Ind Study W3

Content

- CNN architecture
  - Alexnet
  - VGG
  - ResNet
  - GoogLeNet
- Transfer Learning
- Data Augmentation
- Multi-crop

Alexnet

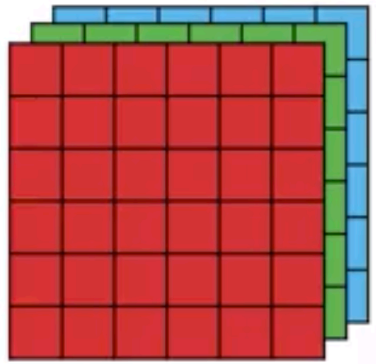
- Data Augmentation
  - Horizontal reflection
  - Altering the intensities of RGB channels in training images (Perform PCA)
- Dropout
  - Helps in coadoption of neurons in absence of some other neuron
- Contains local response normalization

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]

VGG

- Address depth of CNN(Fix all other parameters and increase depth)
- Preprocessing- Subtracting the mean RGB value
- Architechture
  - Filters-3*3 and 1*1, stride-1
  - Same Padding -1 for 3*3 filter
  - Followed by spatial + max pooling (f=2,s=2) for some layers
  - Followed by 3 fully connected layers (same configuration in all network)
  - First two FC 4096 channels each , third 1000 channels
  - Final layer is softmax
  - Contains RELU but not local response normalization
- Having 3*3 instead of 7*7:
  - Decreases number of parameters $27C^2$ instead of $49C^2$
  - Incorporate 3 non-linear rectification function instead of one, making decision function more discriminative
  - Imposing regularization

VGG-Training

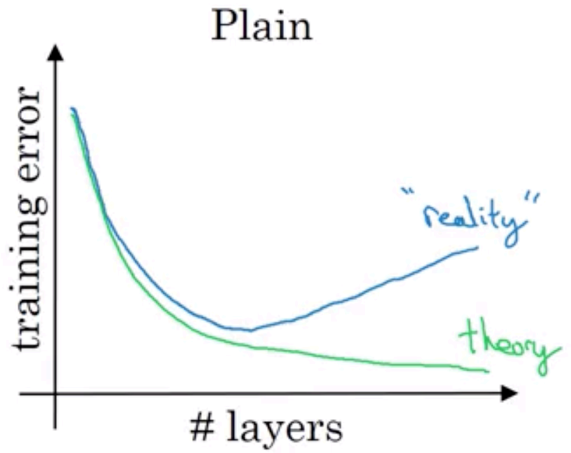
- Multinomial logistic regression with mini-batch gradient descent with momentum
- Batch size-256, momentum -0.9
- Weight decay and regularization (dropout ratio=0.5)
- Learning rate- 10^{-2} , 370K iterations (74 epochs)
- Required less epochs due to
  - Implicit regularization imposed by greater depth and smaller conv. Filter sizes
  - Pre-initialization of certain layers
- Setting Training Image Size S: Rescaling training image by randomly sampling S from a certain range $[S_{min},S_{max}]$

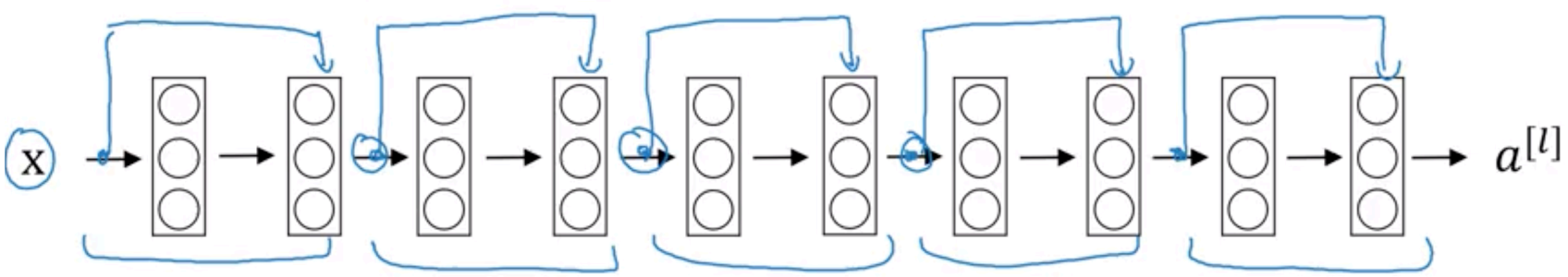
VGG- Experiments

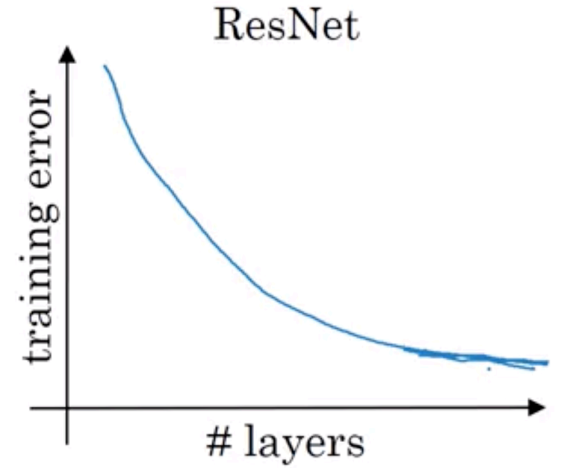
- Single scale Evaluation
  - A deep net with small filters outperforms a shallow filters with larger filters
  - Scale jittering at training time ($S \in [256,512]$) leads to significantly better results than training on images with fixed smallest side (S= 256 or S=384), even though a single scale is used at test time
- Multi-scale Evaluation
  - Running a model over several rescaled version of test image

