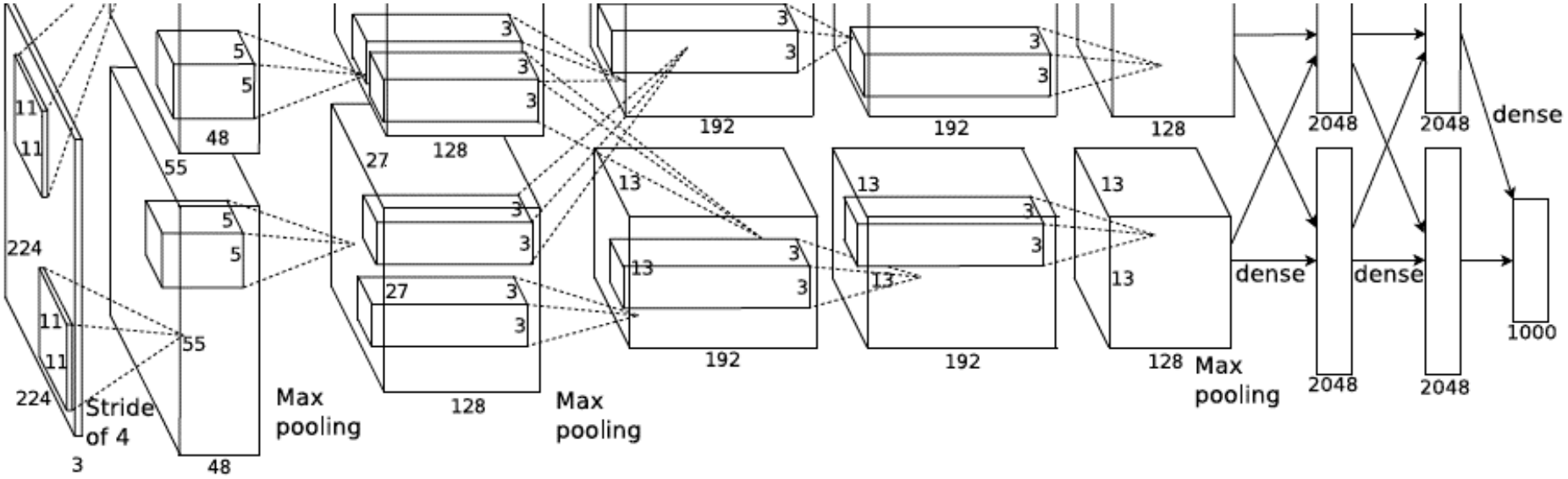
Convolutional Neural Networks



Computer Vision

Jia-Bin Huang, Virginia Tech

Today’s class

- Overview
- Convolutional Neural Network (CNN)
- Training CNN
- Understanding and Visualizing CNN

Image Categorization: Training phase

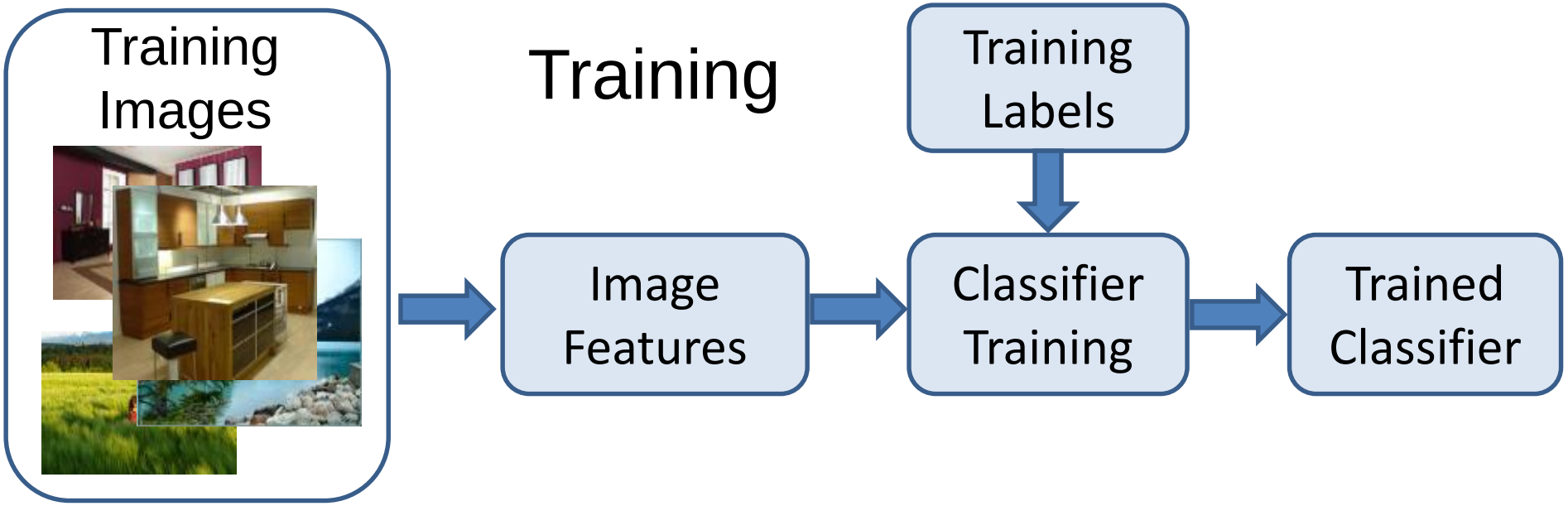
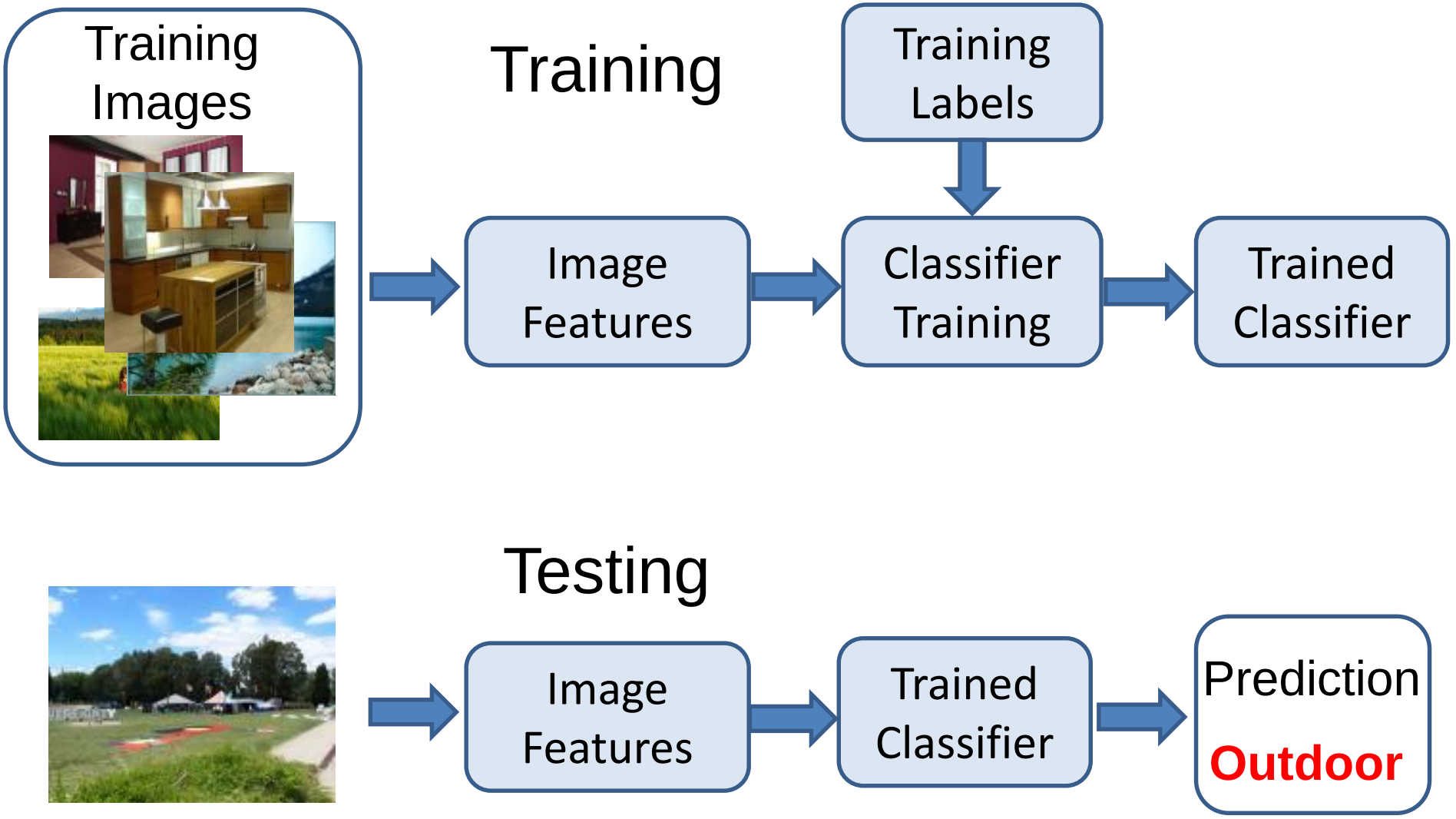
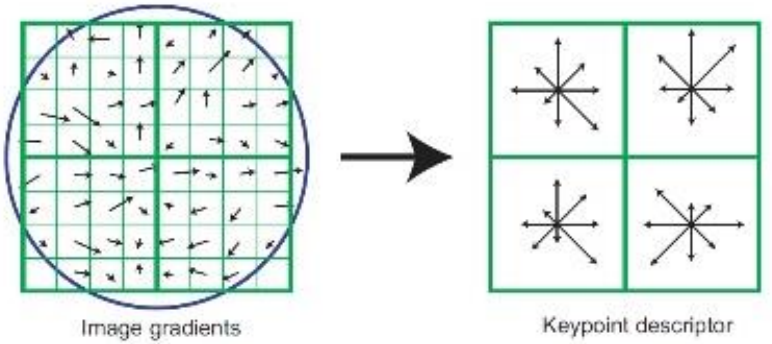


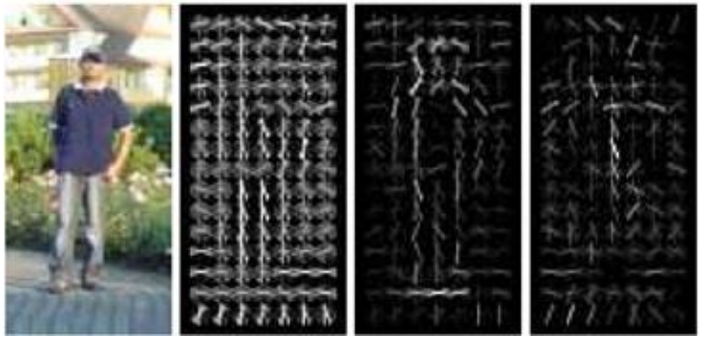
Image Categorization: Testing phase



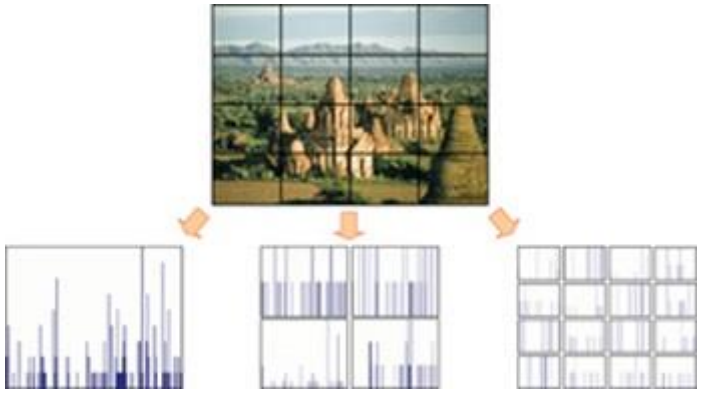
Features are the Keys



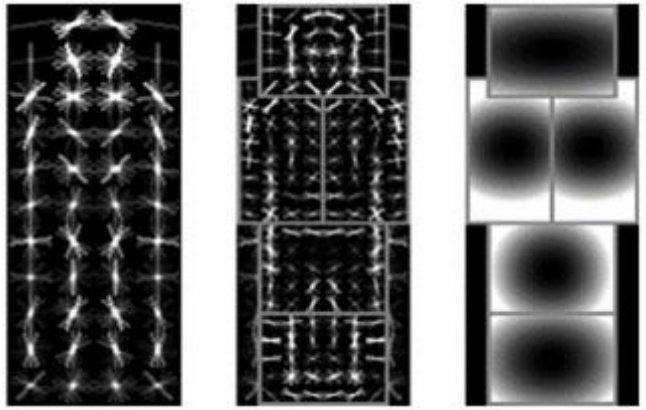
SIFT [Loewe IJCV 04]



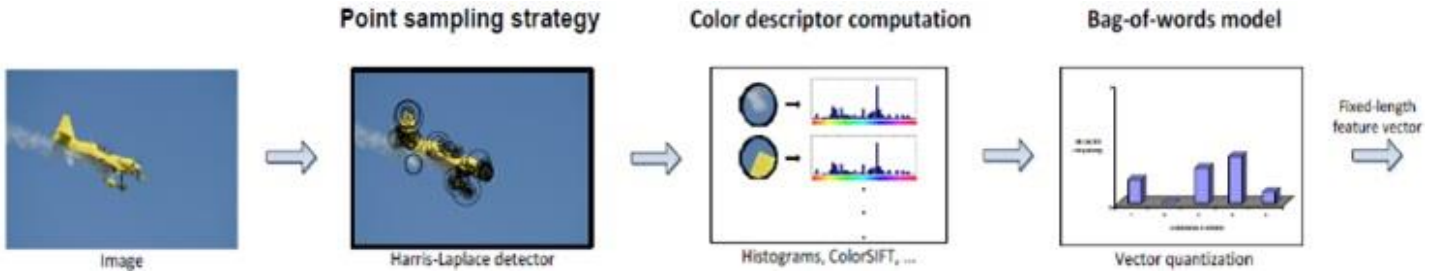
HOG [Dalal and Triggs CVPR 05]



SPM [Lazebnik et al. CVPR 06]



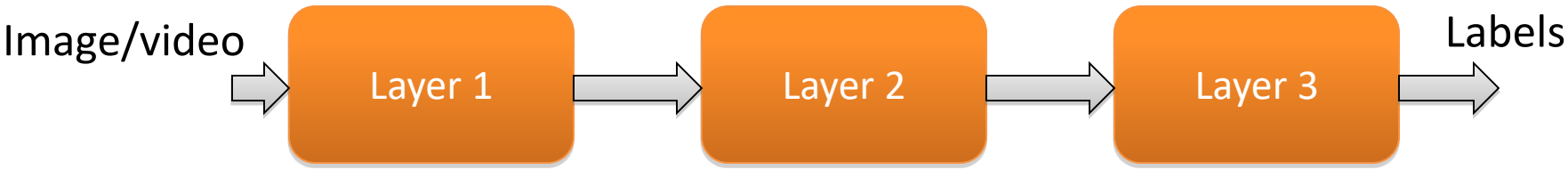
DPM [Felzenszwalb et al. PAMI 10]



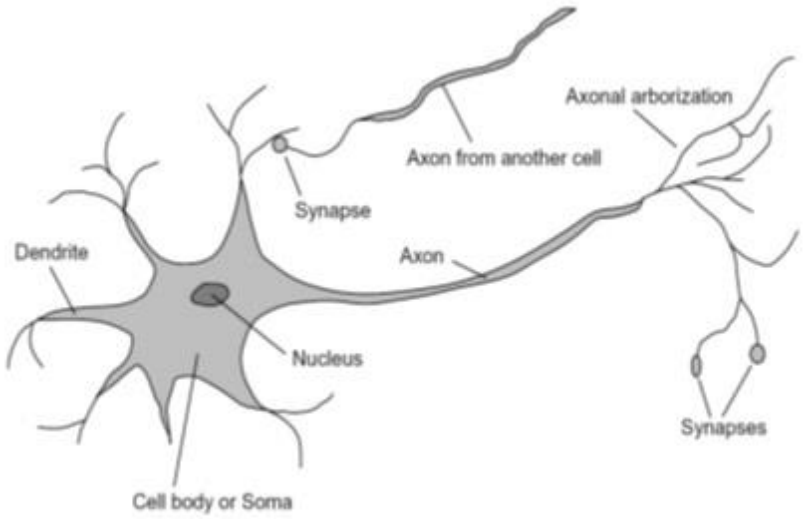
Color Descriptor [Van De Sande et al. PAMI 10]

Learning a Hierarchy of Feature Extractors

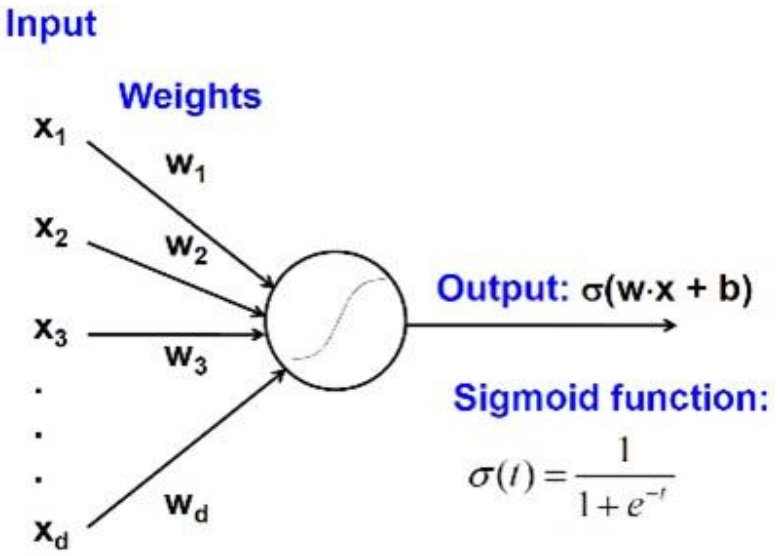
- Each layer of hierarchy extracts features from output of previous layer
- All the way from pixels → classifier
- Layers have the (nearly) same structure



Biological neuron and Perceptrons



A biological neuron

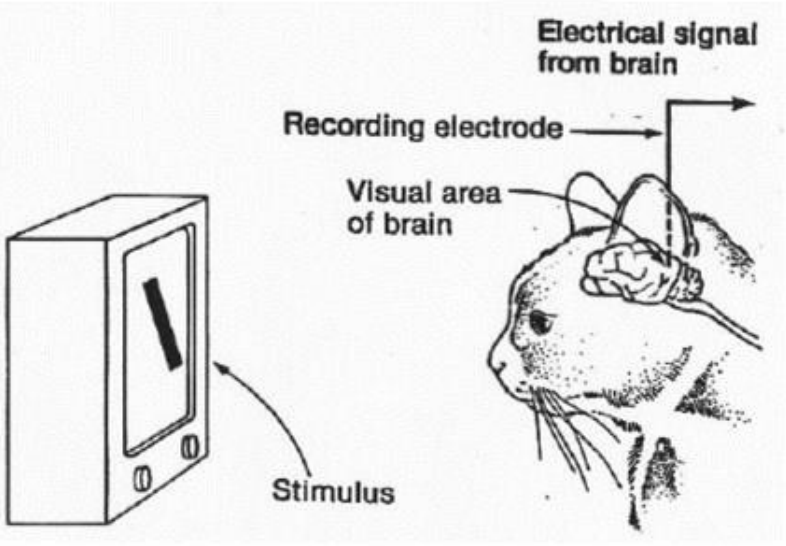


An artificial neuron (Perceptron)
- a linear classifier

Simple, Complex and Hypercomplex cells

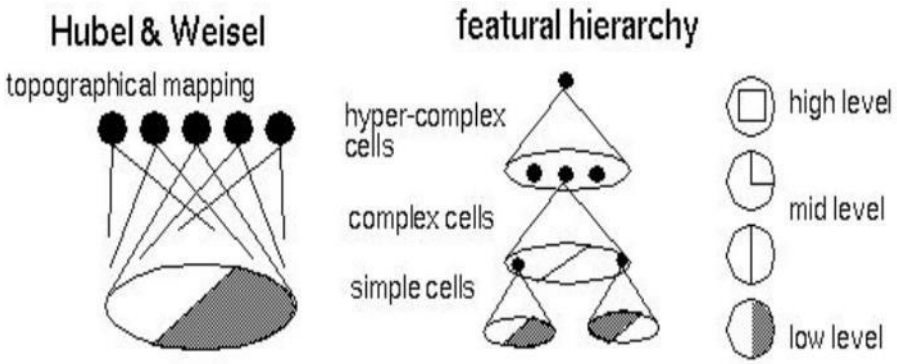
David H. Hubel and Torsten Wiesel

Suggested a **hierarchy** of **feature detectors** in the visual cortex, with higher level features responding to patterns of activation in lower level cells, and propagating activation upwards to still higher level cells.

