

Ahmad Hussameldin Hamed Hassan

Shared Git-hub link: <https://github.com/ahmadhassan1993/sharing-github>

Convolutional Neural Network

Week 2 Summary

Case studies:

Outline

Classic networks:

- LeNet-5 ←
- AlexNet ←
- VGG ←

ResNet (152)

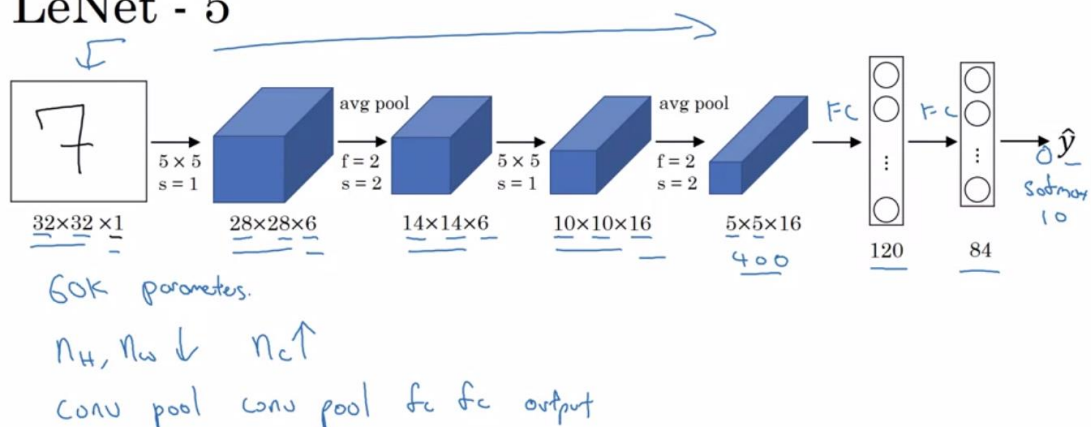
Inception

Andrew Ng

Classic CNNs

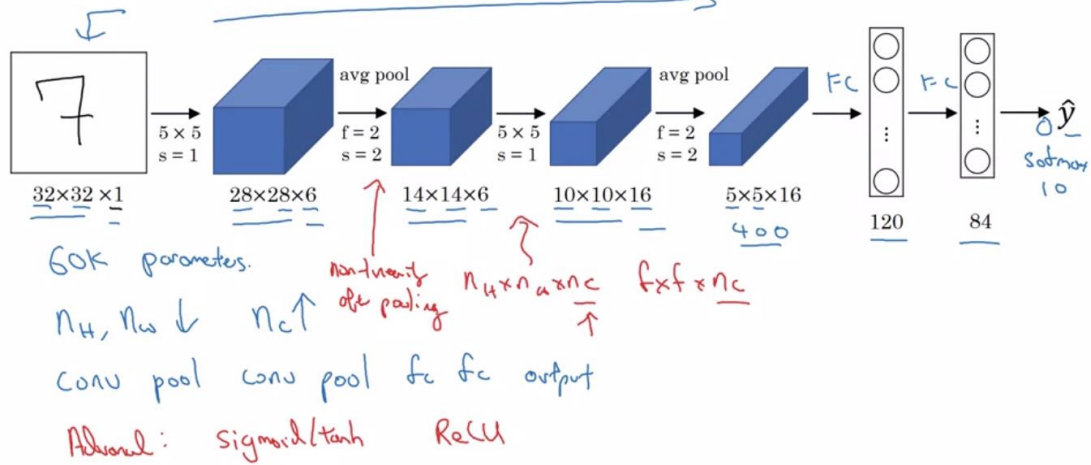
1) LeNet-5:

LeNet - 5



Some focal points where in the cited paper (in red color):

LeNet - 5



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

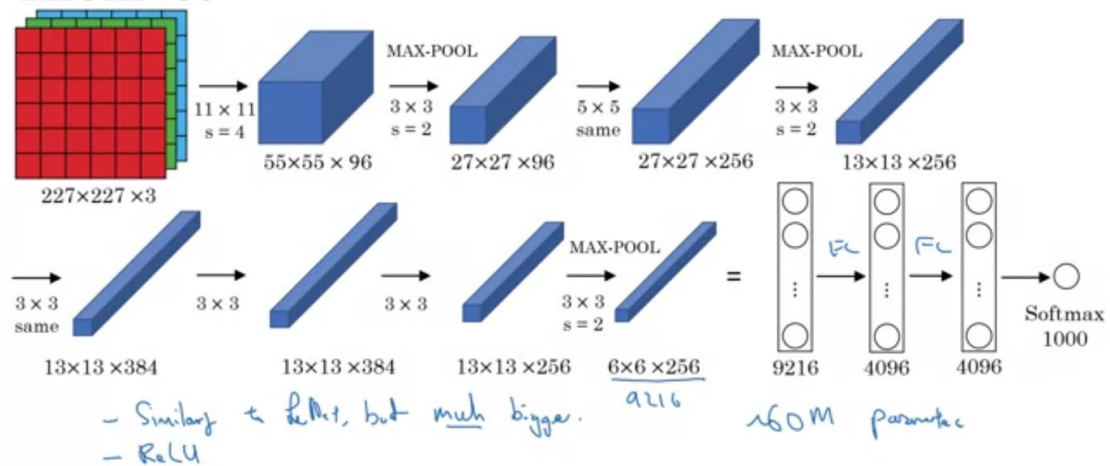
Andrew Ng

2) AlexNet:

It is better than LeNet, as it is:

- 1- Bigger (60 million parameters instead of 60 thousands)
- 2- Use Relu Activation in hidden layers

AlexNet

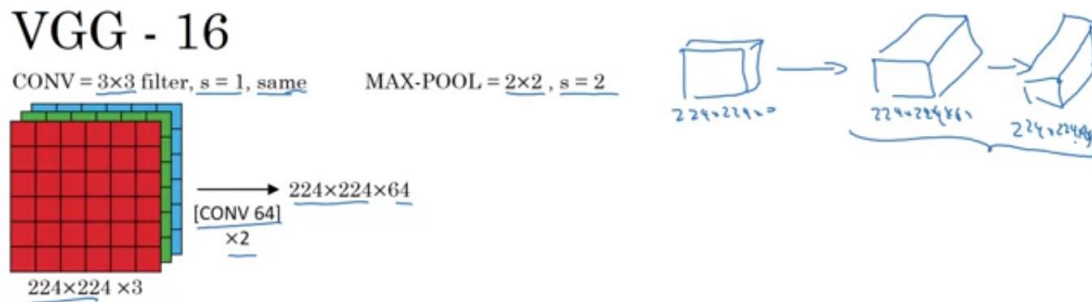


[Krizhevsky et al., 2012. ImageNet classification with deep convolutional neural networks]