ResNet

- Skipping layers to pass over information
- $a^{[l]}$ -> Linear->Relu->Linear->Relu->$a^{[l+2]}$
- $a^{[l+2]} = g(z^{[l+2]} + a^{[l]})$; $a^{[l]}$ passed to relu along with $z^{[l+2]}$
- On plain network the training error tends to decrease first with increasing depth but increases further
- In ResNet error always decreases with depth

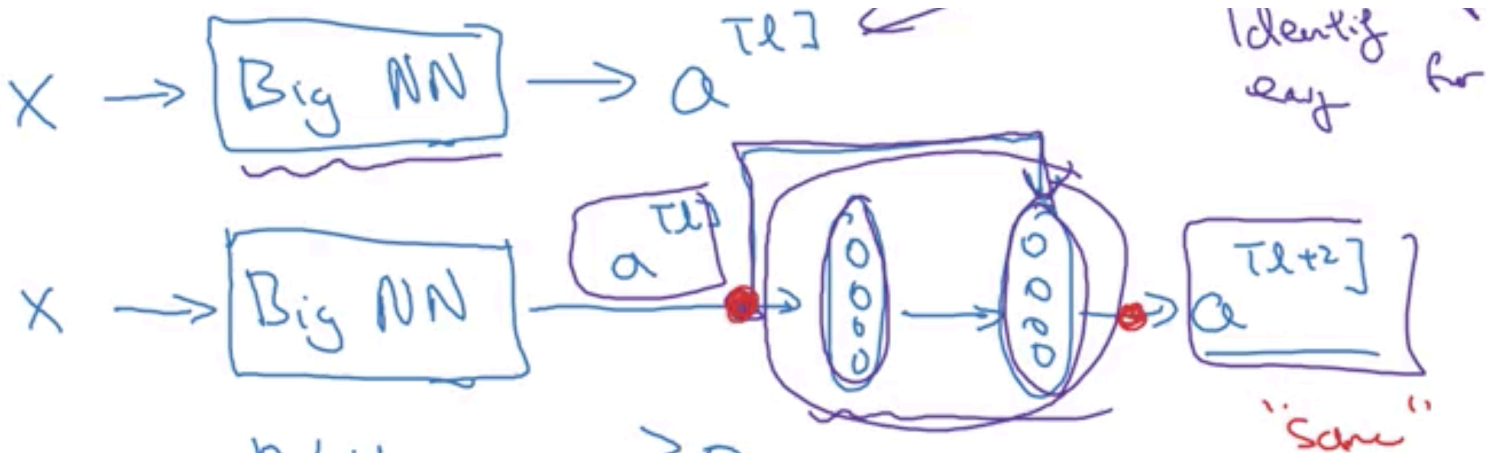






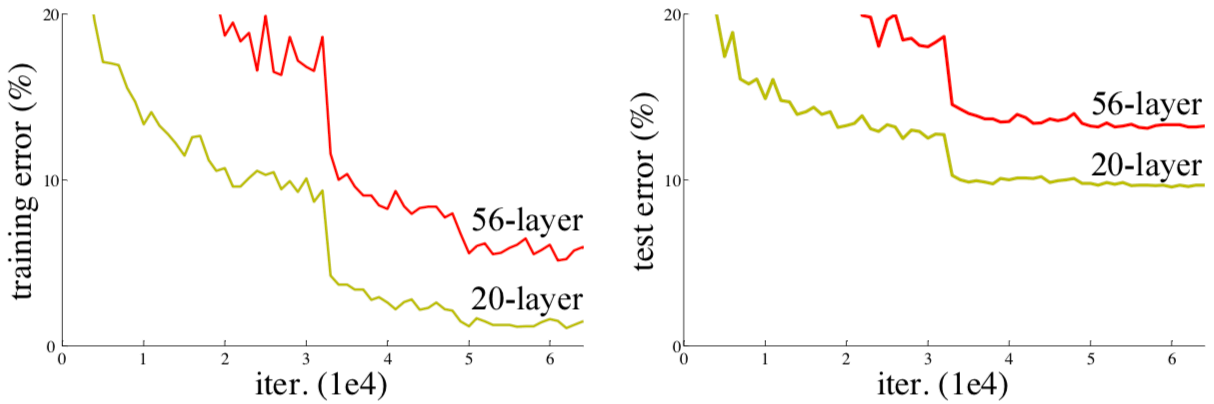
ResNet

- $a^{[l+2]} = g(z^{[l+2]} + a^{[l]}) = a^{[l+2]} = g(w^{[l+2]} . a^{[l+1]} + b^{[l+2]} + a^{[l]})$
- $a^{[l+2]} = g(a^{[l]})$; if $w^{[l+2]} = 0$, $b^{[l+2]} = 0$
- Use same convolution to have dimension of $z^{[l+2]}$ and $a^{[l]}$ same