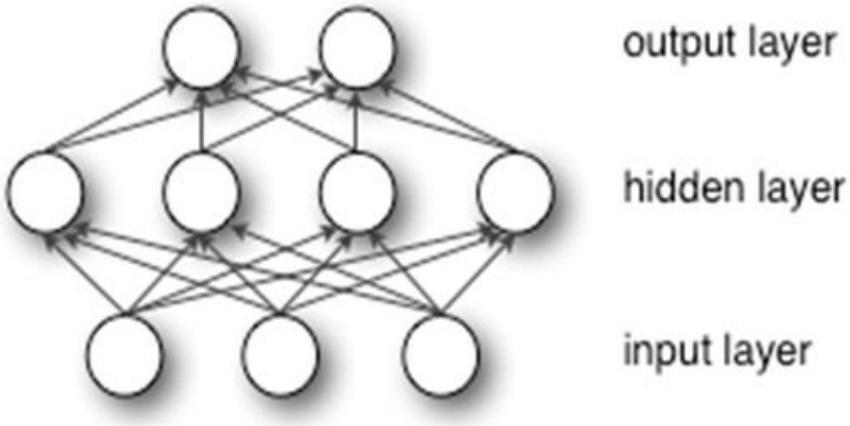


David Hubel's Eye, Brain, and Vision

Hubel/Wiesel Architecture and Multi-layer Neural Network



Hubel and Weisel's architecture



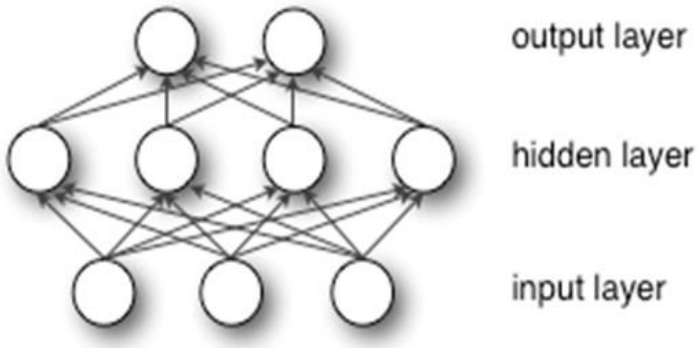
Multi-layer Neural Network
- A *non-linear* classifier

Multi-layer Neural Network

- A non-linear classifier
- **Training:** find network weights **w** to minimize the error between true training labels y_i and estimated labels $f_{\boldsymbol{w}}(\boldsymbol{x_i})$

$$E(\mathbf{w}) = \sum_{i=1}^{N} \left(y_i - f_{\mathbf{w}}(\mathbf{x}_i)\right)^2$$

- Minimization can be done by gradient descent provided f is differentiable
- This training method is called **back-propagation**

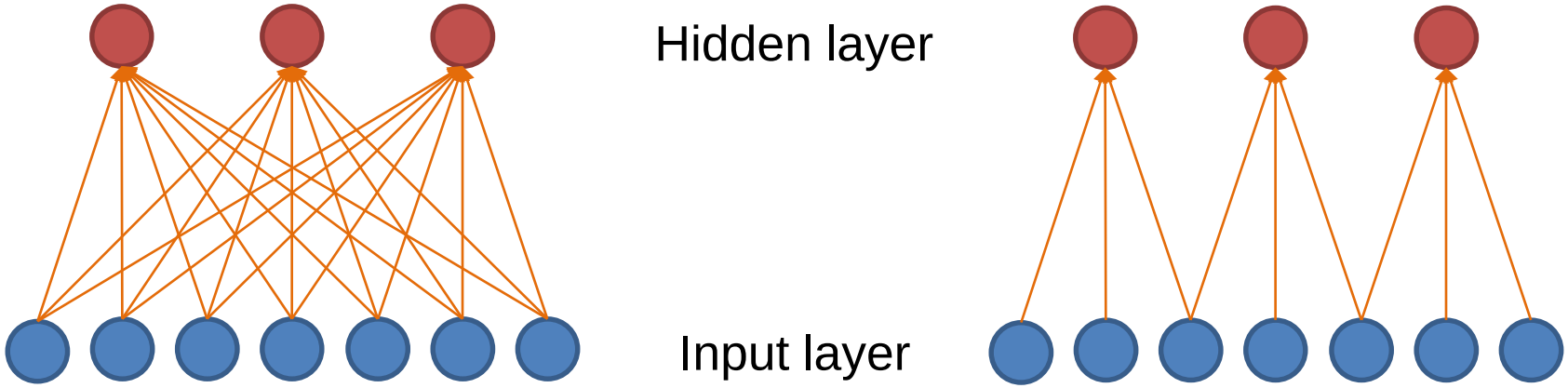


Convolutional Neural Networks

- Also known as CNN, ConvNet, DCN

- CNN = a multi-layer neural network with
  1. Local connectivity
  2. Weight sharing

CNN: Local Connectivity

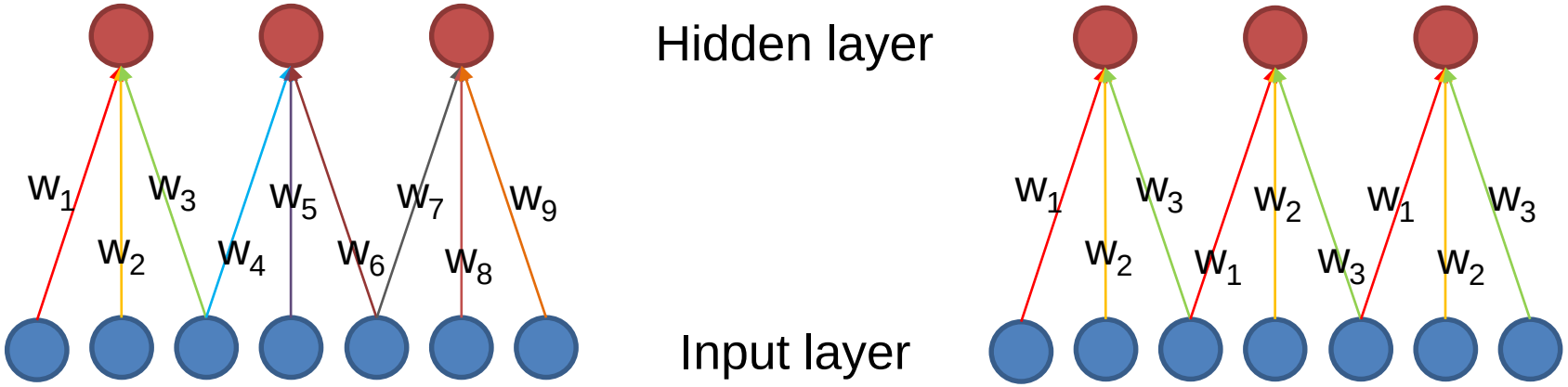


Global connectivity

Local connectivity

- # input units (neurons): 7
- # hidden units: 3
- Number of parameters
  - Global connectivity: 3 x 7 = 21
  - Local connectivity: 3 x 3 = 9

CNN: Weight Sharing

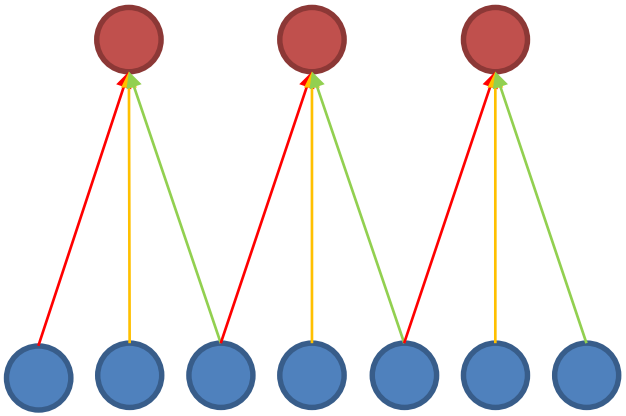


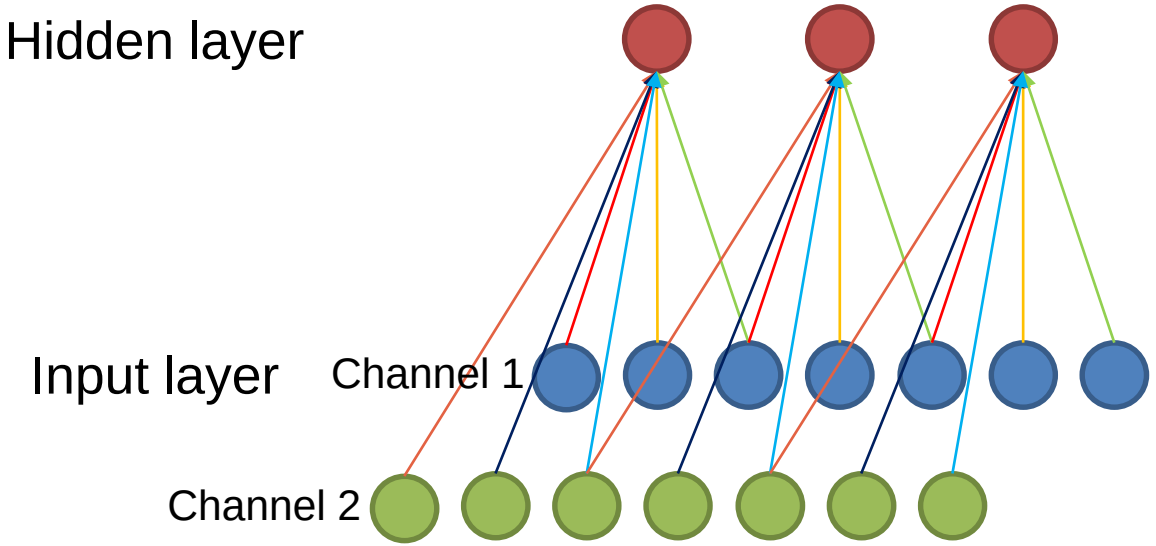
Without weight sharing

With weight sharing

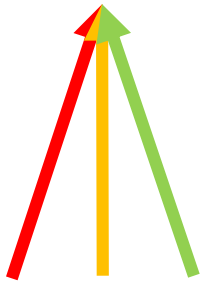
- # input units (neurons): 7
- # hidden units: 3
- Number of parameters
  - Without weight sharing: 3 x 3 = 9
  - With weight sharing : 3 x 1 = 3

CNN with multiple input channels



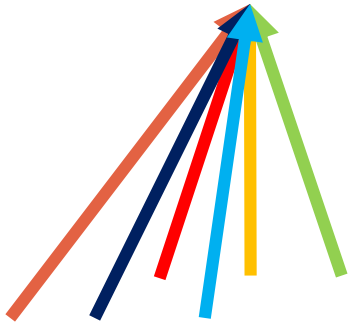


Single input channel



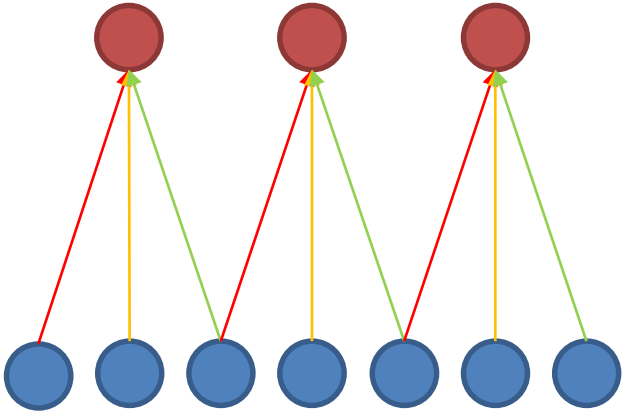
Filter weights

Multiple input channels



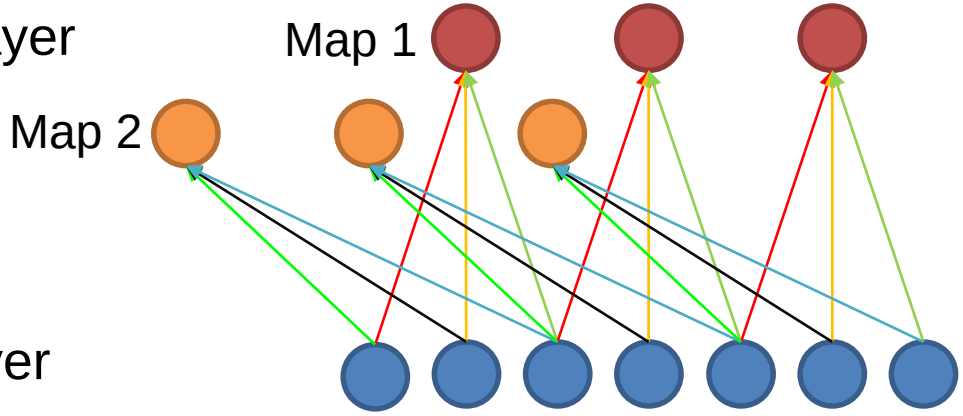
Filter weights

CNN with multiple output maps

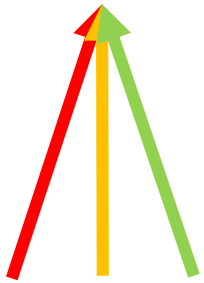


Hidden layer

Input layer

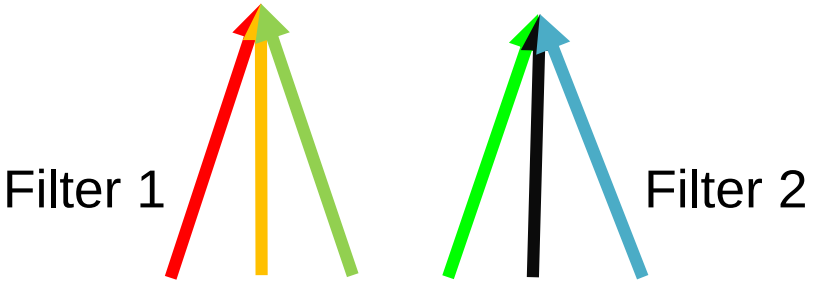


Single output map



Filter weights

Multiple output maps



Filter weights

Putting them together

- Local connectivity
- Weight sharing
- Handling multiple input channels
- Handling multiple output maps

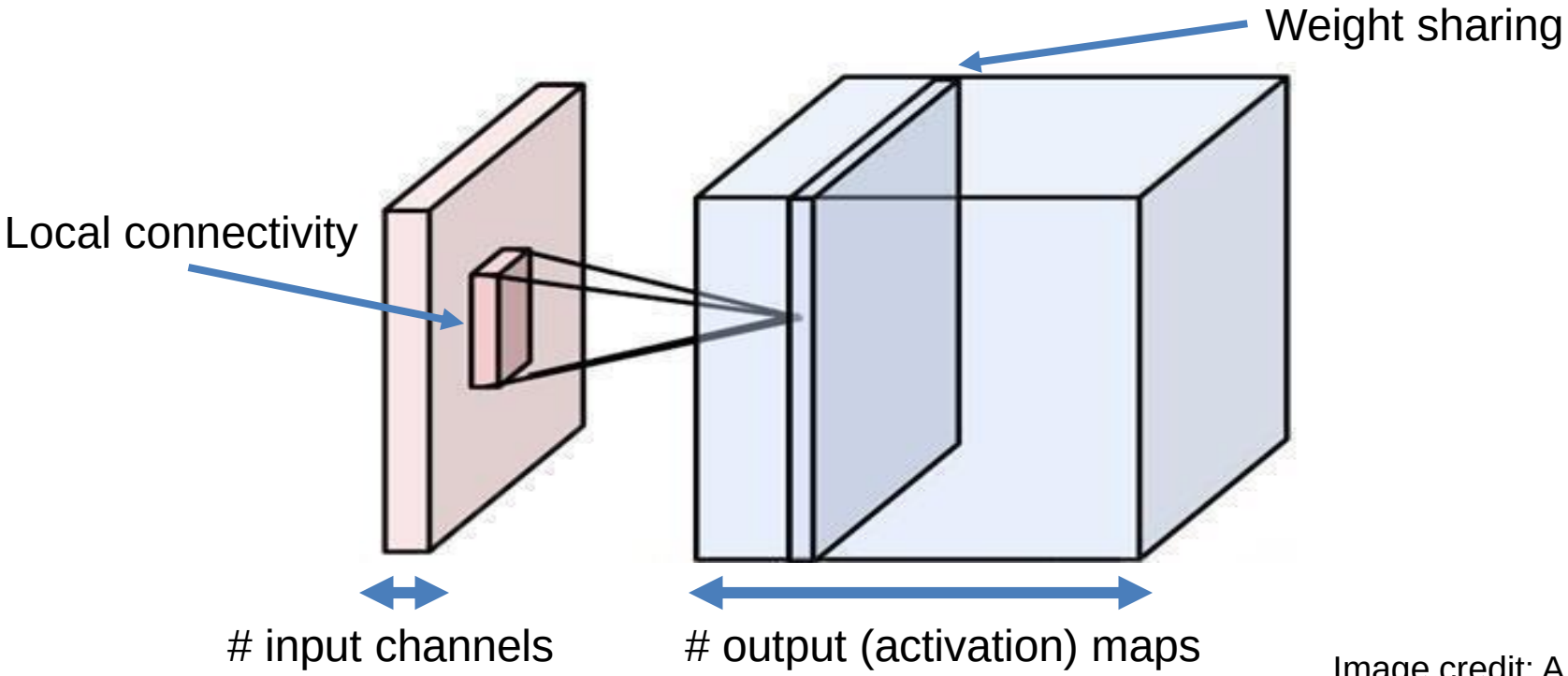


Image credit: A. Karpathy

Neocognitron [Fukushima, Biological Cybernetics 1980]

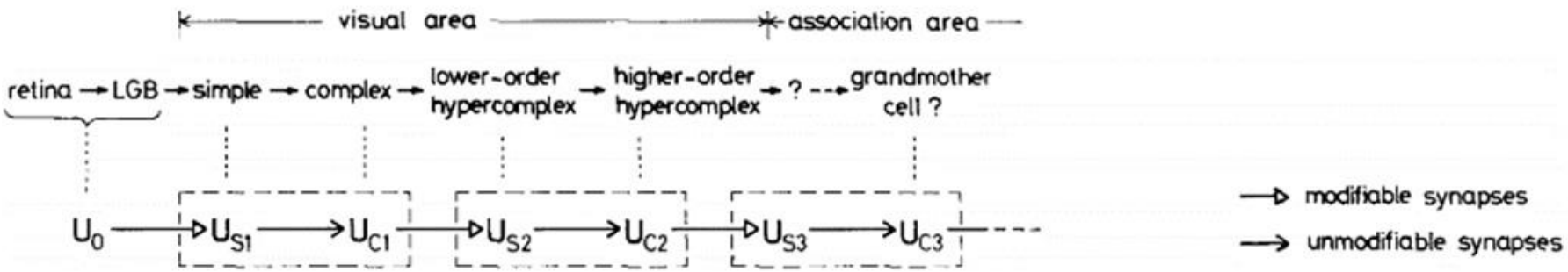
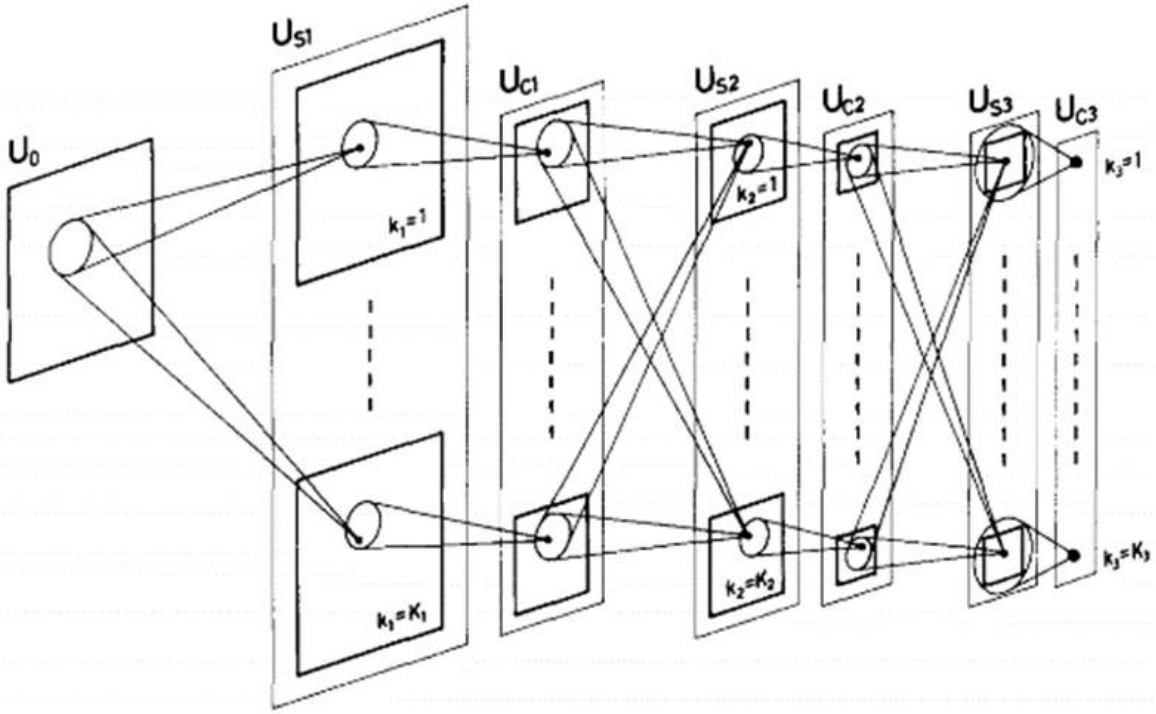


Fig. 1. Correspondence between the hierarchy model by Hubel and Wiesel, and the neural network of the neocognitron

