Image Classification

План

Recap

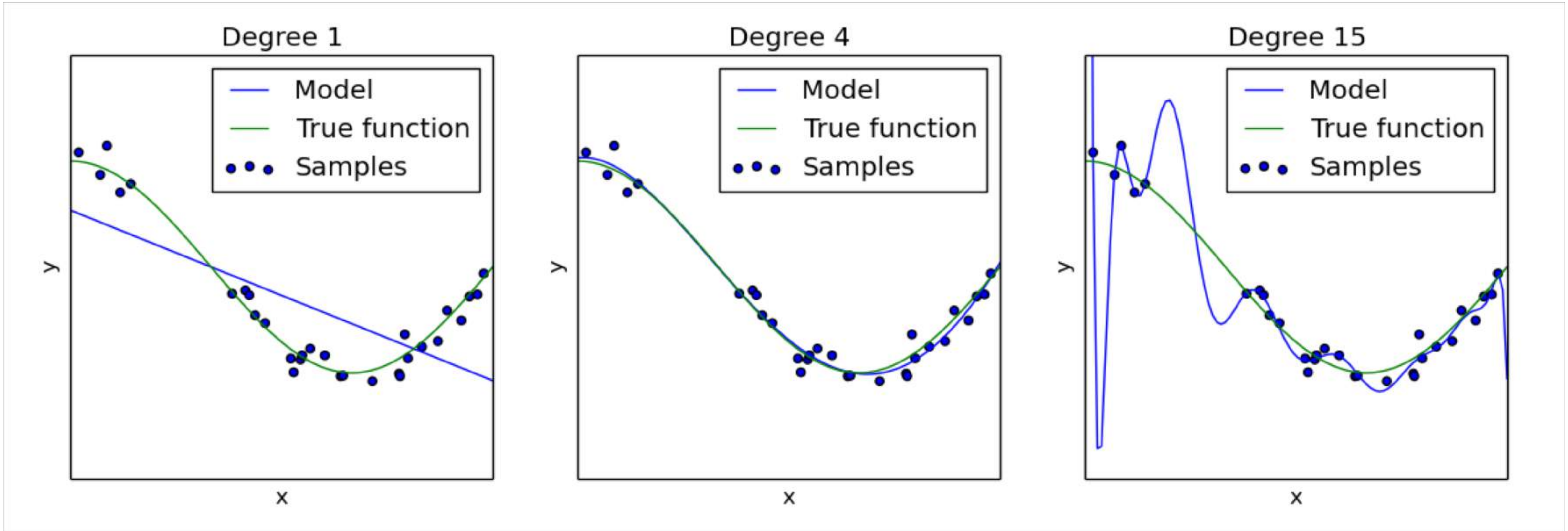
Image Classification

CNN Architectures

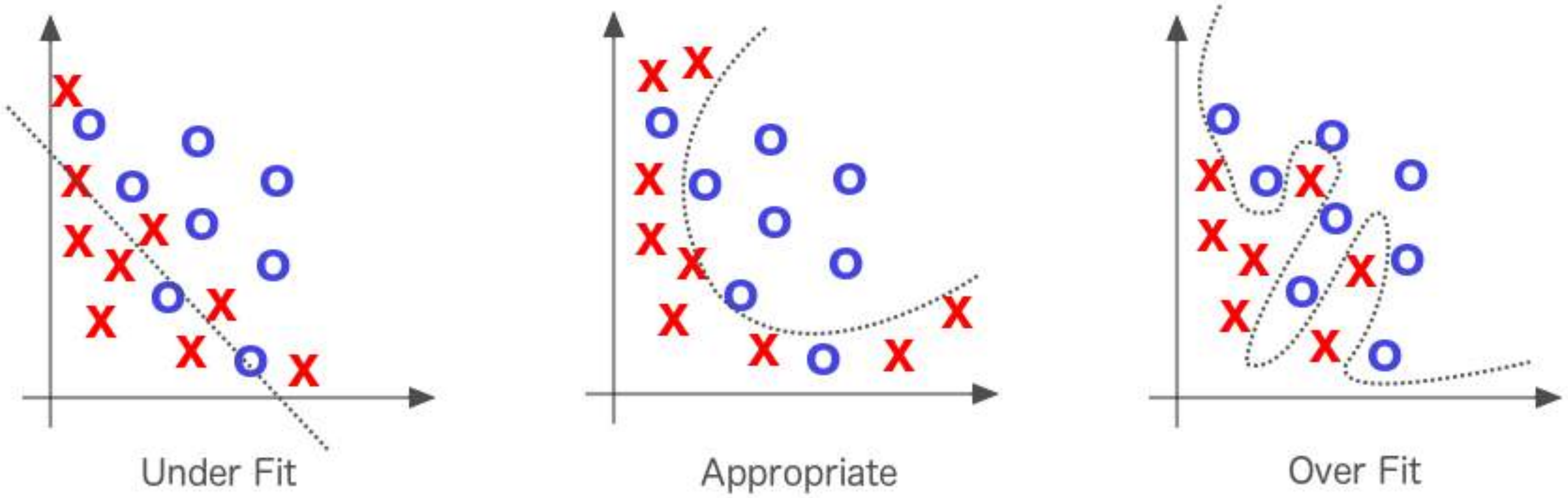
Transfer Learning

Recap: Over/Under fitting

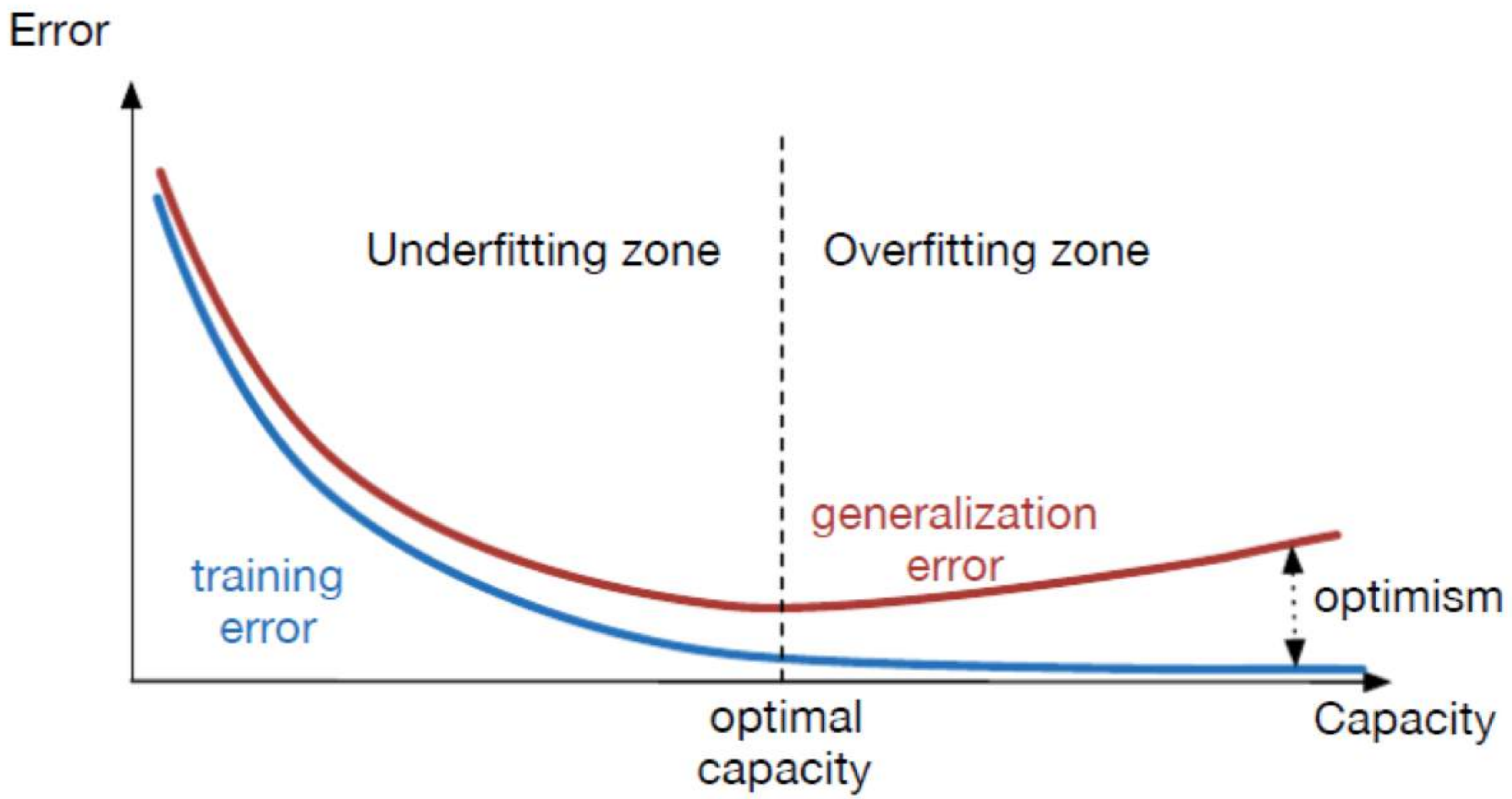
Overfitting/Underfitting



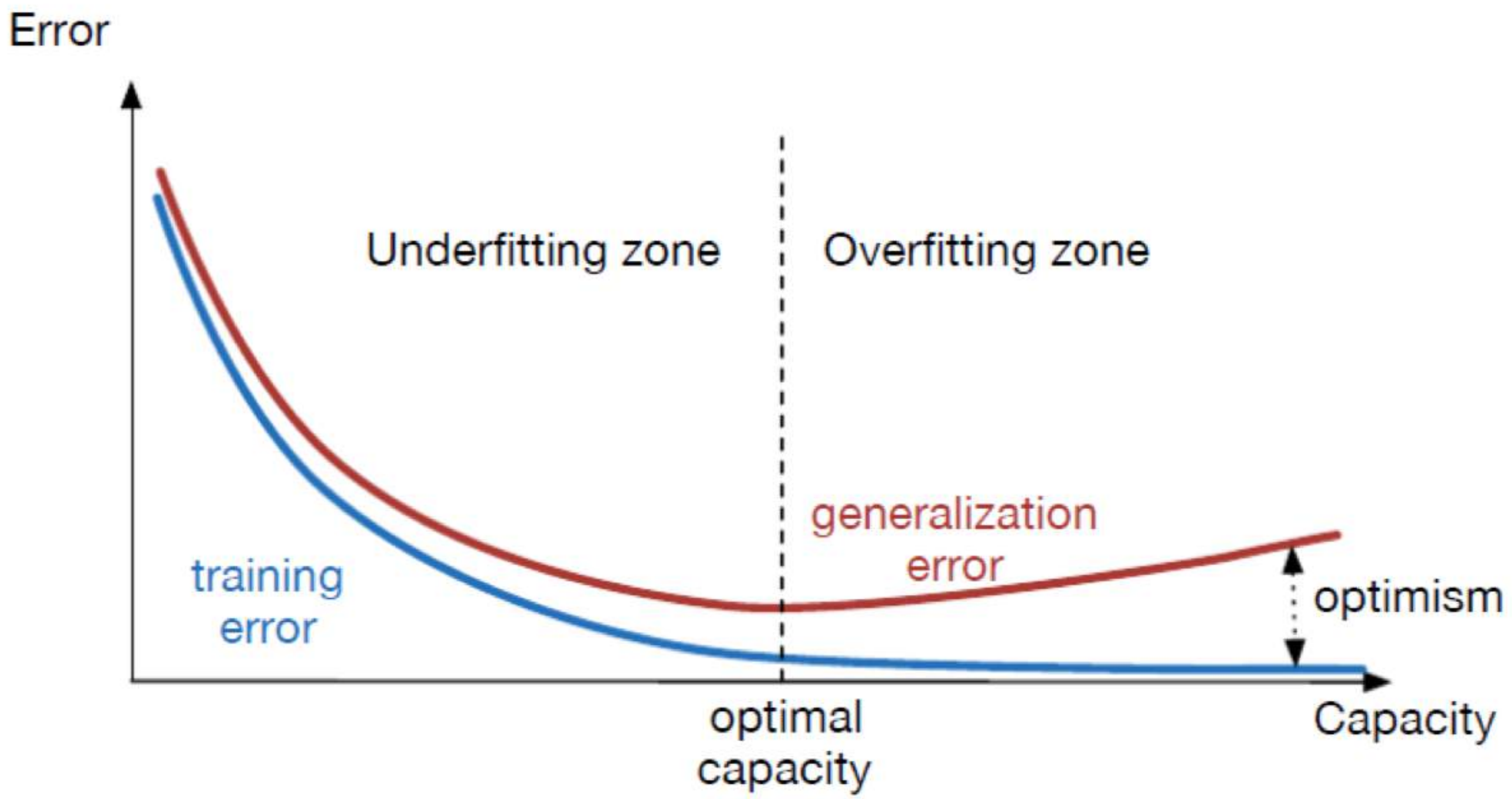
Overfitting/Underfitting



Overfitting/Underfitting



Overfitting/Underfitting



Overfitting/Underfitting

Underfitting = large bias & small variance

Overfitting = small bias & large variance

What is bias and variance?

Bias & Variance

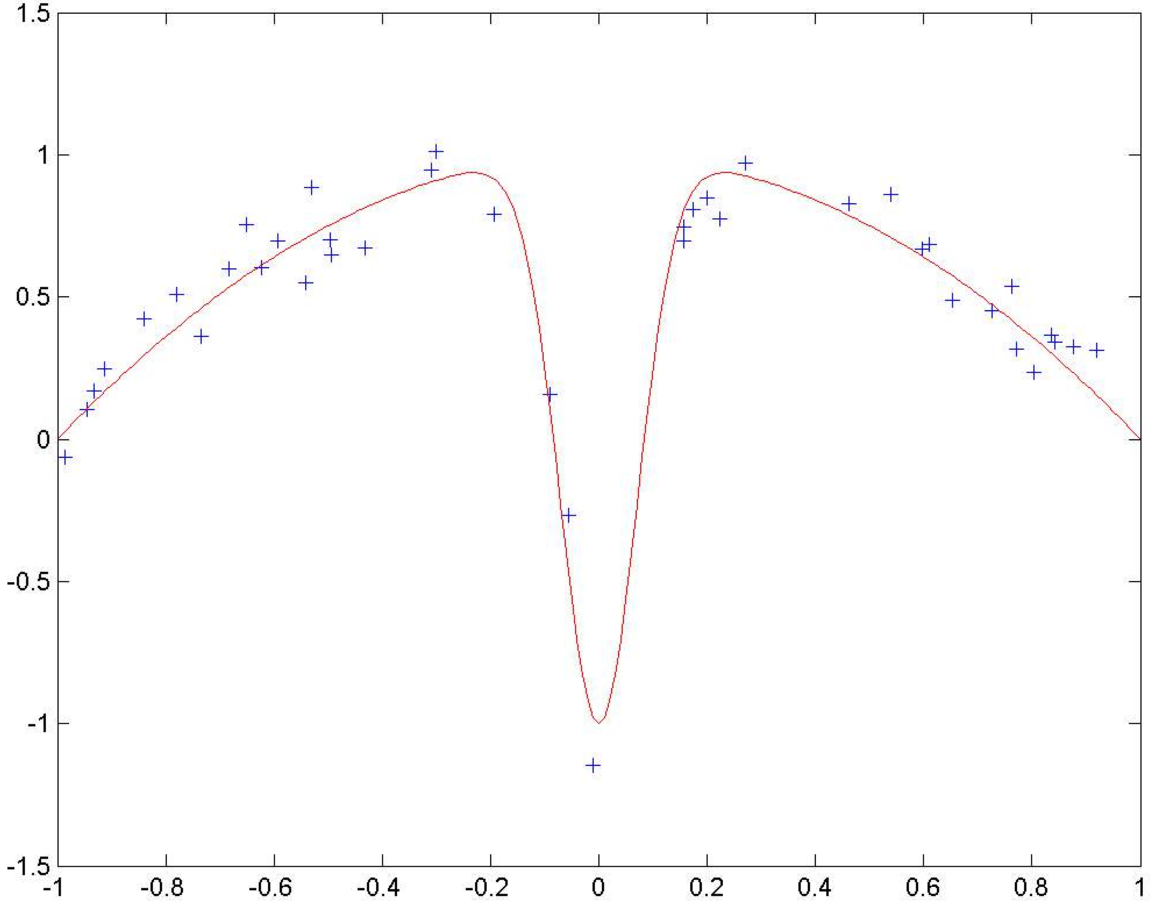
$$y = f(x) + \varepsilon;\ \varepsilon \sim q(0, \sigma^2) \qquad \hat{f}(x) \to f(x)$$

$$\mathrm{Bias}\left[\hat{f}(x)\right] = \mathrm{E}\left[\hat{f}(x)\right] - f(x)$$

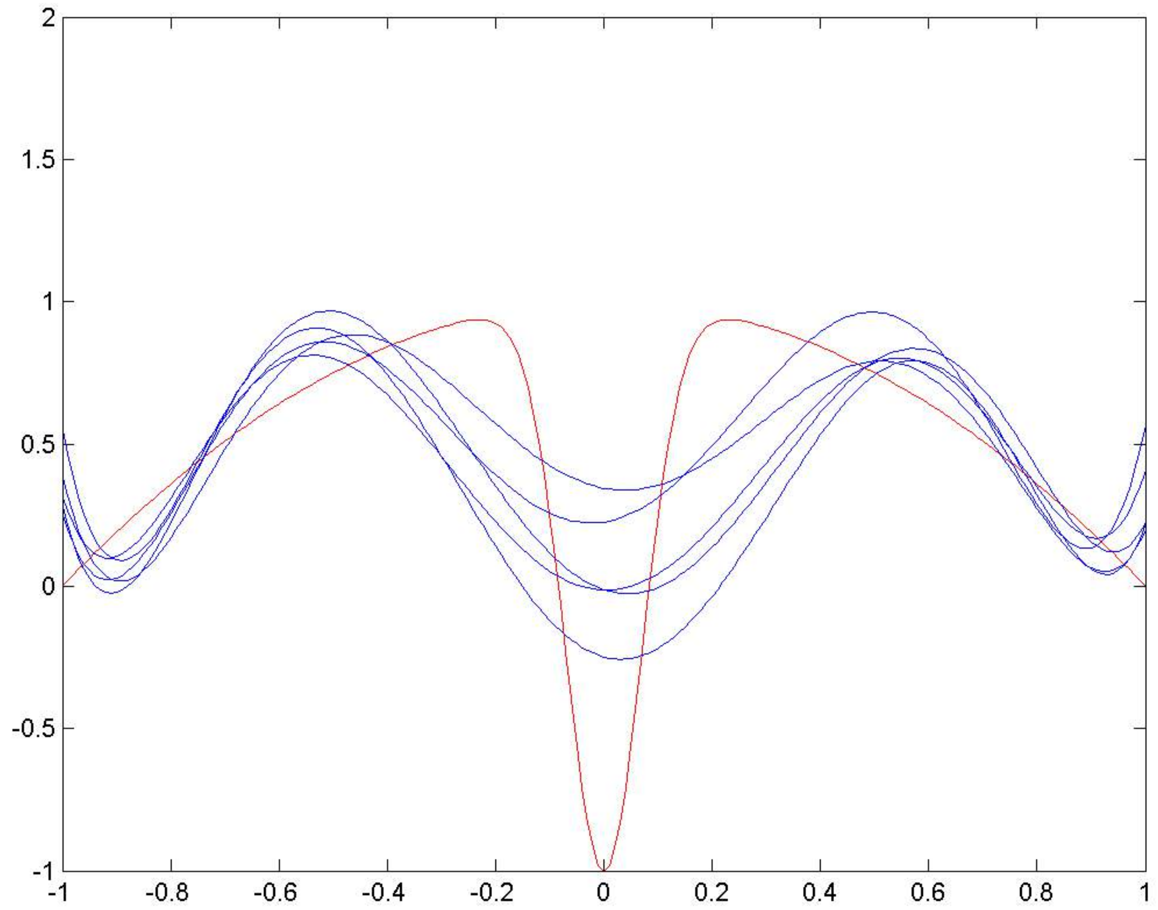
$$\mathrm{Var}\left[\hat{f}(x)\right] = \mathrm{E}[\hat{f}(x)^2] - \mathrm{E}[\hat{f}(x)]^2$$