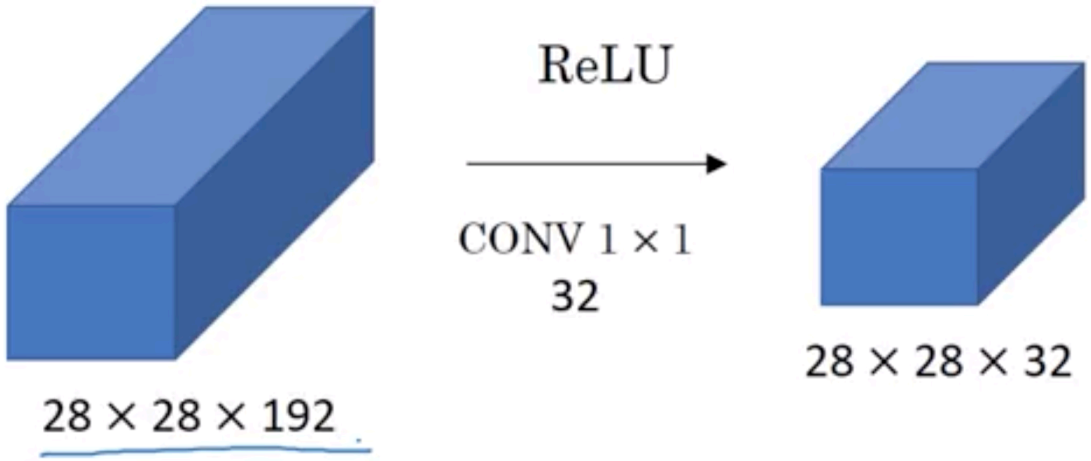
ResNet

- Degradation with deeper network does not happen by overfitting
- Adding more layers to a deep model leads to higher training error
- Shortcut connections act as identity mappings
- ResNet converges faster than plain for 18 layers



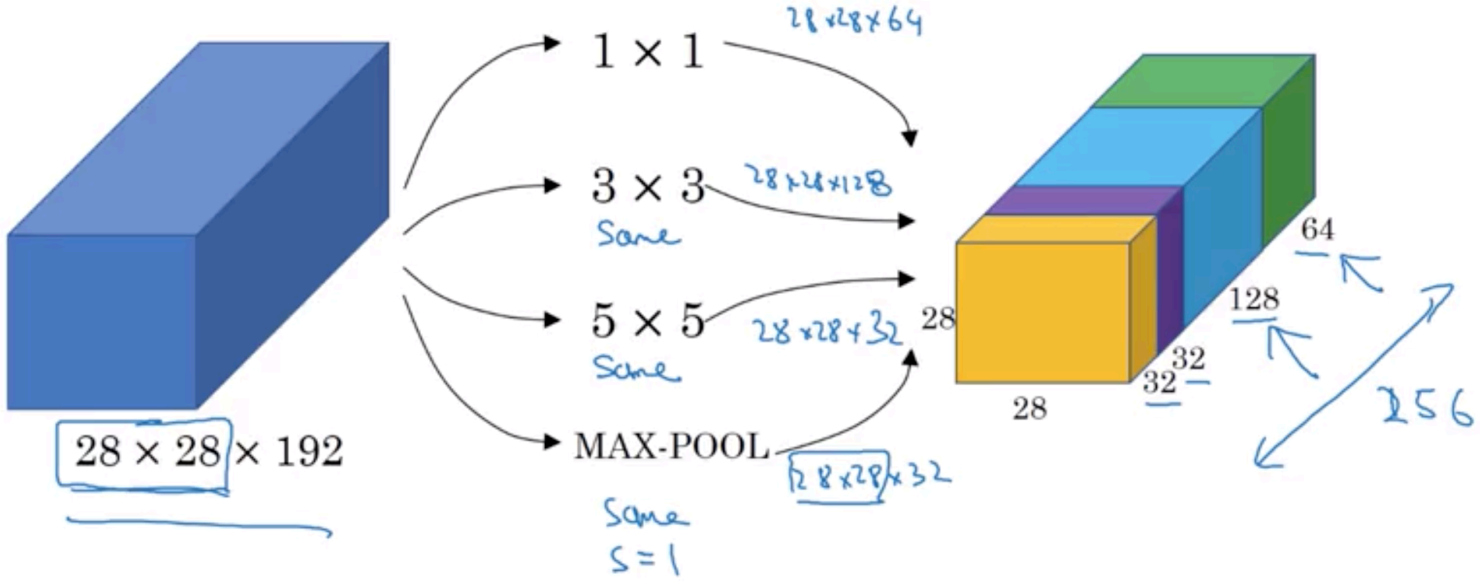
Network in Network (1*1 convolution)

- Multiply individual element with a number
- Helpful in shrinking number of channels and save computations