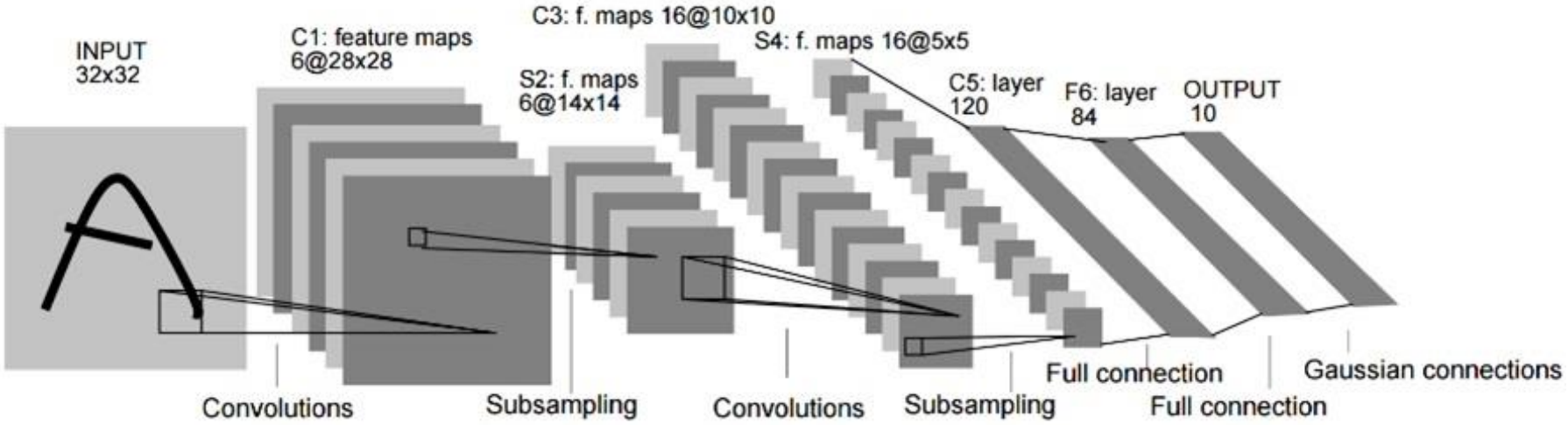


Deformation-Resistant Recognition

S-cells: (simple)
- extract local features

C-cells: (complex)
- allow for positional errors

LeNet [LeCun et al. 1998]

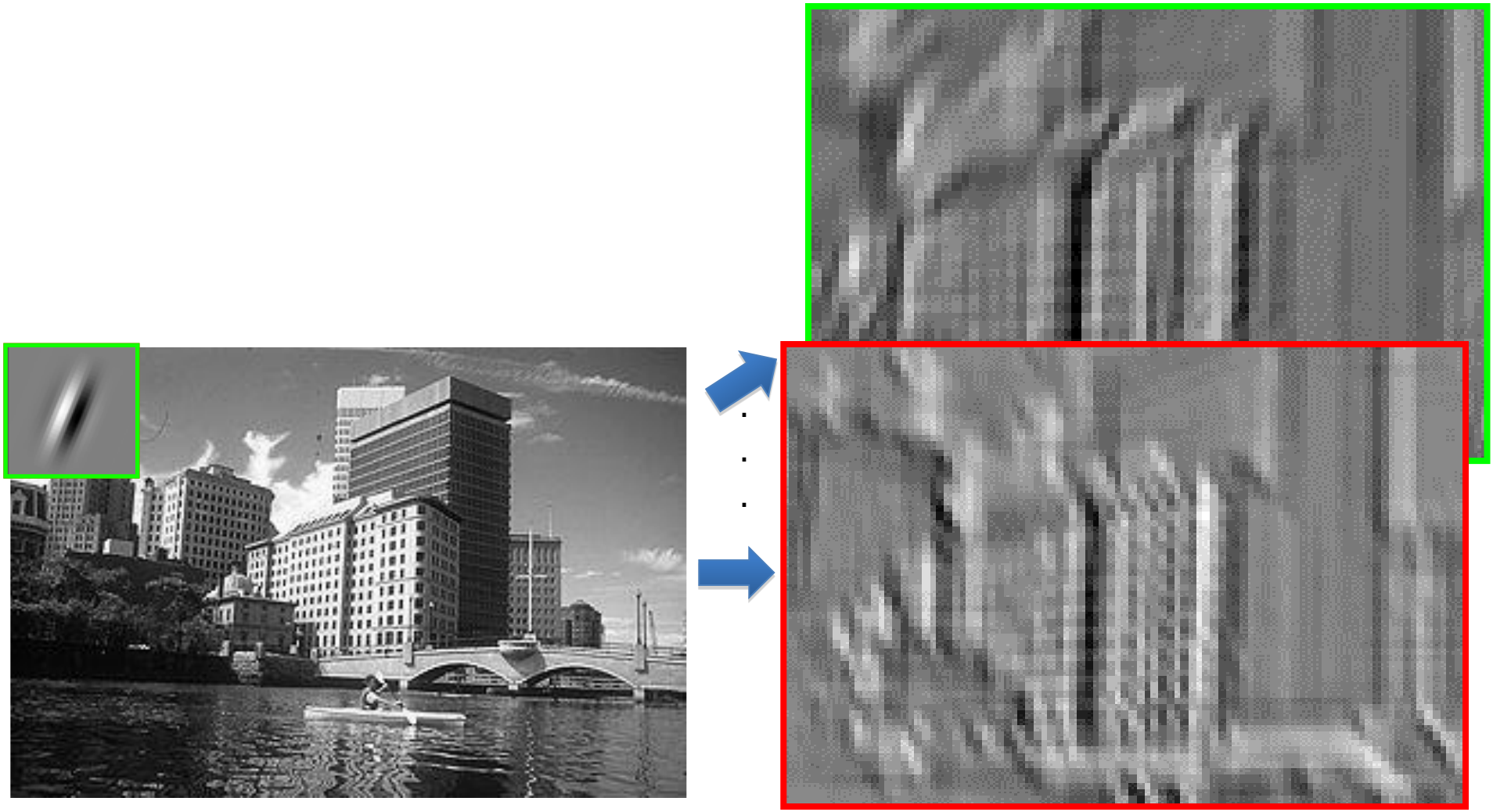


LeNet-1 from 1993

Gradient-based learning applied to document recognition [LeCun, Bottou, Bengio, Haffner 1998]

What is a Convolution?

- Weighted moving sum

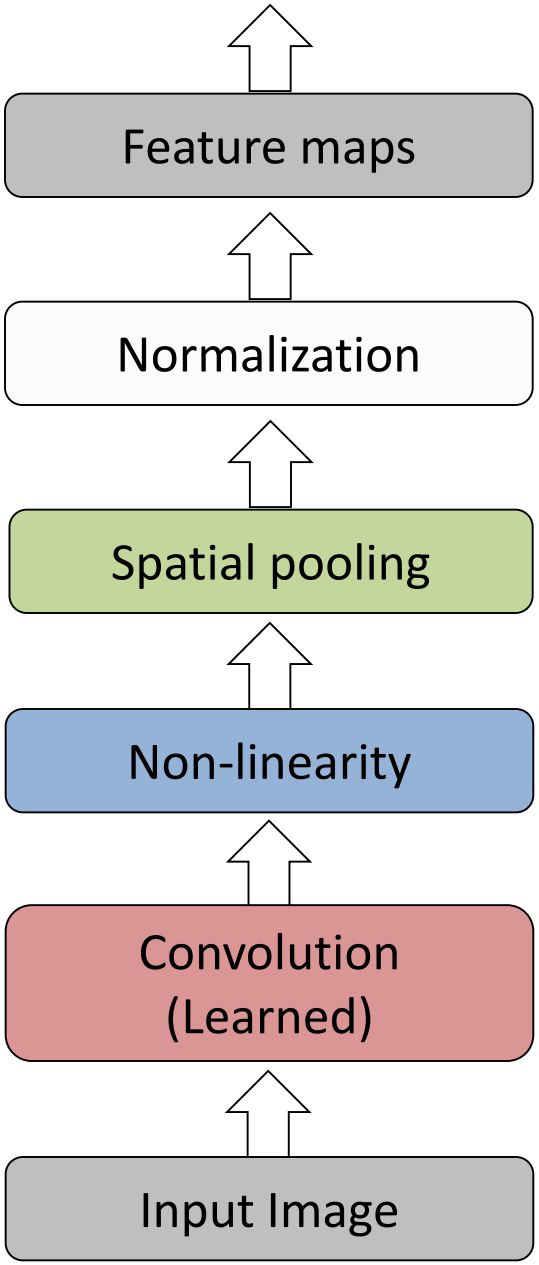


Input

Feature Activation Map

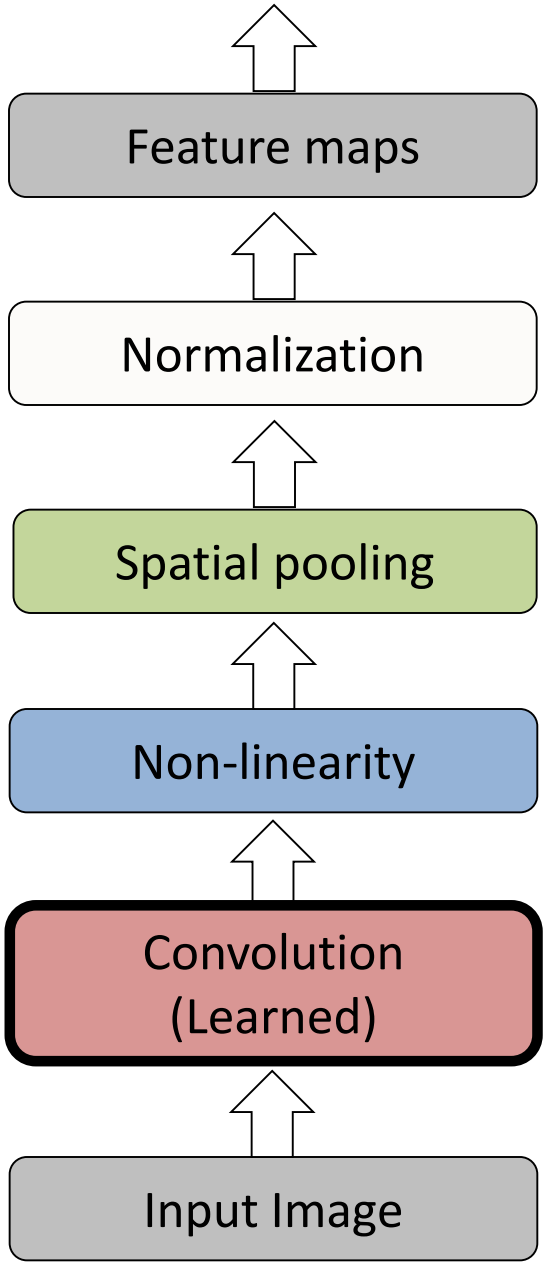
slide credit: S. Lazebnik

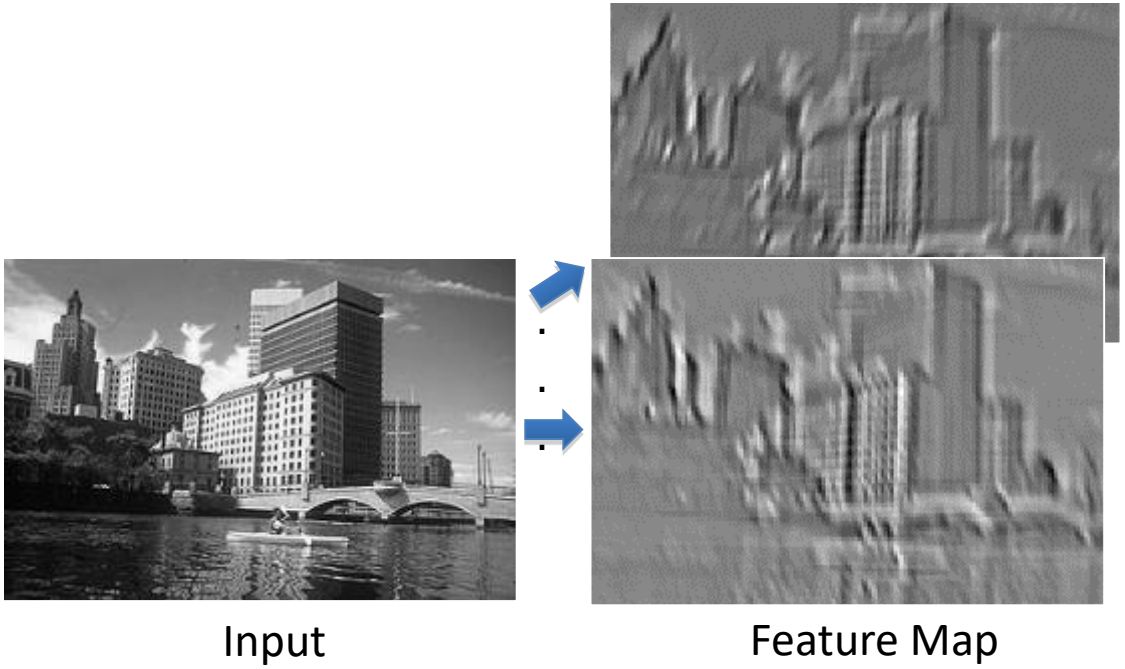
Convolutional Neural Networks



slide credit: S. Lazebnik

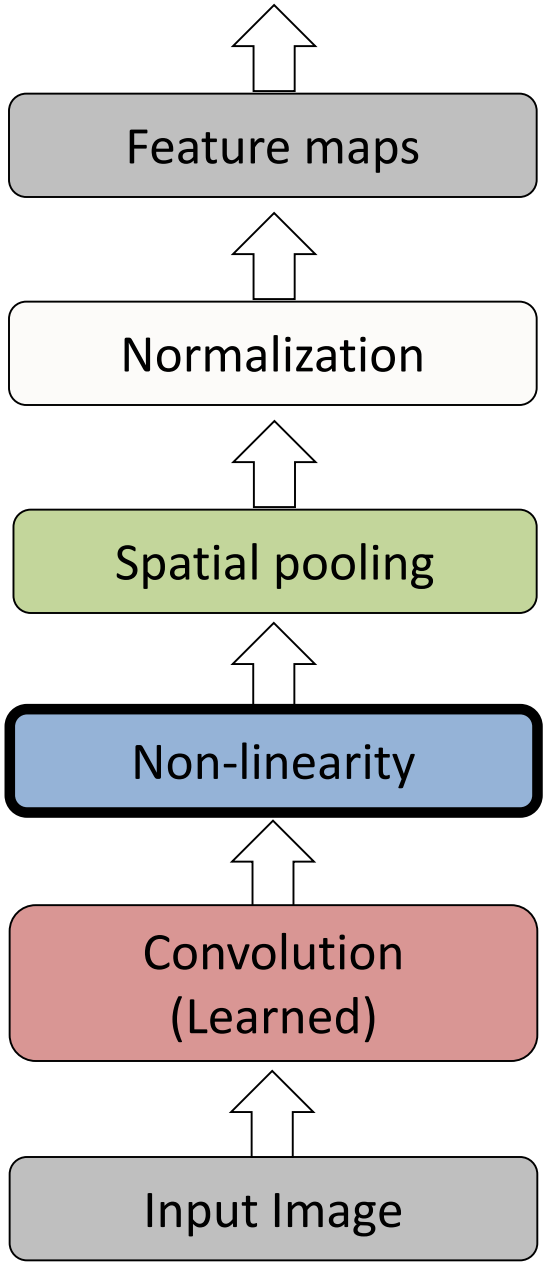
Convolutional Neural Networks



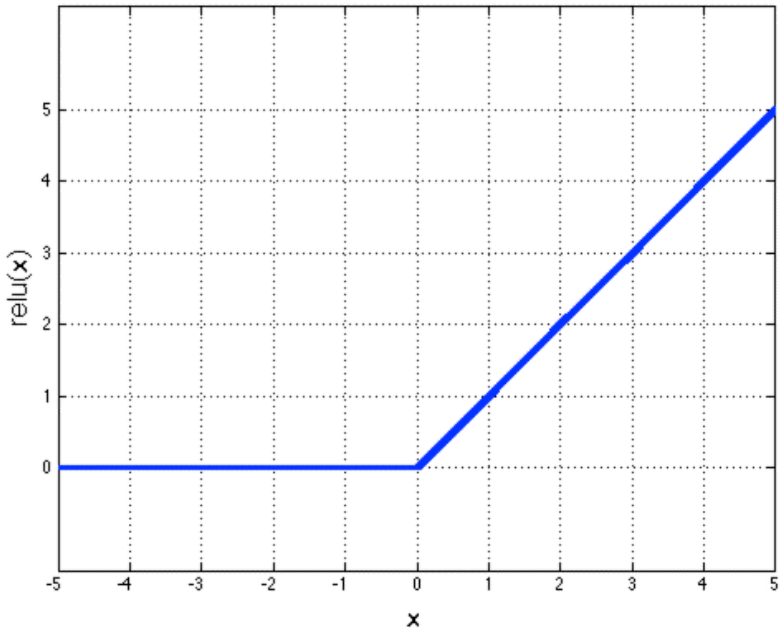


slide credit: S. Lazebnik

Convolutional Neural Networks

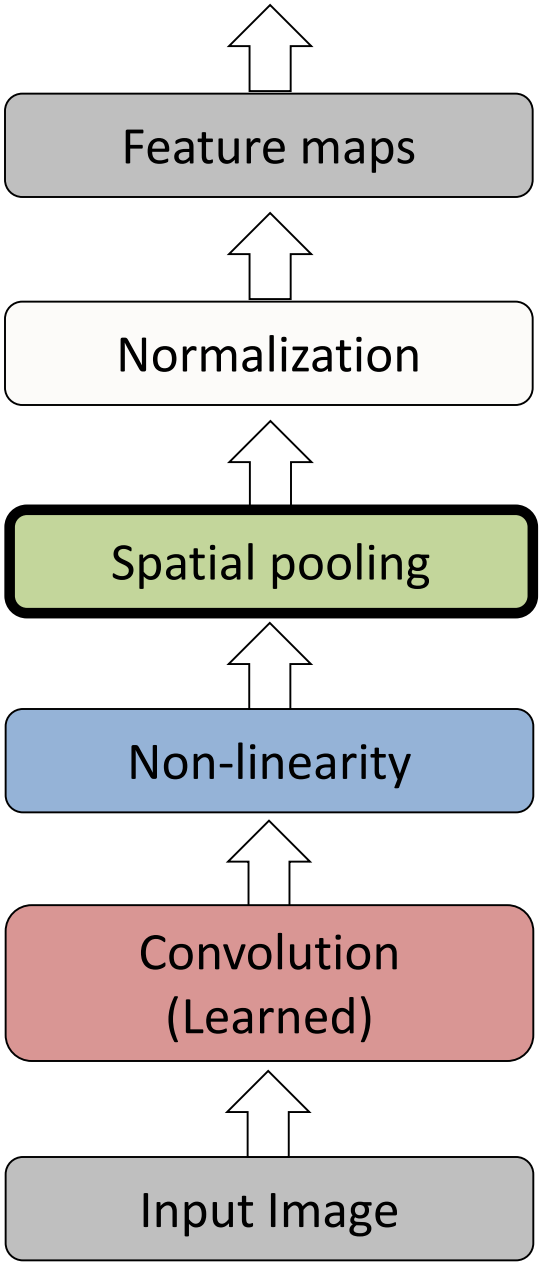


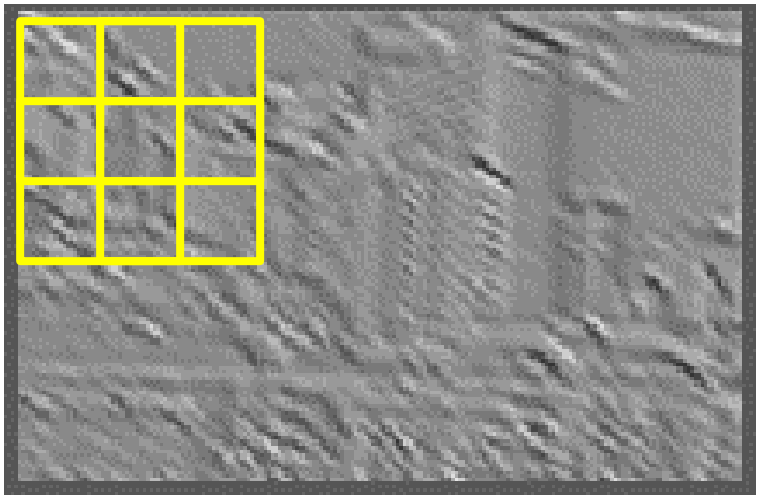
Rectified Linear Unit (ReLU)