Expectation taken over all possible samples from data-generating distribution

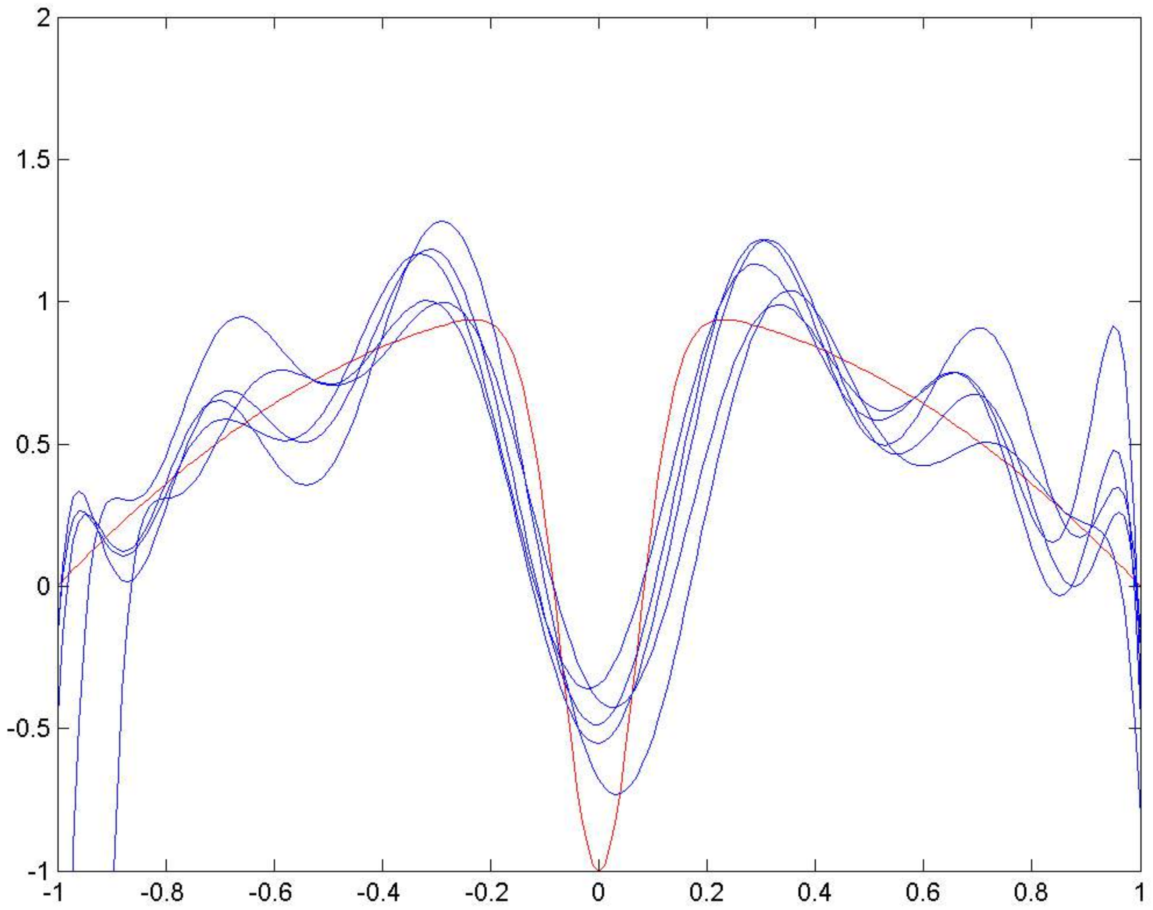
Bias & Variance



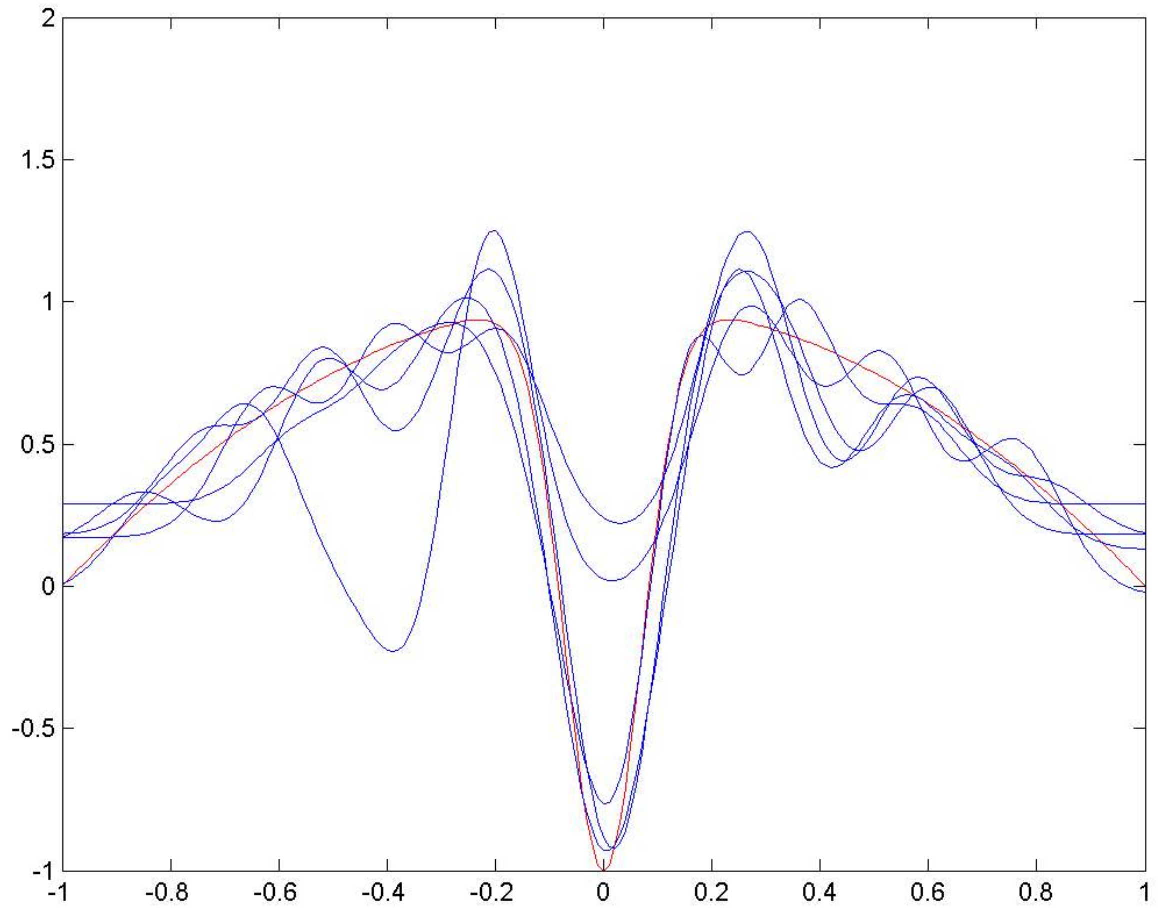
Bias & Variance



Bias & Variance



Bias & Variance



Total Error Decomposition

$$L(y, \hat{y}) \leq \quad approximation\ error$$
$$+\ estimation\ error$$
$$+\ optimization\ error$$

Bayes-optimal classifier/predictor over all predictors

$$f_o = \underset{f \in \widetilde{f f_o}}{\operatorname{argmin}} \int_{(x,y) \sim P(x,y)} l(x, y, f)$$

Approximation Error

$$f_o = \underset{f \in \widetilde{ff_o}}{\operatorname{argmin}} \int_{(x,y) \sim P(x,y)} l(x, y, f)$$

The class of predictors is usually restricted (e.g. to some parametric family)

$$\tilde{f} = \underset{f \in \widetilde{ff}}{\operatorname{argmin}} \int_{(x,y) \sim P(x,y)} l(x, y, f)$$

$||f_o - \tilde{f}||$ - approximation error

Estimation Error

$$\tilde{f} = \underset{f \in \widetilde{ff}}{\operatorname{argmin}} \int_{(x,y)\sim P(x,y)} l(x, y, f)$$

Cannot integrate, have a finite sample of data

$$\hat{f} = \underset{f \in \widetilde{ff}}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} \big(l(x, y, f)\big) + \lambda R(f)$$

$||\tilde{f} - \hat{f}||$ - estimation error

Estimation Error

$$\hat{f} = \underset{f \in \widetilde{f} f}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} \big(l(x, y, f)\big) + \lambda R(f)$$

Have a limited amount of time to optimize

$$\hat{f}^{(t)} = \underset{f \in \widetilde{f} f}{\operatorname{argmin}}^{(t)} \frac{1}{N} \sum_{i=1}^{N} \big(l(x, y, f)\big) + \lambda R(f)$$

$||\hat{f} - \hat{f}^{(t)}||$ - optimization error

Total Error Decomposition

$$L(y,\hat{y}) \leq \quad approximation\ error$$
$$+\ estimation\ error$$
$$+\ optimization\ error$$

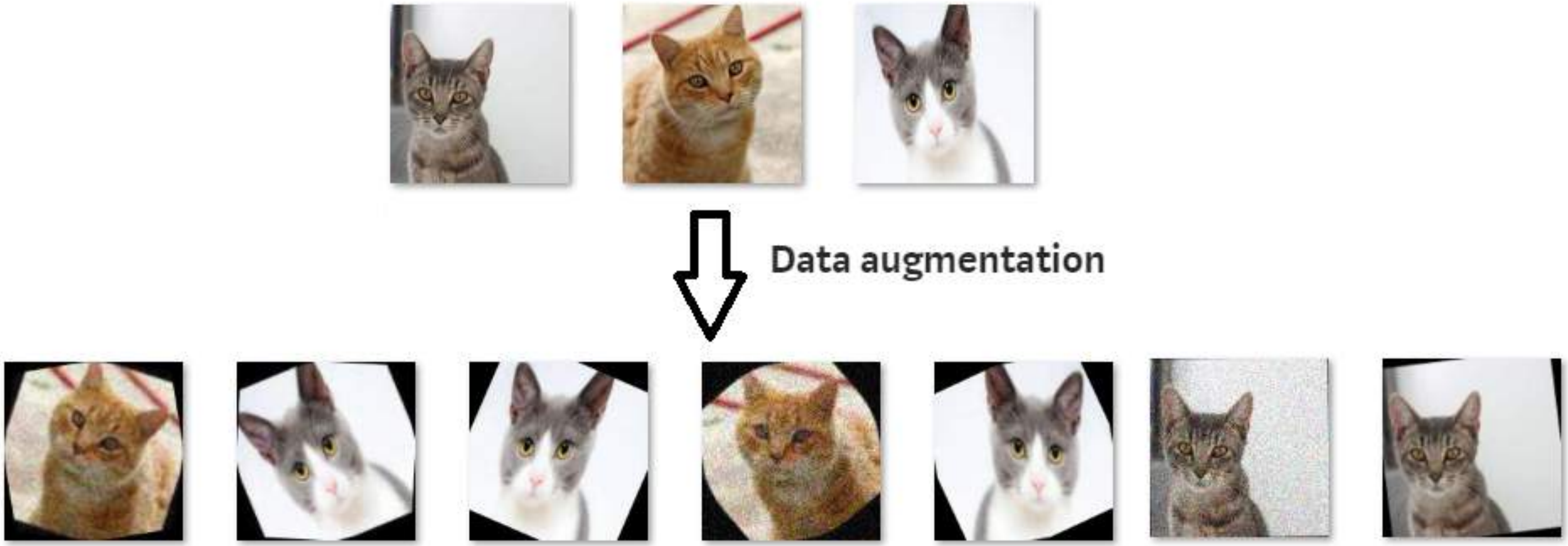
$$\hat{f}^{(t)} = \underset{f \in \widetilde{f} f}{\operatorname{argmin}}^{(t)} \frac{1}{N} \sum_{i=1}^{N} \big(l(x, y, f) \big) + \lambda R(f)$$

Recap: Regularization

Regularization Types

1. Model regularization
    - Dropout, Stochastic Depth etc.
    - L1, L2 (weight decay)
    - BatchNorm
2. Data regularization (augmentation)
3. Optimization regularization
    - SGD instead of Adam
    - Learning rate/batch size tradeoff
4. Ensembling