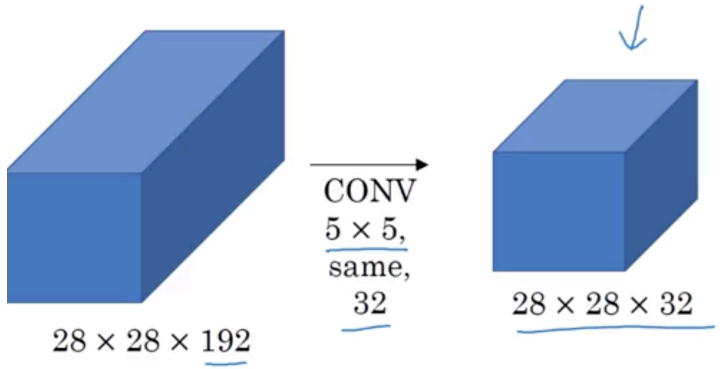
GoogleNet(Inception)

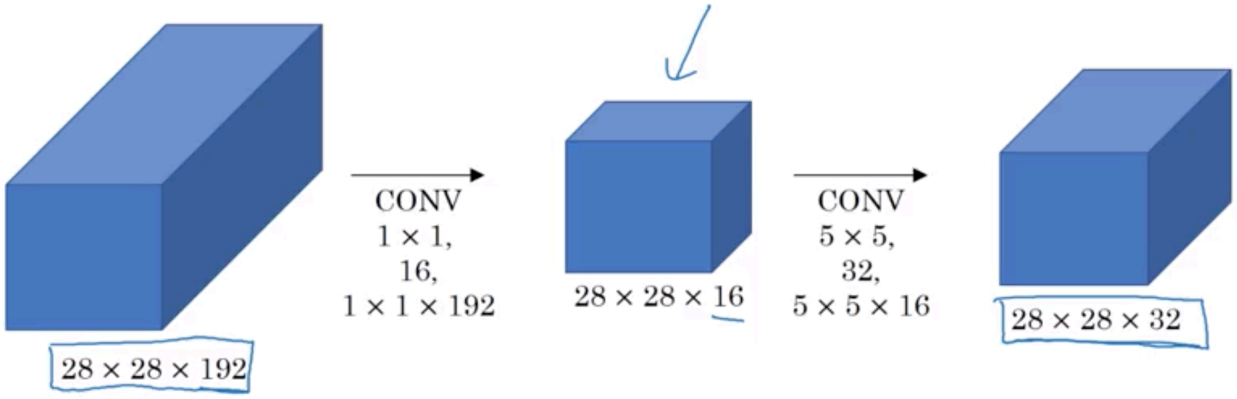
- Try all type of convolutions 1*1, 3*3, 5*5 and MAX-POOL
- Concatenate all the output and let network learn whatever parameters it want to use