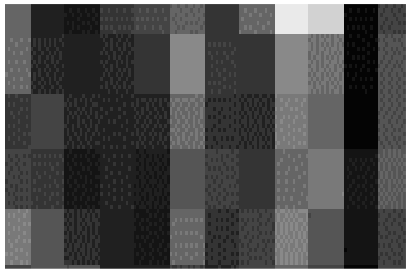
slide credit: S. Lazebnik

Convolutional Neural Networks





Max pooling

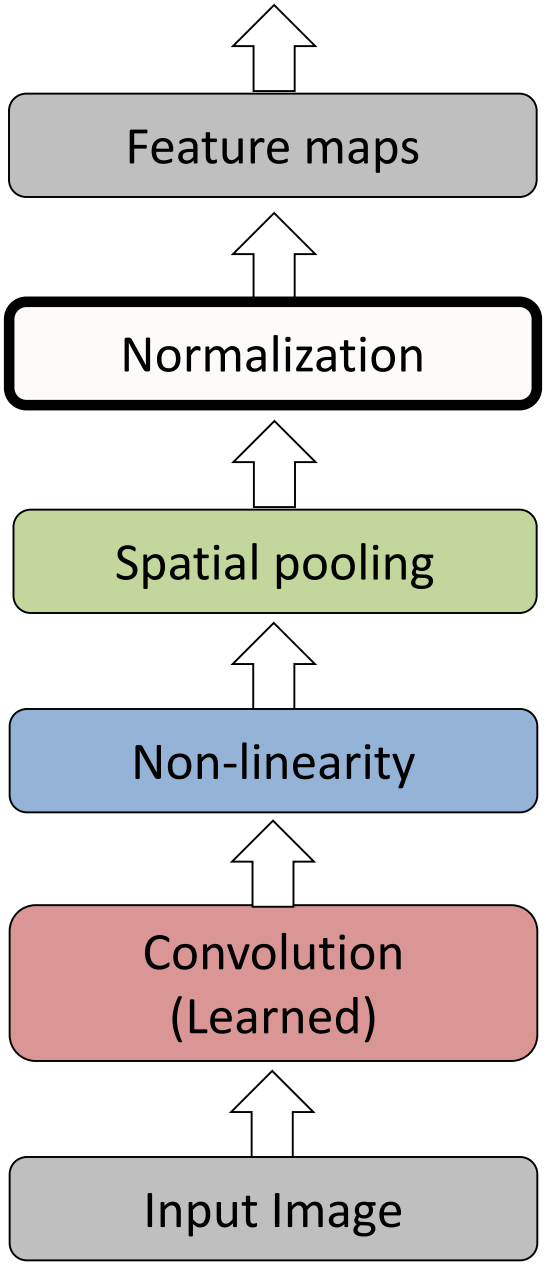


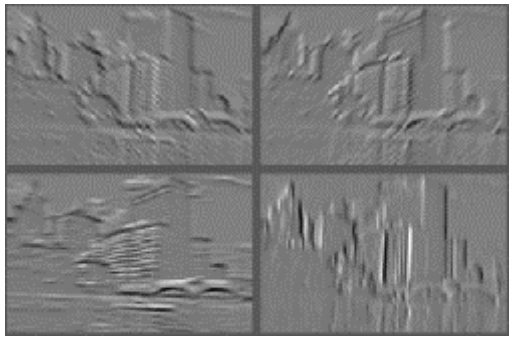
Max-pooling: a non-linear down-sampling

Provide *translation invariance*

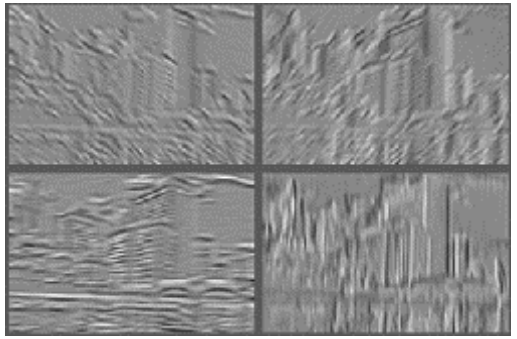
slide credit: S. Lazebnik

Convolutional Neural Networks





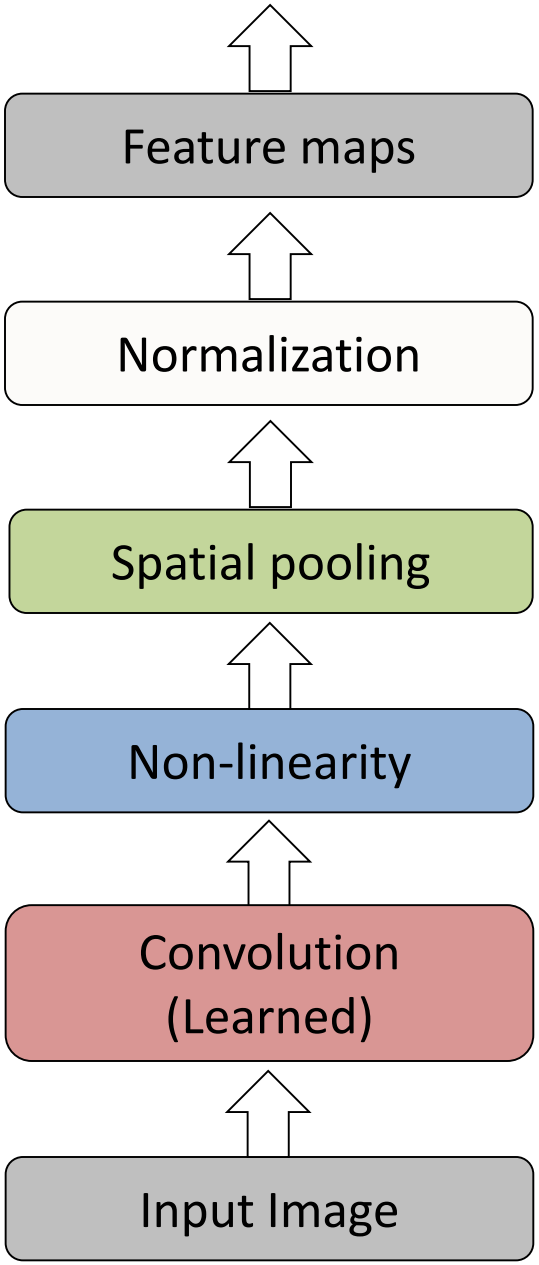
Feature Maps



Feature Maps
After Contrast
Normalization

slide credit: S. Lazebnik

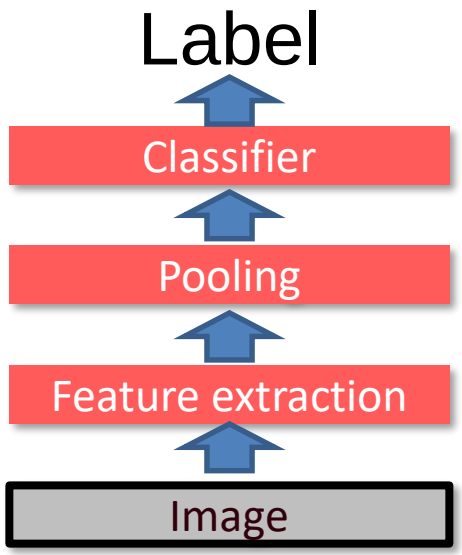
Convolutional Neural Networks

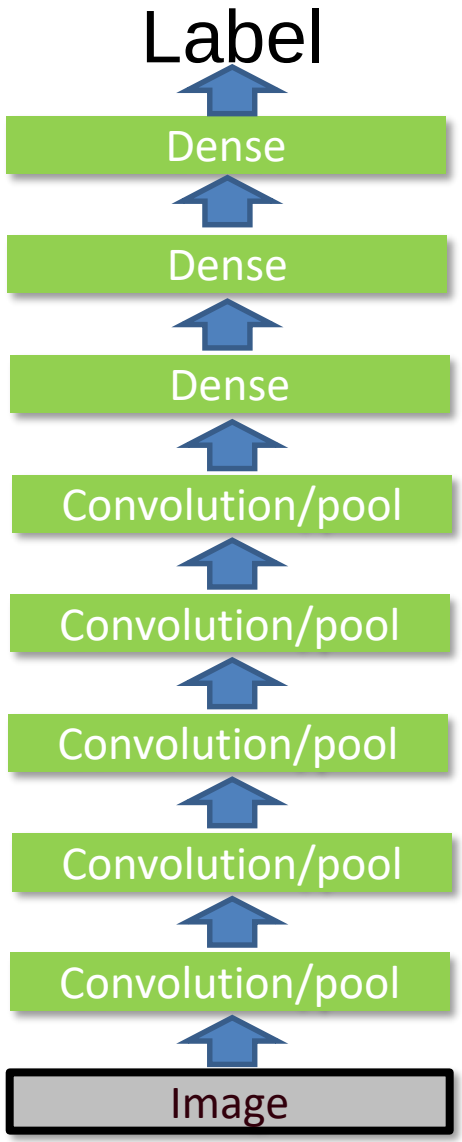


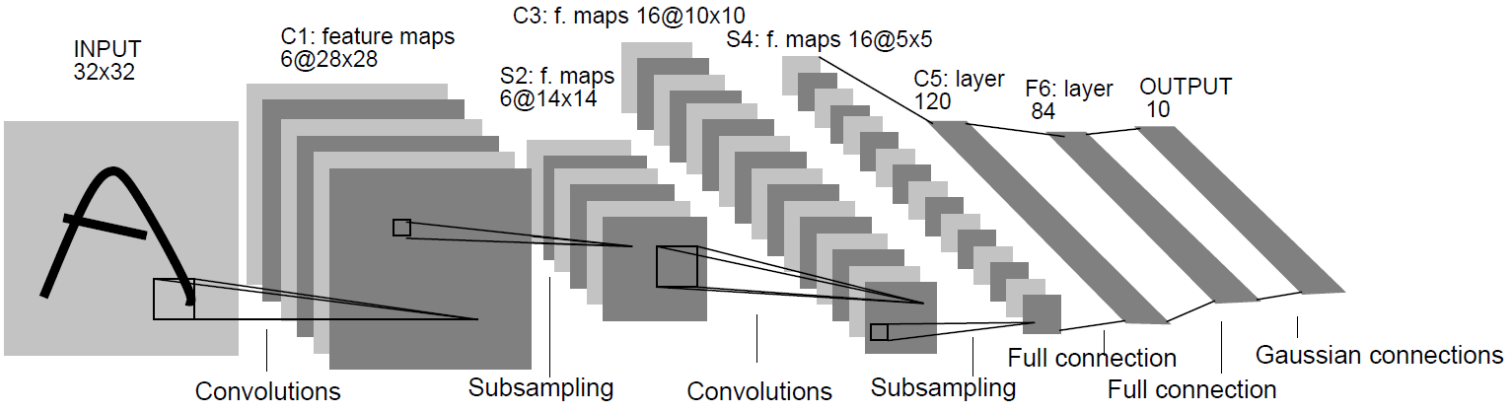
slide credit: S. Lazebnik

Engineered vs. learned features

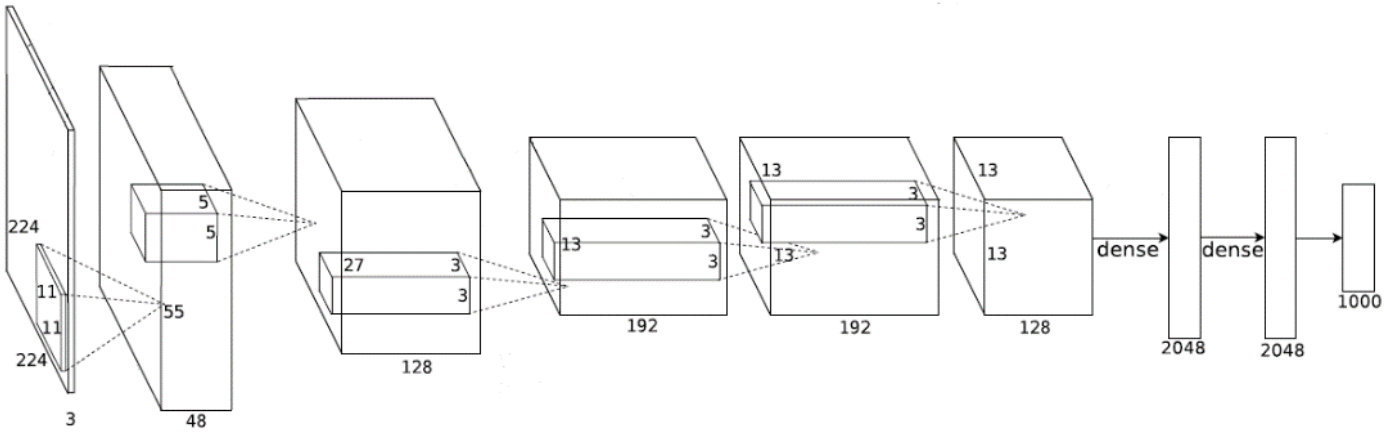
Convolutional filters are trained in a supervised manner by back-propagating classification error





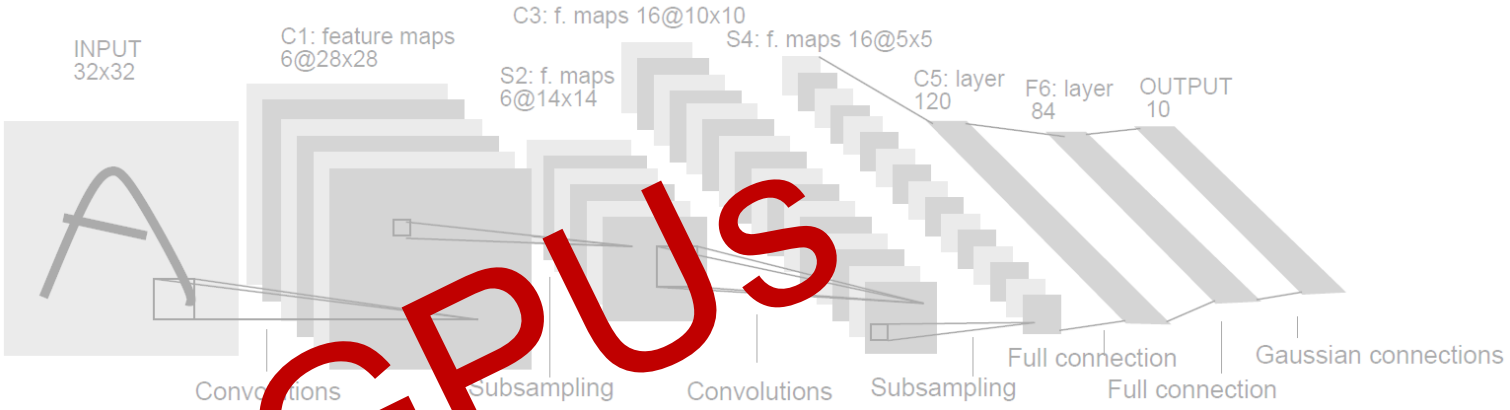


Gradient-Based Learning Applied to Document Recognition, LeCun, Bottou, Bengio and Haffner, Proc. of the IEEE, **1998**



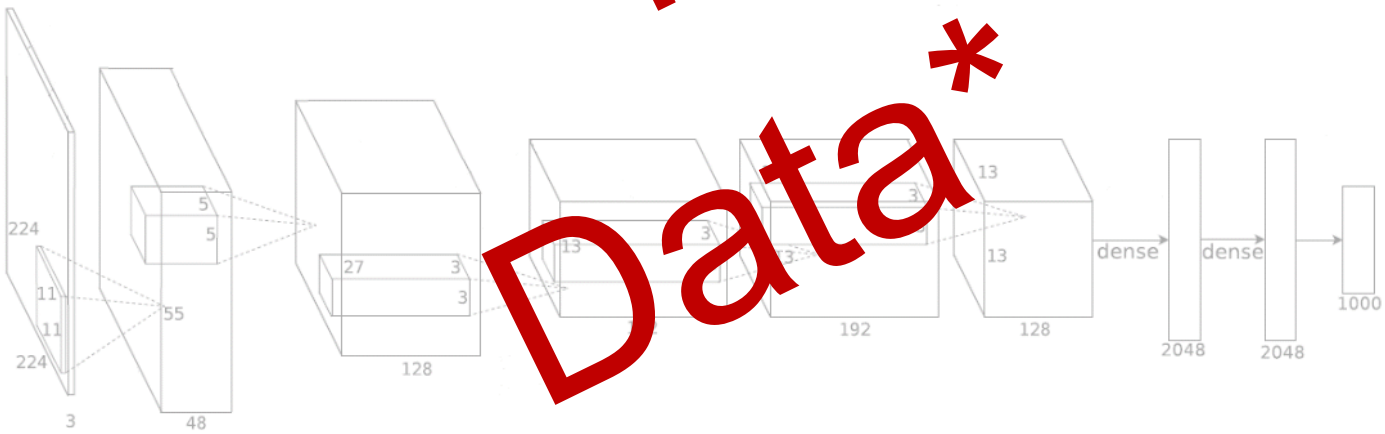
Imagenet Classification with Deep Convolutional Neural Networks, Krizhevsky, Sutskever, and Hinton, NIPS **2012**

Slide Credit: L. Zitnick



Gradient-Based Learning Applied to Document Recognition, LeCun, Bottou, Bengio and Haffner, Proc. of the IEEE, **1998**

GPUs + Data*



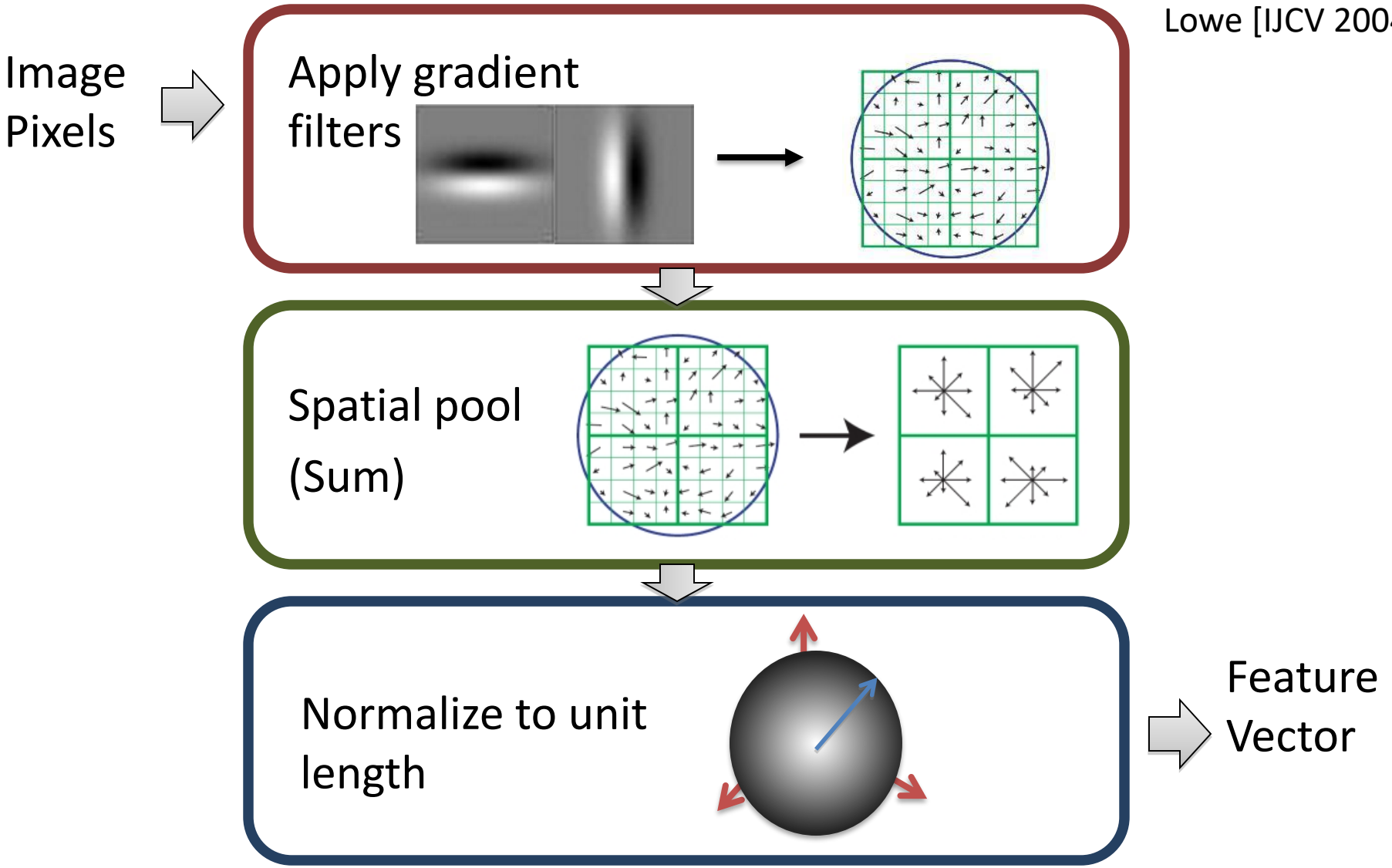
Imagenet Class Networks, Krizh

* Rectified activations and dropout

Slide Credit: L. Zitnick

SIFT Descriptor

Lowe [IJCV 2004]

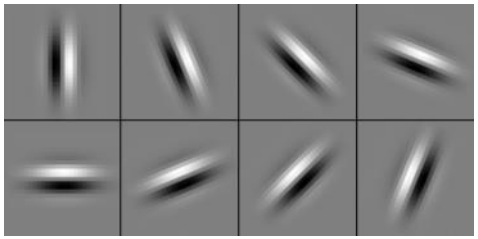


SIFT Descriptor

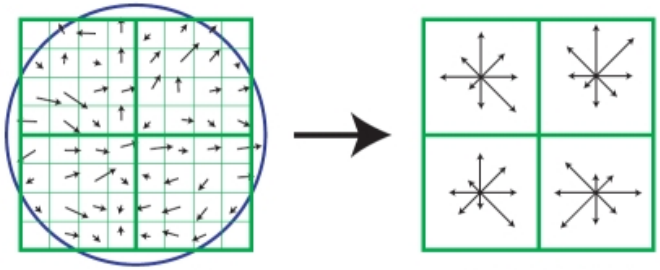
Lowe [IJCV 2004]

Image Pixels

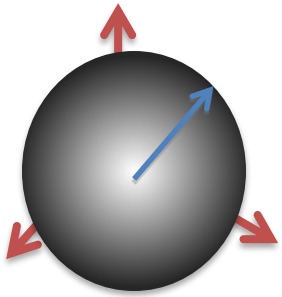
Apply oriented filters



Spatial pool (Sum)



