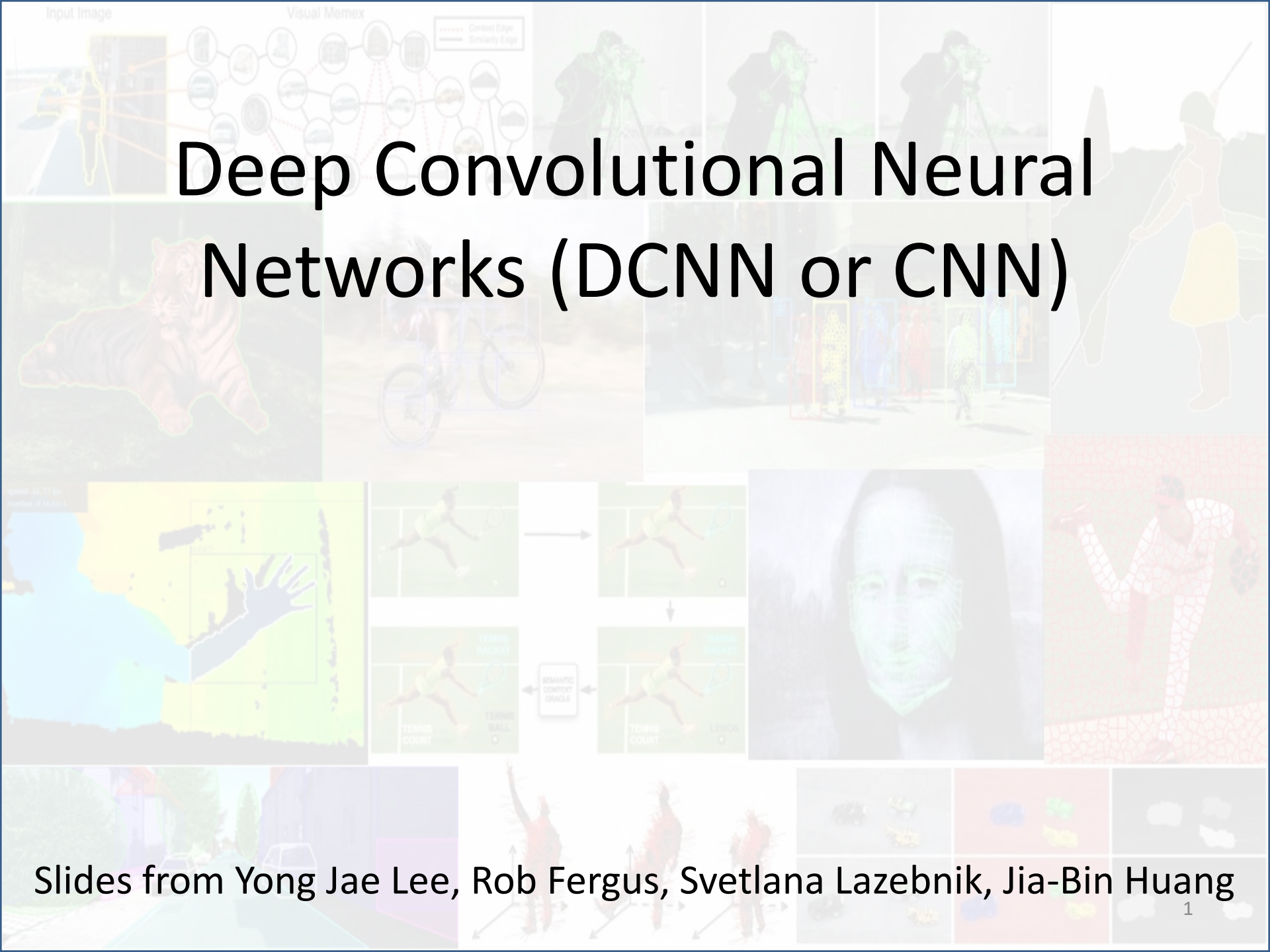
Deep Convolutional Neural Networks (DCNN or CNN)

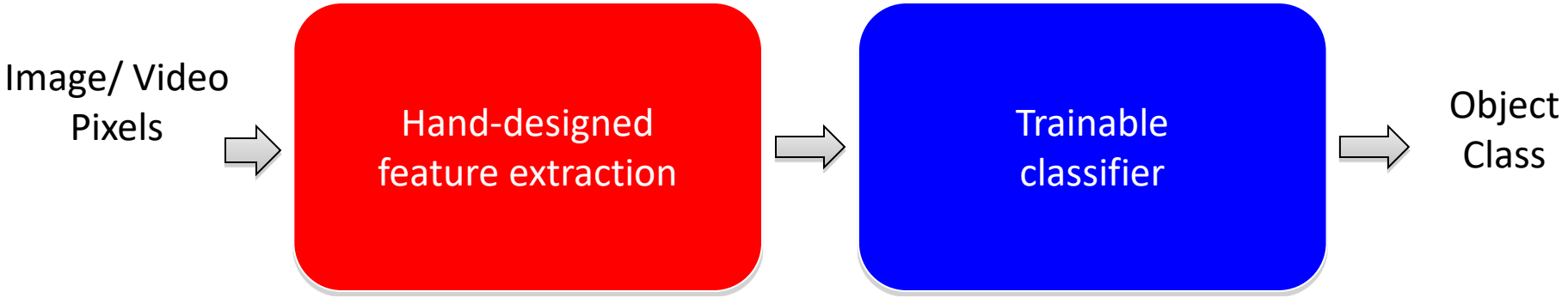


Slides from Yong Jae Lee, Rob Fergus, Svetlana Lazebnik, Jia-Bin Huang

Overview

- Background
- Convolutional Neural Networks (CNNs)
- Understanding and Visualizing CNNs
- Applications
- Packages

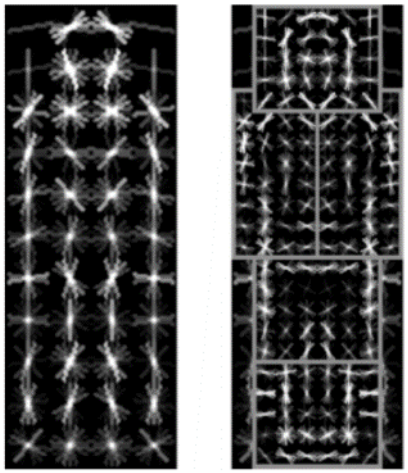
Traditional Recognition Approach



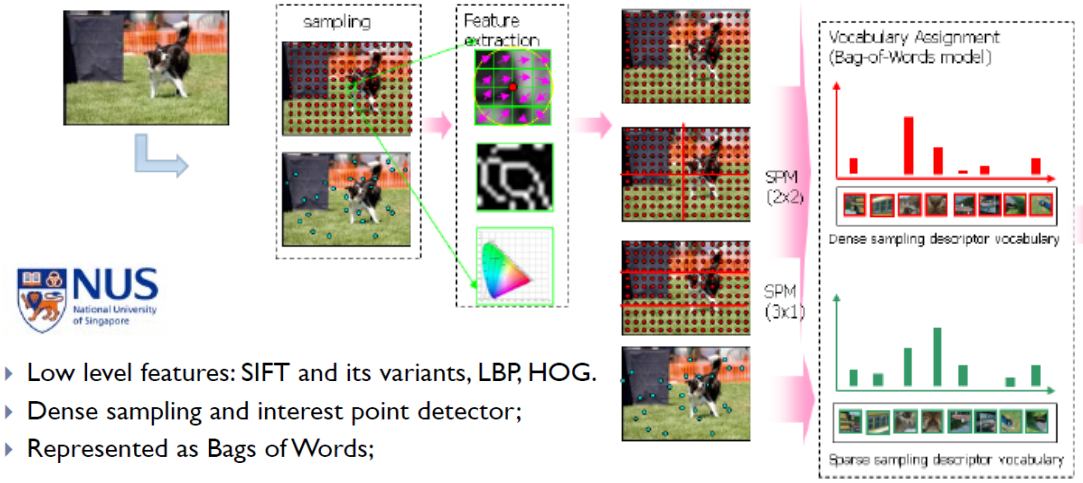
- Features are not learned
- Trainable classifier is often generic (e.g. SVM)

Traditional Recognition Approach

- Features are key to recent progress in recognition
- Multitude of hand-designed features currently in use
  - SIFT, HOG, ……
- Where next? Better classifiers? Or keep building more features?

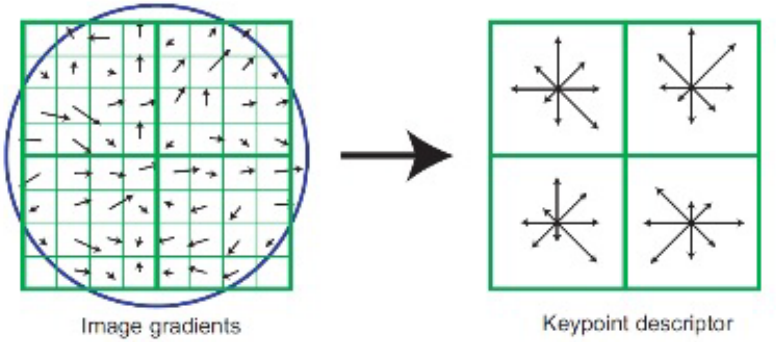


Felzenszwalb, Girshick,
McAllester and Ramanan, PAMI 2007

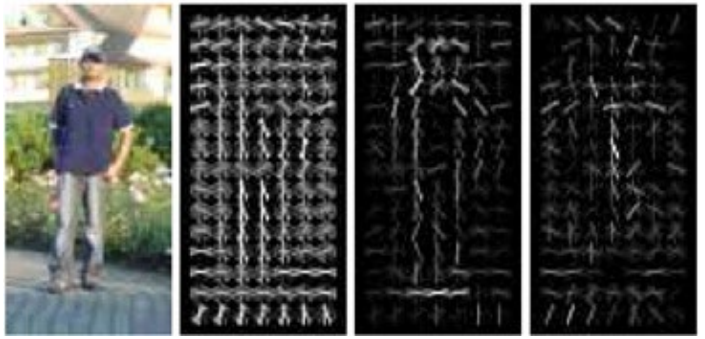


Yan & Huang
(Winner of PASCAL 2010 classification competition)

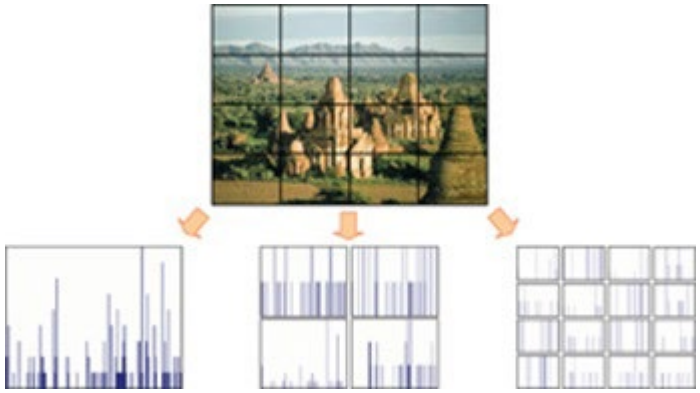
Features are key to recent progress in recognition



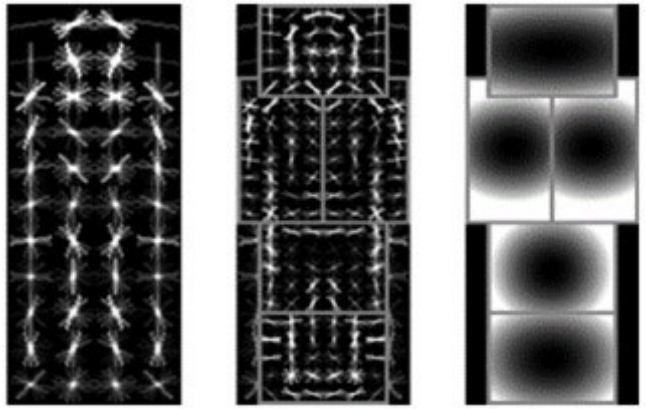
SIFT [Loewe IJCV 04]



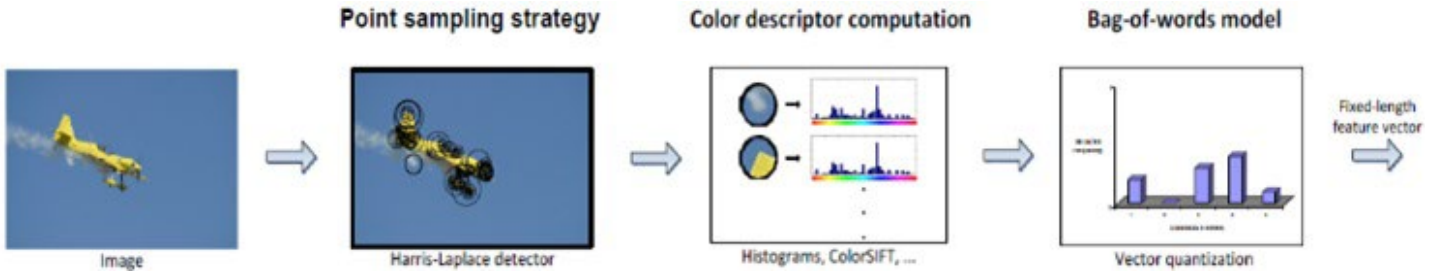
HOG [Dalal and Triggs CVPR 05]



SPM [Lazebnik et al. CVPR 06]



DPM [Felzenszwalb et al. PAMI 10]



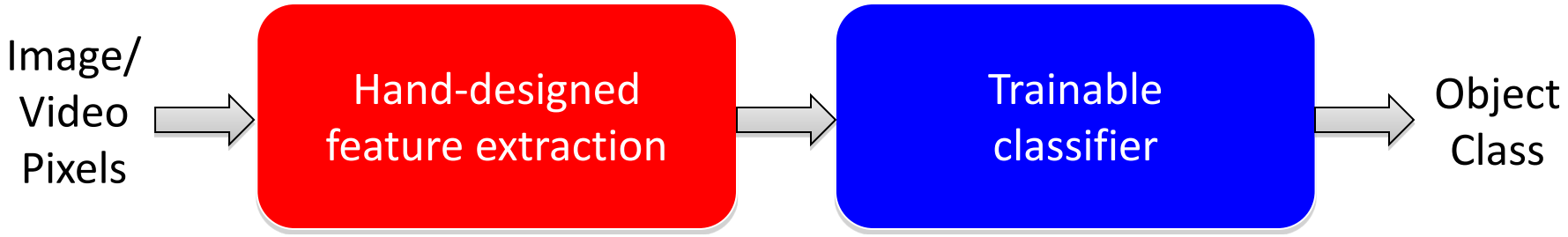
Color Descriptor [Van De Sande et al. PAMI 10]

What about learning the features?

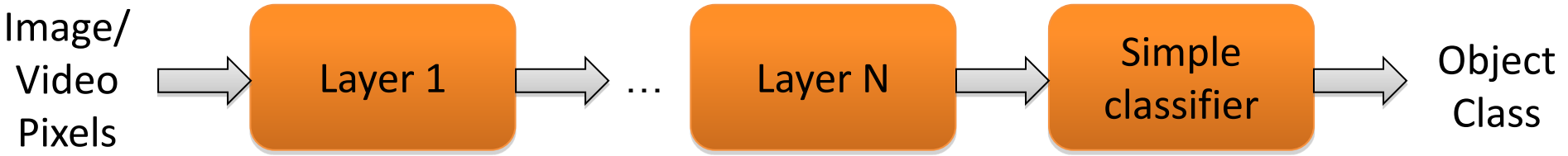
- Learn a *feature hierarchy* all the way from pixels to classifier
- Each layer extracts features from the output of previous layer
- Layers have (nearly) the same structure
- Train all layers jointly

“Shallow” vs. “deep” architectures

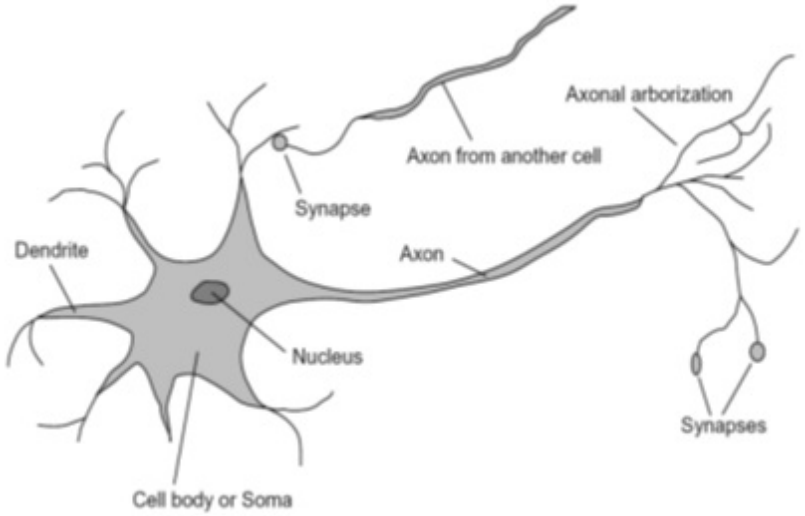
Traditional recognition: “Shallow” architecture



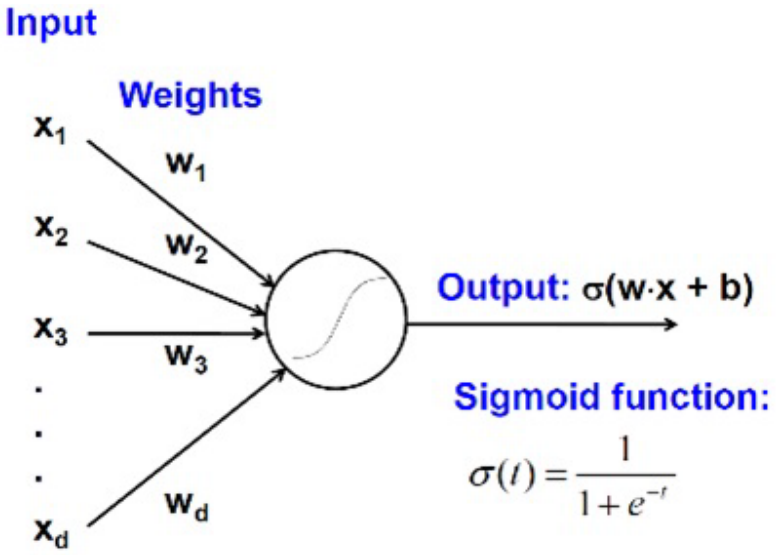
Deep learning: “Deep” architecture



Biological neuron and Perceptrons



A biological neuron



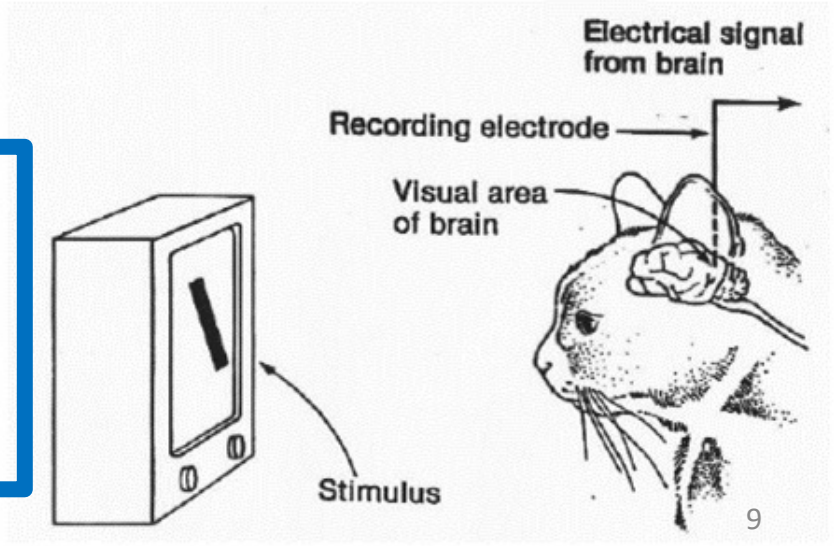
An artificial neuron (Perceptron)
- a linear classifier

Simple, Complex and Hyper-complex cells

David H. Hubel and Torsten Wiesel

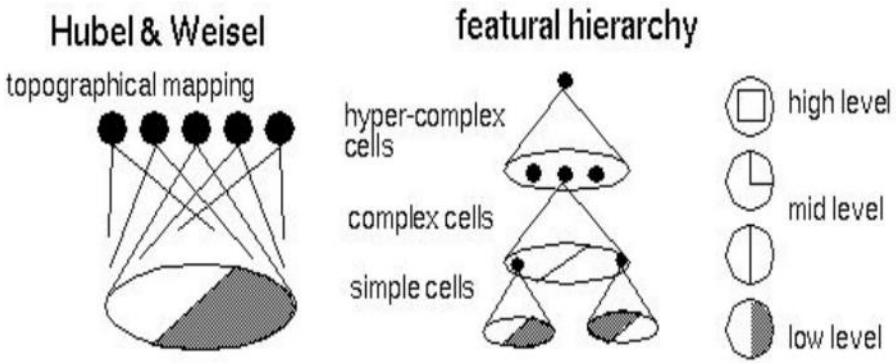
video

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.