Data Augmentation



Data Augmentation

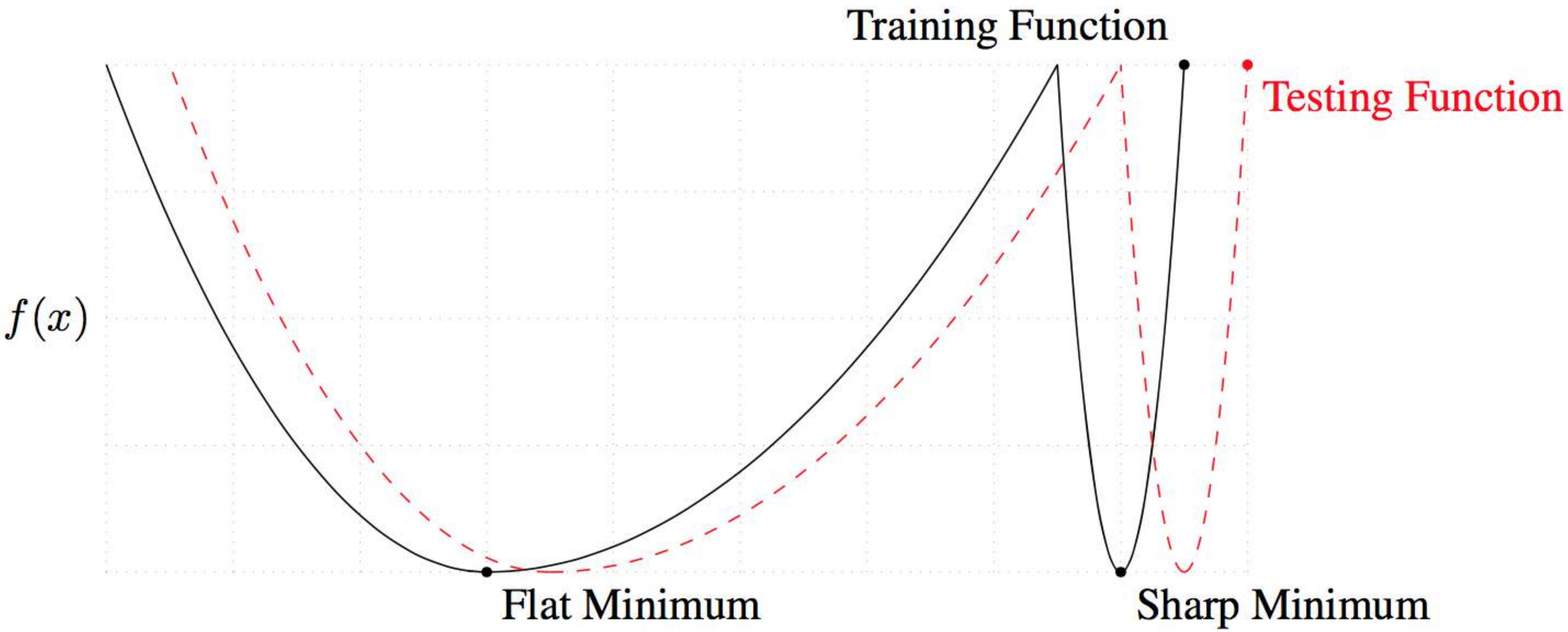
Label = Jeff Bezos

Label = Geoffrey Hinton

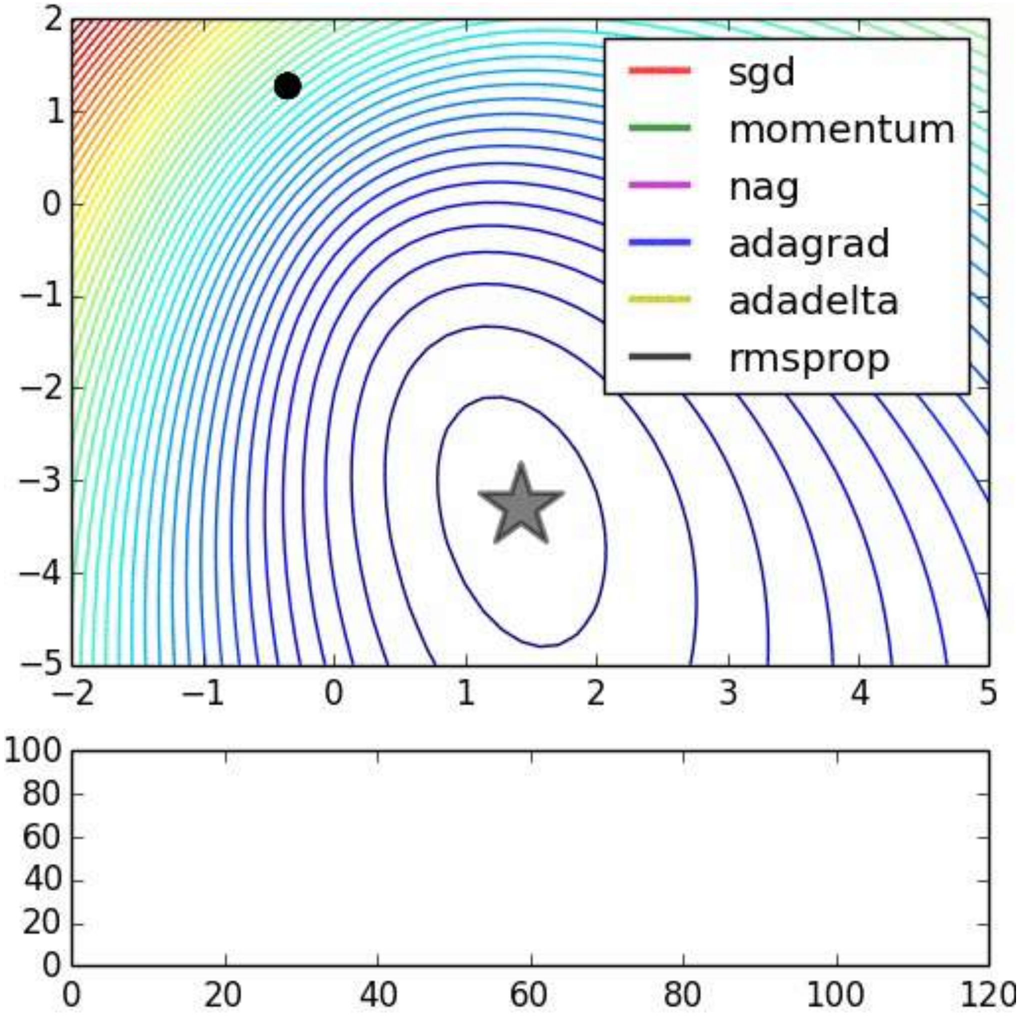
Data Augmentation

Label = 0.5 * Jeff Bezos & 0.5 * Geoffrey Hinton

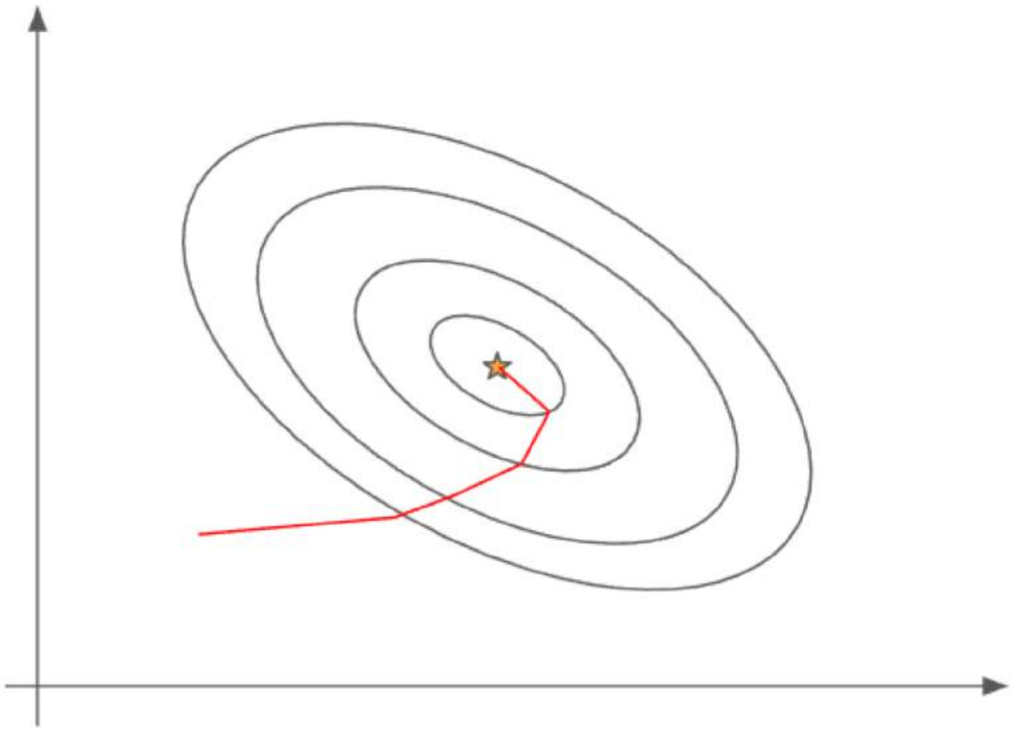
Optimization Regularization



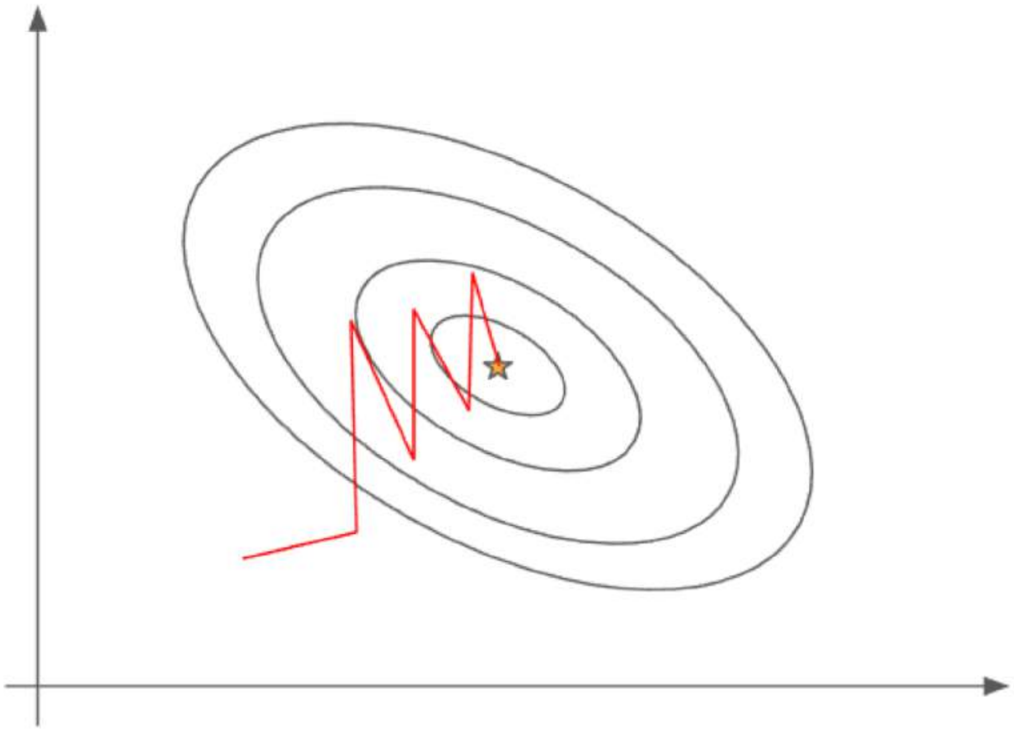
SGD instead of Adam



Learning Rate/Batch Size Tradeoff

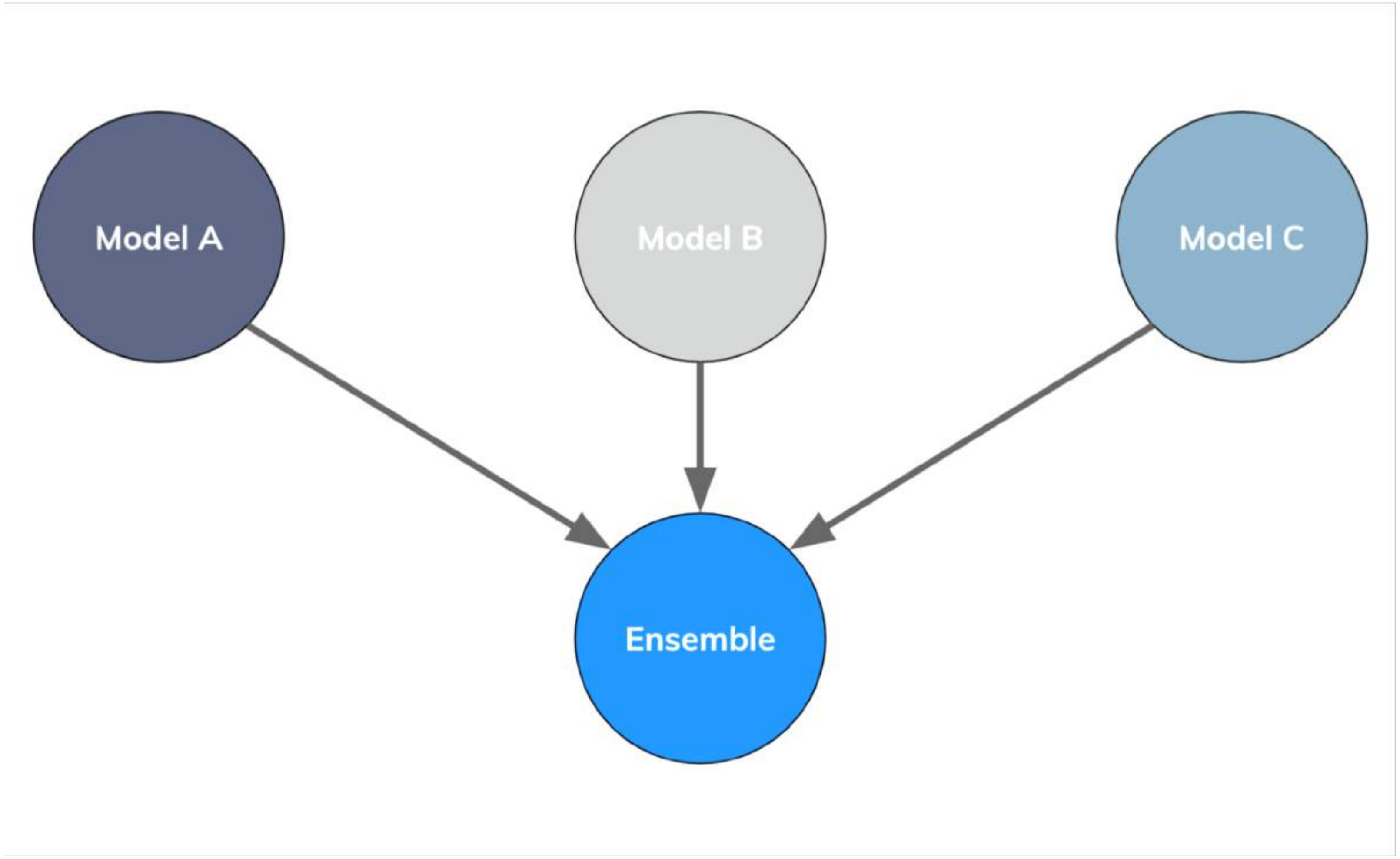


Gradient Descent

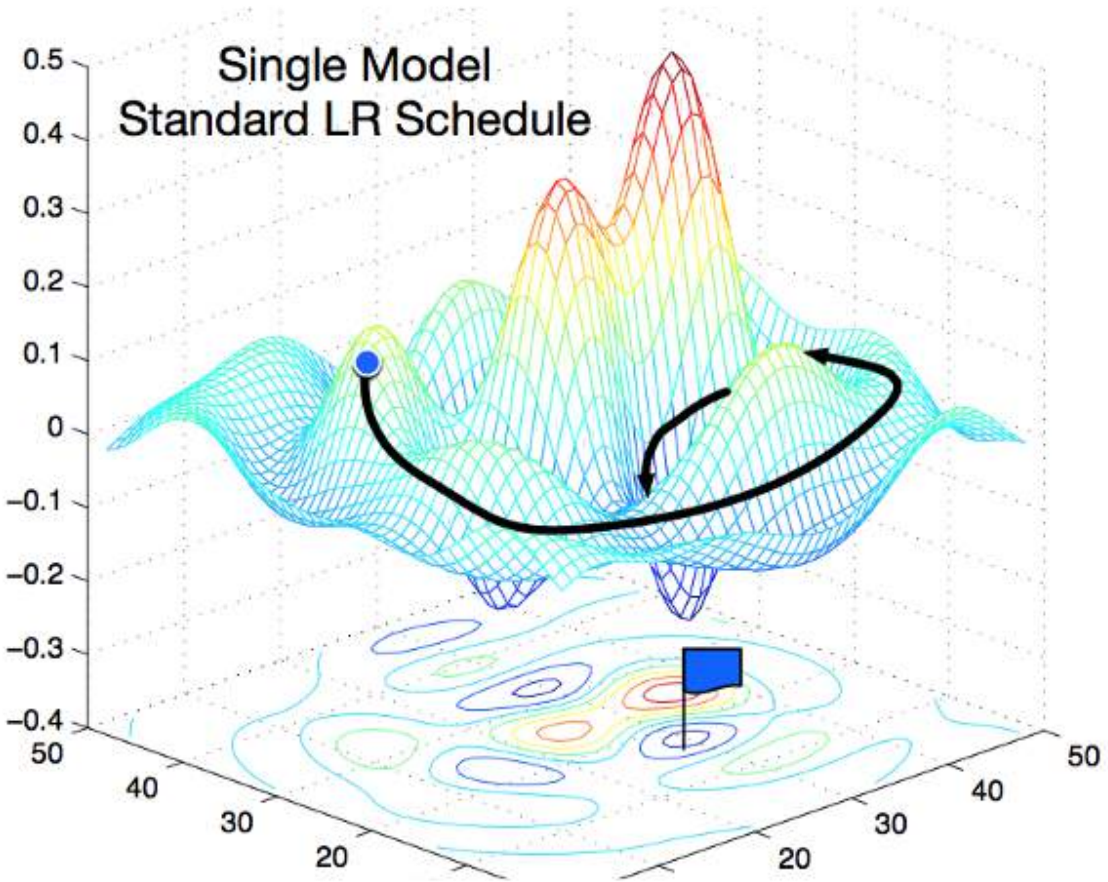


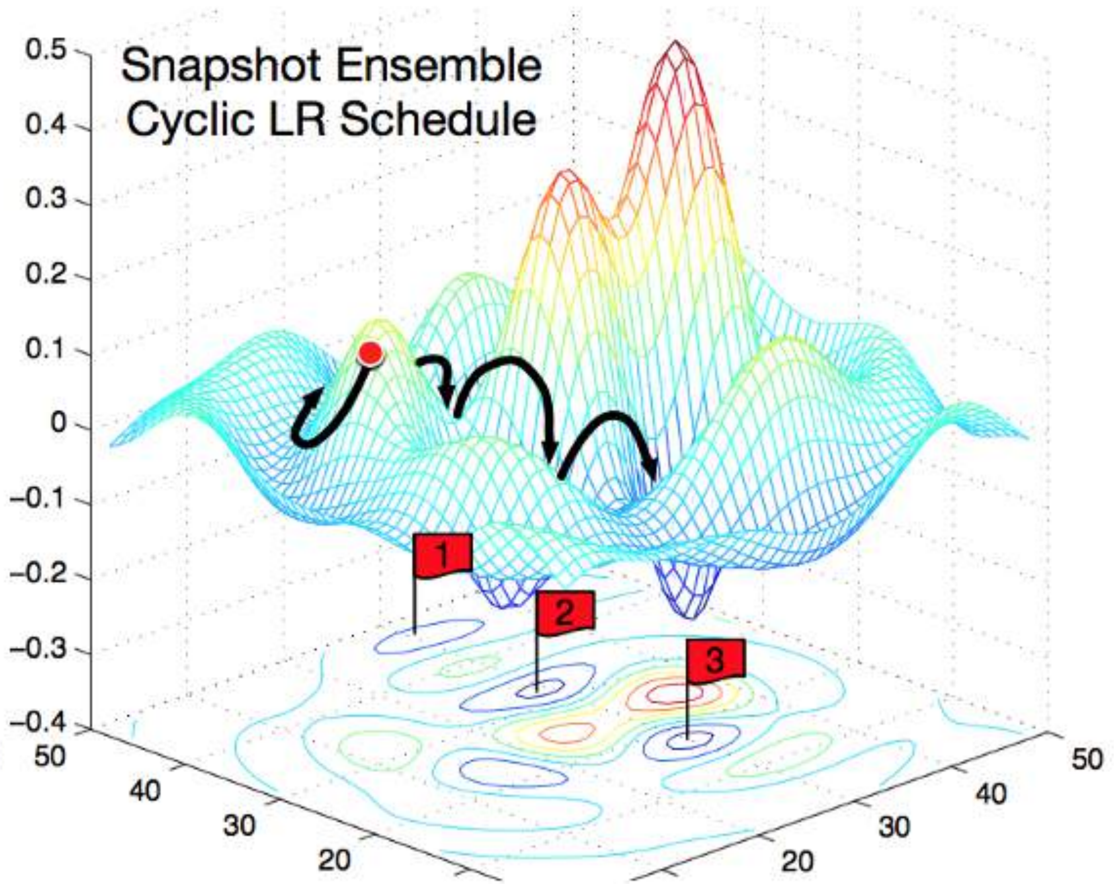
Stochastic Gradient Descent

Ensembling

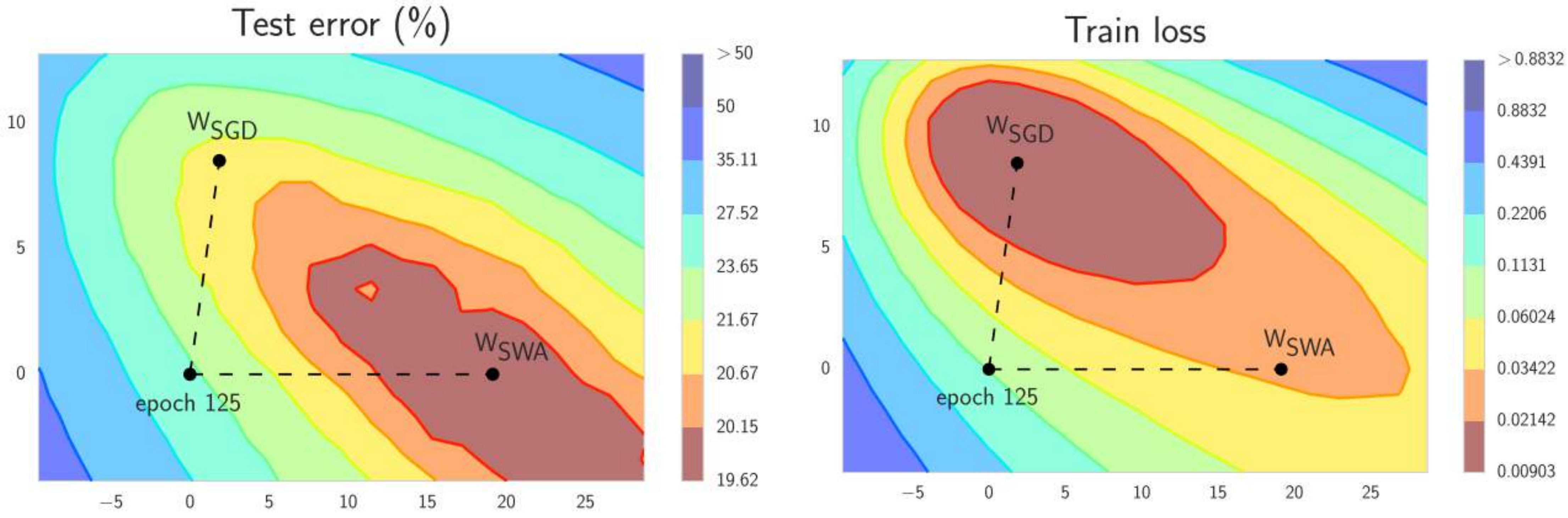


Ensembling: Stochastic Weight Averaging





Ensembling: Stochastic Weight Averaging



Ensembling: Stochastic Weight Averaging

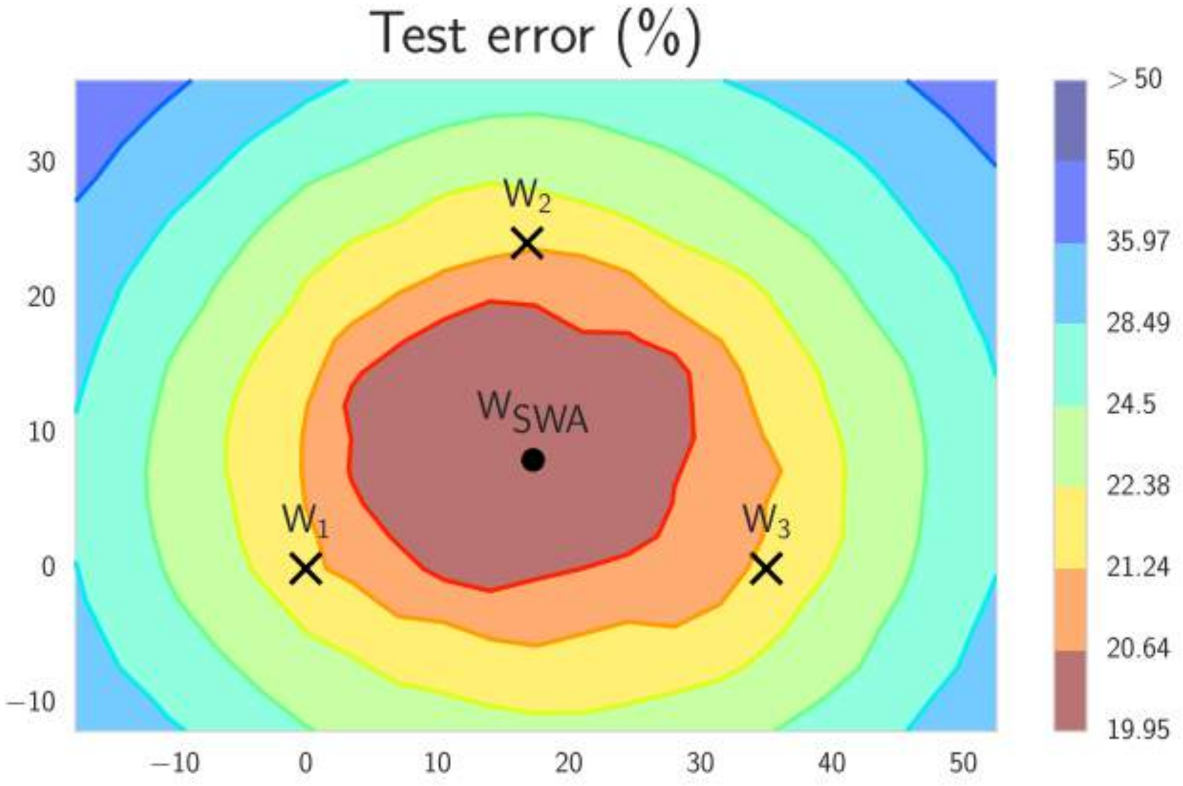
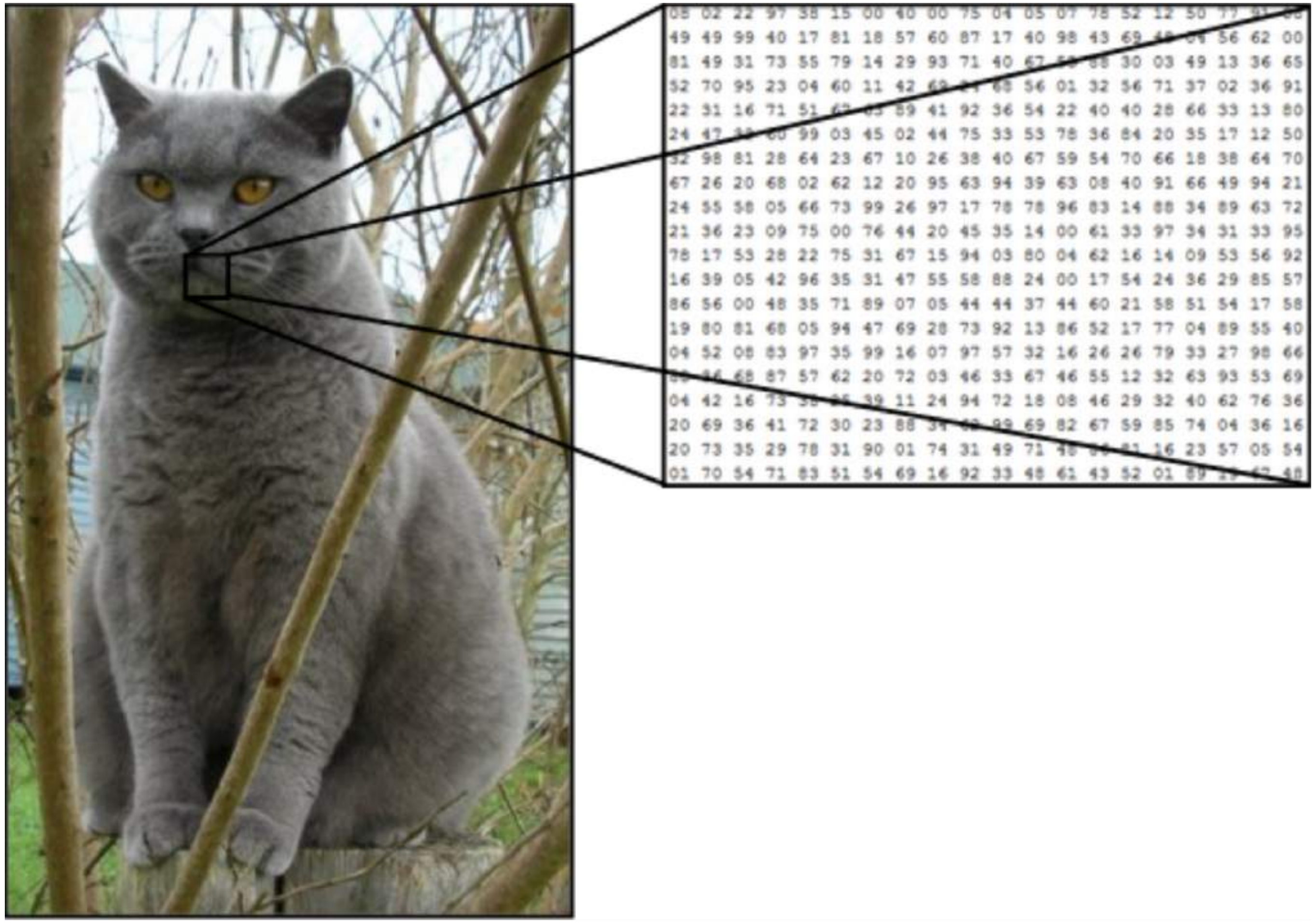


