

CSPB 3202 Artificial Intelligence

2020 Spring

Convolutional Neural Networks (2)

Geena Kim

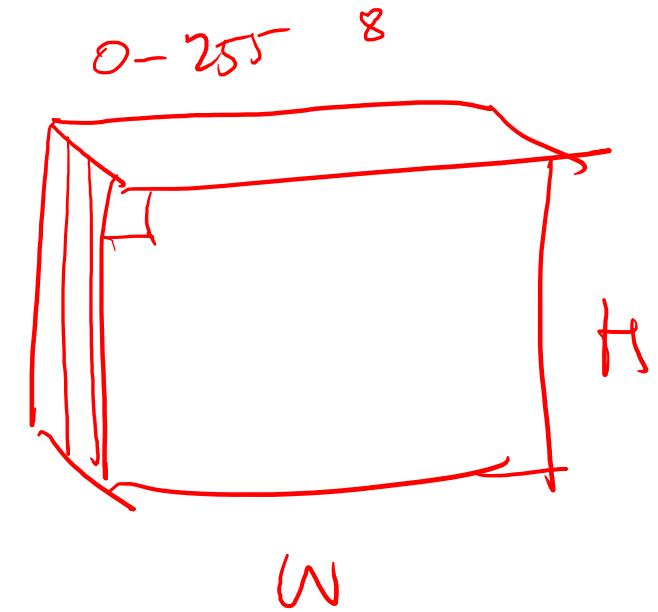


University of Colorado Boulder

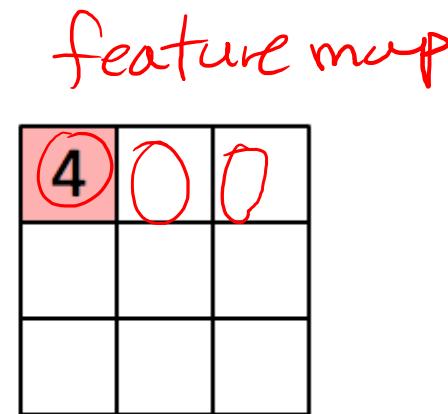
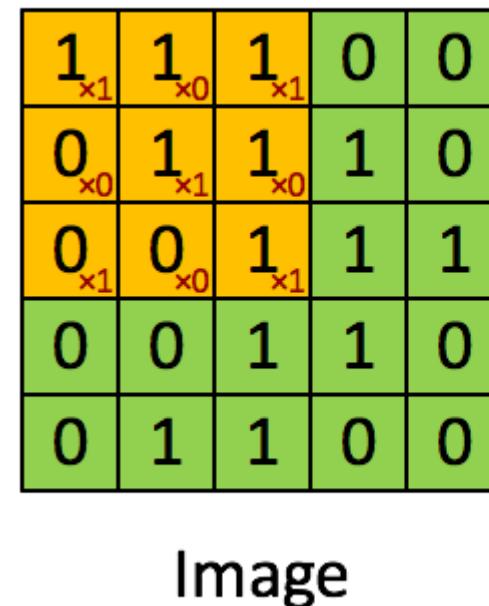
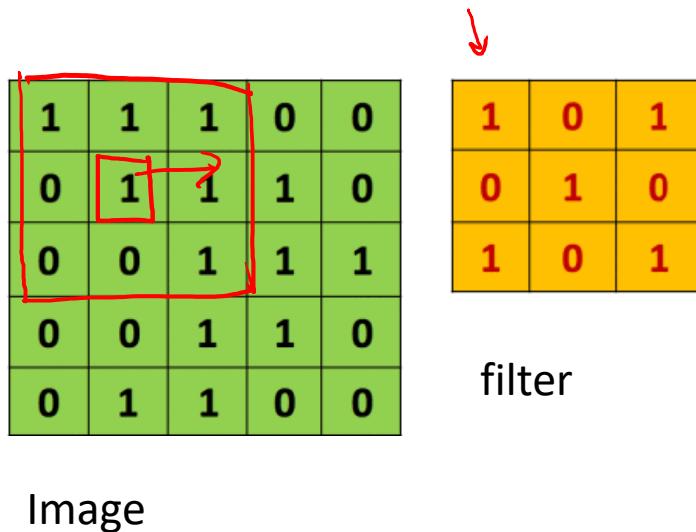
Dealing with images

An image is a multi-dimension array

| | | Width | | | | | |
|--|--|-------|-----|-----|-----|----|------------------|
| | | | | | | | Depth or Channel |
| | | | | | | | 165 187 209 58 7 |
| | | 253 | 144 | 120 | 251 | 41 | 147 204 |
| | | 67 | 100 | 32 | 241 | 23 | 165 30 |
| | | 209 | 118 | 124 | 27 | 58 | 201 79 |
| | | 210 | 236 | 105 | 169 | 19 | 218 156 |
| | | 35 | 178 | 199 | 197 | 4 | 14 218 |
| | | 115 | 104 | 34 | 111 | 19 | 196 |
| | | 32 | 69 | 231 | 203 | 74 | |



What is Convolution

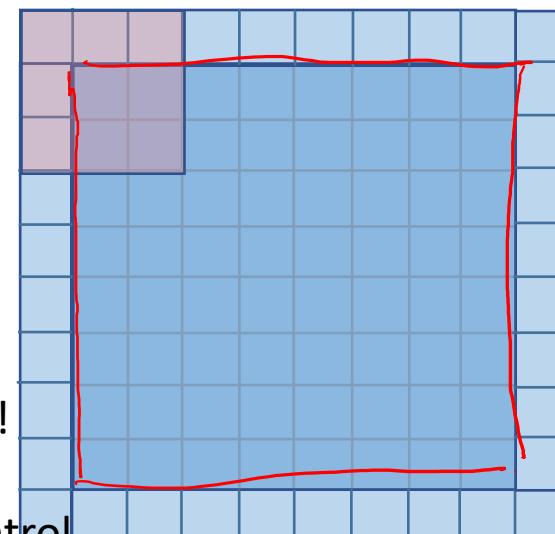
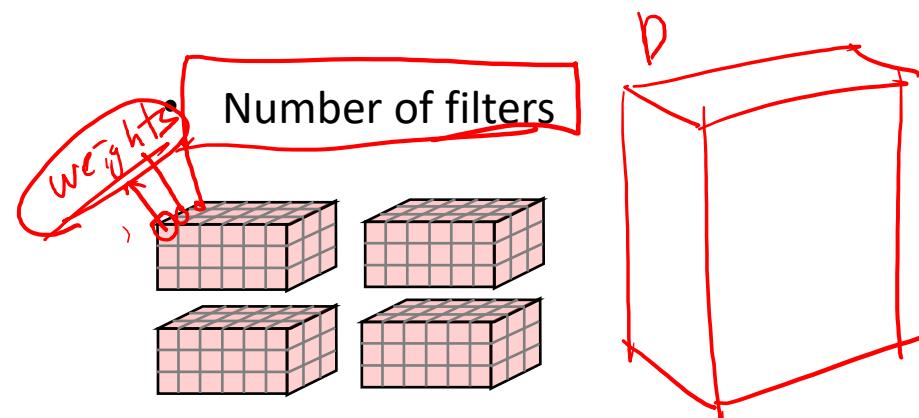
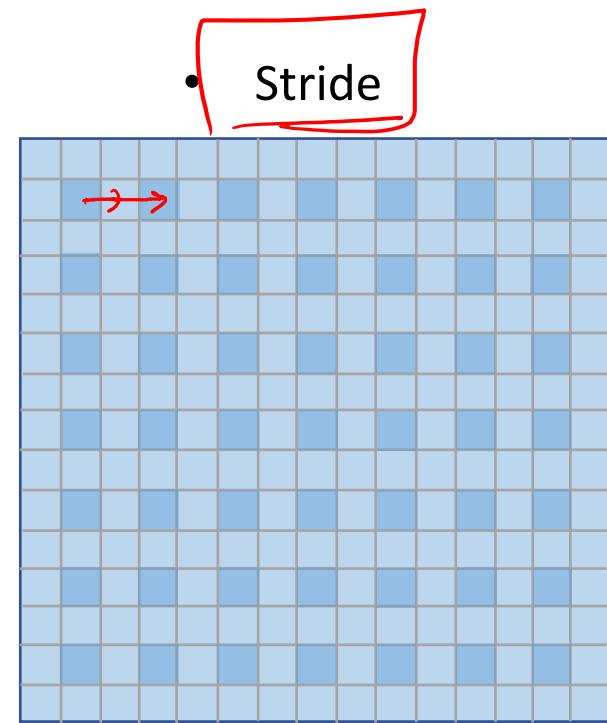
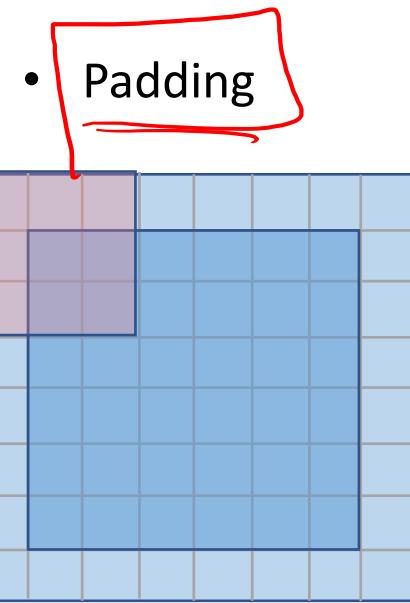
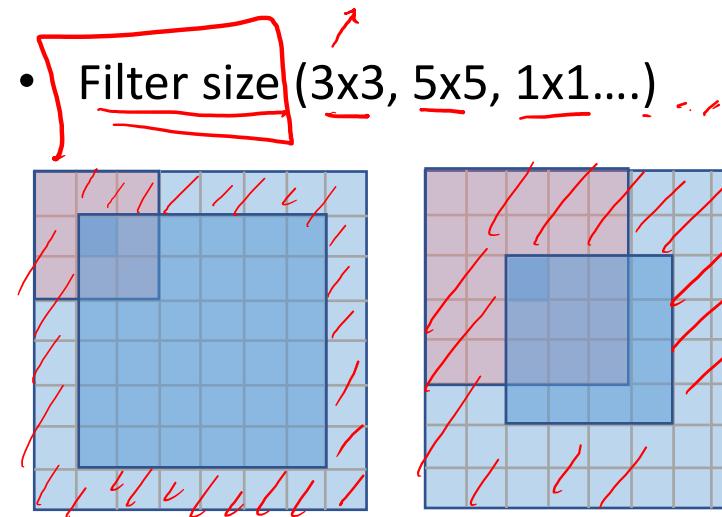


Convolved
Feature

$$\left(\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} * \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \right) [2, 2]$$

$$= (i \cdot 1) + (h \cdot 2) + (g \cdot 3) + (f \cdot 4) + (e \cdot 5) + (d \cdot 6) + (c \cdot 7) + (b \cdot 8) + (a \cdot 9)$$