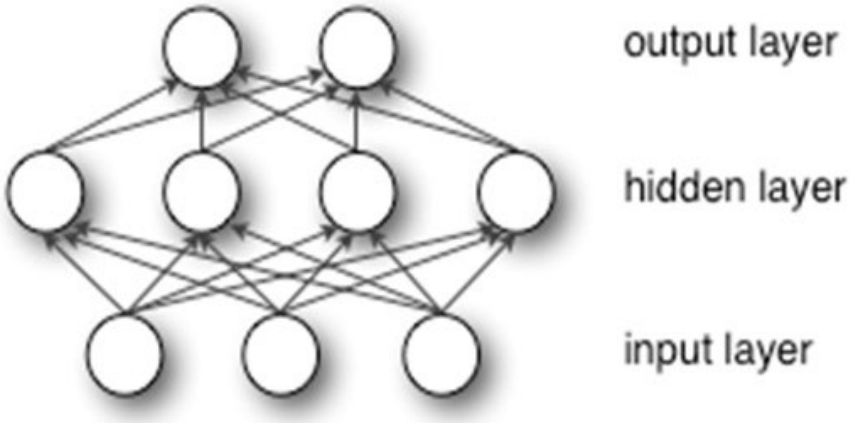


David Hubel's Eye, Brain, and Vision

Hubel/Wiesel Architecture and Multi-layer Neural Network



Hubel and Weisel's architecture



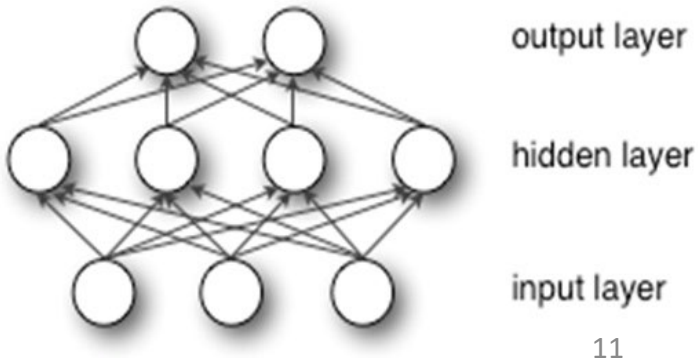
Multi-layer Neural Network
- A *non-linear* classifier

Multi-layer Neural Network

- A non-linear classifier
- **Training:** find network weights **w** to minimize the error between true training labels and estimated labels

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

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



Neocognitron [Fukushima, Biological Cybernetics 1980]

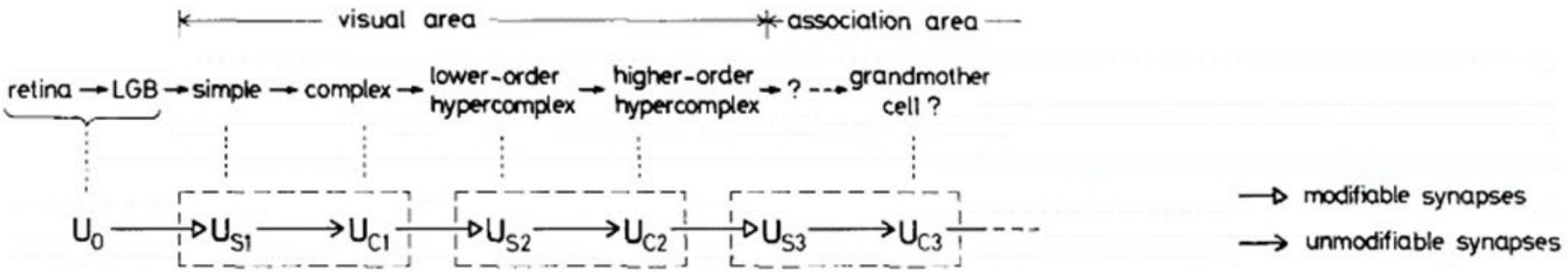
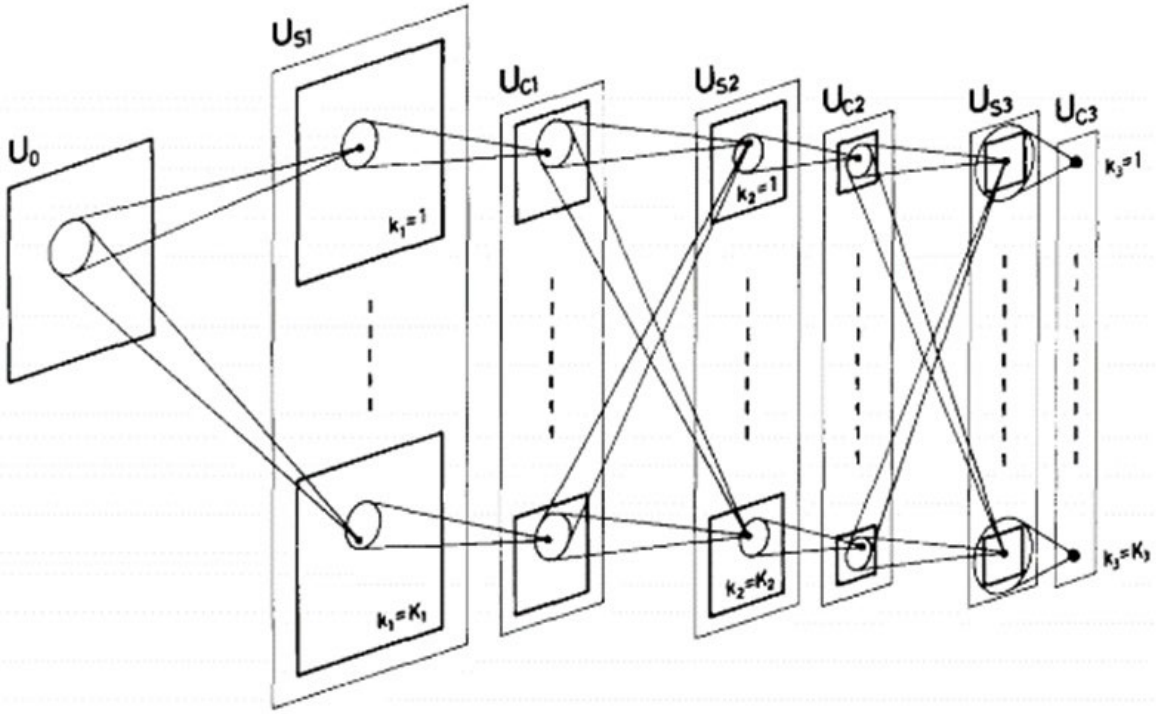


Fig. 1. Correspondence between the hierarchy model by Hubel and Wiesel, and the neural network of the neocognitron



Deformation-Resistant Recognition

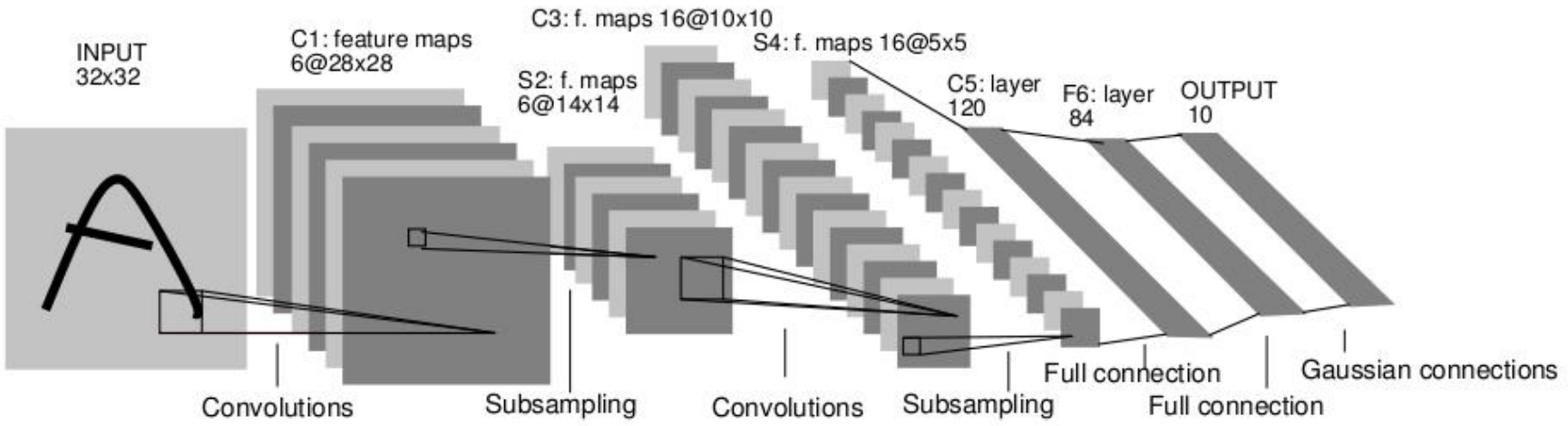
S-cells: (simple)
- extract local features

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

Convolutional Neural Networks (CNN, Convnet)

- Multi-layer Neural network with
  - **Local** connectivity
  - **Shared** weight parameters across spatial positions
- Stack multiple stages of feature extractors
- Higher stages compute more global, more invariant features
- Classification layer at the end

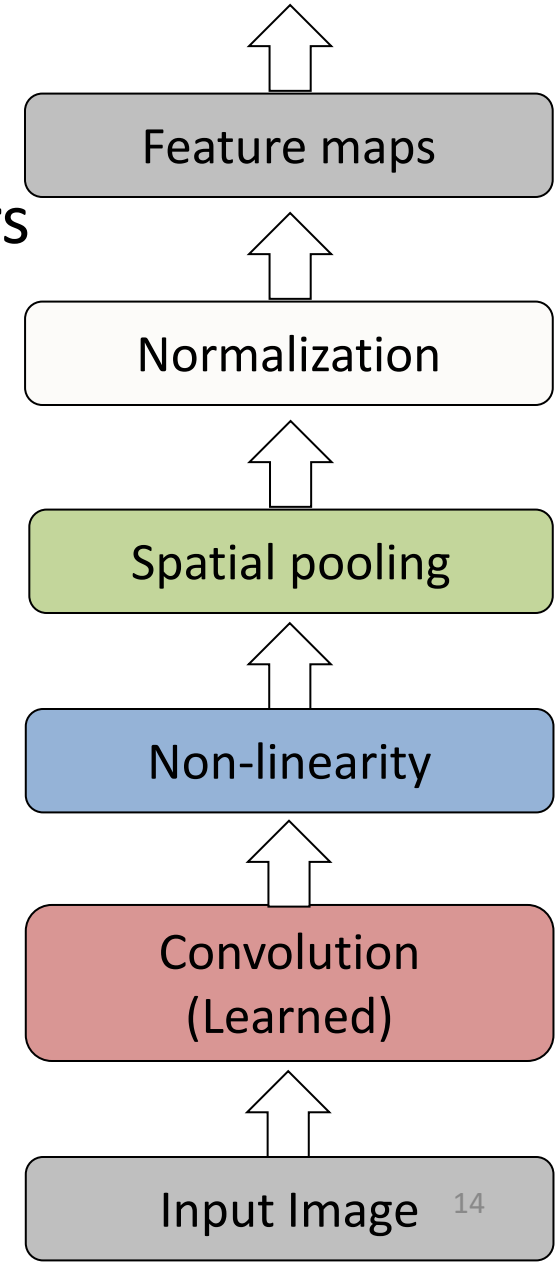
LeNet-1 from 1993



Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86(11): 2278–2324, 1998.

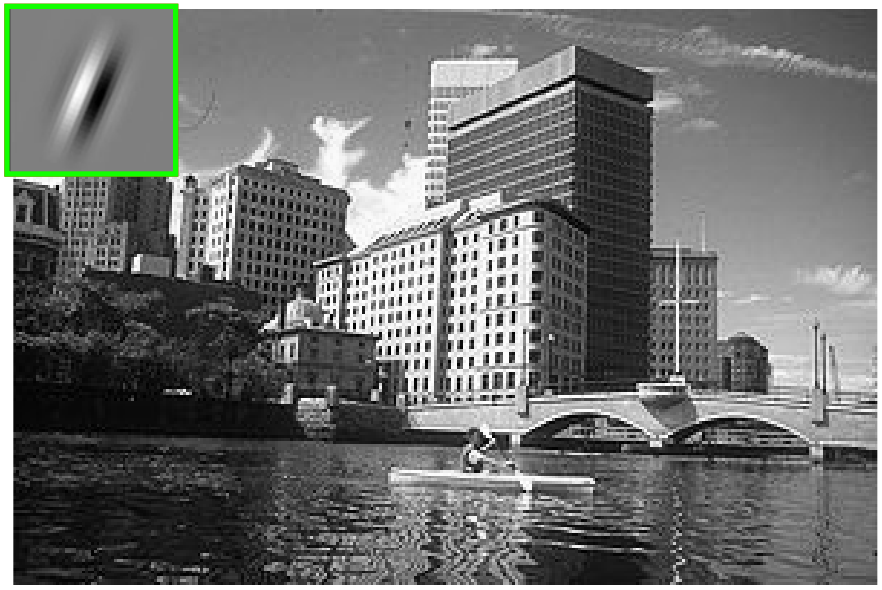
Convolutional Neural Networks (CNN, Convnet)

- Feed-forward feature extraction:
  1. Convolve input with learned filters
  2. Non-linearity
  3. Spatial pooling
  4. Normalization
- Supervised training of convolutional filters by back-propagating classification error

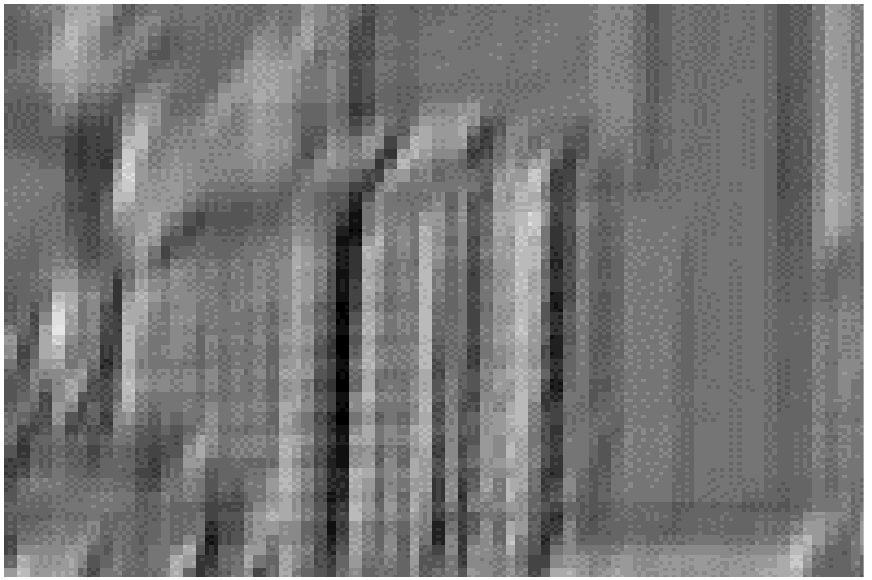


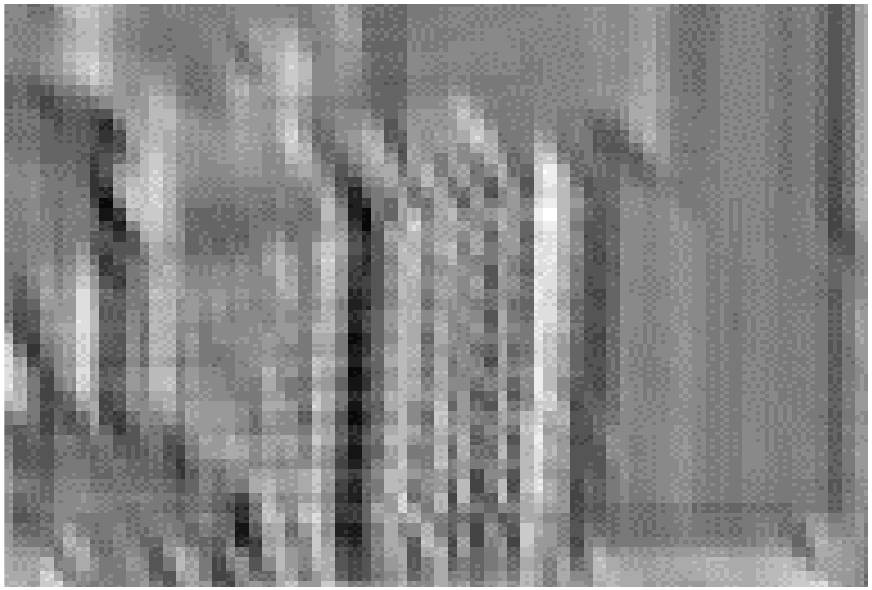
1. Convolution

- Dependencies are local
- Translation invariance
- Few parameters (filter weights)
- Stride can be greater than 1 (faster, less memory)



Input

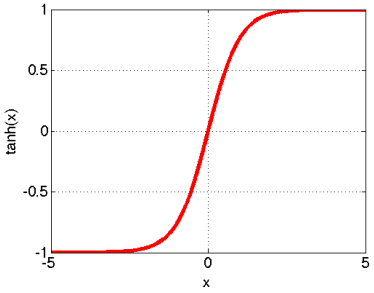


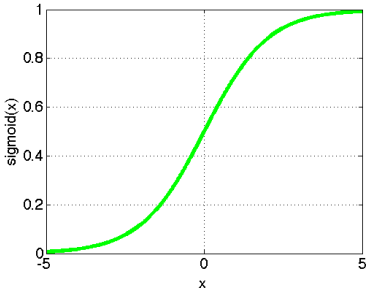


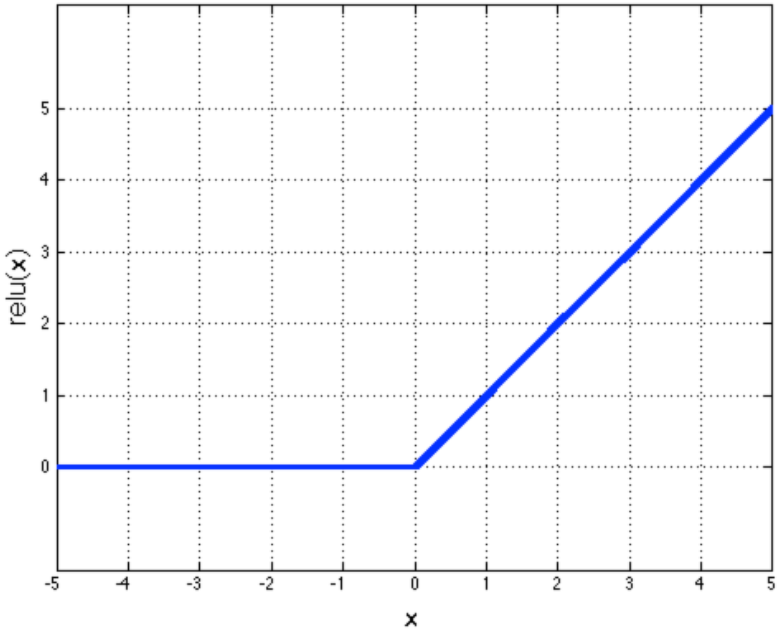
Feature Map

2. Non-Linearity

- Per-element (independent)
- Options:
  - Tanh
  - Sigmoid: 1/(1+exp(-x))
  - Rectified linear unit (ReLU)
    - Simplifies backpropagation
    - Makes learning faster
    - Avoids saturation issues
      → Preferred option

