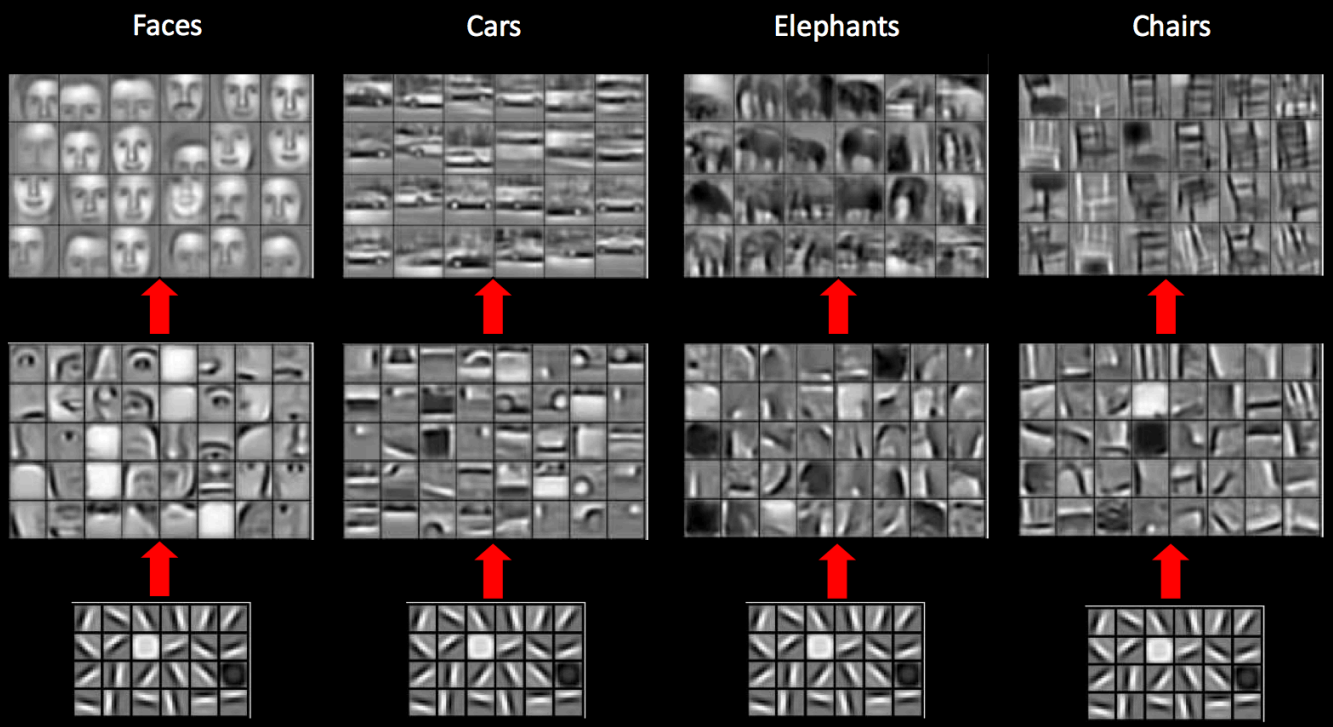
Machine Learning for Computer Vision



Lecture 8: Deep Learning for High-Level Vision

Iasonas Kokkinos

Iasonas.kokkinos@ecp.fr

Center for Visual Computing
Ecole Centrale Paris

Galen Group
INRIA-Saclay

Slide credits & pointers

R. Fergus/K. Yu/M. A. Ranzatto (CVPR 12 Tutorial):
http://cs.nyu.edu/~fergus/tutorials/deep_learning_cvpr12/
S. Liu/S. Roth (ICCV 09 Tutorial):
http://www.gris.informatik.tu-darmstadt.de/teaching/iccv2009/
E. Simoncelli (Class notes):
http://www.cns.nyu.edu/~eero/imrep-course/
A. Vedadi (BMVC14 Tutorial)
http://www.robots.ox.ac.uk/~vedaldi/assets/teach/vedaldi14bmvc-tutorial.pdf
Y. Bengio (Tutorial):
http://www.iro.umontreal.ca/~bengioy/talks/mlss-beijing.pdf
G. Papandreou, BASIS Tutorial
http://cvn.ecp.fr/personnel/iasonas/basis14/
Fei-Fei Li, Olga Russakovsky
http://ai.stanford.edu/~olga/slides/ImageNetAnalysis_bavm_10_5_13.pptx
R. Salakhudinov
http://www.cs.toronto.edu/~rsalakhu/kdd.html
IPAM summer school on deep learning:
http://www.ipam.ucla.edu/programs/summer-schools/graduate-summer-school-deep-learning-feature-learning/

The computer vision quest

Backpack

Flute

Strawberry

Traffic light

Backpack

Matchstick

Sea lion

Bathing cap

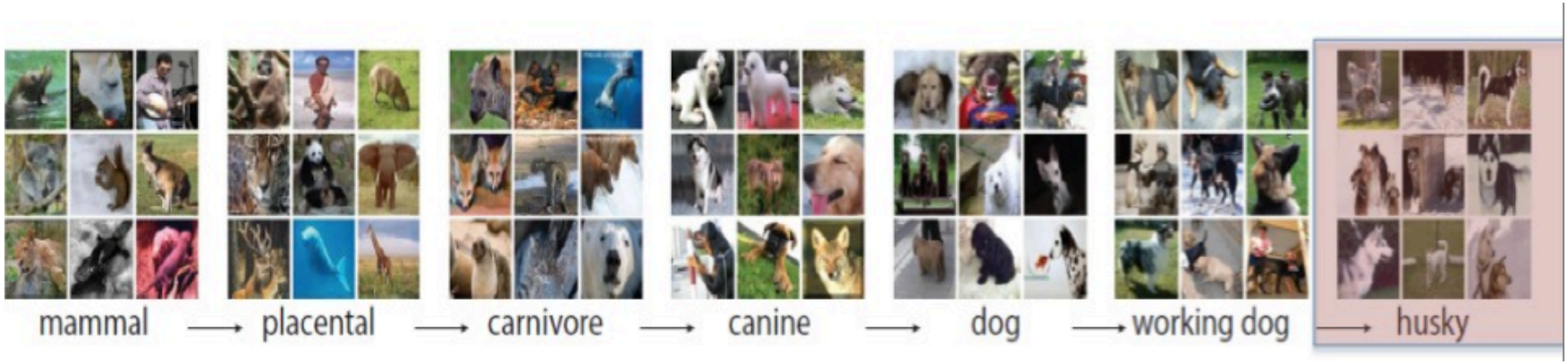
Racket


WAIT

Computer Vision Data: Big and Complicated

http://www.image-net.org/

IMAGENET



Deng et. al, ImageNet: A Large-Scale Hierarchical Image Database, CVPR'09

Computer Vision Data: Big and Complicated

http://www.image-net.org/

IMAGENET

Overall

- Total number of non-empty synsets: 21841
- Total number of images: 14,197,122
- Number of images with bounding box annotations: 1,034,908

High level category	# synset (subcategories)	Avg # images per synset	Total # images
amphibian	94	591	56K
animal	3822	732	2799K
appliance	51	1164	59K
bird	856	949	812K
covering	946	819	774K
device	2385	675	1610K
fabric	262	690	181K
fish	566	494	280K
flower	462	735	339K
food	1495	670	1001K
fruit	309	607	188K
fungus	303	453	137K
furniture	187	1043	195K
geological formation	151	838	127K
invertebrate	728	573	417K
mammal	1138	821	934K
musical instrument	157	891	140K
plant	1666	600	999K
reptile	268	707	190K
sport	166	1207	200K
structure	1239	763	946K
tool	316	551	174K
tree	993	568	564K
utensil	86	912	78K
vegetable	176	764	135K
vehicle	481	778	374K
person	2035	468	952K

Deng et. al, ImageNet: A Large-Scale Hierarchical Image Database, CVPR'09

Computer Vision Data: Big and Complicated

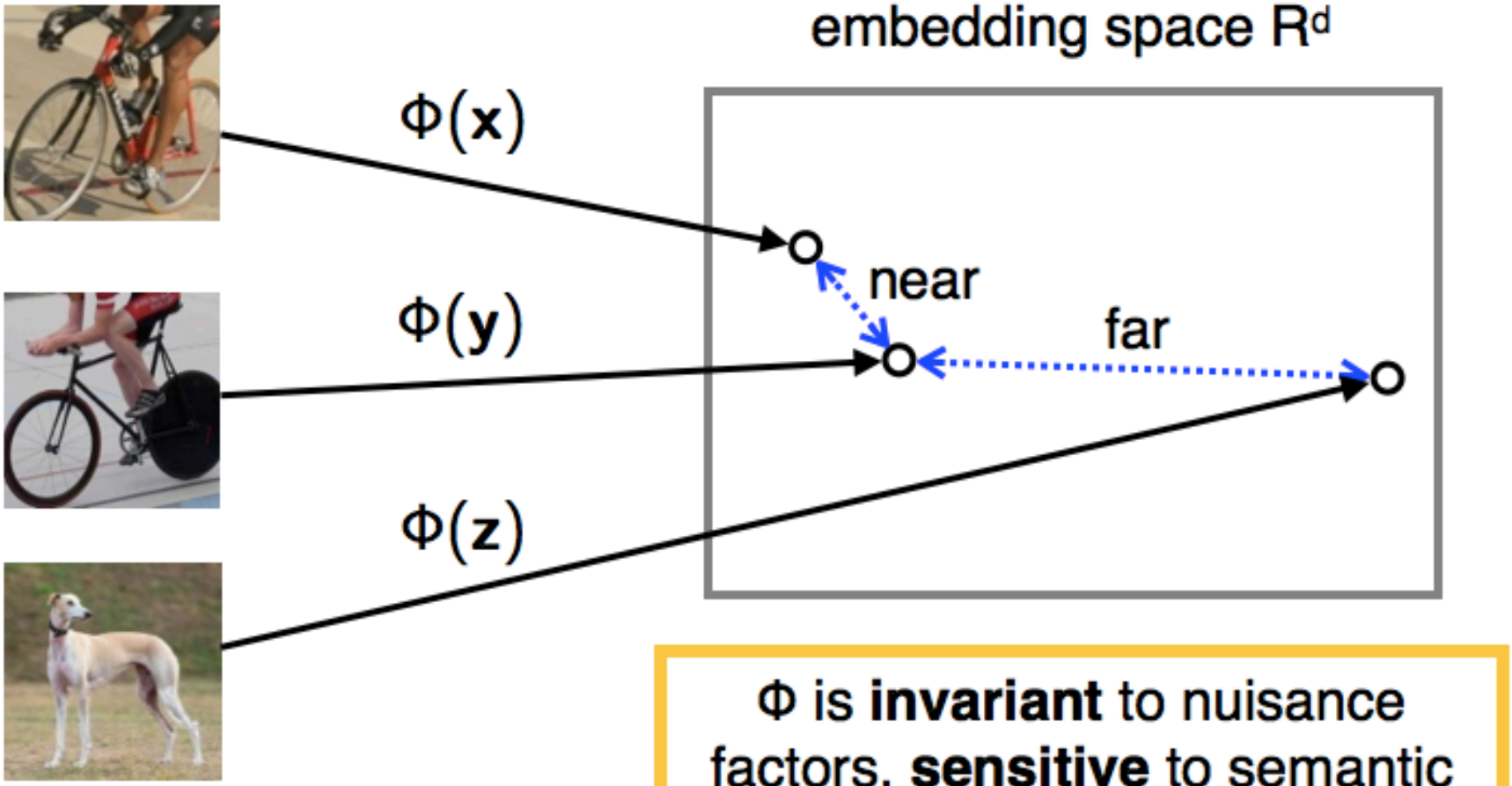
Examples of hammer:

The real challenge: image features

$\mathbf{x}$ 

encoder Φ → representation $\Phi(\mathbf{x}) \in \mathbb{R}^d$

Desirable feature properties



1980's

pixels → edge → texton → motif → part → object

2000-2010

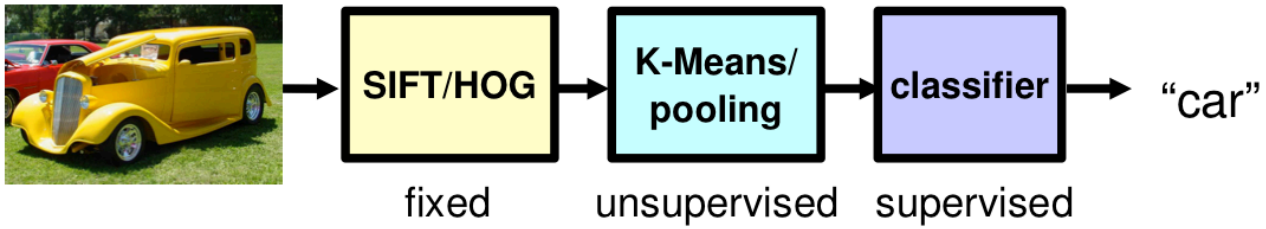
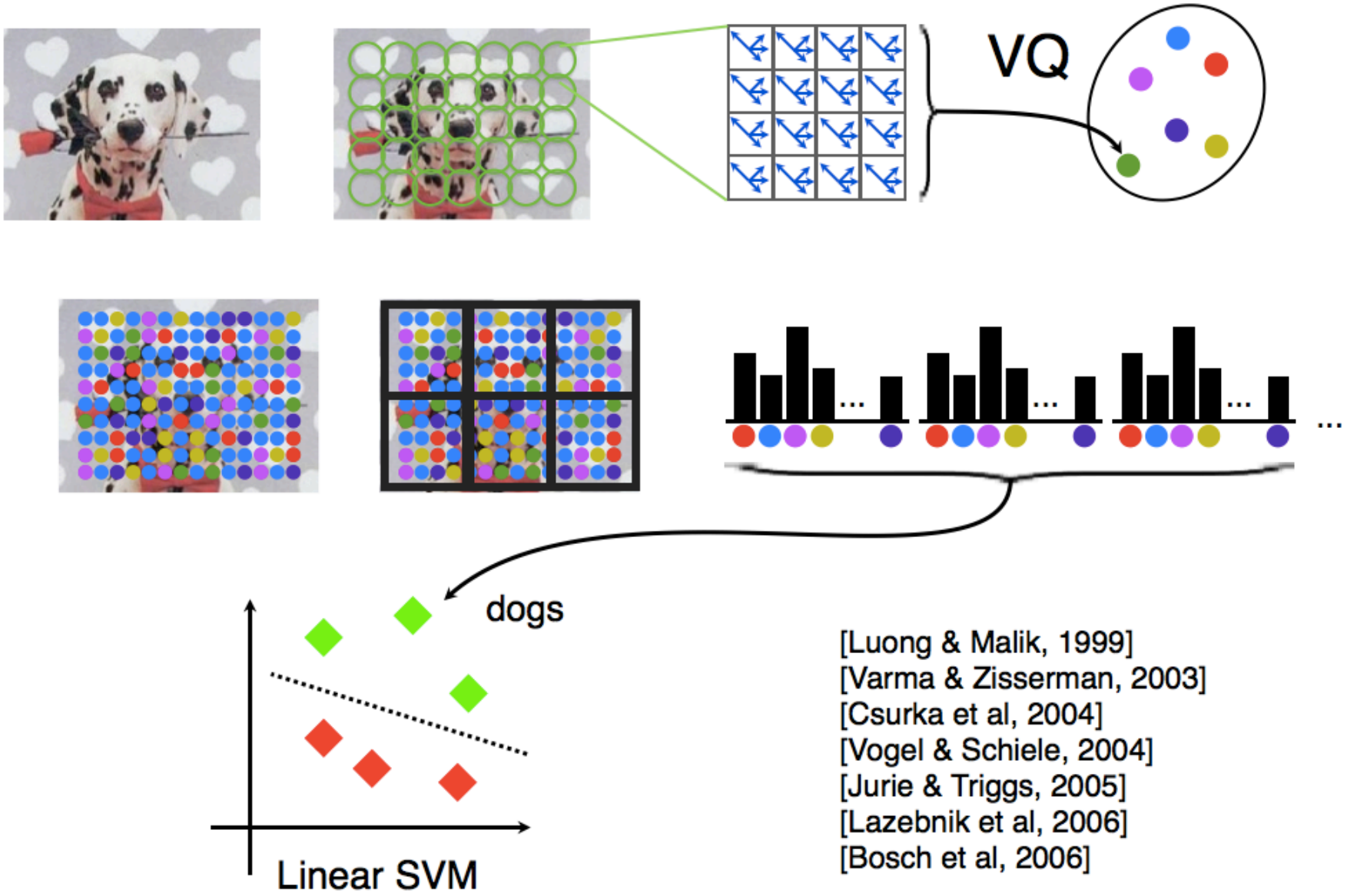


Image classification in a nutshell



[Luong & Malik, 1999]
[Varma & Zisserman, 2003]
[Csurka et al, 2004]
[Vogel & Schiele, 2004]
[Jurie & Triggs, 2005]
[Lazebnik et al, 2006]
[Bosch et al, 2006]

Deep Learning and Computer Vision

1980's pixels → edge → texton → motif → part → object

2000-2010

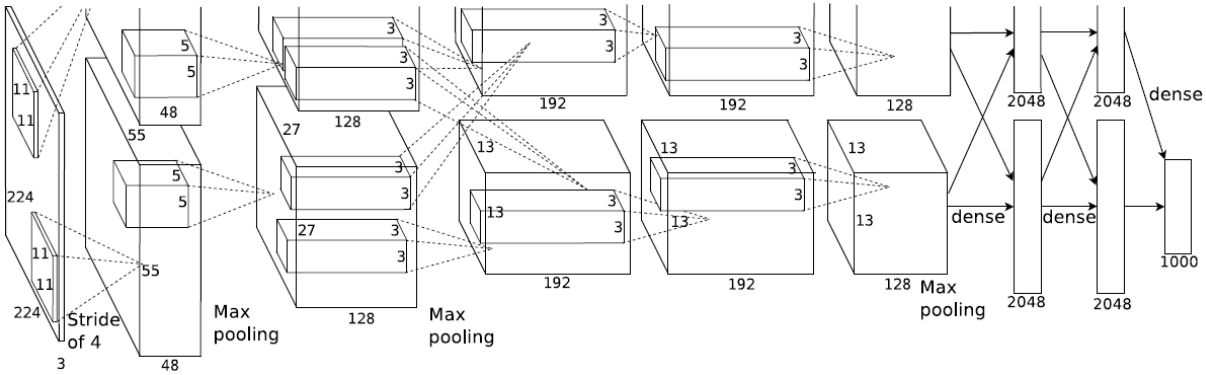
SIFT/HOG → K-Means/ pooling → classifier → "car"

fixed unsupervised supervised

2010+

$x \rightarrow max(0, W^1 x) \rightarrow h^1 \rightarrow max(0, W^2 h^1) \rightarrow h^2 \rightarrow W^3 h^2 \rightarrow o$

Breakthrough: Imagenet 2012

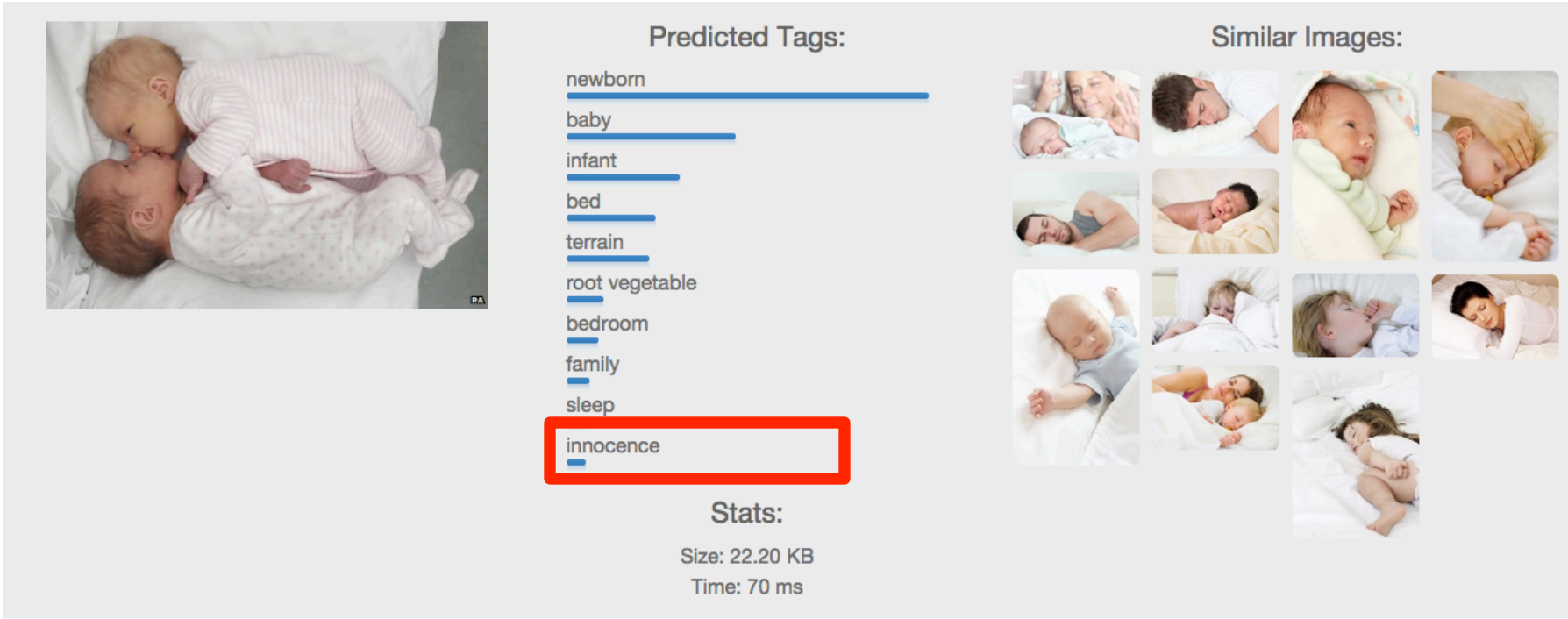


A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. *NIPS*13

Imagenet top-5 error rate: 36%-> 18% (2012) -> 6% (2014)

A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. *NIPS*13

Demo from http://www.clarifai.com/



Demo from http://www.clarifai.com/

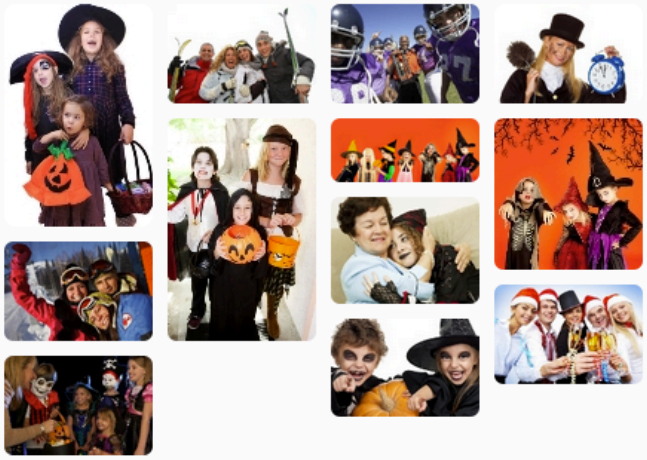
Predicted Tags:

fashion
party
hat
two
family
portrait
female
celebration
friendship
costume

Stats:

Size: 87.78 KB
Time: 48 ms

Similar Images:



1 4387

The spirit of a dog reincarnated as a tree destined to become a door and end up online

(i.imgur.com)

submitted 9 hours ago by GallowBoob

509 comments share

Demo from http://www.clarifai.com/

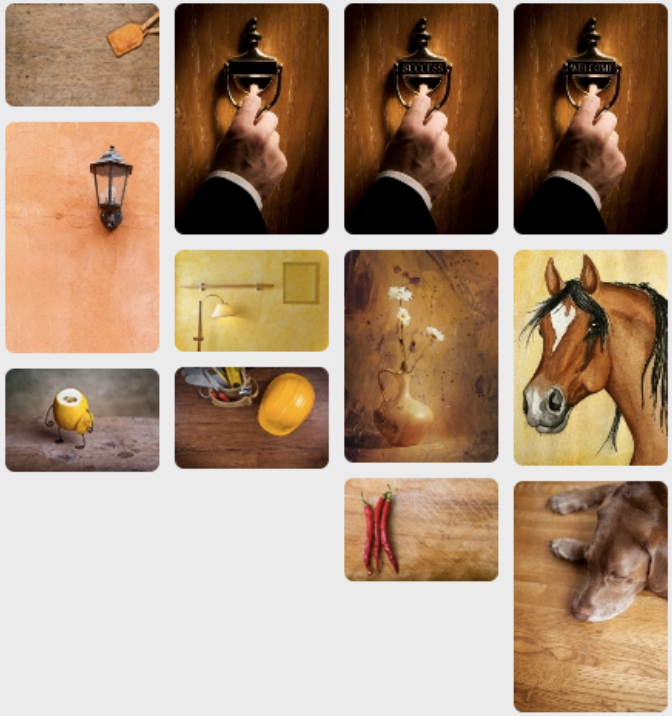
Predicted Tags:

wood
door
wall
cute
texture
abstract
dark
colour

Stats:

Size: 69.47 KB
Time: 66 ms

Similar Images:



Demo from http://www.clarifai.com/

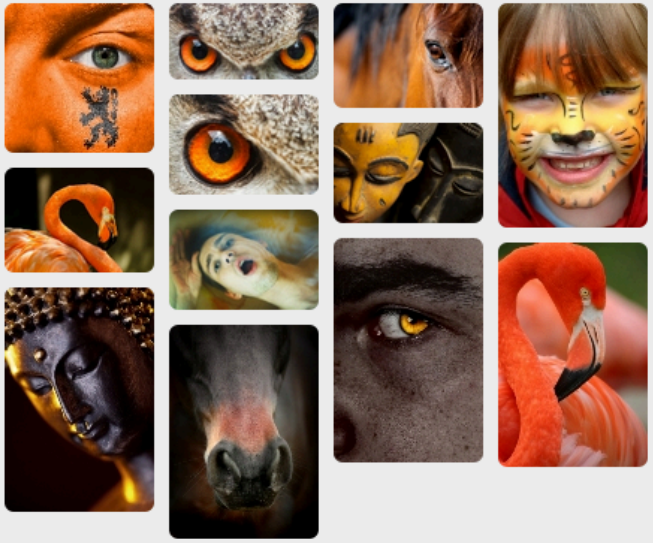
Predicted Tags:

baby
nose
cute
nobody
mammal
eyes
pet
small
blur
mouth

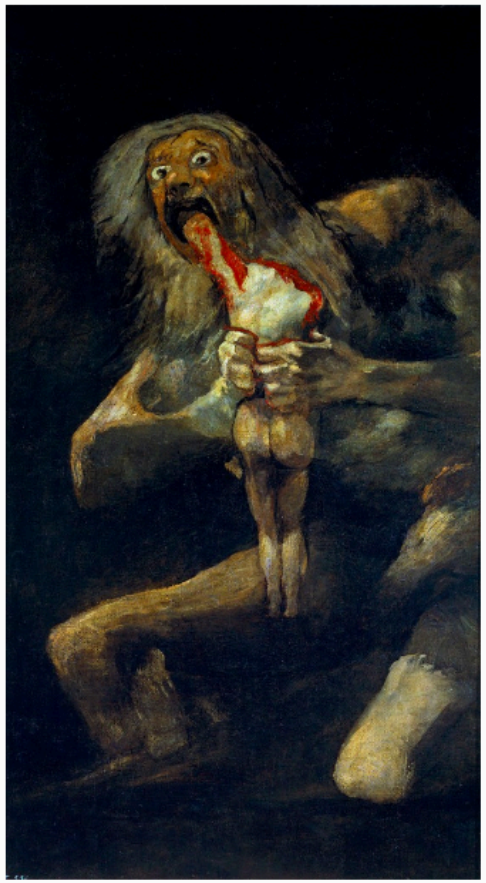
Stats:

Size: 152.83 KB
Time: 52 ms

Similar Images:



Demo from http://www.clarifai.com/



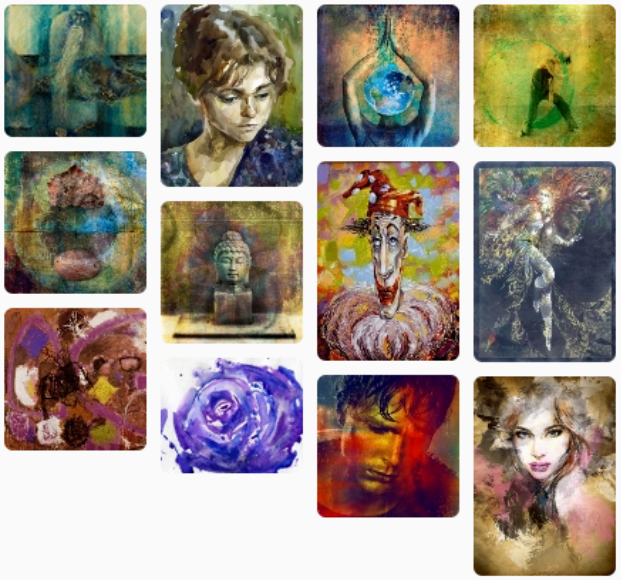
Predicted Tags:

fine art
painting
modernist
cooking oil
european
death
login
art
sculpture
god

Stats:

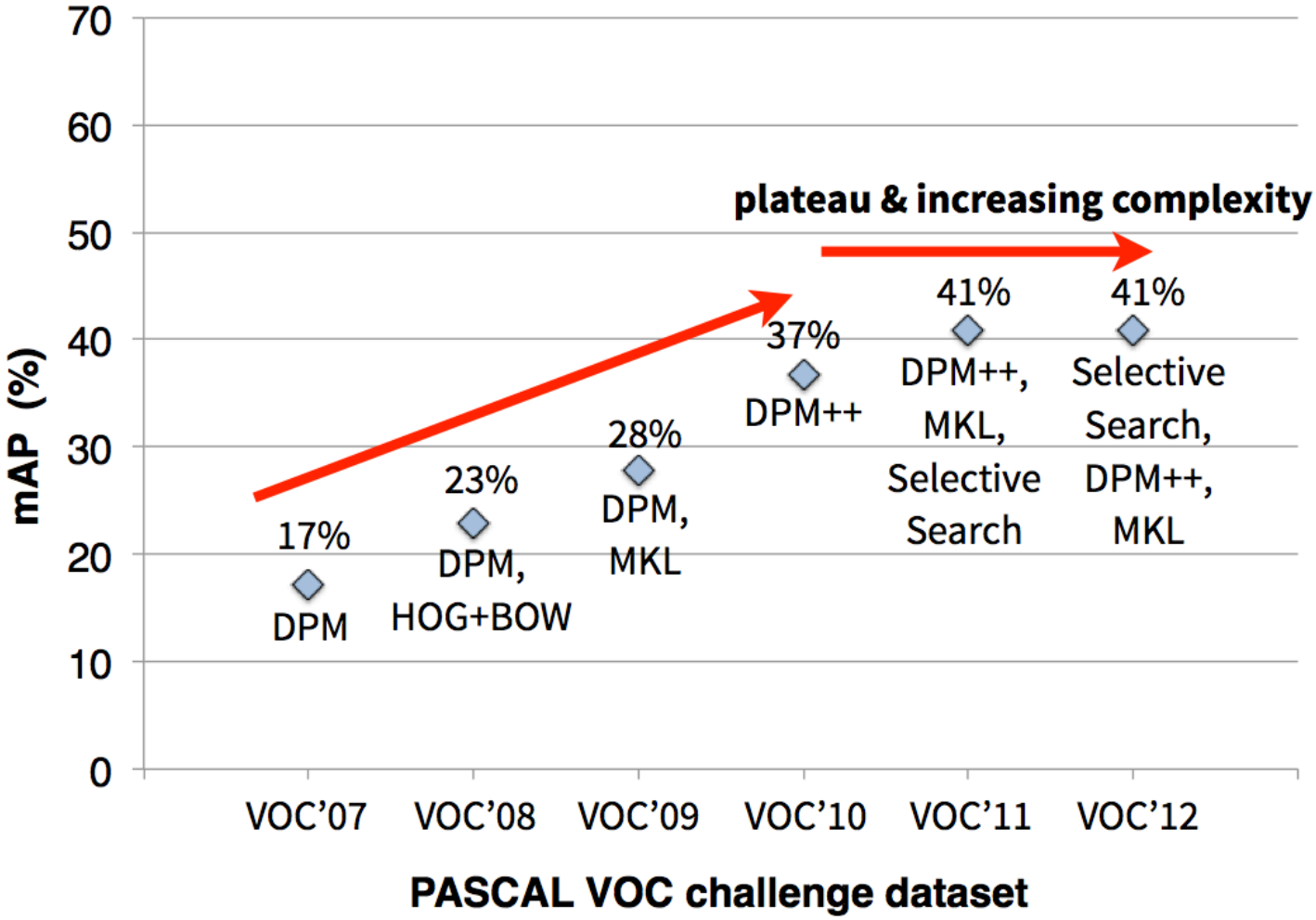
Size: 188.54 KB
Time: 81 ms

Similar Images:



Saturn devouring his son, F. Goya

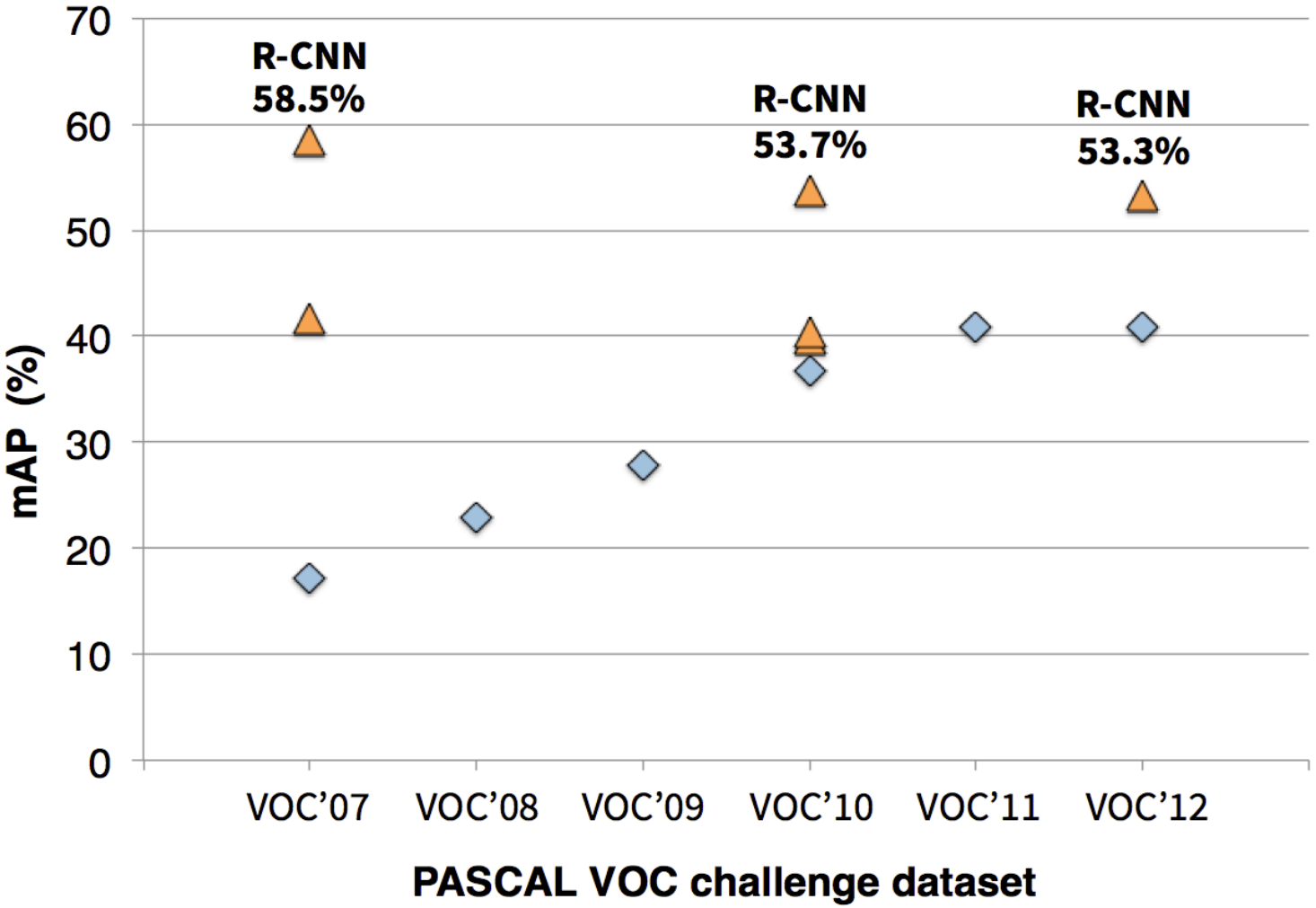
Object recognition 2010-2012: saturation



2005

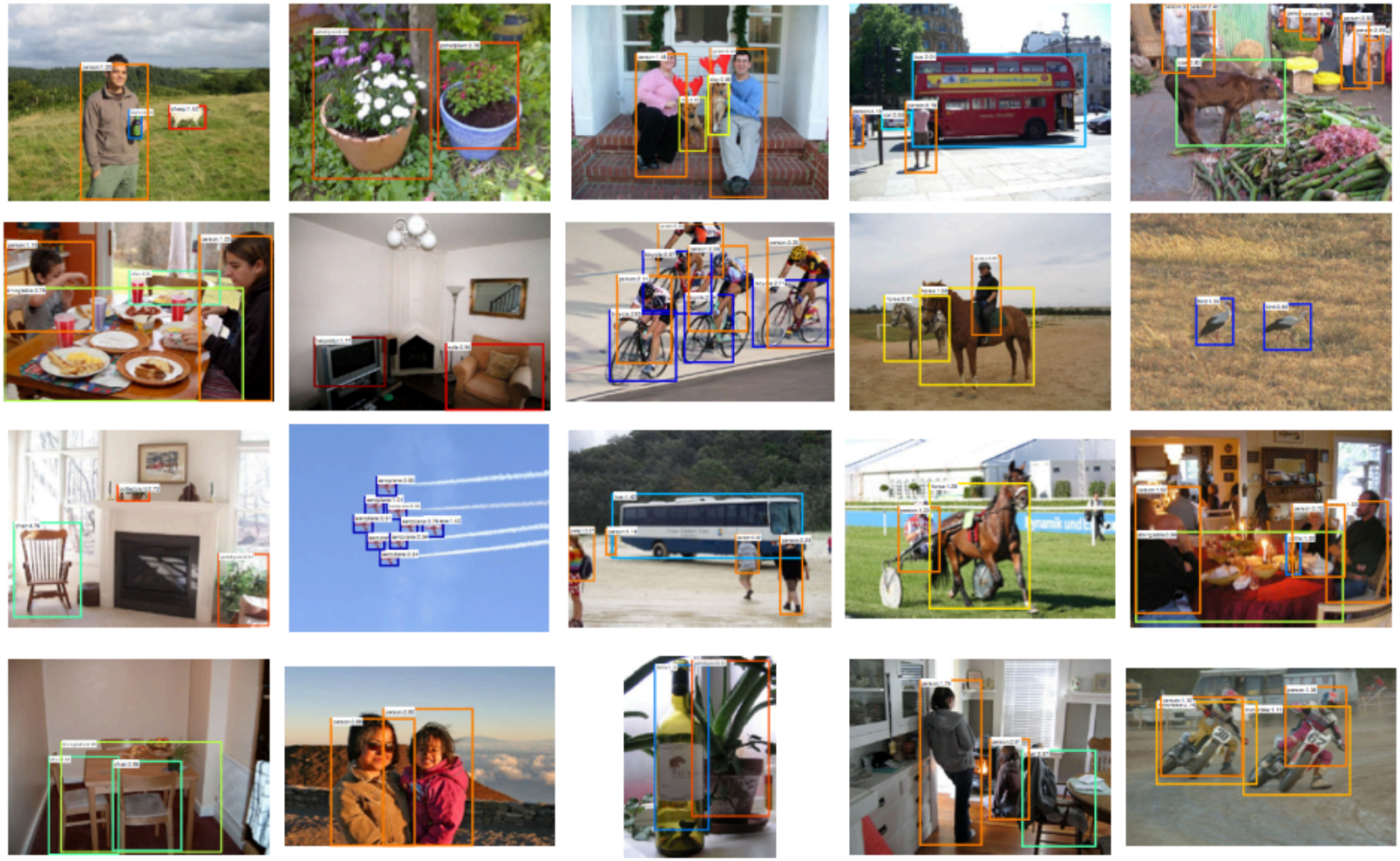
2012

Object recognition, 2013



A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. *NIPS*13
P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. ICLR 14.
R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CVPR* 14

Object recognition, 2014



A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. *NIPS*13
P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. ICLR 14.
R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CVPR* 14

Deep Learning and AI

DPM

VISION

pixels → edge → texton → motif → part → object

SPEECH

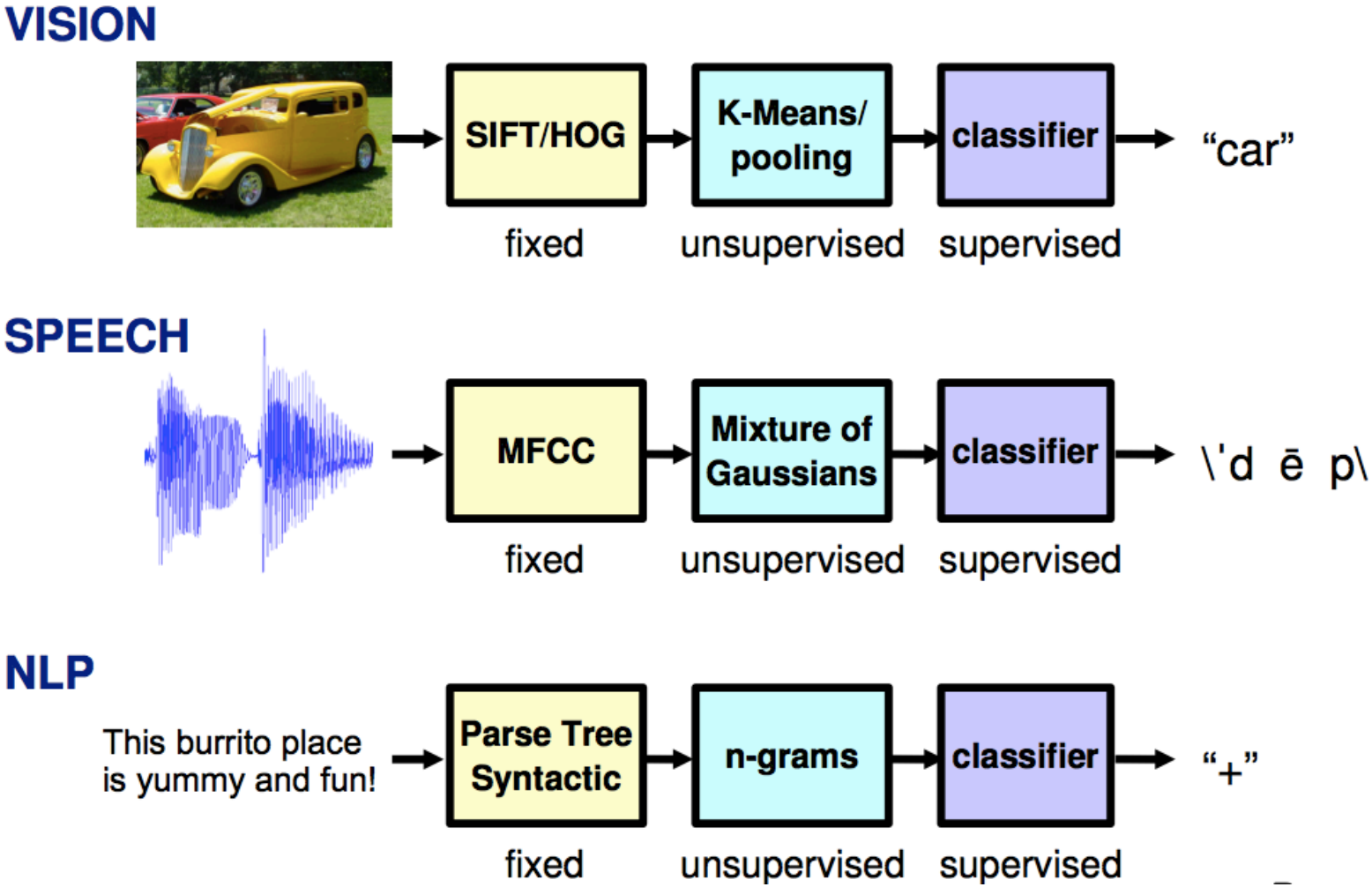
sample → spectral band → formant → motif → phone → word

