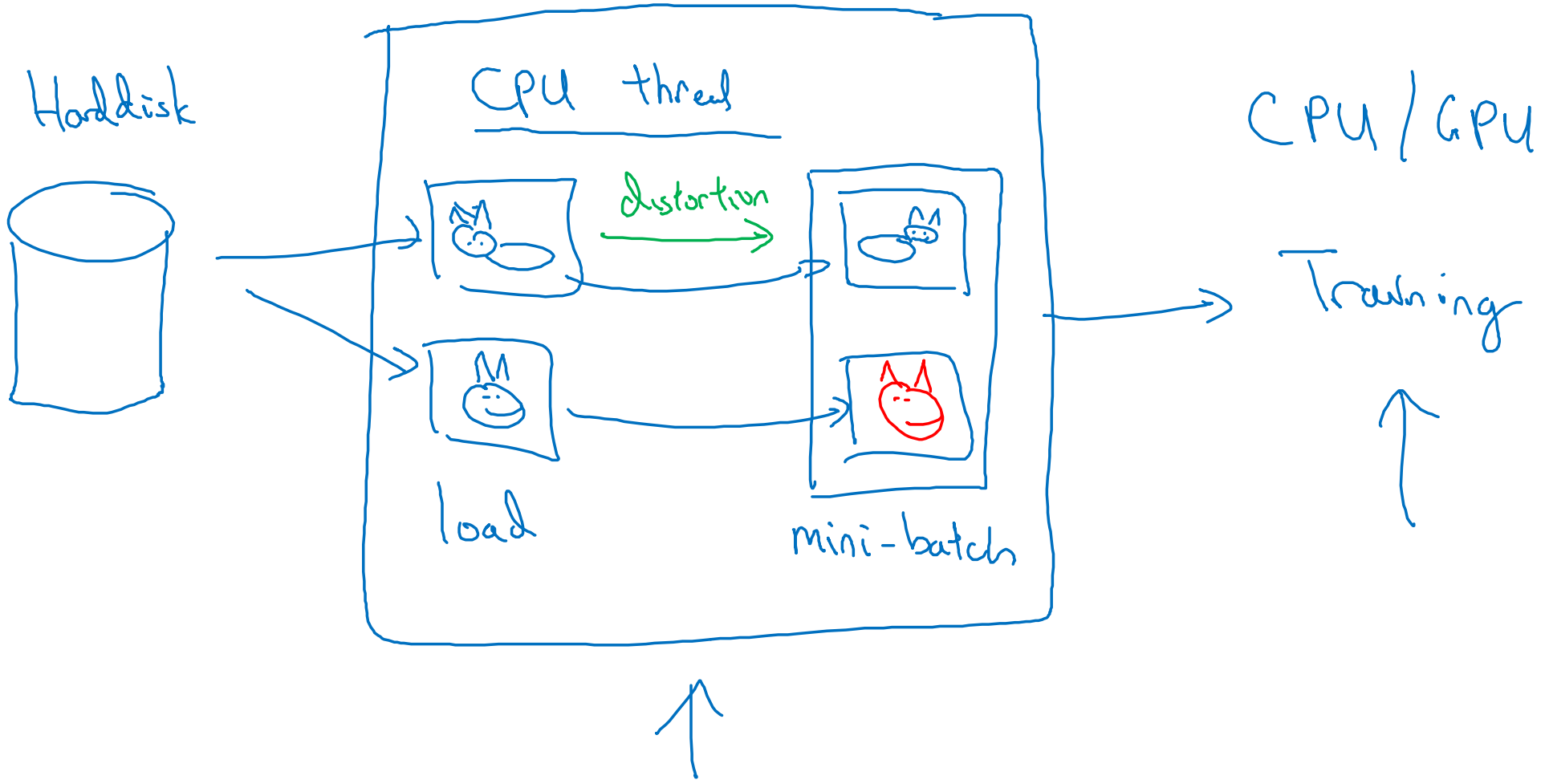
ml-class.org

[AlexNet paper

["PCA color augmentation."