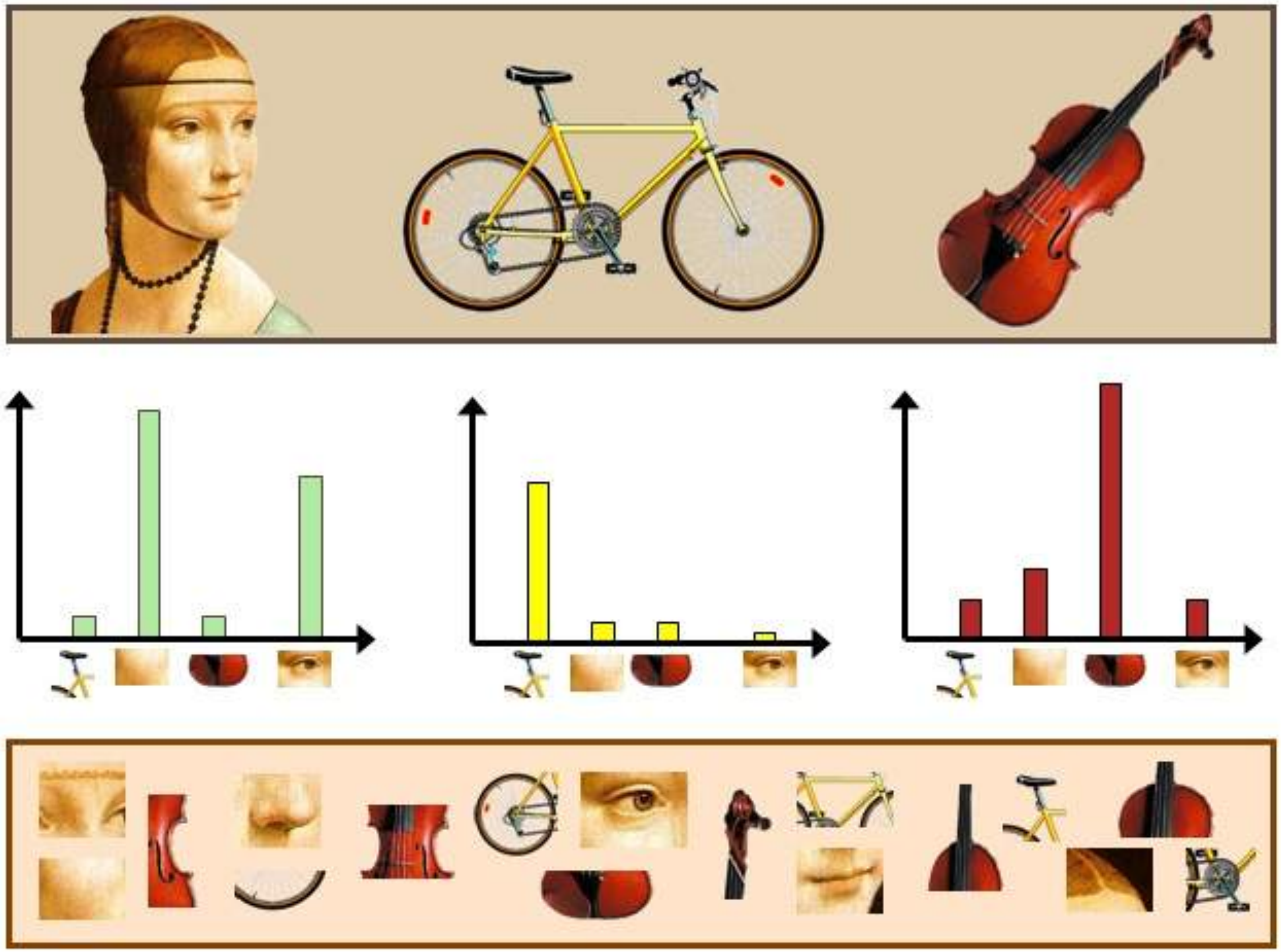
Image Classification

Computer View

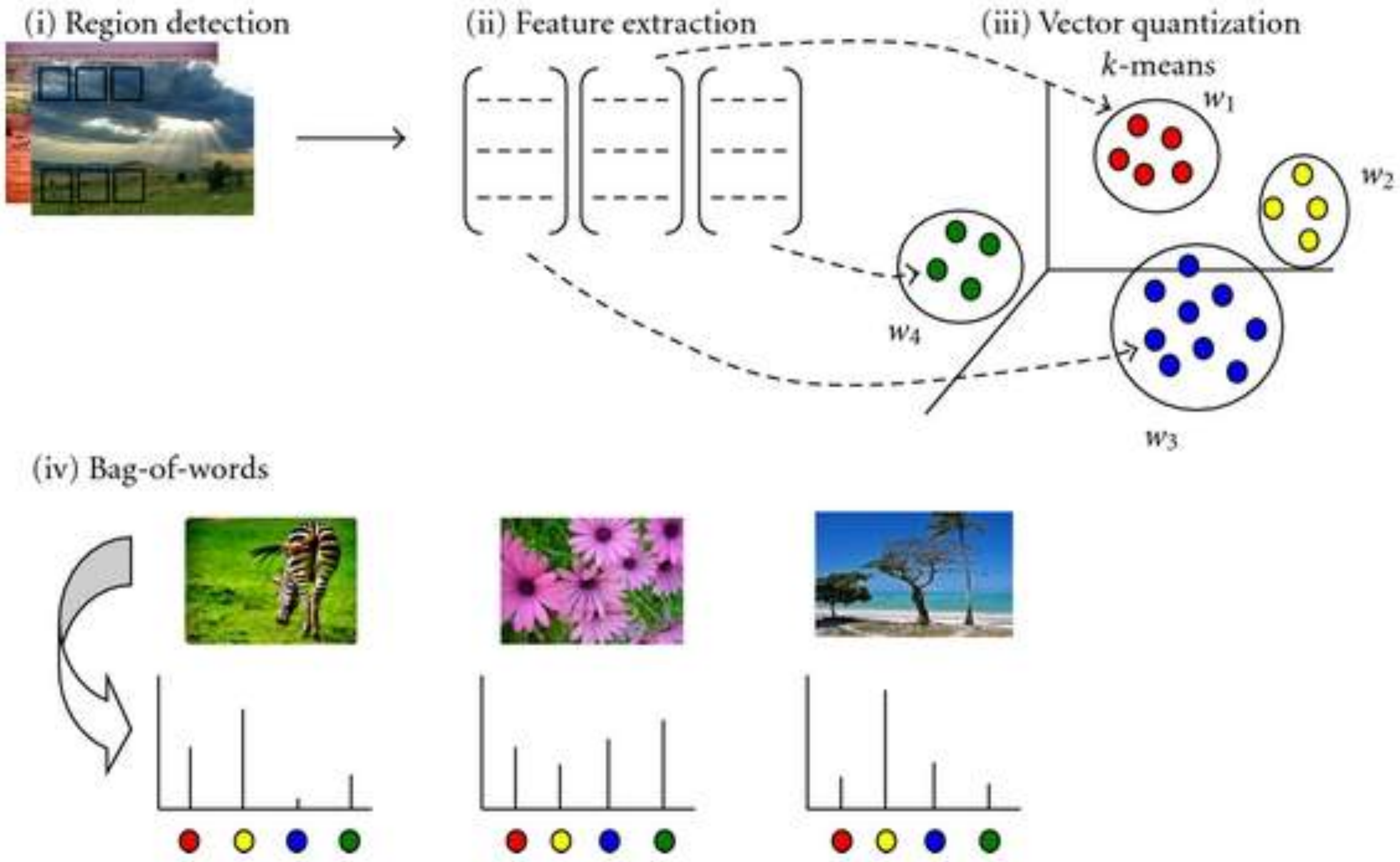


Human View

Bag of Visual Words

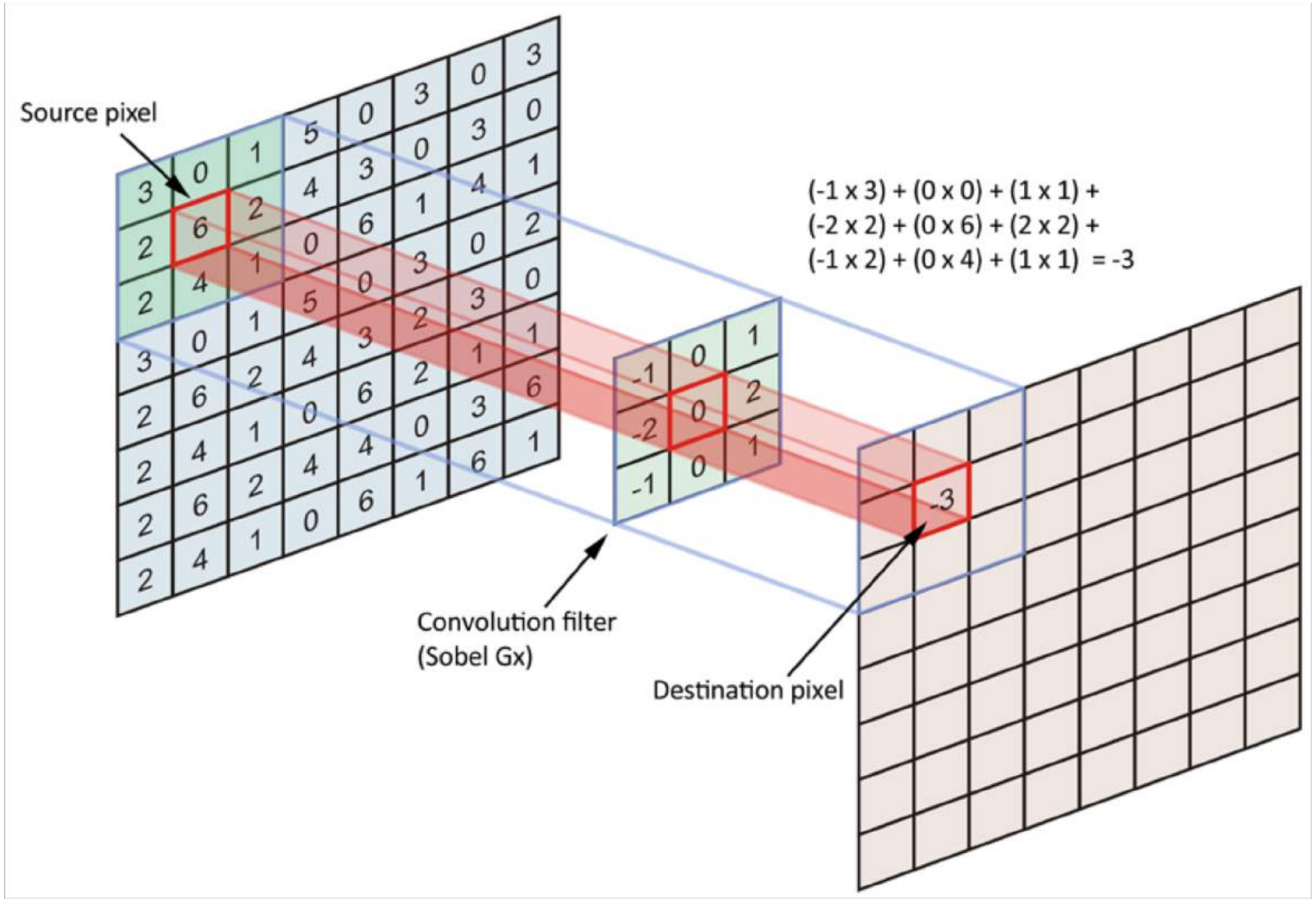


Bag of Visual Words

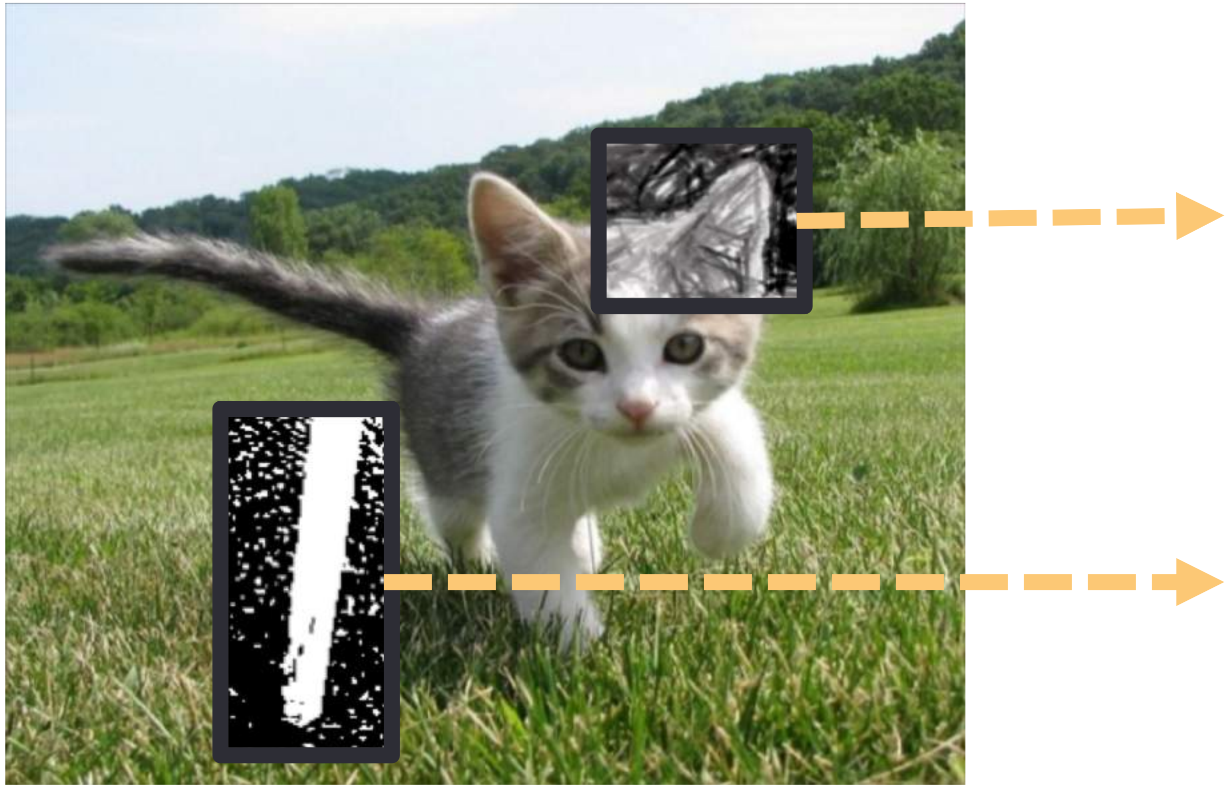


Recap CNN

Convolution



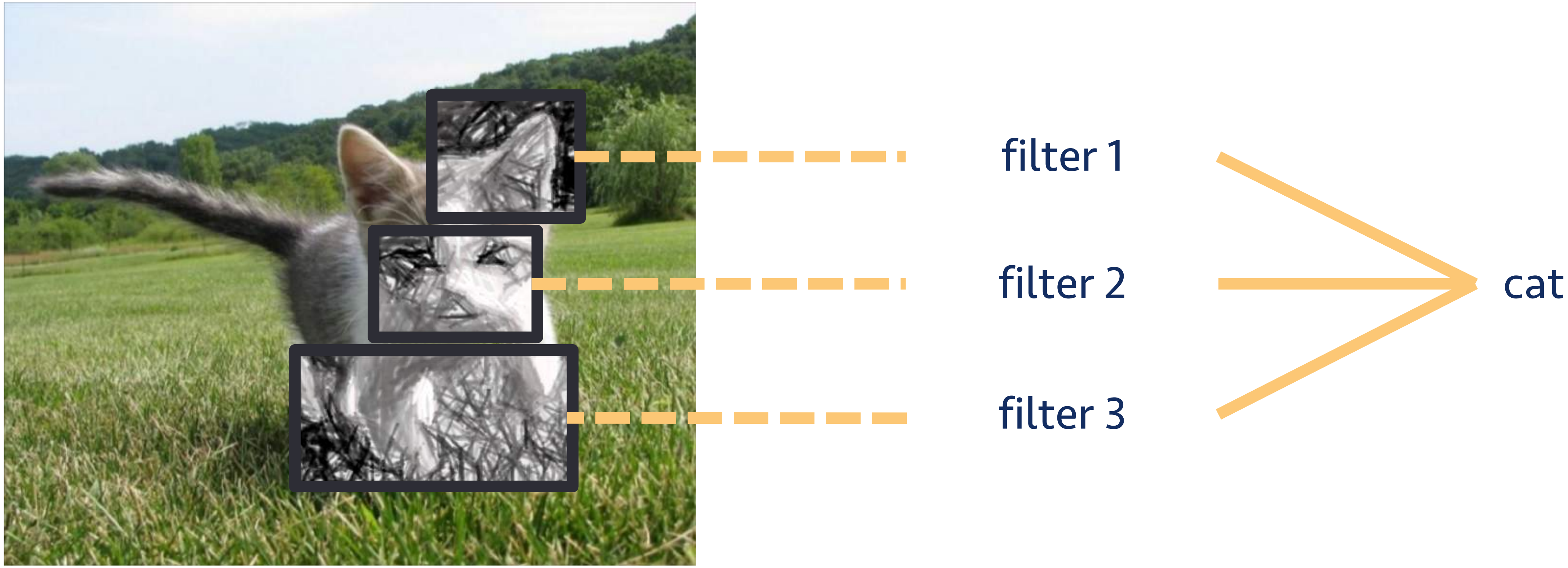
Convolution



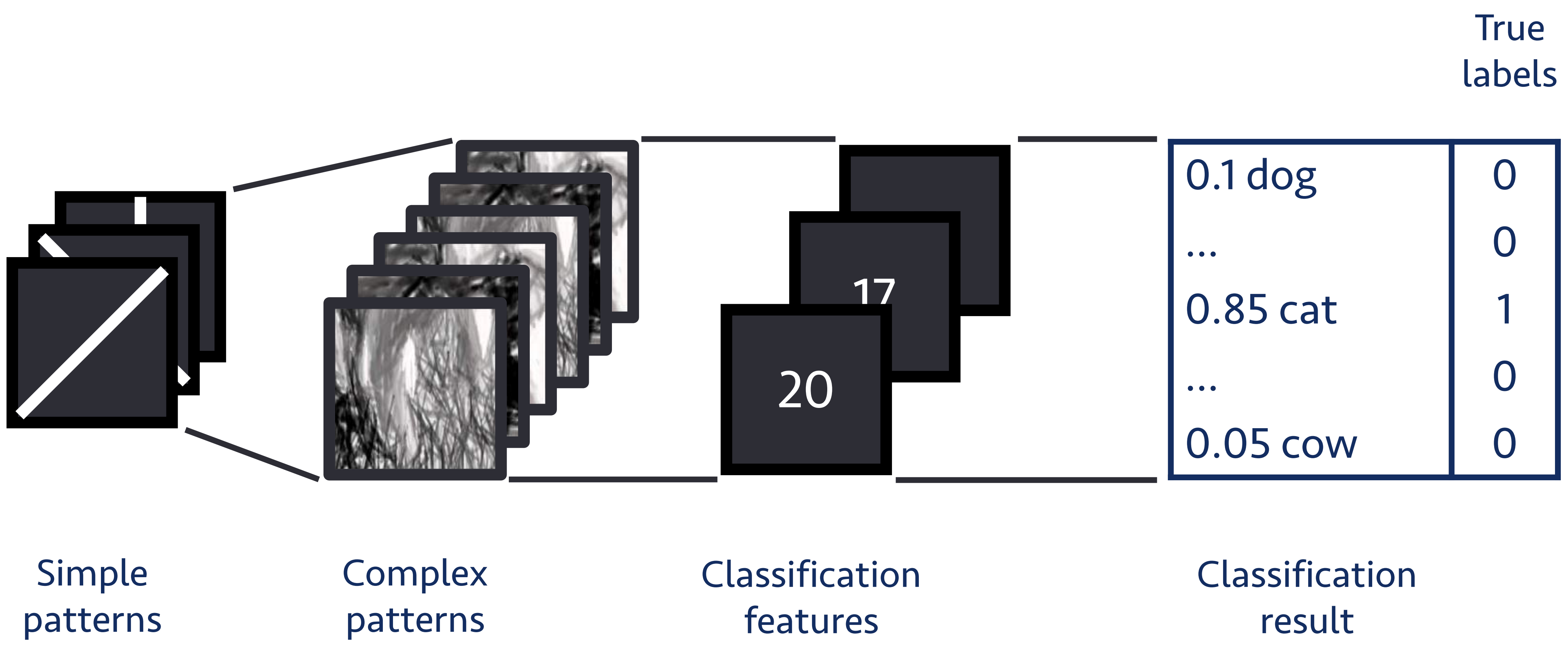
100 (sharp ear present)

0 (chair leg is absent)