Normalize to unit length



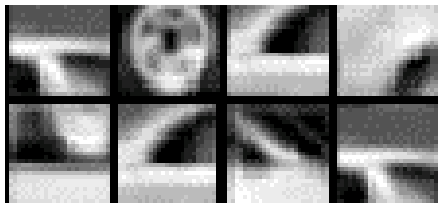
Feature Vector

slide credit: R. Fergus

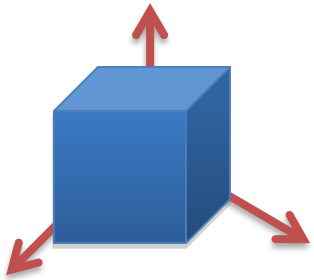
Spatial Pyramid Matching

SIFT Features

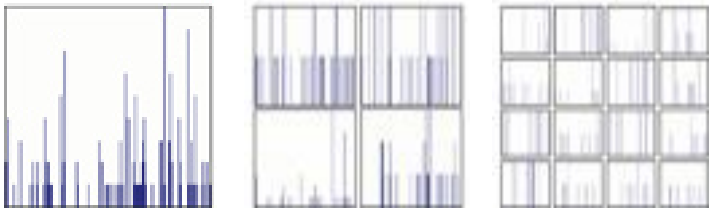
Filter with Visual Words



Max



Multi-scale spatial pool (Sum)

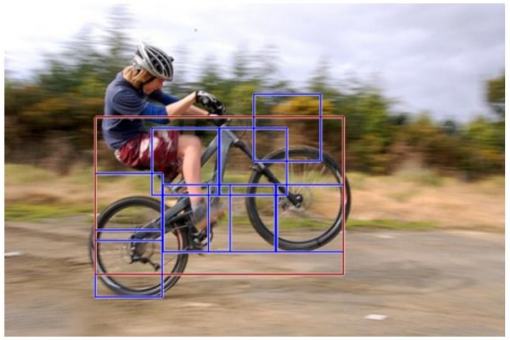


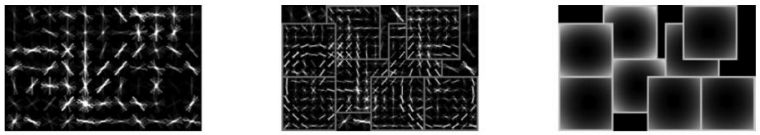
Classifier

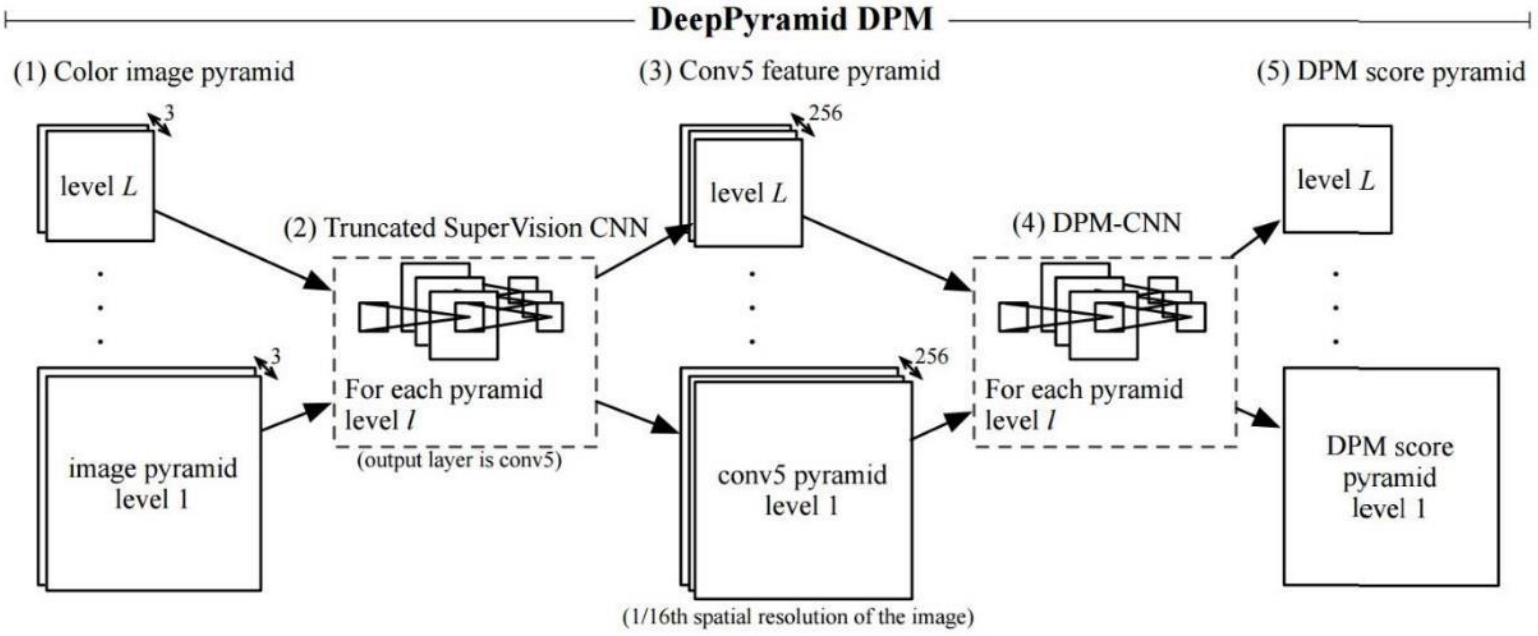
Lazebnik, Schmid, Ponce [CVPR 2006]

slide credit: R. Fergus

Deformable Part Model



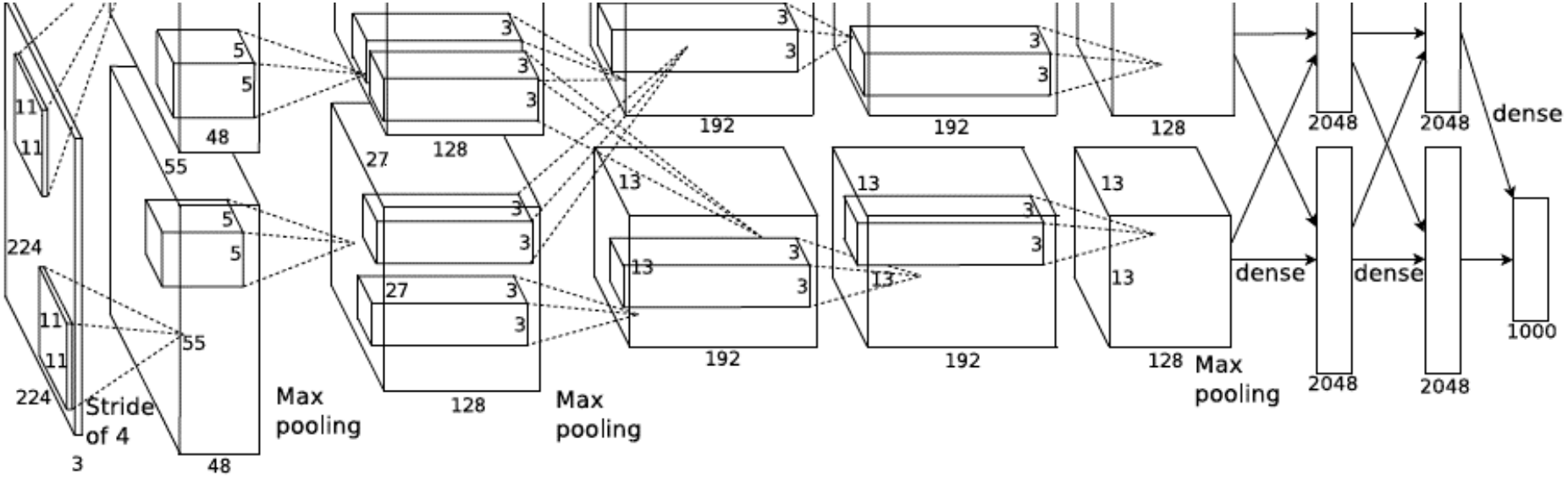




Deformable Part Models are Convolutional Neural Networks [Girshick et al. CVPR 15]

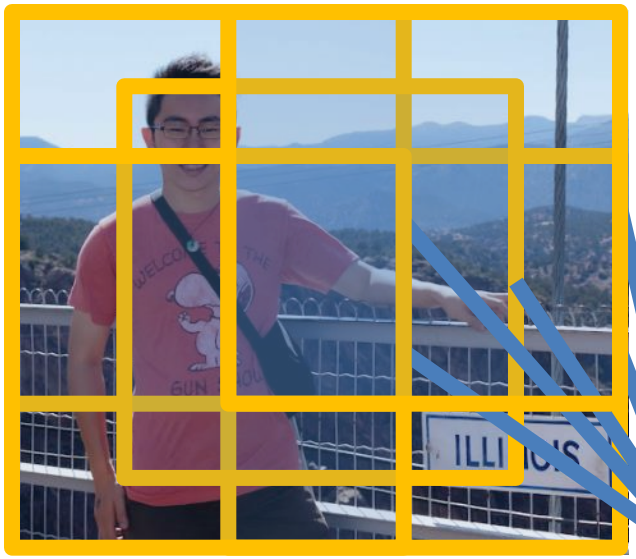
AlexNet

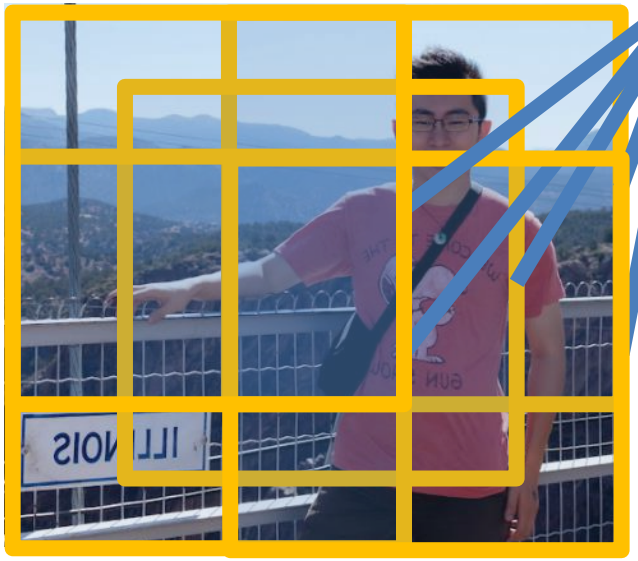
- Similar framework to LeCun'98 but:
  - Bigger model (7 hidden layers, 650,000 units, 60,000,000 params)
  - More data (10^6 vs. 10^3 images)
  - GPU implementation (50x speedup over CPU)
    - Trained on two GPUs for a week



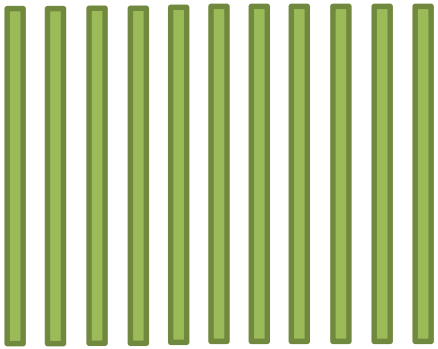
A. Krizhevsky, I. Sutskever, and G. Hinton,
ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012

Using CNN for Image Classification

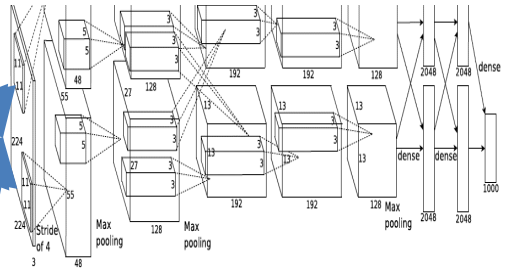




Fully connected layer Fc7
d = 4096



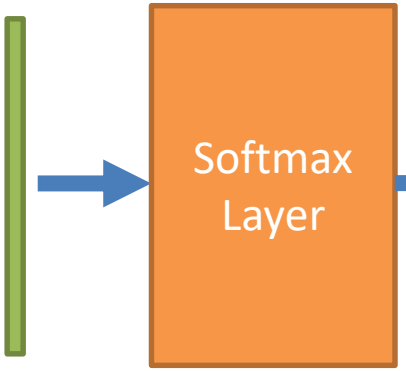
AlexNet



Averaging

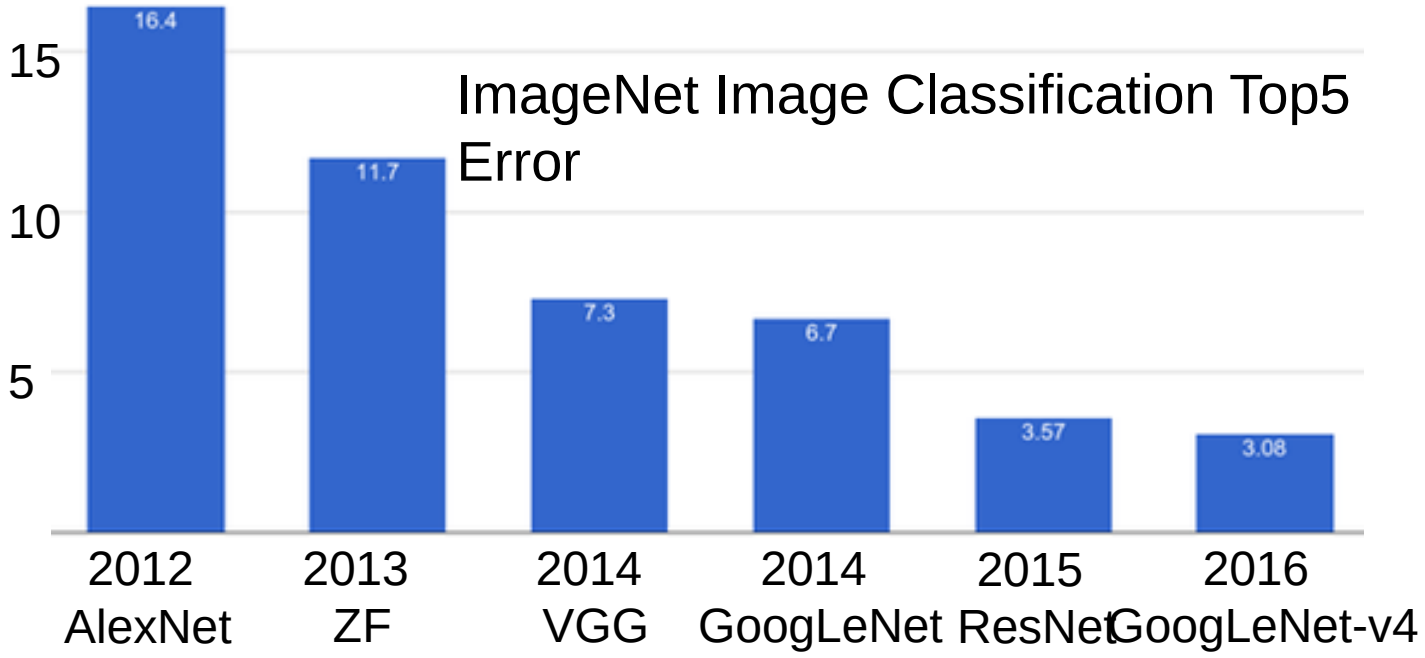
Fixed input size:
224x224x3

d = 4096



"Jia-Bin"

Progress on ImageNet



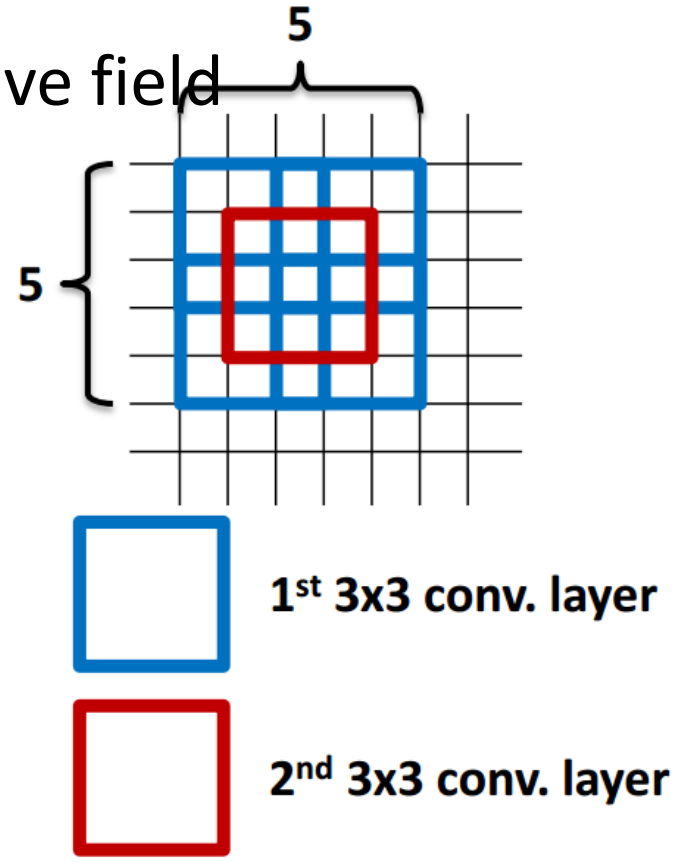
VGG-Net

- The deeper, the better

- Key design choices:
  - 3x3 conv. Kernels
    - very small
  - conv. stride 1
    - no loss of information
- Other details:
  - Rectification (ReLU) non-linearity
  - 5 max-pool layers (x2 reduction)
  - no normalization
  - 3 fully-connected (FC) layers

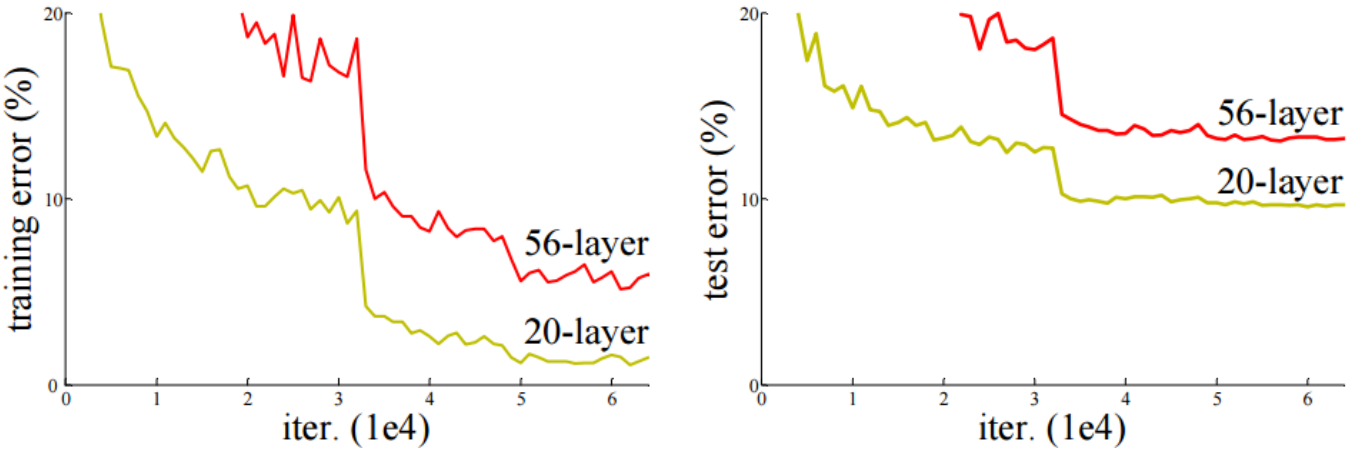
VGG-Net

- Why 3x3 layers?
  - Stacked conv. layers have a large receptive field
  - two 3x3 layers – 5x5 receptive field
  - three 3x3 layers – 7x7 receptive field
- More non-linearity
  - Less parameters to learn
  - ~140M per net

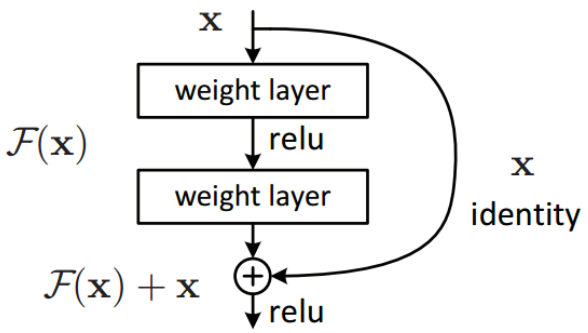


ResNet

- Can we just increase the #layer?