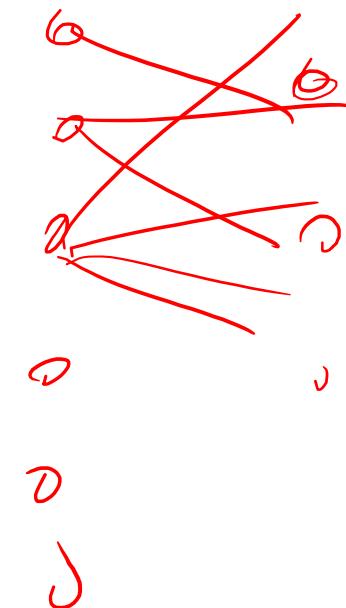
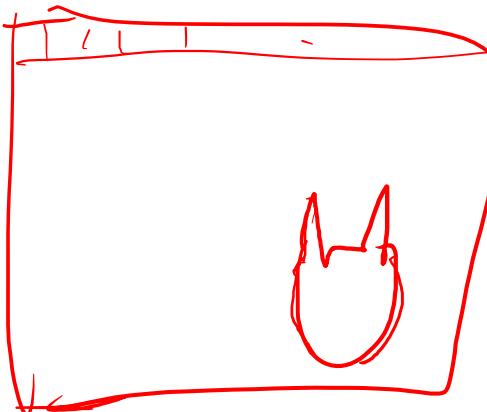
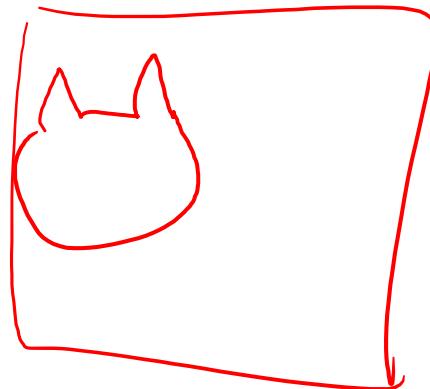
Convolutional Layer- hyper parameters



** Parameters and Hyper parameters are different!
Parameters = Weights to be optimized.
Hyperparameters = design parameters you can control

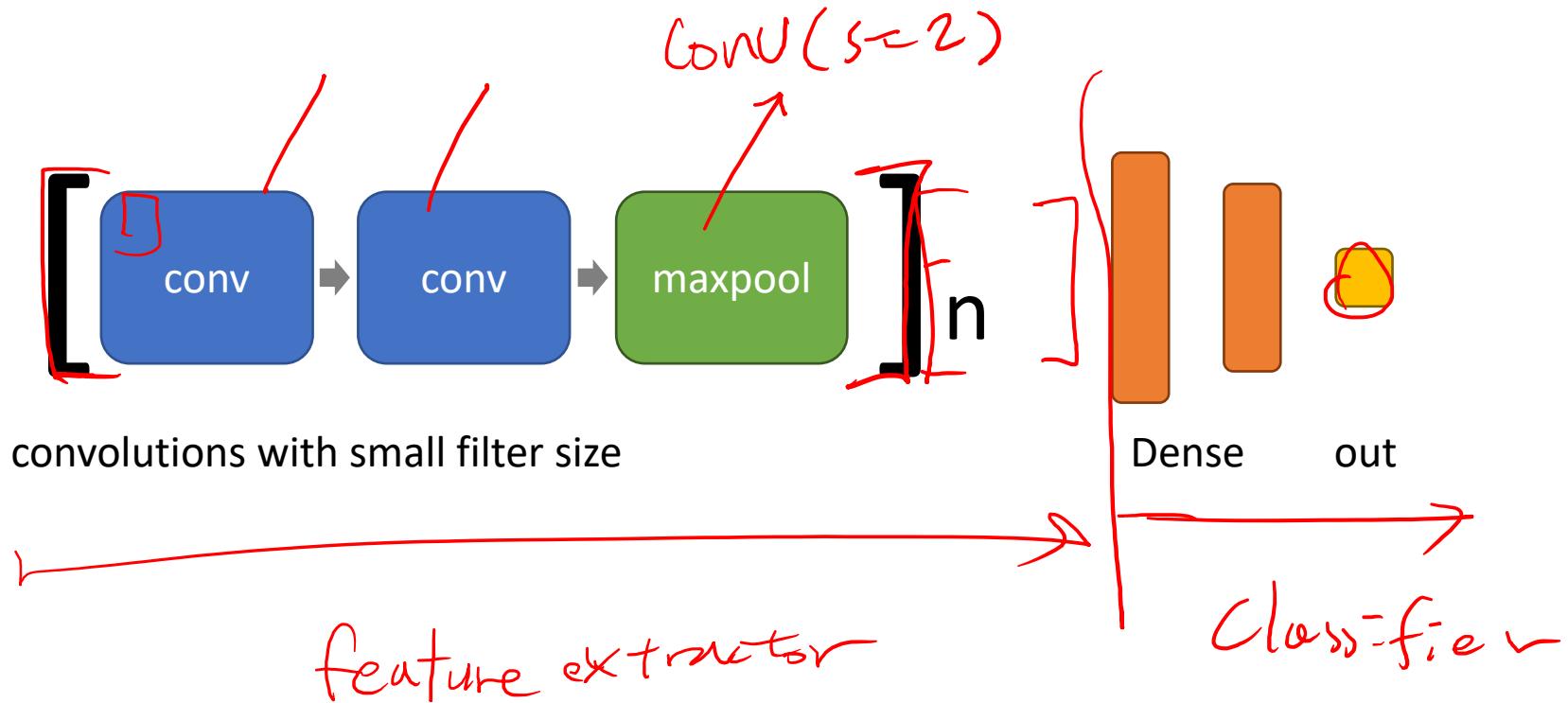
Why CNN?

- Images have big pixels!
- Fully-connected neural network would have too many parameters!
- Translational invariance in images

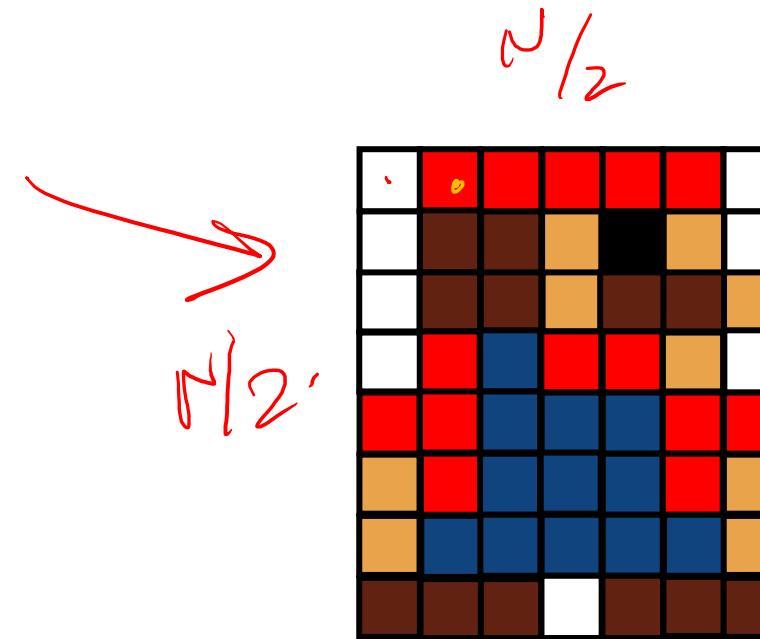
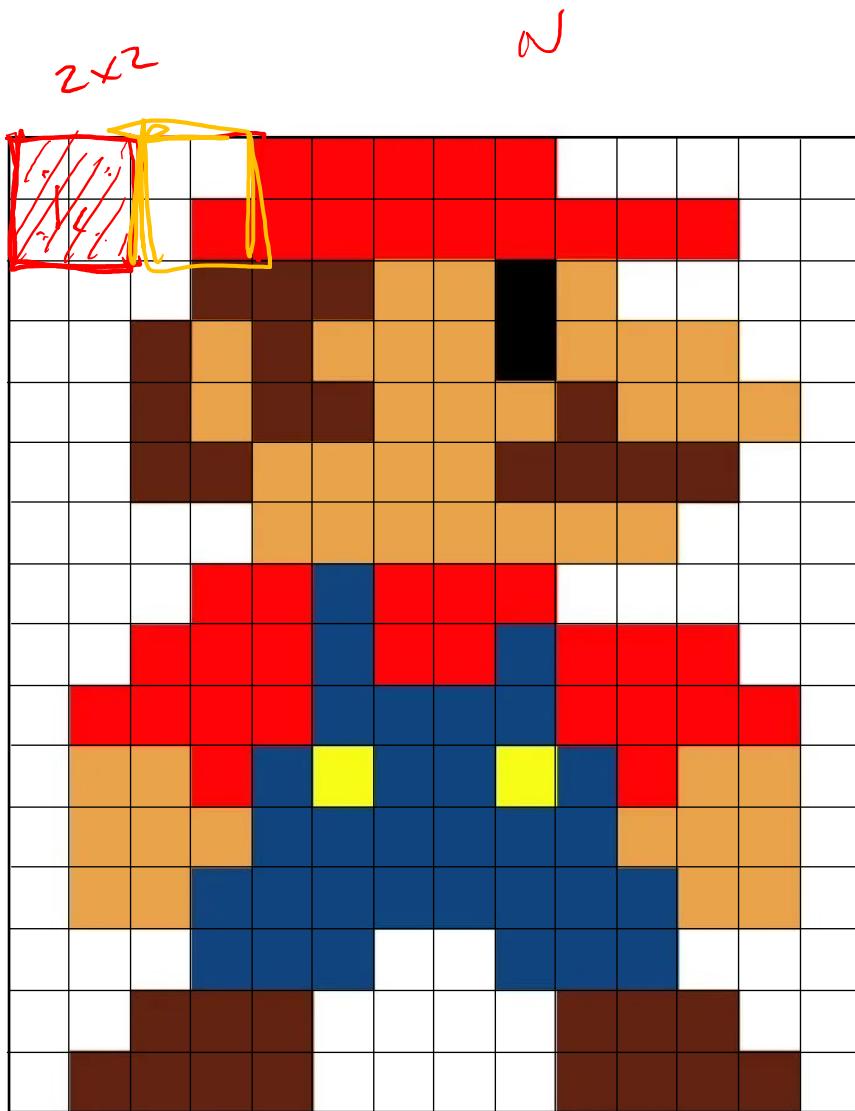