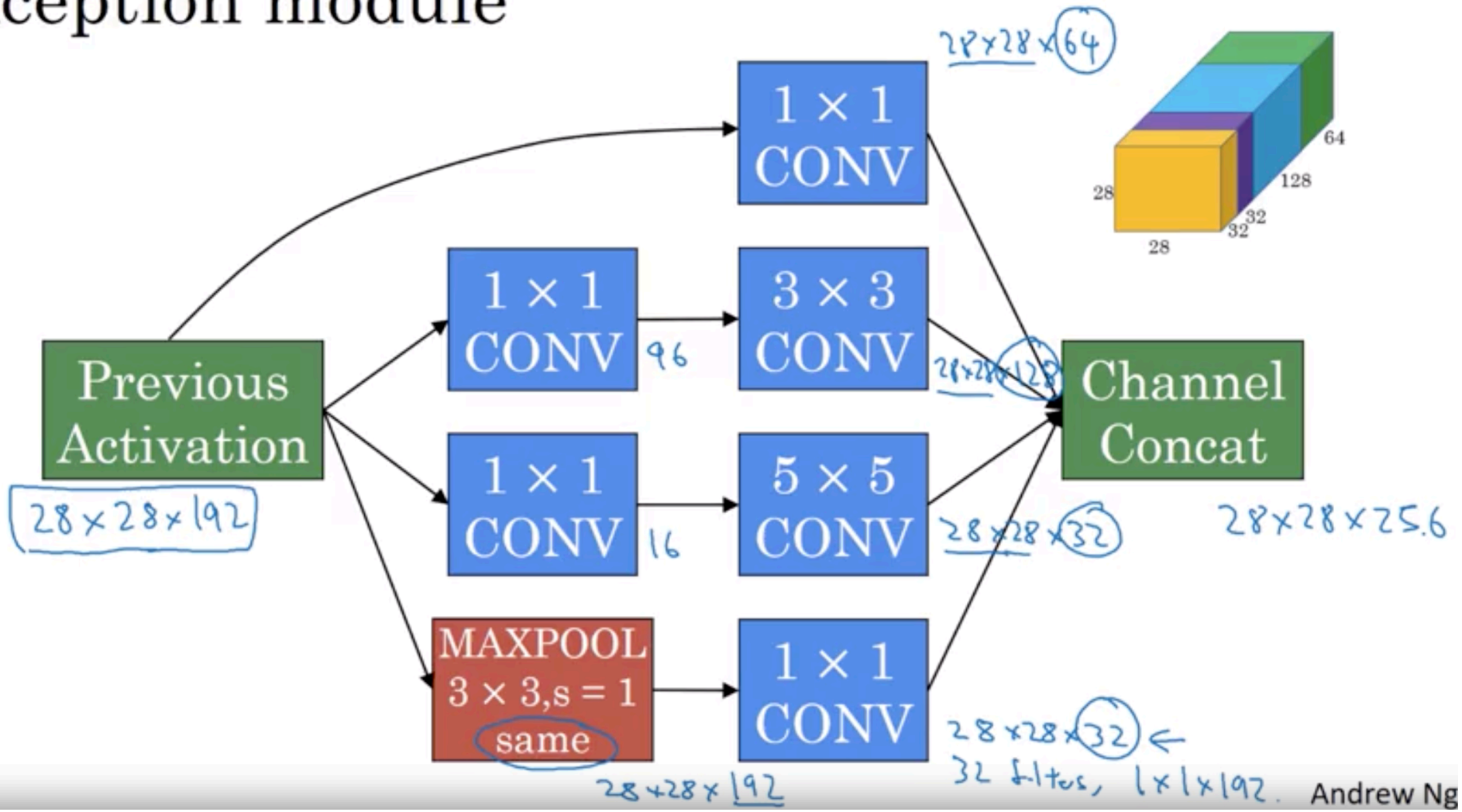
Inception

- Computation cost for 5*5 CONV:
  - 32 filters of 5*5*192
  - 28*28*32* 5*5*192= 120M (expensive)
  - Using 1*1 convolution reduces cost by 1/10th
  - Introduce a bottleneck layer (28*28*16) in between
  - New cost=28*28*16*1*1*192 + 28*28*32*5*5*16= 12.4 M