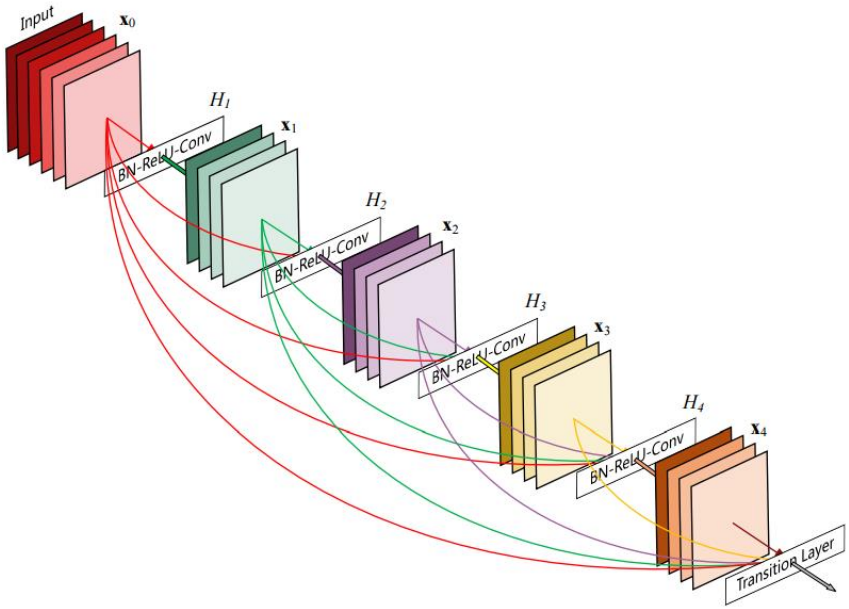
- How can we train very deep network?
  - Residual learning

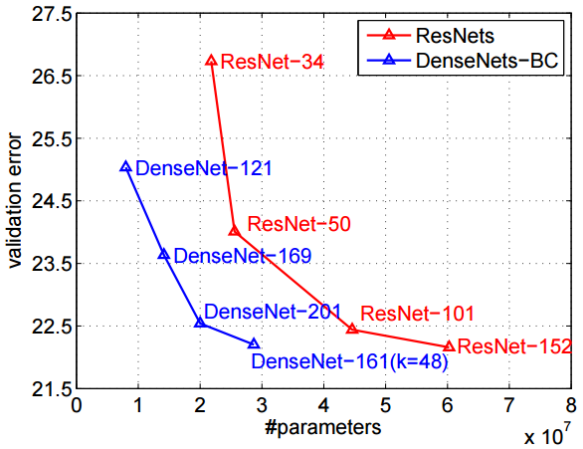


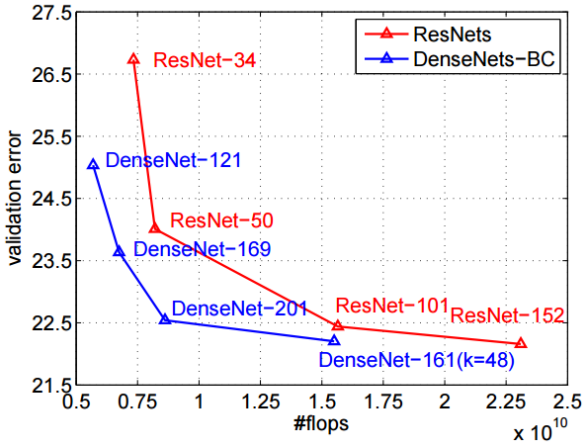
method	top-5 err. (**test**)
VGG [41] (ILSVRC'14)	7.32
GoogLeNet [44] (ILSVRC'14)	6.66
VGG [41] (v5)	6.8
PReLU-net [13]	4.94
BN-inception [16]	4.82
ResNet (ILSVRC'15)	**3.57**

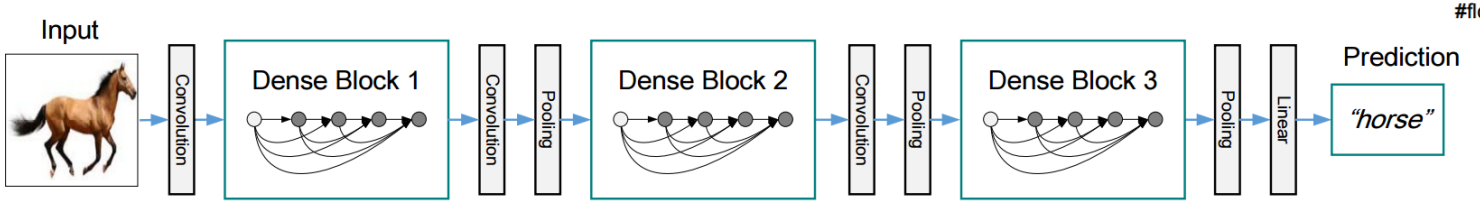
DenseNet

- Shorter connections (like ResNet) help
- Why not just connect them all?









Training Convolutional Neural Networks

- Backpropagation + stochastic gradient descent with momentum
  - Neural Networks: Tricks of the Trade
- Dropout
- Data augmentation
- Batch normalization
- Initialization
  - Transfer learning

Training CNN with gradient descent

- A CNN as composition of functions

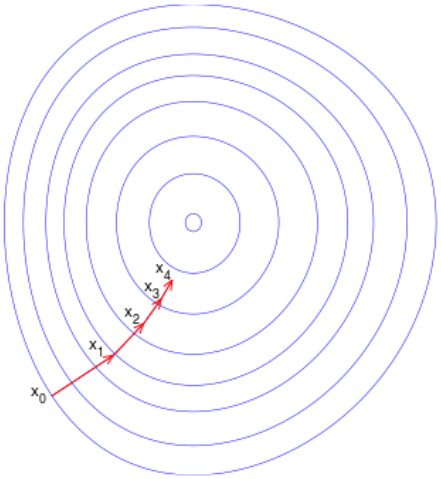
$$f_{\boldsymbol{w}}(\boldsymbol{x}) = f_L(\dots\ (f_2(f_1(\boldsymbol{x}; \boldsymbol{w}_1); \boldsymbol{w}_2)\ \dots; \boldsymbol{w}_L)$$

- Parameters

$$\boldsymbol{w} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \dots \boldsymbol{w}_L)$$

- Empirical loss function

$$L(\boldsymbol{w}) = \frac{1}{n}\sum_i l(z_i, f_{\boldsymbol{w}}(\boldsymbol{x}_i))$$



- Gradient descent

$$\boldsymbol{w}^{t+1} = \boldsymbol{w}^t - \eta_t \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}}(\boldsymbol{w}^t)$$

New weight

Old weight

Learning rate

Gradient

An Illustrative example

$$f(x, y) = xy, \qquad \frac{\partial f}{\partial x} = y, \frac{\partial f}{\partial y} = x$$

Example: $x = 4, y = -3 \Rightarrow f(x, y) = -12$

Partial derivatives

$$\frac{\partial f}{\partial x} = -3, \qquad \frac{\partial f}{\partial y} = 4$$

Gradient

$$\nabla f = [\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}]$$

Example credit: Andrej Karpathy

$$f(x, y, z) = (x + y)z = qz$$

$q = x + y$

$\frac{\partial q}{\partial x} = 1, \qquad \frac{\partial q}{\partial y} = 1$

$f = qz$

$\frac{\partial f}{\partial q} = z, \qquad \frac{\partial f}{\partial z} = q$

Goal: compute the gradient

$$\nabla f = [\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}]$$

Example credit: Andrej Karpathy

$$f(x, y, z) = (x + y)z = qz$$

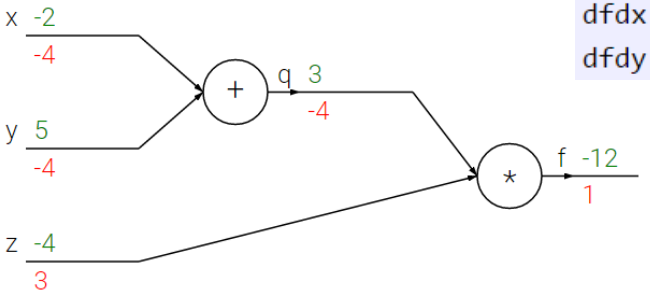
$$q = x + y$$
$$\frac{\partial q}{\partial x} = 1, \qquad \frac{\partial q}{\partial y} = 1$$

$$f = qz$$
$$\frac{\partial f}{\partial q} = z, \qquad \frac{\partial f}{\partial z} = q$$

Chain rule:

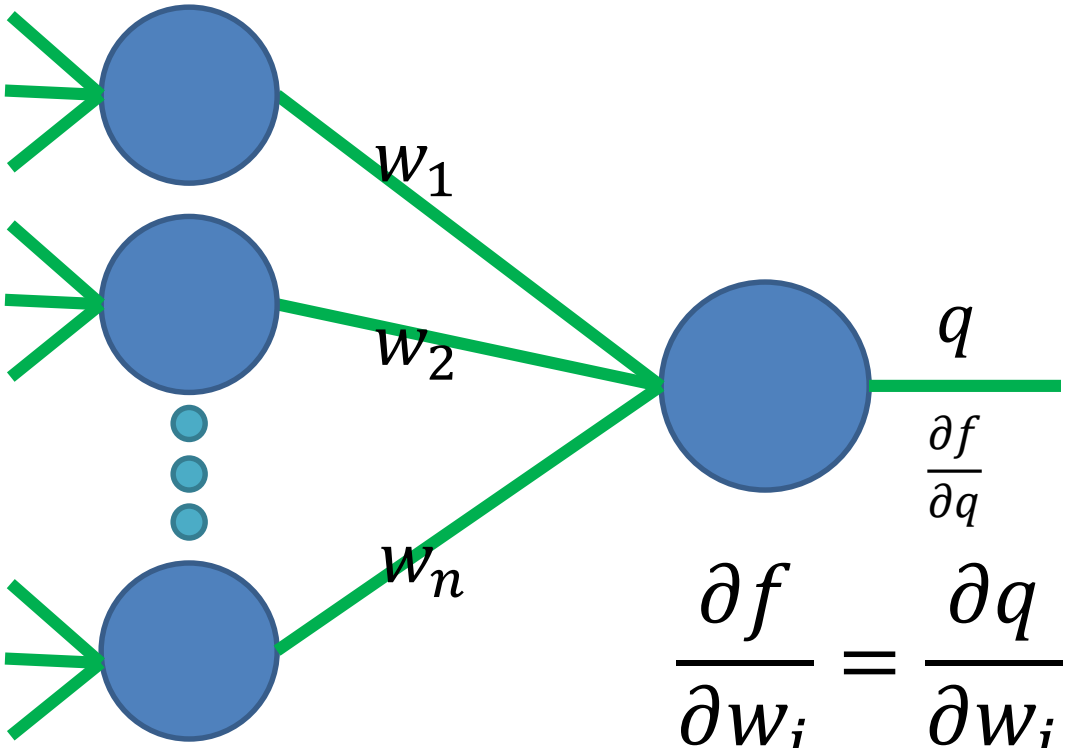
$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial x}$$

```
# set some inputs
x = -2; y = 5; z = -4

# perform the forward pass
q = x + y # q becomes 3
f = q * z # f becomes -12

# perform the backward pass (backpropagation) in reverse order:
# first backprop through f = q * z
dfdz = q # df/dz = q, so gradient on z becomes 3
dfdq = z # df/dq = z, so gradient on q becomes -4
# now backprop through q = x + y
dfdx = 1.0 * dfdq # dq/dx = 1. And the multiplication here is the chain rule!
dfdy = 1.0 * dfdq # dq/dy = 1
```



Example credit: Andrej Karpathy

Backpropagation (recursive chain rule)



$$\frac{\partial f}{\partial w_i} = \frac{\partial q}{\partial w_i}\frac{\partial f}{\partial q}$$

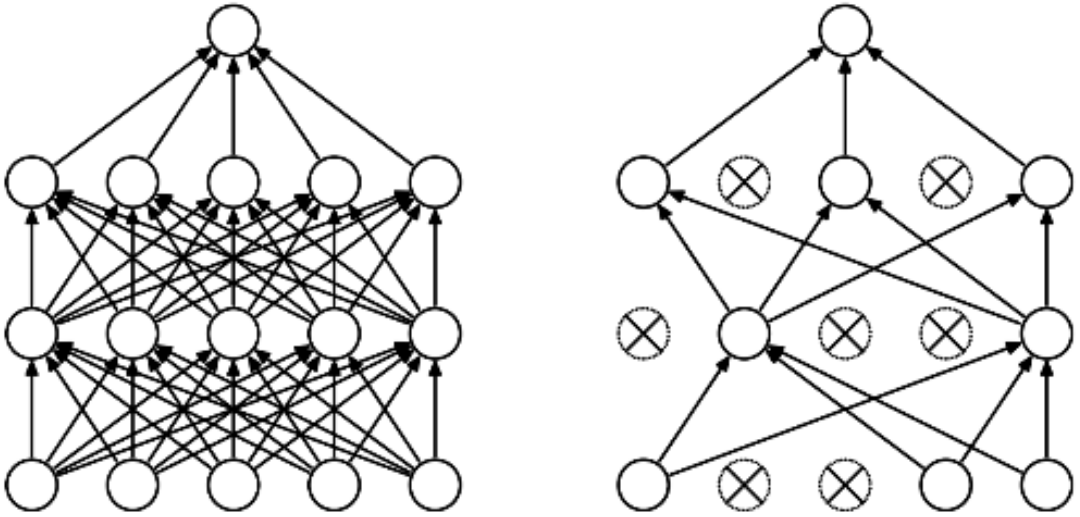
Local gradient

Gate gradient

Can be computed during forward pass

The gate receives this during backprop

Dropout



(a) Standard Neural Net (b) After applying dropout.

Intuition: successful conspiracies

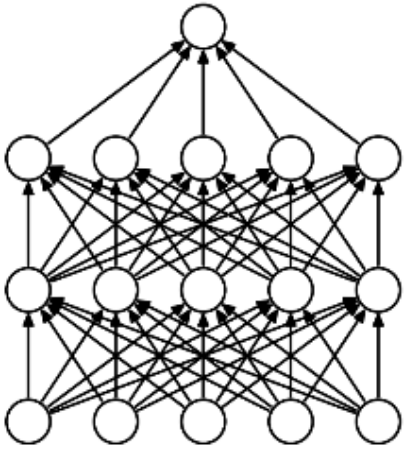
- 50 people planning a conspiracy

- Strategy A: plan a big conspiracy involving 50 people
  - Likely to fail. 50 people need to play their parts correctly.

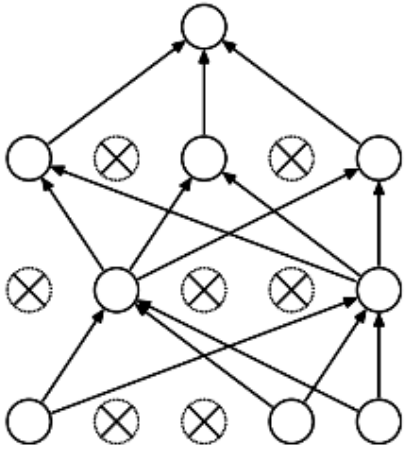
- Strategy B: plan 10 conspiracies each involving 5 people
  - Likely to succeed!

Dropout: A simple way to prevent neural networks from overfitting [Srivastava JMLR 2014]

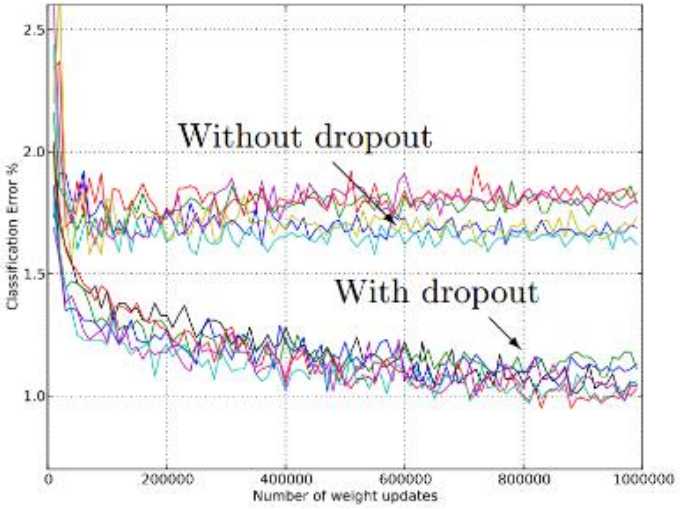
Dropout

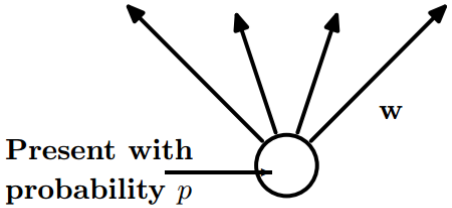


(a) Standard Neural Net

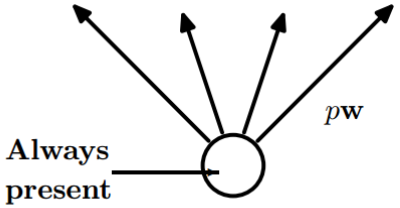


(b) After applying dropout.





(a) At training time



(b) At test time

Main Idea: approximately combining exponentially many different neural network architectures efficiently

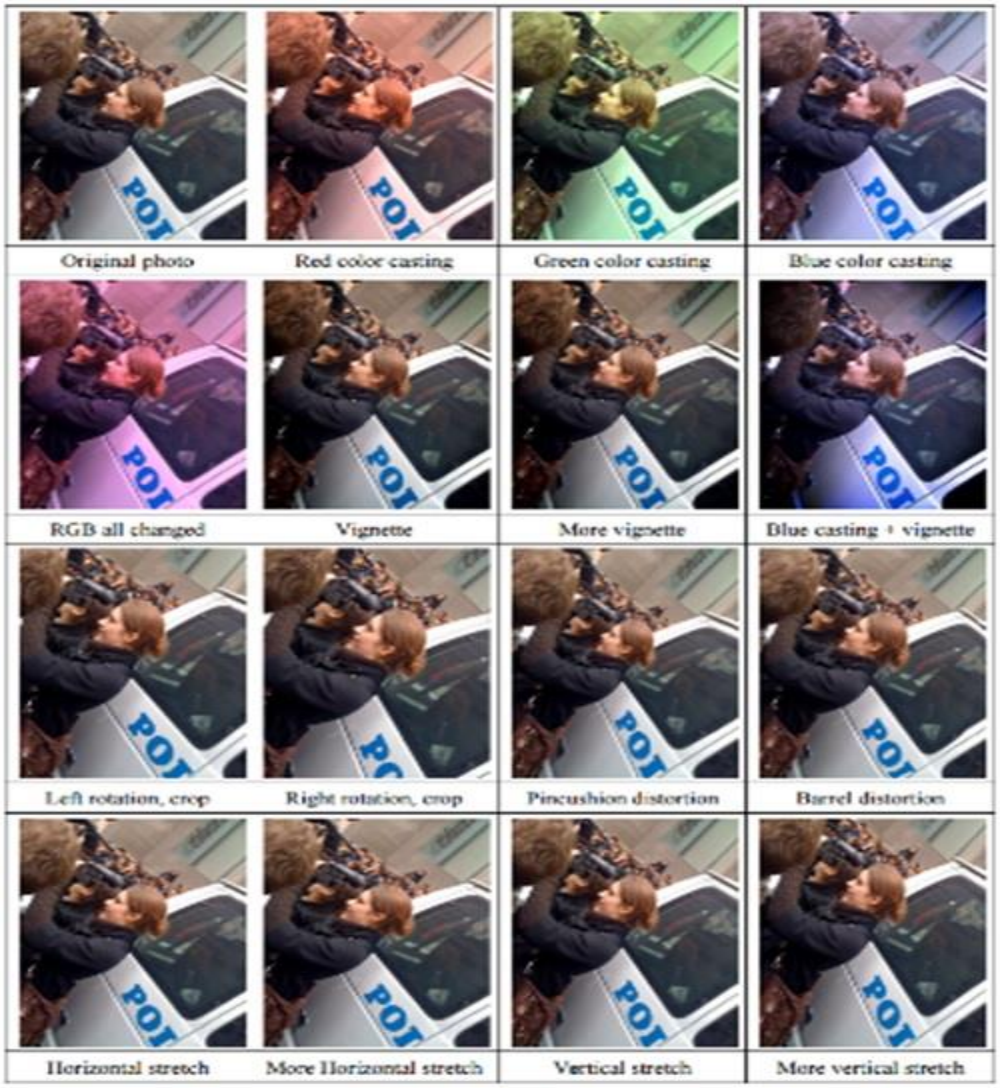
Model	Top-1 (val)	Top-5 (val)	Top-5 (test)
SVM on Fisher Vectors of Dense SIFT and Color Statistics	-	-	27.3
Avg of classifiers over FVs of SIFT, LBP, GIST and CSIFT	-	-	26.2
Conv Net + dropout (Krizhevsky et al., 2012)	40.7	18.2	-
Avg of 5 Conv Nets + dropout (Krizhevsky et al., 2012)	38.1	16.4	16.4

Table 6: Results on the ILSVRC-2012 validation/test set.