Deep convolutional neural network Architectures

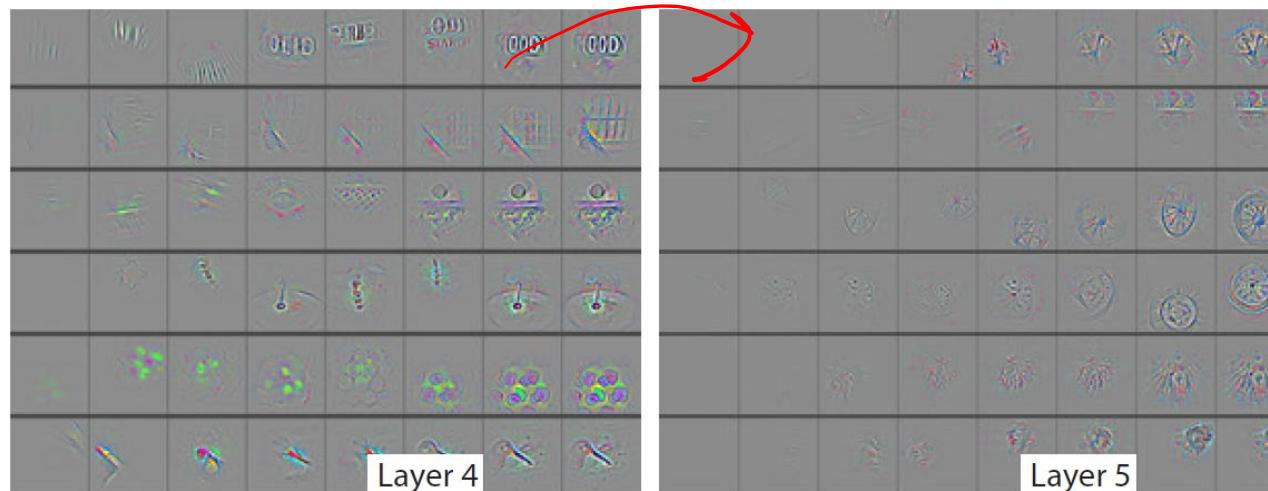
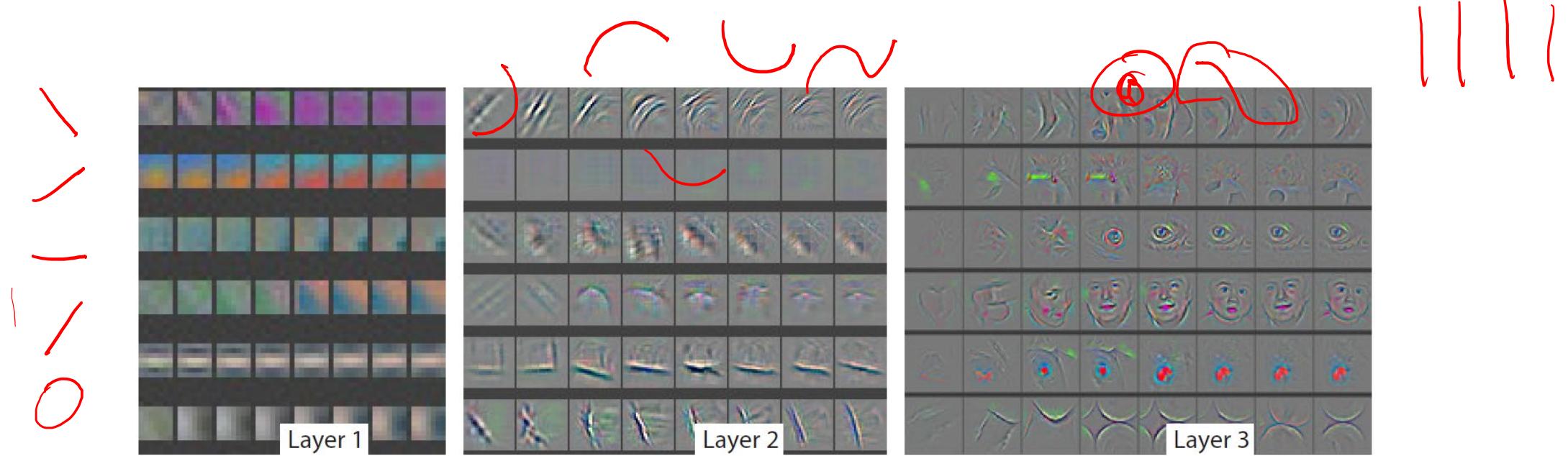
Typical CNN architecture



Max pooling operation

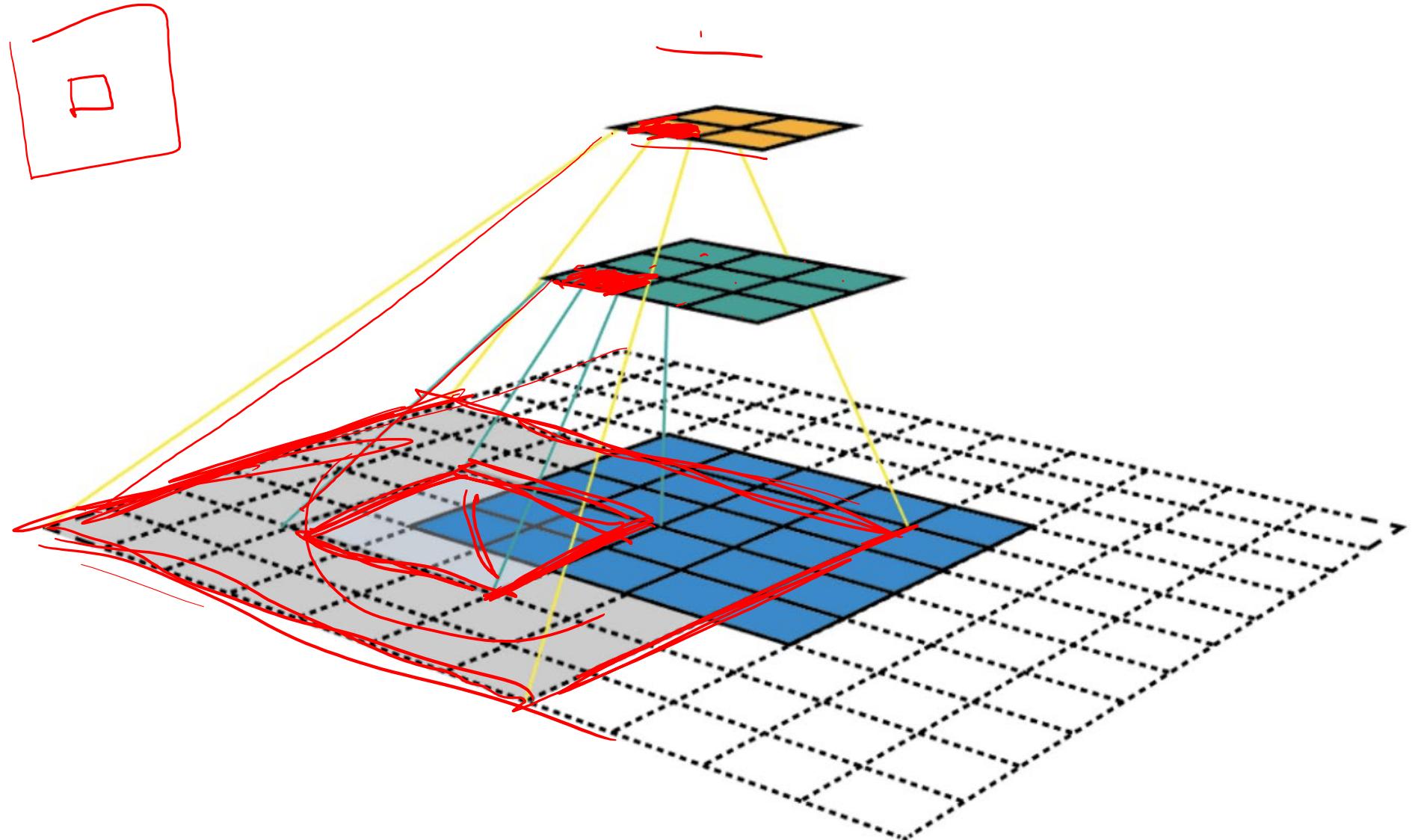
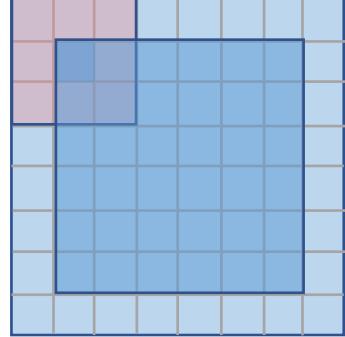


What does the deep convolutional layers do?

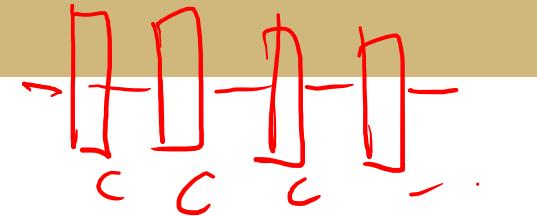


[arXiv:1311.2901v3](https://arxiv.org/abs/1311.2901v3)