Andrew Ng

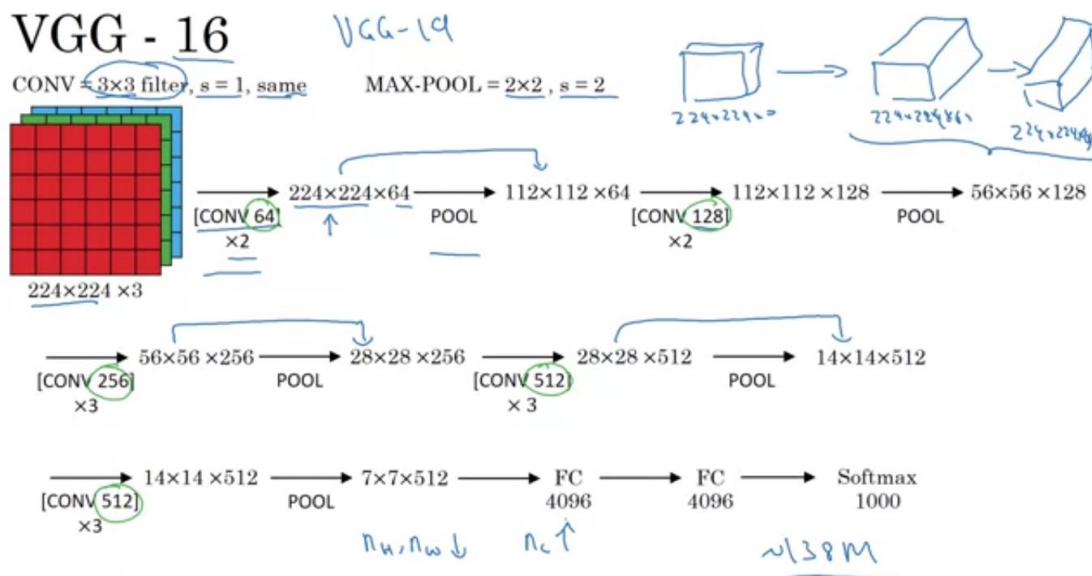
3) VGG-16:

It is simple (as it has uniform architecture) and has bigger size than the previous two networks (138 million parameter). It is called 16, as it has 16 layers that have same weights. The text (CONV 64 x2, means two convolutions simultaneously with 64 filters):



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

Andrew Ng

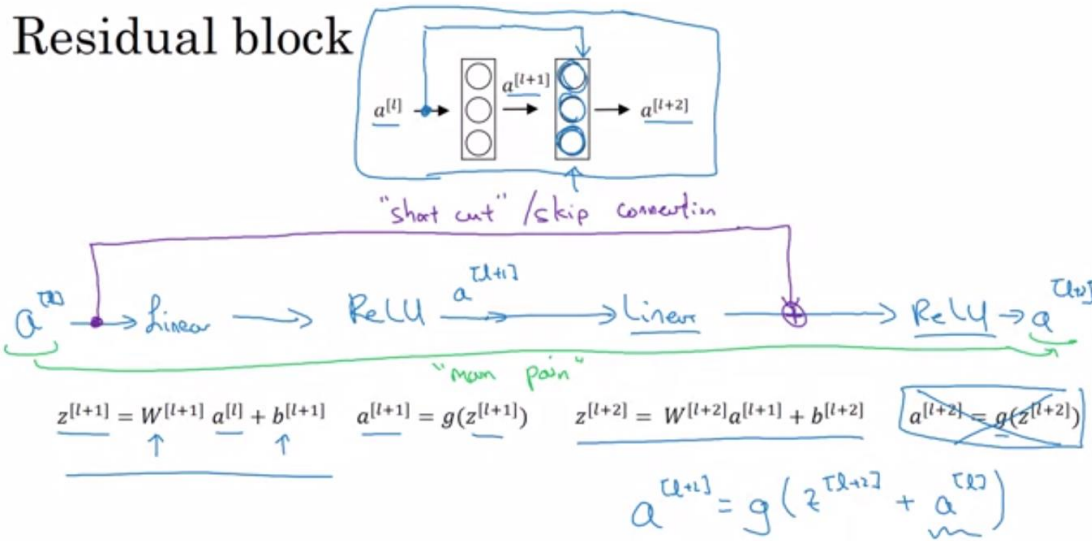


[Simonyan & Zisserman 2015. Very deep convolutional networks for large-scale image recognition]

Andrew Ng

Residual Networks (ResNets)

Used to train very deep NNs, as it repeats a residual block:



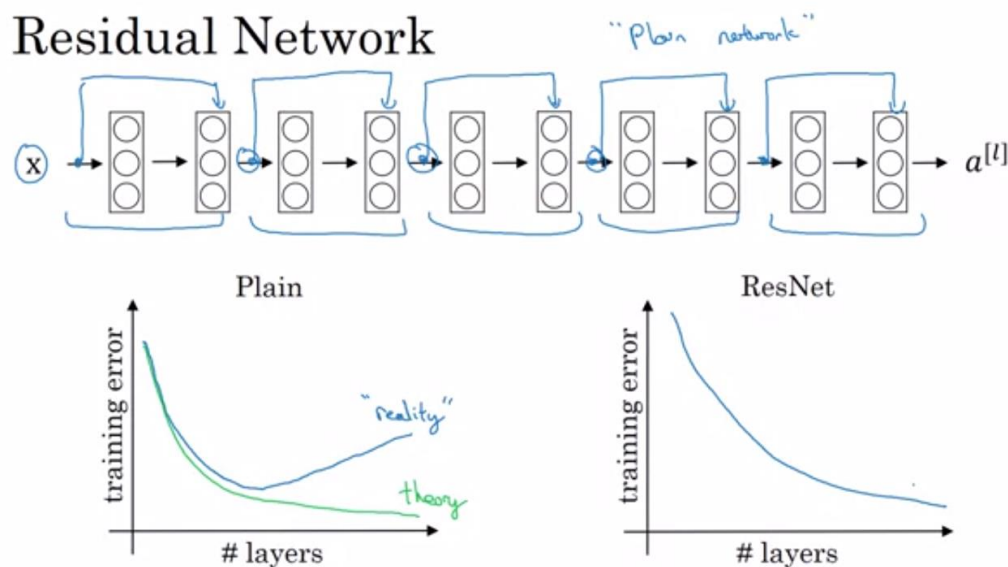
[He et al., 2015. Deep residual networks for image recognition]

Andrew Ng

So, rather than doing the last activation function as before, we just modify it to the above figure.

It is called skip connection, as it passes through multi layers. The short cut is added after linear and before nonlinear (Relu).

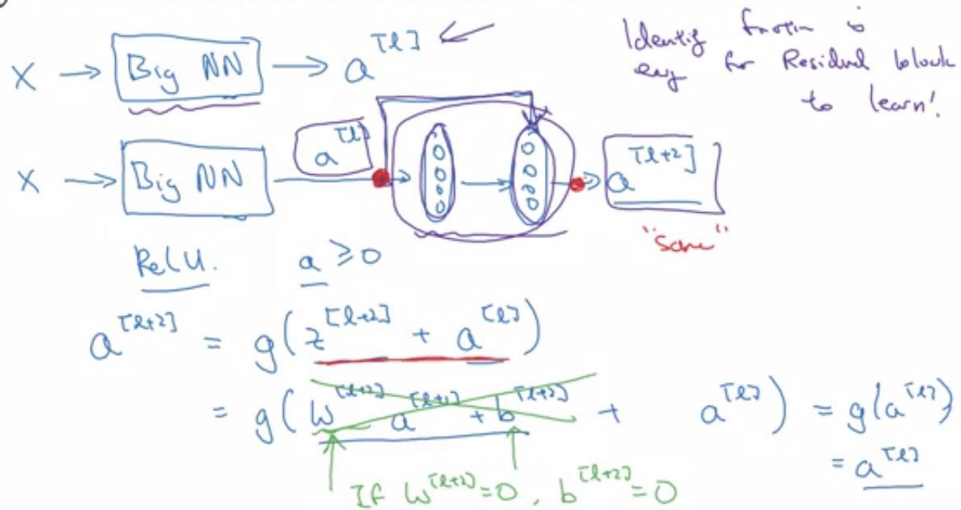
It is more practical than ordinary plain NN in training very deep NN, as gradient descent make errors after long time on large layers:



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

Andrew Ng

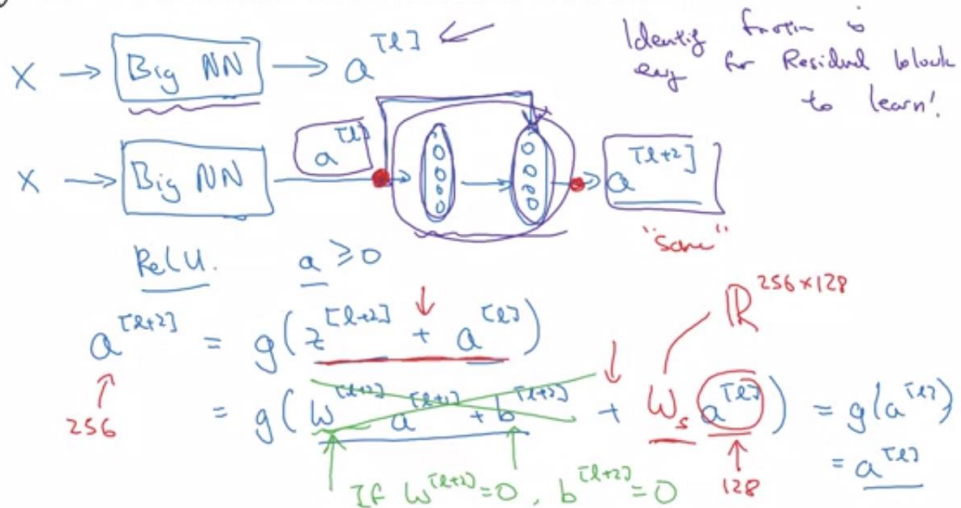
Why do residual networks work?



Andrew Ng

If $a^{[l]}$ and $z^{[l+2]}$ have different dimensions, we add W_s or zero padding $a^{[l]}$:

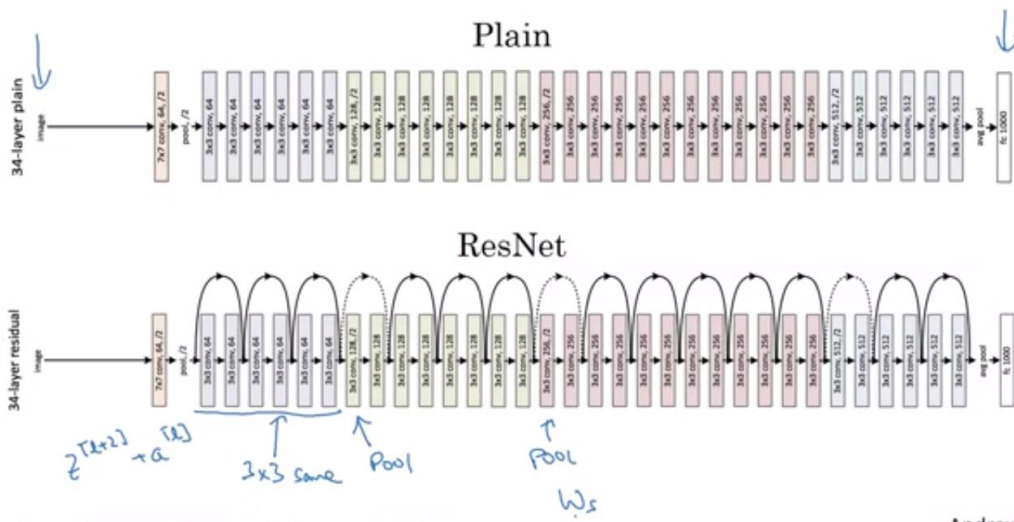
Why do residual networks work?



Andrew Ng

Example:

ResNet



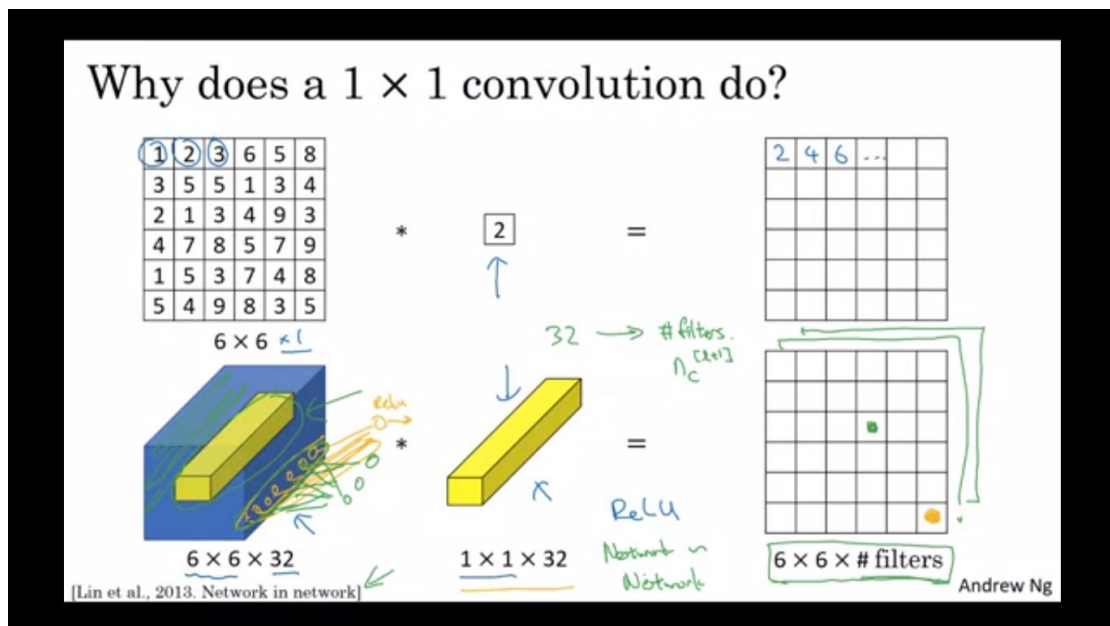
[He et al., 2015. Deep residual networks for image recognition]

Andrew Ng

With making W_s adjustment to pooling layers.