Dropout: A simple way to prevent neural networks from overfitting [Srivastava JMLR 2014]

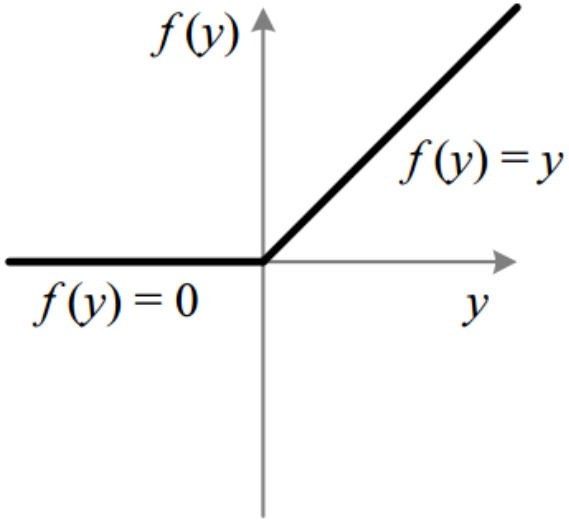
Data Augmentation (Jittering)

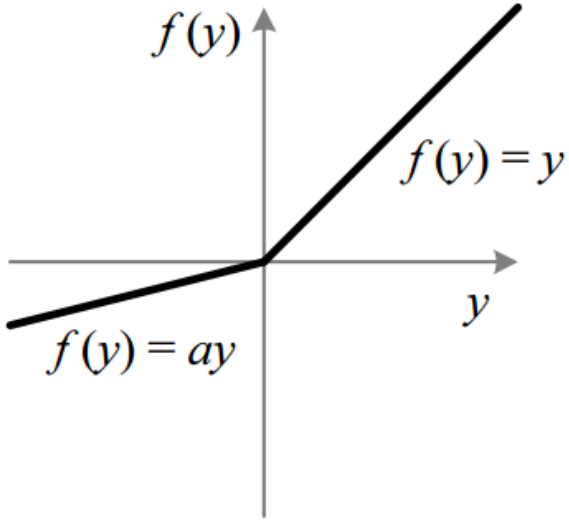
- Create *virtual* training samples
  - Horizontal flip
  - Random crop
  - Color casting
  - Geometric distortion



Deep Image [Wu et al. 2015]

Parametric Rectified Linear Unit

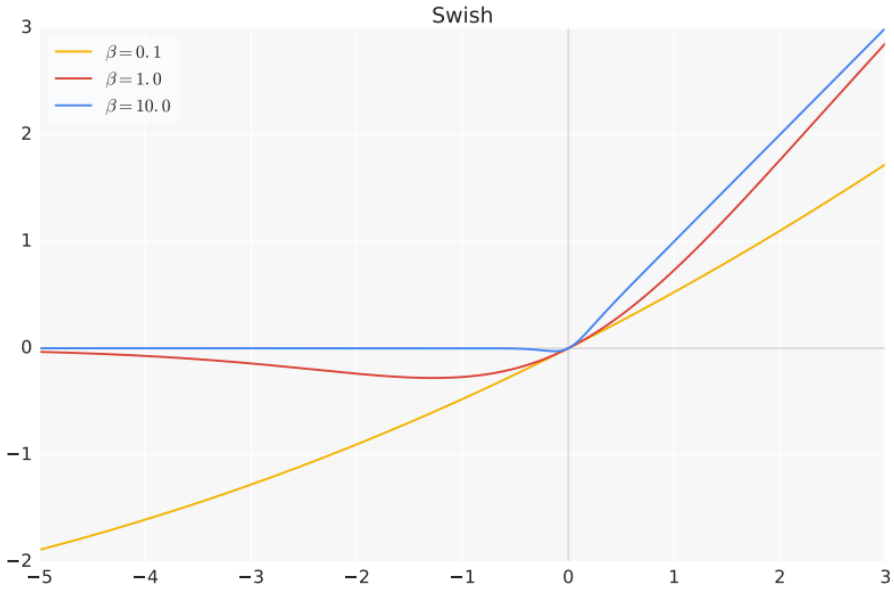




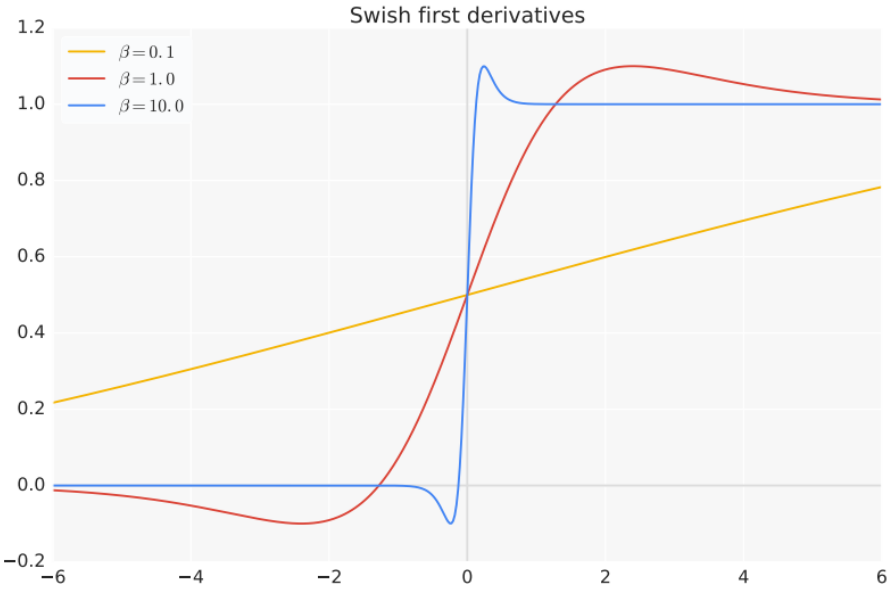
	team	top-5 (**test**)
in competition ILSVRC 14	MSRA, SPP-nets [11]	8.06
	VGG [25]	7.32
	GoogLeNet [29]	6.66
post-competition	VGG [25] (arXiv v5)	6.8
	Baidu [32]	5.98
	MSRA, PReLU-nets	**4.94**

Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification [He et al. 2015]

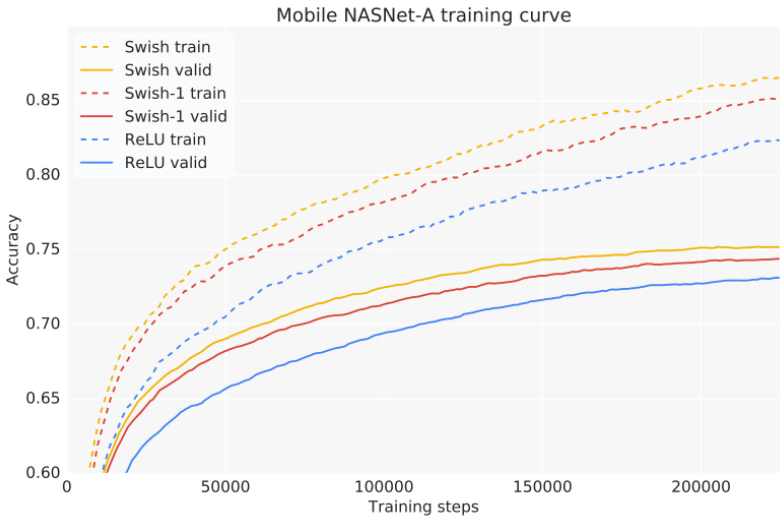
Swish



The Swish activation function



First derivatives of Swish



Model	Top-1 Acc. (%)			Top-5 Acc. (%)		
LReLU	73.8	73.9	74.2	91.6	91.9	91.9
PReLU	74.6	74.7	74.7	92.4	92.3	92.3
Softplus	74.0	74.2	74.2	91.6	91.8	91.9
ELU	74.1	74.2	74.2	91.8	91.8	91.8
SELU	73.6	73.7	73.7	91.6	91.7	91.7
GELU	74.6	-	-	92.0	-	-
ReLU	73.5	73.6	73.8	91.4	91.5	91.6
Swish-1	74.6	74.7	74.7	92.1	92.0	92.0
Swish	74.9	74.9	75.2	92.3	92.4	92.4

https://arxiv.org/abs/1710.05941

Batch Normalization

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β

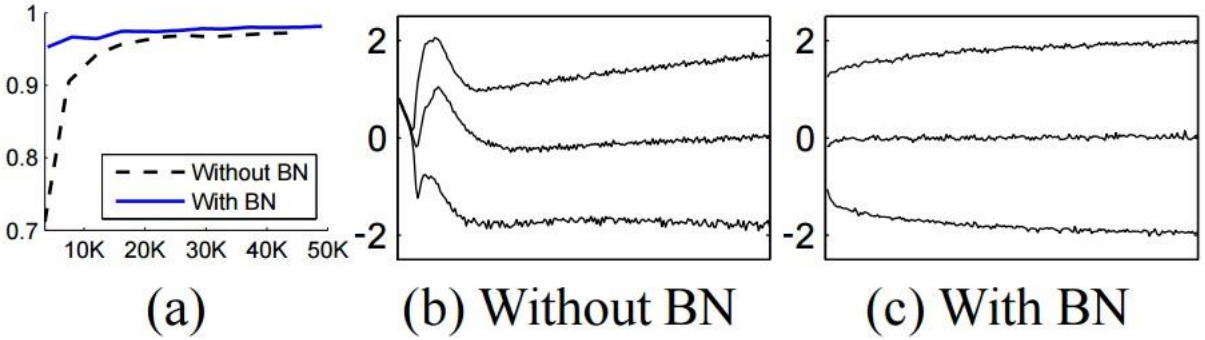
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$



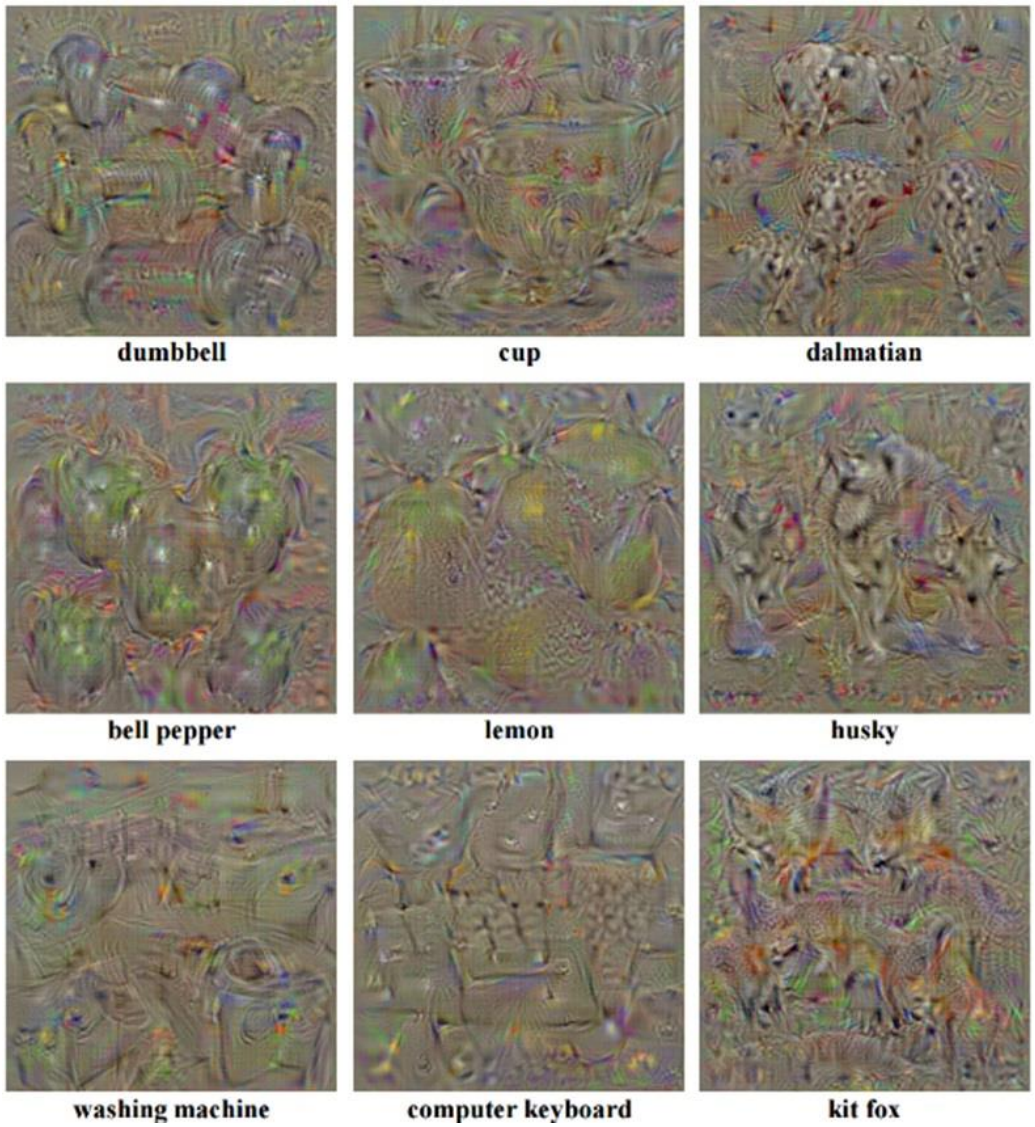
(a) (b) Without BN (c) With BN

Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift [Ioffe and Szegedy 2015]

Understanding and Visualizing CNN

- Find images that maximize some class scores
- Individual neuron activation
- Breaking CNNs

Find images that maximize some class scores



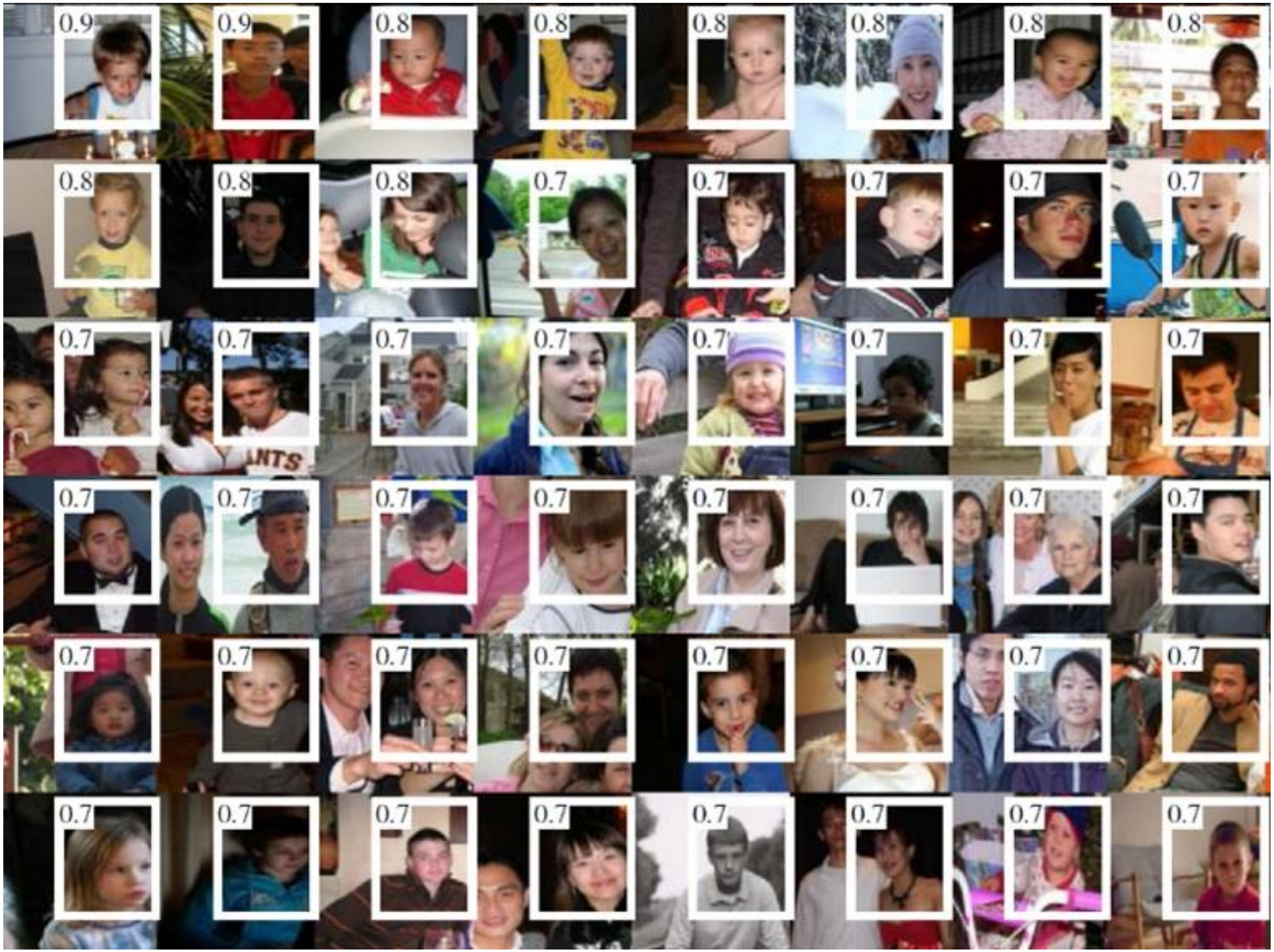
person: HOG template

Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps
[Simonyan et al. ICLR Workshop 2014]

Individual Neuron Activation

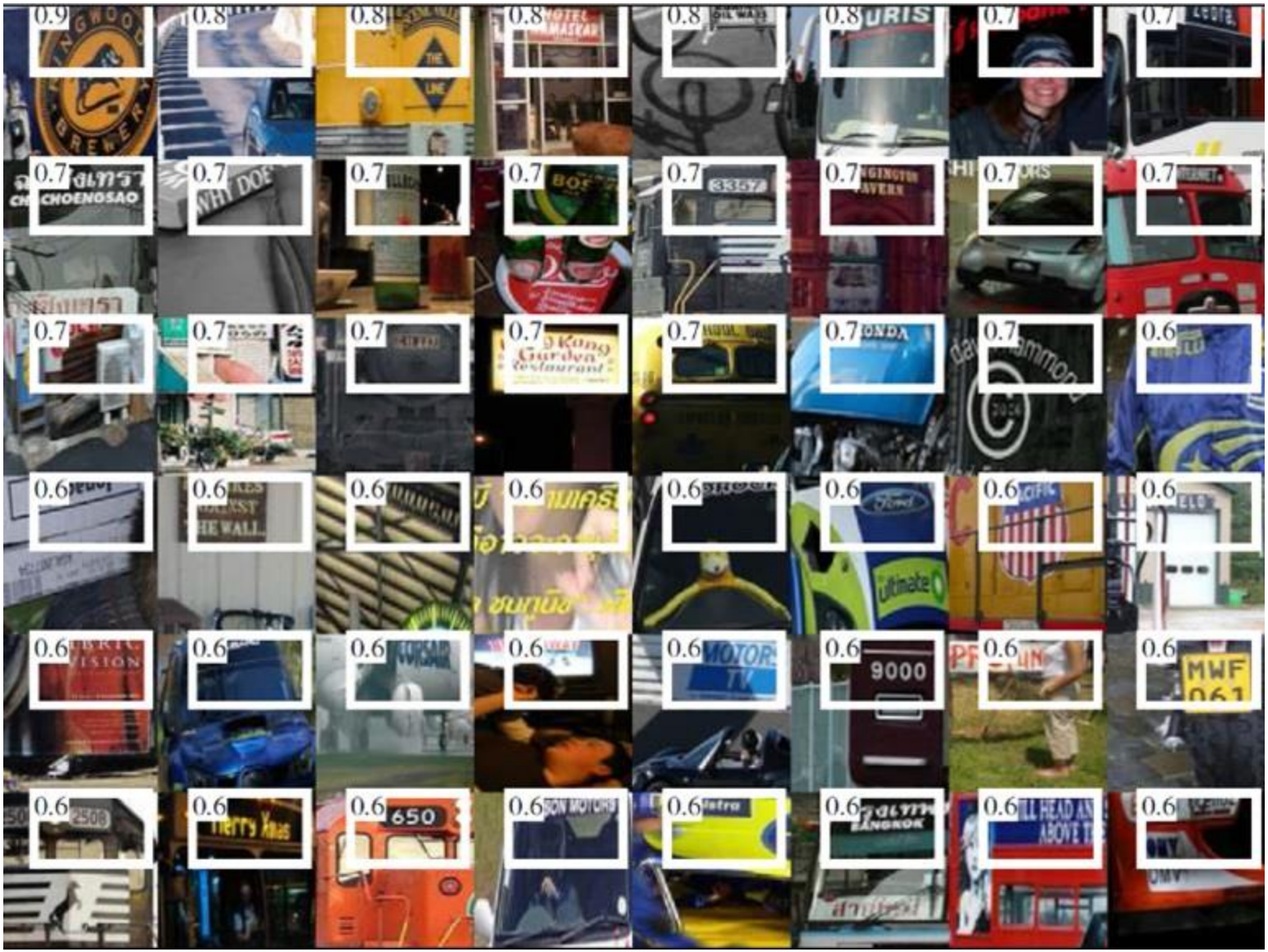
RCNN [Girshick et al. CVPR 2014]

Individual Neuron Activation



RCNN [Girshick et al. CVPR 2014]

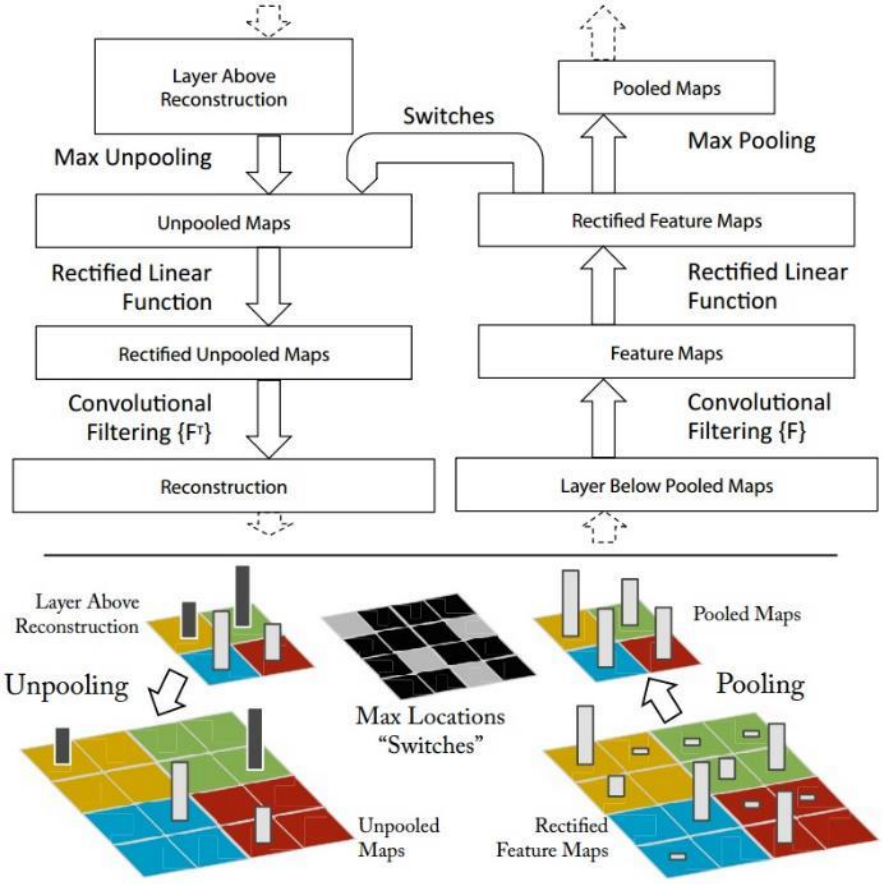
Individual Neuron Activation



RCNN [Girshick et al. CVPR 2014]