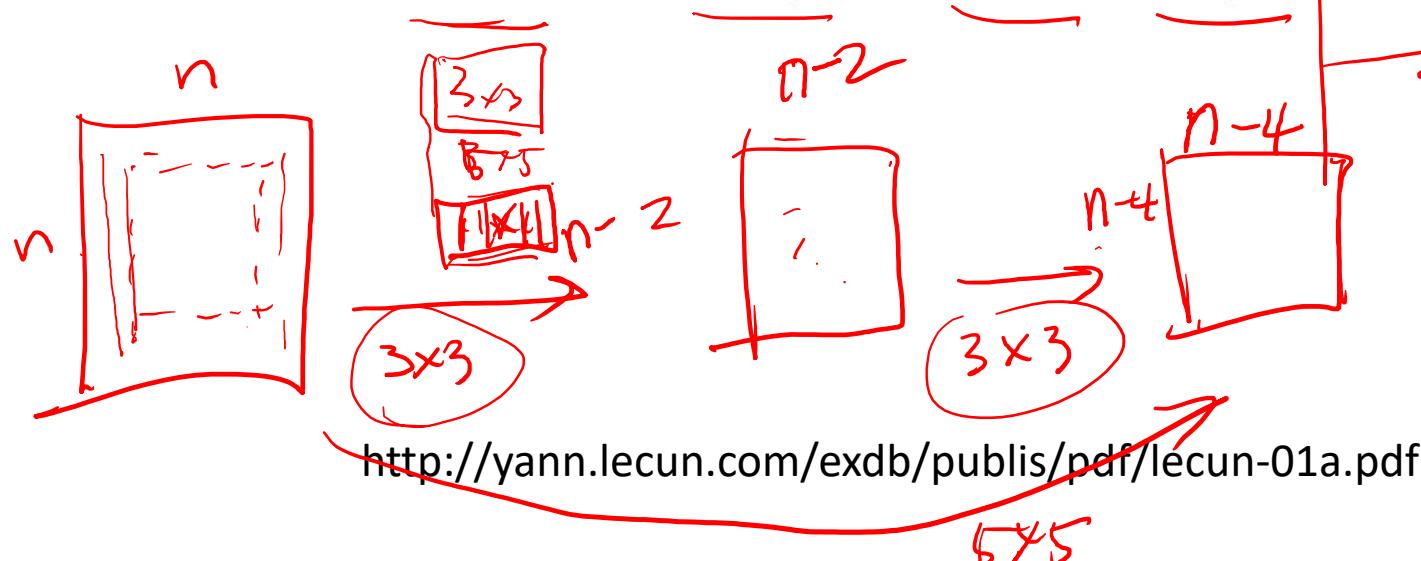
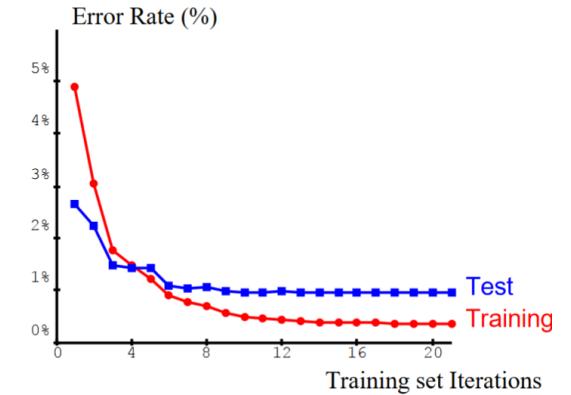
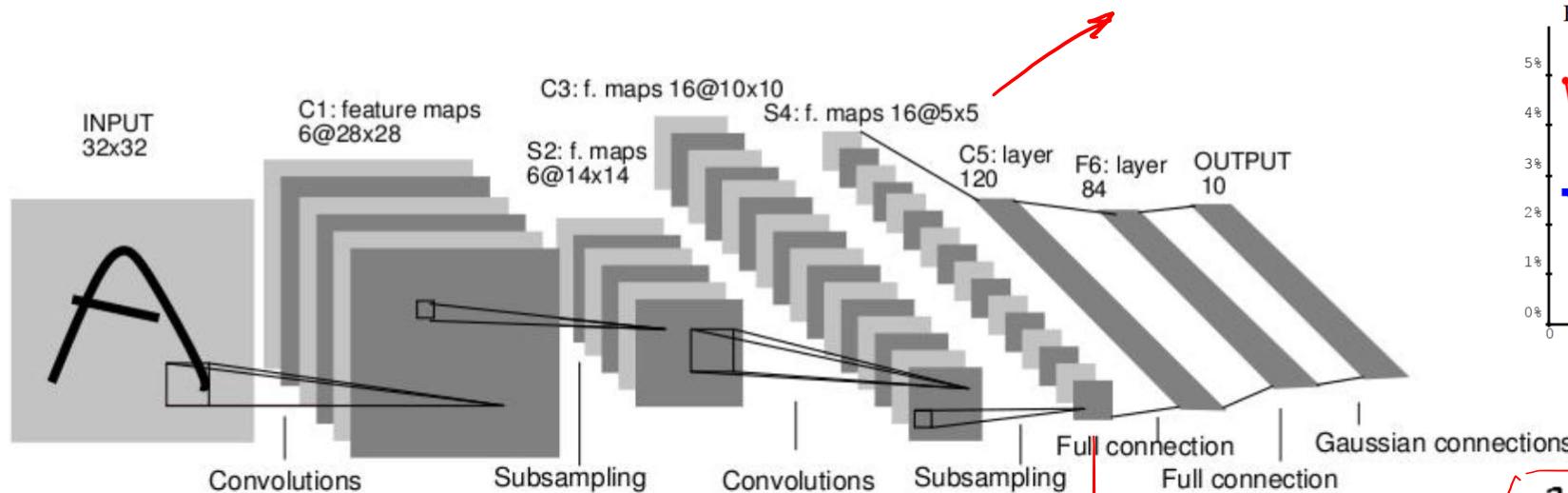
What do the multiple convolution layers do?



Modern CNN



LeNet 5 (1998)



A 10x10 grid of handwritten digits from the MNIST dataset, enclosed in a red border. The digits are labeled with their corresponding values: 3, 6, 8, 1, 7, 9, 6, 6, 9, 1, 6, 7, 5, 7, 8, 6, 3, 4, 8, 5, 2, 1, 7, 9, 7, 1, 2, 8, 4, 6, 4, 8, 1, 9, 0, 1, 8, 8, 9, 4, 7, 6, 1, 8, 6, 4, 1, 5, 6, 0, 7, 5, 9, 2, 6, 5, 8, 1, 9, 7, 1, 2, 2, 2, 2, 3, 4, 4, 8, 0, 0, 2, 3, 8, 0, 7, 3, 8, 5, 7, 0, 1, 4, 6, 4, 6, 0, 2, 4, 3, 7, 1, 2, 8, 7, 6, 9, 8, 6, 1.

MNIST dataset

Natural Images Dataset

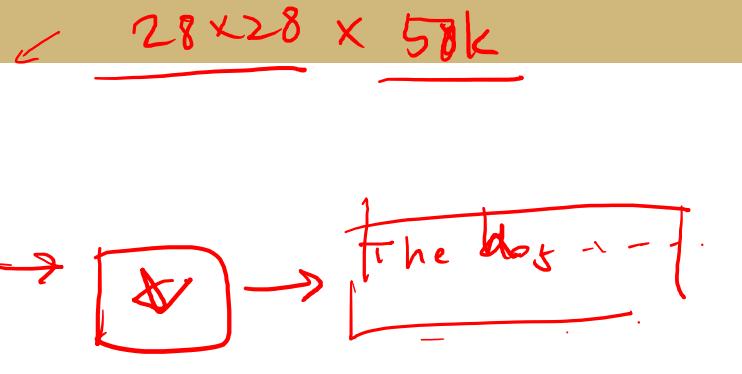
IMAGENET

SUN, 131K
[Xiao et al. '10]

LabelMe, 37K
[Russell et al. '07]

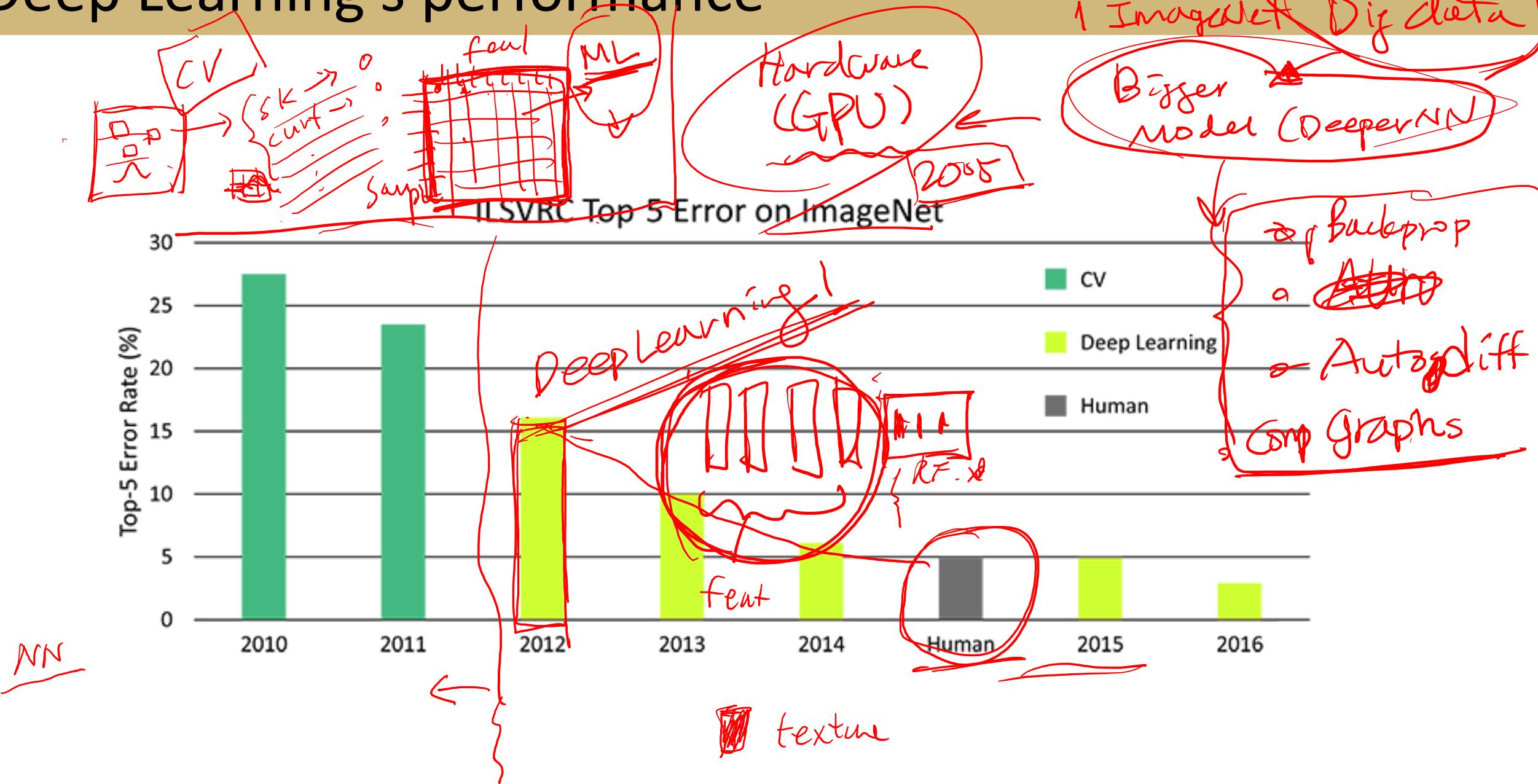
PASCAL VOC, 30K
[Everingham et al. '06-'12]

Caltech101, 9K
[Fei-Fei, Fergus, Perona, '03]



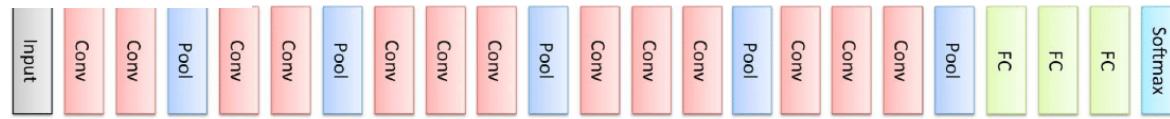
- * Massive size
- * High resolution
- * High quality annotation
- * Ontology
 - Carnivore
 - Canine
 - Dog
 - + Working Dog
 - Husky
 - * Free