R B G

Implementing distortions during training



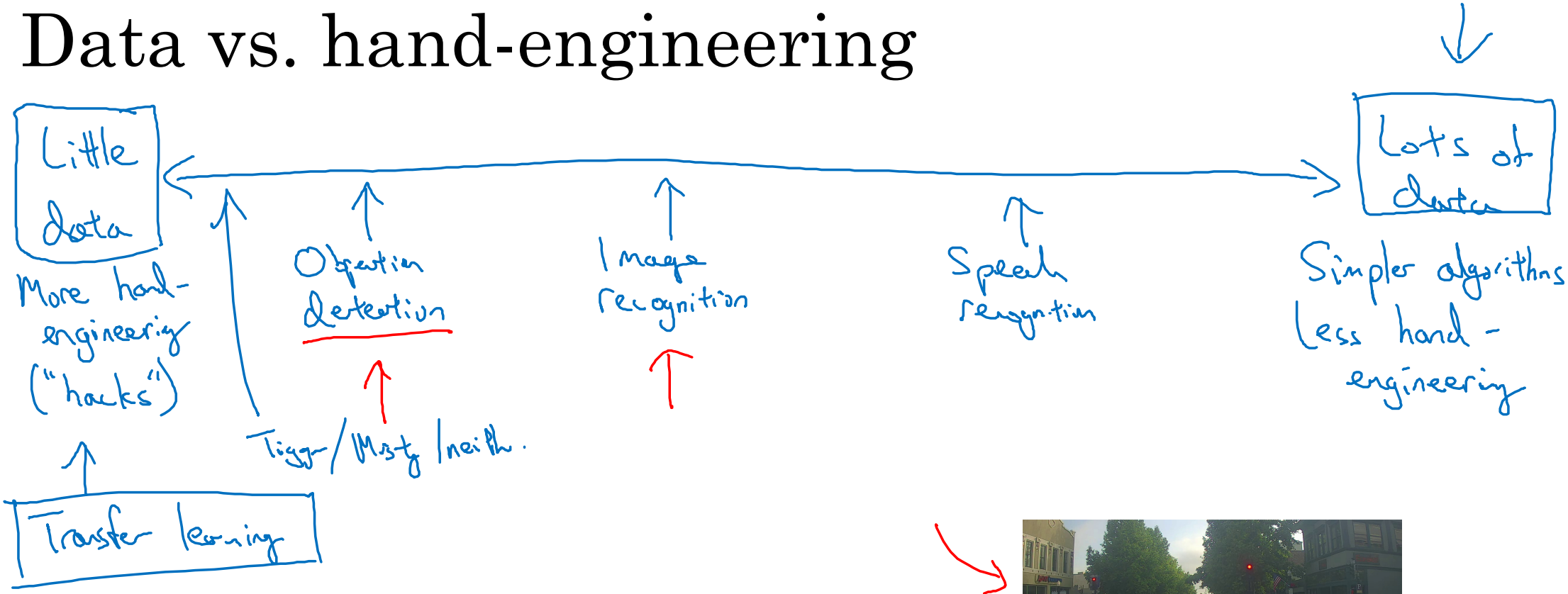
Andrew Ng

deeplearning.ai

Practical advice for using ConvNets

The state of computer vision

Data vs. hand-engineering



Two sources of knowledge

- Labeled data (x, y)
- Hand engineered features/network architecture/other components

Andrew Ng

Tips for doing well on benchmarks/winning competitions

Ensembling

3-15 networks → $\hat{y}$

- Train several networks independently and average their outputs

Multi-crop at test time

- Run classifier on multiple versions of test images and average results

10-crop

Andrew Ng

Use open source code

- Use architectures of networks published in the literature
- Use open source implementations if possible
- Use pretrained models and fine-tune on your dataset

Andrew Ng