NLP

character → word → NP/VP/.. → clause → sentence → story

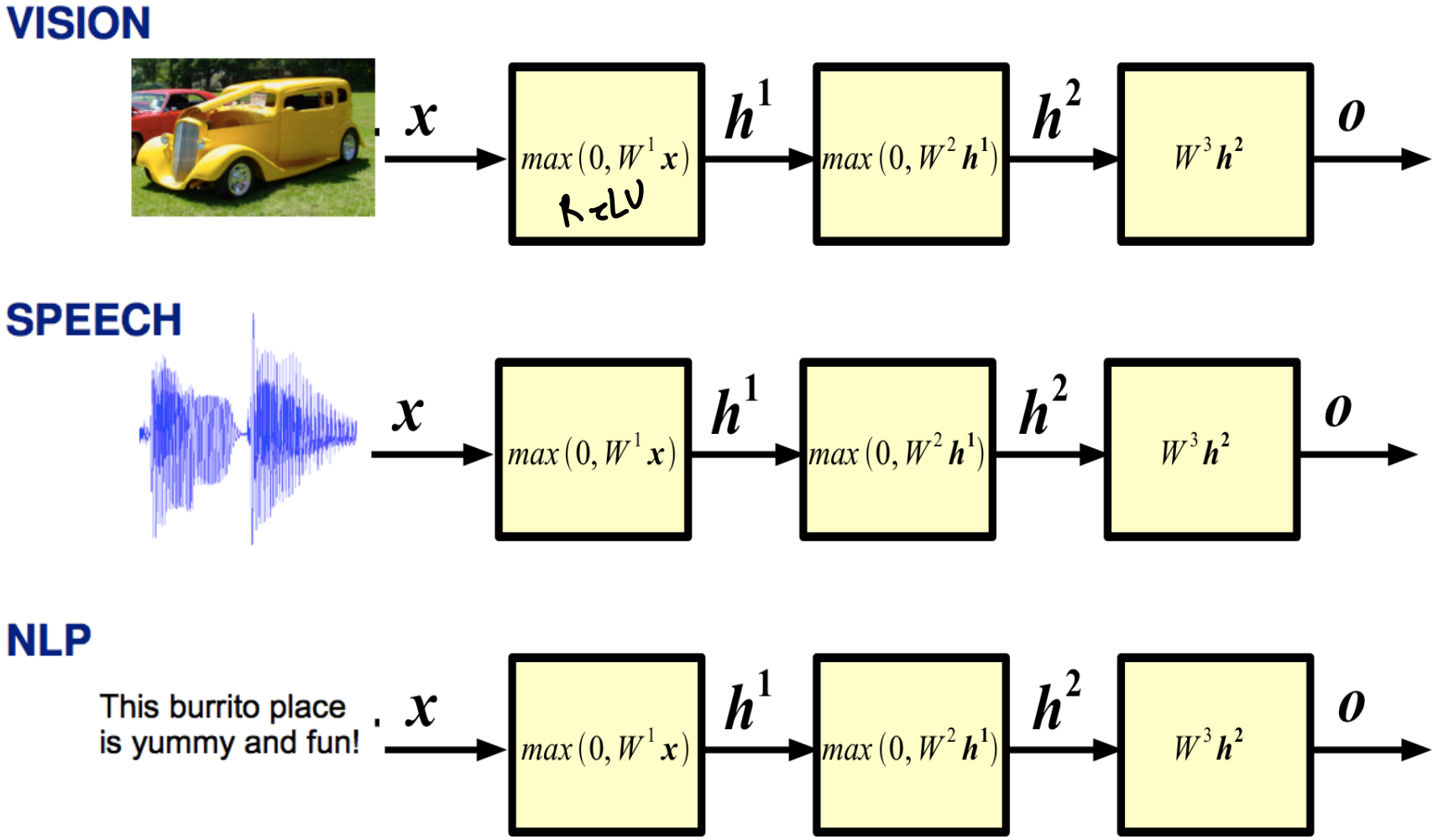
Slide credits: M-A. Ranzatto

Deep Learning and AI



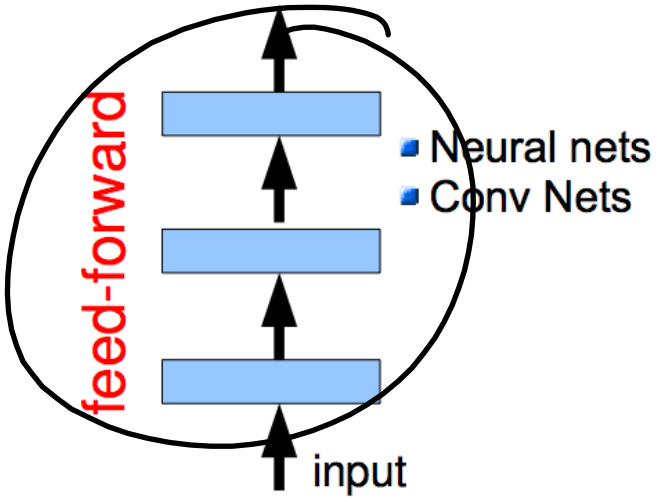
Slide credits: M-A. Ranzatto

Deep Learning and AI

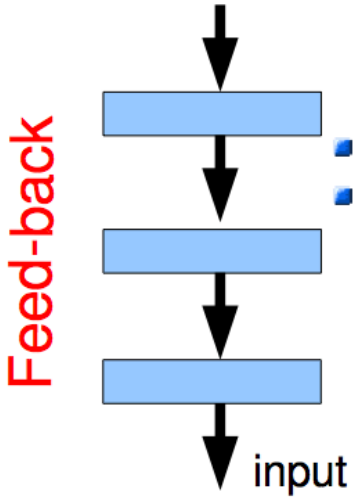


'Deep Learning reads Wikipedia and discovers the meaning of life'
Geoff Hinton

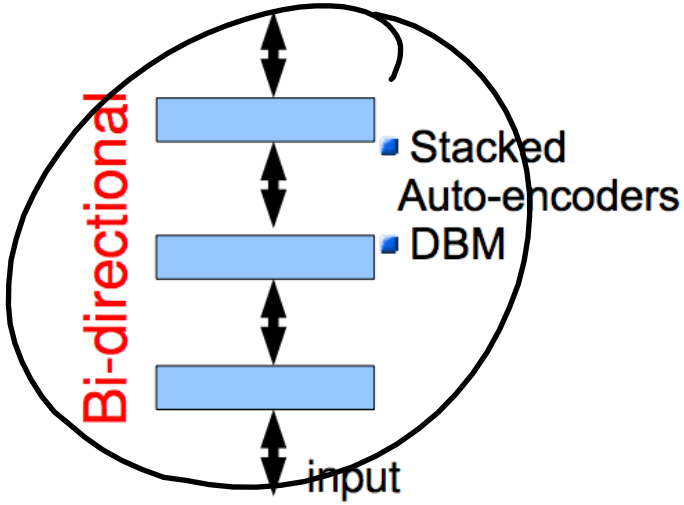
- Main types of deep architectures



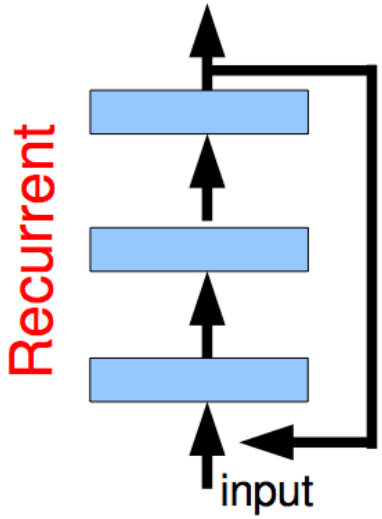
- Neural nets
- Conv Nets



- Hierar. Sparse Coding
- Deconv Nets

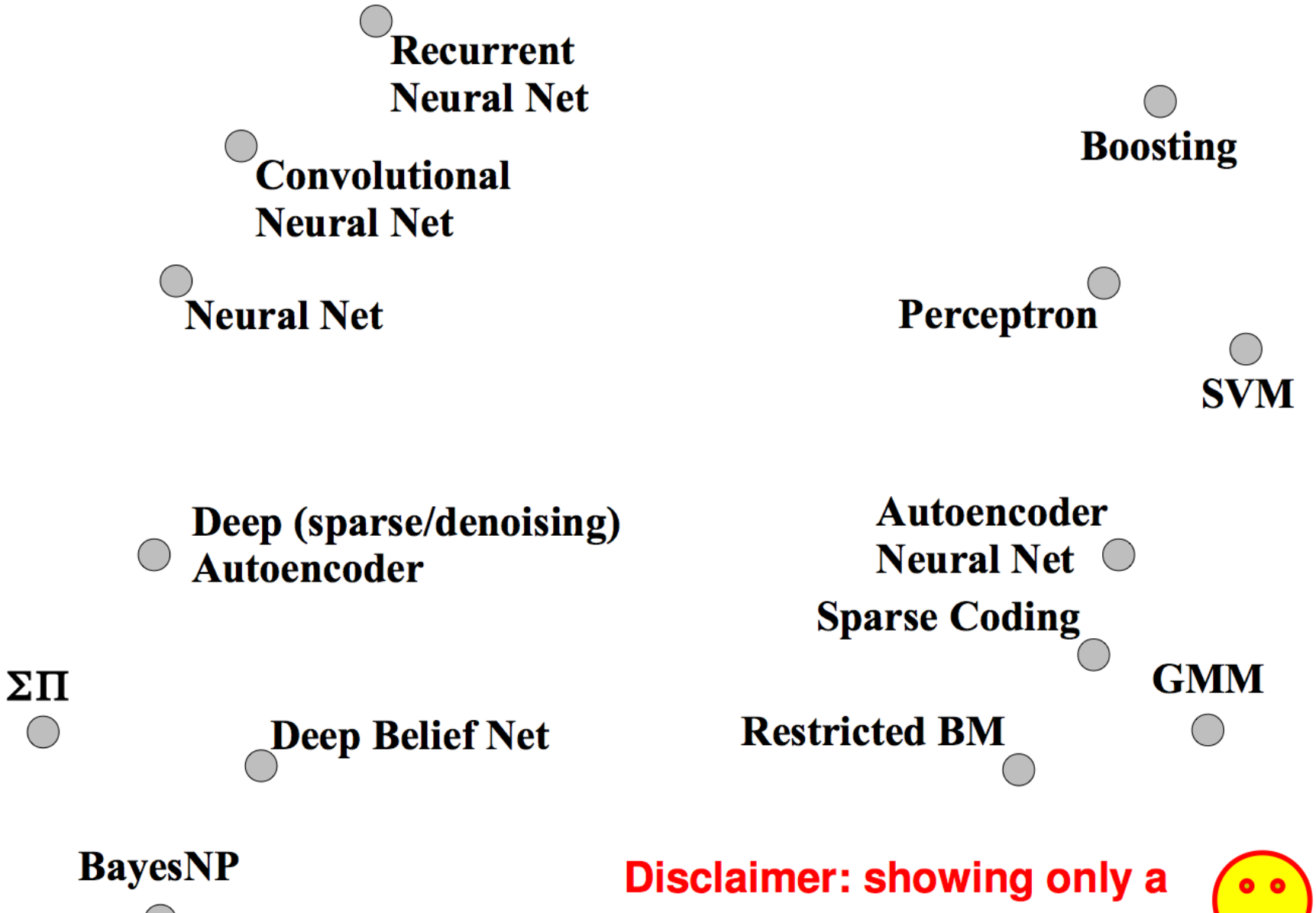


- Stacked Auto-encoders
- DBM



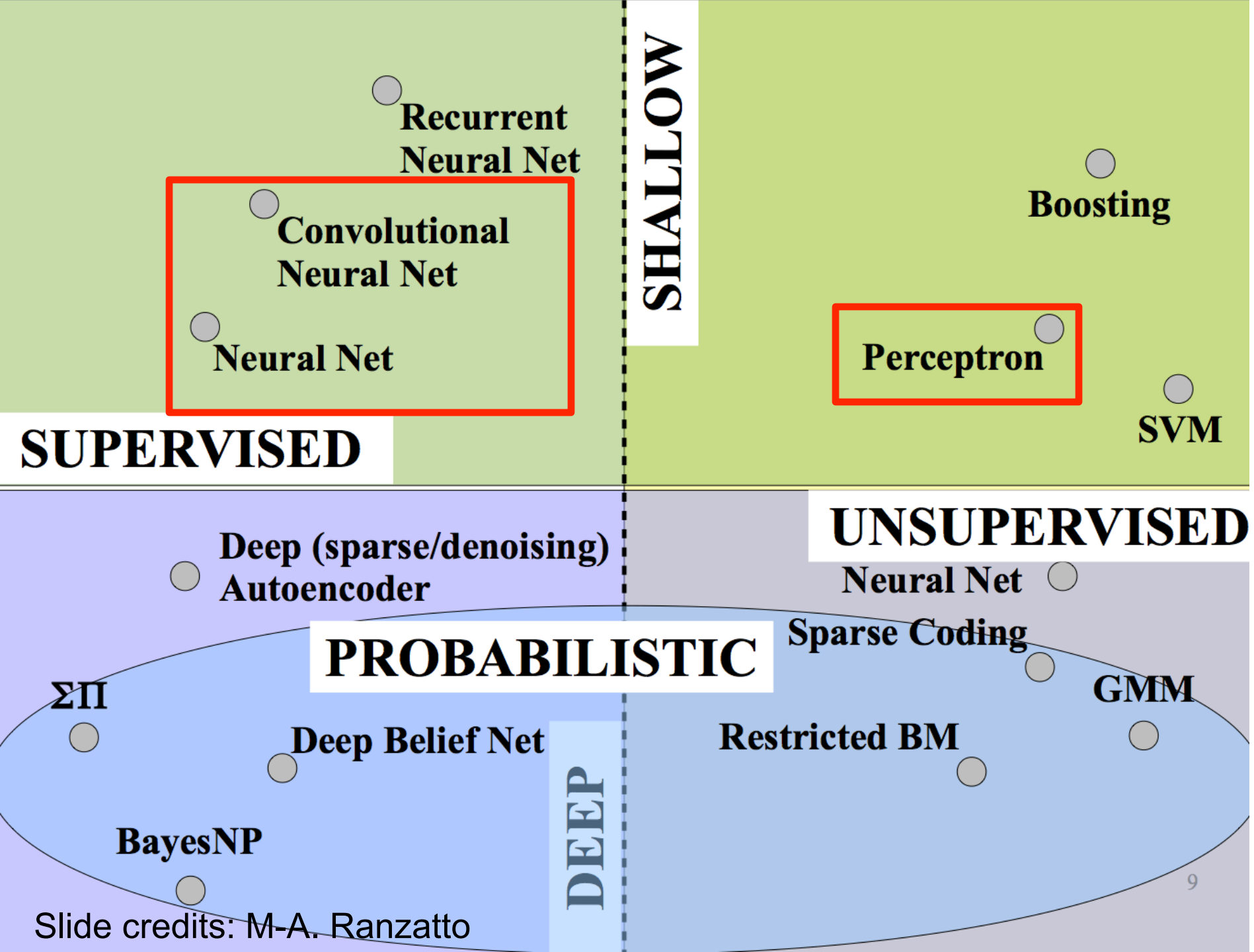
- Recurrent Neural nets
- Recursive Nets
- LISTA

Ranzato



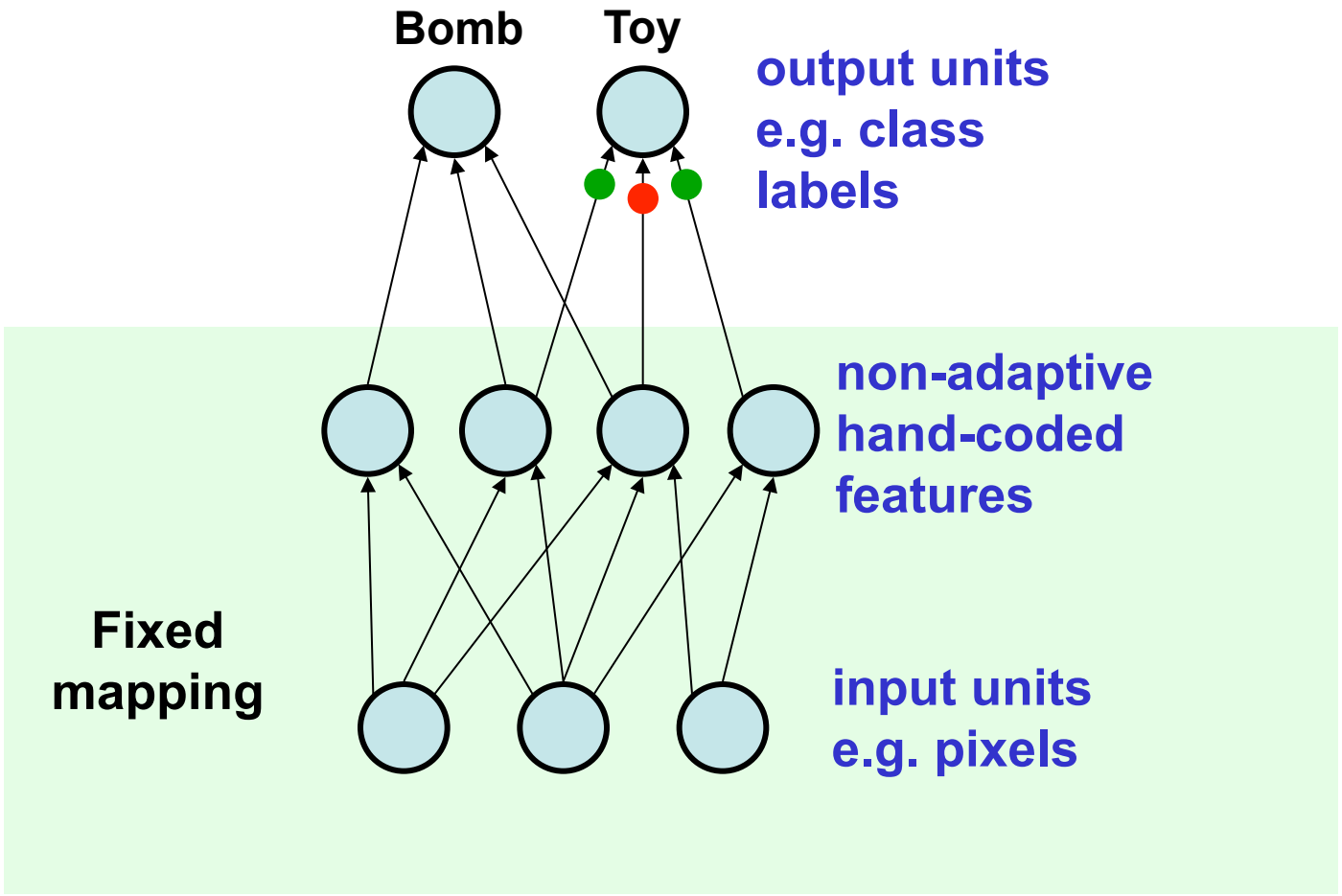
Slide credits: M-A. Ranzatto

Disclaimer: showing only a subset of the known methods



Slide credits: M-A. Ranzatto

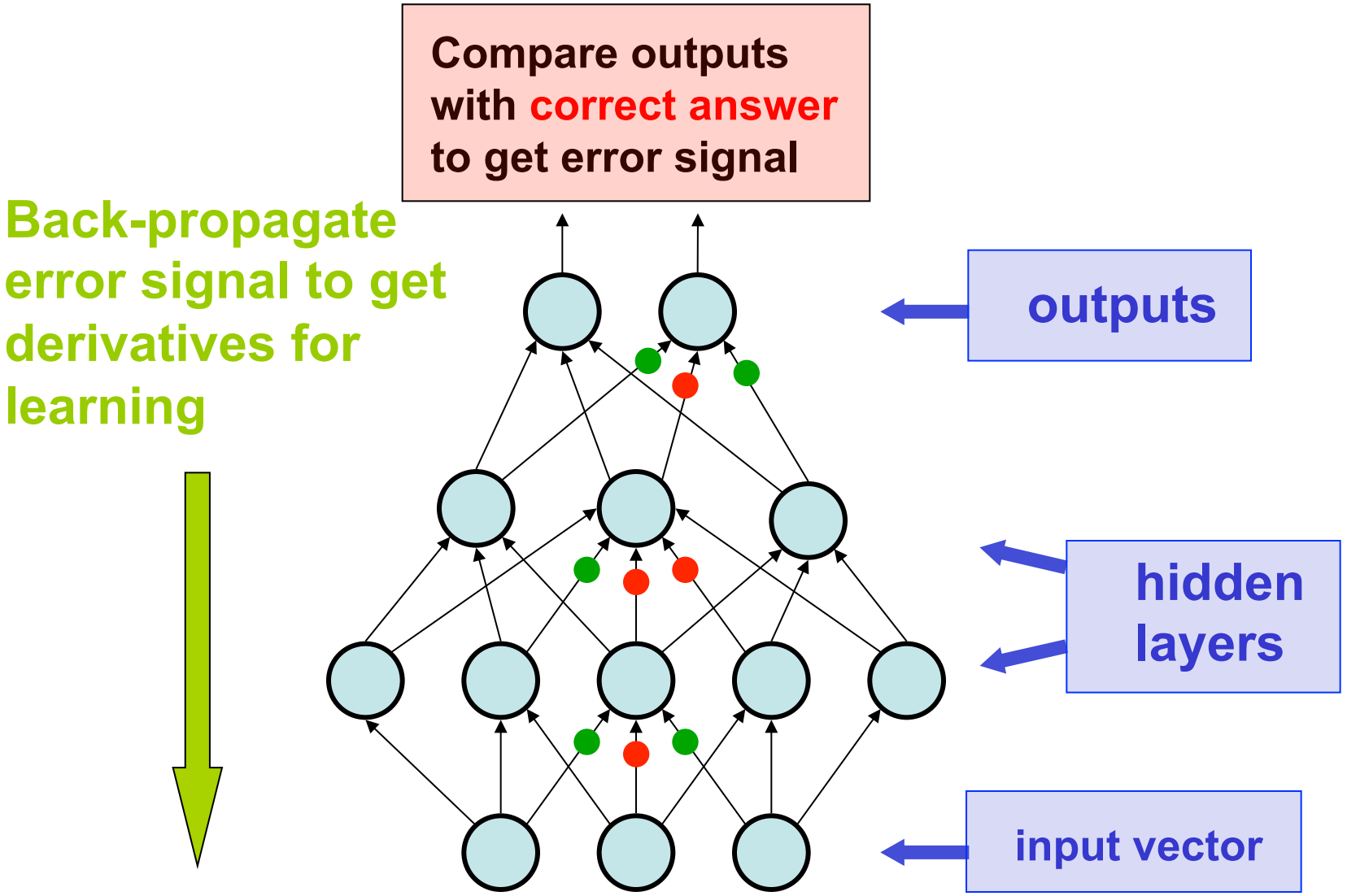
Perceptron, ‘60s



One of the first data-driven models in AI

Slide credits: G. Hinton

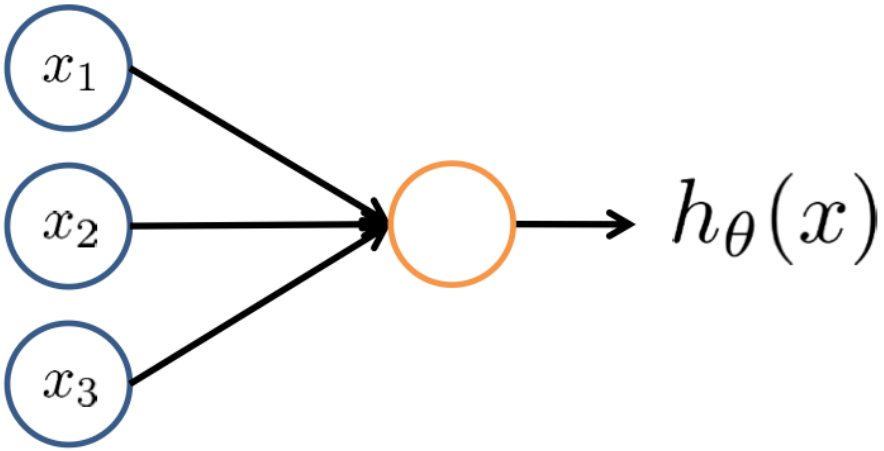
Second generation neural networks (~1985)



Slide credits: G. Hinton

Neuron model: Logistic unit

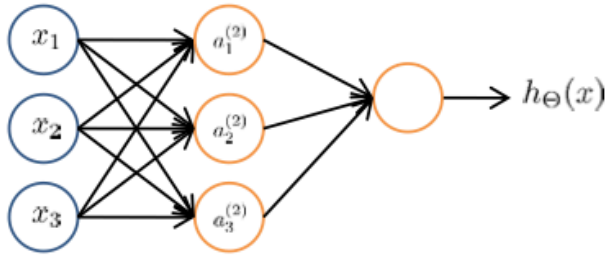
$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$$



Sigmoid (logistic) activation function.

Slide credits: A. Ng

Neural Network



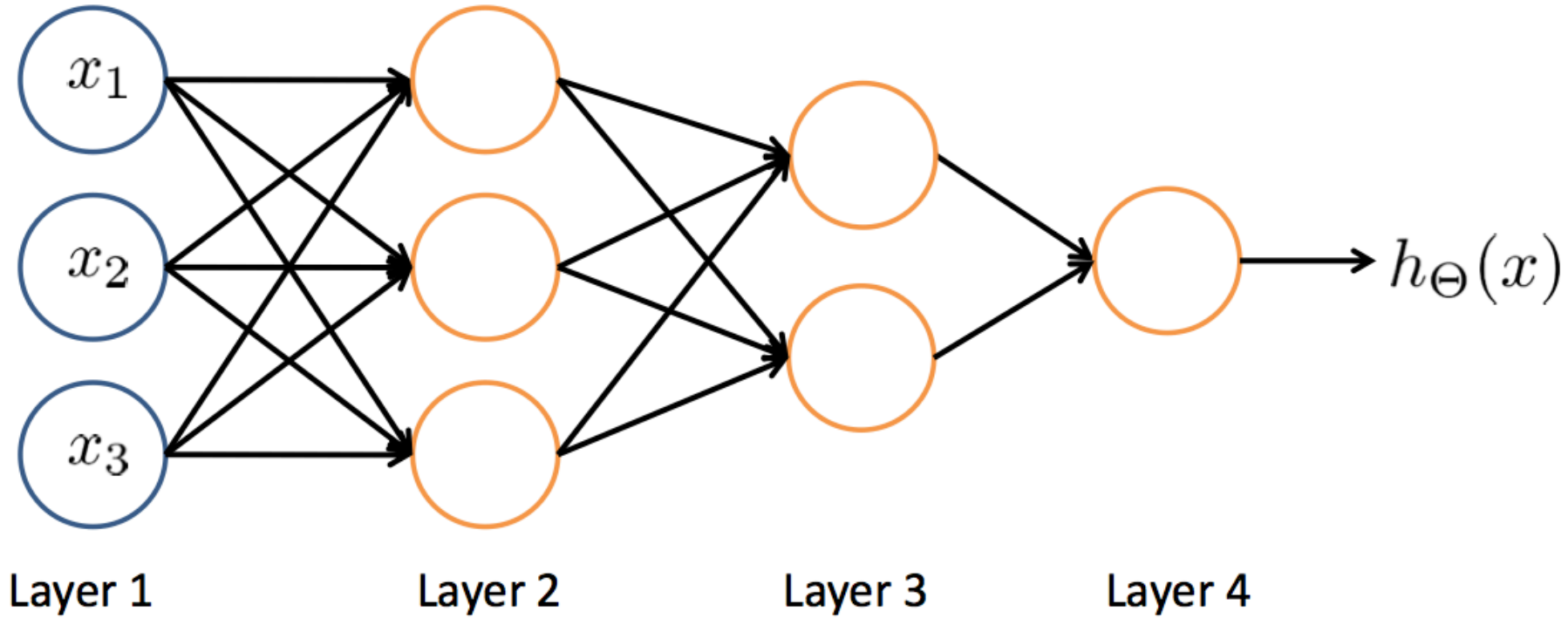
$a_i^{(j)}$ = "activation" of unit i in layer j

$\Theta^{(j)}$ = matrix of weights controlling function mapping from layer j to layer $j+1$

$$a_1^{(2)} = g(\Theta_{10}^{(1)} x_0 + \Theta_{11}^{(1)} x_1 + \Theta_{12}^{(1)} x_2 + \Theta_{13}^{(1)} x_3)$$
$$a_2^{(2)} = g(\Theta_{20}^{(1)} x_0 + \Theta_{21}^{(1)} x_1 + \Theta_{22}^{(1)} x_2 + \Theta_{23}^{(1)} x_3)$$
$$a_3^{(2)} = g(\Theta_{30}^{(1)} x_0 + \Theta_{31}^{(1)} x_1 + \Theta_{32}^{(1)} x_2 + \Theta_{33}^{(1)} x_3)$$
$$h_\Theta(x) = a_1^{(3)} = g(\Theta_{10}^{(2)} a_0^{(2)} + \Theta_{11}^{(2)} a_1^{(2)} + \Theta_{12}^{(2)} a_2^{(2)} + \Theta_{13}^{(2)} a_3^{(2)})$$

If network has s_j units in layer j, s_{j+1} units in layer $j+1$, then $\Theta^{(j)}$ will be of dimension $s_{j+1} \times (s_j + 1)$.