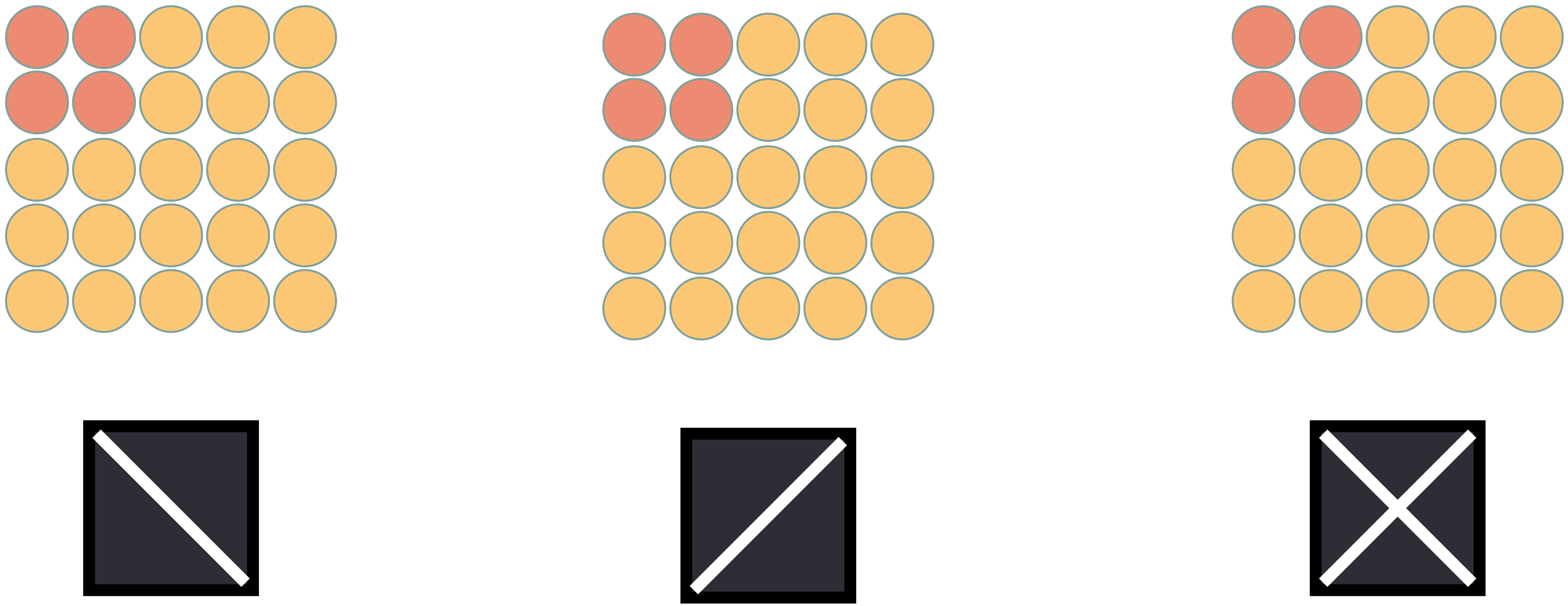
Convolution



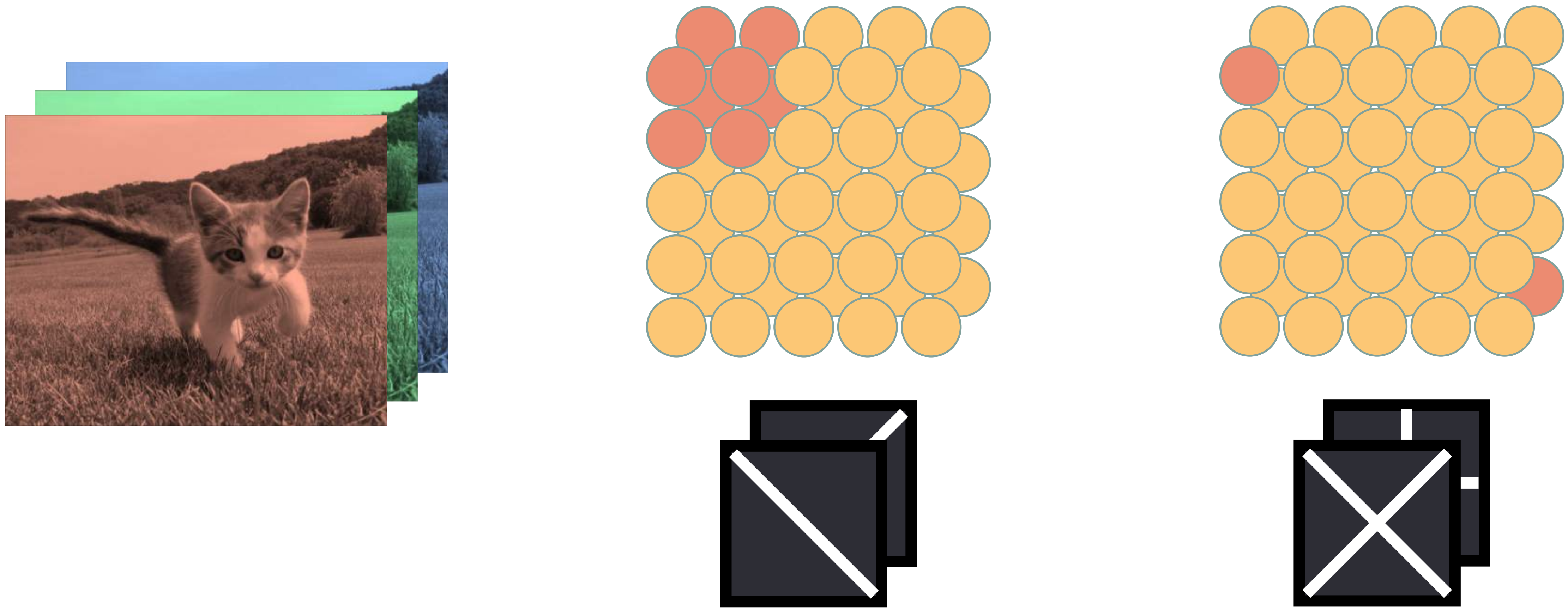
Stacked Convolutions



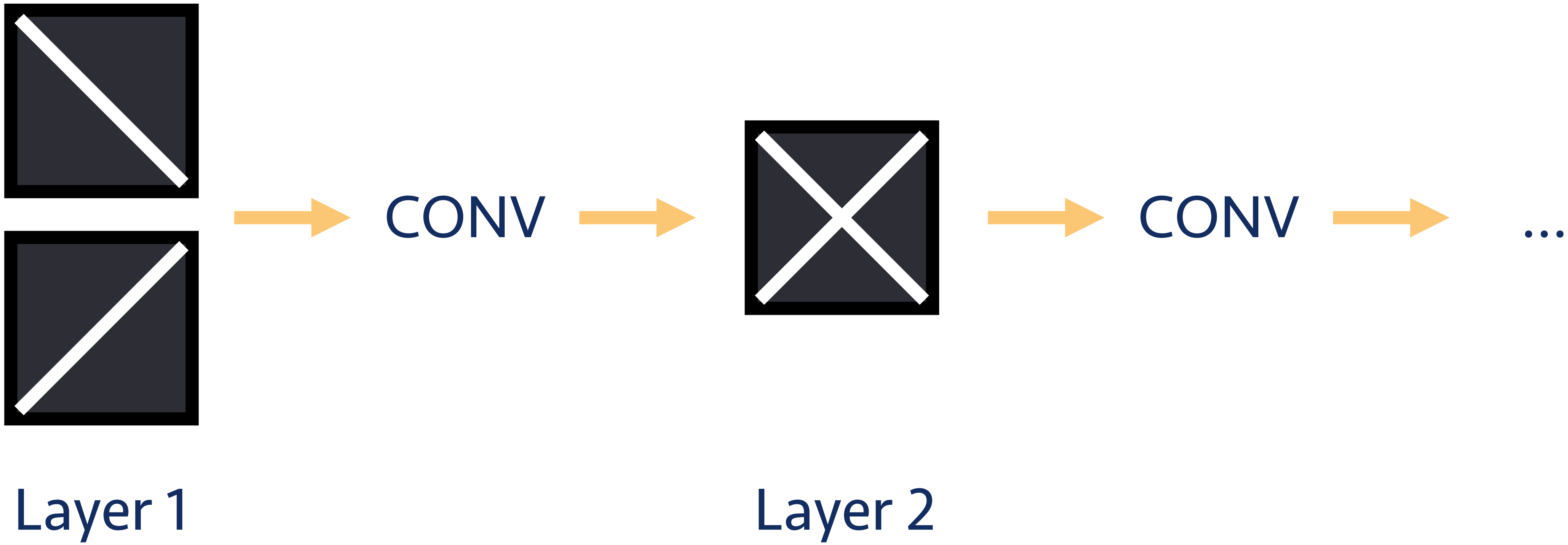
From Simple to Complex Patterns



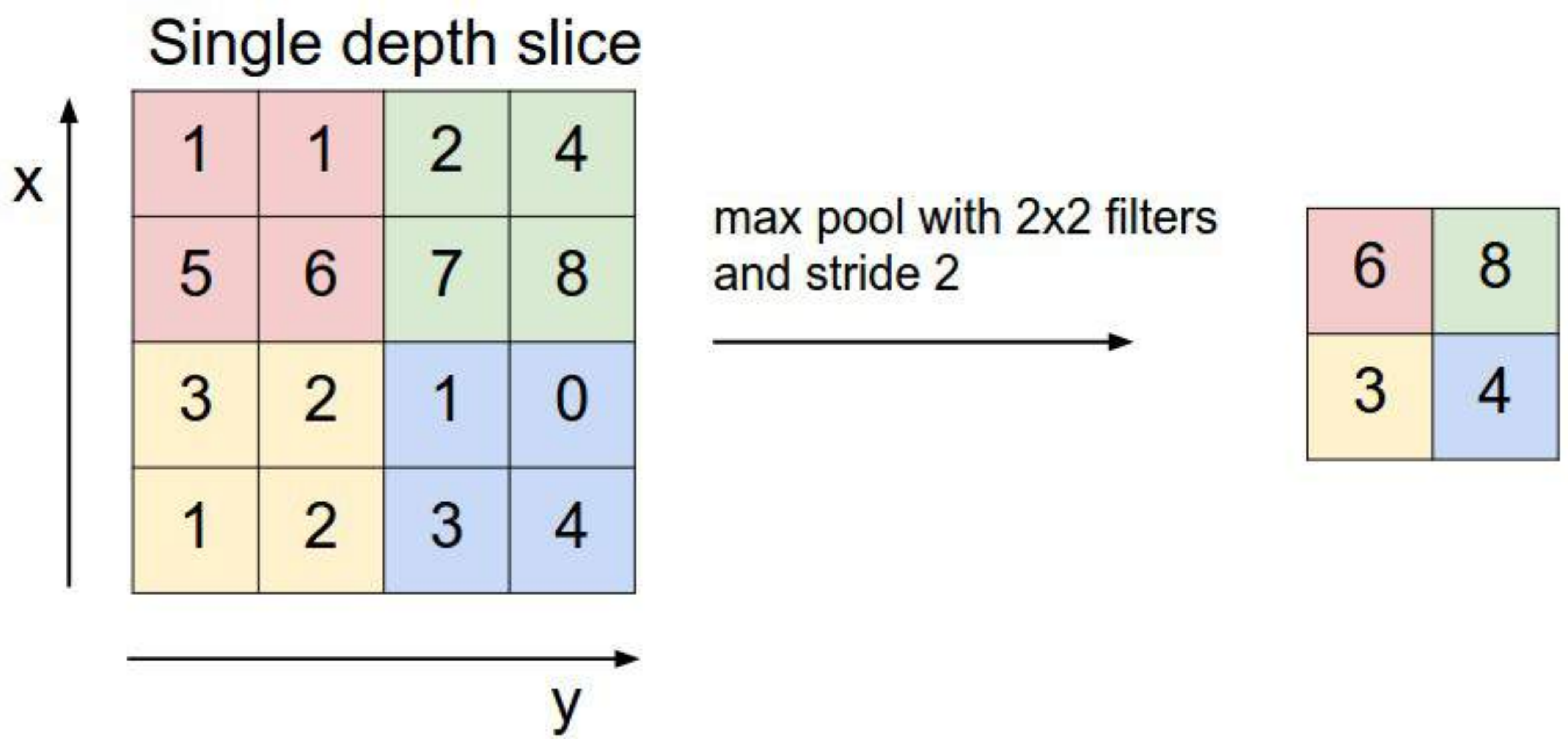
From Simple to Complex Patterns



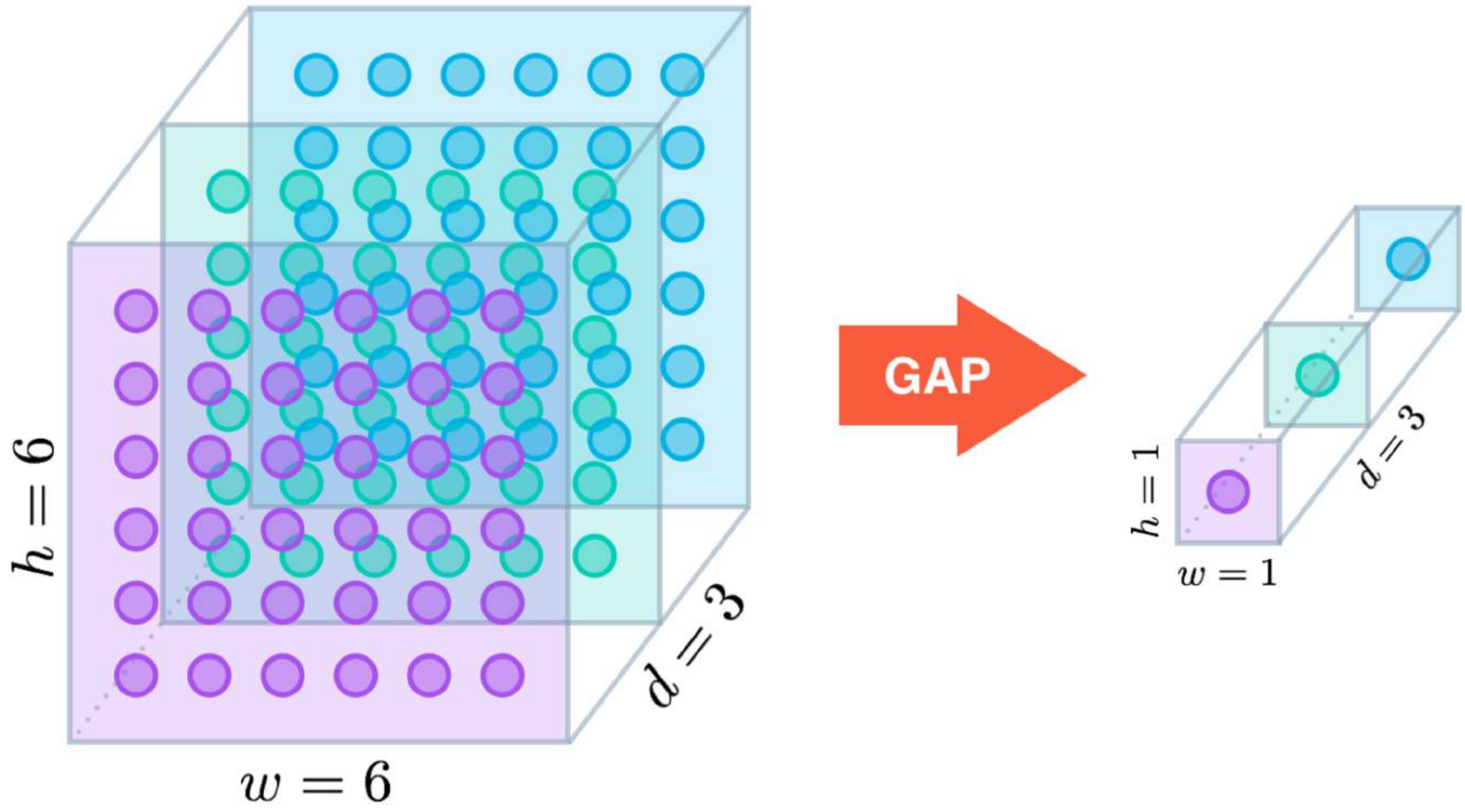
From Simple to Complex Patterns



Pooling


Single depth slice
x
y
1 1 2 4
5 6 7 8
3 2 1 0
1 2 3 4
max pool with 2x2 filters
and stride 2
6 8
3 4