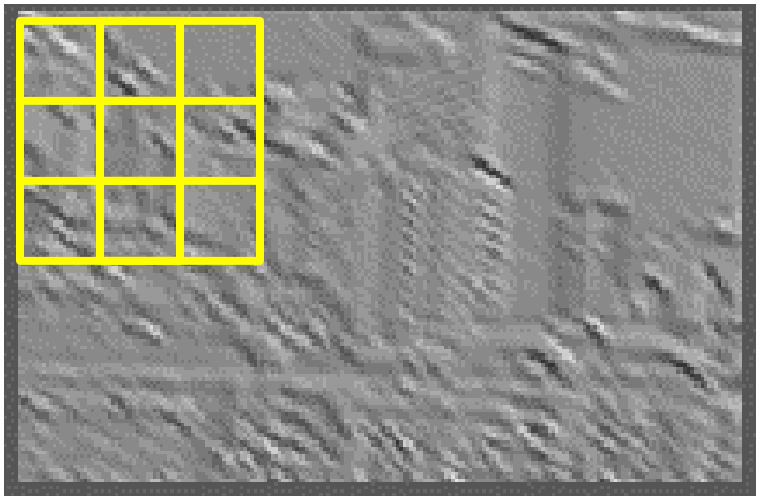




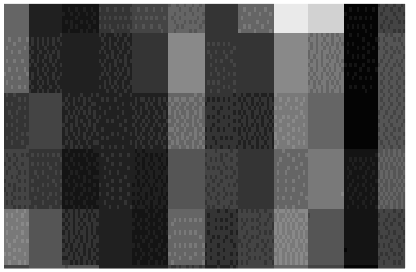


3. Spatial Pooling

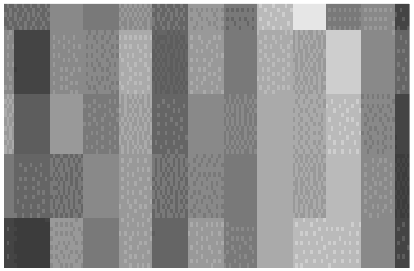
- Sum or max
- Non-overlapping / overlapping regions
- Role of pooling:
  - Invariance to small transformations
  - Larger receptive fields (see more of input)



Max

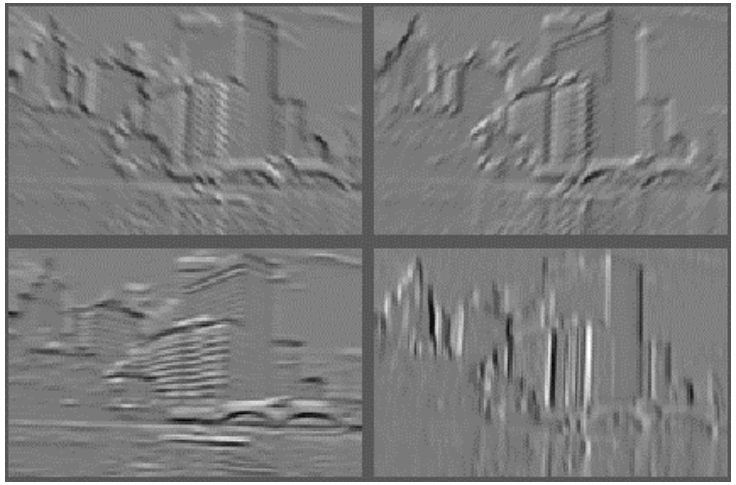


Sum

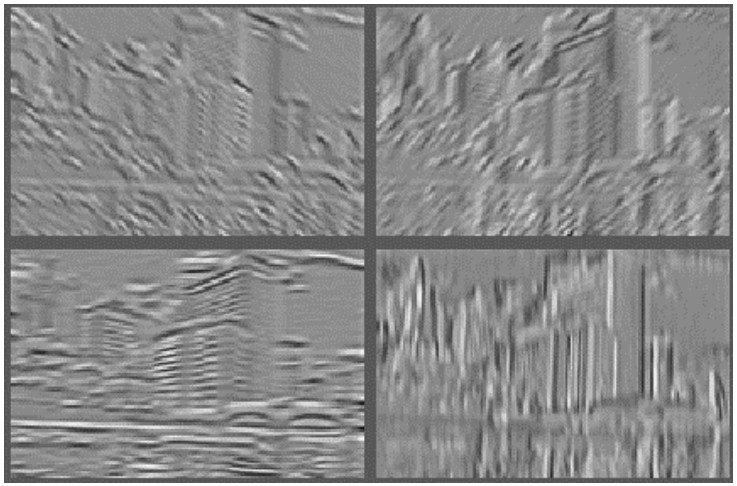


4. Normalization

- Within or across feature maps
- Before or after spatial pooling



Feature Maps

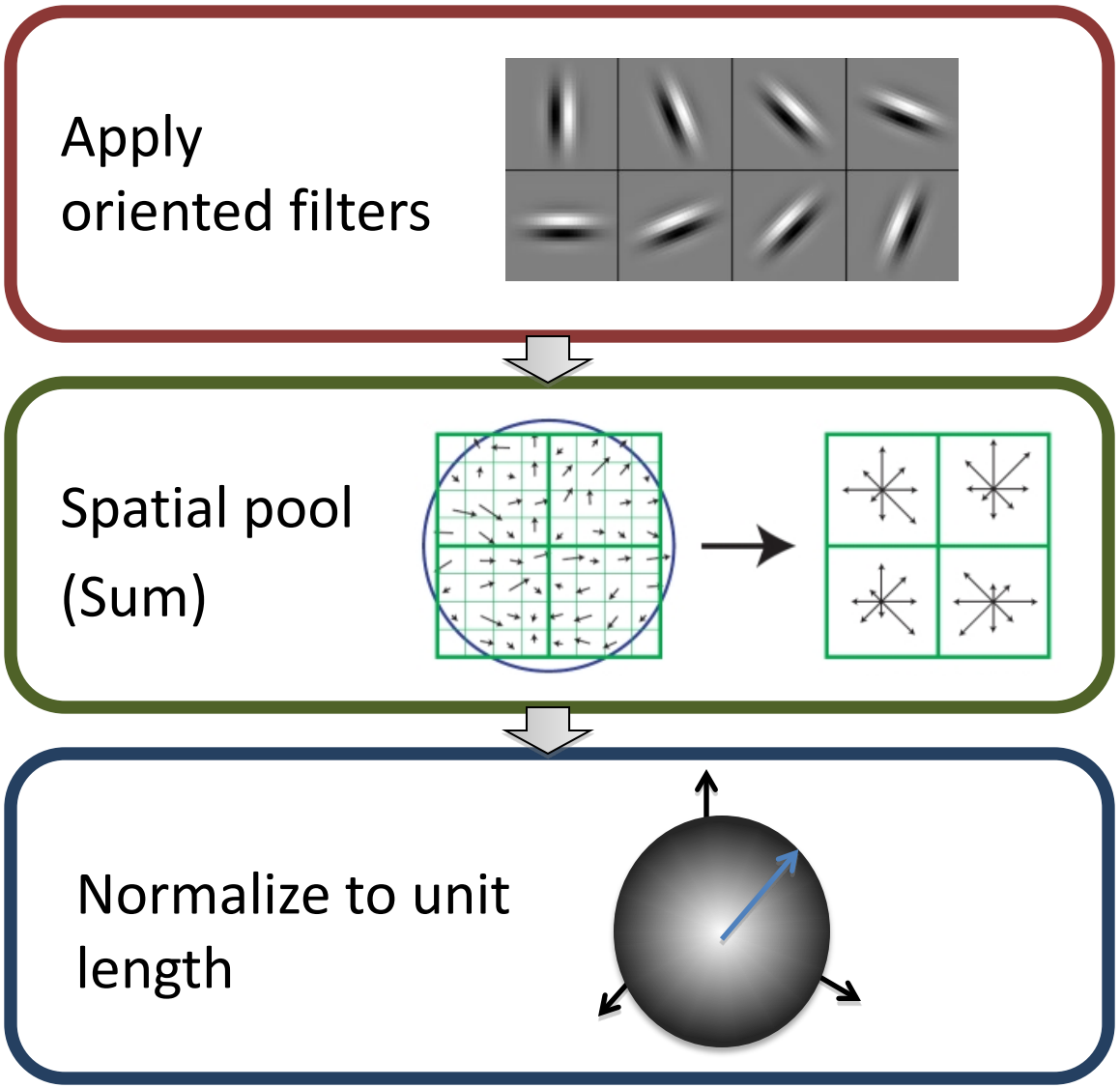


Feature Maps
After Contrast Normalization

Compare: SIFT Descriptor

Lowe
[IJCV 2004]

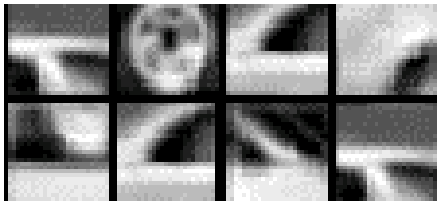
Image Pixels



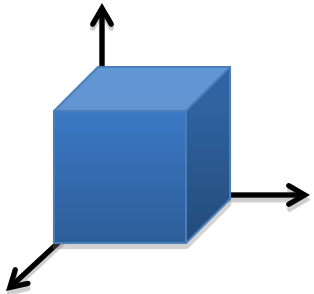
Compare: Spatial Pyramid Matching

SIFT features

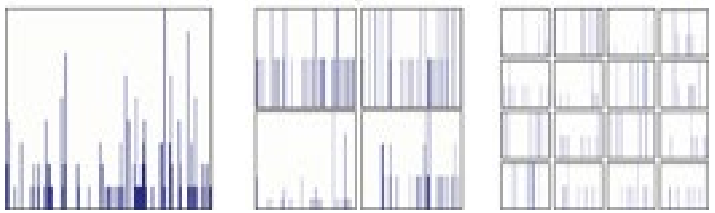
Filter with Visual Words



Take max VW response



Multi-scale spatial pool (Sum)



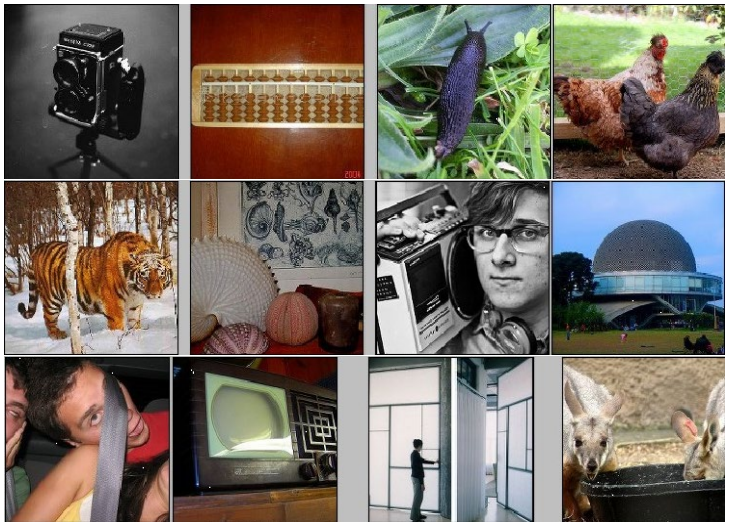
Global image descriptor

Lazebnik, Schmid, Ponce [CVPR 2006]

Convnet Successes

- Handwritten text/digits
  - MNIST (0.17% error [Ciresan et al. 2011])
  - Arabic & Chinese [Ciresan et al. 2012]
- Simpler recognition benchmarks
  - CIFAR-10 (9.3% error [Wan et al. 2013])
  - Traffic sign recognition
    - 0.56% error vs 1.16% for humans [Ciresan et al. 2011]
- But until recently, less good at more complex datasets
  - Caltech-101/256 (few training examples)

ImageNet Challenge 2012



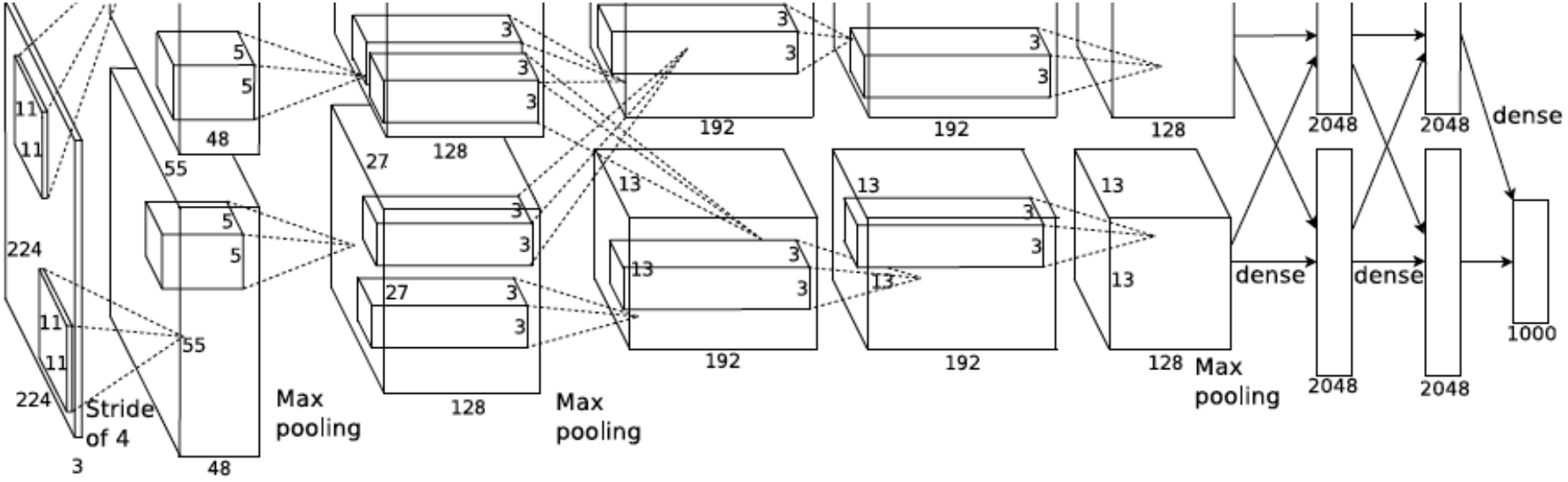
[Deng et al. CVPR 2009]

- ~14 million labeled images, 20k classes
- Images gathered from Internet
- Human labels via Amazon Turk
- **ImageNet Challenge**: *1.2 million training images, 1000 classes*

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

ImageNet Challenge 2012

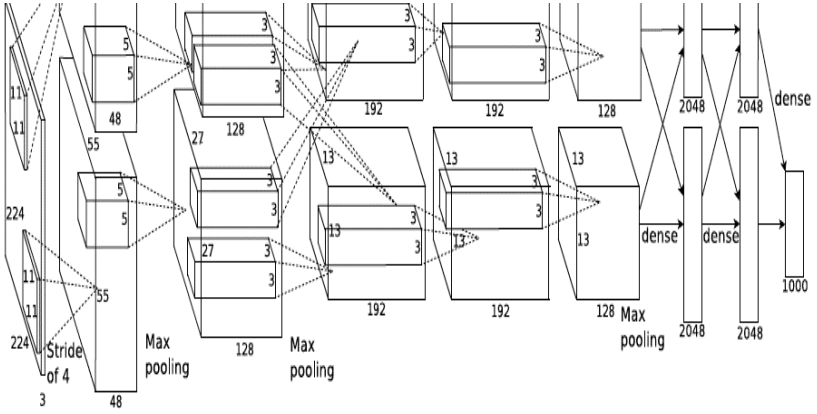
- "AlexNet": Similar framework to LeCun'98 but:
  - Bigger model (7 hidden layers, 650,000 units, 60,000,000 params)
  - More data (10^6 vs. 10^3 images)
  - GPU implementation (50x speedup over CPU)
    - Trained on two GPUs for a week
  - Better regularization for training (DropOut)



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

Using CNN for Image Classification

AlexNet

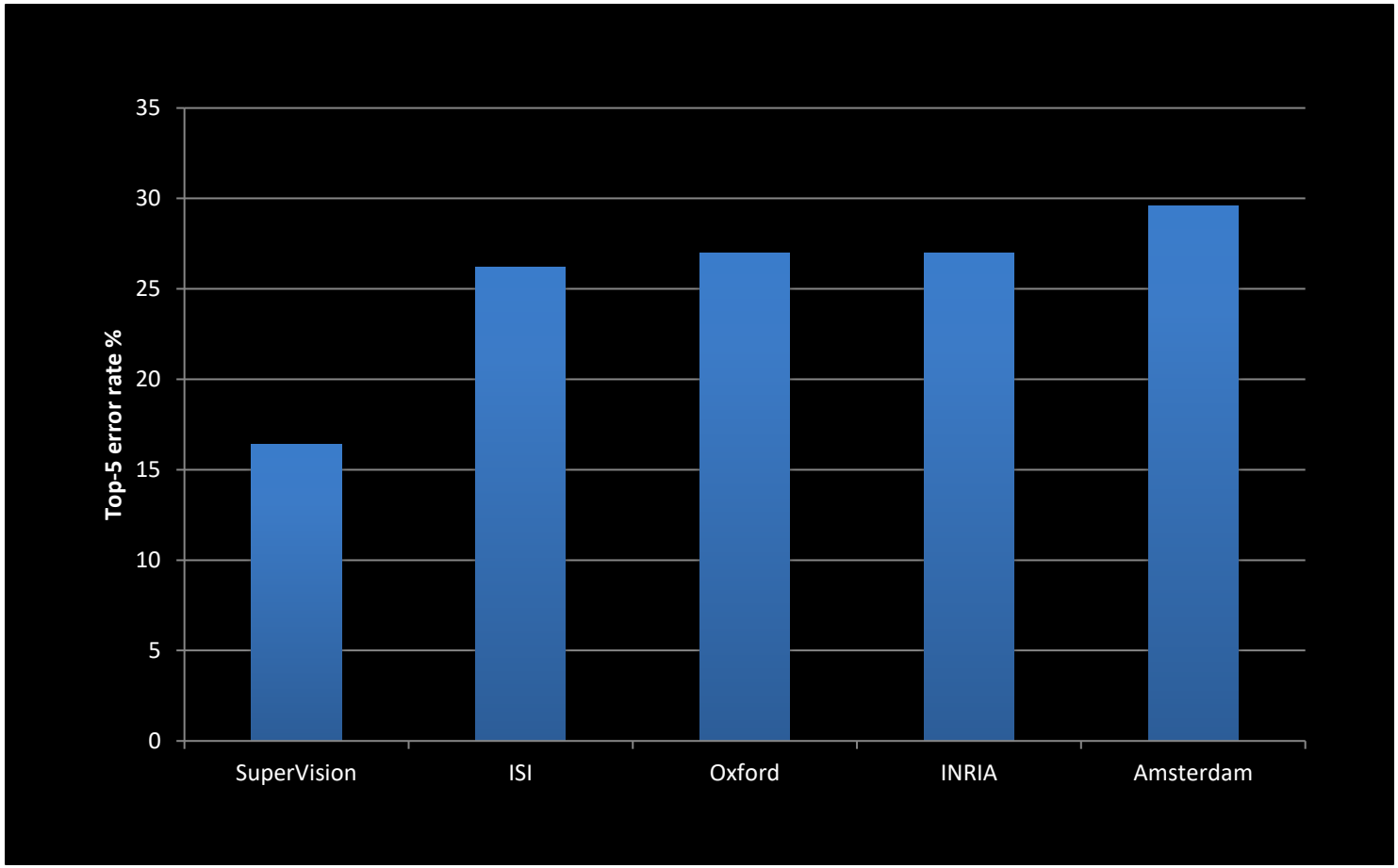


"car"

Fixed input size: 224x224x3

ImageNet Challenge 2012

- Krizhevsky et al. -- **16.4% error** (top-5)
- Next best (non-convnet) – **26.2% error**