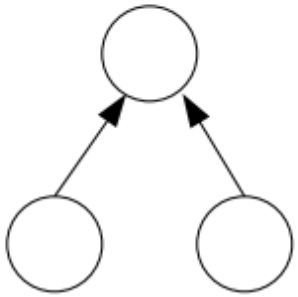
Slide credits: A. Ng

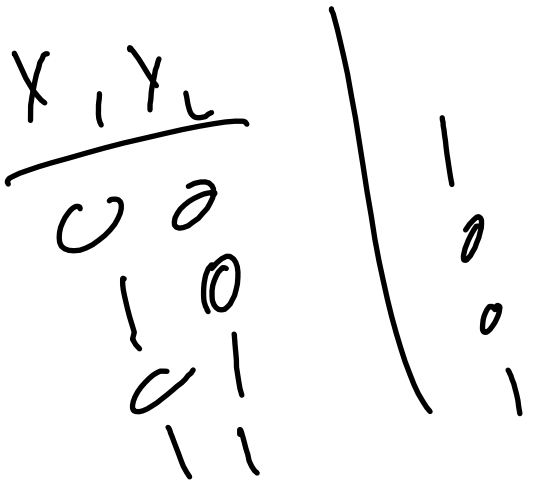


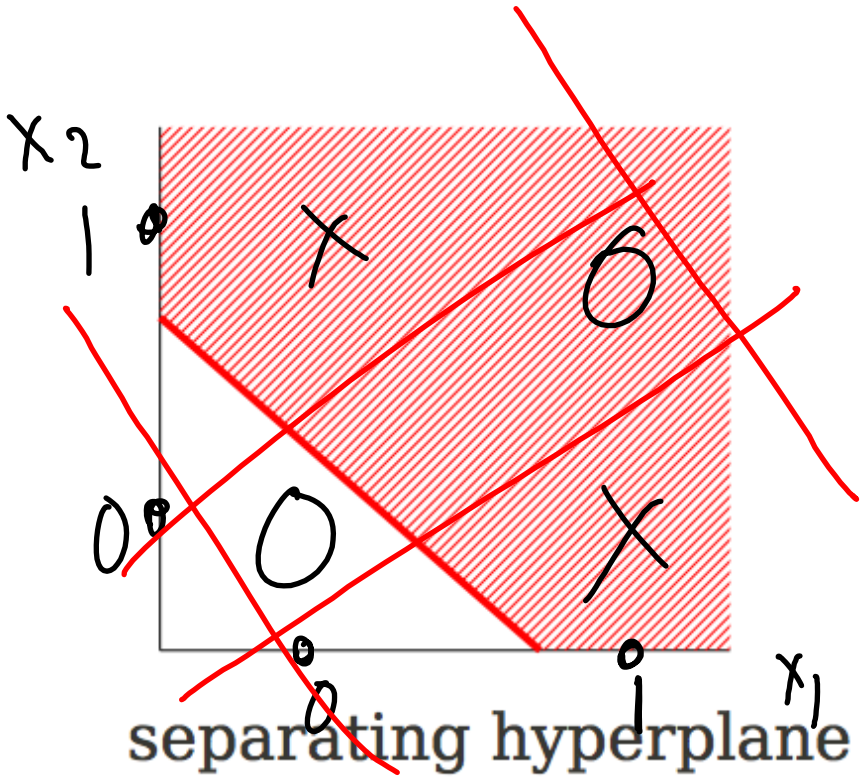
Slide credits: A. Ng

From flat to deep

1 layer of trainable weights



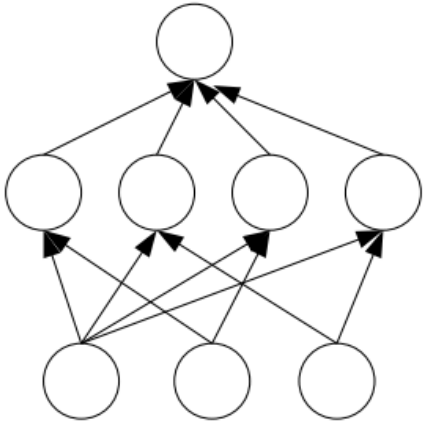


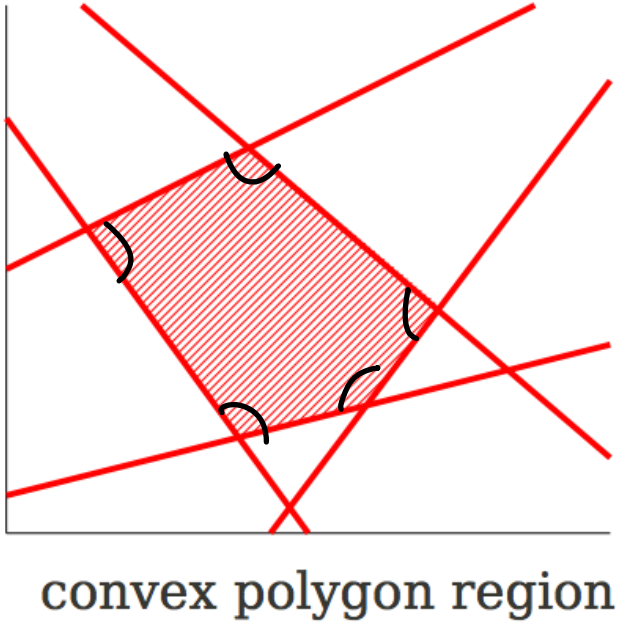


REctified Linear Units (RELUs) $\boldsymbol{h}^1 = max(0, W^1 \boldsymbol{x} + \boldsymbol{b}^1)$

From flat to deep

2 layers of trainable weights

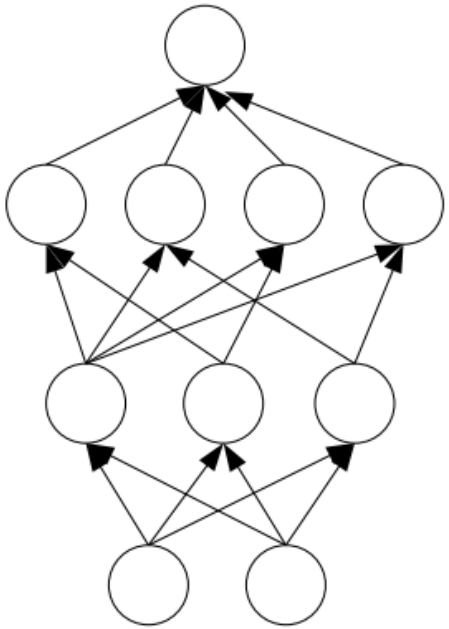


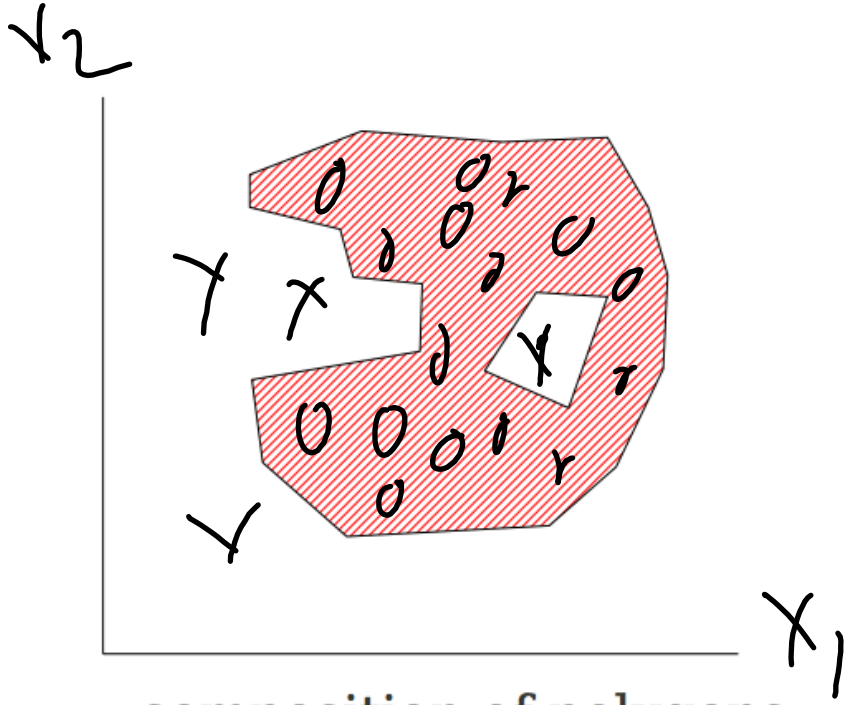


convex polygon region

From flat to deep

3 layers of trainable weights

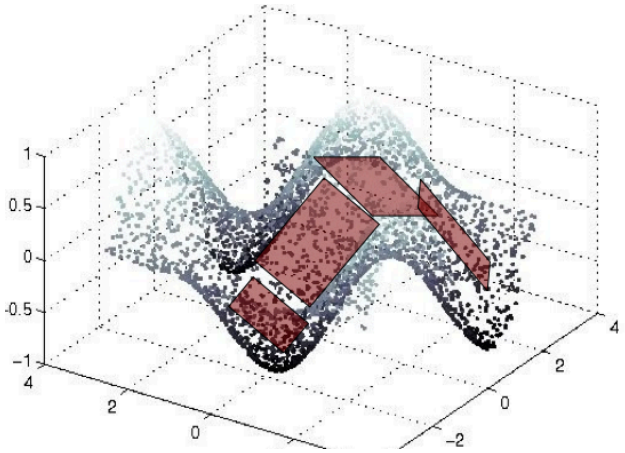


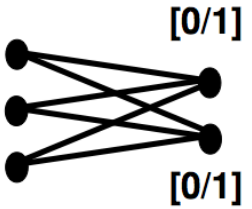


composition of polygons: convex regions

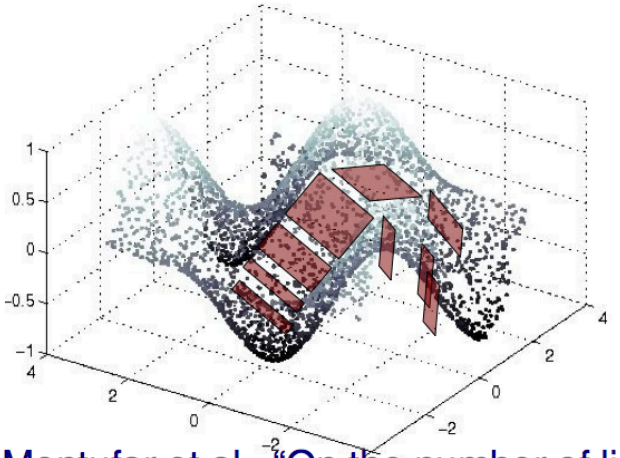
REctified Linear Units (RELUs)

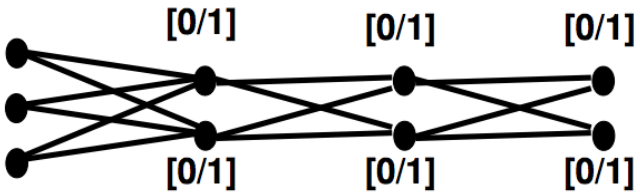
$$h^1 = max(0, W^1 x + b^1)$$





ReLU layers do local linear approximation. Number of planes grows exponentially with number of hidden units. Multiple layers yeild exponential savings in number of parameters (parameter sharing).



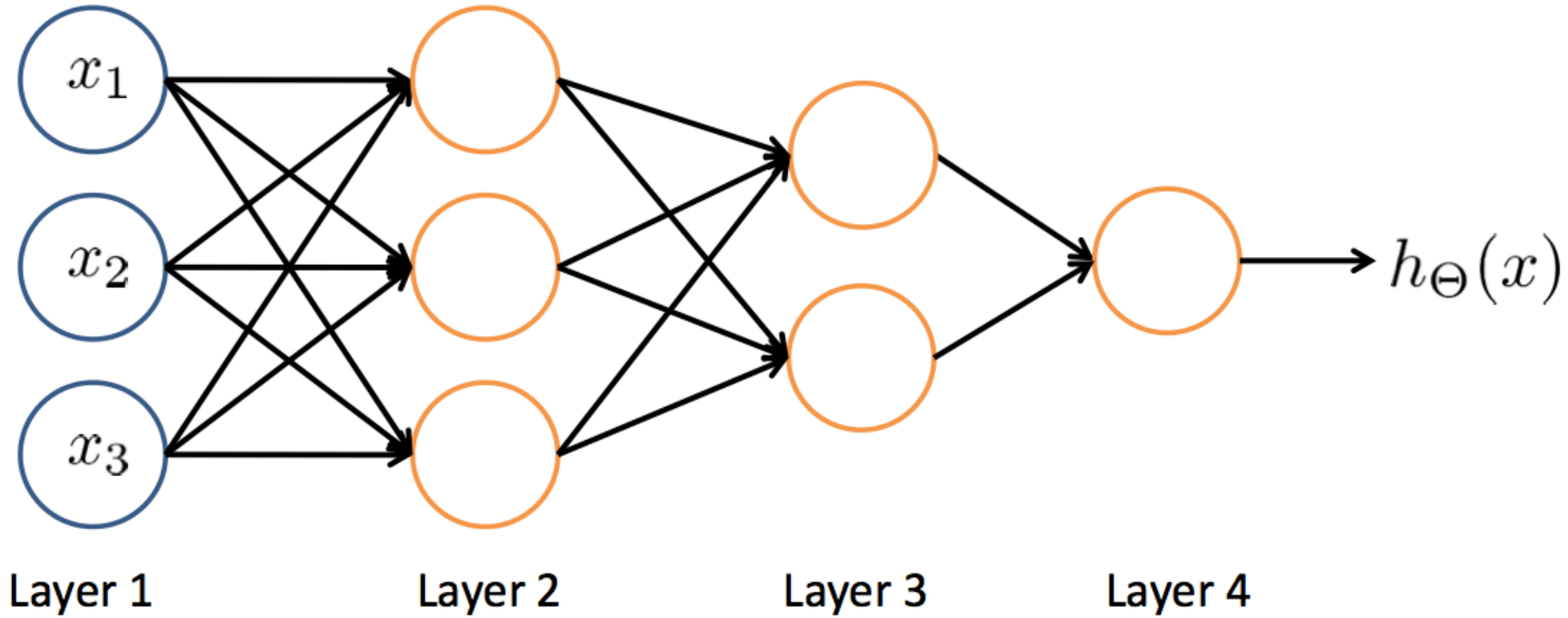


Montufar et al. "On the number of linear regions of DNNs" arXiv 2014

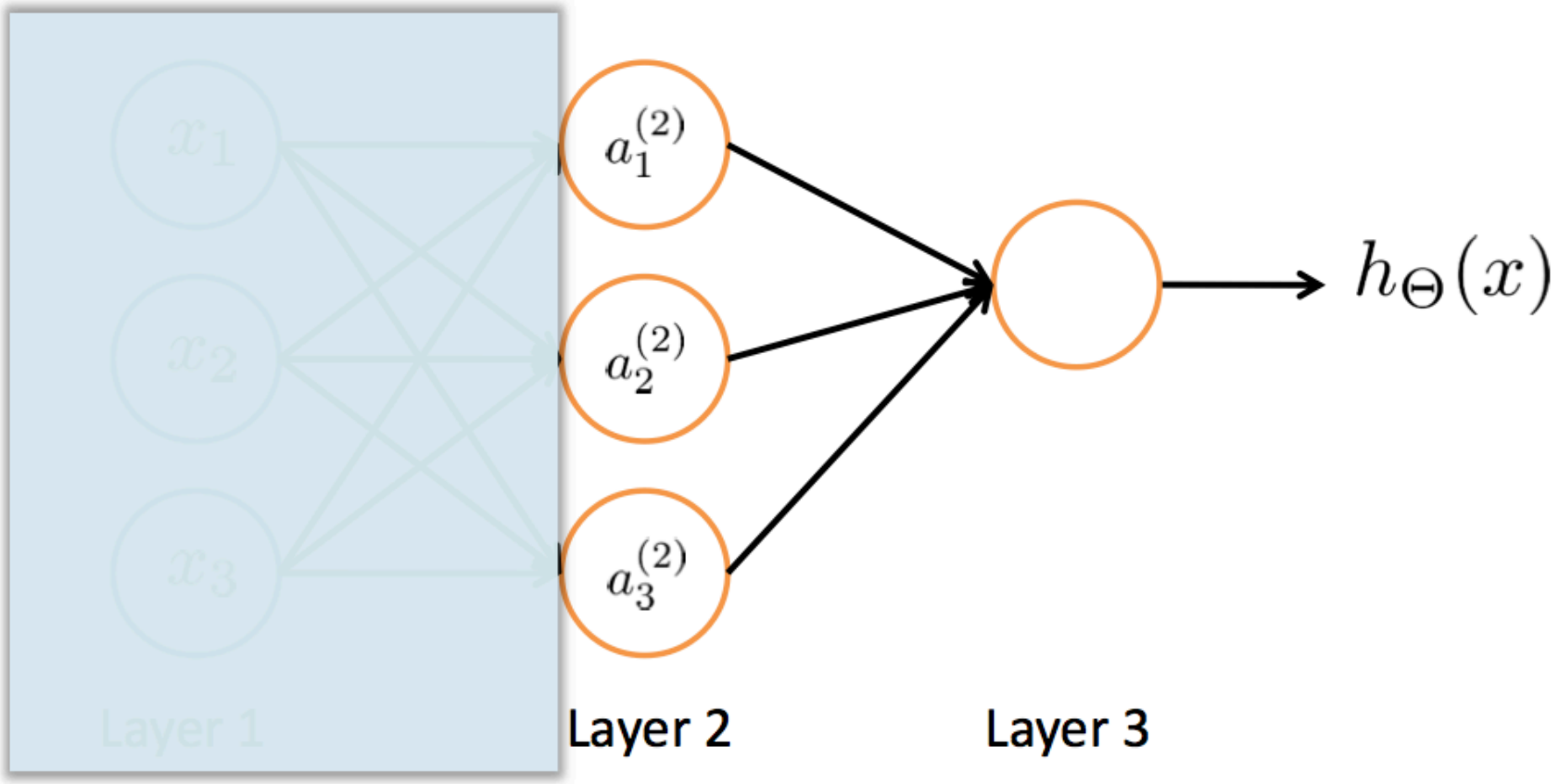
23

Ranzato

Slide credits: M. A. Ranzatto



Slide credits: A. Ng



Learned features

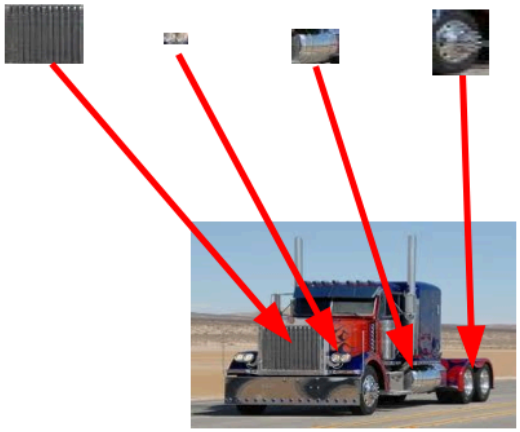
Slide credits: A. Ng

Features, from flat to deep

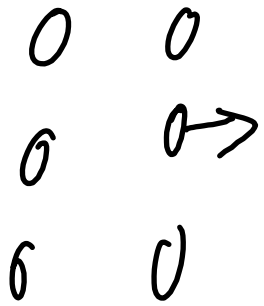
Question: Why do we need many layers?

Answer: When input has hierarchical structure, the use of a hierarchical architecture is potentially more efficient because intermediate computations can be re-used. DL architectures are efficient also because they use **distributed representations** which are shared across classes.

[0 0 1 0 0 0 0 1 0 0 1 1 0 0 1 0 …] truck feature



Exponentially more efficient than a 1-of-N representation (a la k-means)



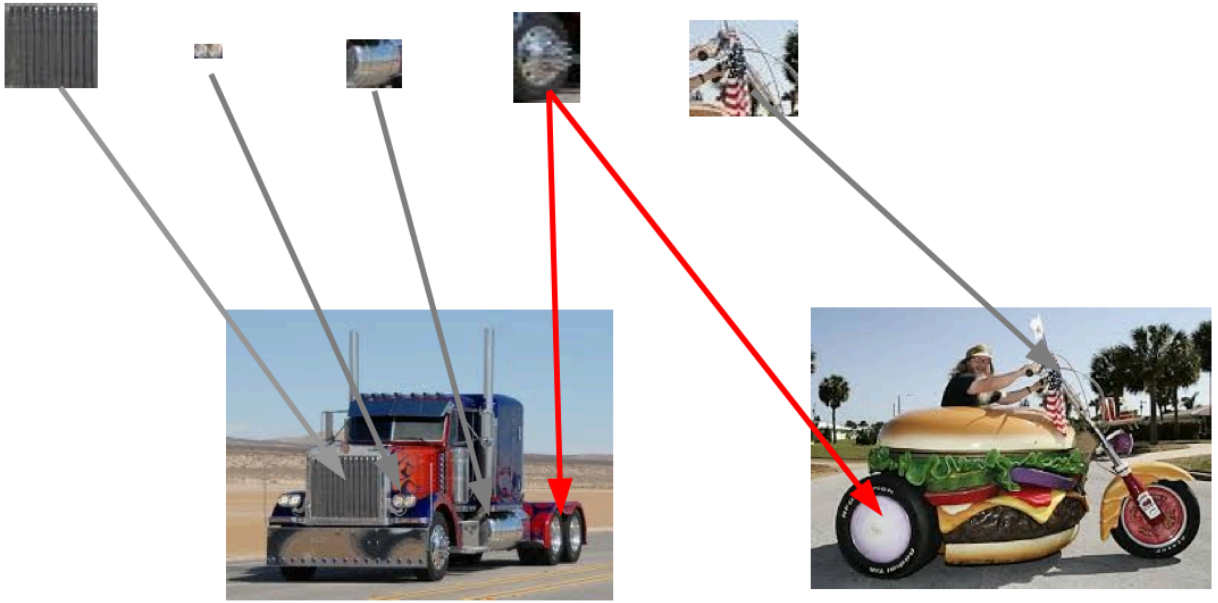
Ranzato

Features, from flat to deep

Interpretation

[1 1 0 0 0 1 0 **1** 0 0 0 0 1 1 0 1…] motorbike

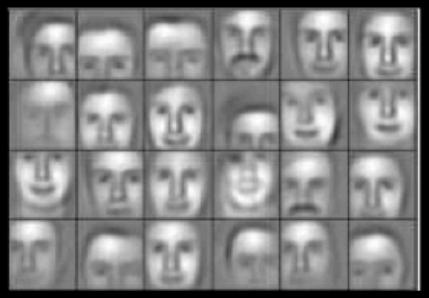
[0 0 **1** 0 0 0 0 **1** 0 0 1 1 0 0 1 0 …] truck



Ranzato

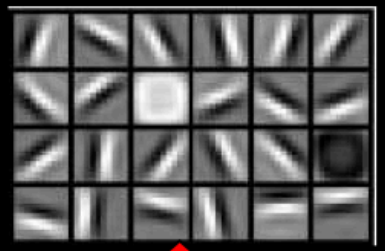
Hierarchical Sparse coding (Sparse DBN): Trained on face images

Training set: Aligned images of faces.



object models

object parts (combination of edges)



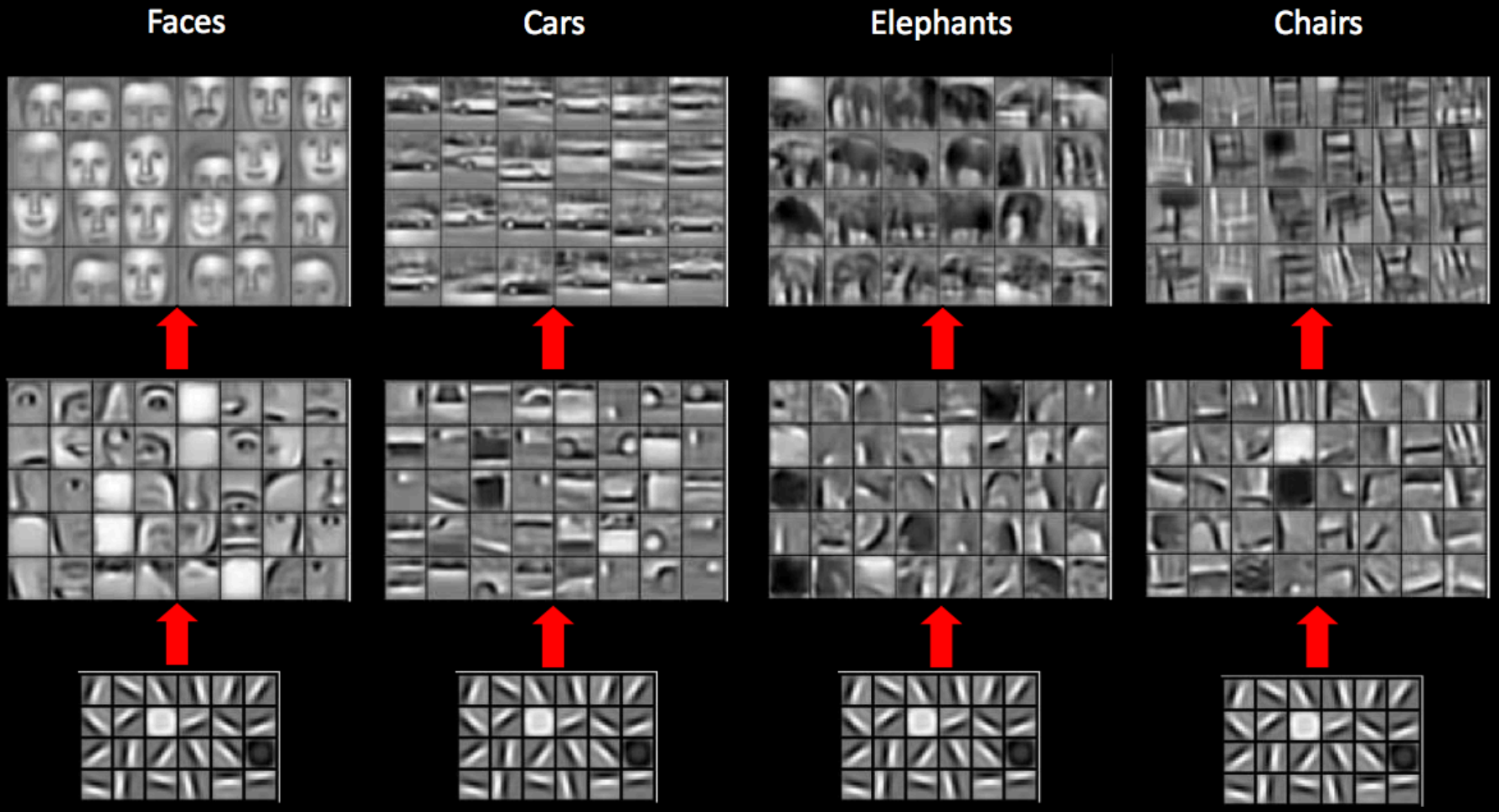
edges

pixels

[Honglak Lee]

Hierarchical Sparse coding (Sparse DBN)

Features learned from training on different object classes.