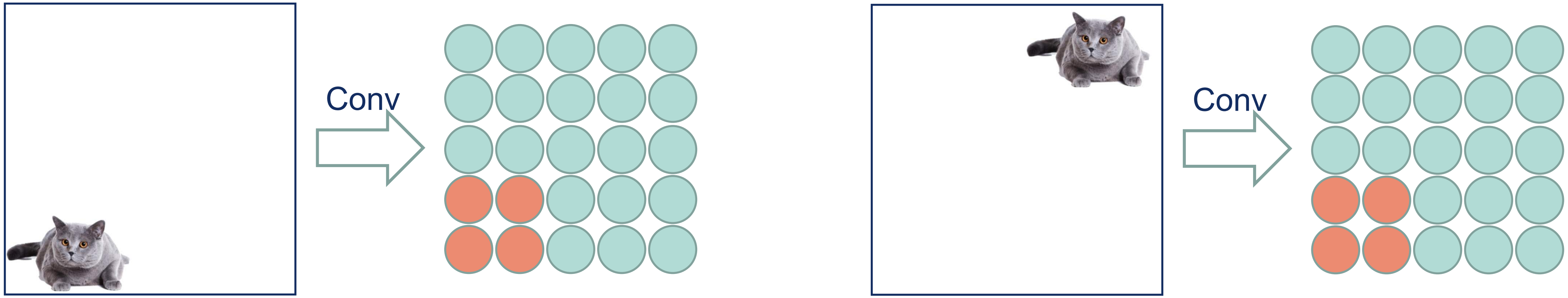
Global Pooling



CNN Problems

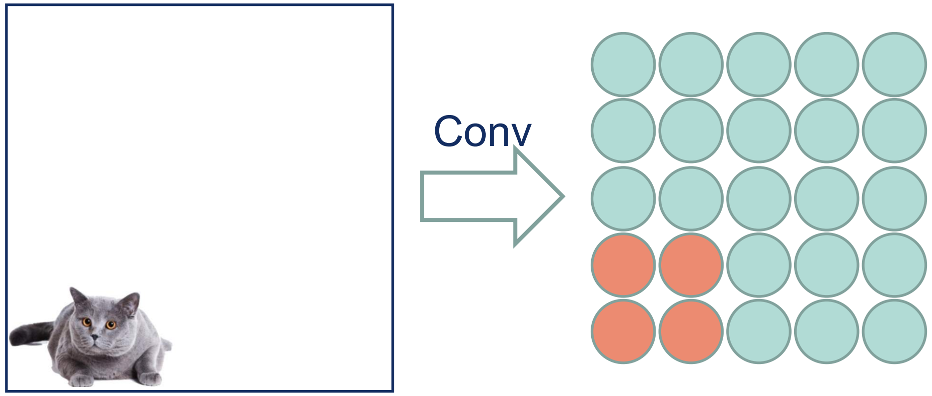
Invariance

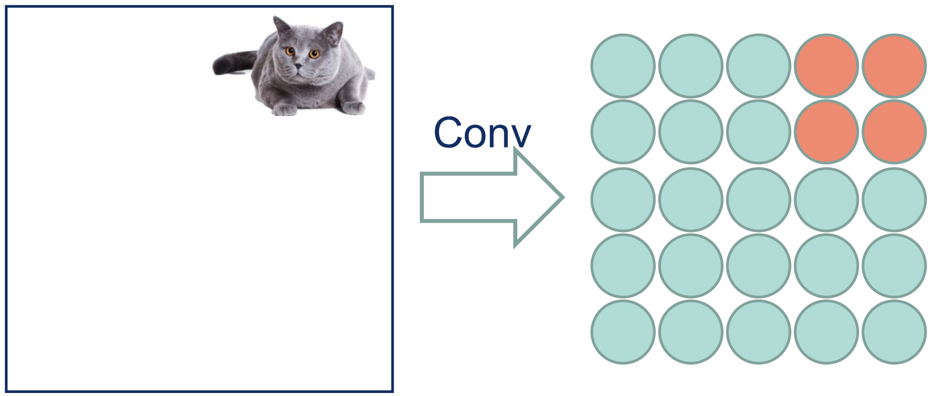
$$f\big(T(x)\big) = f(x)$$



Equivariance

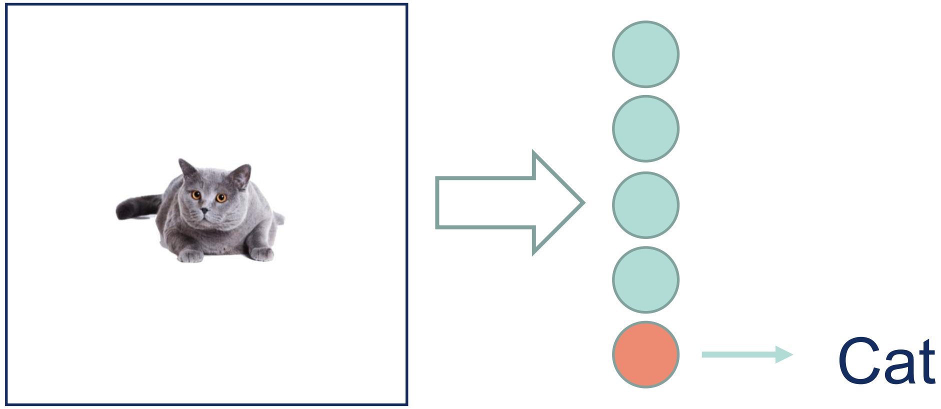
$$f\big(T(x)\big) = T(f(x))$$



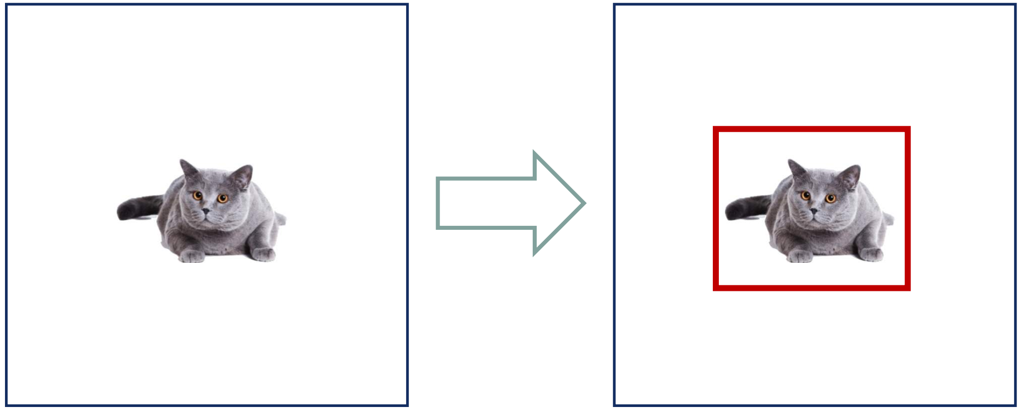


Why Invariance/Equivariance?

To classify
you need invariance

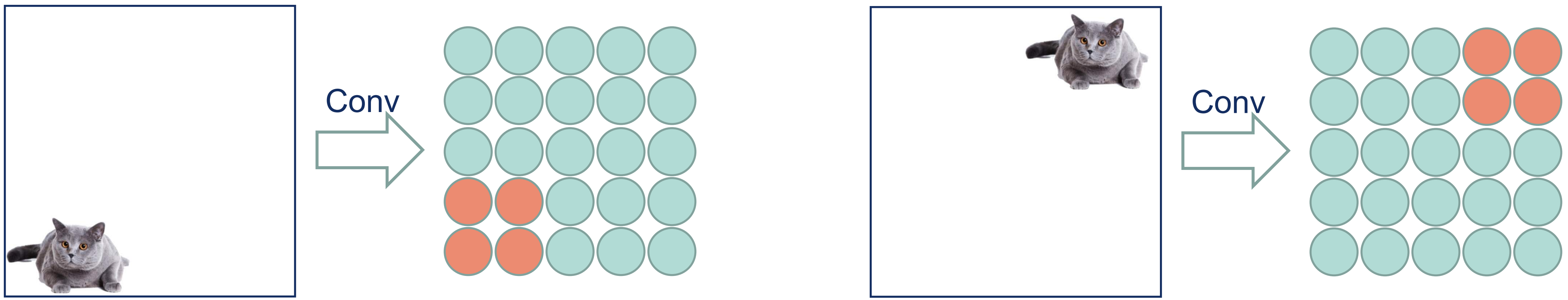


To detect
you need equivariance



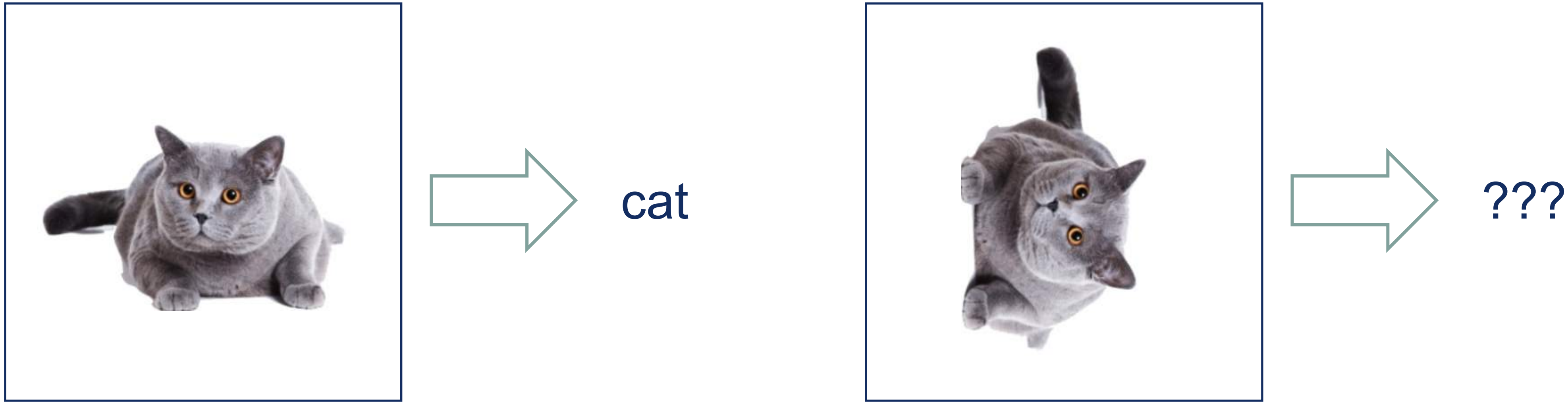
Convolution

Convolution is equivariant to shifts



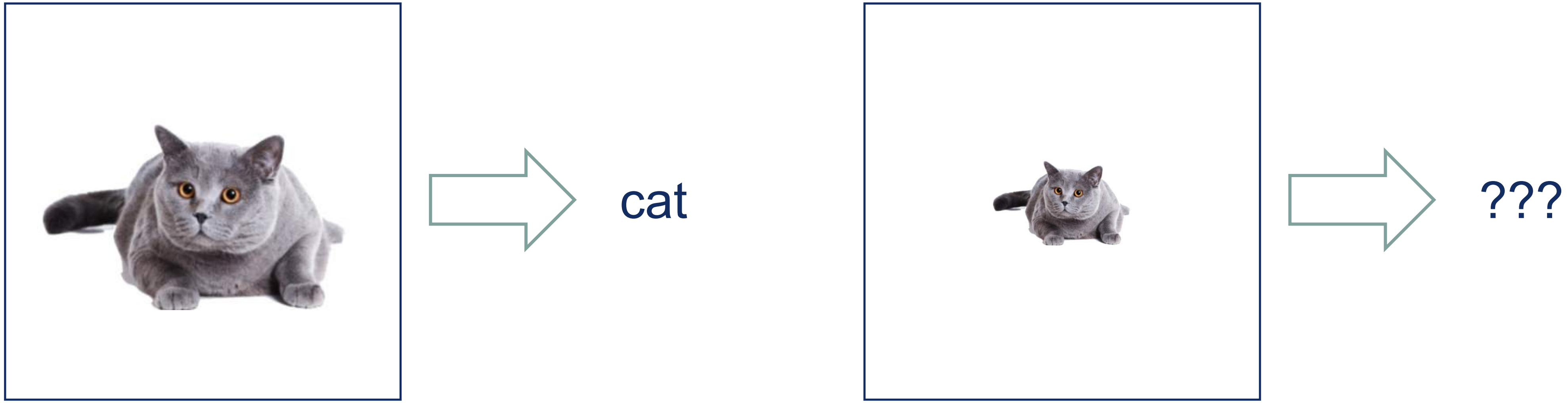
Convolution

Convolution is neither equivariant nor invariant to rotation



Convolution

Convolution is neither equivariant nor invariant to scale



Convolutional Neural Network

CNN is:

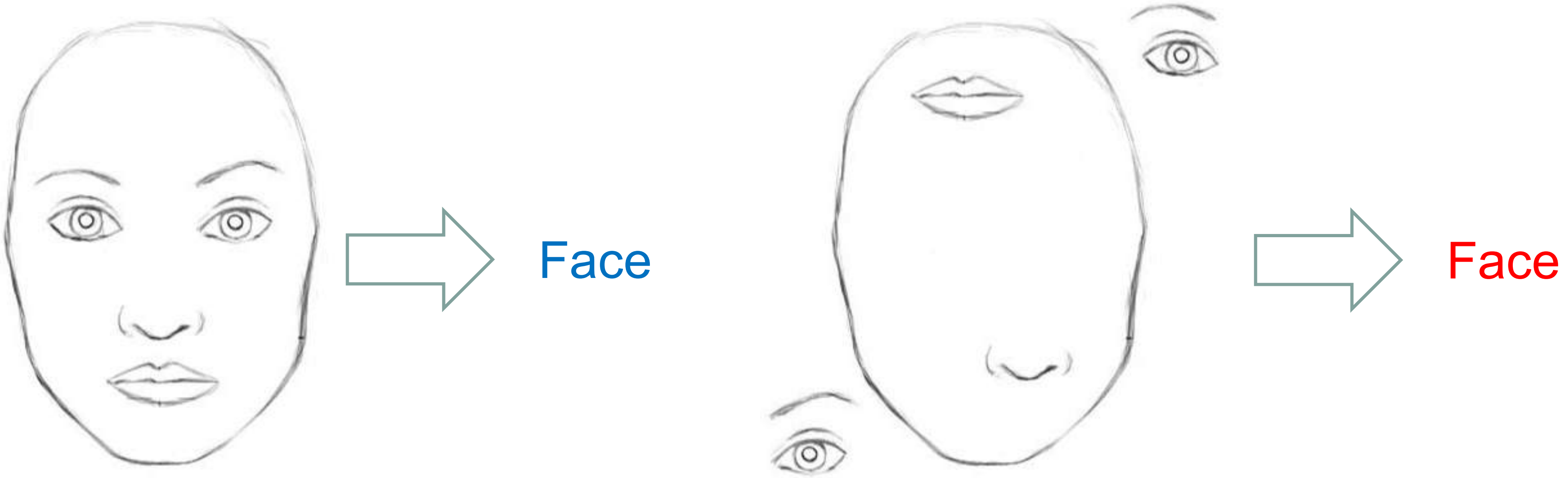
- invariant to shifts
- neither equivariant nor invariant to rotation
- neither equivariant nor invariant to scale

How to make CNN invariant to scale and rotation?

Use data augmentation

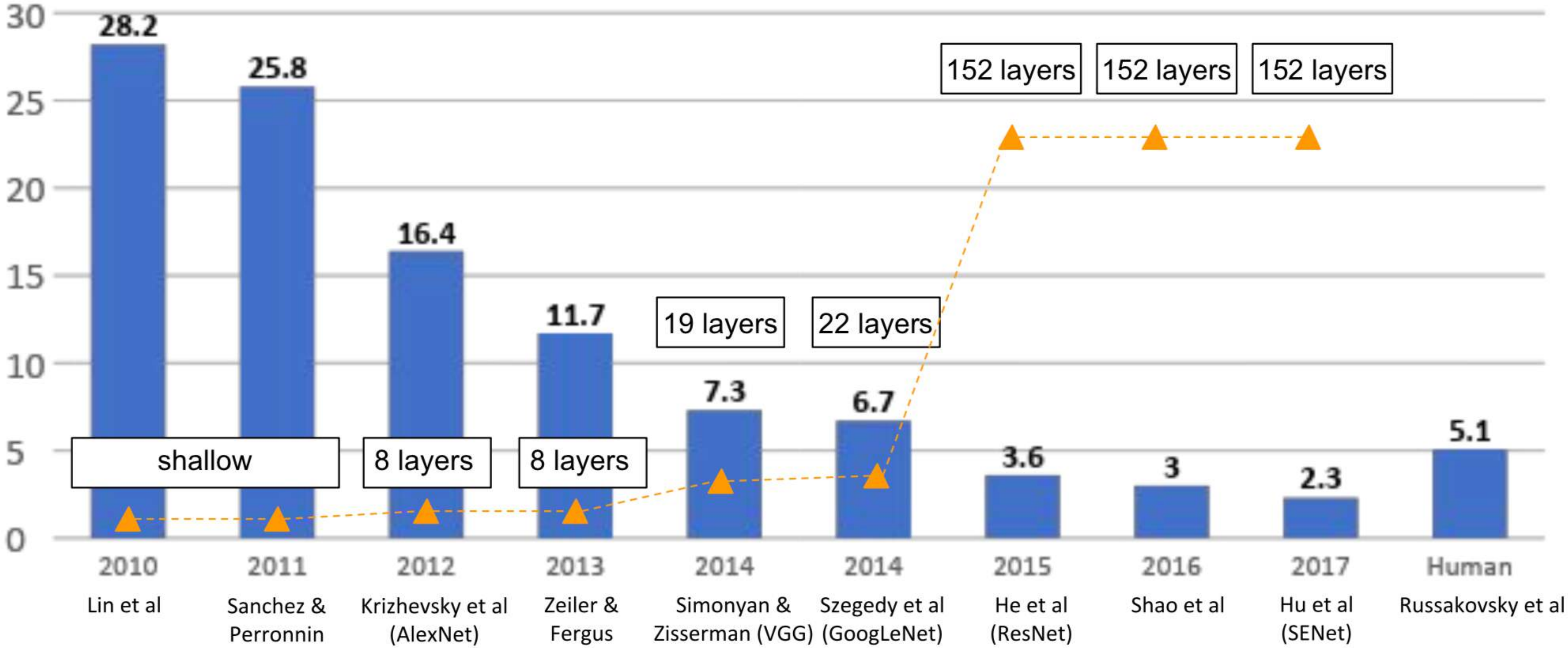
Convolutional Neural Network: Bonus Problem

Changes in relative positions of object parts



CNN Architectures

ImageNet Large Scale Visual Recognition



VGG

Small filters (3x3)

Why not take 7x7?