1x1 Convolution:

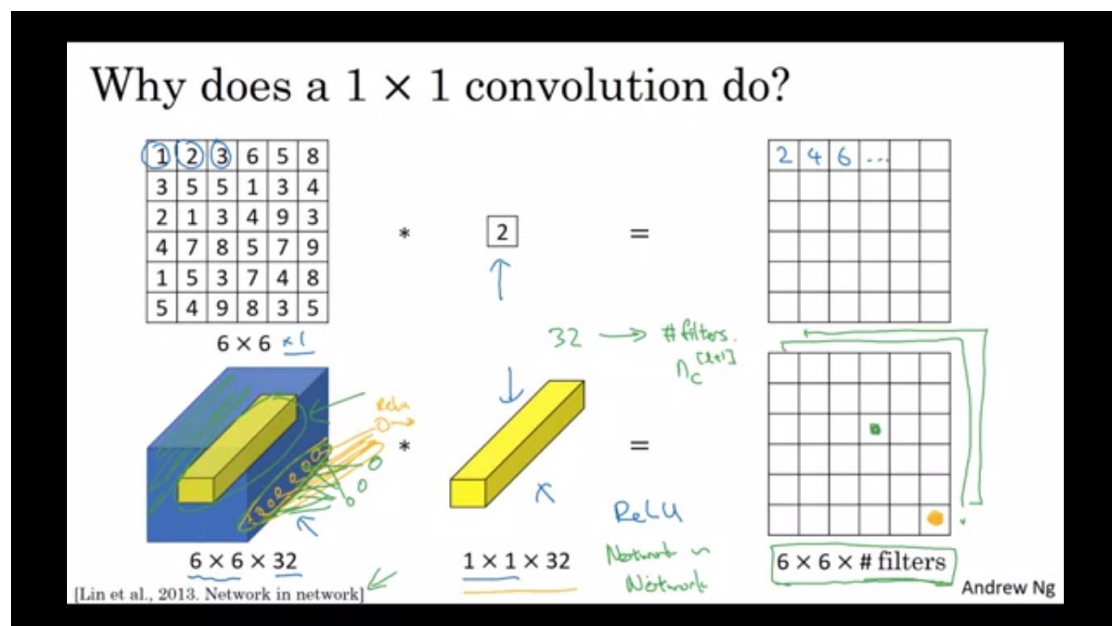
Single filter



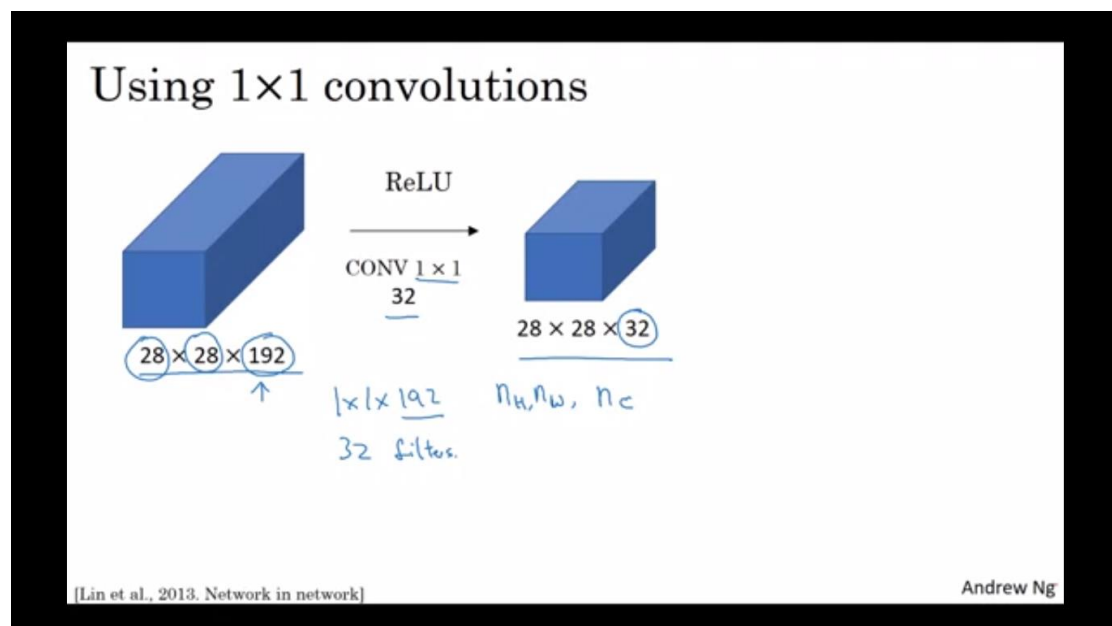
Andrew Ng

Network in Network (1x1 Convolution):

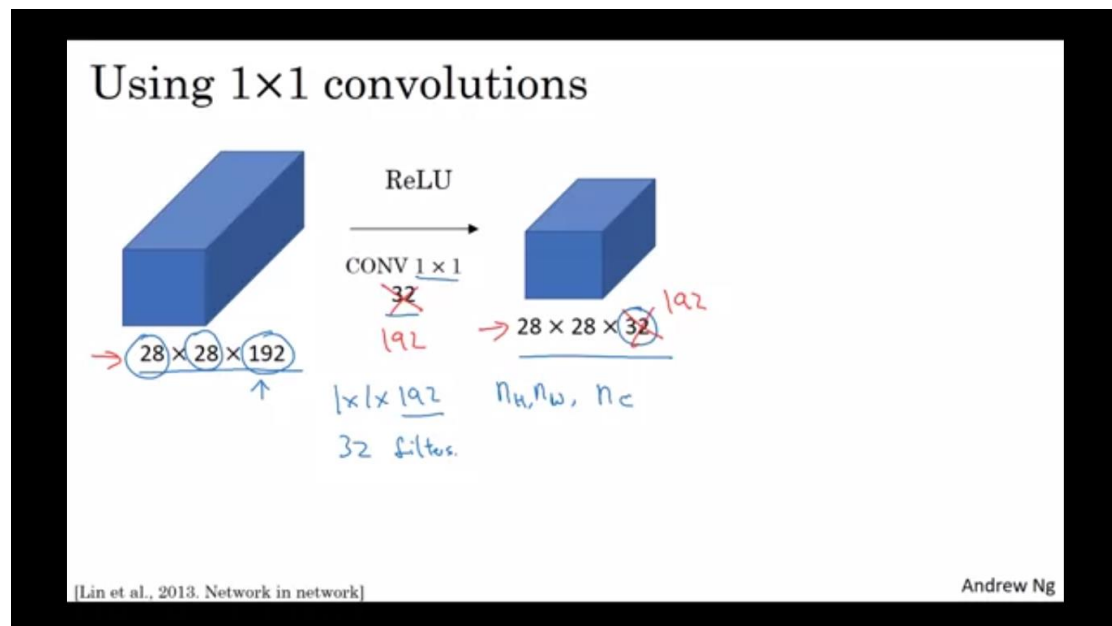
Multi filters



Useful in shrinking number of channels at the output:



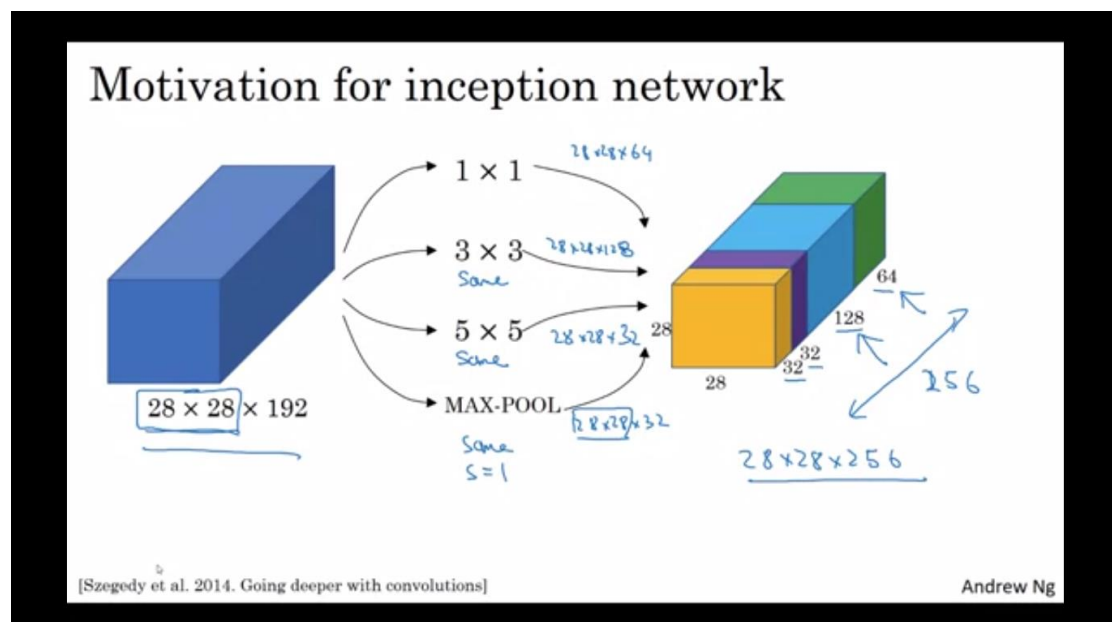
We can also keep the number of channels by making number of filter same as the input:



Or it can be used to even increase the number of channels.

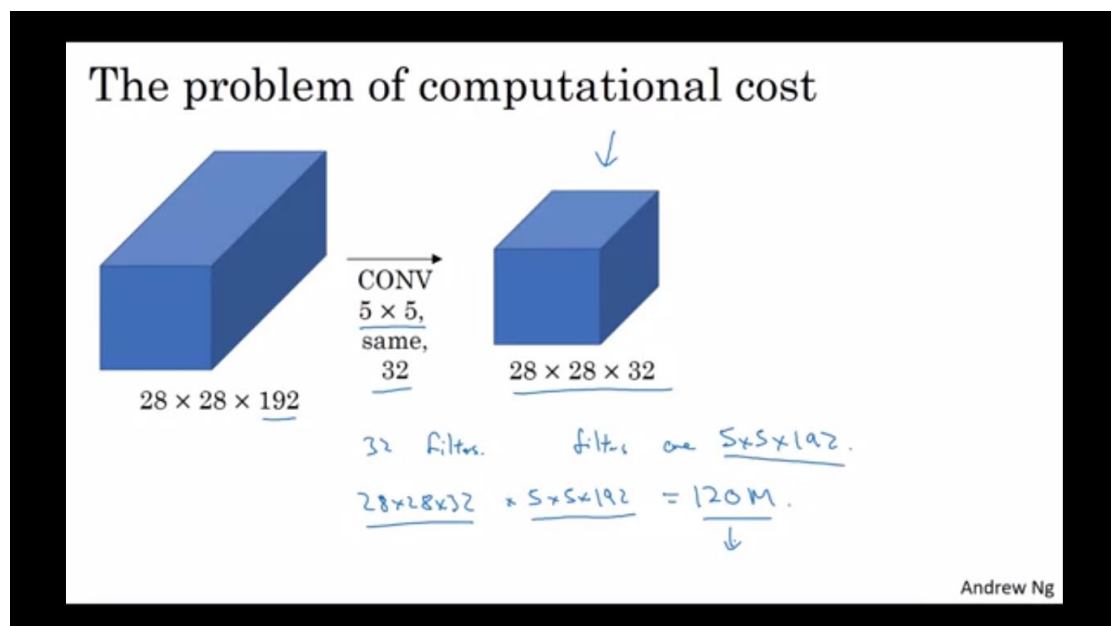
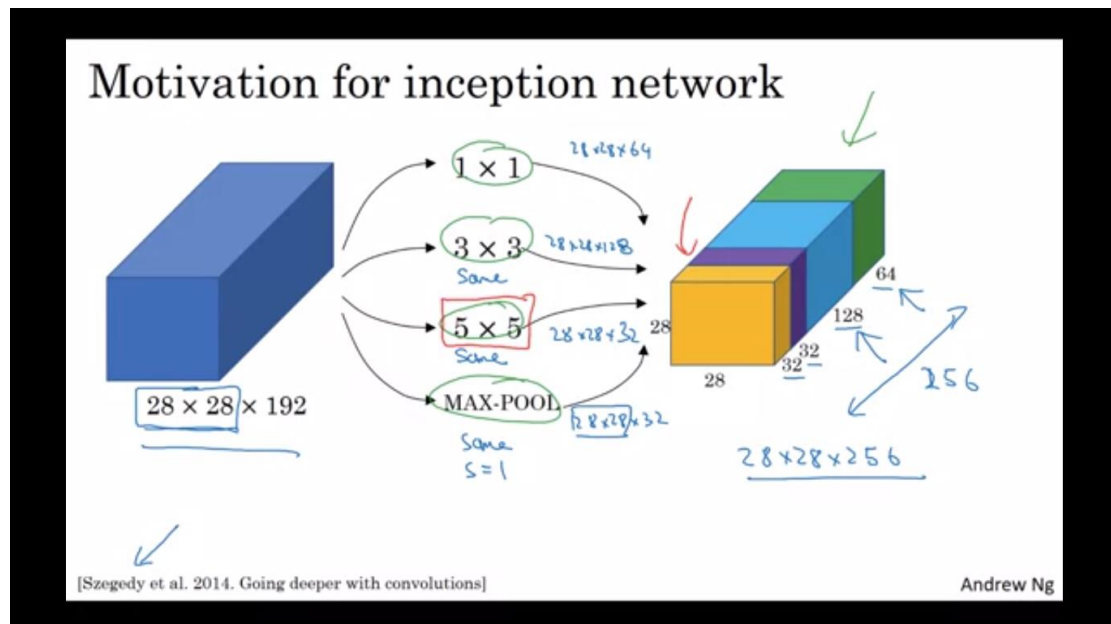
Inception Network

It is a concatenating multi types of convolutions and keep the same n_H and n_W dimensions:



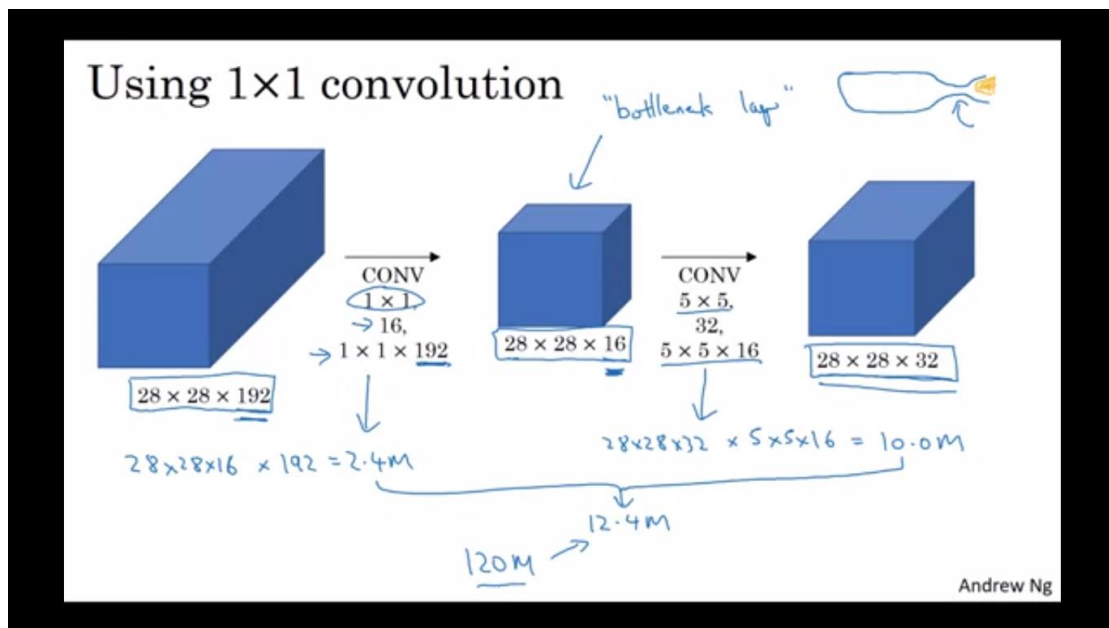
Computational cost:

For example, for the 5x5 filter:



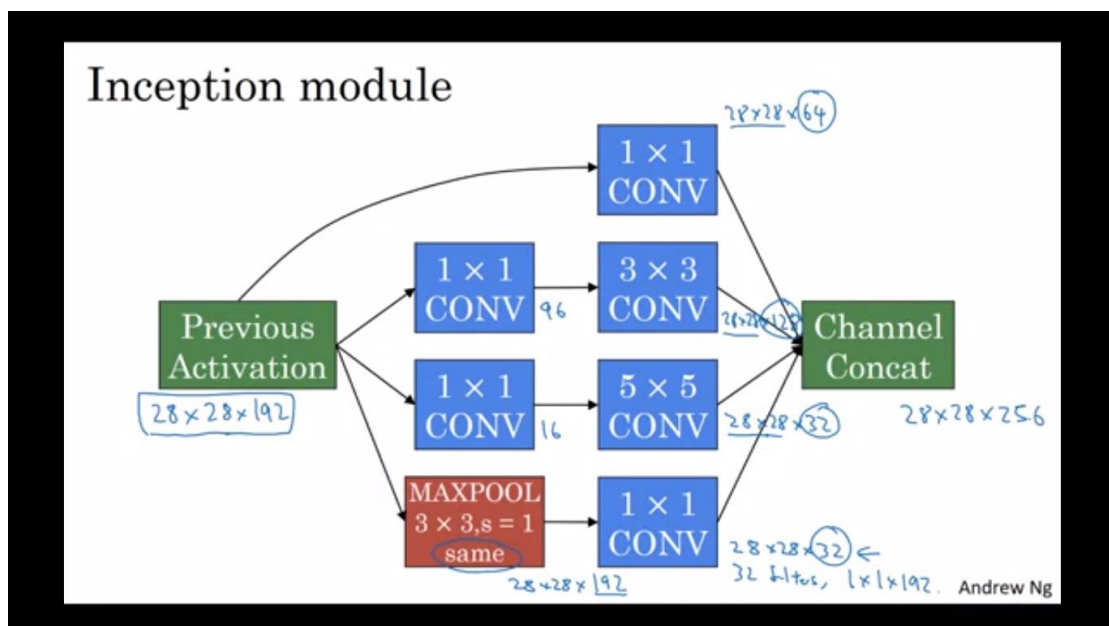
The cost is the multiplication of the number wanted to determine each of the $28 \times 28 \times 32$ times the $28 \times 28 \times 32$ which leads to 120million.

This can be reduced by 1x1 conv:

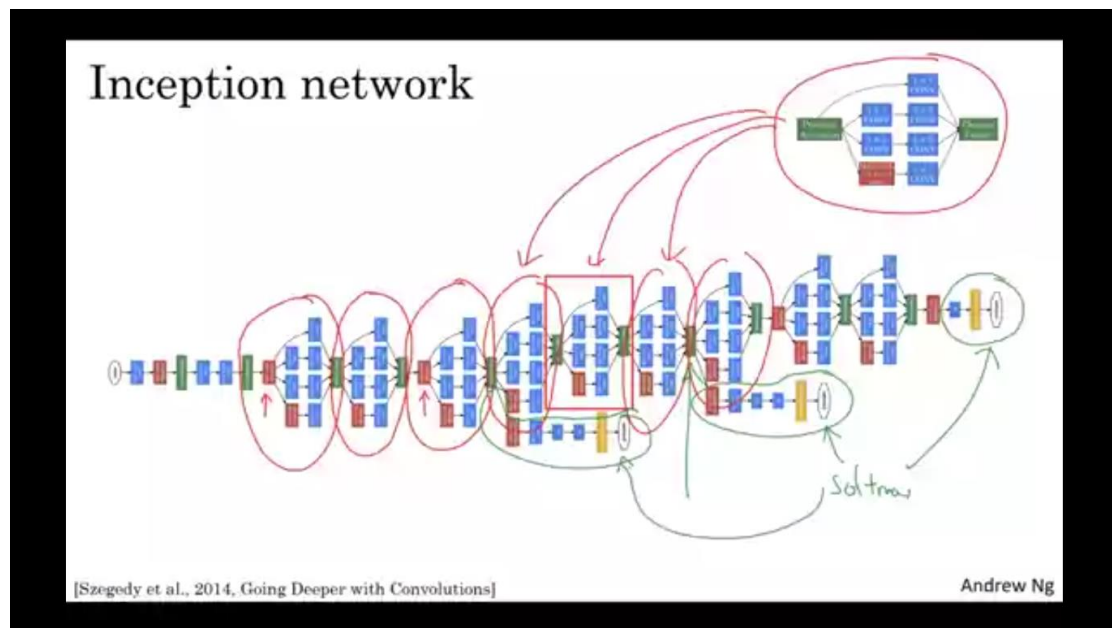


It is also called (bottleneck) and it doesn't hurt the NN.

Inception module:



Inception Network (gooLeNet):



It is several Inception modules. There are numbers of softmax layers taken from different hidden layers to make prediction and prevent overfitting.

Note: inception name comes from (we need to go deeper).