ImageNet Challenge 2012-2014

Best non-convnet in 2012: 26.2%

Team	Year	Place	Error (top-5)	External data
SuperVision – Toronto (7 layers)	2012	-	16.4%	no
SuperVision	2012	1st	15.3%	ImageNet 22k
Clarifai – NYU (7 layers)	2013	-	11.7%	no
Clarifai	2013	1st	11.2%	ImageNet 22k
VGG – Oxford (16 layers)	2014	2nd	7.32%	no
GoogLeNet (19 layers)	2014	1st	6.67%	no
Human expert*			5.1%	

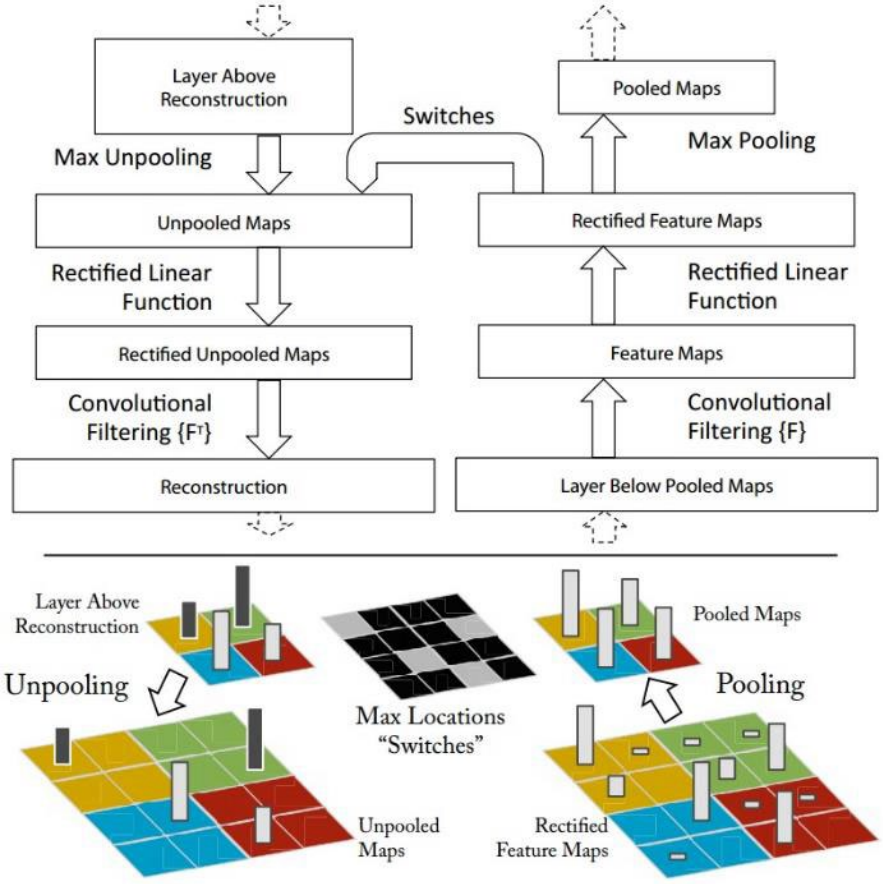
Team	Method	Error (top-5)
DeepImage - Baidu	Data augmentation + multi GPU	5.33%
PReLU-nets - MSRA	Parametric ReLU + smart initialization	4.94%
BN-Inception ensemble - Google	Reducing internal covariate shift	4.82%

Understanding and Visualizing CNN

- Visualize input pattern using deconvnet
- Individual neuron activation
- Fooling CNNs

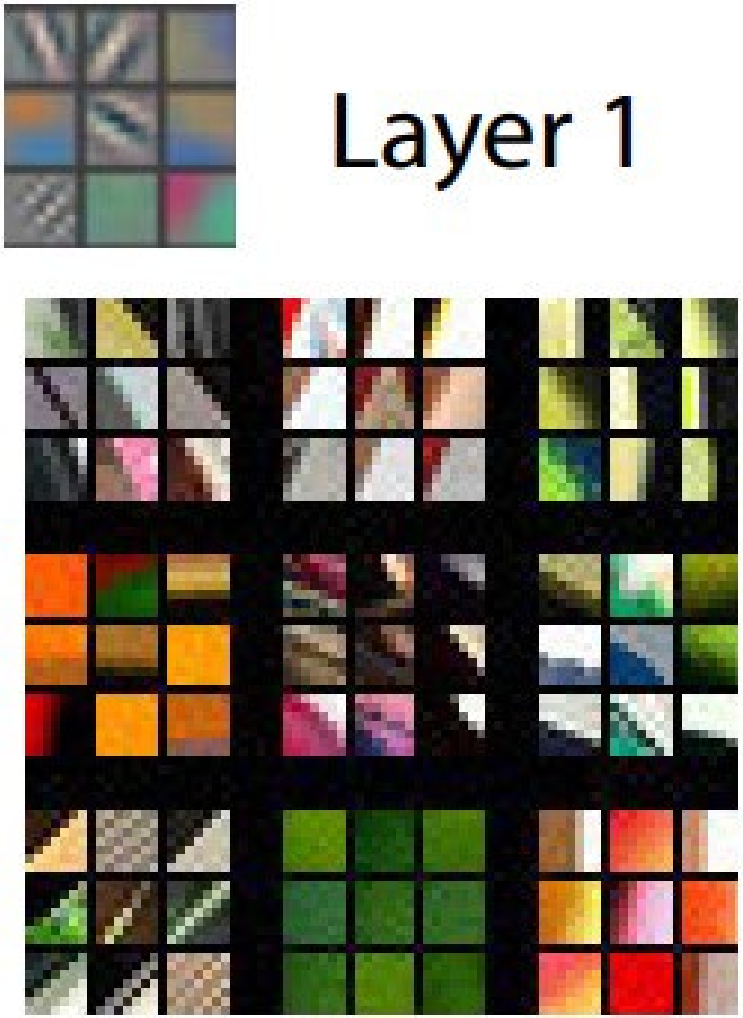
Map activation back to the input pixel space

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



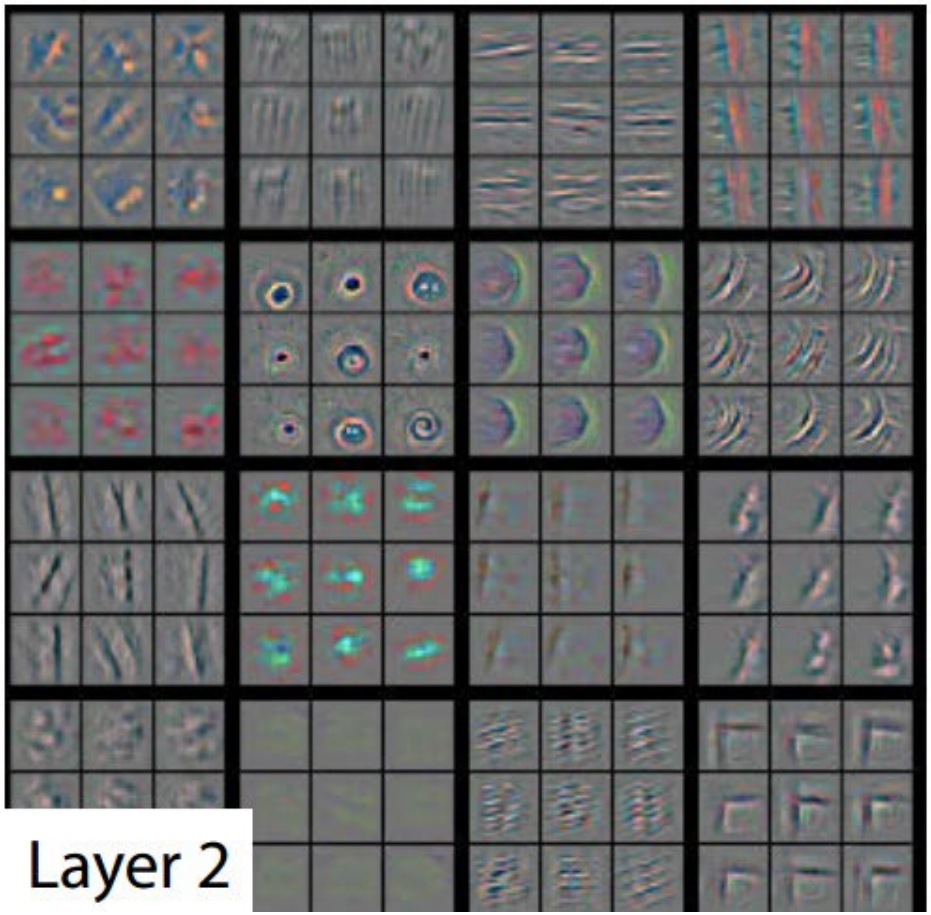
Visualizing and Understanding Convolutional Networks [Zeiler and Fergus, ECCV 2014]

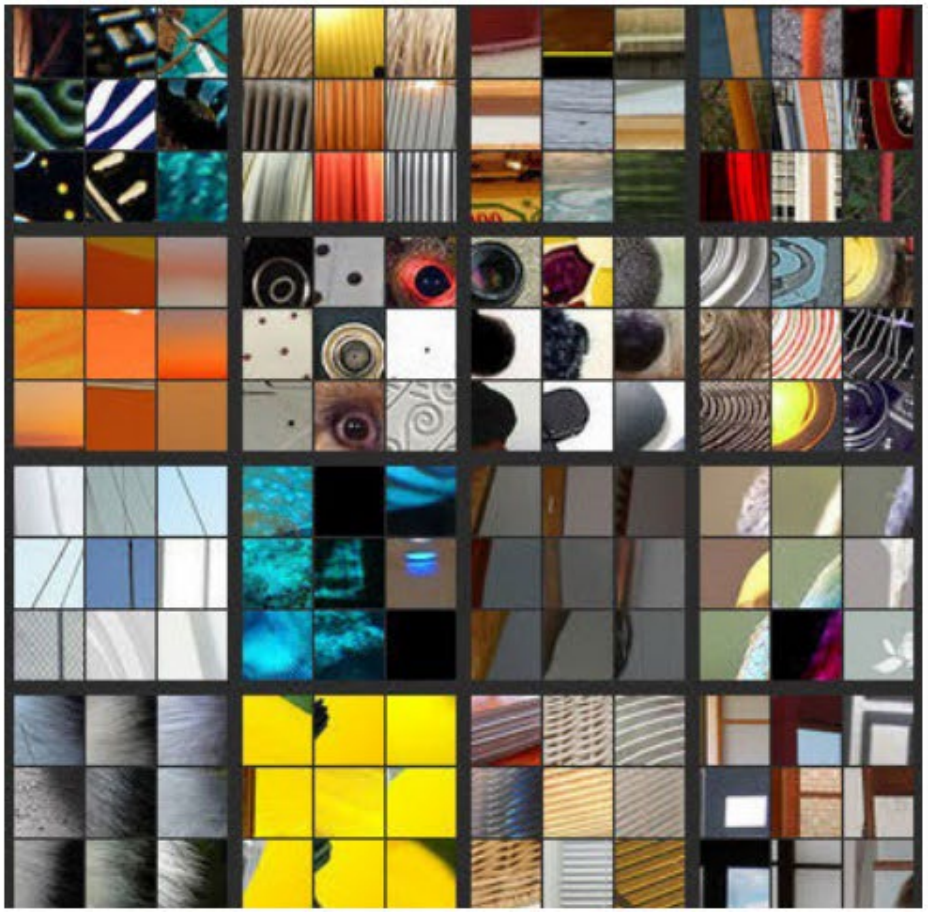
Layer 1



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

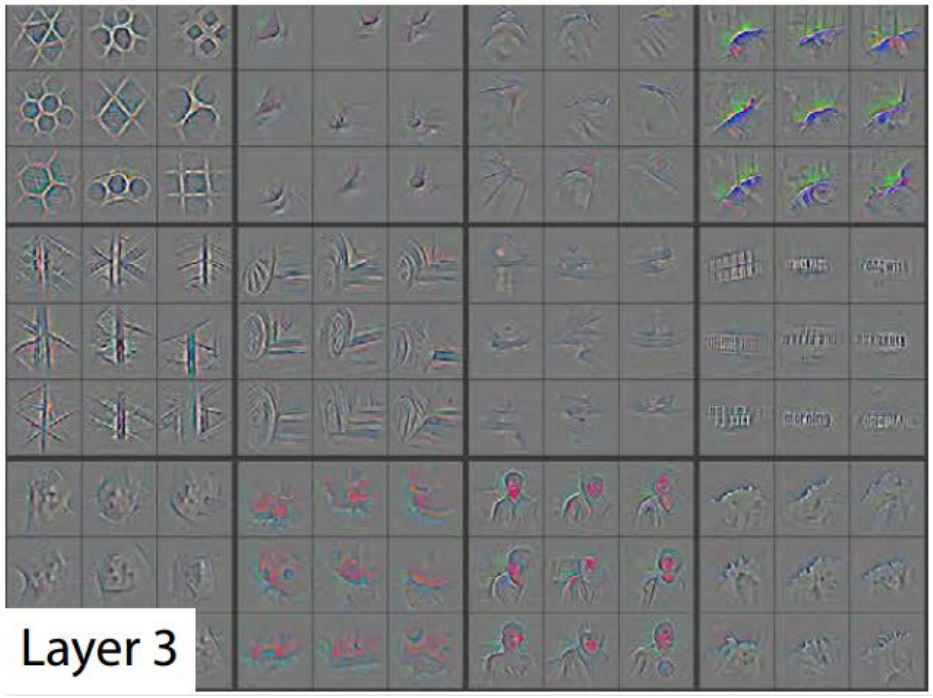
Layer 2





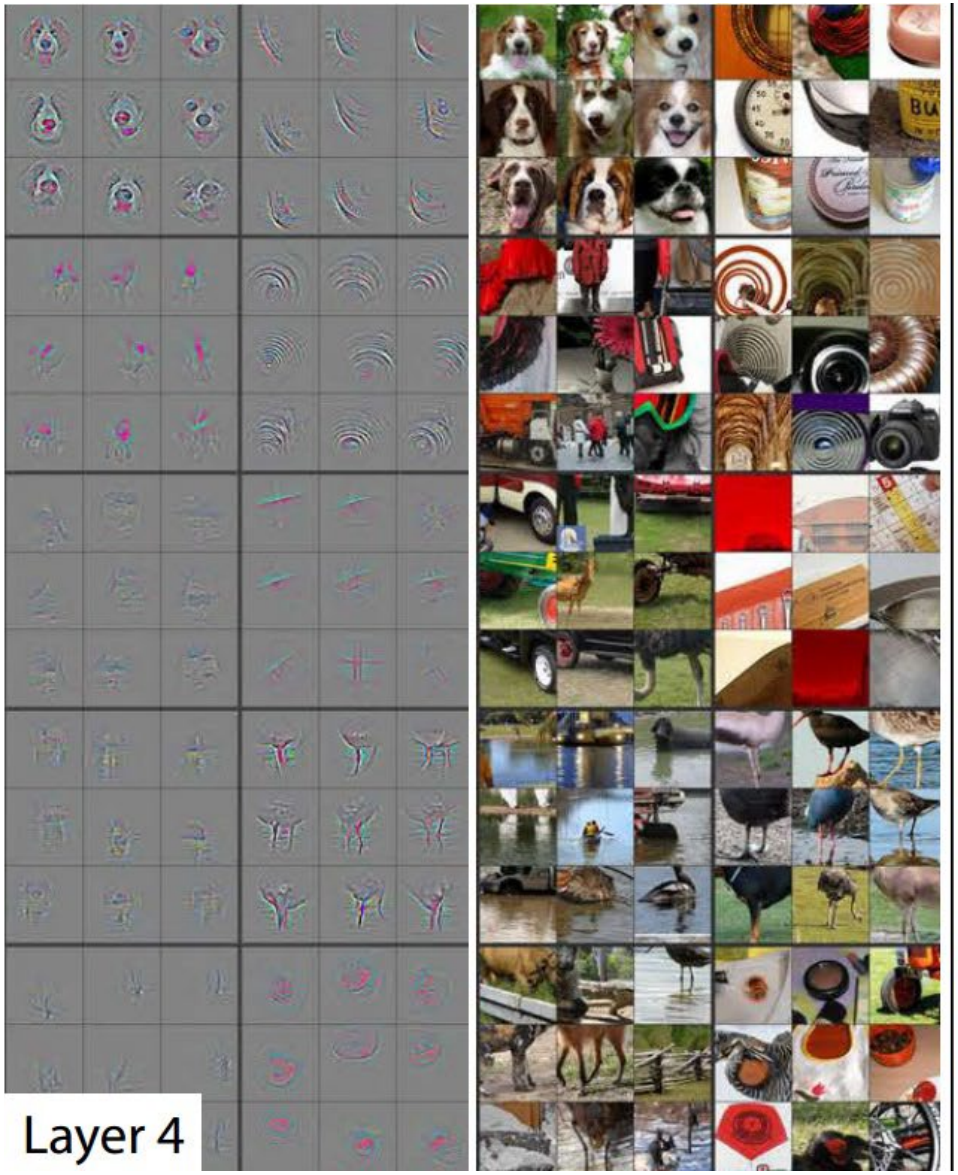
Visualizing and Understanding Convolutional Networks [Zeiler and Fergus, ECCV 2014]

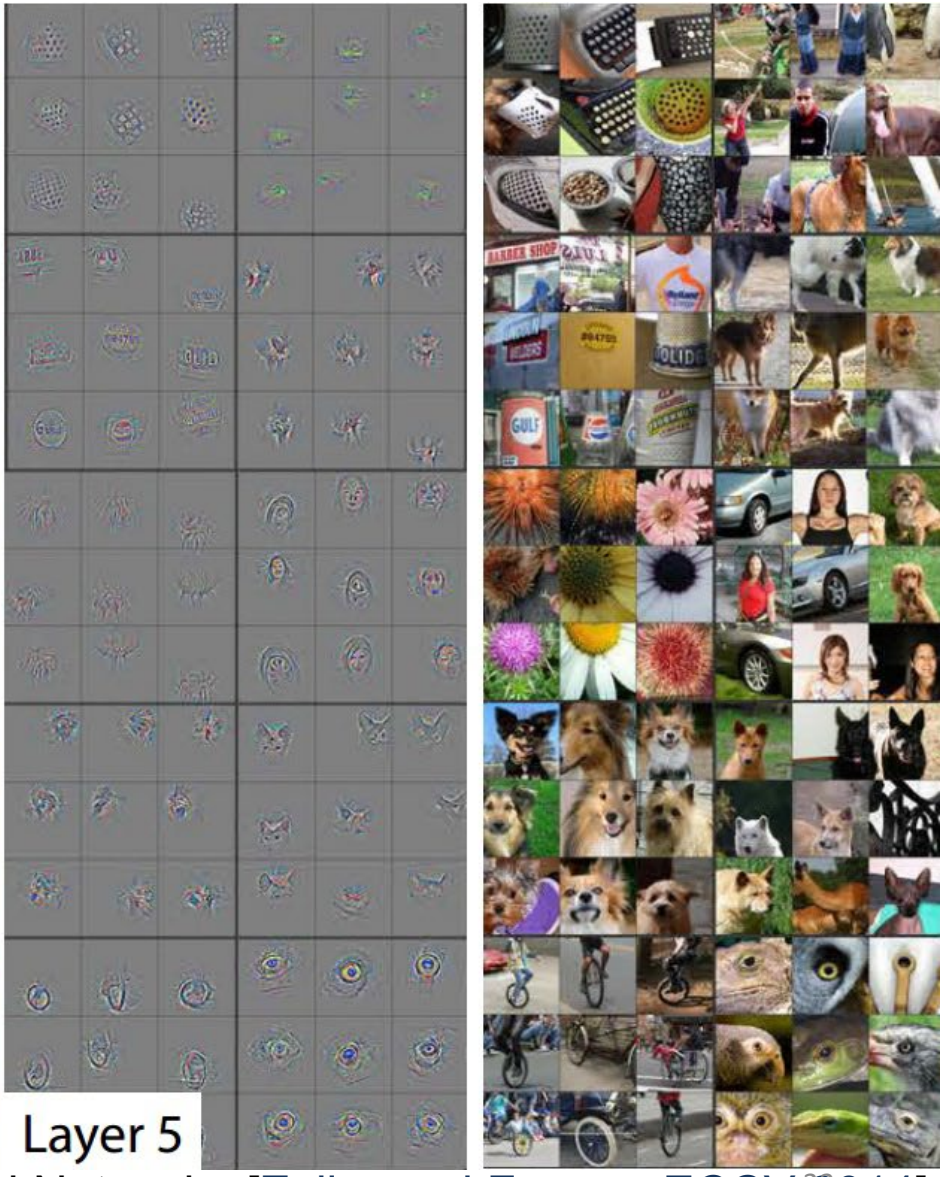
Layer 3



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

Layer 4 and 5





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

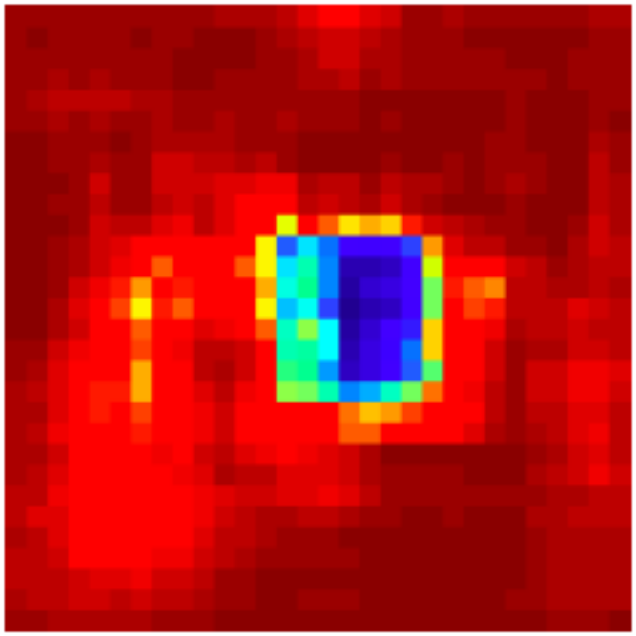
Occlusion Experiment

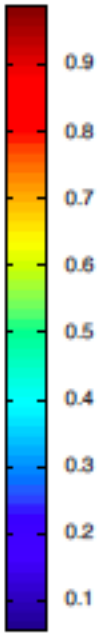
- Mask parts of input with occluding square