7x7 #params = 7^2 * C^2
3 * 3x3 #params = 3 * (3^2 * C^2)

3 filters with 3x3 kernel
have same receptive field as 7x7 kernel,
but deeper with more nonlinearities

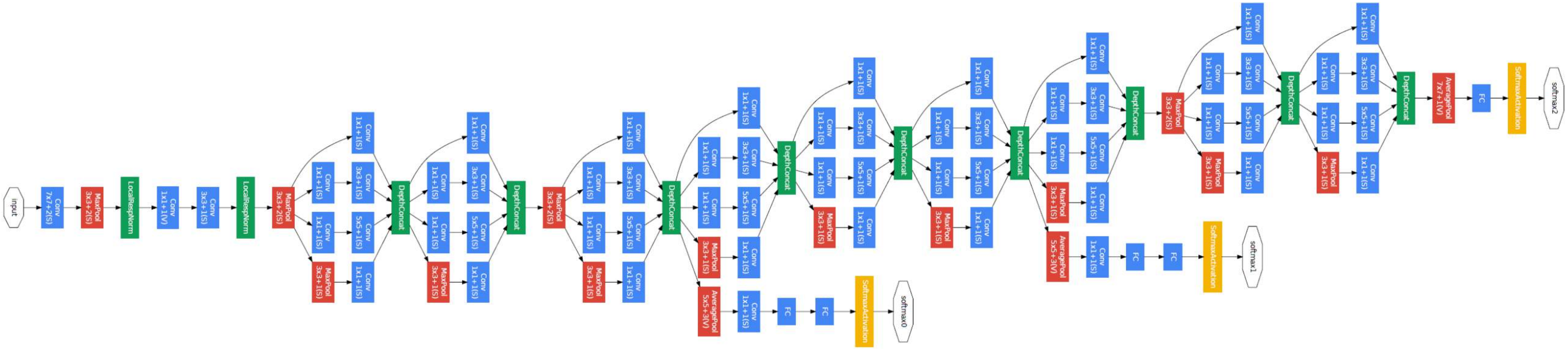
VGG

Most of parameters come from last 3 layers

FC 1: 7*7*512*4096 = 103M
FC 2: 4096*4096 = 17M
FC 3: 4096*1000 = 4M

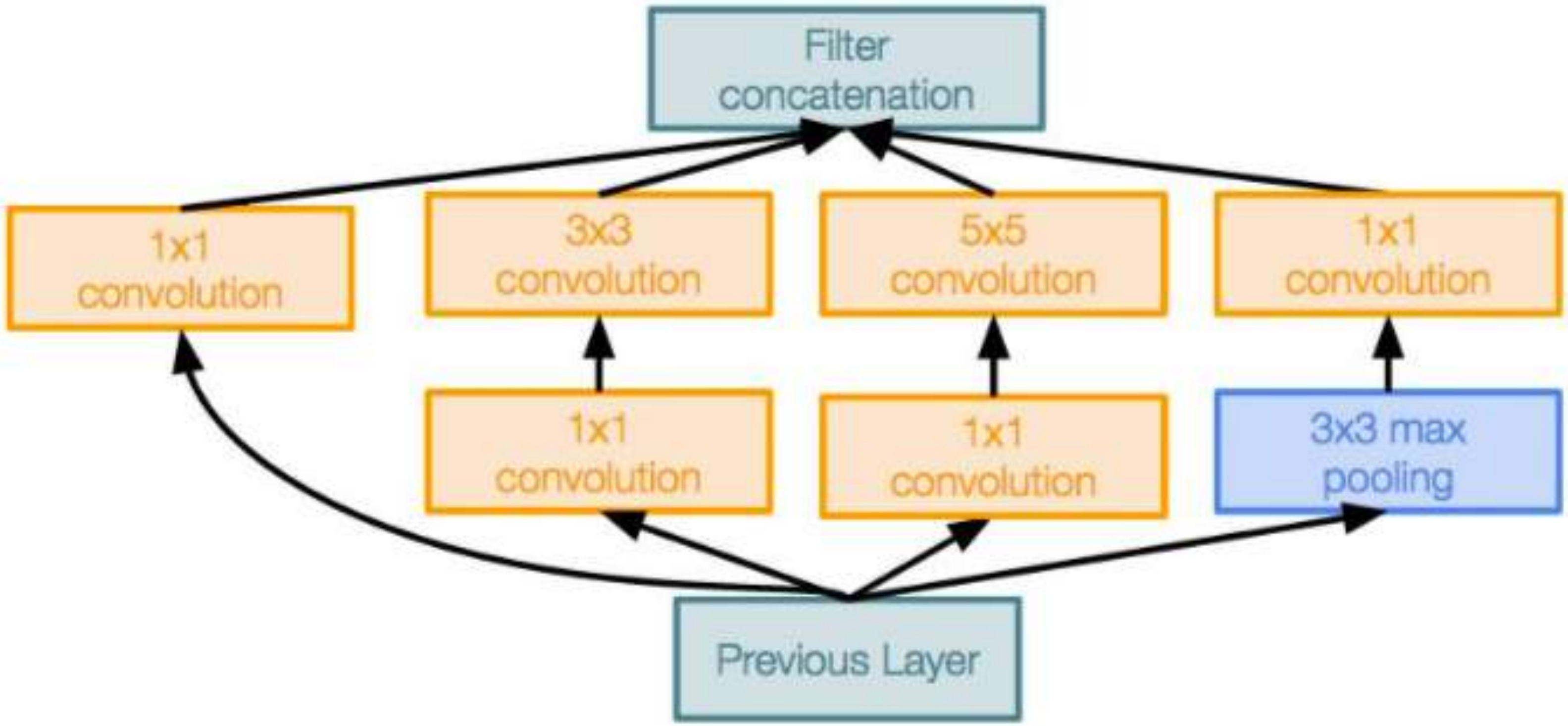
Total #params: 138M

GoogleNet

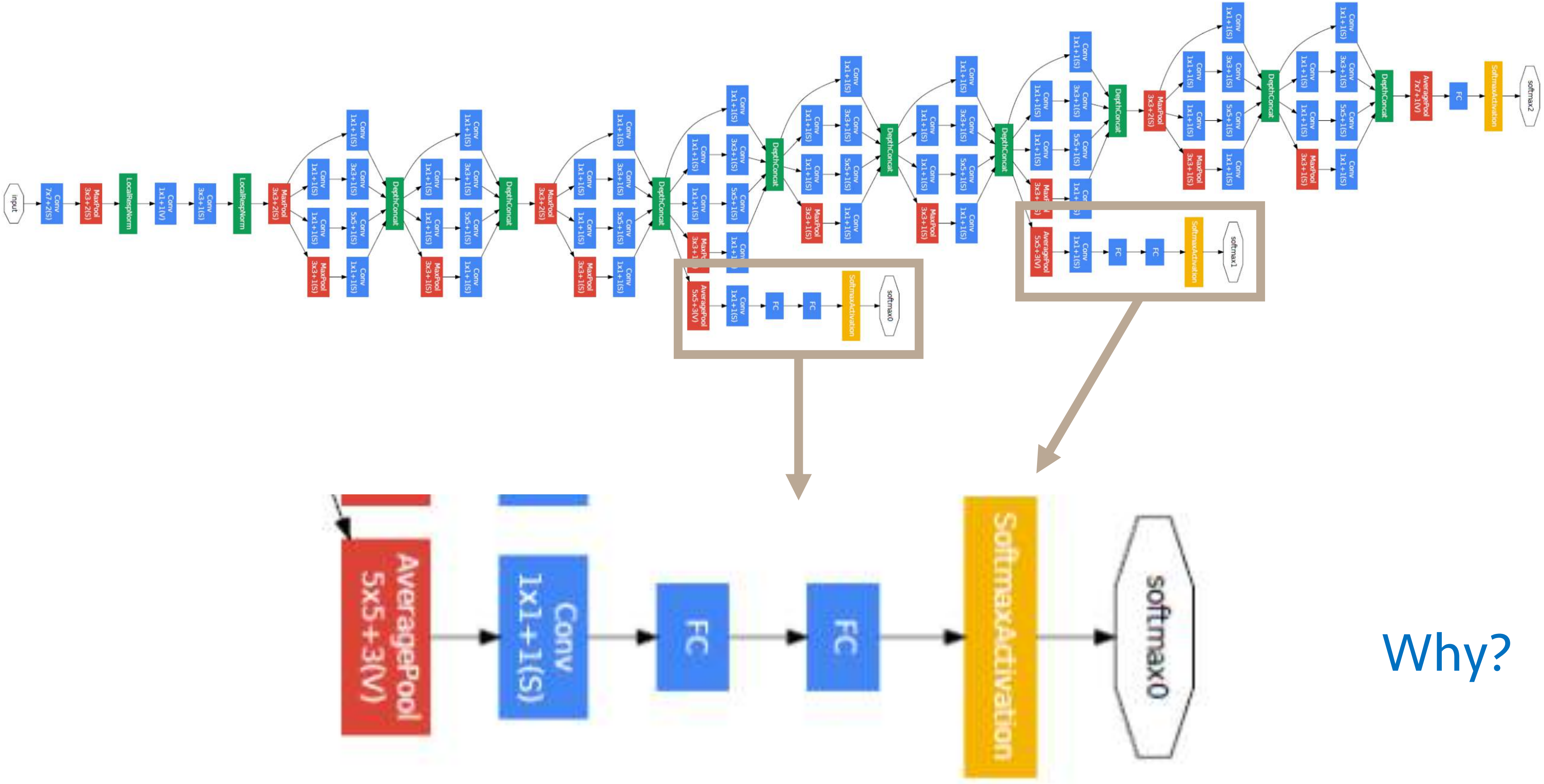


- 22 layers
- Efficient “Inception” module
- No FC layers
- Only 5 million parameters

GoogleNet: Inception Module



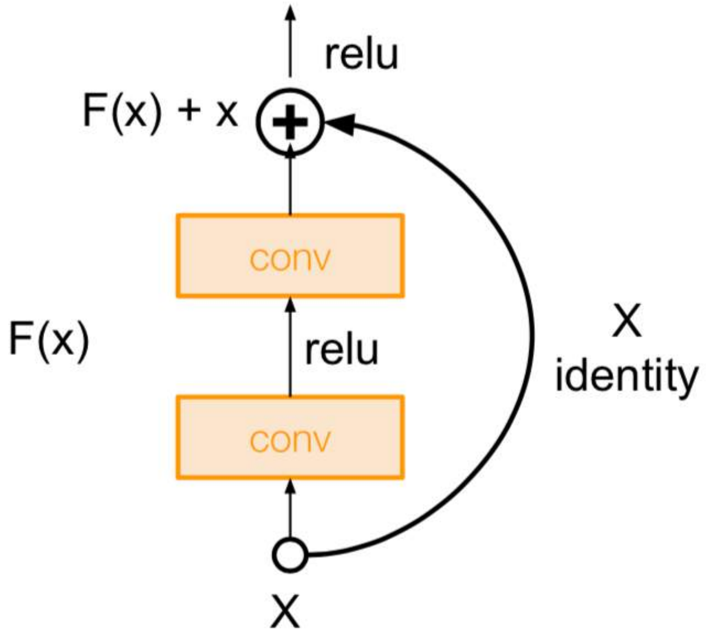
GoogleNet: Many Classifiers



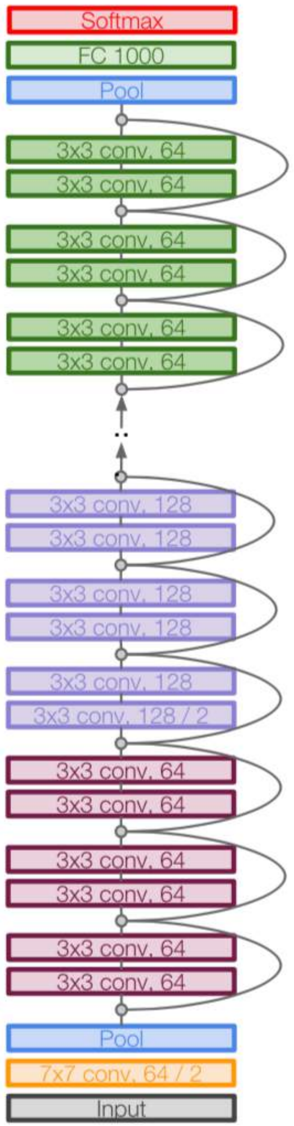
Why?

ResNet

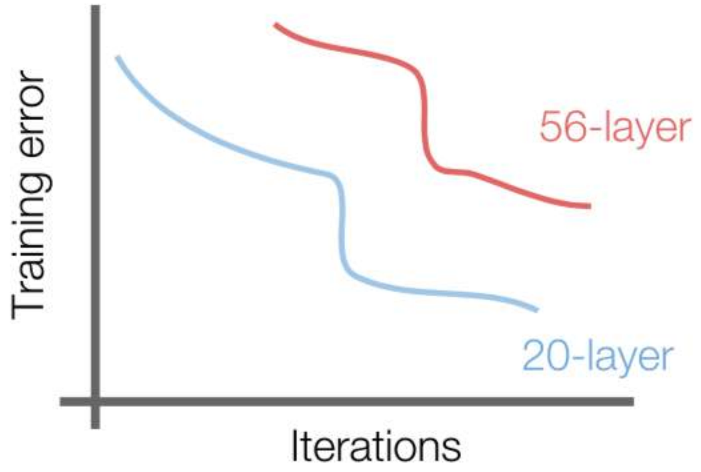
152-layer model
for ImageNet

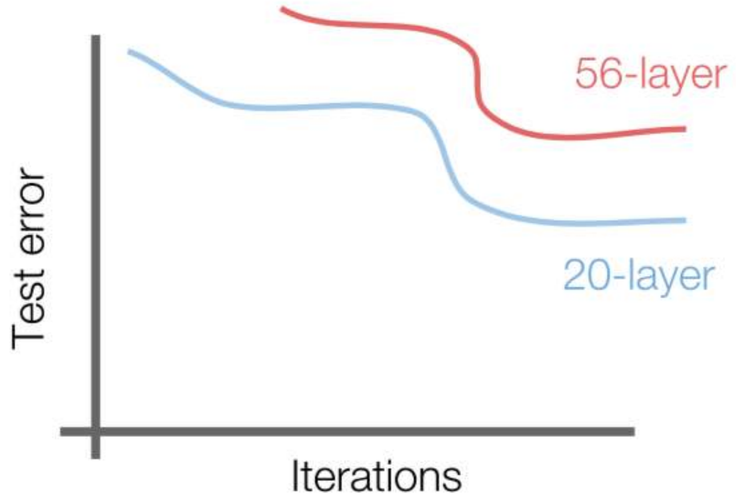


Residual block



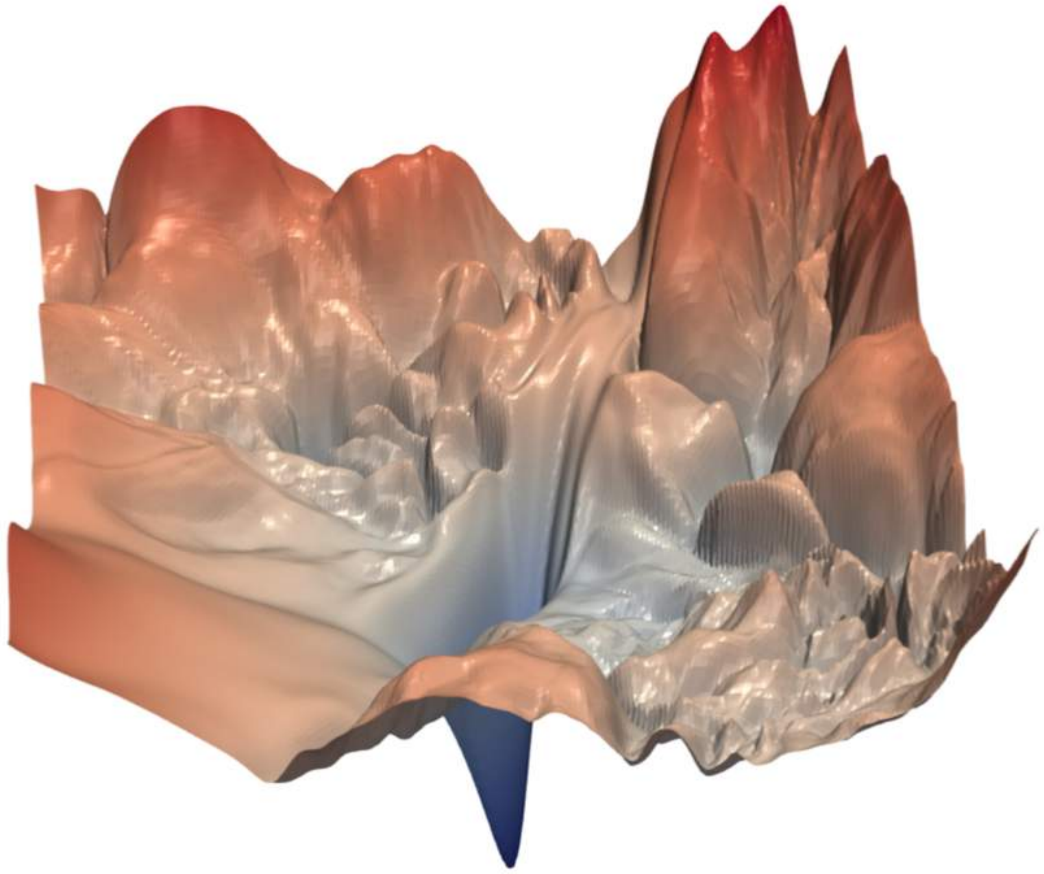
ResNet



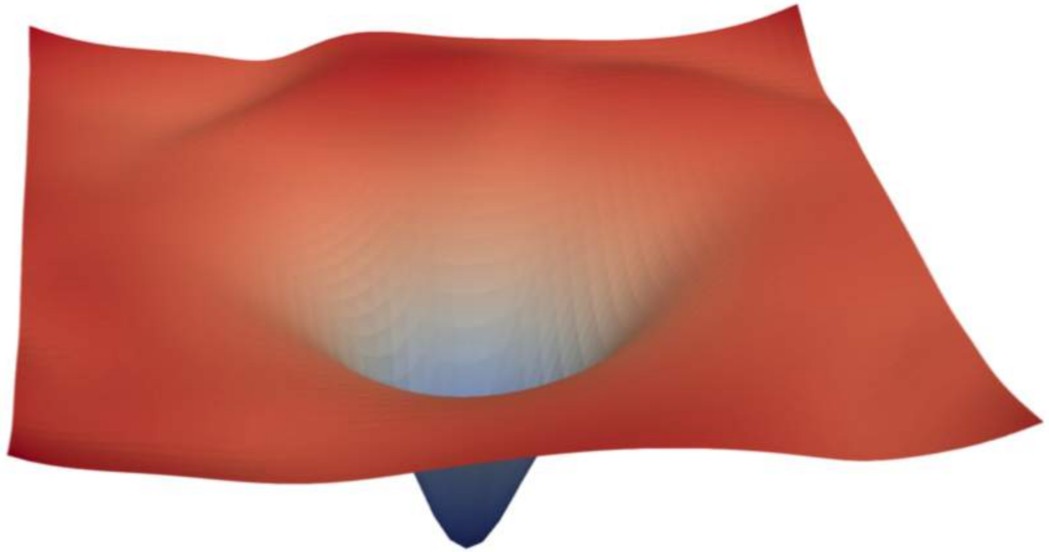


Overfitting or hard optimization?