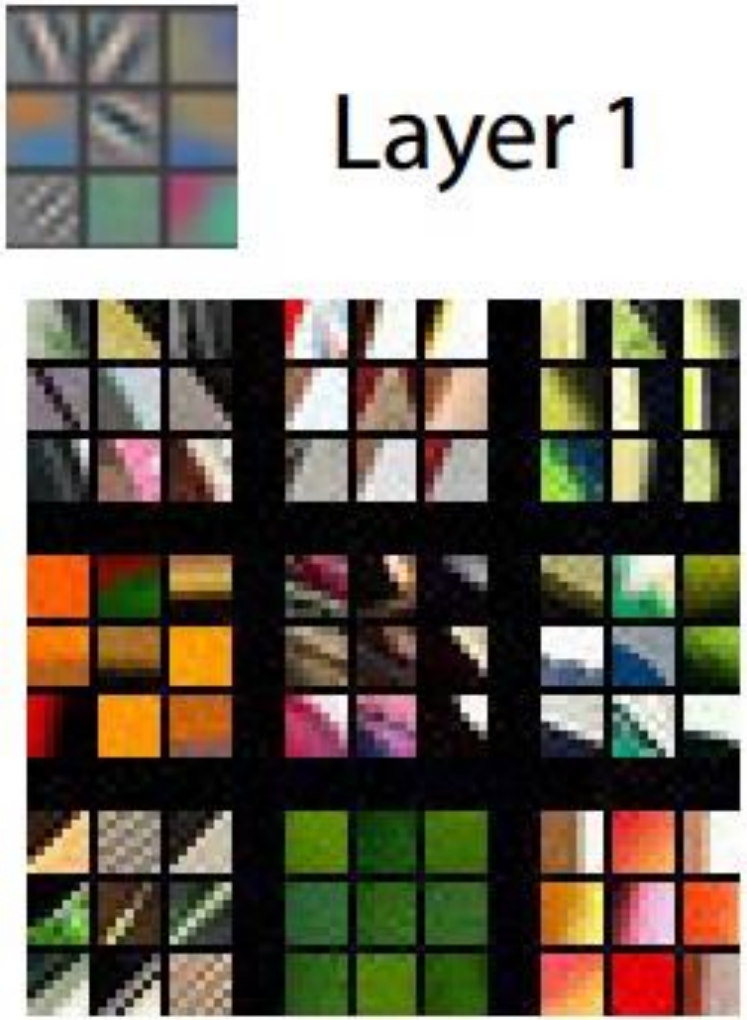
Map activation back to the input pixel space

- What input pattern originally caused a given activation in the feature maps?



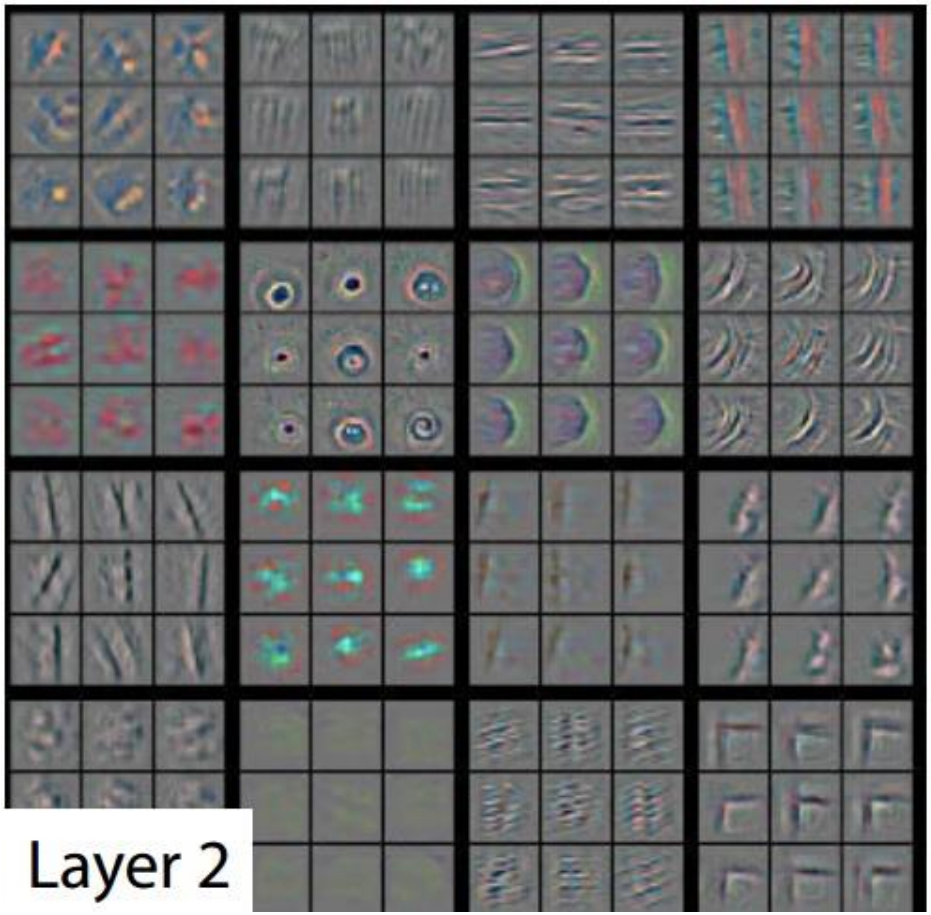
Visualizing and Understanding Convolutional Networks [Zeiler and Fergus, ECCV 2014]

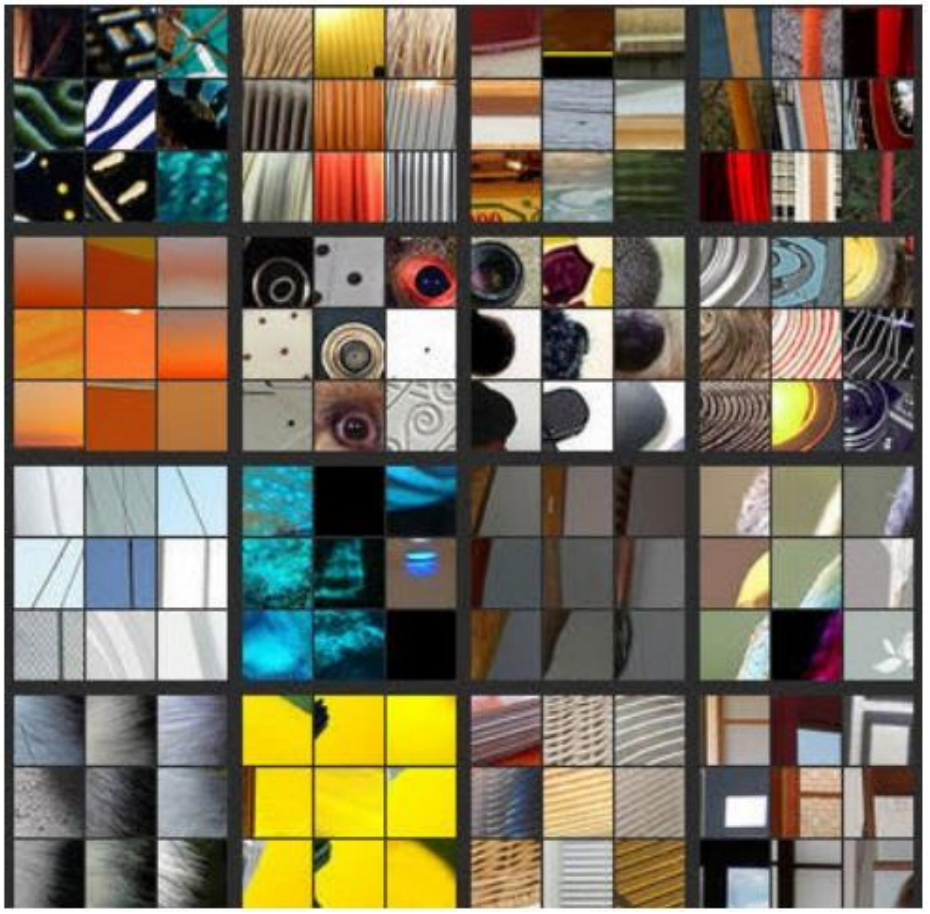
Layer 1



Visualizing and Understanding Convolutional Networks [Zeiler and Fergus, ECCV 2014]

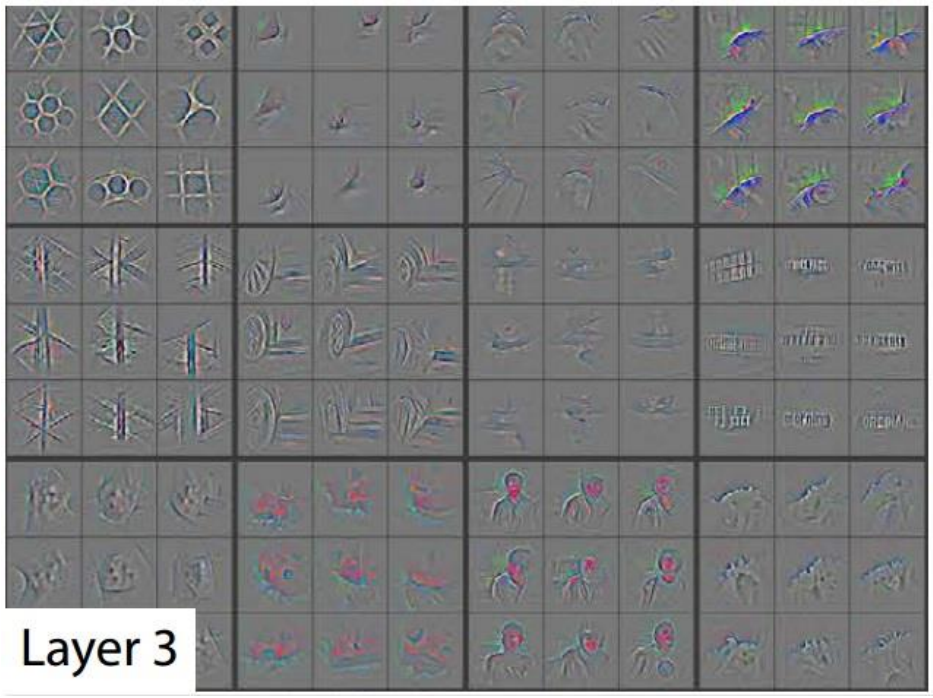
Layer 2





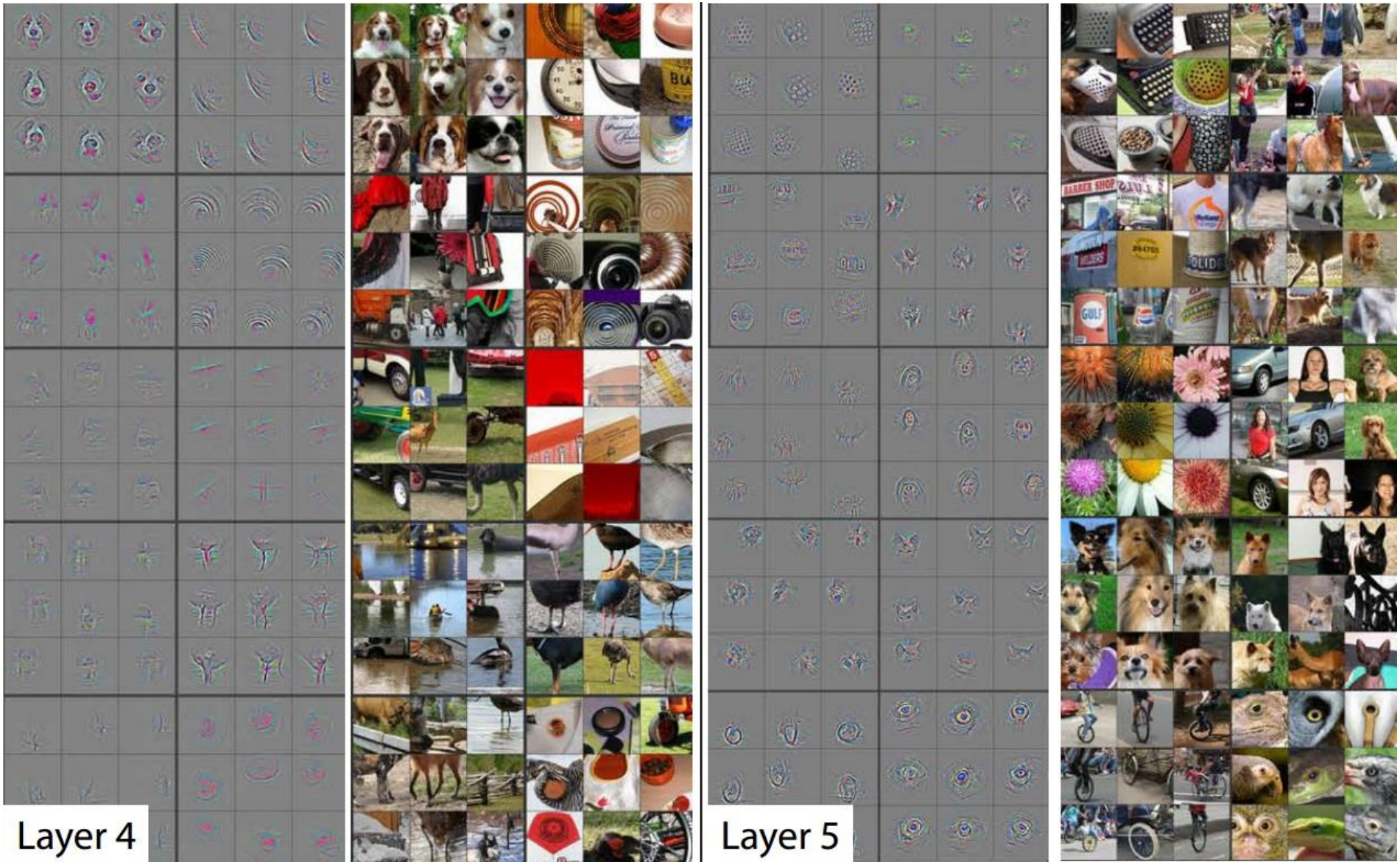
Visualizing and Understanding Convolutional Networks [Zeiler and Fergus, ECCV 2014]

Layer 3



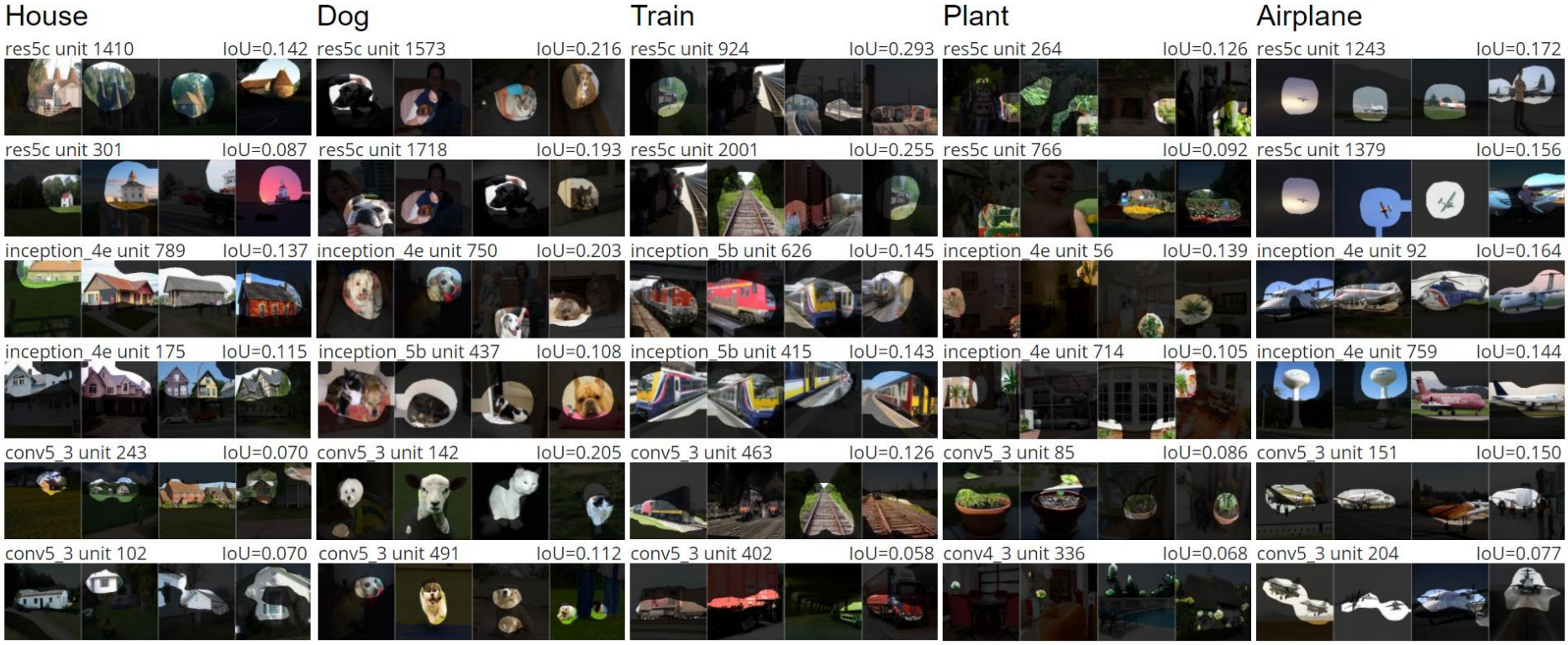
Visualizing and Understanding Convolutional Networks [Zeiler and Fergus, ECCV 2014]

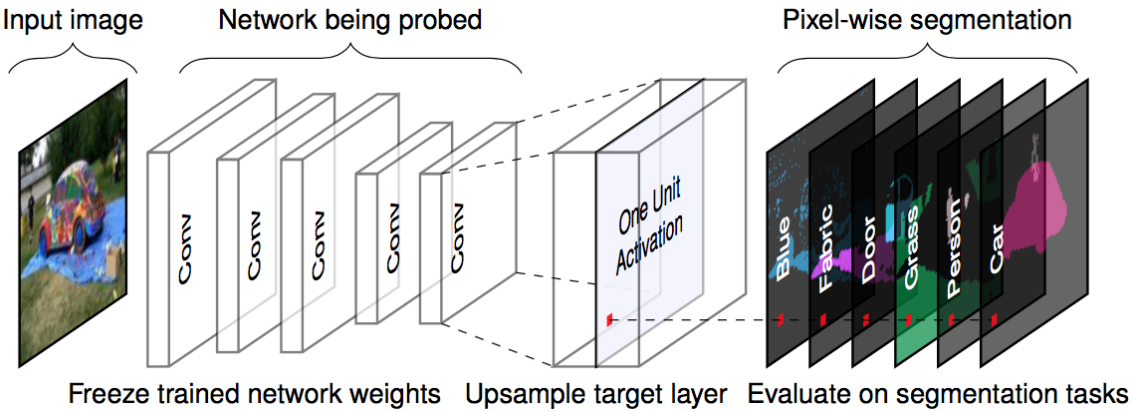
Layer 4 and 5



Visualizing and Understanding Convolutional Networks [Zeiler and Fergus, ECCV 2014]

Network Dissection





http://netdissect.csail.mit.edu/

Deep learning library

- TensorFlow
  - Research + Production

- PyTorch
  - Research

- Caffe2
  - Production

Things to remember

- Convolutional neural networks
  - A cascade of conv + ReLU + pool
  - Representation learning
  - Advanced architectures
  - Tricks for training CNN
- Visualizing CNN
  - Activation
  - Dissection

Resources

- http://deeplearning.net/
  - Hub to many other deep learning resources
- https://github.com/ChristosChristofidis/awesome-deep-learning
  - A resource collection deep learning
- https://github.com/kjw0612/awesome-deep-vision
  - A resource collection deep learning for computer vision
- http://cs231n.stanford.edu/syllabus.html
  - Nice course on CNN for visual recognition

Things to remember

- Overview
  - Neuroscience, Perceptron, multi-layer neural networks
- Convolutional neural network (CNN)
  - Convolution, nonlinearity, max pooling
  - CNN for classification and beyond
- Understanding and visualizing CNN
  - Find images that maximize some class scores; visualize individual neuron activation, input pattern and images; breaking CNNs
- Training CNN
  - Dropout; data augmentation; batch normalization; transfer learning


xtranormal