- Monitor output

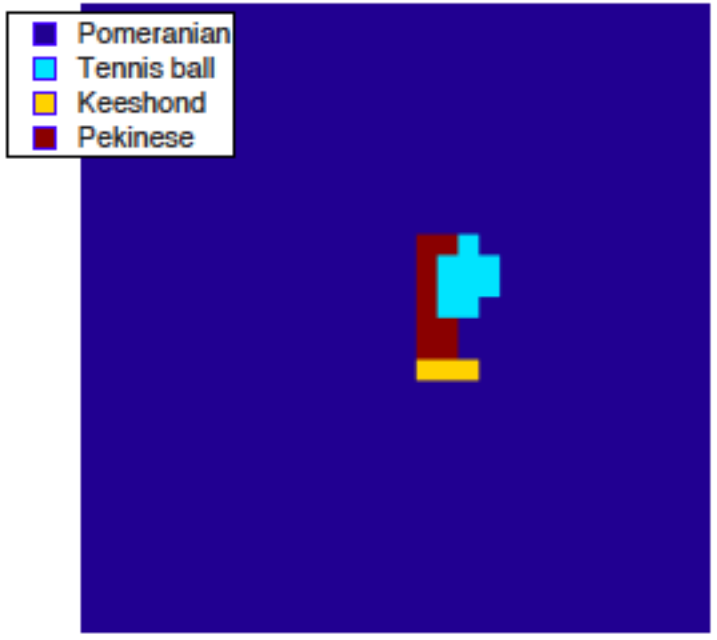
Input image

p(True class)



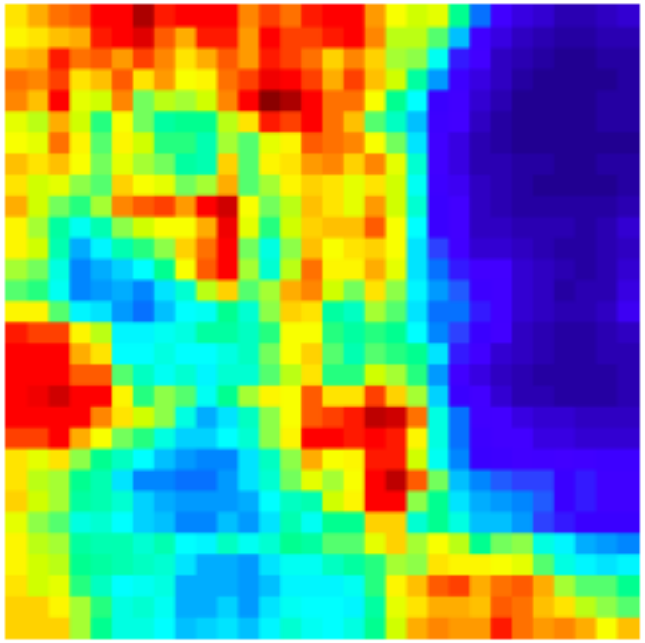


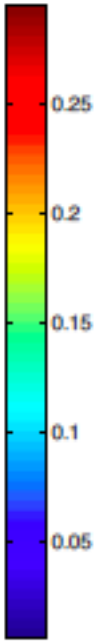
Most probable class



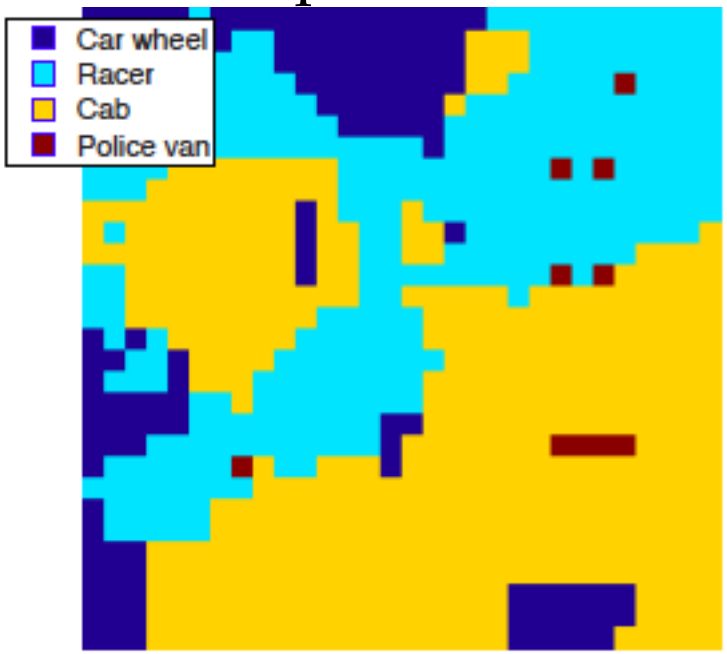
Input image

p(True class)

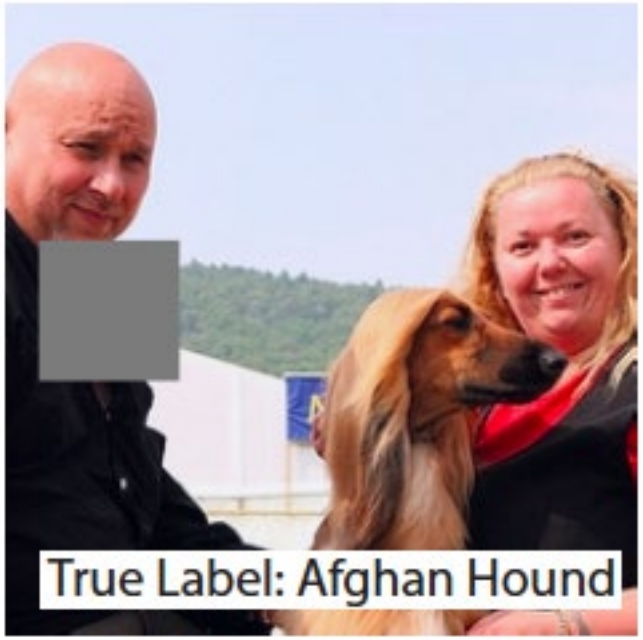




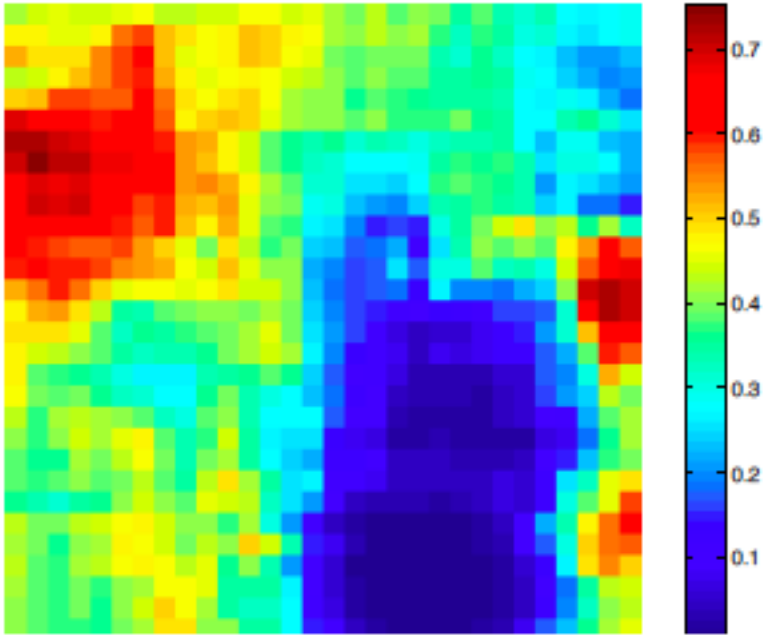
Most probable class



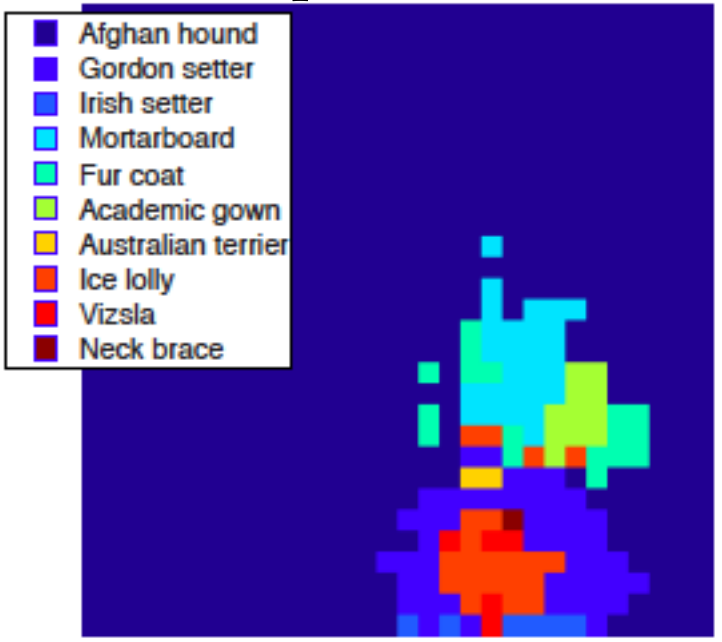
Input image



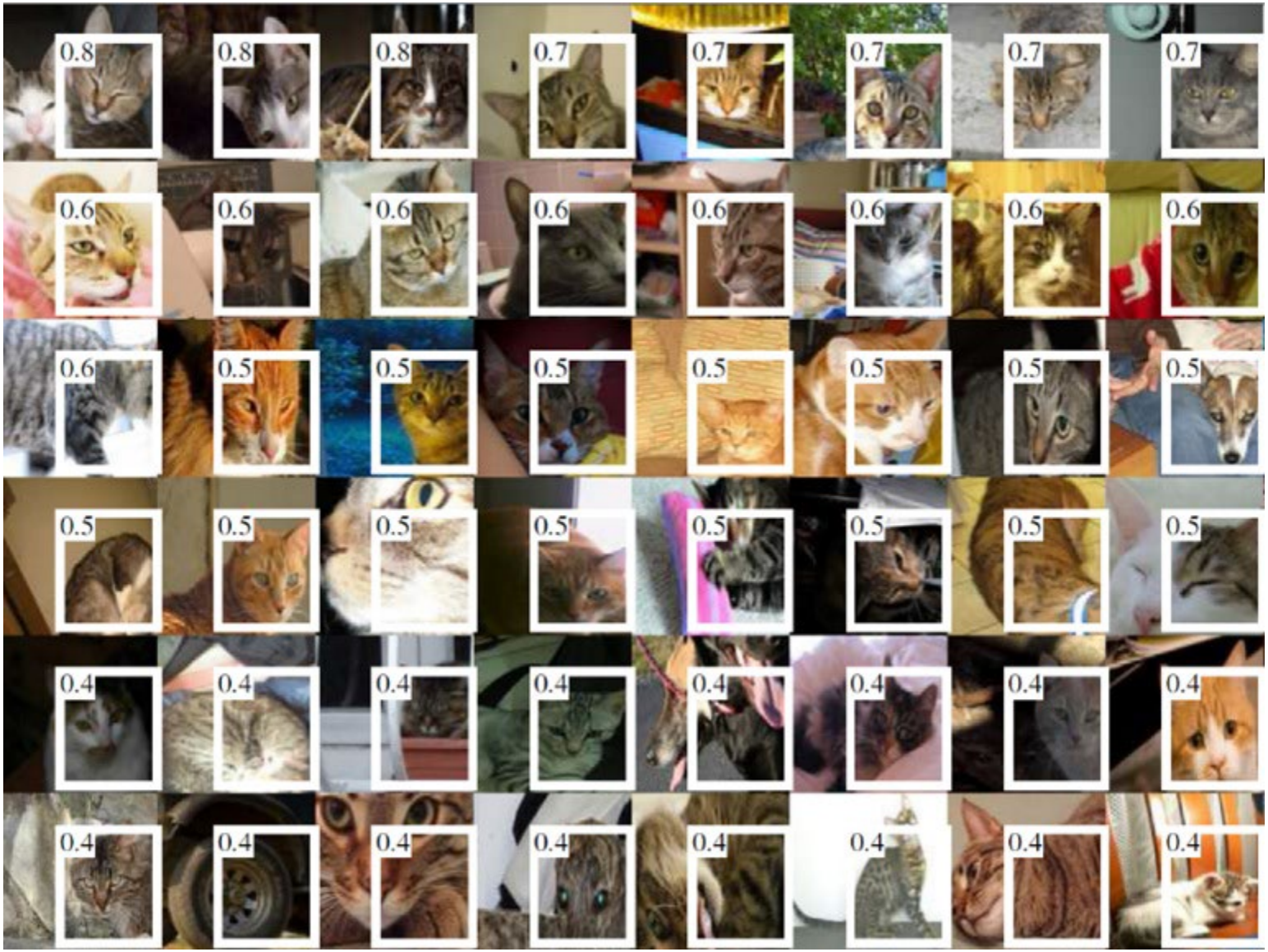
p(True class)



Most probable class

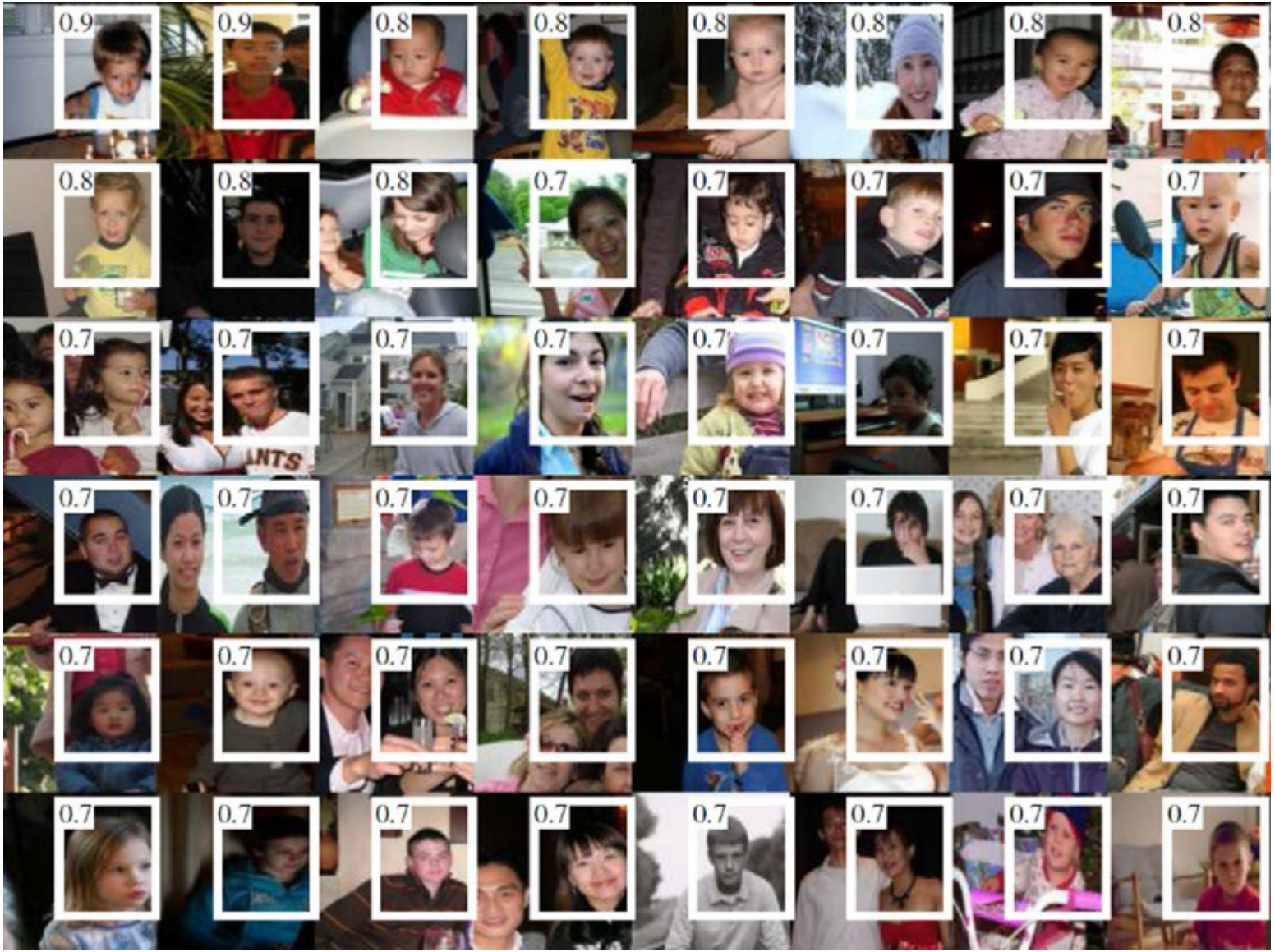


Individual Neuron Activation



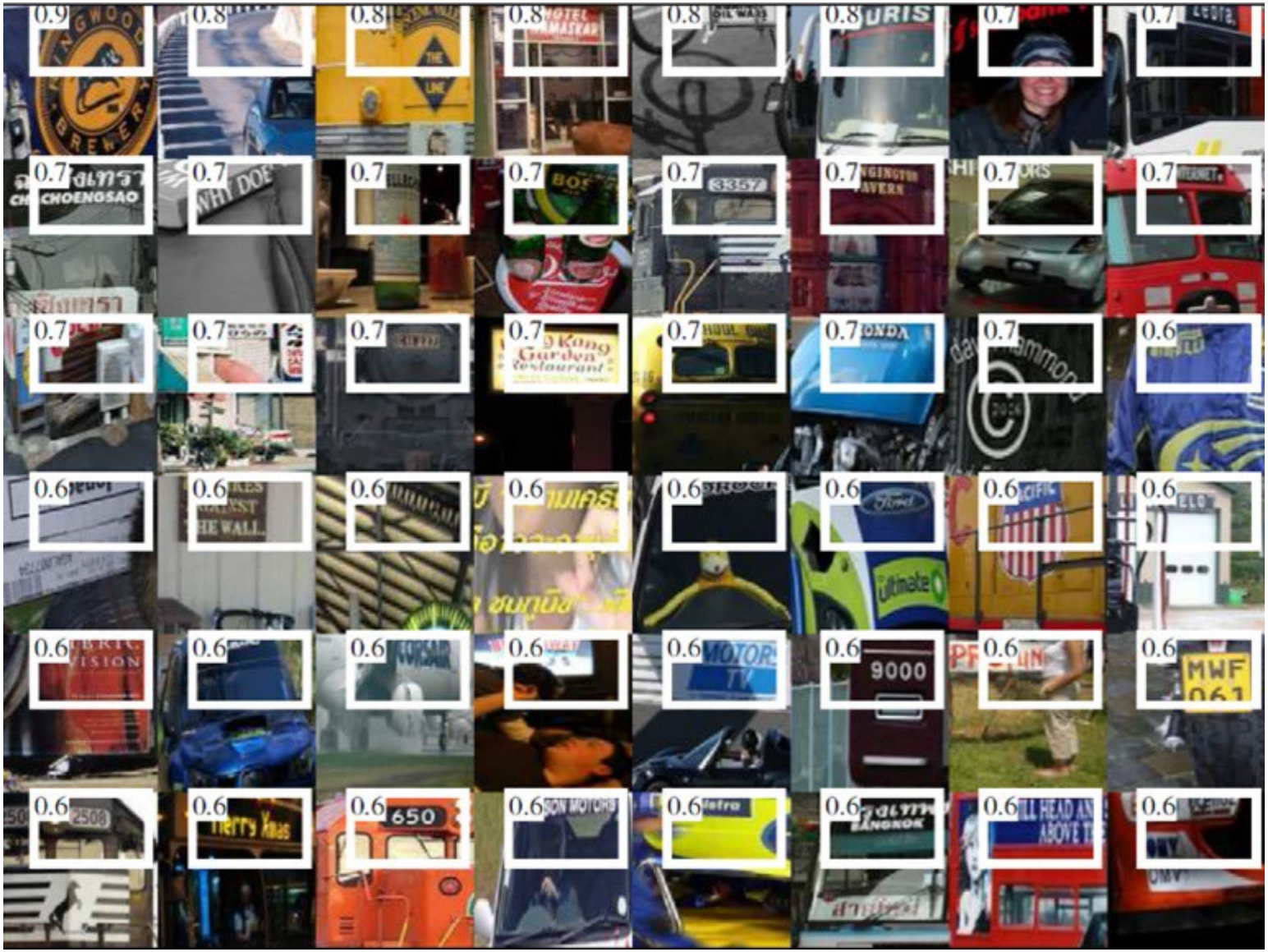
RCNN [Girshick et al. CVPR 2014]