ResNet

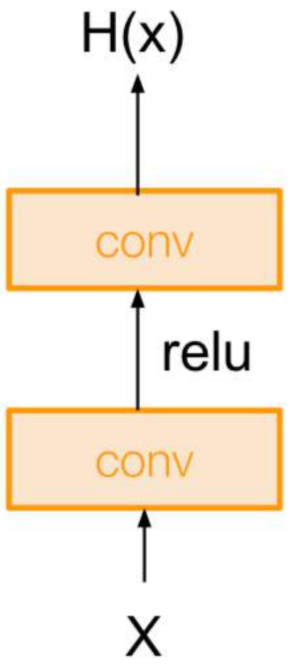


No skip-connections

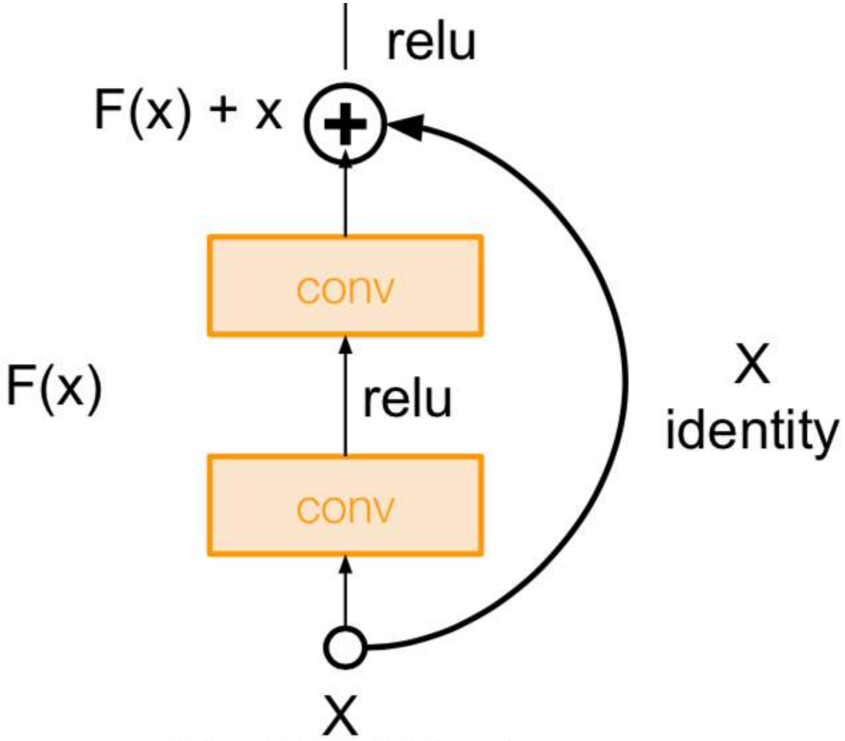


With skip-connections

ResNet

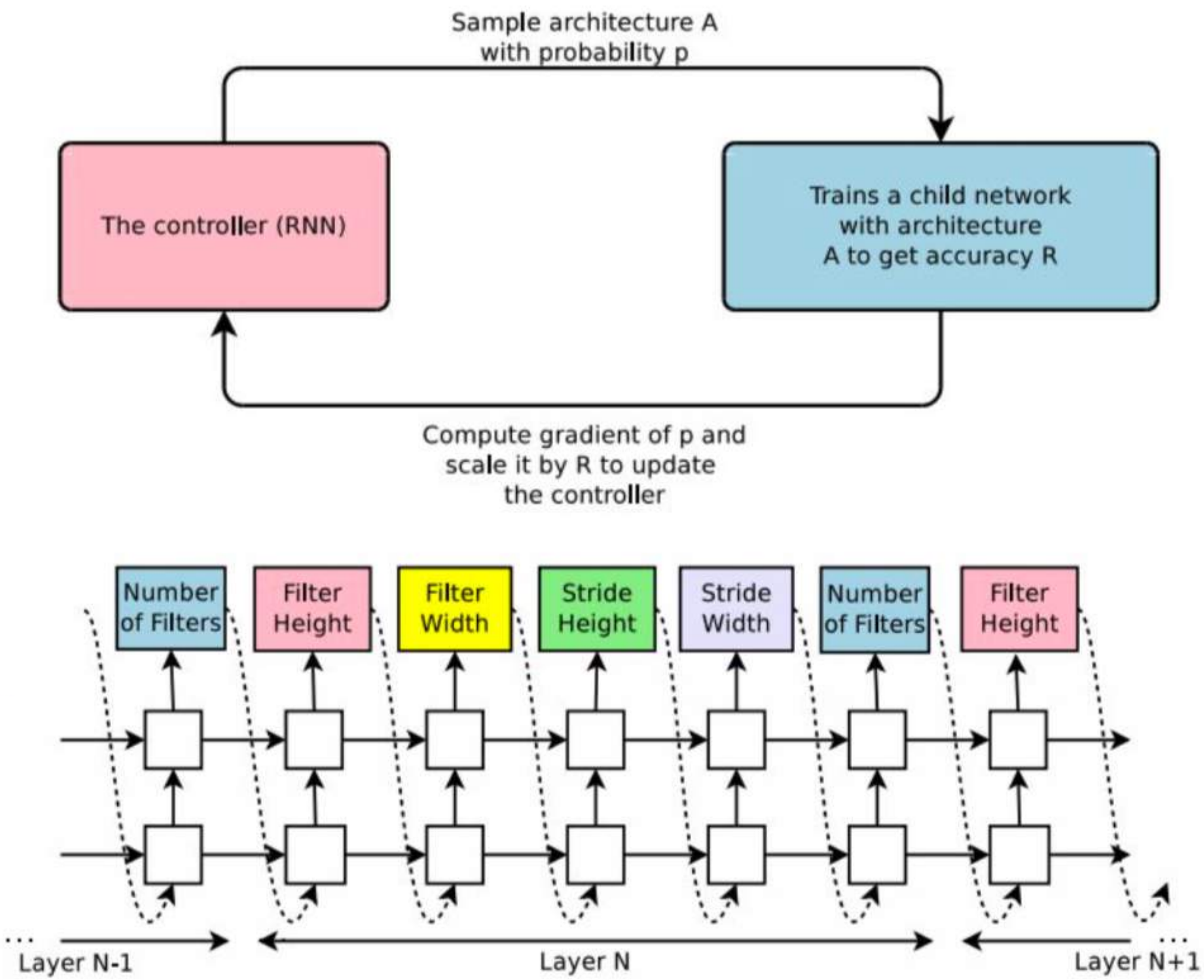


“Plain” layers



Residual block

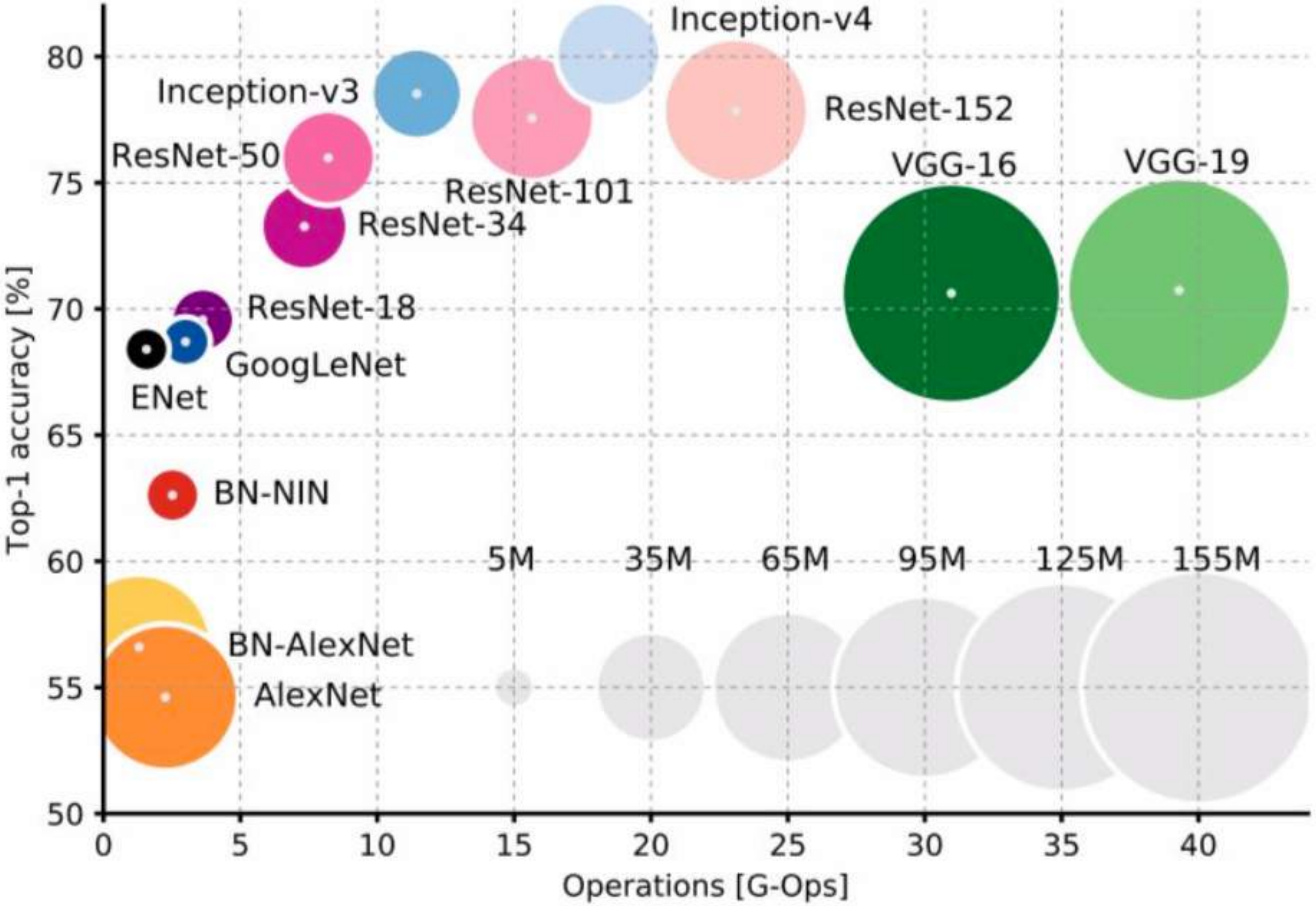
NasNet


Sample architecture A
with probability p
The controller (RNN)
Trains a child network
with architecture
A to get accuracy R
Compute gradient of p and
scale it by R to update
the controller
Number
of Filters
Filter
Height
Filter
Width
Stride
Height
Stride
Width
Number
of Filters
Filter
Height
Layer N-1
Layer N
Layer N+1

Also

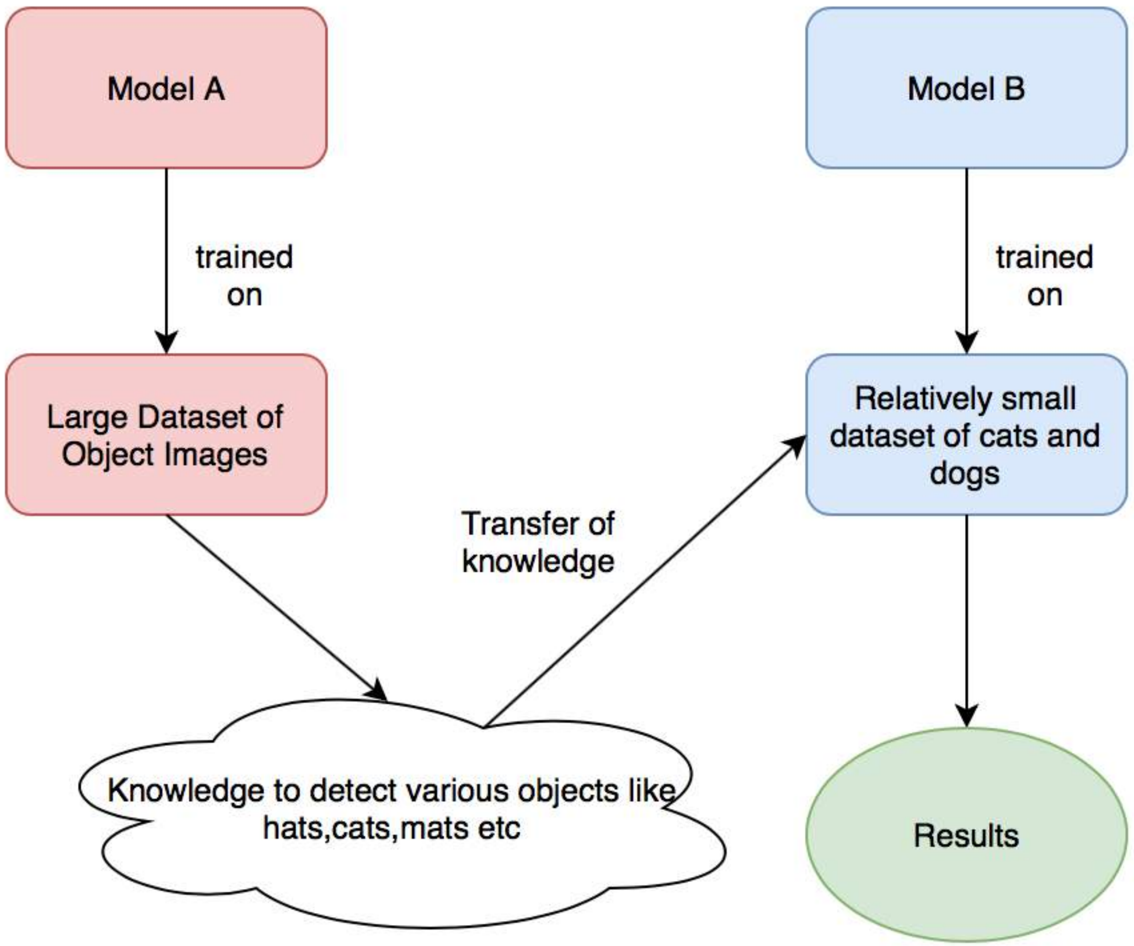
- NiN (Network in Network)
- Wide ResNet
- ResNeXT
- Stochastic Depth
- Squeeze-and-Excitation Network
- DenseNet
- FractalNet
- SqueezeNet

Architecture Comparison

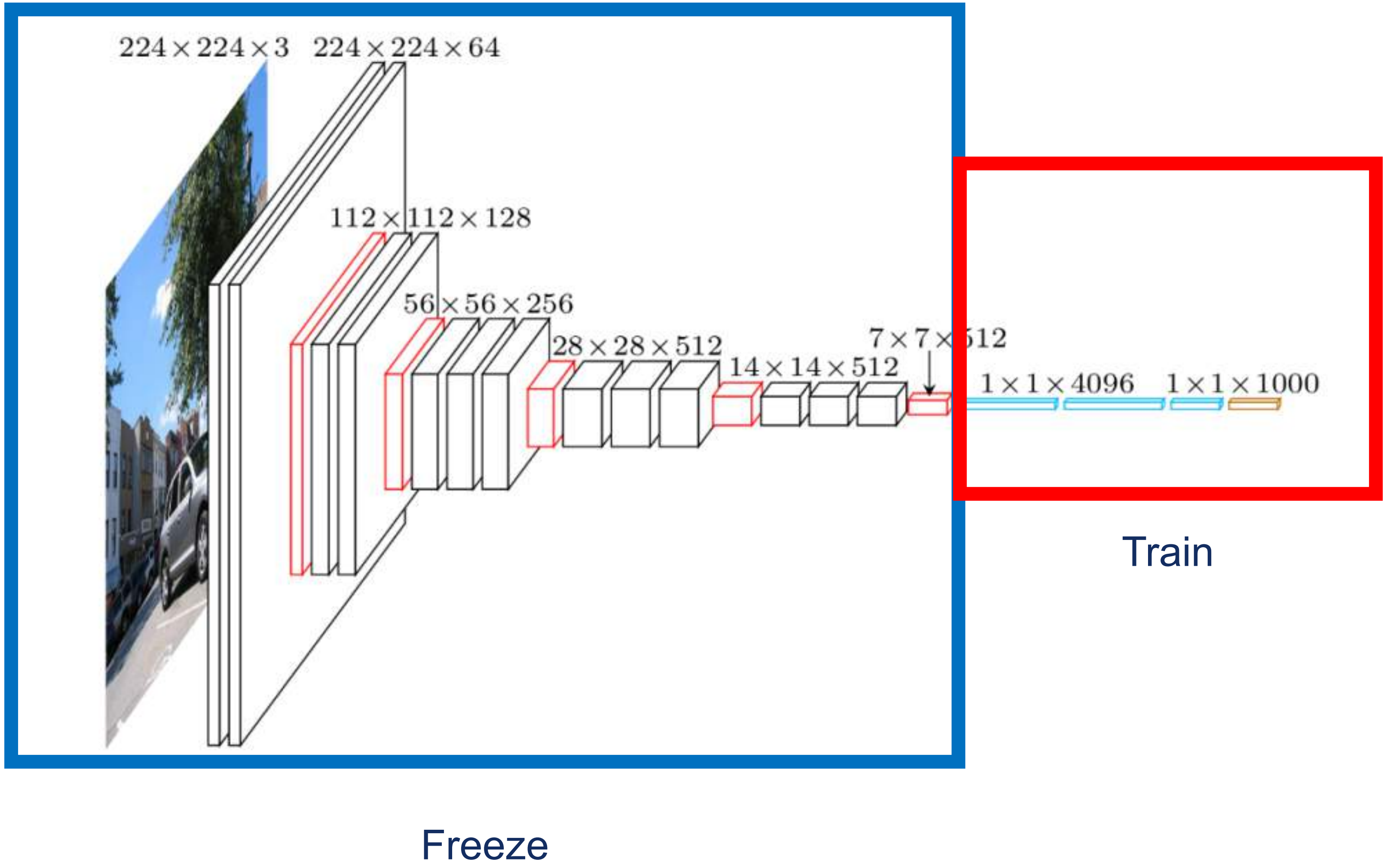


Transfer Learning

Transfer Learning



Transfer Learning



Transfer Learning Tips & Tricks

- The more similar are 2 datasets the less layers you need to train
- The less data you have the less layers you need to train
- Unfreeze layers one by one during training

Materials

cs231n Lecture 2: https://www.youtube.com/watch?v=OoUX-nOEjG0&list=PLC1qU-LWwrF64f4QKQT-Vg5Wr4qEE1Zxk&index=2

Cs231n Lecture 9: https://www.youtube.com/watch?v=DAOcjicFr1Y&list=PLC1qU-LWwrF64f4QKQT-Vg5Wr4qEE1Zxk&index=9

Transfer Learning: https://towardsdatascience.com/transfer-learning-946518f95666

Stochastic Weight Averaging—a New Way to Get State of the Art Results in Deep Learning: https://towardsdatascience.com/stochastic-weight-averaging-a-new-way-to-get-state-of-the-art-results-in-deep-learning-c639ccf36a