Deep Learning's performance



Most successful CNN models from ImageNet

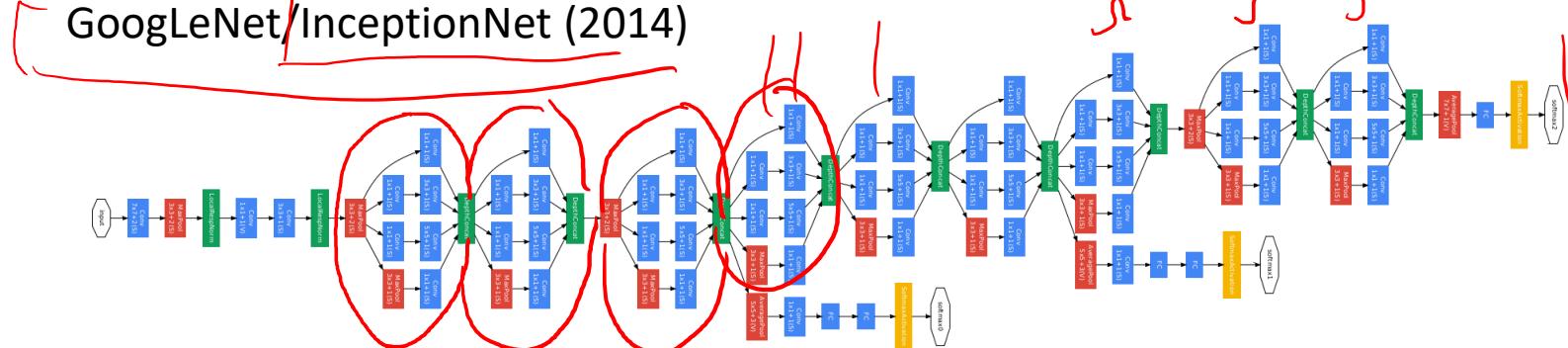
VGGNet (2014)



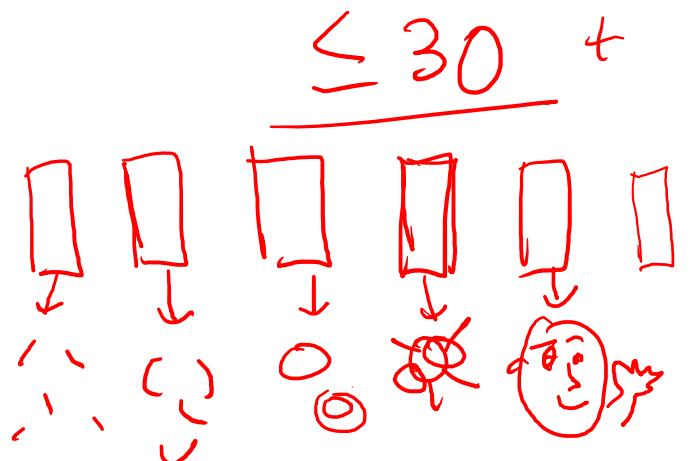
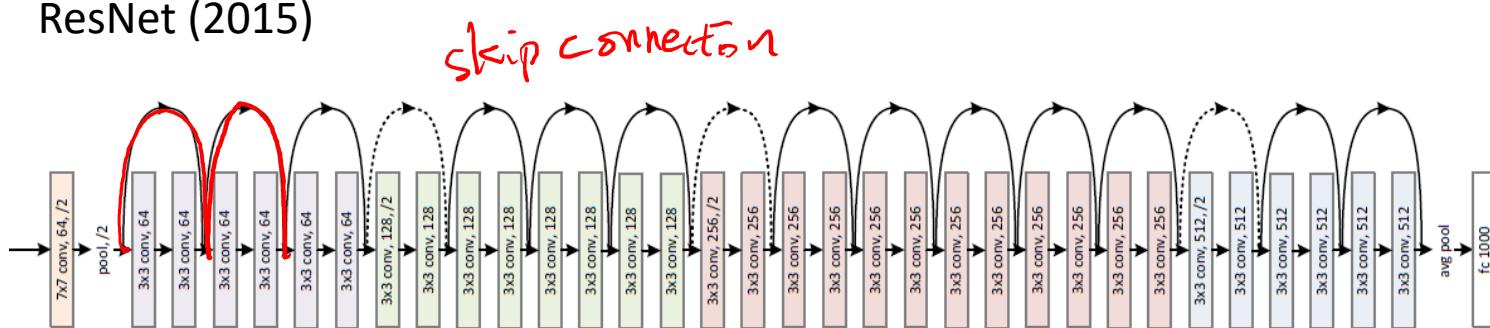
+ Simple

≤ 20 +

GoogLeNet/InceptionNet (2014)



ResNet (2015)



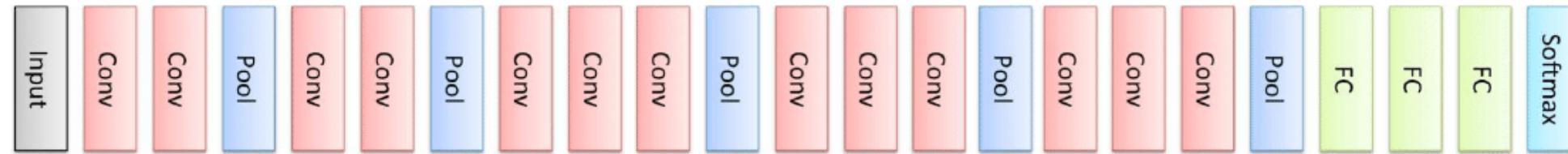
≤ 30 +

Convolutional Neural Network



VGGNet

VGGNet



| ConvNet Configuration | | | | | |
|-----------------------------|------------------------|-------------------------------|--|--|---|
| A | A-LRN | B | C | D | E |
| 11 weight layers | 11 weight layers | 13 weight layers | 16 weight layers | 16 weight layers | 19 weight layers |
| input (224 × 224 RGB image) | | | | | |
| conv3-64 | conv3-64 LRN | conv3-64 conv3-64 | conv3-64 conv3-64 | conv3-64 conv3-64 | conv3-64 conv3-64 |
| maxpool | | | | | |
| conv3-128 | conv3-128 | conv3-128 conv3-128 | conv3-128 conv3-128 | conv3-128 conv3-128 | conv3-128 conv3-128 |
| maxpool | | | | | |
| conv3-256 conv3-256 | conv3-256 conv3-256 | conv3-256 conv3-256 | conv3-256 conv3-256 conv1-256 | conv3-256 conv3-256 conv3-256 | conv3-256 conv3-256 conv3-256 conv3-256 |
| maxpool | | | | | |
| conv3-512 conv3-512 | conv3-512 conv3-512 | conv3-512 conv3-512 | conv3-512 conv3-512 conv1-512 | conv3-512 conv3-512 conv3-512 | conv3-512 conv3-512 conv3-512 conv3-512 |
| maxpool | | | | | |
| conv3-512 conv3-512 | conv3-512 conv3-512 | conv3-512 conv3-512 | conv3-512 conv3-512 conv1-512 | conv3-512 conv3-512 conv3-512 | conv3-512 conv3-512 conv3-512 conv3-512 |
| maxpool | | | | | |
| FC-4096 | | | | | |
| FC-4096 | | | | | |
| FC-1000 | | | | | |
| soft-max | | | | | |