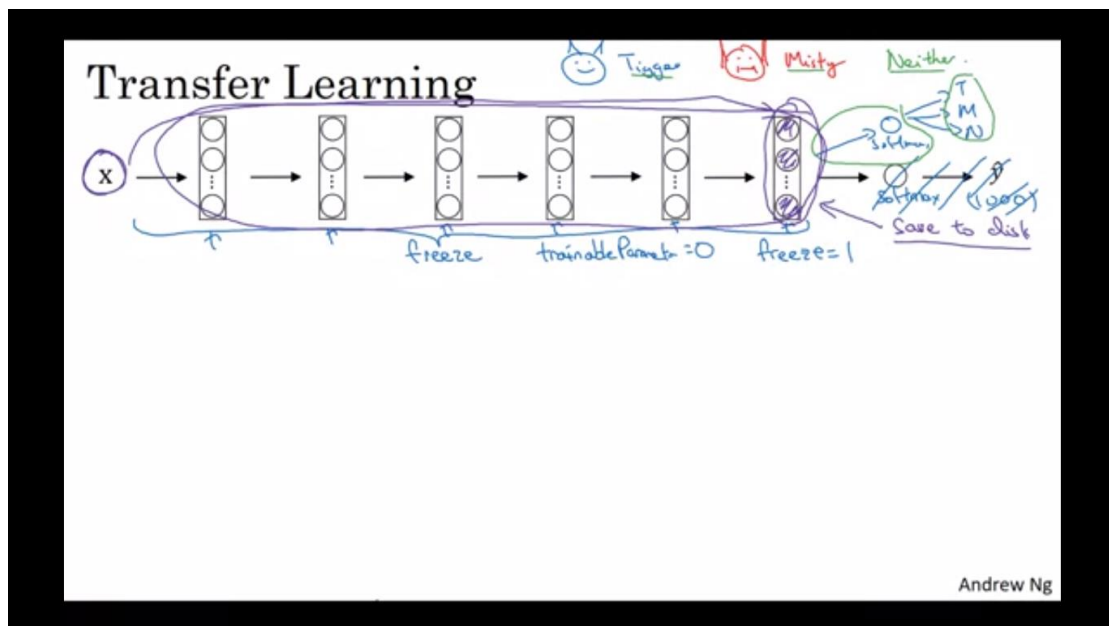
Practical advices for using ConvNets

1- Using open-source implementations

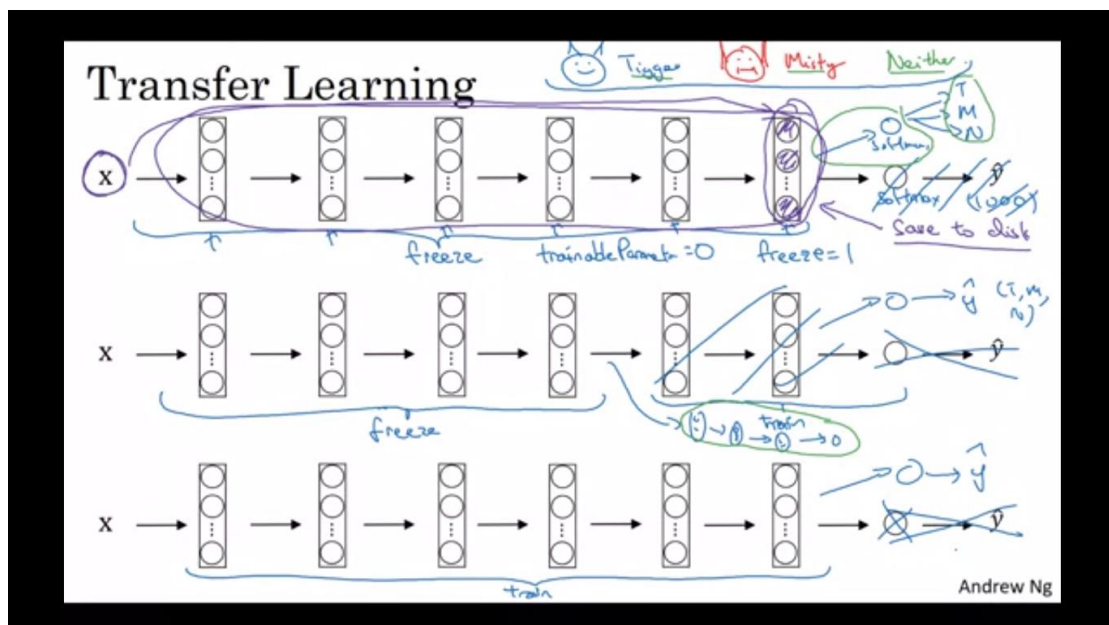
Like using git-hub.

2- Transfer Learning

For small training set, for example, freeze the hidden layers parameters like weights and compute the activation function firstly then save them to disk. Make your softmax layer and use the hidden layers and activation functions as they are:

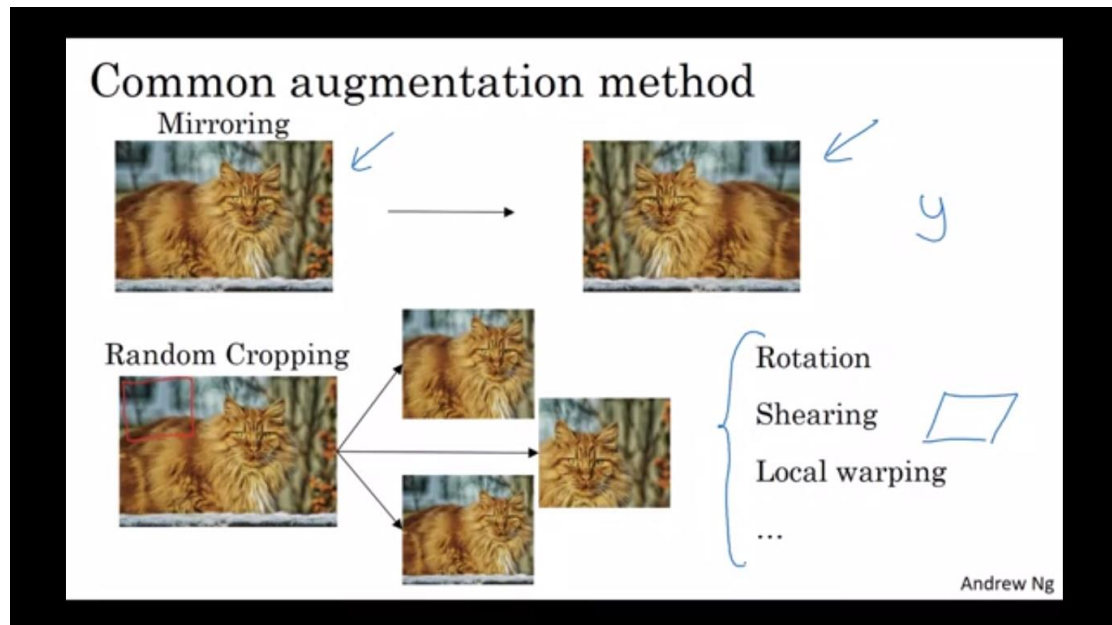


If we have large number of training set, freeze a smaller number of layers. The rest layers, train them or replace them with our new layers. If the train set is very big, we must train the full NN (no freezing) and apply gradient descent for training:

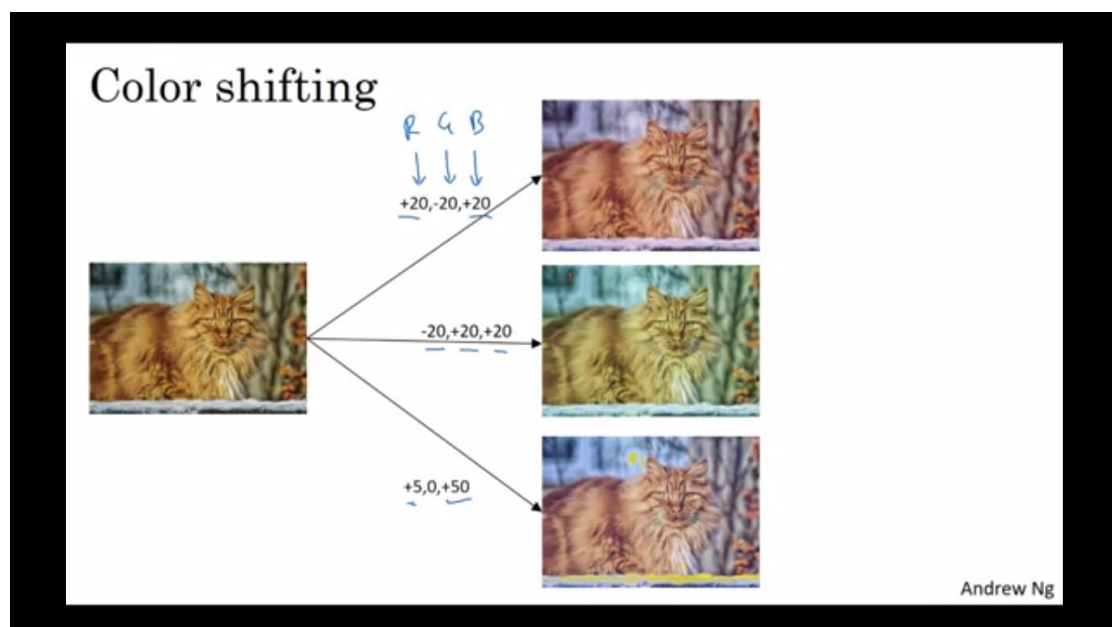


3- Data Augmentation

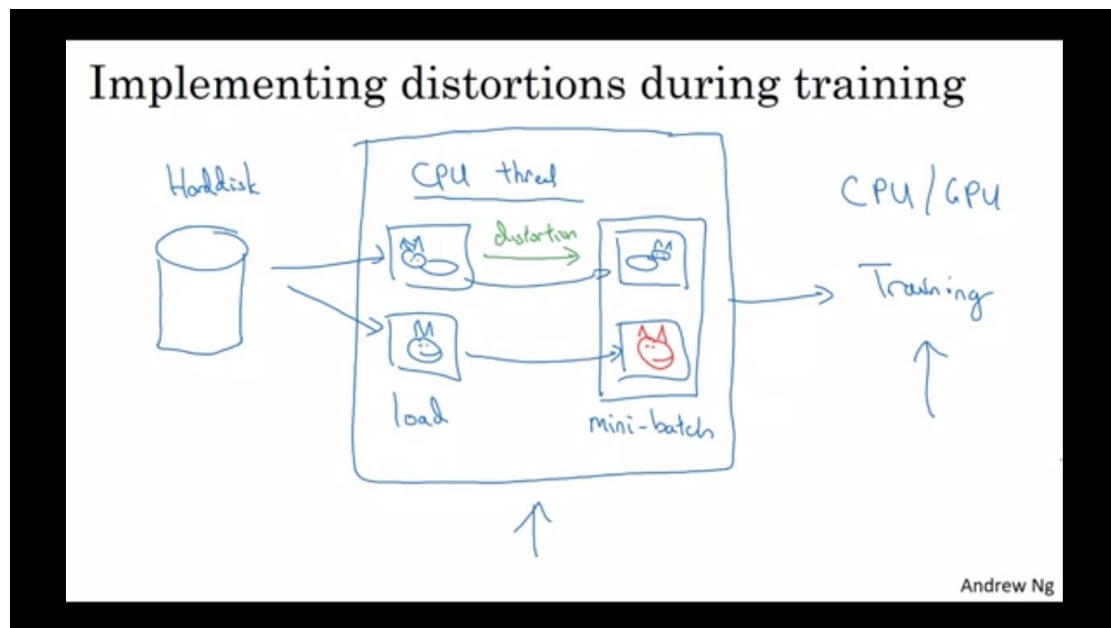
It is used to be sure that NN works well in many data sets. Most common types of augmentation are mirroring and random cropping. The rest in the figure are less used due to their complexity:



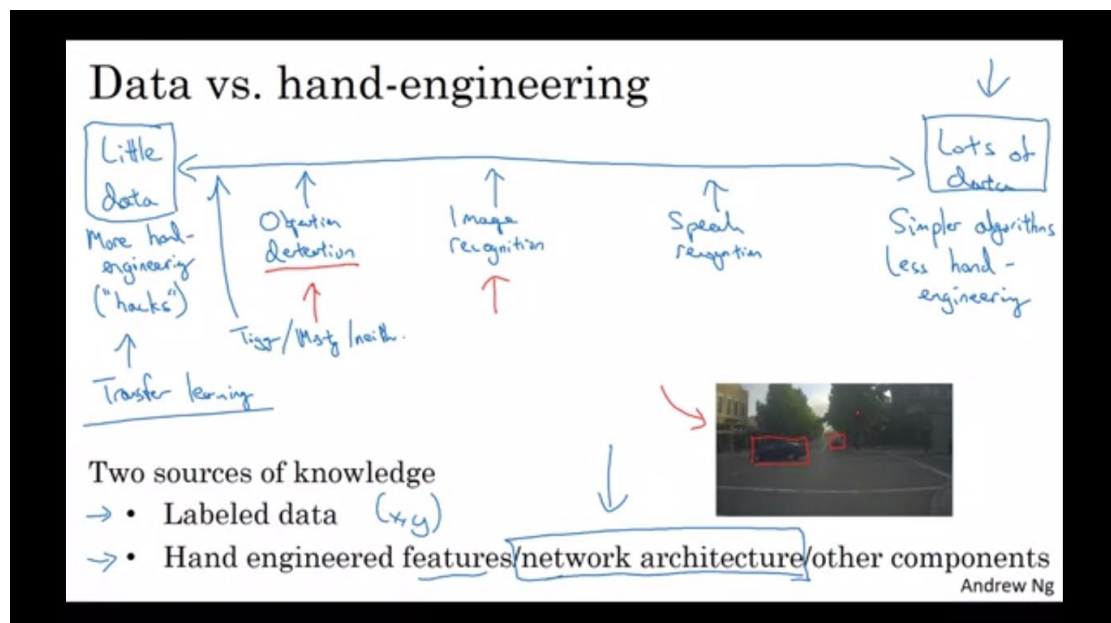
Another data augmentation is color shifting, which make NN robust to color changes:



In implementation, we can make augmentation and training in parallel:



4- Relation between data amount and hand engineering (NN Arch. and algorithms) is reciprocal. Transfer learning helps a lot in less data amount:



- 5- Common tips for making benchmarks: test on many NNs and take the average (Ensembling) with disadvantage of large memory needed for NNs. 10-Crops is another way which uses data augmentation and take the average:

Tips for doing well on benchmarks/winning competitions

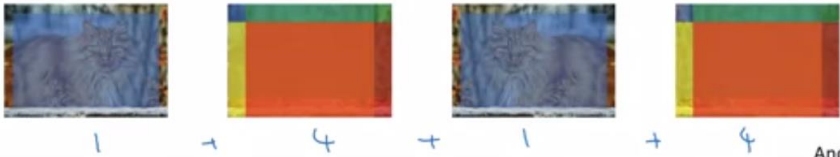
Ensembling 3-15 networks → \hat{y}

- 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

Summary:

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