[Honglak Lee]

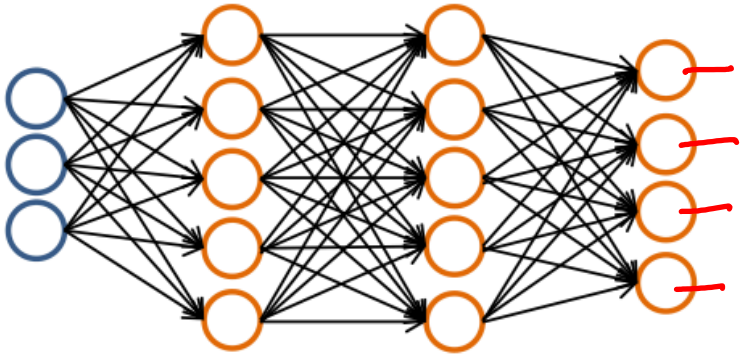
Multiple output units: One-vs-all.

Pedestrian

Car

Motorcycle

Truck



$h_\Theta(x) \in \mathbb{R}^4$

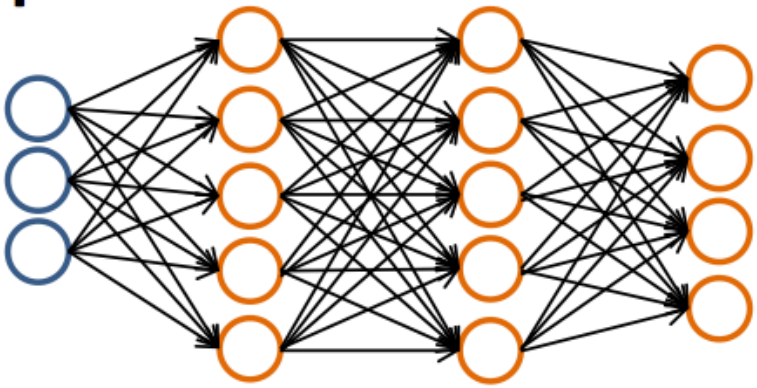
one-hot

Want $h_\Theta(x) \approx \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$, $h_\Theta(x) \approx \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$, $h_\Theta(x) \approx \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$, etc.

when pedestrian when car when motorcycle

Slide credits: A. Ng

Multiple output units: One-vs-all.



$h_\Theta(x) \in \mathbb{R}^4$

Want $h_\Theta(x) \approx \begin{bmatrix}1\\0\\0\\0\end{bmatrix}$, $h_\Theta(x) \approx \begin{bmatrix}0\\1\\0\\0\end{bmatrix}$, $h_\Theta(x) \approx \begin{bmatrix}0\\0\\1\\0\end{bmatrix}$, etc.

when pedestrian when car when motorcycle

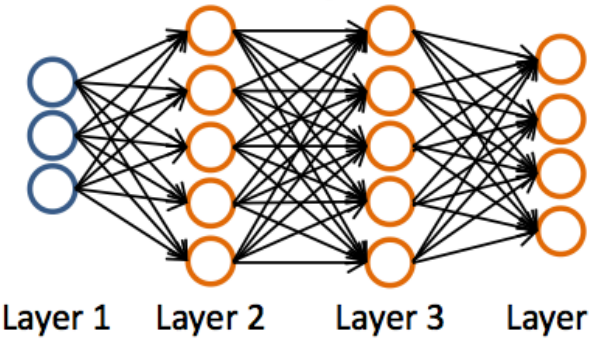
Training set: $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$

$y^{(i)}$ one of $\begin{bmatrix}1\\0\\0\\0\end{bmatrix}$, $\begin{bmatrix}0\\1\\0\\0\end{bmatrix}$, $\begin{bmatrix}0\\0\\1\\0\end{bmatrix}$, $\begin{bmatrix}0\\0\\0\\1\end{bmatrix}$

pedestrian car motorcycle truck

Slide credits: A. Ng

Neural Network (Classification)



$\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$

$L =$ total no. of layers in network

$s_l =$ no. of units (not counting bias unit) in layer l

Binary classification

$y = 0$ or 1

1 output unit

Multi-class classification (K classes)

$y \in \mathbb{R}^K$ E.g. $\begin{bmatrix}1\\0\\0\\0\end{bmatrix}$, $\begin{bmatrix}0\\1\\0\\0\end{bmatrix}$, $\begin{bmatrix}0\\0\\1\\0\end{bmatrix}$, $\begin{bmatrix}0\\0\\0\\1\end{bmatrix}$

pedestrian car motorcycle truck

K output units

Slide credits: A. Ng

Cost function

SoftMax

Logistic regression:

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n} \theta_j^2$$

Neural network:

$$h_\Theta(x) \in \mathbb{R}^K \quad (h_\Theta(x))_i = i^{th} \text{ output}$$

$$J(\Theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log(h_\Theta(x^{(i)}))_k + (1 - y_k^{(i)}) \log(1 - (h_\Theta(x^{(i)}))_k)\right]$$

$$+\frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}(\Theta_{ji}^{(l)})^2$$

Slide credits: A. Ng

Gradient computation

$$J(\Theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log h_\theta(x^{(i)})_k + (1 - y_k^{(i)}) \log(1 - h_\theta(x^{(i)})_k)\right]$$
$$+\frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}(\Theta_j^{(l)})^2$$

$$\min_{\Theta} J(\Theta)$$

Need code to compute:

- $J(\Theta)$
- $\frac{\partial}{\partial \Theta_{ij}^{(l)}} J(\Theta)$

Slide credits: A. Ng

Backpropagation algorithm

Training set $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$

Set $\triangle_{ij}^{(l)} = 0$ (for all l, i, j).

For $i = 1$ to m

Set $a^{(1)} = x^{(i)}$

Perform forward propagation to compute $a^{(l)}$ for $l = 2, 3, \ldots, L$

Using $y^{(i)}$, compute $\delta^{(L)} = a^{(L)} - y^{(i)}$

Compute $\delta^{(L-1)}, \delta^{(L-2)}, \ldots, \delta^{(2)}$

$\triangle_{ij}^{(l)} := \triangle_{ij}^{(l)} + a_j^{(l)} \delta_i^{(l+1)}$

$$D_{ij}^{(l)} := \frac{1}{m} \triangle_{ij}^{(l)} + \lambda \Theta_{ij}^{(l)} \text{ if } j \neq 0$$

$$D_{ij}^{(l)} := \frac{1}{m} \triangle_{ij}^{(l)} \text{ if } j = 0$$

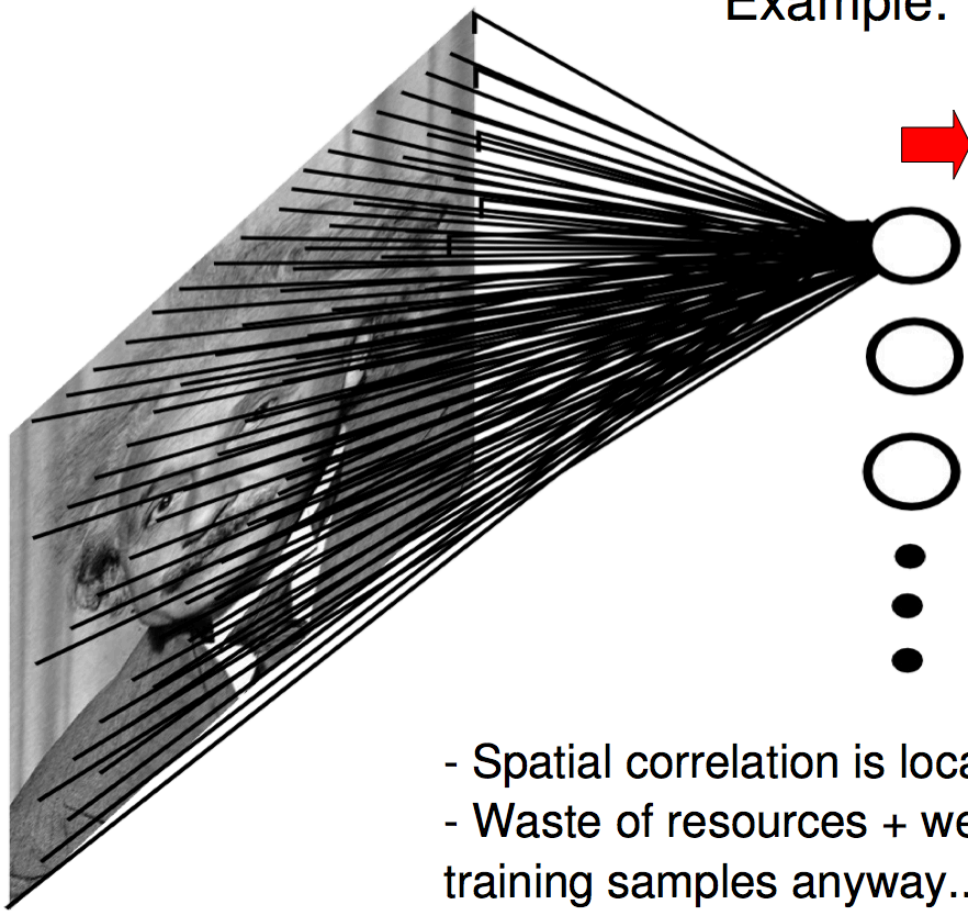
$$\frac{\partial}{\partial \Theta_{ij}^{(l)}} J(\Theta) = D_{ij}^{(l)}$$

stocastic gradient Descent

Slide credits: A. Ng

Network connectivity

Fully Connected Layer



Example: 200x200 image
40K hidden units
~2B parameters!!!

- Spatial correlation is local
- Waste of resources + we have not enough training samples anyway..

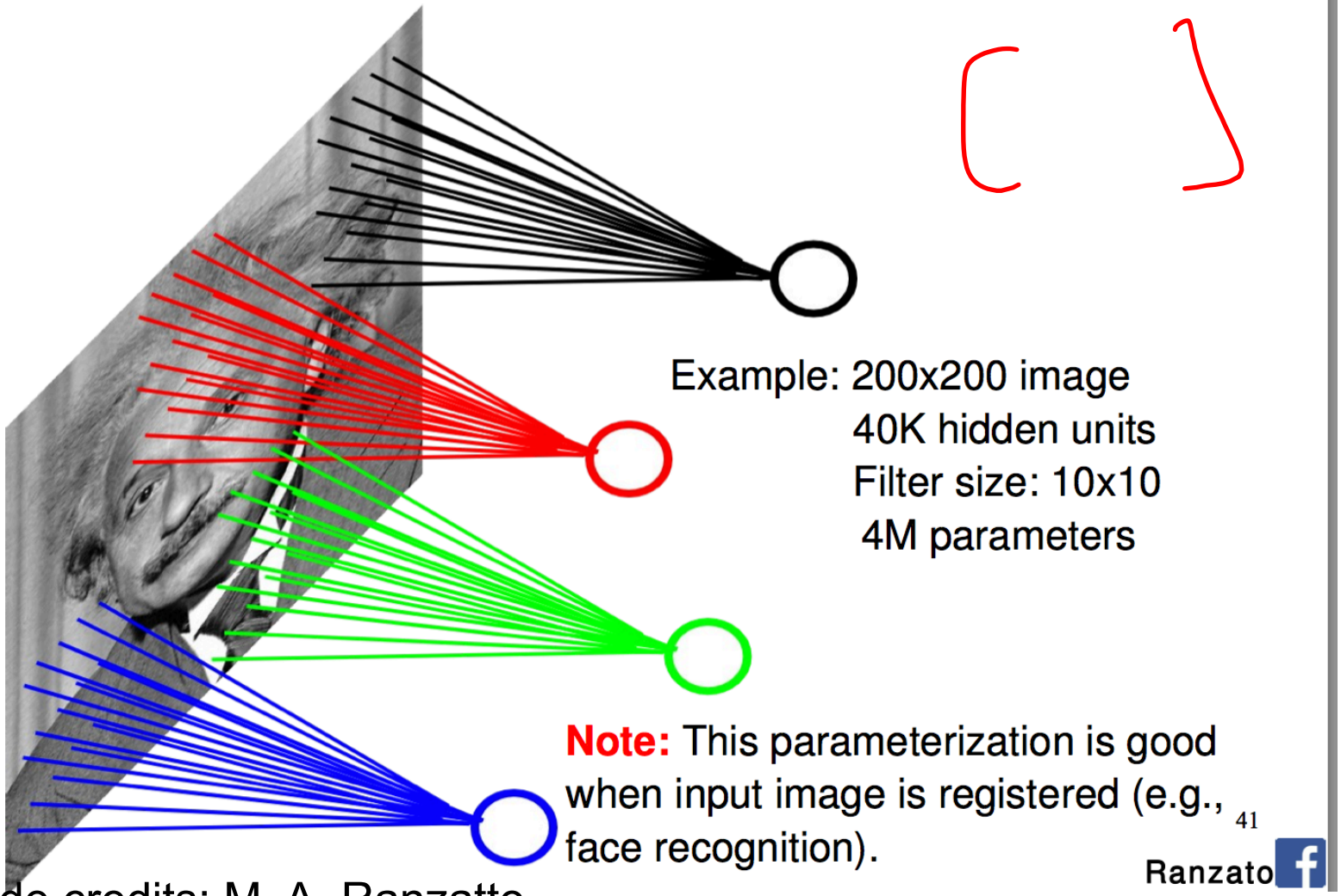
40
Ranzato

Slide credits: M. A. Ranzatto

Network connectivity

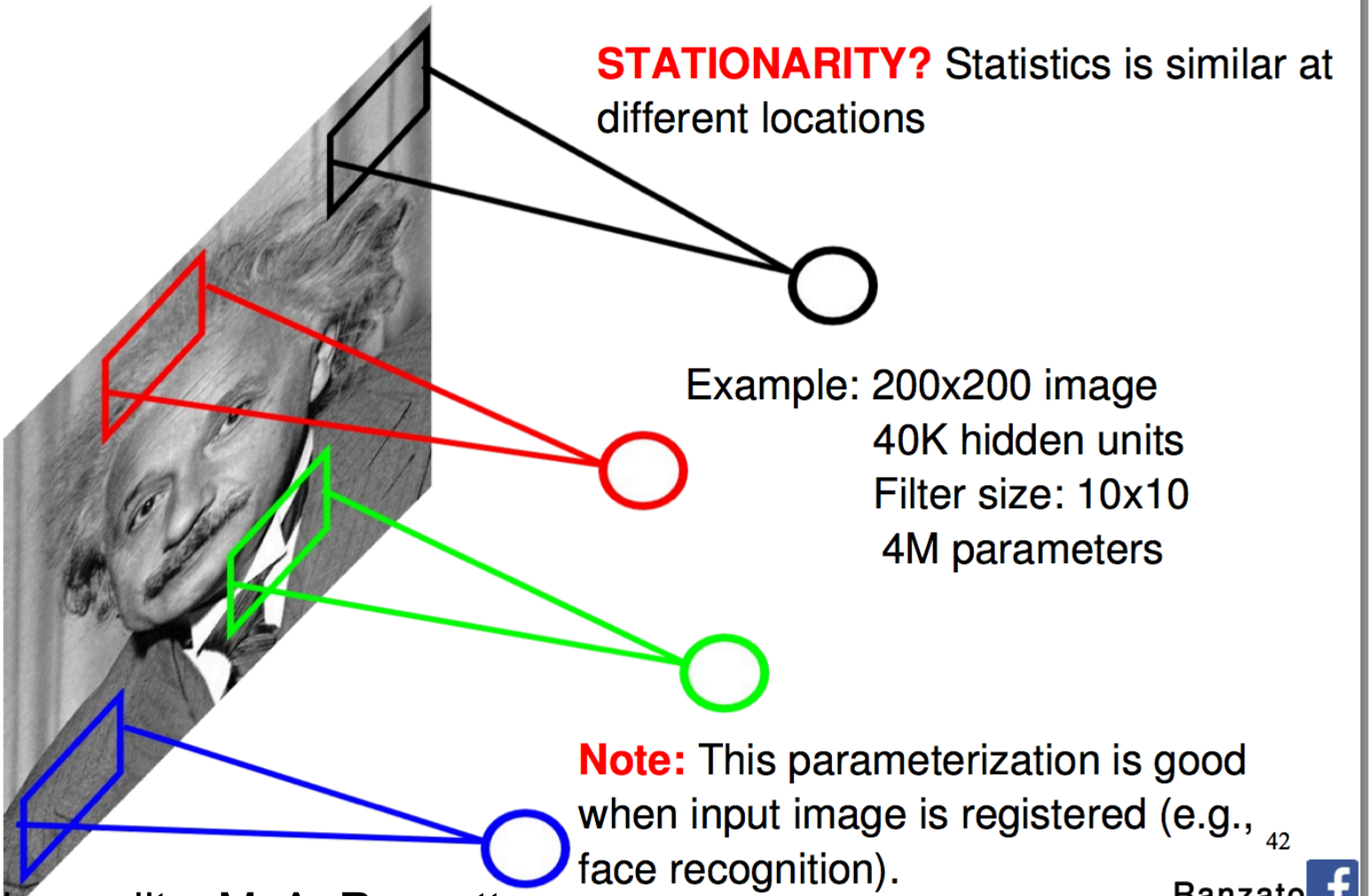
Convolutional Neural Networks CNN

Locally Connected Layer



Slide credits: M. A. Ranzatto

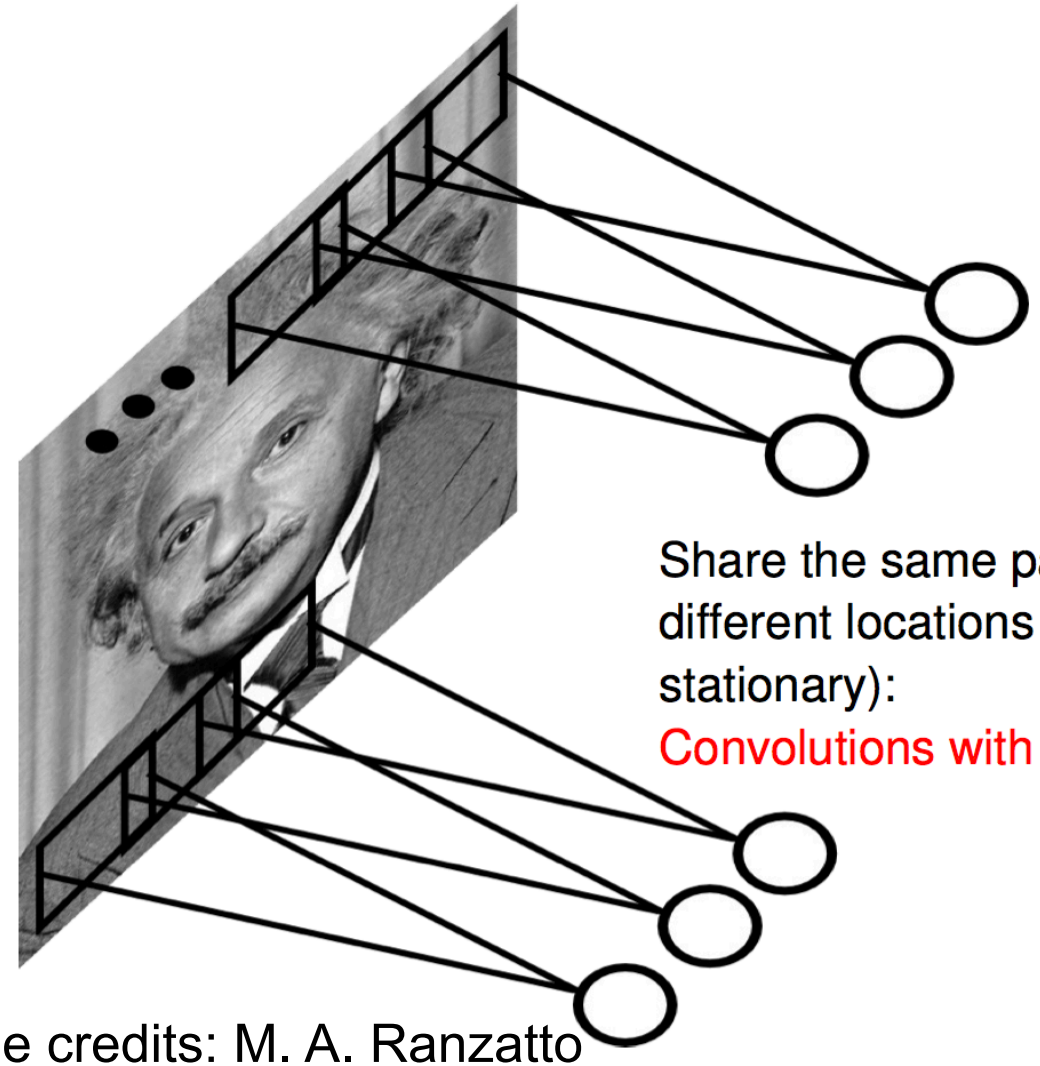
Network connectivity



Slide credits: M. A. Ranzatto

Network connectivity

Convolutional Layer



Share the same parameters across different locations (assuming input is stationary):
Convolutions with learned kernels