<https://arxiv.org/abs/1409.1556>

64

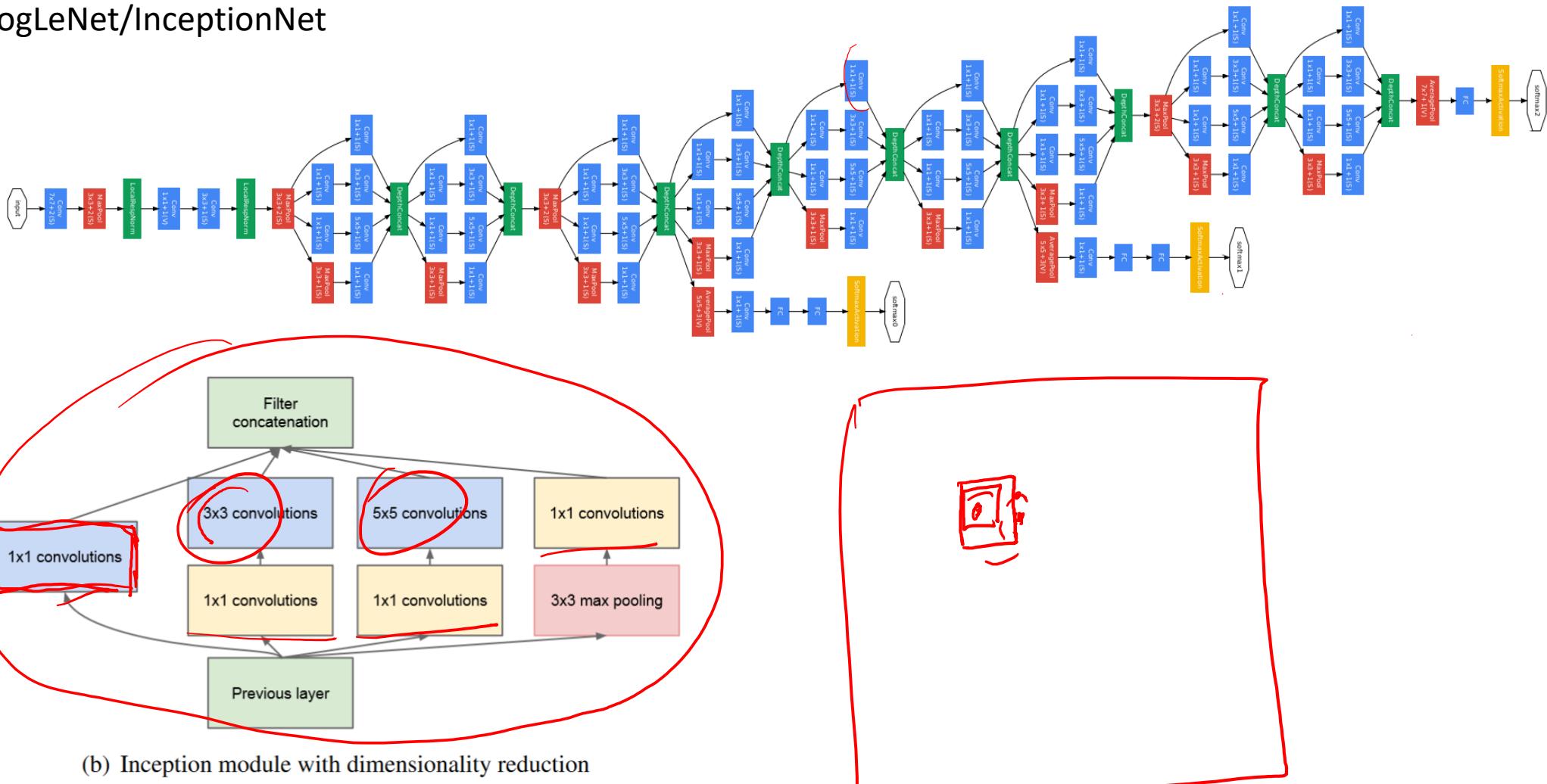
128

256

512

GoogLeNet

GoogLeNet/InceptionNet



ResNet

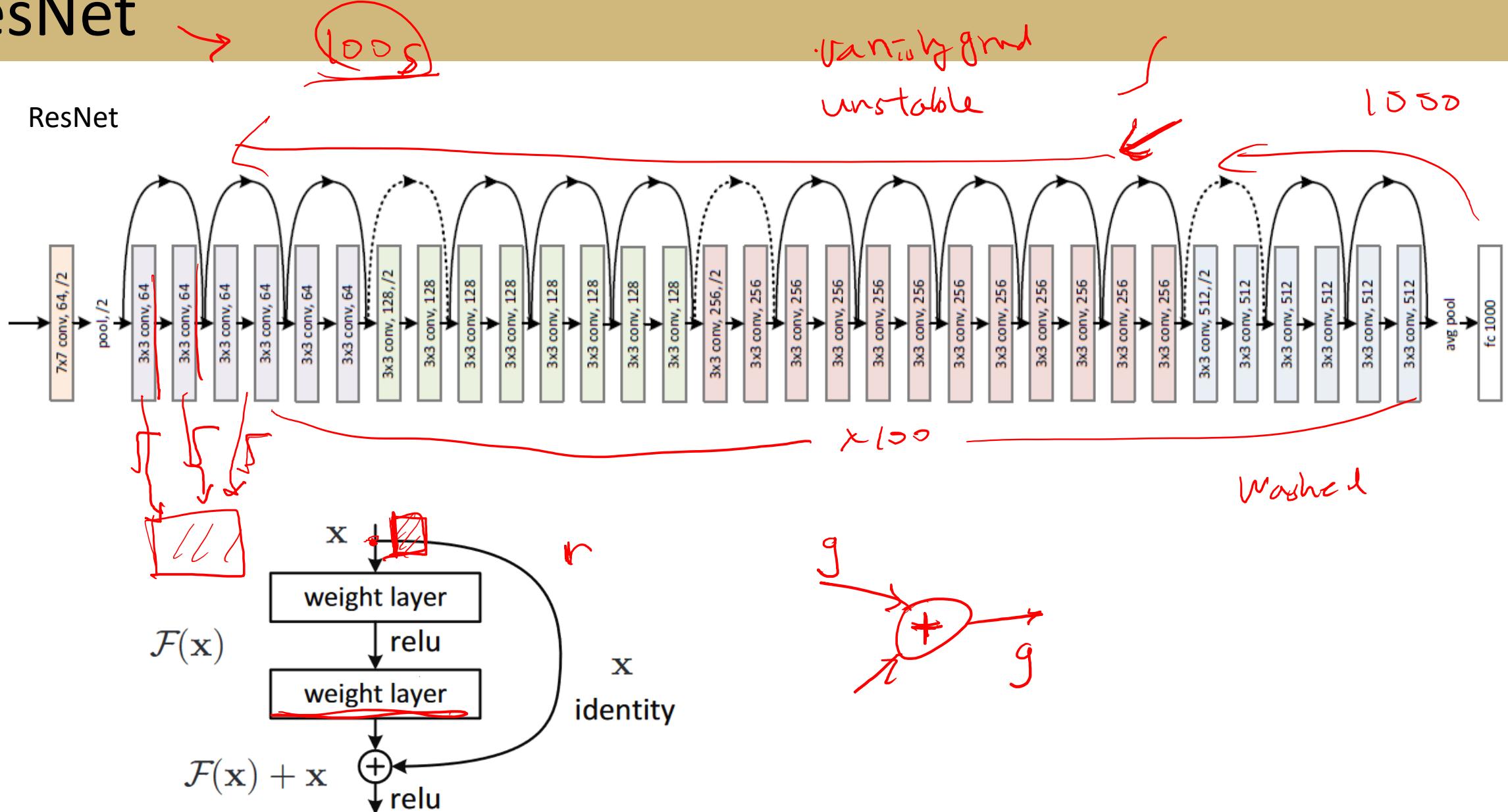
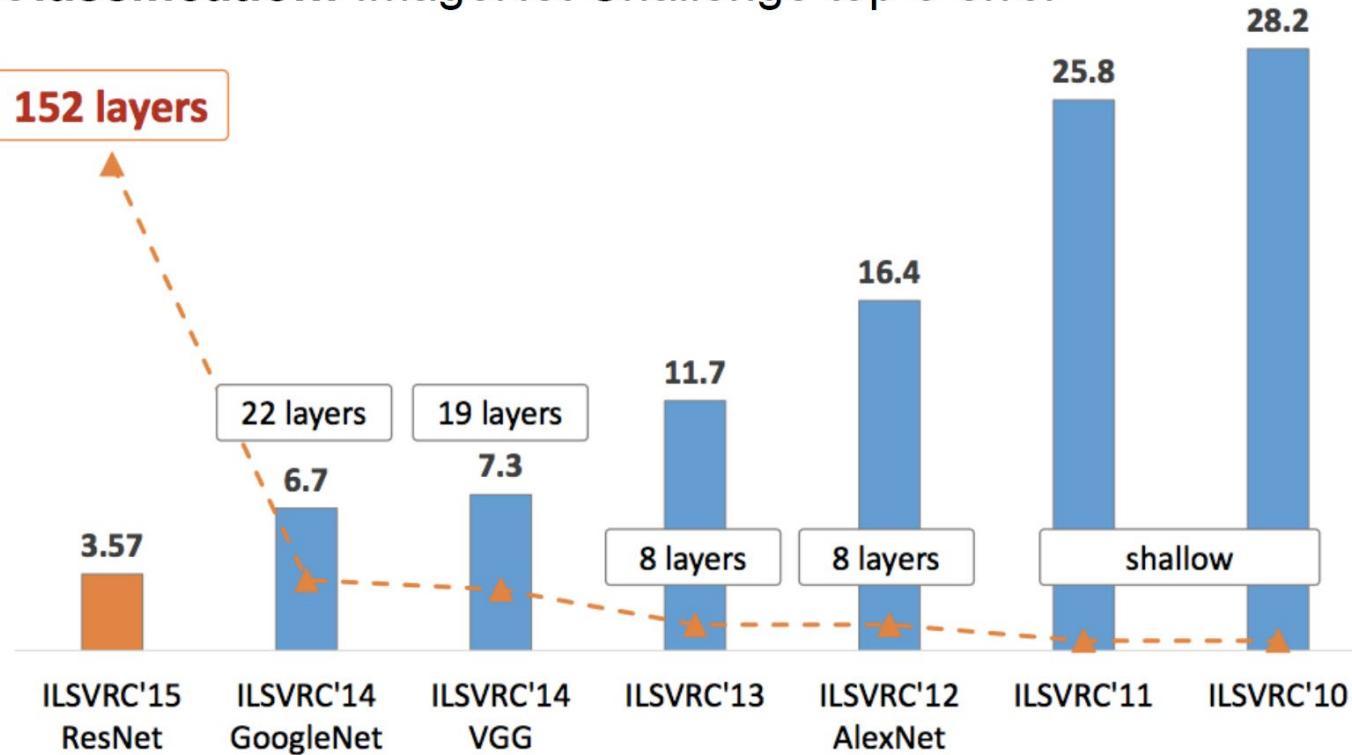


Figure 2. Residual learning: a building block.

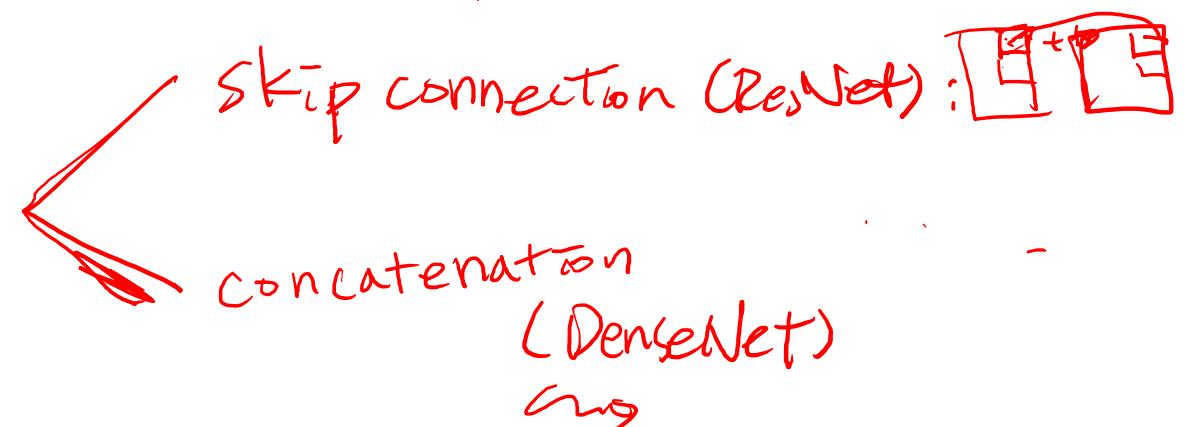
ResNet

Classification: ImageNet Challenge top-5 error



Architecture summary

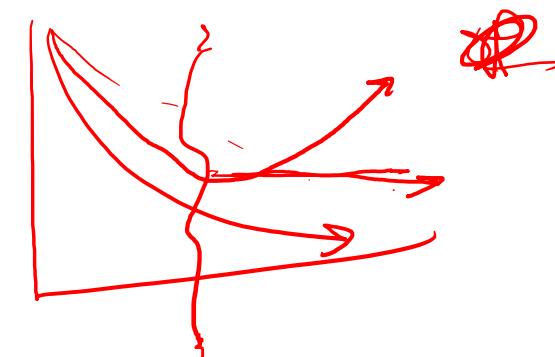
- If the data is bigger and complex, deep network is beneficial
- Improve computational efficiency while keeping the same depth
 - e.g. small filter size, 1x1 filter compression
- Designs to have a better gradient propagation (avoid vanishing or exploding gradients)



Design Choices or Training

Key Tuning Parameters in Training

- Optimization method SGD ; Adam, RMSProp, ...
- Learning rate, momentum, etc
- Number of epochs
- Regularization,
 • L1, L2 ;
 • Dropout
 • Batch Norm