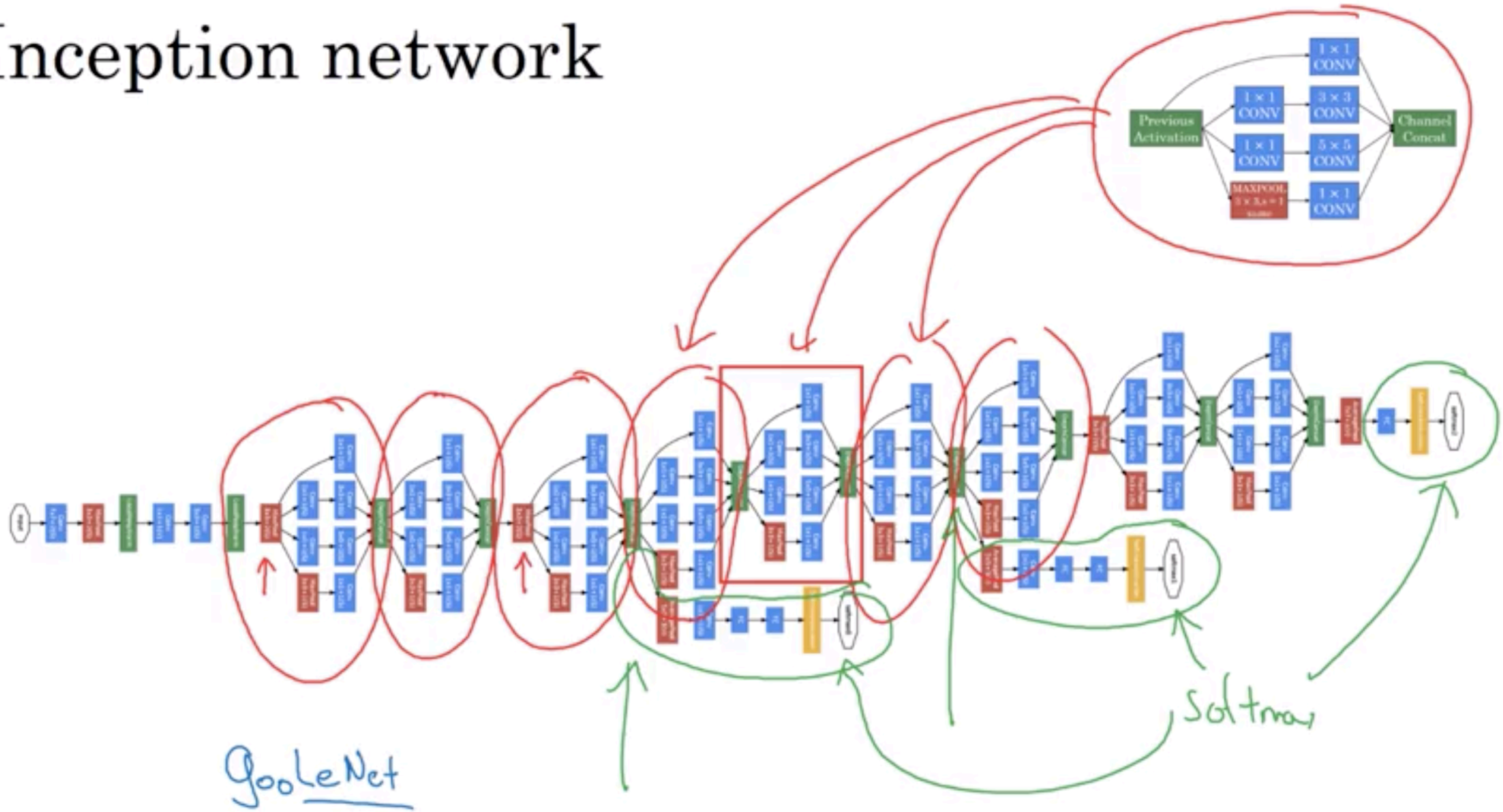




Inception module



Inception network



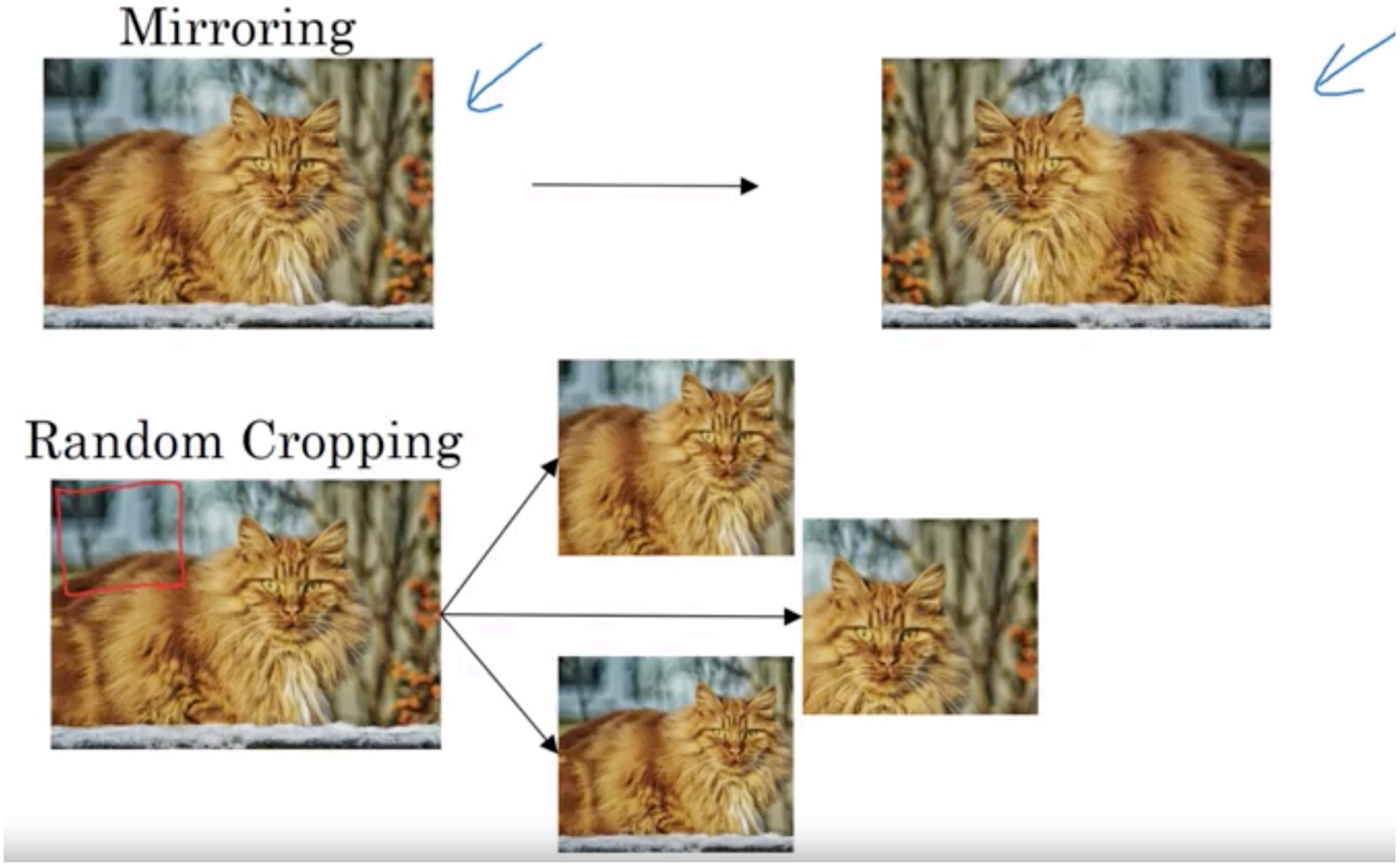
Transfer Learning

- Reusing an existing model as starting point for other model
- It saves time and computation power which is required to build from scratch.