43

Ranzato

Slide credits: M. A. Ranzatto

Network connectivity

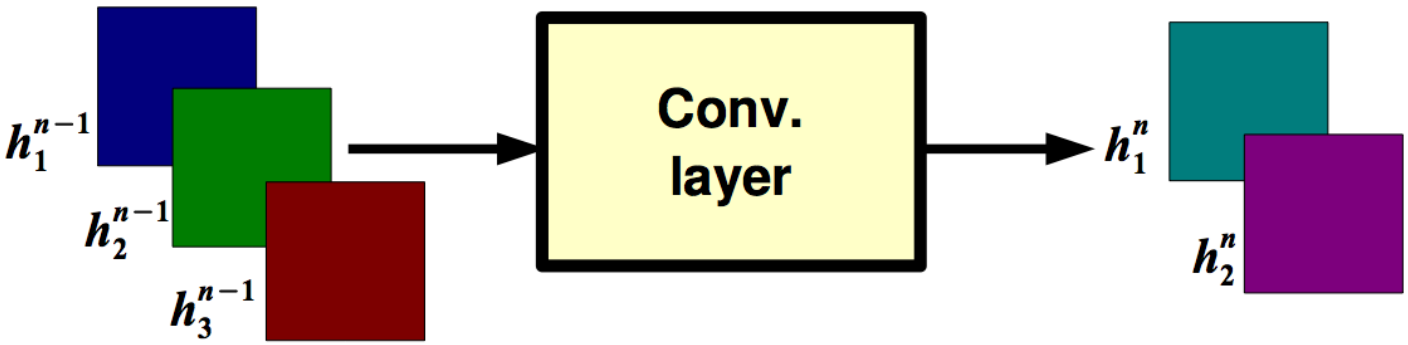
Convolutional Layer

$$h_j^n = max\left(0, \sum_{k=1}^{K} h_k^{n-1} * w_{kj}^n\right)$$

output feature map

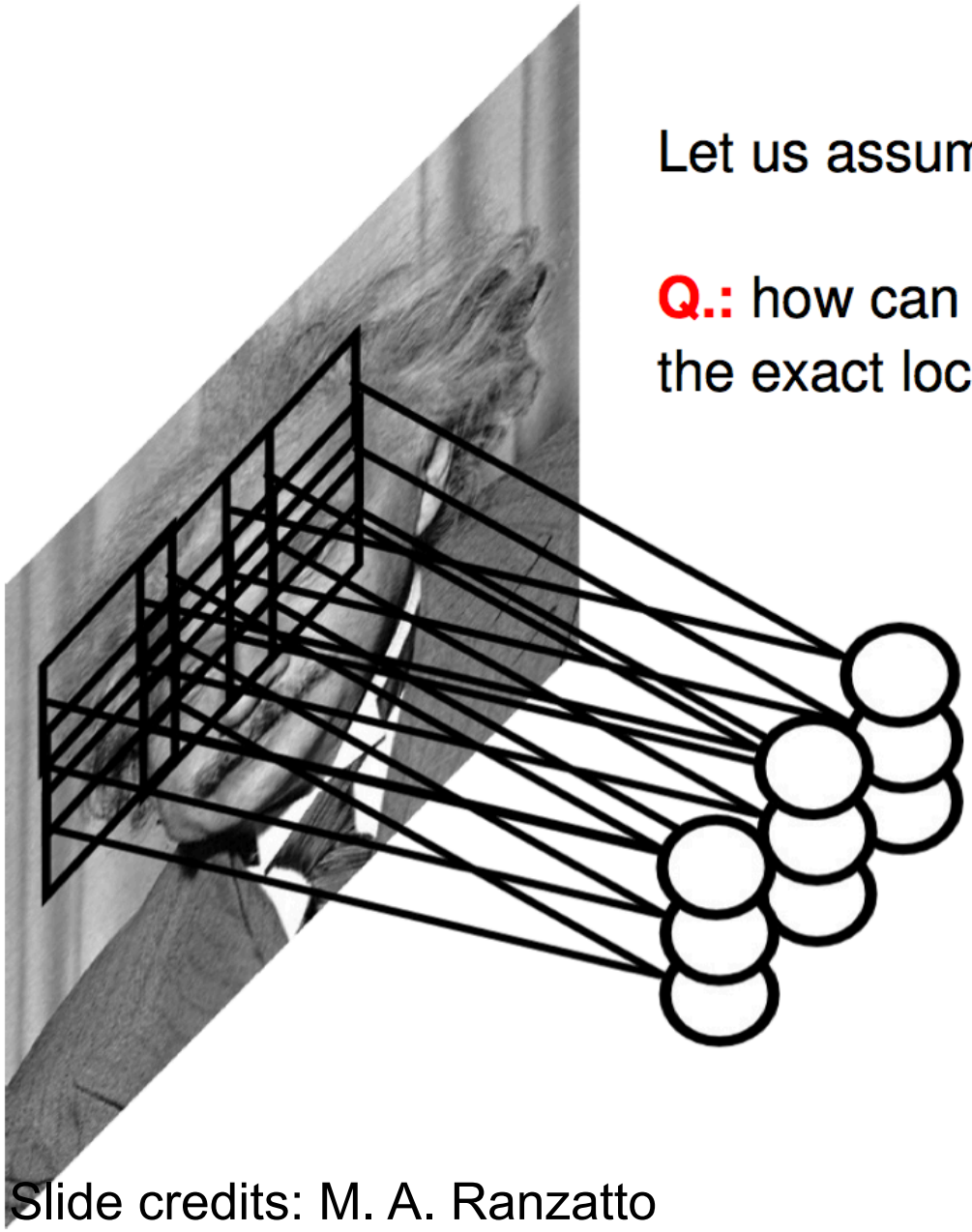
input feature map

kernel



Slide credits: M. A. Ranzatto

Pooling Layer

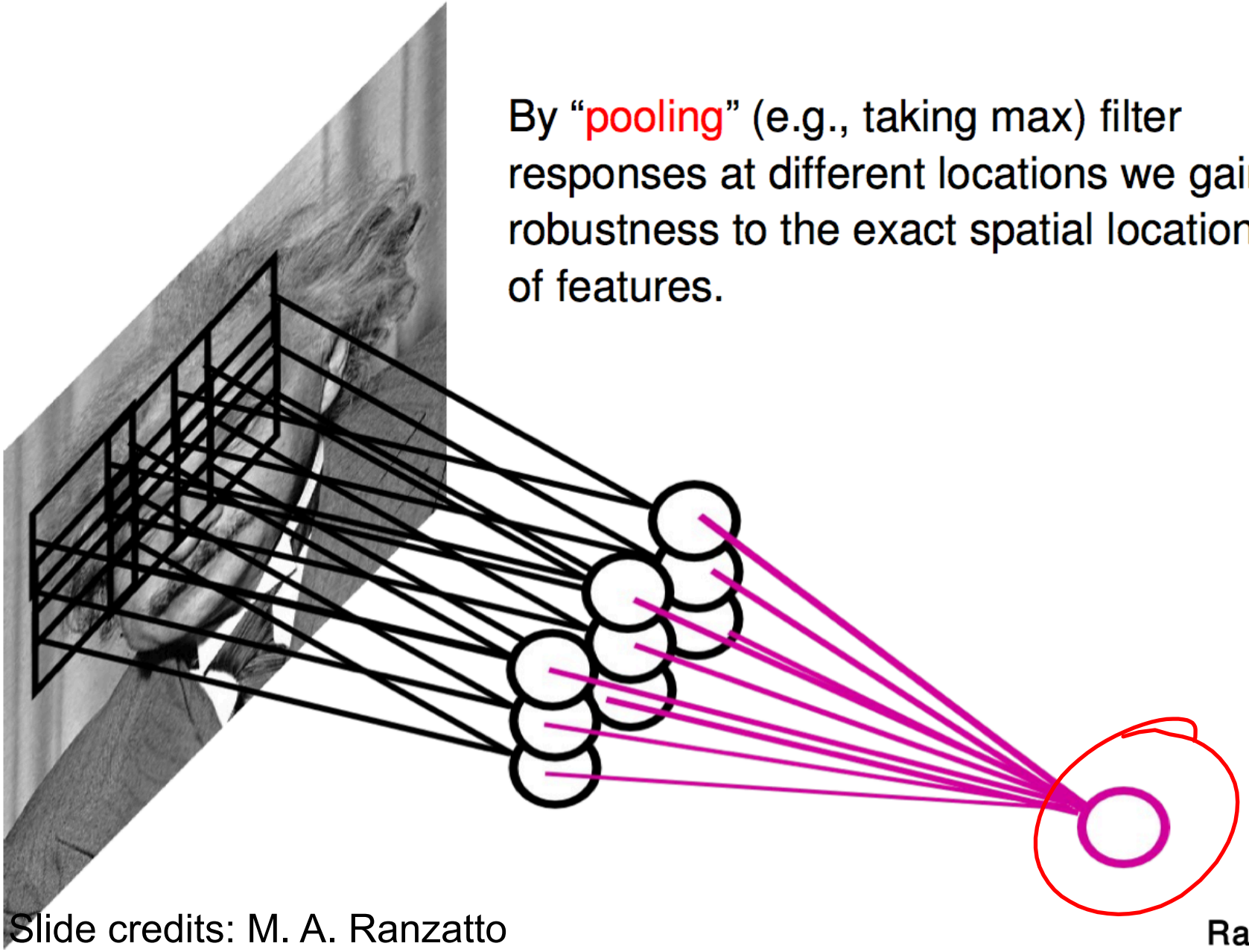


Let us assume filter is an “eye” detector.

Q.: how can we make the detection robust to the exact location of the eye?

Slide credits: M. A. Ranzatto

Pooling Layer



By "pooling" (e.g., taking max) filter responses at different locations we gain robustness to the exact spatial location of features.

Slide credits: M. A. Ranzatto

Pooling Layer: Examples

Max-pooling:

$$h_j^n(x,y) = max_{\bar{x}\in N(x),\, \bar{y}\in N(y)} h_j^{n-1}(\bar{x},\bar{y})$$

Average-pooling:

$$h_j^n(x,y) = 1/K \sum_{\bar{x}\in N(x),\, \bar{y}\in N(y)} h_j^{n-1}(\bar{x},\bar{y})$$

L2-pooling:

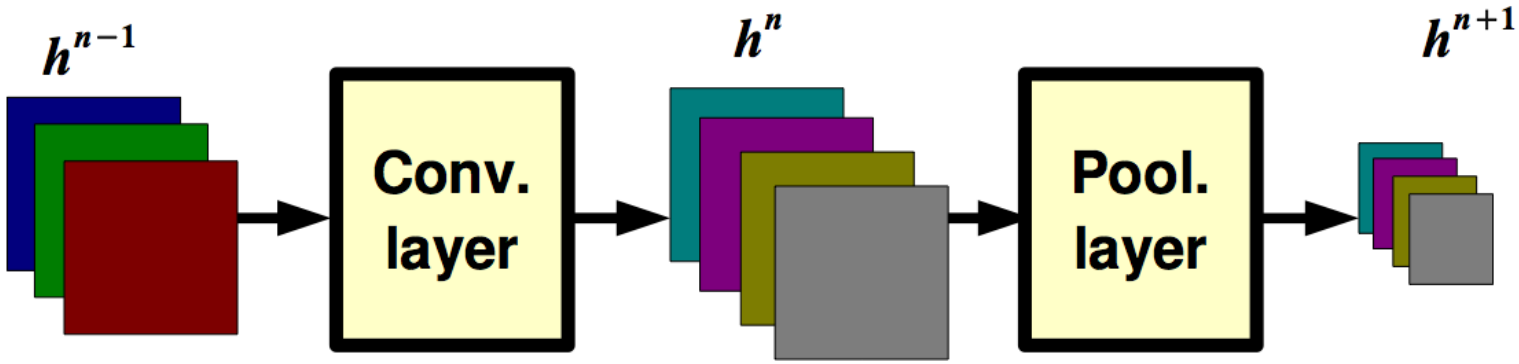
$$h_j^n(x,y) = \sqrt{\sum_{\bar{x}\in N(x),\, \bar{y}\in N(y)} h_j^{n-1}(\bar{x},\bar{y})^2}$$

L2-pooling over features:

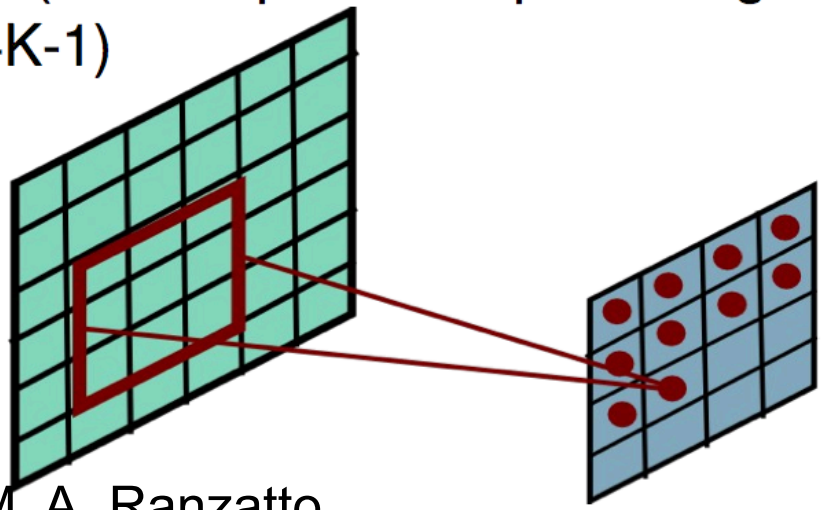
$$h_j^n(x,y) = \sqrt{\sum_{k\in N(j)} h_k^{n-1}(x,y)^2}$$

Slide credits: M. A. Ranzato

Pooling Layer: Receptive Field Size



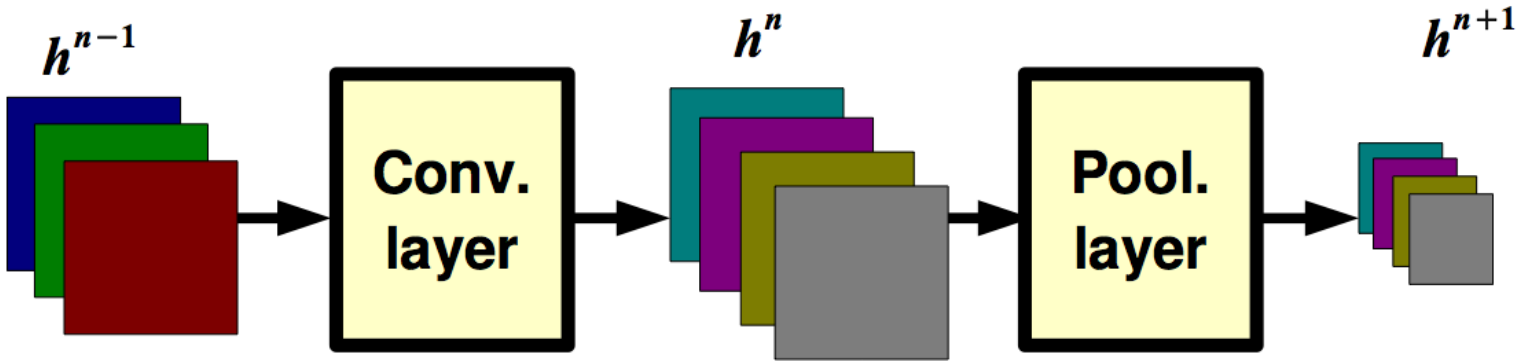
If convolutional filters have size KxK and stride 1, and pooling layer has pools of size PxP, then each unit in the pooling layer depends upon a patch (at the input of the preceding conv. layer) of size: (P+K-1)x(P+K-1)



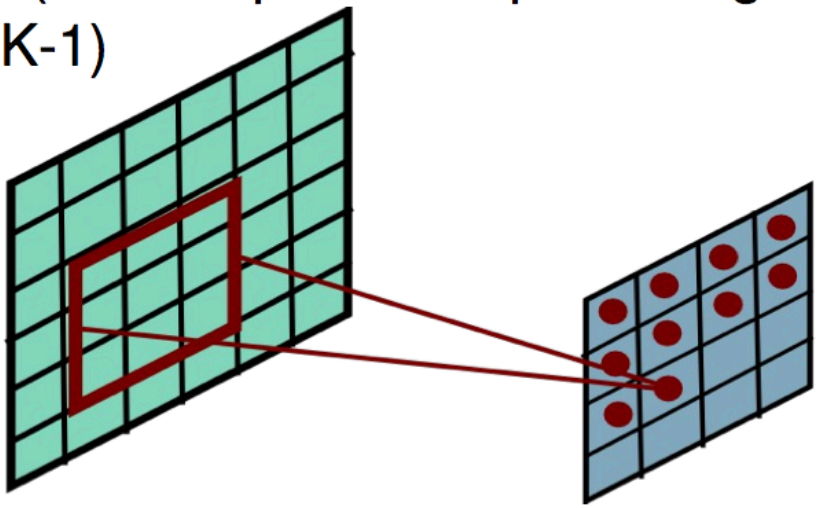
Slide credits: M. A. Ranzatto

Ranzato

Pooling Layer: Receptive Field Size

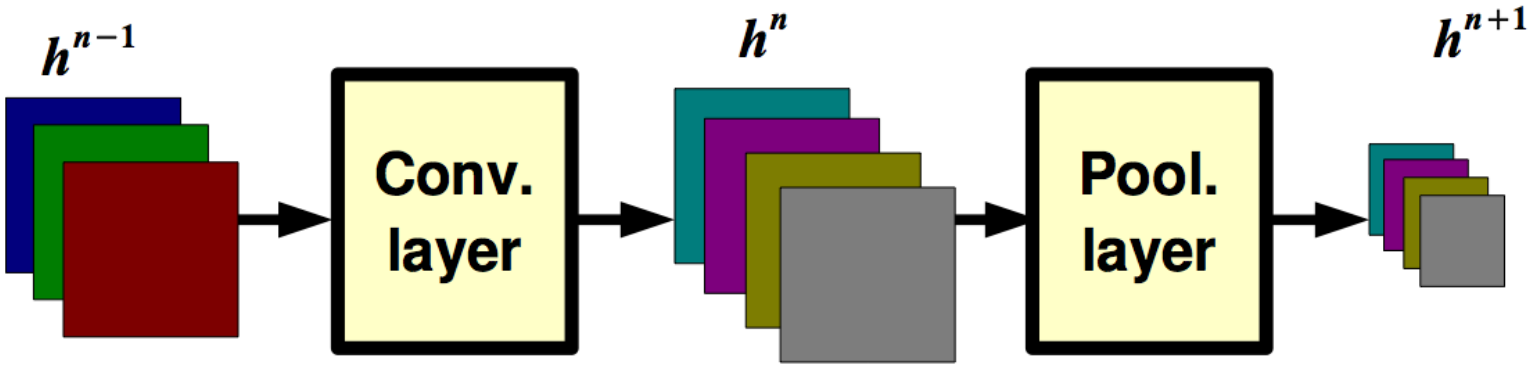


If convolutional filters have size KxK and stride 1, and pooling layer has pools of size PxP, then each unit in the pooling layer depends upon a patch (at the input of the preceding conv. layer) of size: (P+K-1)x(P+K-1)

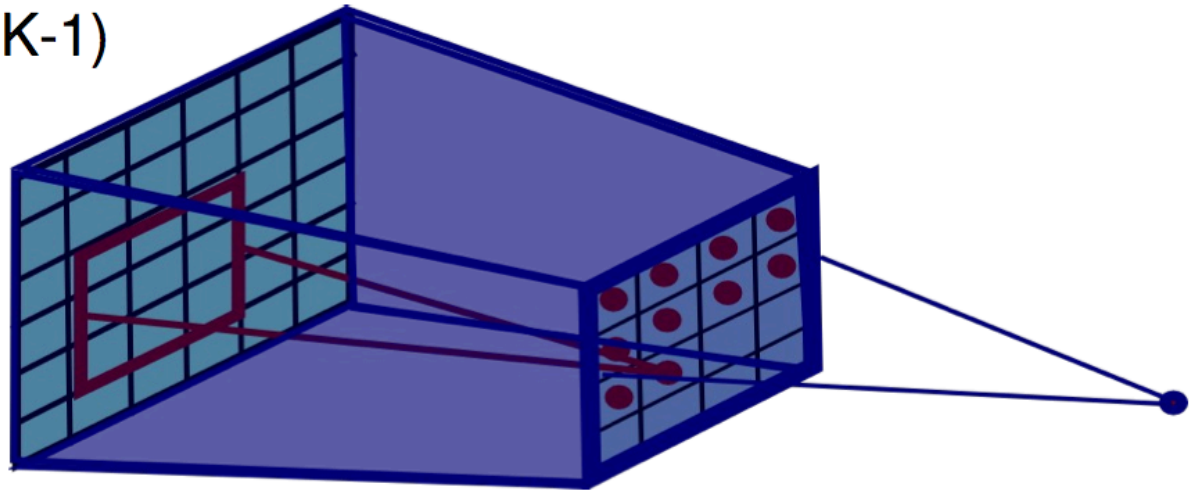


Ranzato

Pooling Layer: Receptive Field Size

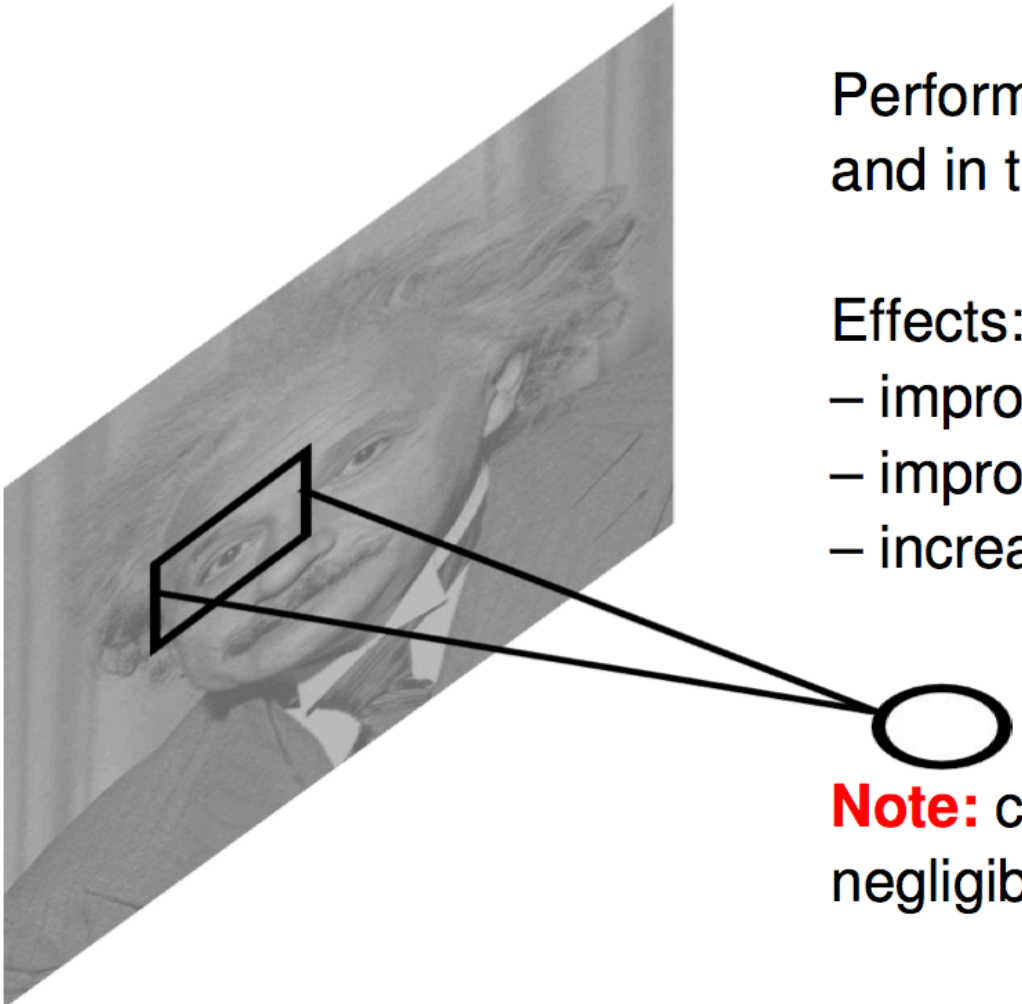


If convolutional filters have size KxK and stride 1, and pooling layer has pools of size PxP, then each unit in the pooling layer depends upon a patch (at the input of the preceding conv. layer) of size: (P+K-1)x(P+K-1)



Local Contrast Normalization

$$h^{i+1}(x,y) = \frac{h^i(x,y) - m^i(N(x,y))}{max(\epsilon, \sigma^i(N(x,y)))}$$



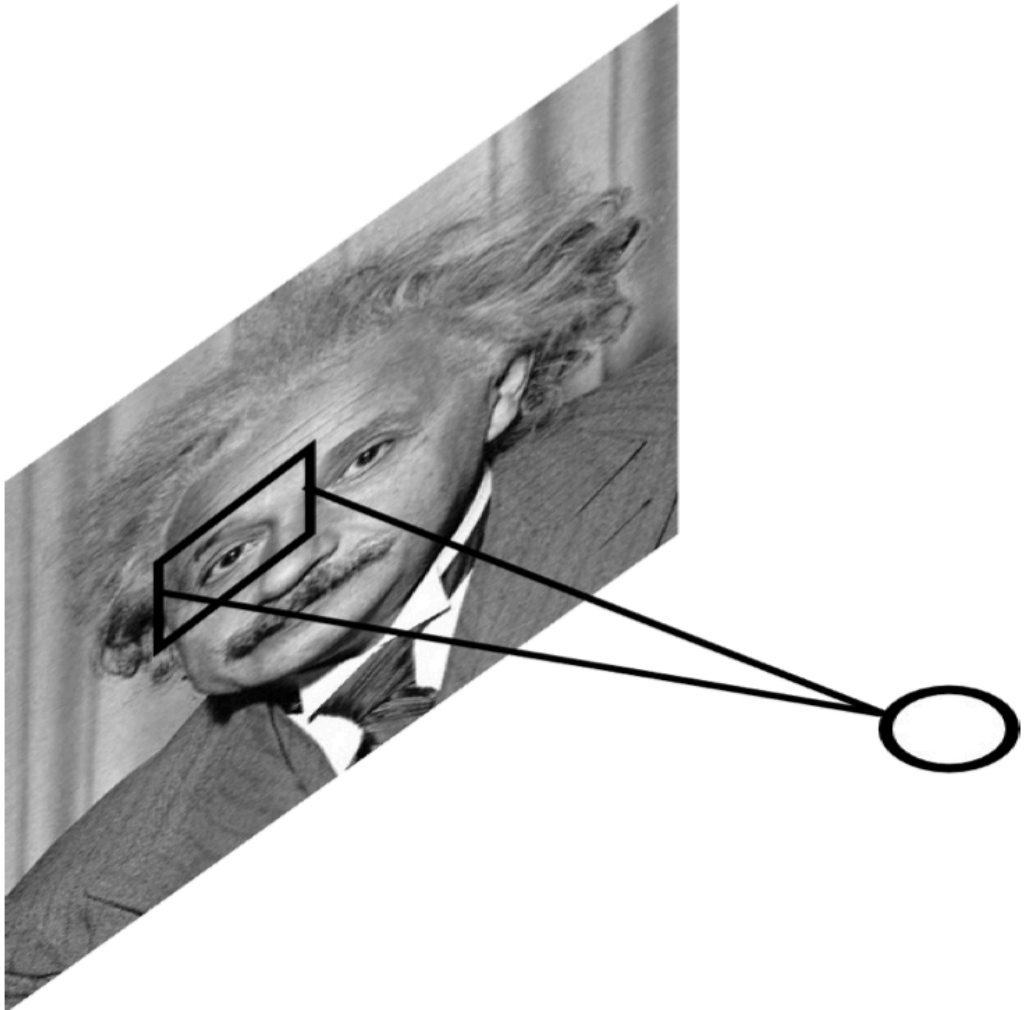
Performed also across features and in the higher layers..