Individual Neuron Activation



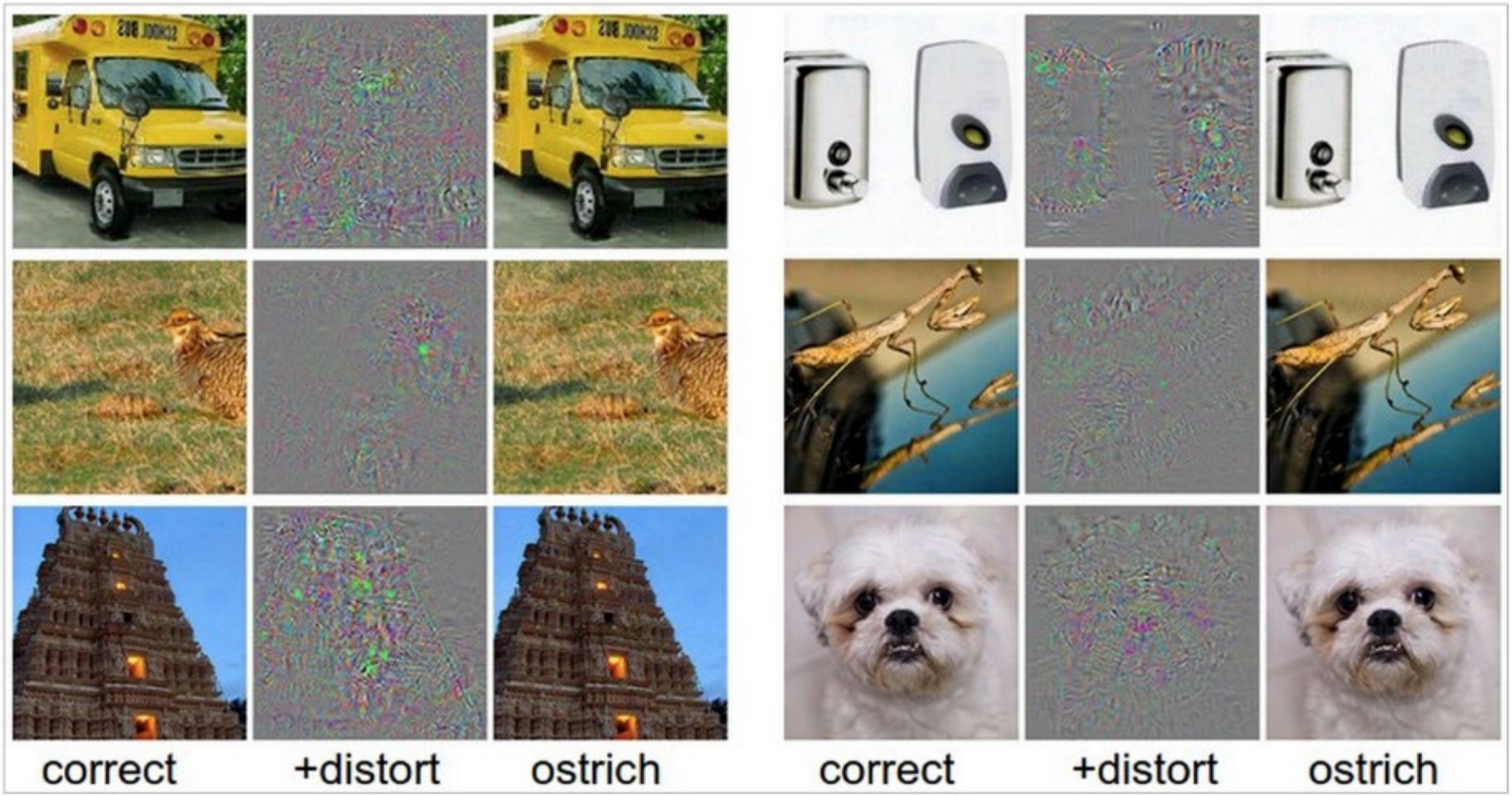
RCNN [Girshick et al. CVPR 2014]

Individual Neuron Activation



RCNN [Girshick et al. CVPR 2014]

Fooling CNNs



Take a correctly classified image (left image in both columns), and add a tiny distortion (middle) to fool the ConvNet with the resulting image (right).

Intriguing properties of neural networks [Szegedy ICLR 2014]

What is going on?

- Recall gradient descent training: modify the weights to reduce classifier error

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \frac{\partial E}{\partial \mathbf{w}}$$

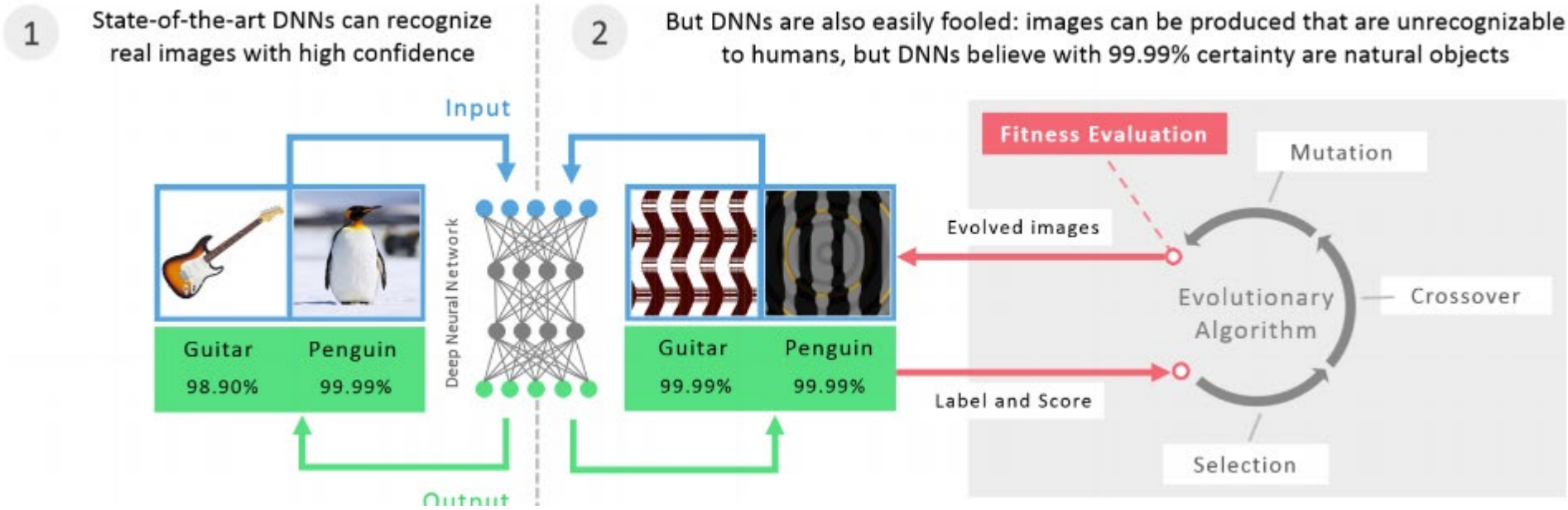
- Adversarial examples: modify the *image* to *increase* classifier error

$$\mathbf{x} \leftarrow \mathbf{x} + \alpha \frac{\partial E}{\partial \mathbf{x}}$$

Explaining and Harnessing Adversarial Examples [Goodfellow ICLR 2015]

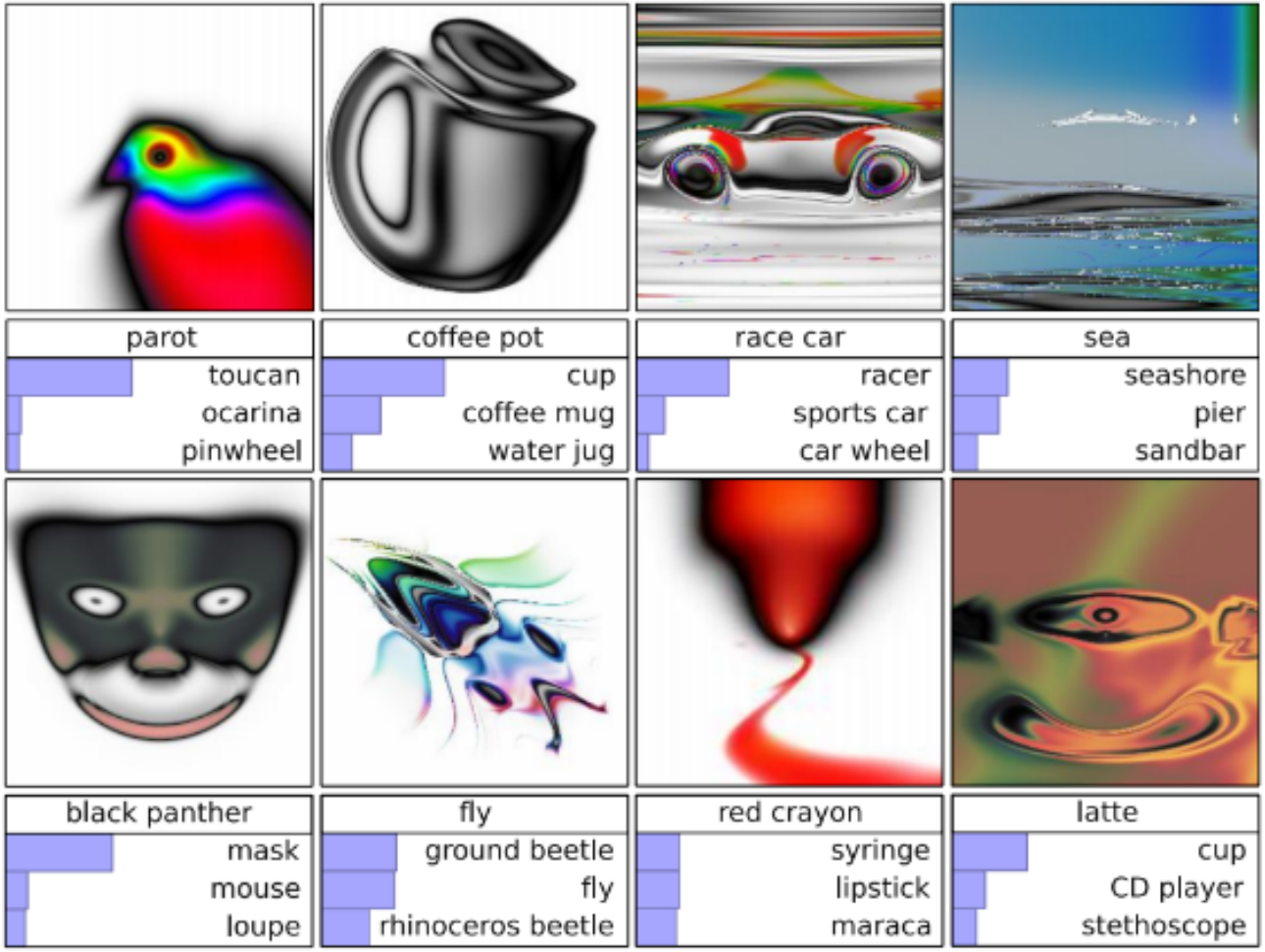
http://karpathy.github.io/2015/03/30/breaking-convnets/

Fooling CNNs



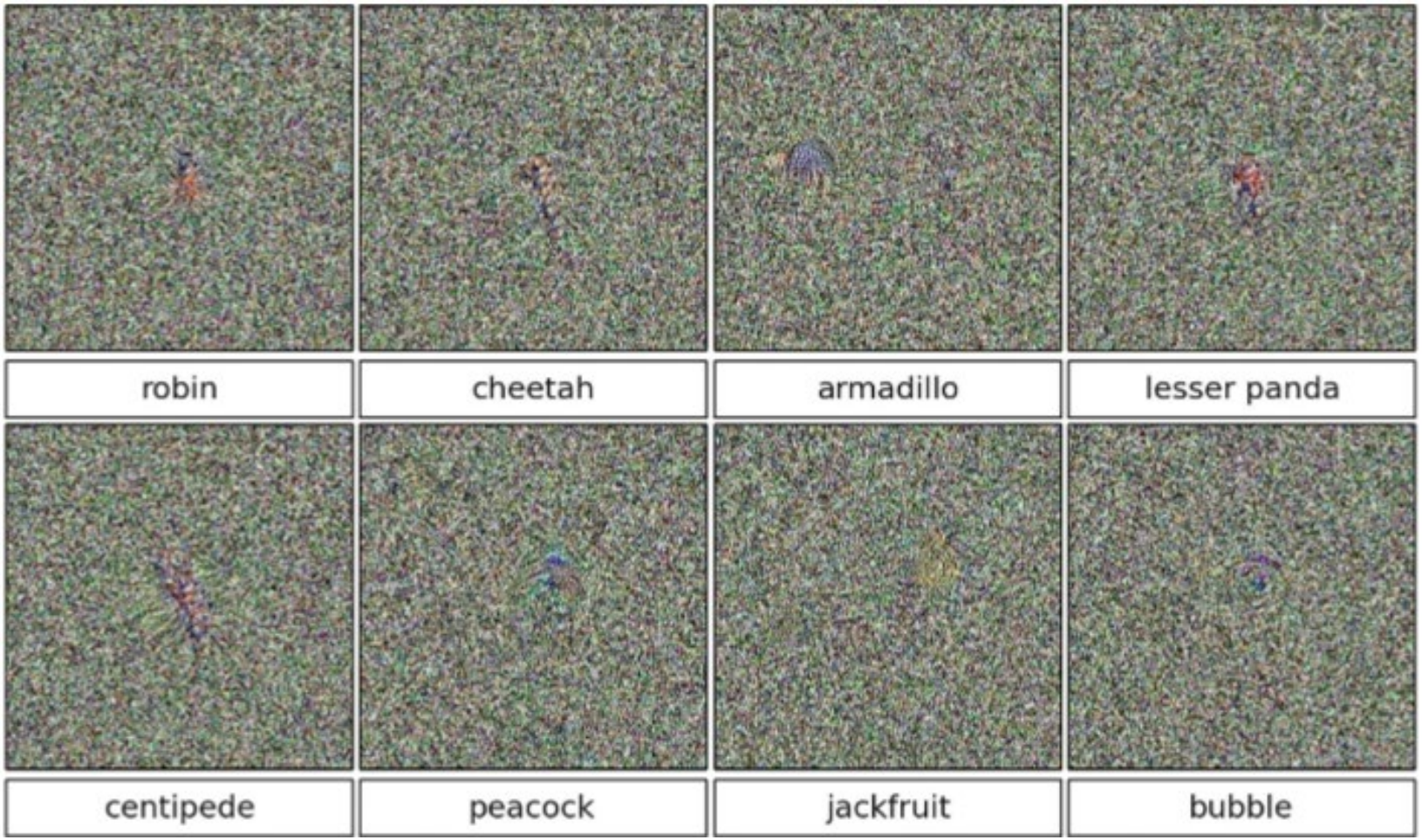
Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images [Nguyen et al. CVPR 2015]

Images that both CNN and Human can recognize



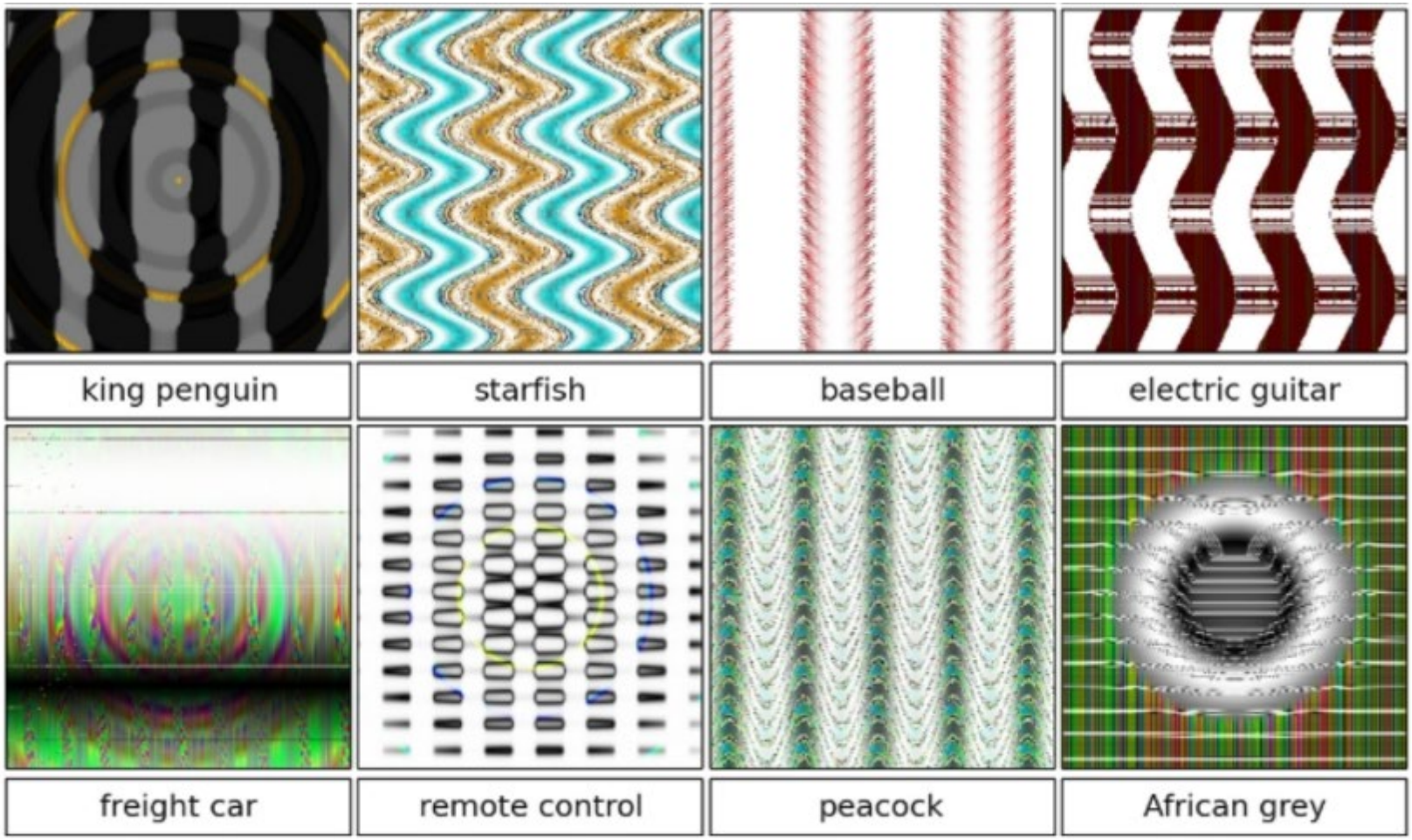
Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images [Nguyen et al. CVPR 2015]