Monitoring Overfitting in Training

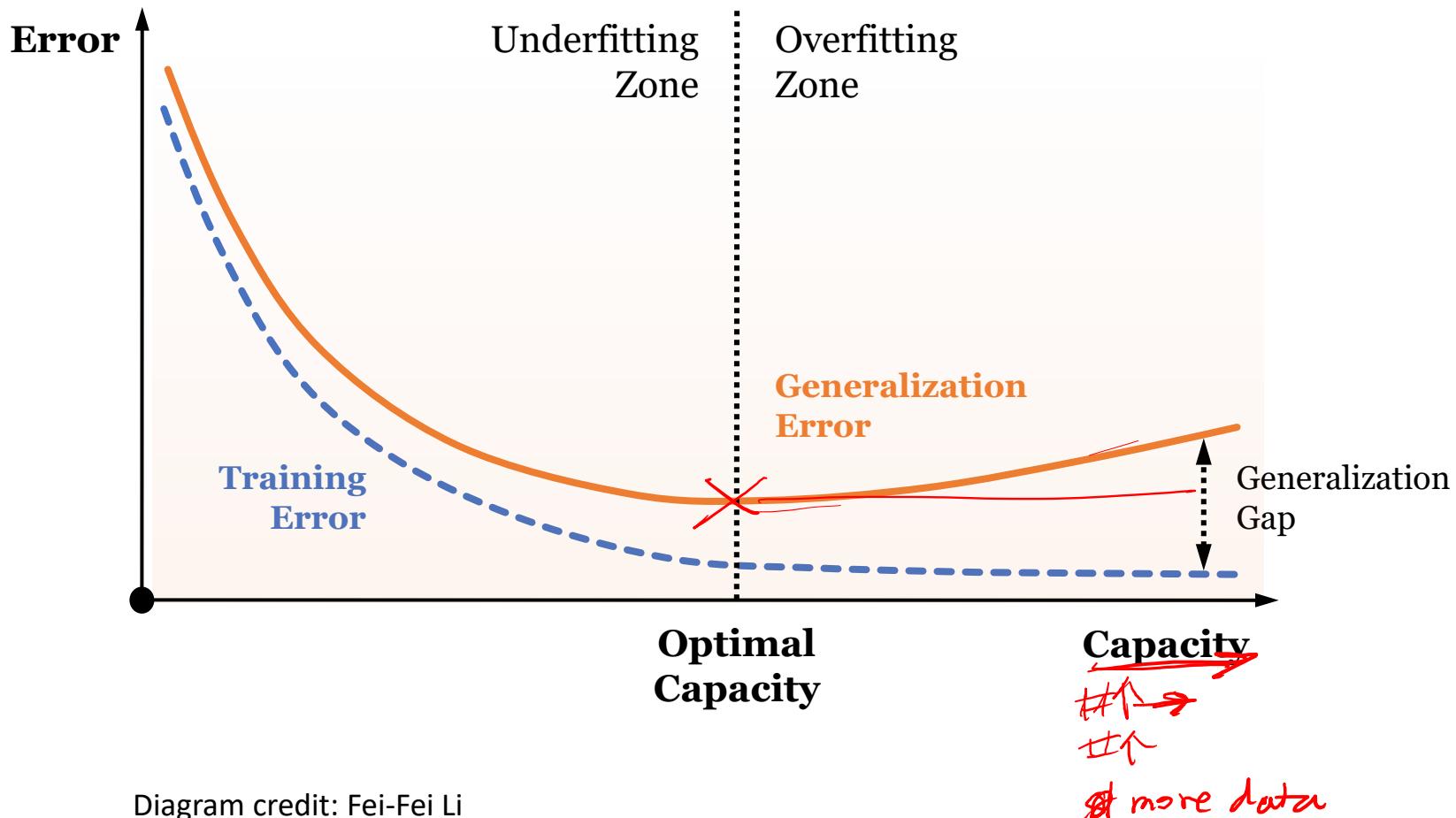
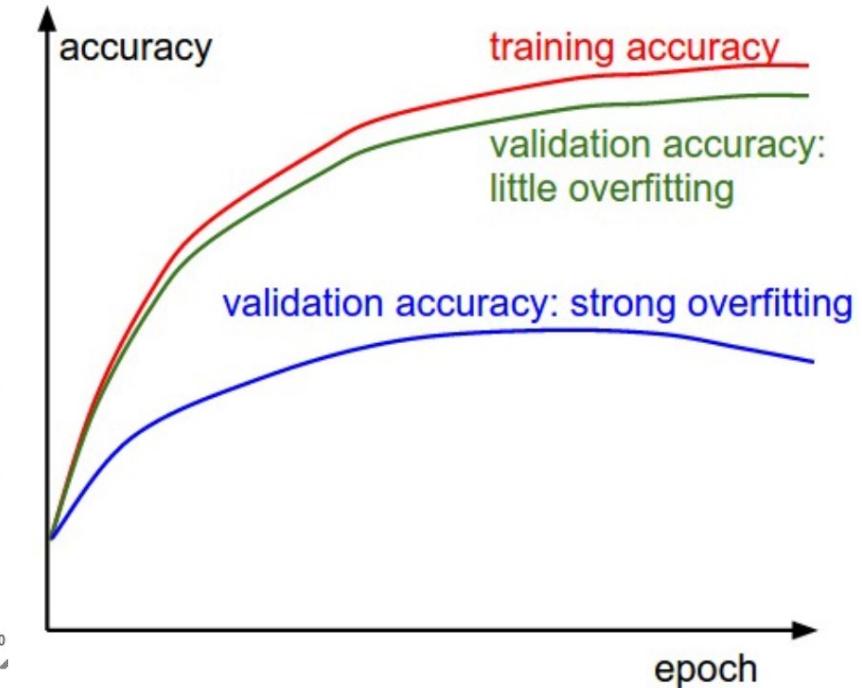
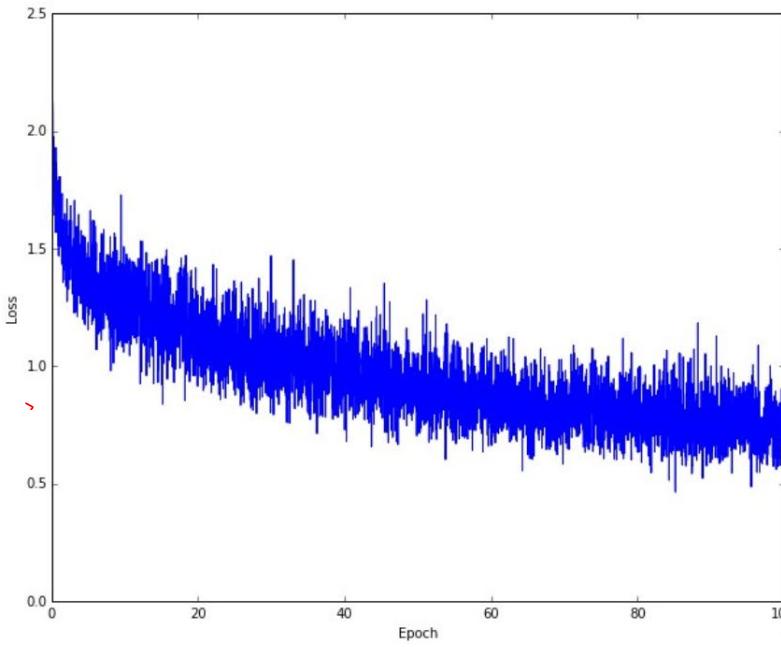
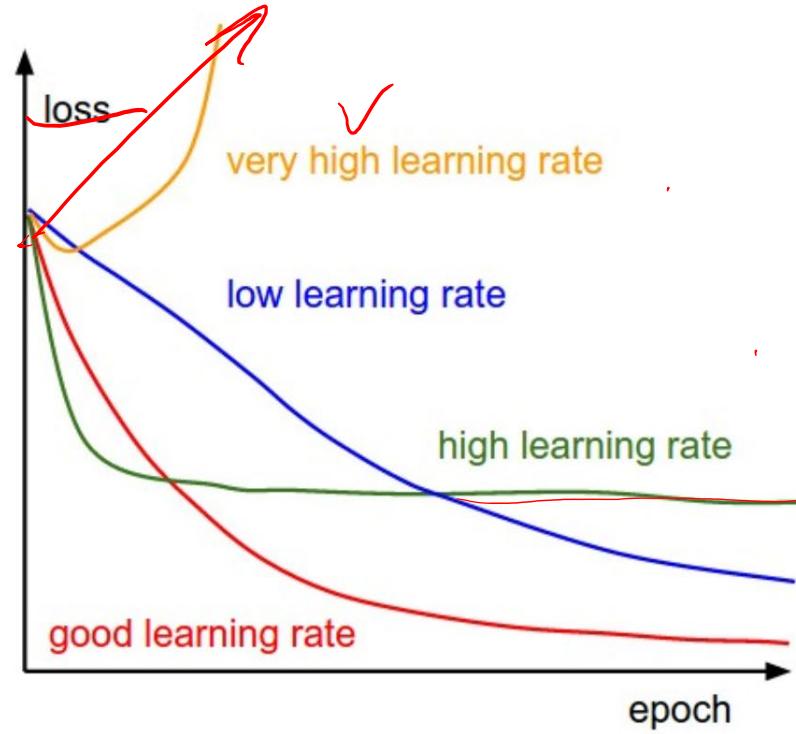


Diagram credit: Fei-Fei Li

Monitoring Overfitting in Training



```
tf.keras.optimizers.SGD(  
    learning_rate=0.01, momentum=0.0, nesterov=False, name='SGD', **kwargs  
)
```

Popular options to tweak

- learning_rate: the base learning rate
- momentum
- decay ✓
- nestrov
- (advanced) callback

Stochastic Gradient Descent with momentum

SGD with learning rate alone is slow to converge

Adding a momentum (moving average of a weight) can make it faster

$$\begin{aligned} \mathbf{v} &\leftarrow \alpha \mathbf{v} - \epsilon \nabla_{\theta} \left(\frac{1}{m} \sum_{i=1}^m L(f(x^{(i)}; \theta), y^{(i)}) \right), \\ \theta &\leftarrow \theta + \mathbf{v}. \end{aligned}$$

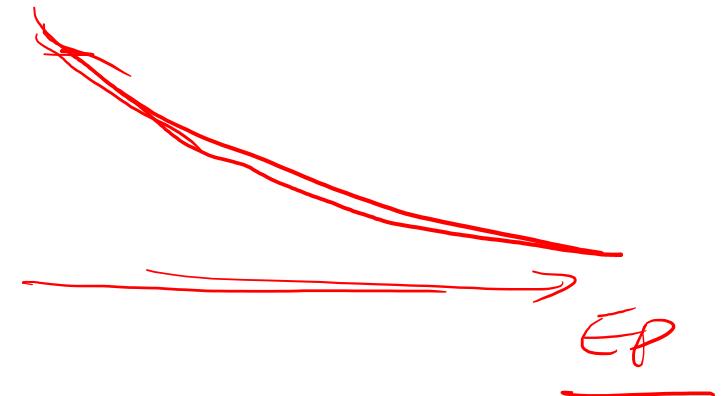
** see what happens when the gradient is 0 (on plateau)

Stochastic Gradient Descent with decay

Learning rate scheduling using decay

For iteration k (epoch)

$$\epsilon_k = (1 - \alpha)\epsilon_0 + \alpha\epsilon_\tau \quad \alpha = \frac{k}{\tau}$$



** In the algorithm pseudocode k is for step (each mini batch),
and decay learning rate by step,
but normally we decrease learning rate each epoch

warning- different notations used (from deeplearningbook.org)

Advanced optimization

RMSprop

Variant of Adadelta

RMSprop takes a moving average when it calculate the RMS of the gradient

$$E[g^2]_t = 0.9E[g^2]_{t-1} + 0.1g_t^2$$
$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$$