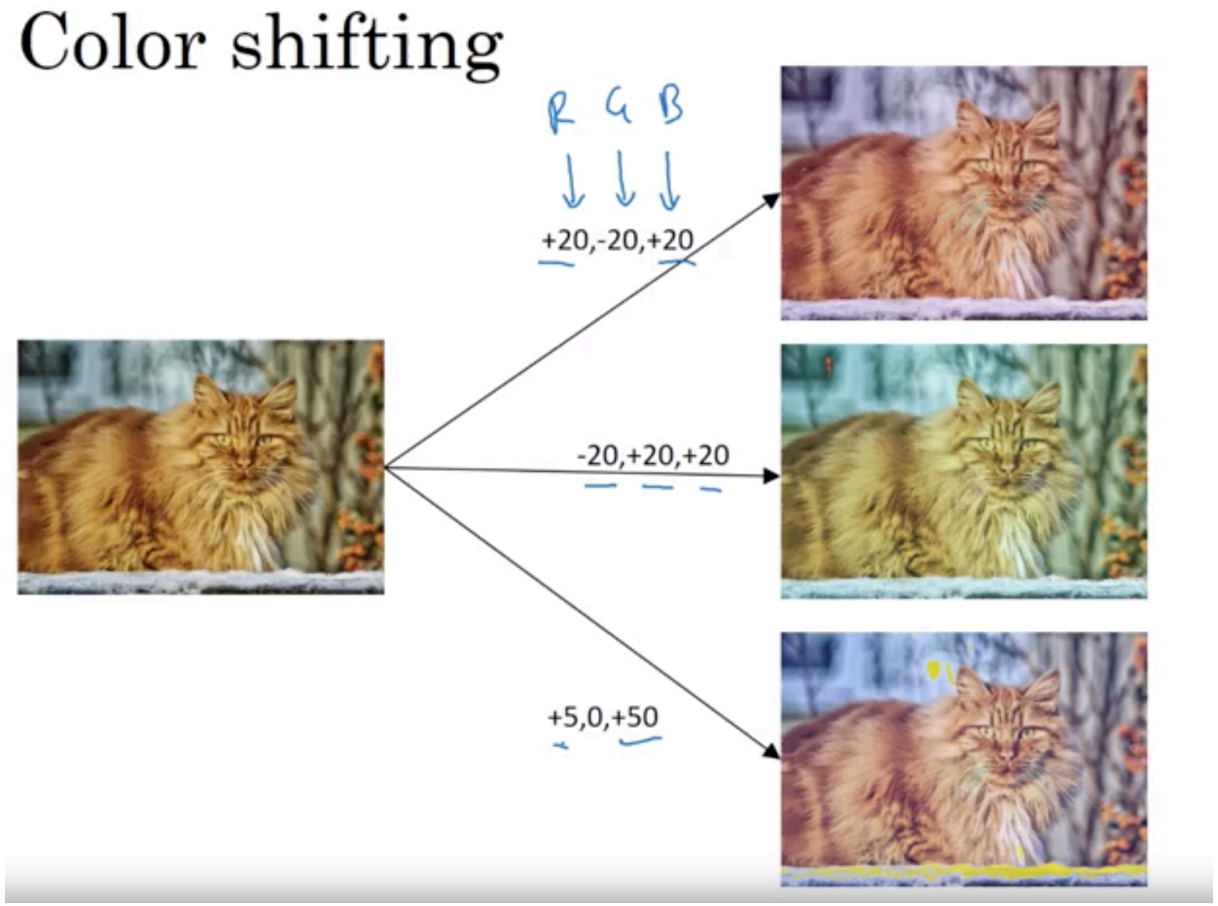
Data Augmentation

- Mirroring
- Random Cropping
- Rotation
- Shearing
- Local Warping
- Color shifting- take different values of R, G, and B and use them to distort the color channels, bring more robustness with colors
- PCA color augmentation- if image is purple it mainly contains red and blue but lesser green, so it adds and subtracts more of RG and less of G