Direct Encoding



Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images [Nguyen et al. CVPR 2015]

Indirect Encoding



Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images [Nguyen et al. CVPR 2015]

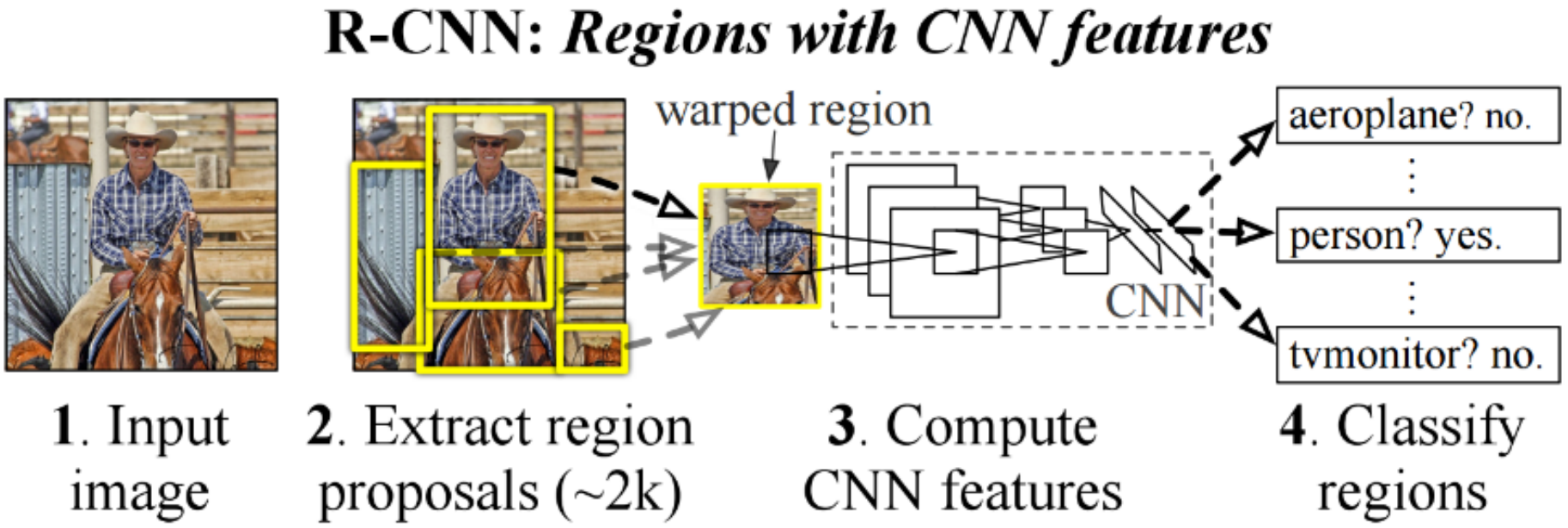
Beyond classification

- Detection
- Segmentation
- Regression
- Pose estimation
- Matching patches
- Synthesis

and many more…

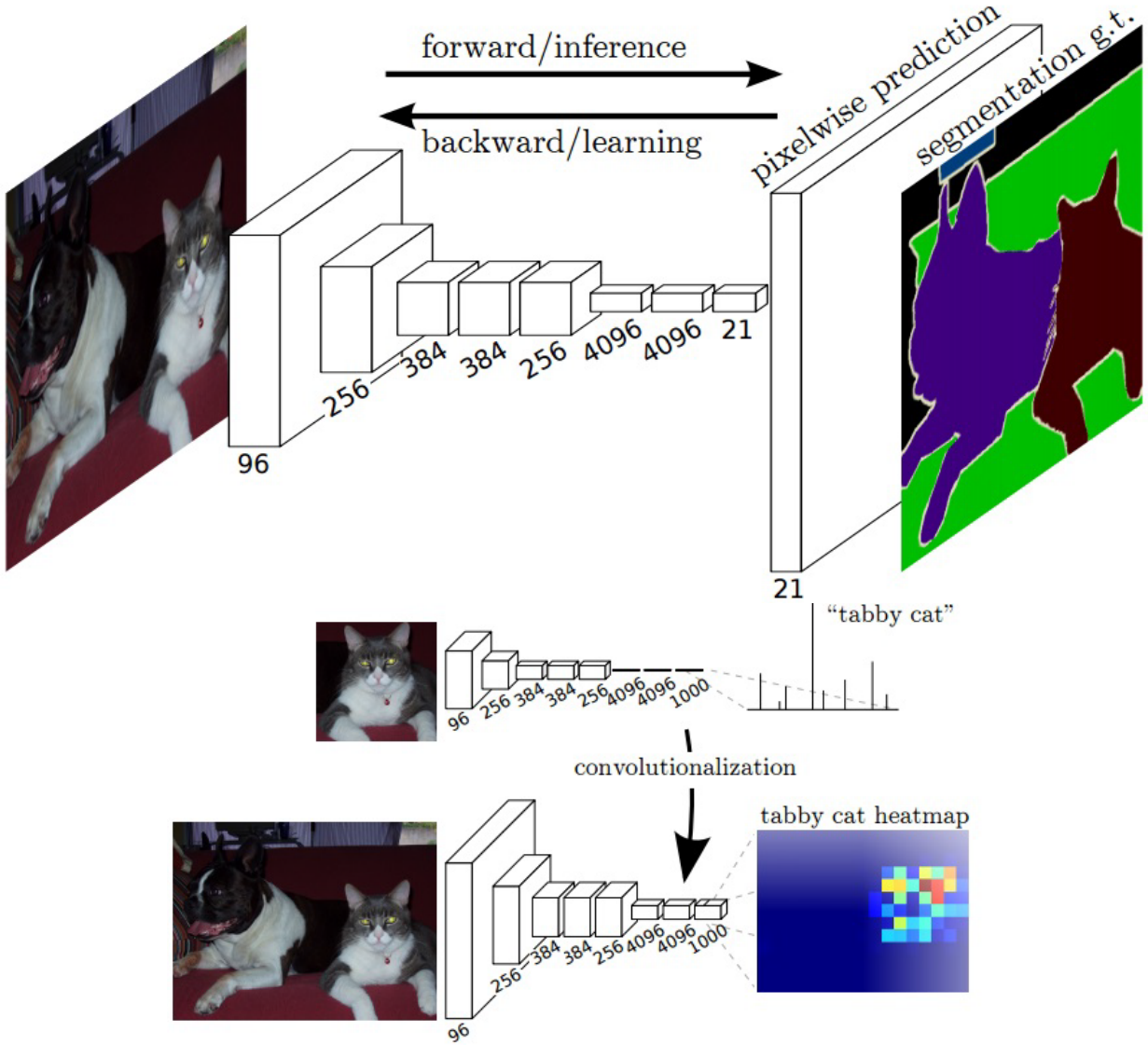
R-CNN: Regions with CNN features

- Trained on ImageNet classification
- Finetune CNN on PASCAL



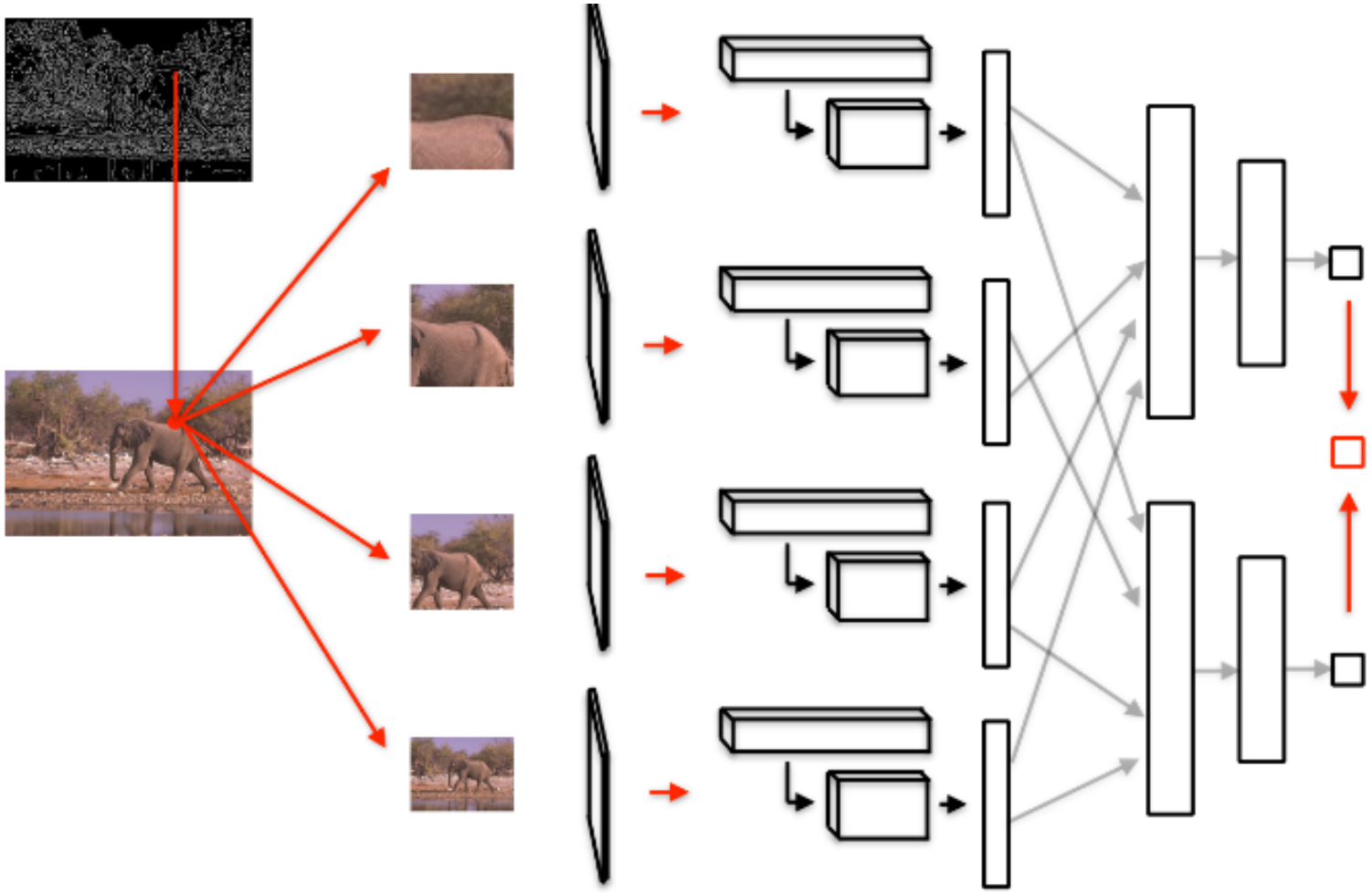
RCNN [Girshick et al. CVPR 2014]

Labeling Pixels: Semantic Labels



Fully Convolutional Networks for Semantic Segmentation [Long et al. CVPR 2015]

Labeling Pixels: Edge Detection

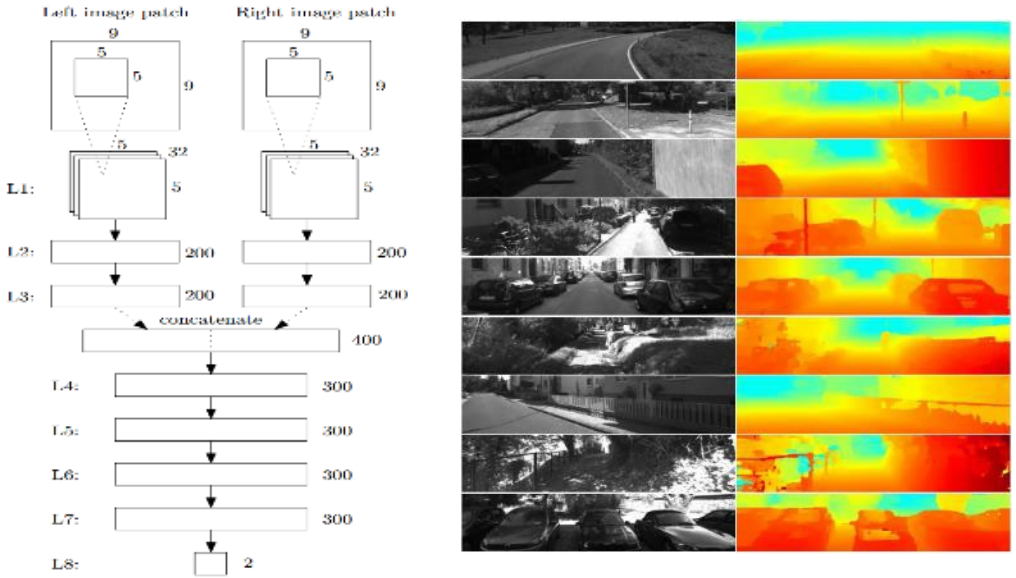


DeepEdge: A Multi-Scale Bifurcated Deep Network for Top-Down Contour Detection
[Bertasius et al. CVPR 2015]

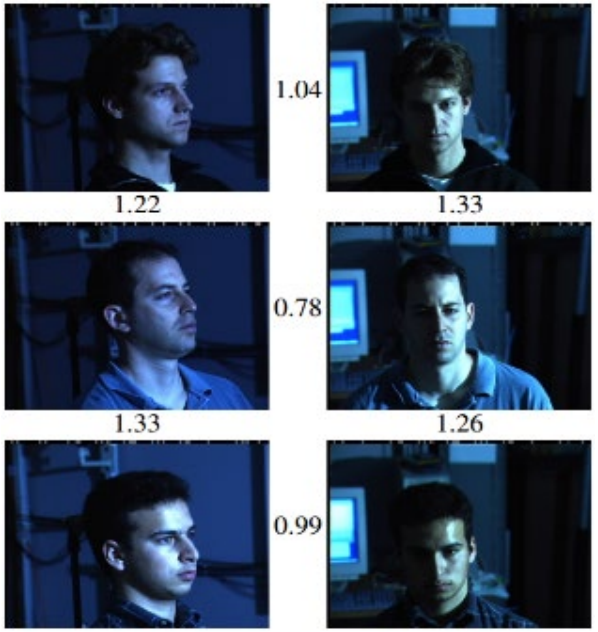
CNN for Regression

DeepPose [Toshev and Szegedy CVPR 2014]

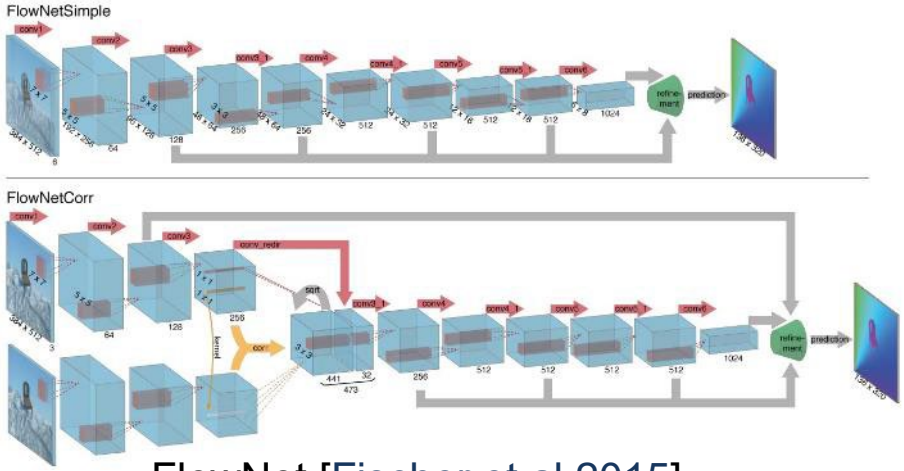
CNN as a Similarity Measure for Matching



Stereo matching [Zbontar and LeCun CVPR 2015]
Compare patch [Zagoruyko and Komodakis 2015]



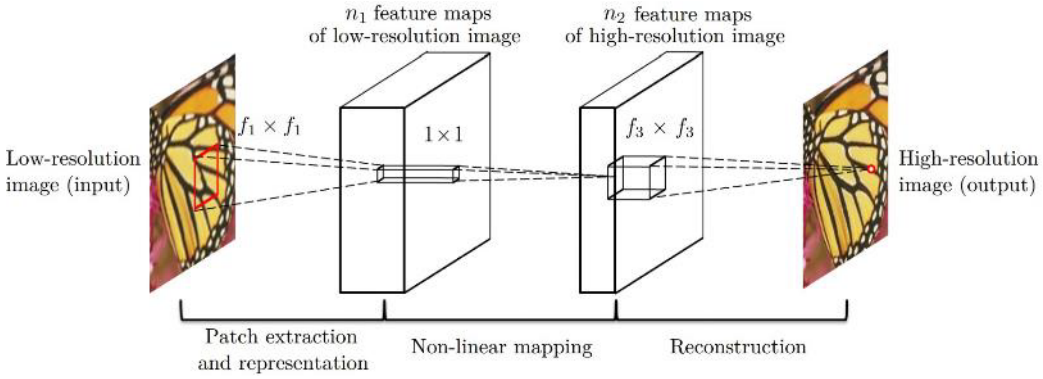
FaceNet [Schroff et al. 2015]



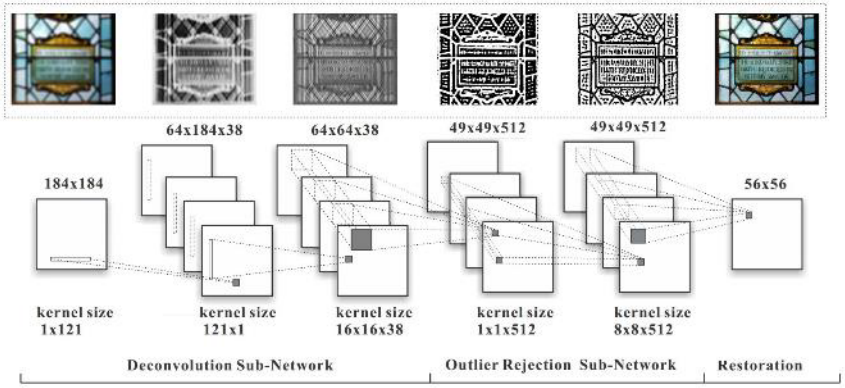
FlowNet [Fischer et al 2015]

Match ground and aerial images
[Lin et al. CVPR 2015]

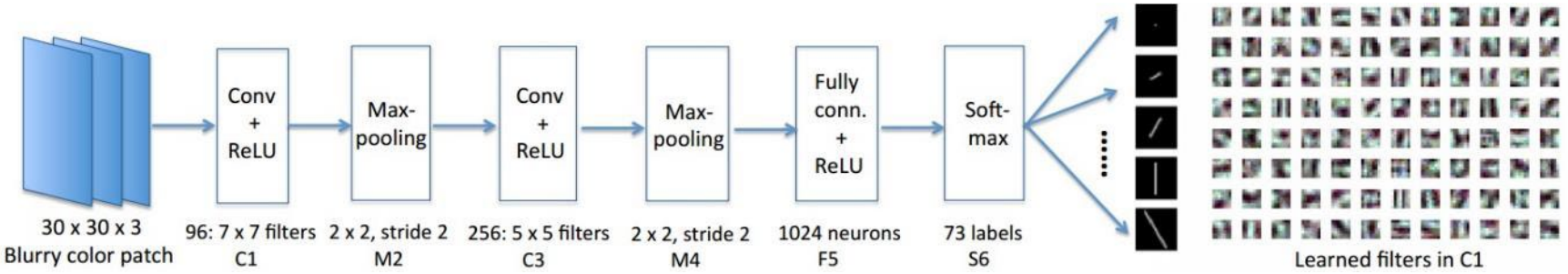
CNN for Image Restoration/Enhancement



Super-resolution
[Dong et al. ECCV 2014]