✓
✗

Advanced optimization

Adaptive Moment Estimation (Adam)

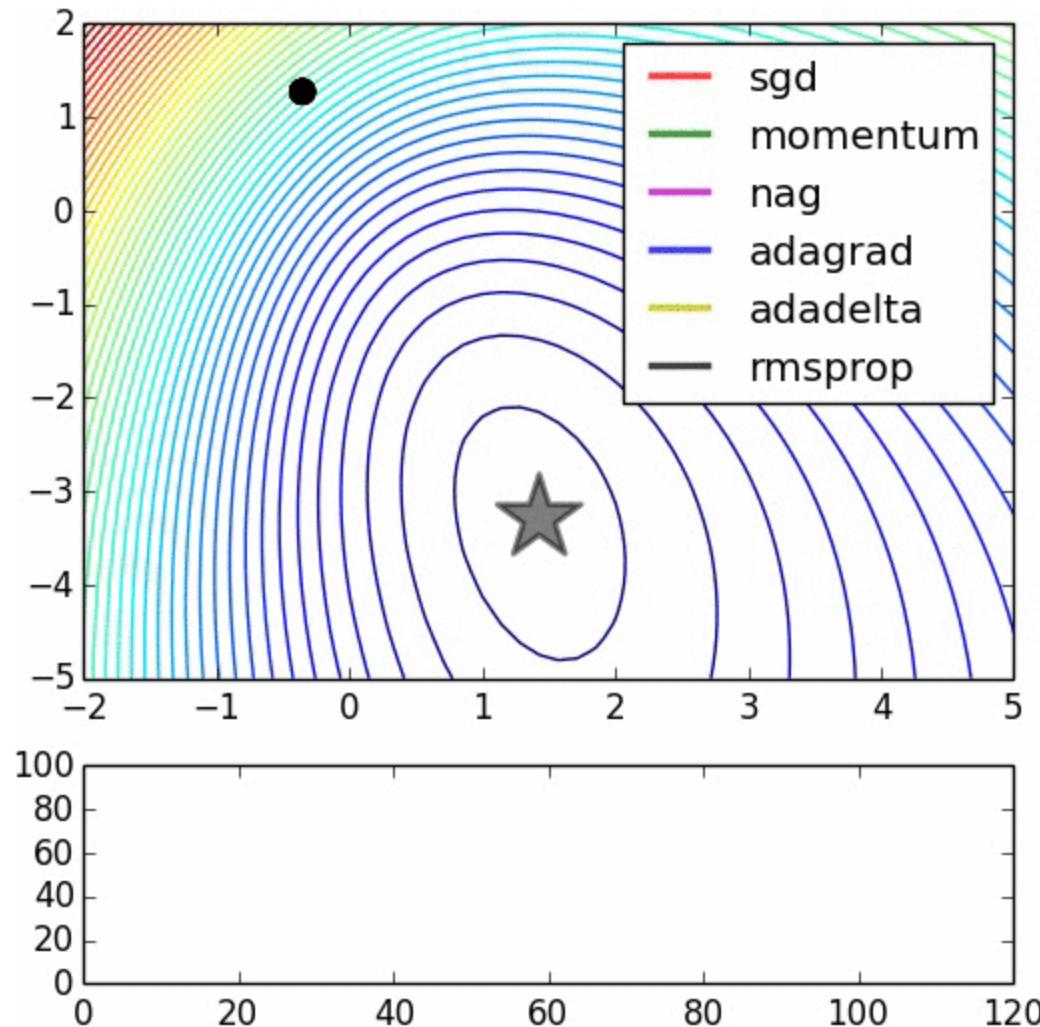
Mimics momentum for gradient and gradient-squared

m_t and v_t are estimates of the first moment (the mean) and the second moment (the uncentered variance) of the gradients

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1 - \beta_1) g_t \\ v_t &= \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \end{aligned}$$

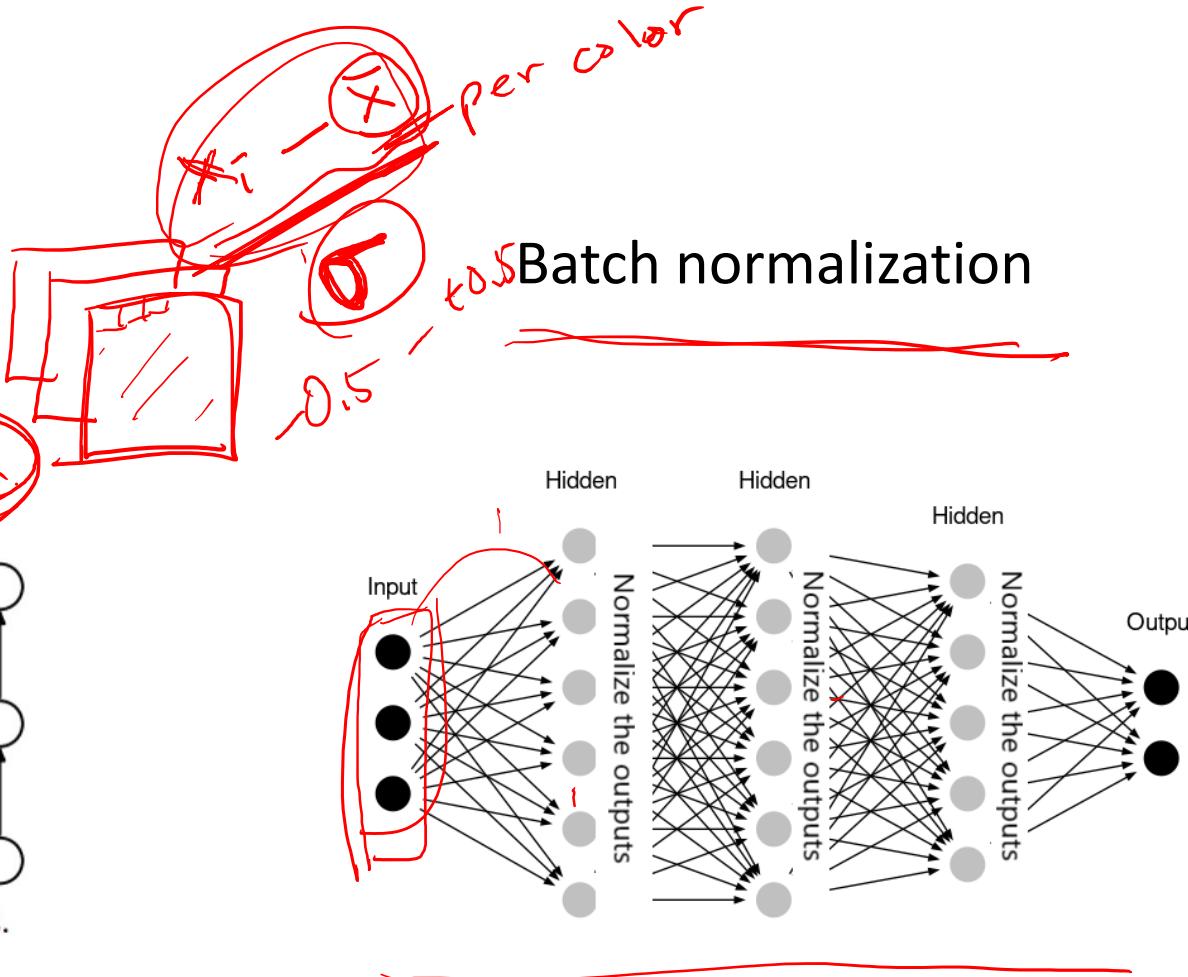
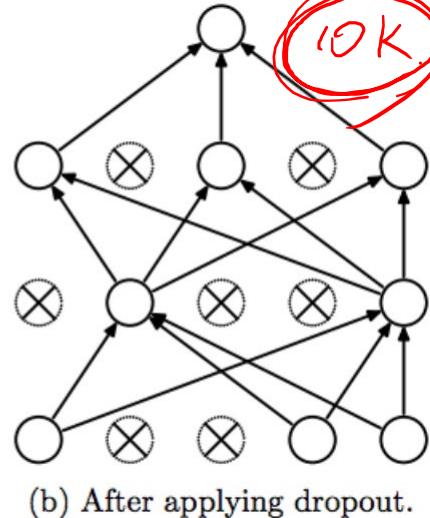
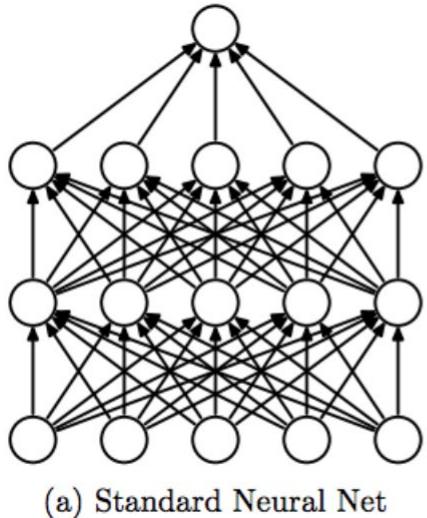
$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t$$

Optimization: Momentum



Ways to reduce overfitting

Dropout



Practical Tips for CNN

Optimization

- Learning rate (0.01~0.0001)
- Optimization method: Adam or RMSProp

Architecture

- ReLU/PReLU for hidden layers, (Sigmoid/Softmax/Tanh/PReLU) for the output layer
- 3x3 filters
- [Conv-Conv-MaxPool]_n structure

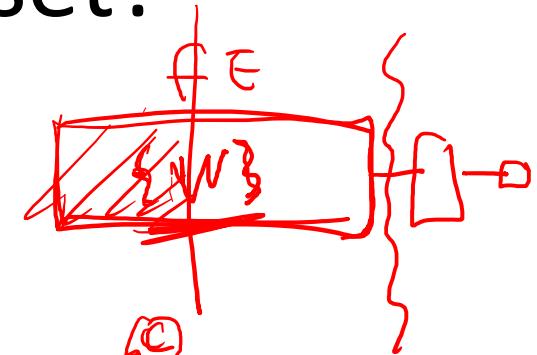
Regularization

- L2 regularization
- Dropouts)
- Batch Normalization ✓

Transfer Learning

How long it takes to train on a large dataset?

$$\text{Multi GPU} + \text{Deep CNN} + \text{Large Dataset} = \text{2-3 Weeks Training time}$$



Model (ImageNet) $\rightarrow \{0\}$

Transfer learning

- As fixed feature extractor: remove the output layer, weights frozen
- Fine-tuning the CNN: also let weights updated
- Use part of layers

Transfer Learning

When should I use a pre-trained network?

- New dataset is small and similar to original dataset (X) *Overfit*
- New dataset is large and similar to the original dataset (O) *Fit*
- New dataset is small but very different from the original dataset (X)
- New dataset is large and very different from the original dataset (O)

M.I.



Various models pre-trained on ImageNet

Keras <https://keras.io/applications/>