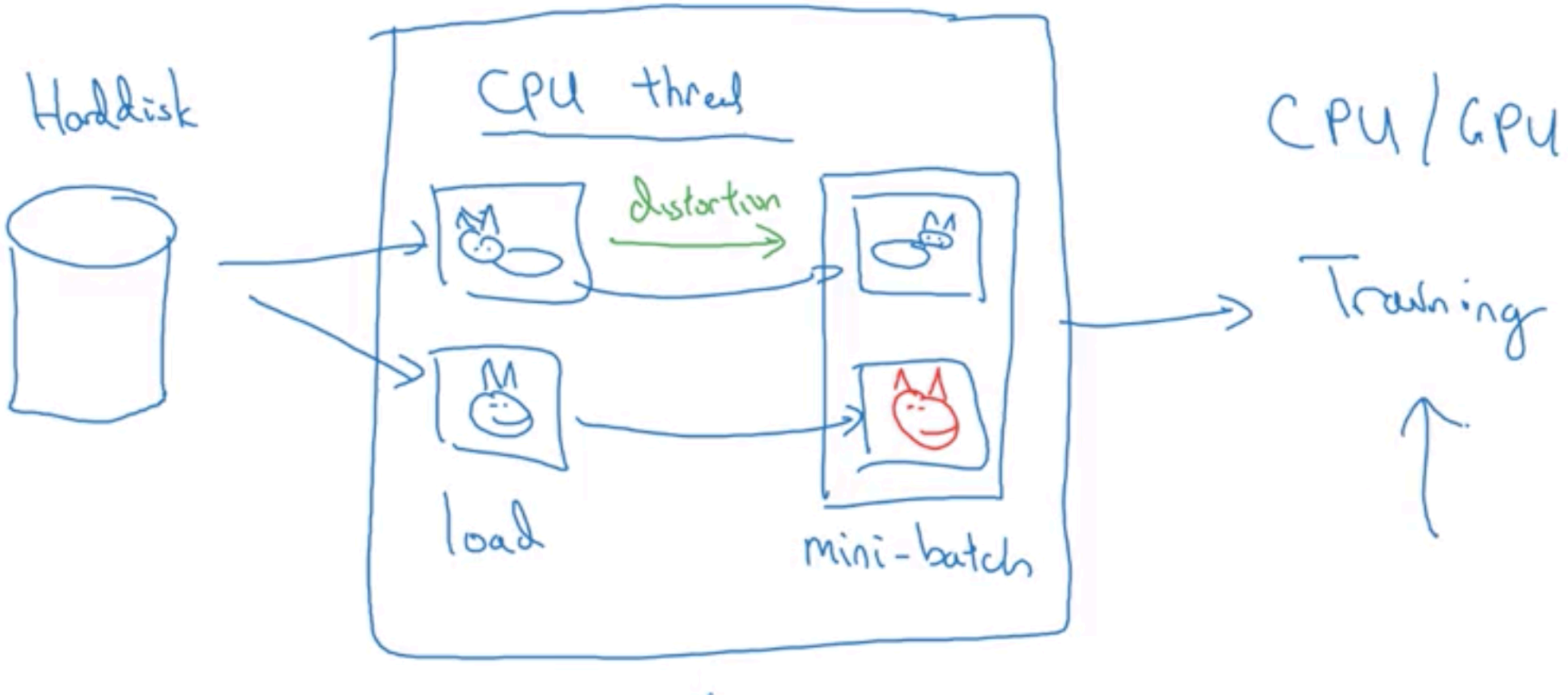
Common Augmentation method



Common Augmentation method

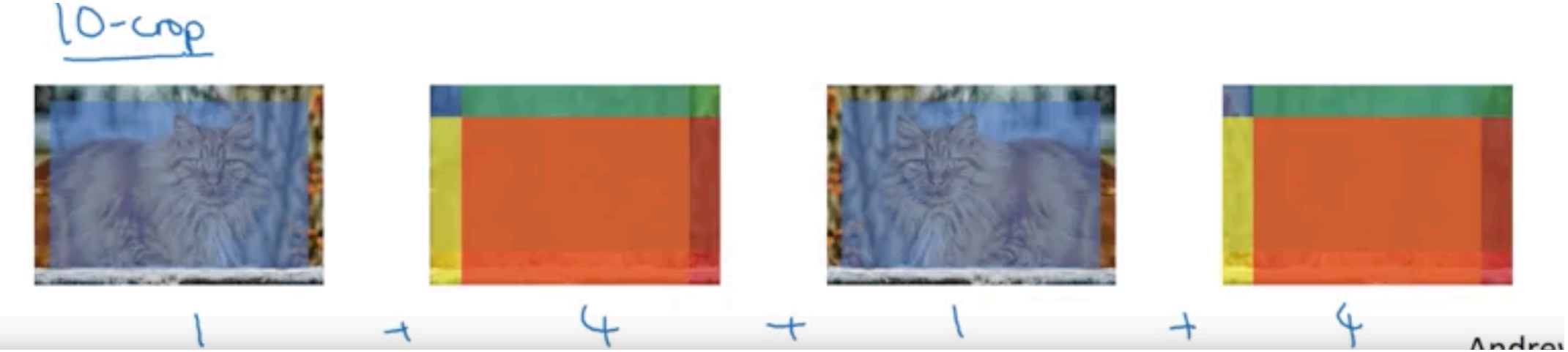


Implementing distortions during training



Multi crop at test time

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



Quiz

✖ 0 / 1 point

3. In order to be able to build very deep networks, we usually only use pooling layers to downsize the height/width of the activation volumes while convolutions are used with “valid” padding. Otherwise, we would downsize the input of the model too quickly.

✖ 0 / 1 point

6. Which ones of the following statements on Residual Networks are true? (Check all that apply.)

✖ 0 / 1 point

9. Which ones of the following statements on Inception Networks are true? (Check all that apply.)

✖ 0 / 1 point

10. Which of the following are common reasons for using open-source implementations of ConvNets (both the model and/or weights)? Check all that apply.

References

- CUDA implementation of CNN: https://code.google.com/archive/p/cuda-convnet/
- Keras documentation: https://keras.io/models/model/
- ResNet Blog: http://torch.ch/blog/2016/02/04/resnets.html
- ResNet Github: https://github.com/KaimingHe/deep-residual-networks
- ResNet original paper: https://arxiv.org/abs/1512.03385
- Identity mapping in ResNet: https://arxiv.org/abs/1603.05027