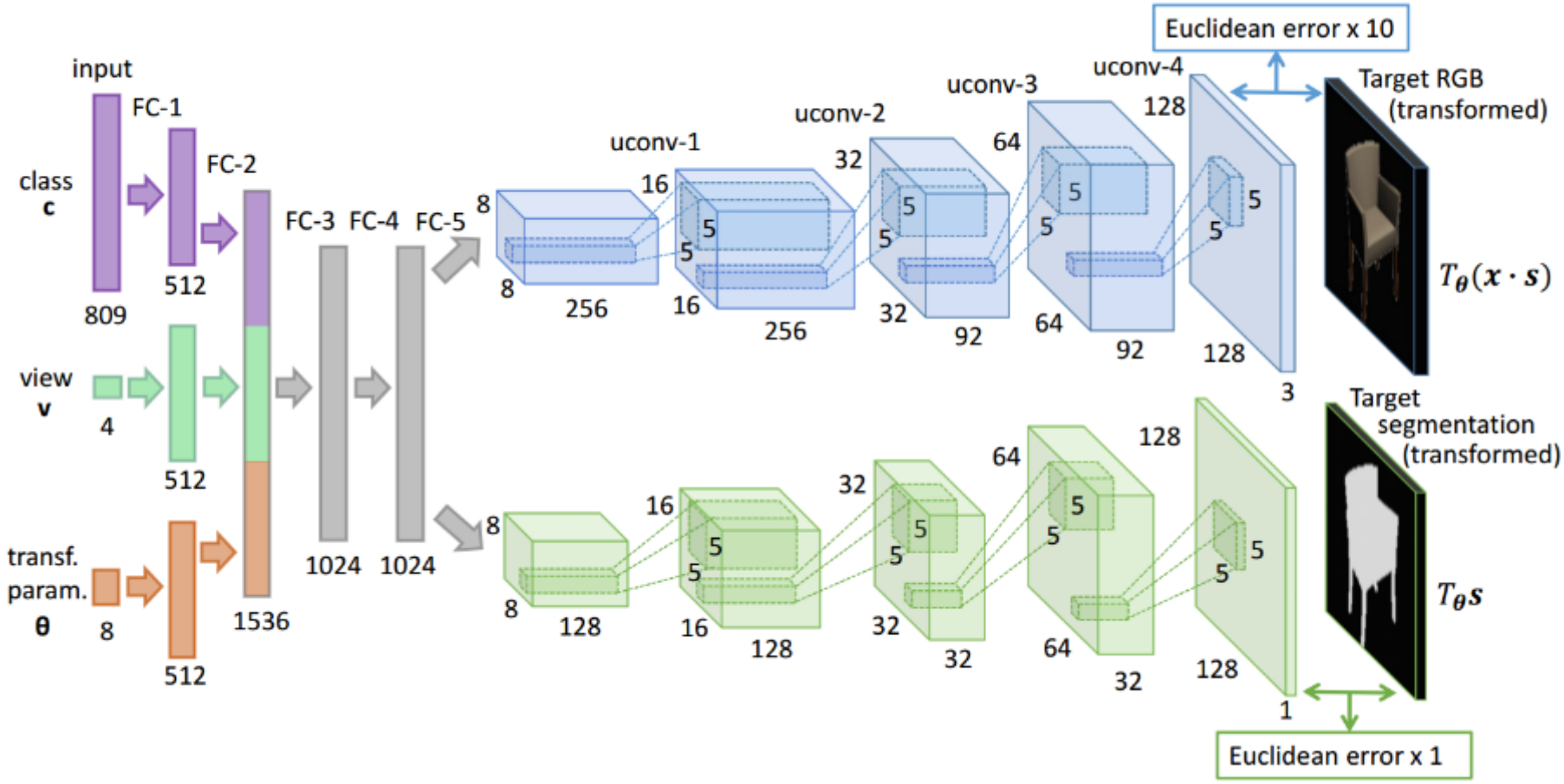


Non-blind deconvolution
[Xu et al. NIPS 2014]



Non-uniform blur estimation
[Sun et al. CVPR 2015]

CNN for Image Generation



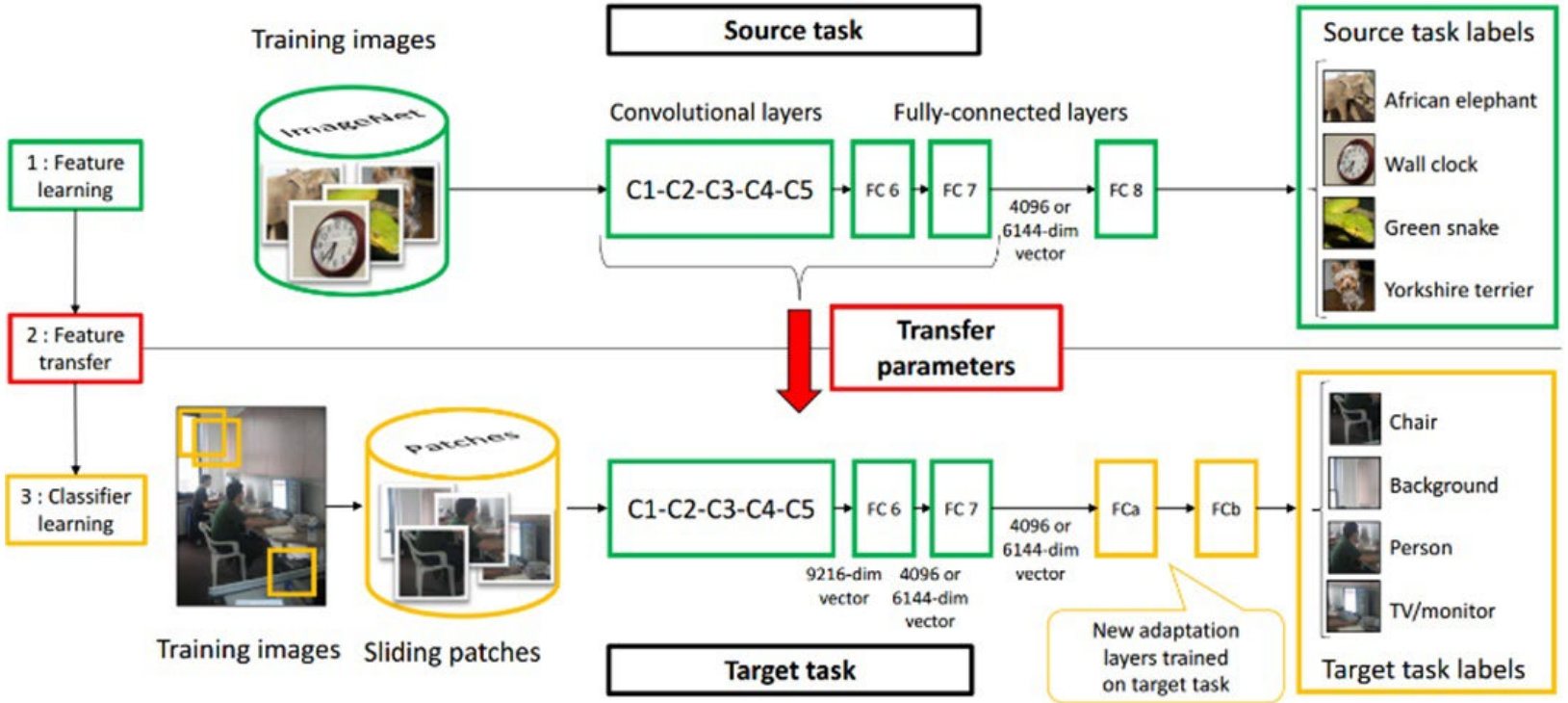
Learning to Generate Chairs with Convolutional Neural Networks [Dosovitskiy et al. CVPR 2015]

Chair Morphing

Learning to Generate Chairs with Convolutional Neural Networks [Dosovitskiy et al. CVPR 2015]

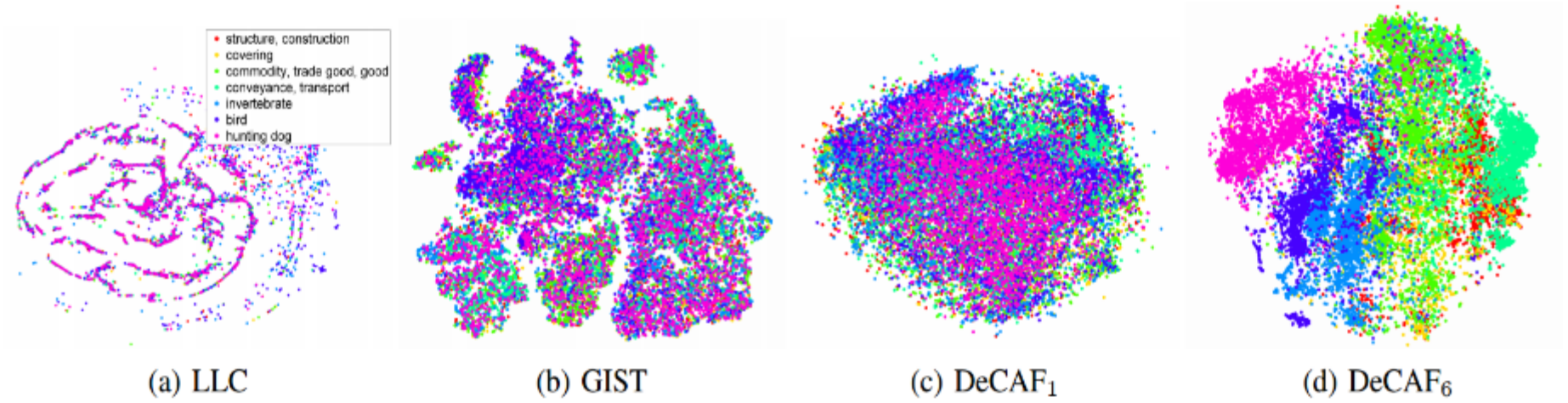
Transfer Learning

- Improvement of learning in a **new** task through the *transfer of knowledge* from a **related** task that has already been learned.
- Weight initialization for CNN

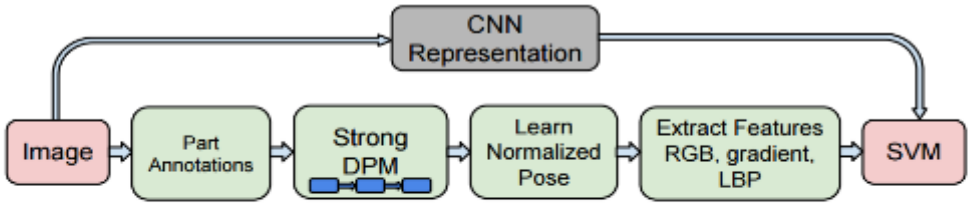


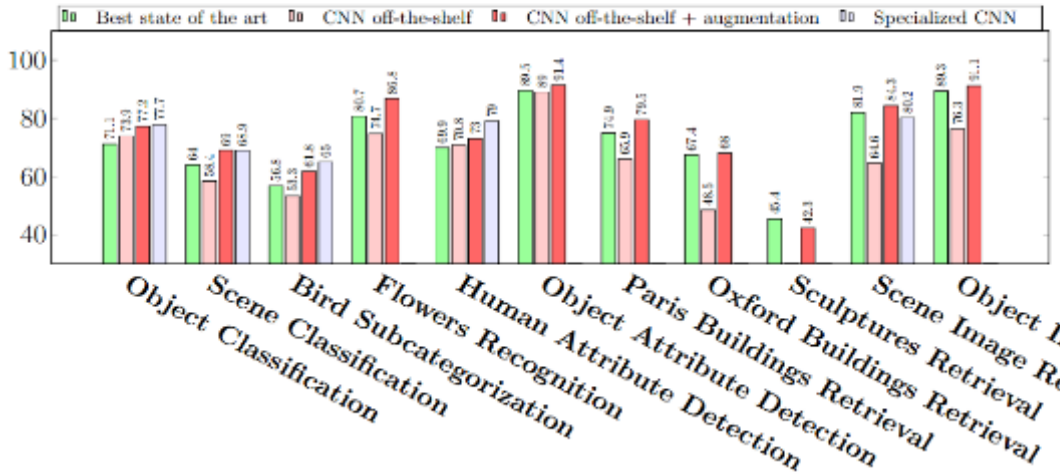
Learning and Transferring Mid-Level Image Representations using Convolutional Neural Networks [Oquab et al. CVPR 2014]

Convolutional activation features



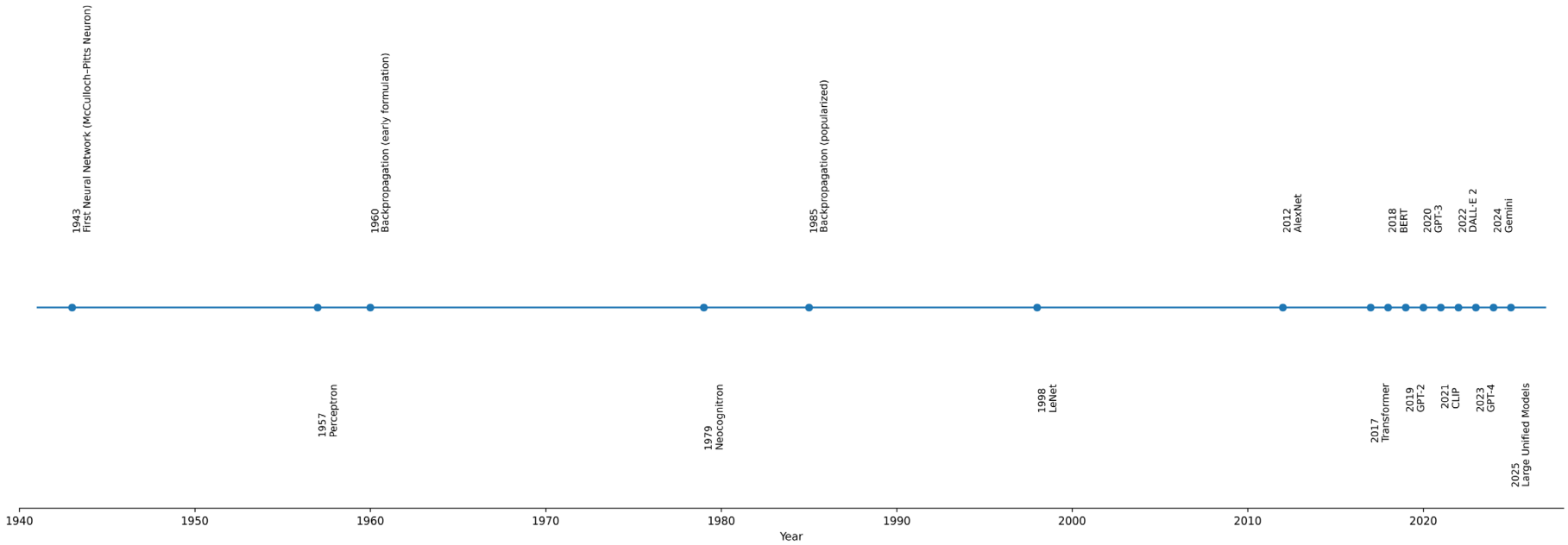
[Donahue et al. ICML 2013]



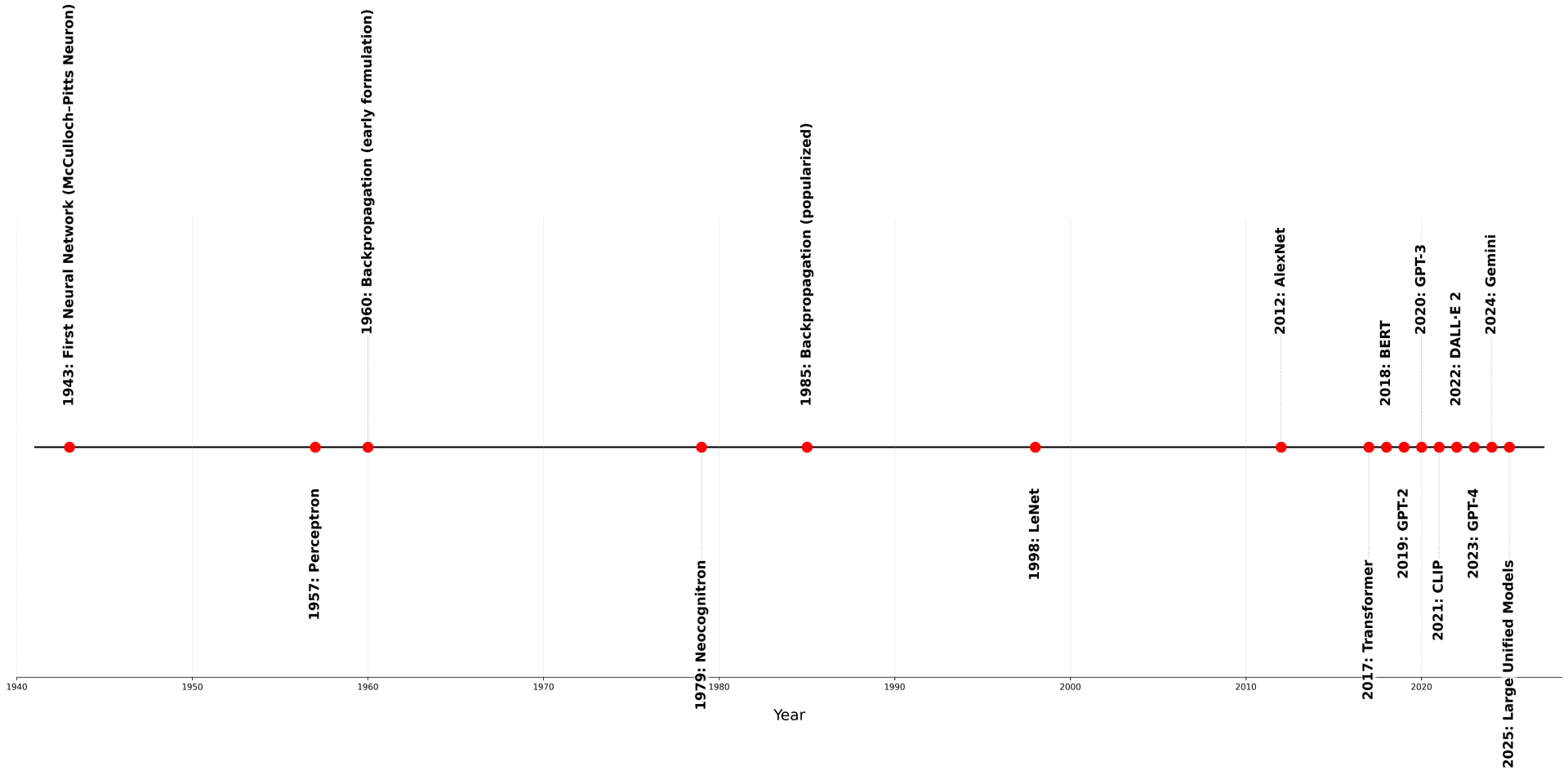


CNN Features off-the-shelf:
an Astounding Baseline for Recognition
[Razavian et al. 2014]

Timeline for different models



Timeline for different models



CNN packages

- Cuda-convnet (A. Krizhevsky, Google)
- Caffe (Y. Jia, Berkeley)
  - Replacement of deprecated Decaf
- Overfeat (NYU)
- Torch
- MatConvNet (A. Vedaldi, Oxford)

Resources

- http://deeplearning.net/
- https://github.com/ChristosChristofidis/awesome-deep-learning
- http://cs231n.stanford.edu/syllabus.html