Effects:
- improves invariance
- improves optimization
- increases sparsity

Note: computational cost is negligible w.r.t. conv. layer.

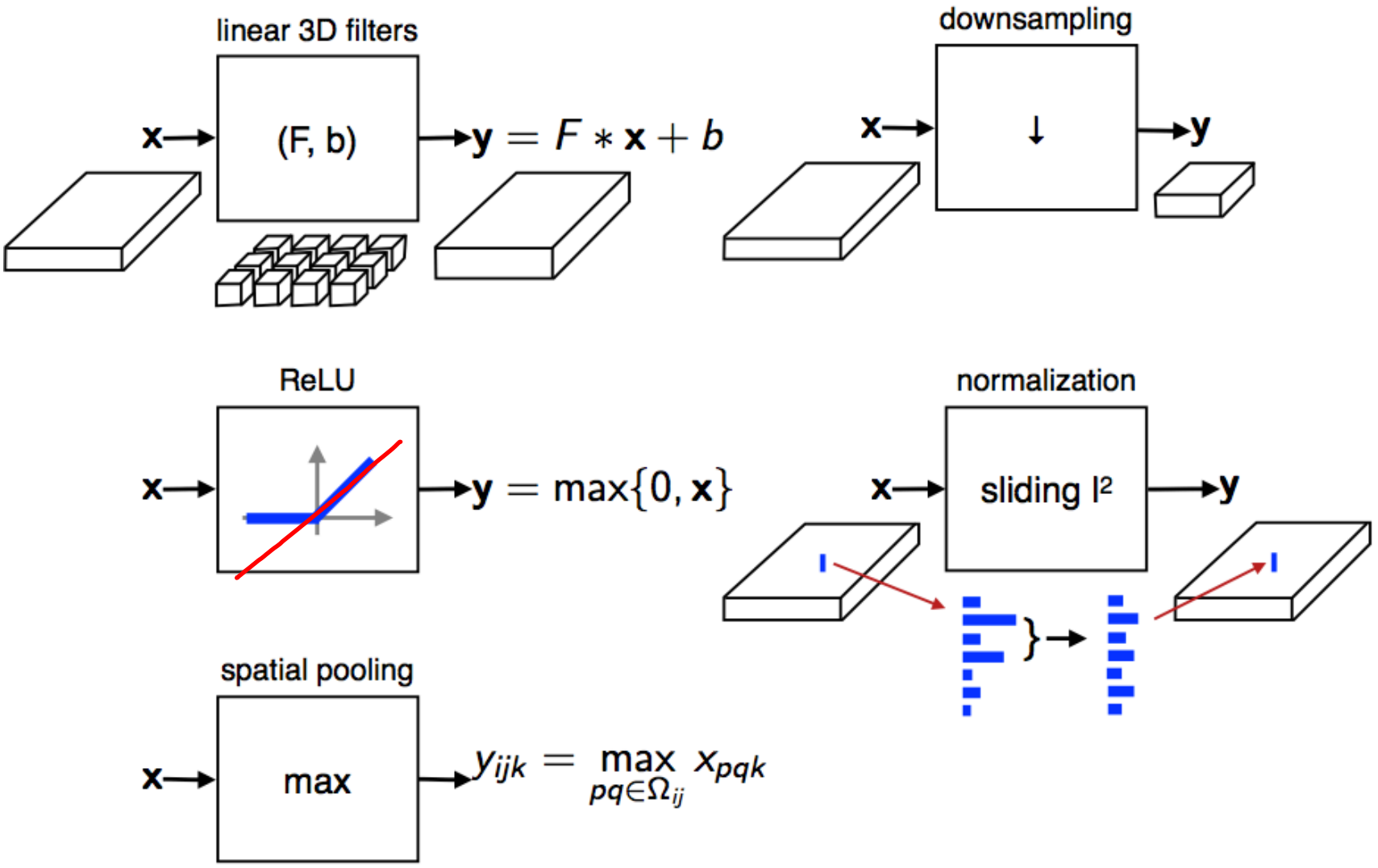
Local Contrast Normalization

$$h^{i+1}(x,y) = \frac{h^i(x,y) - m^i(N(x,y))}{max(\epsilon, \sigma^i(N(x,y)))}$$



CNN components

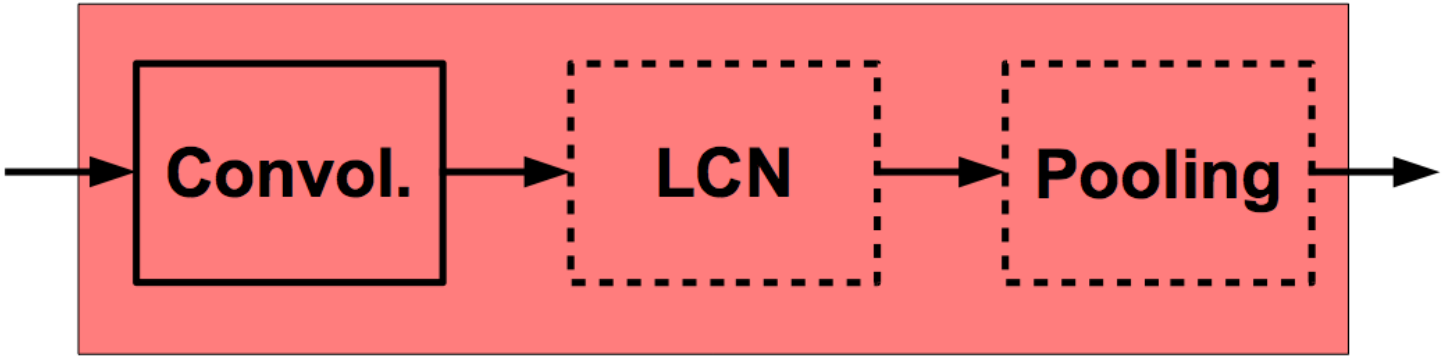
75

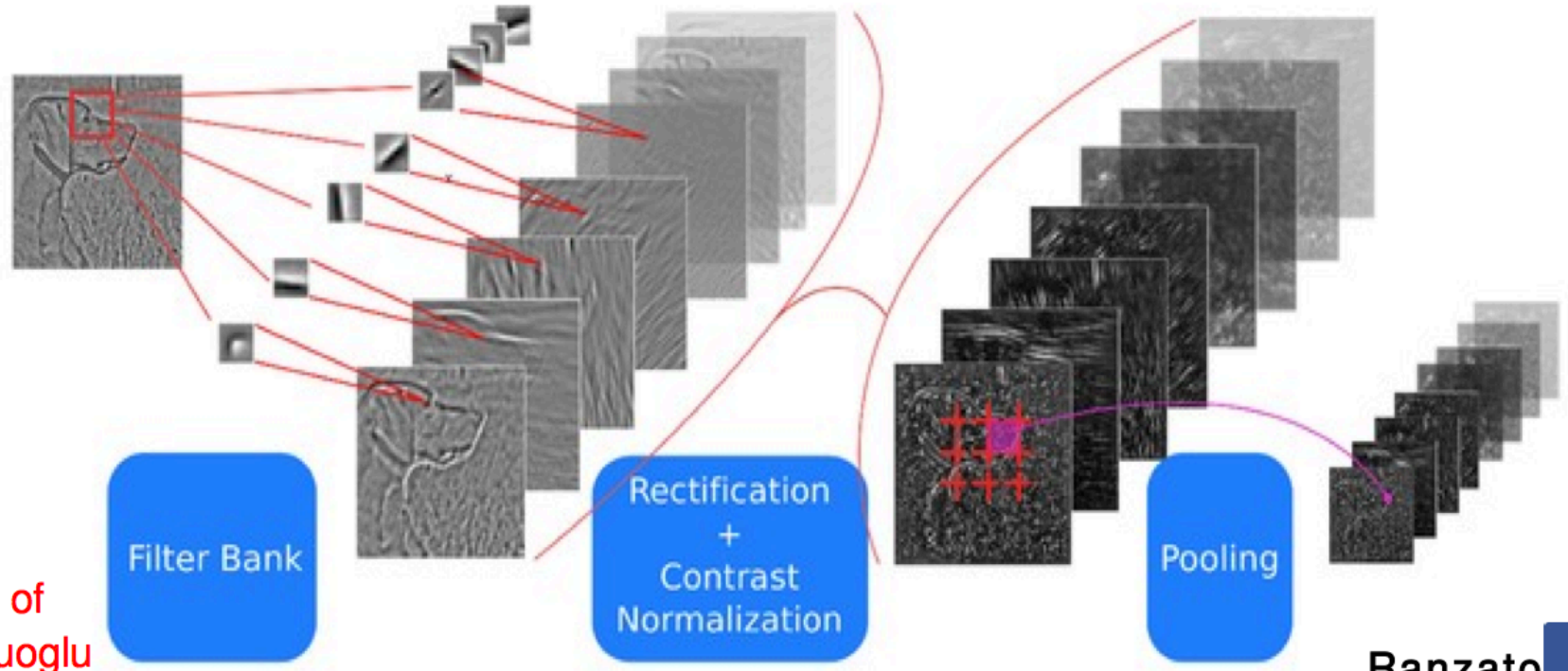


Slide credits: A. Vedaldi

ConvNets: Typical Stage

One stage (zoom)



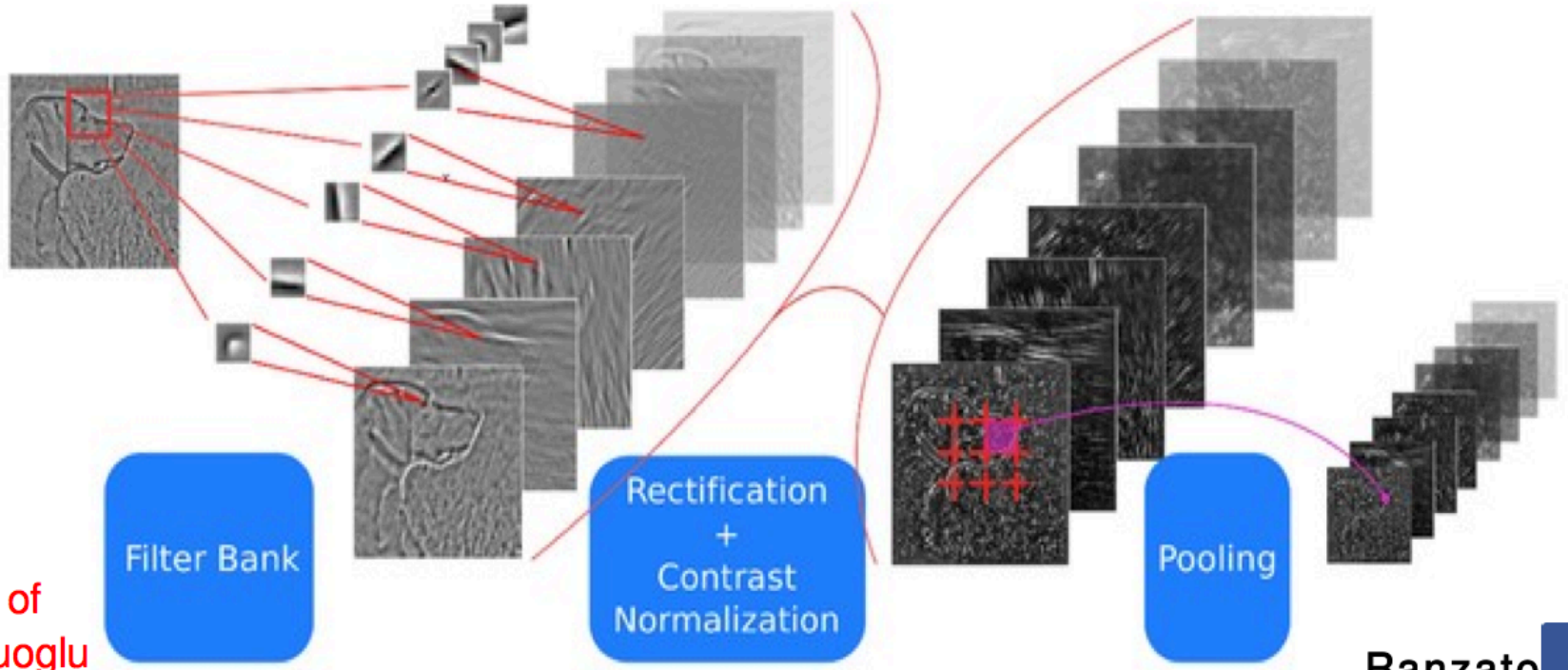


courtesy of
K. Kavukcuoglu

Ranzato

Note: after one stage the number of feature maps is usually increased (conv. layer) and the spatial resolution is usually decreased (stride in conv. and pooling layers). Receptive field gets bigger.

Reasons:
- gain invariance to spatial translation (pooling layer)
- increase specificity of features (approaching object specific units)



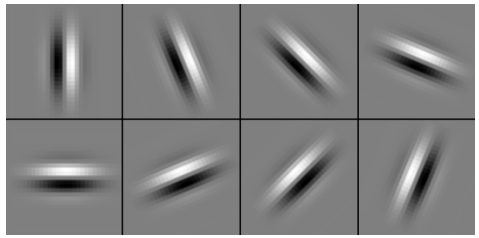
courtesy of
K. Kavukcuoglu

Ranzato

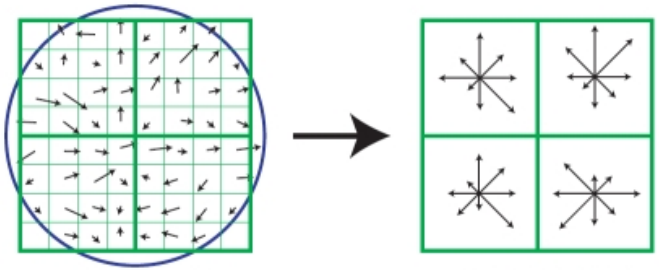
SIFT Descriptor

Image Pixels

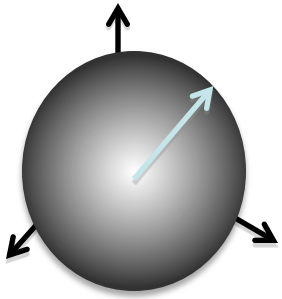
Apply Gabor filters



Spatial pool (Sum)



Normalize to unit length



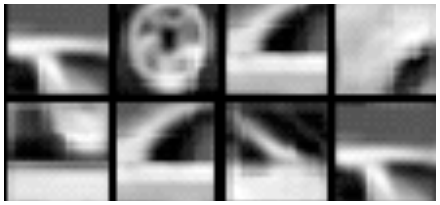
Feature Vector

Slide: R. Fergus

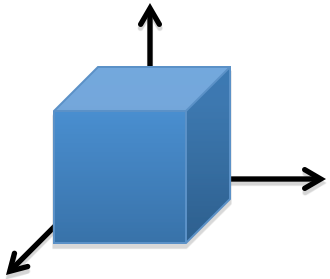
Spatial Pyramid Matching

SIFT Features

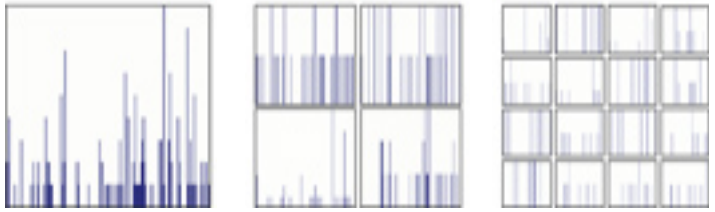
Filter with Visual Words



Max



Multi-scale spatial pool (Sum)

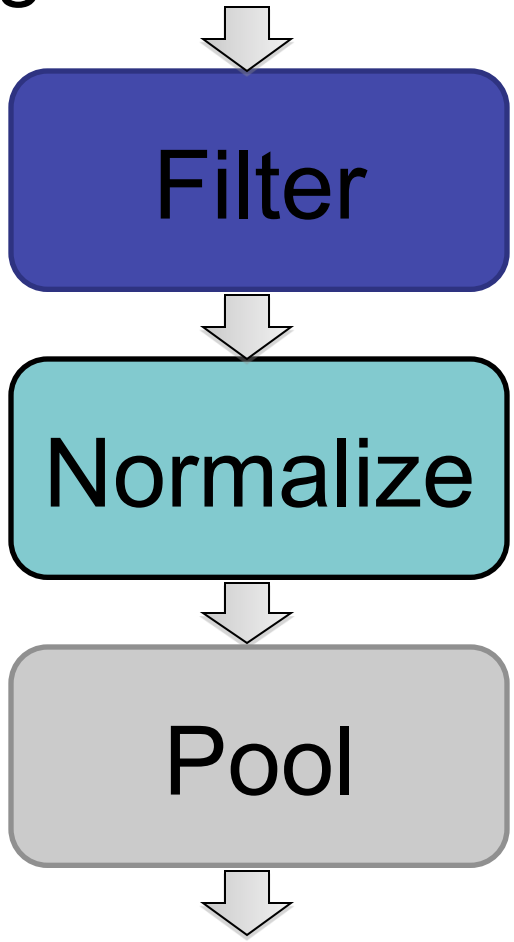


Classifier

Slide: R. Fergus

Single Layer Architecture

Input: Image Pixels / Features

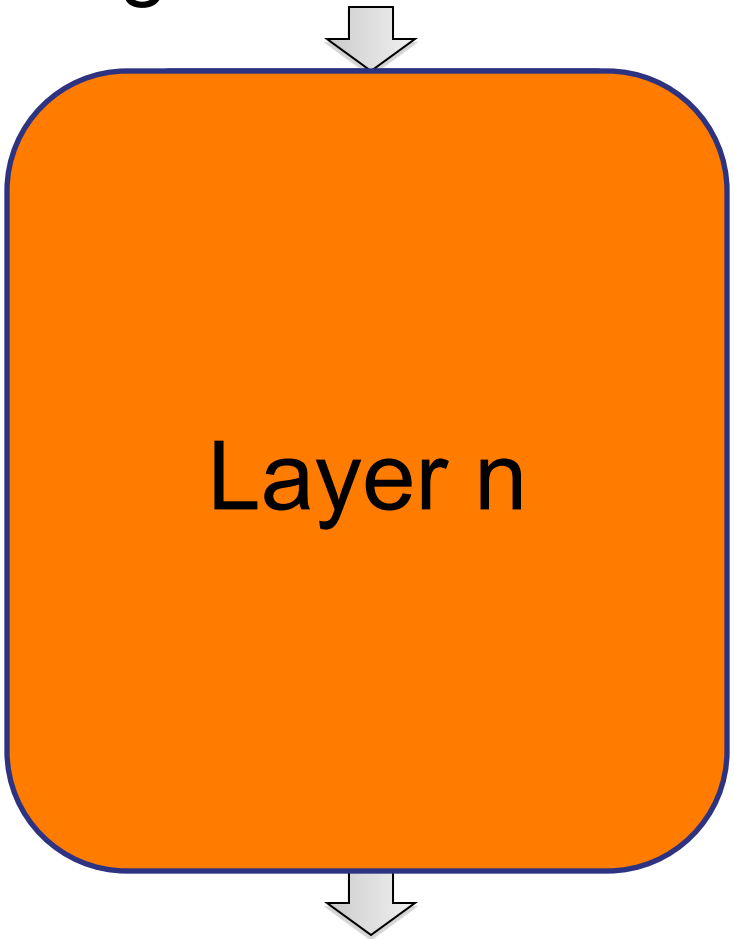


Output: Features / Classifier

Slide: R. Fergus

Single Layer Architecture

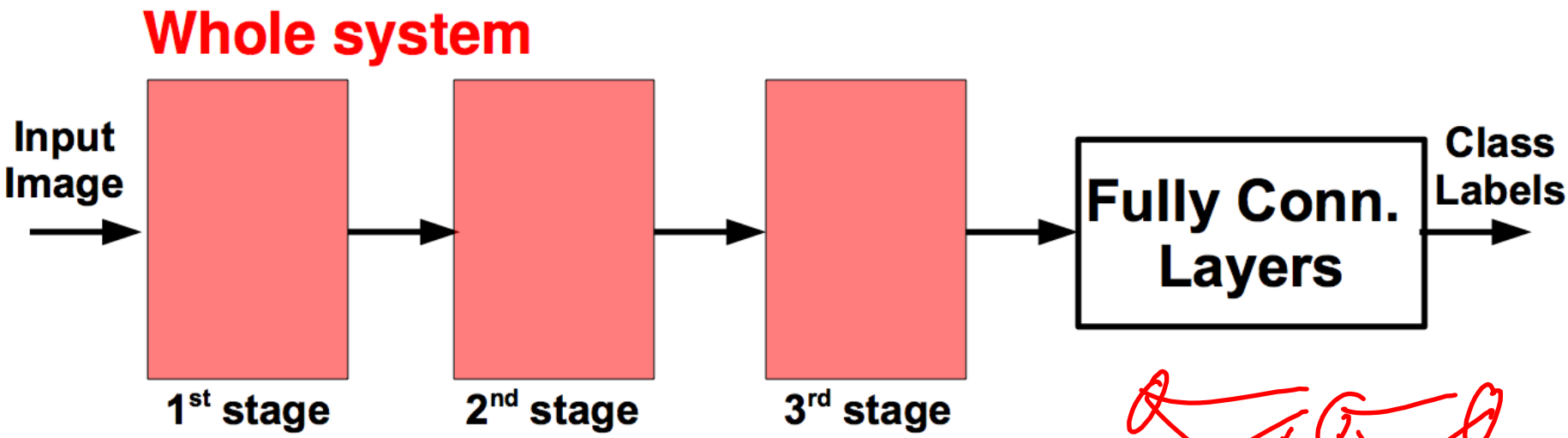
Input: Image Pixels / Features

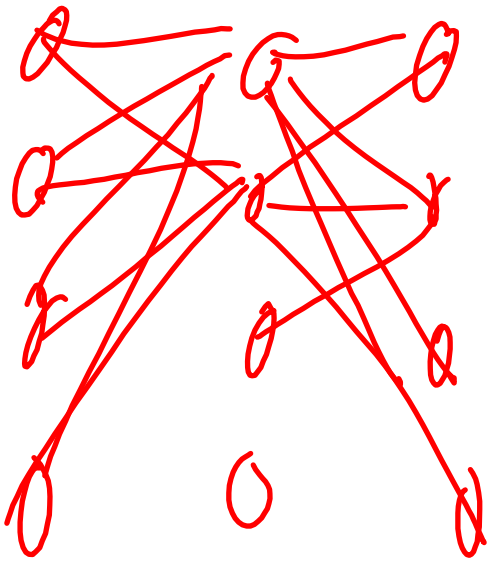


Output: Features / Classifier

Slide: R. Fergus

ConvNets: Typical Architecture





Conceptually similar to:

SIFT → K-Means → Pyramid Pooling → SVM

Lazebnik et al. "...Spatial Pyramid Matching..." CVPR 2006

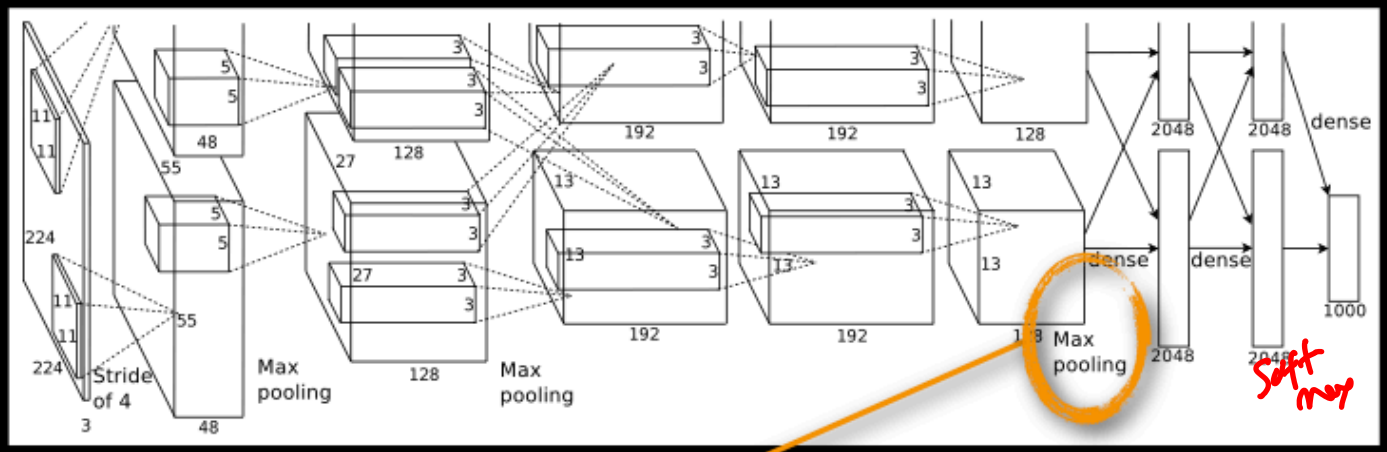
SIFT → Fisher Vect. → Pooling → SVM

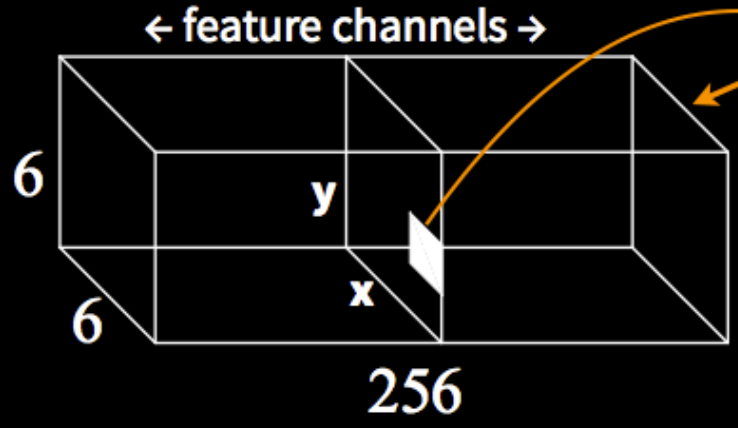
Sanchez et al. "Image classifcation with F.V.: Theory and practice" IJCV 2012

Slide: M-A Ranzatto

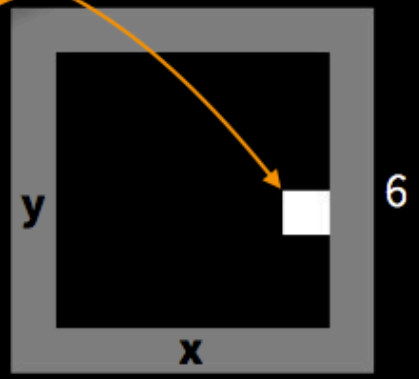
What is in a CNN?

"stimulus"

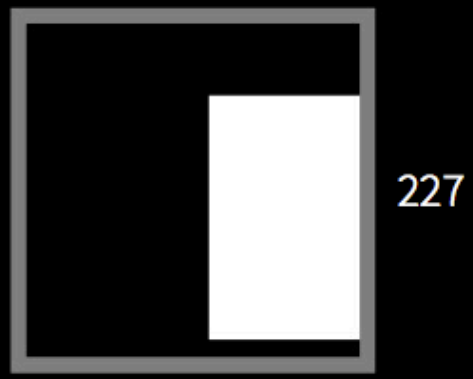




pool$_5$ feature map

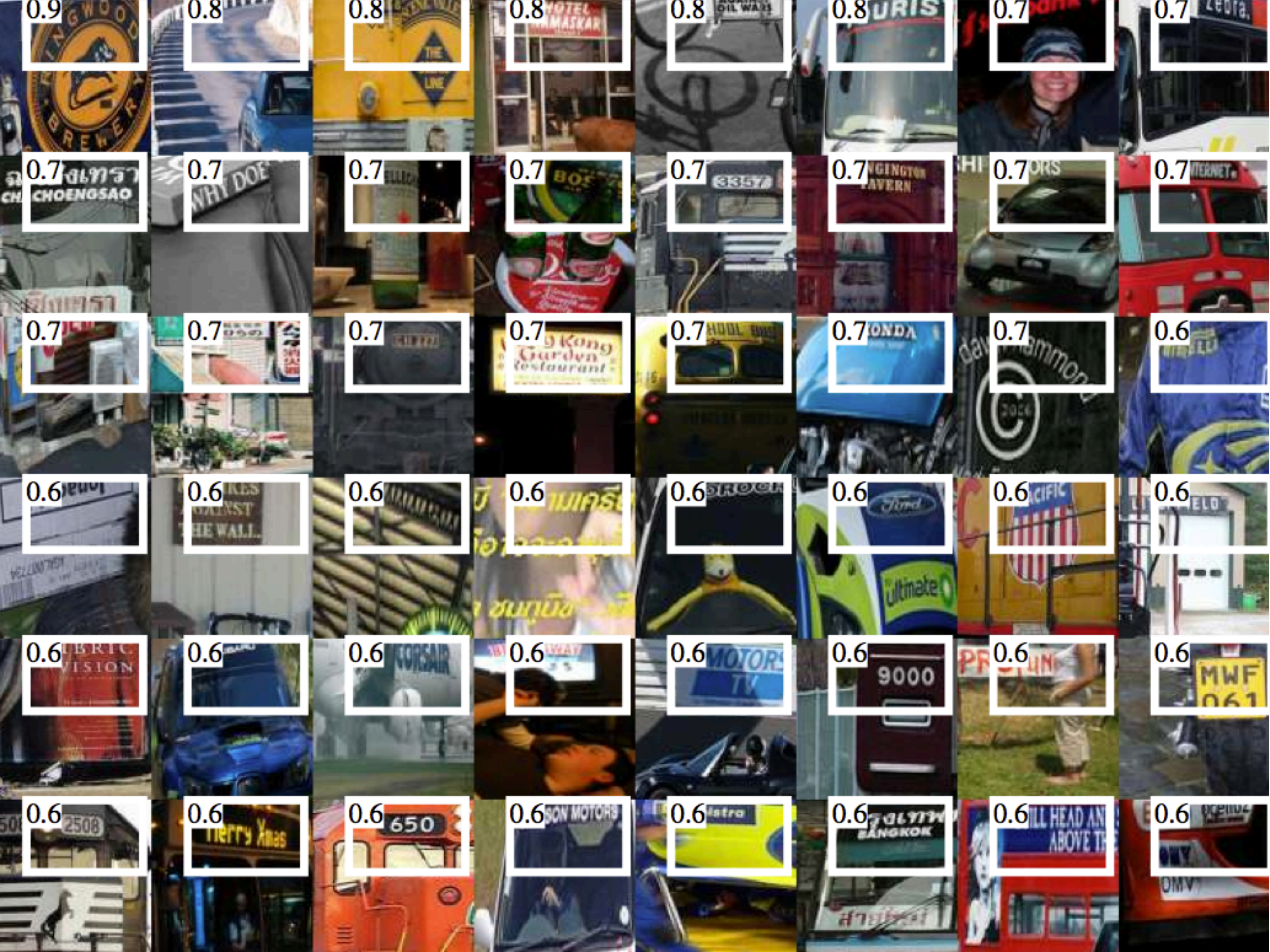


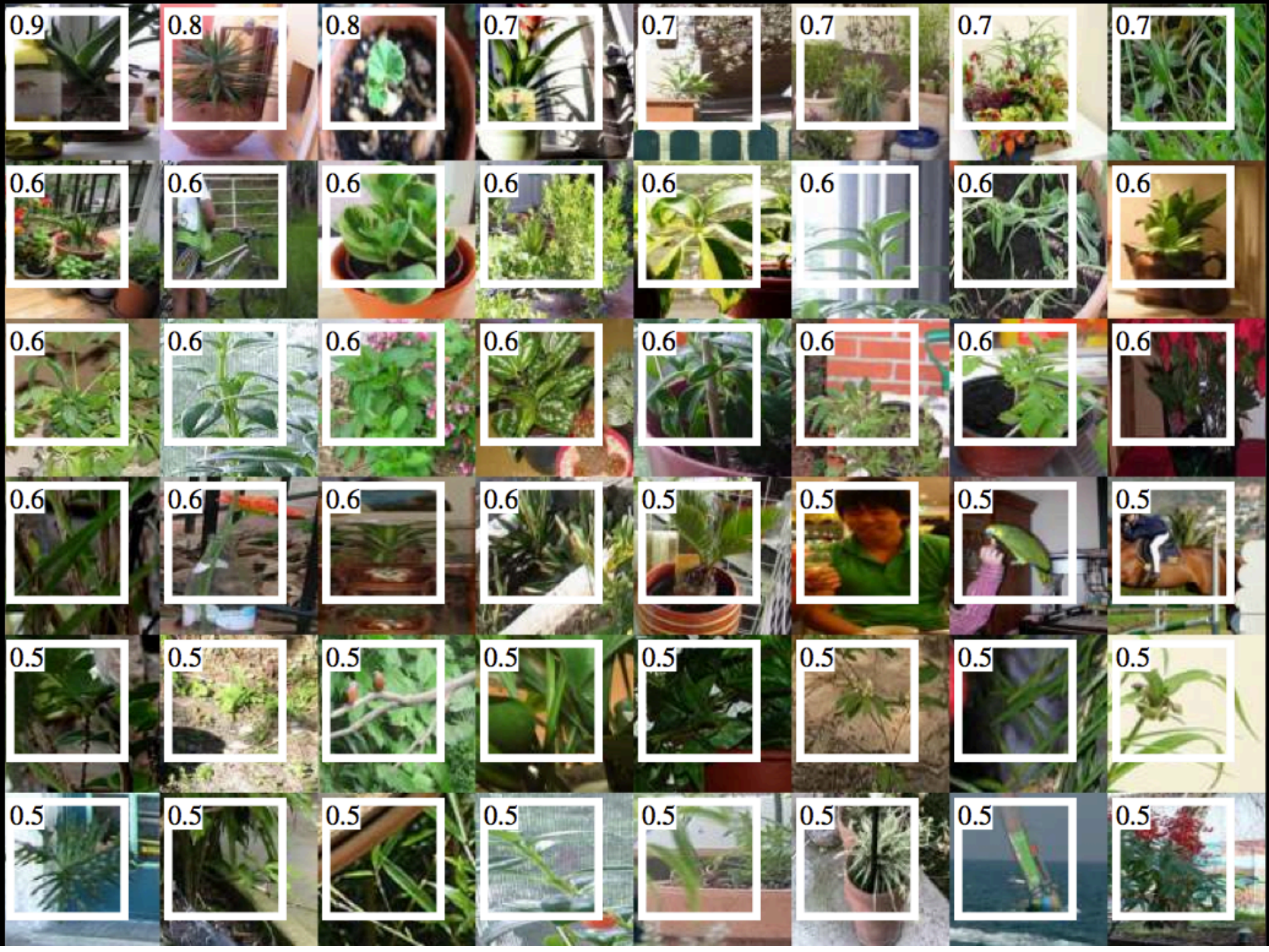
feature position

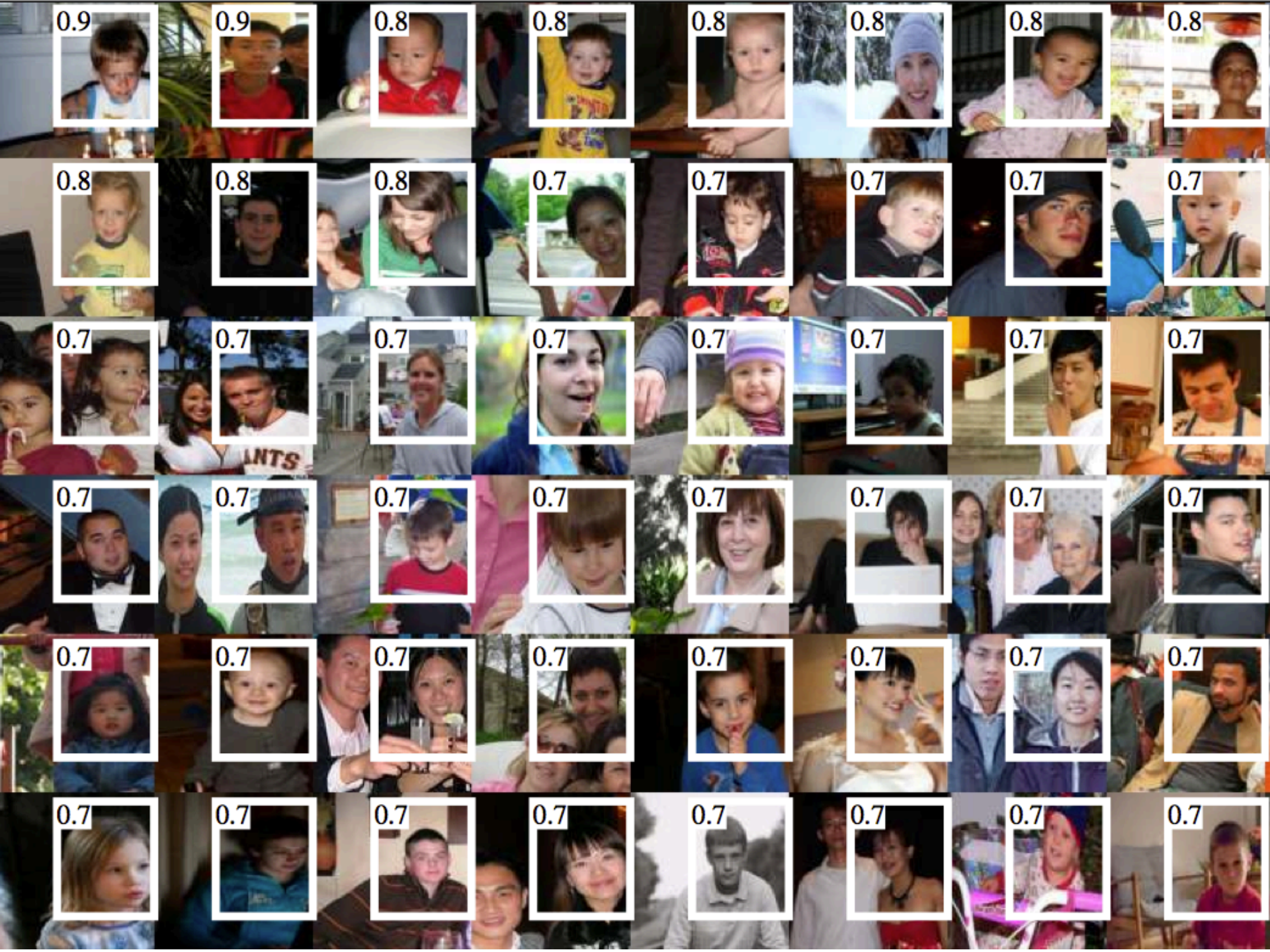


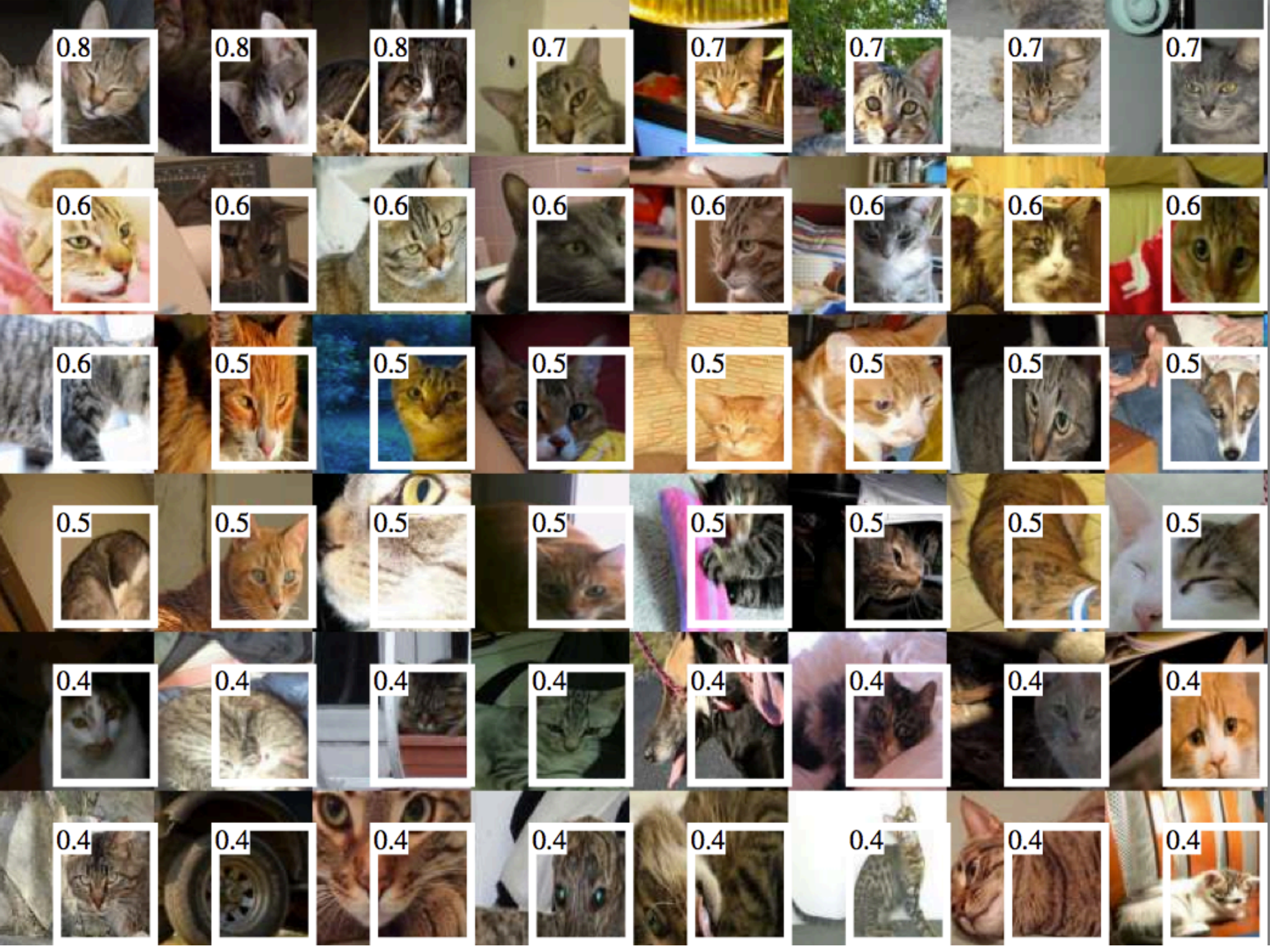
receptive field

Visualize images that activate pool$_5$ a feature

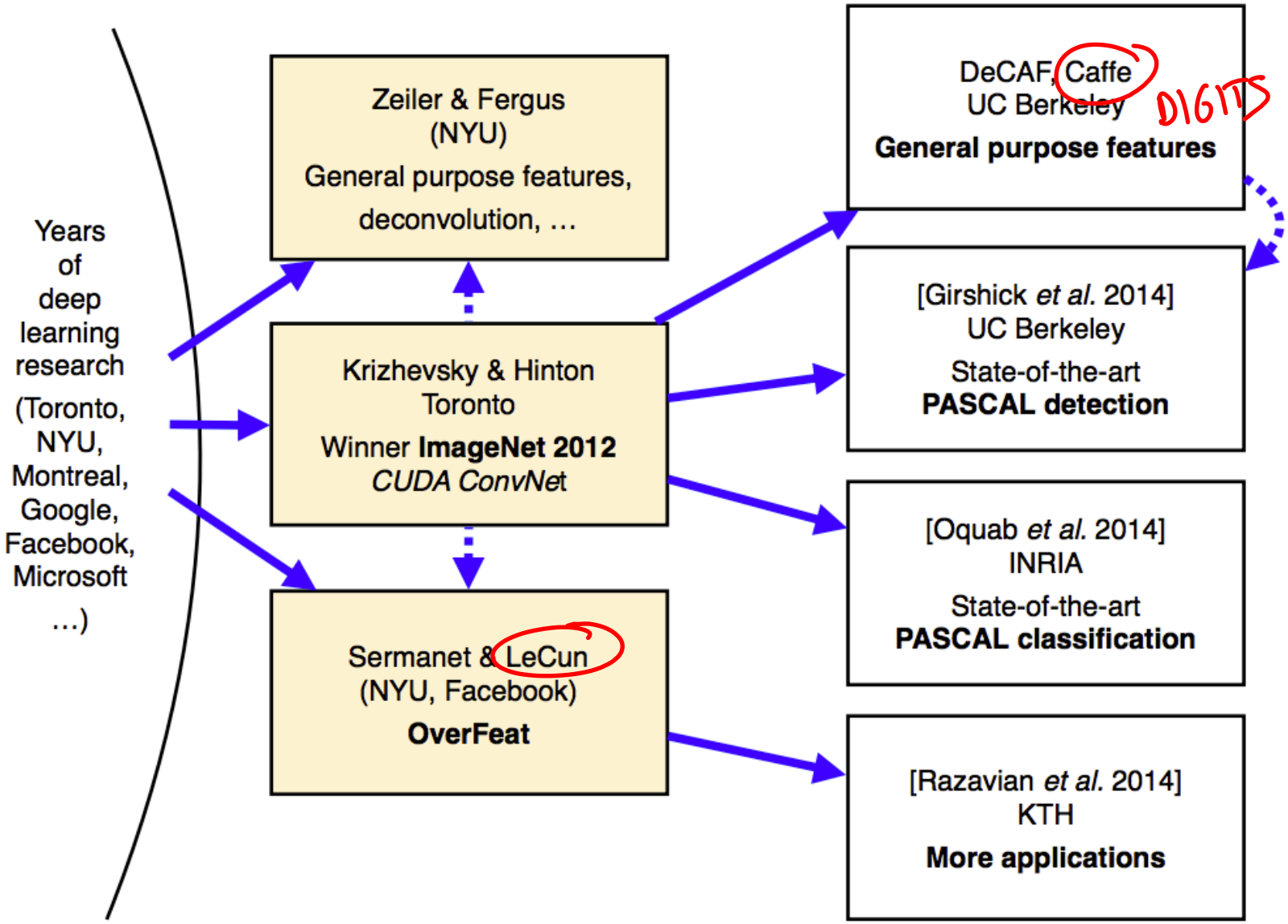

0.9 0.8 0.8 0.8 0.8 0.8 0.7 0.7
0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7
0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.6
0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6
0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6
0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6
CHOENGSAO
3357
TAVERN
Garden
Restaurant
THE WALL.
Ford
ultimate
MOTORS
TV
9000
MWF
2508
650
BANGKOK


0.9
0.8
0.8
0.7
0.7
0.7
0.7
0.7
0.6
0.6
0.6
0.6
0.6
0.6
0.6
0.6
0.6
0.6
0.6
0.6
0.6
0.6
0.6
0.6
0.6
0.6
0.6
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5


0.9 0.9 0.8 0.8 0.8 0.8 0.8 0.8
0.8 0.8 0.8 0.7 0.7 0.7 0.7 0.7
0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7
0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7
0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7
0.7 0.7 0.7 0.7 0.7 0.7 0.7 0.7


0.8
0.8
0.8
0.7
0.7
0.7
0.7
0.7
0.6
0.6
0.6
0.6
0.6
0.6
0.6
0.6
0.6
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.5
0.4
0.4
0.4
0.4
0.4
0.4
0.4
0.4
0.4
0.4
0.4
0.4
0.4
0.4
0.4
0.4

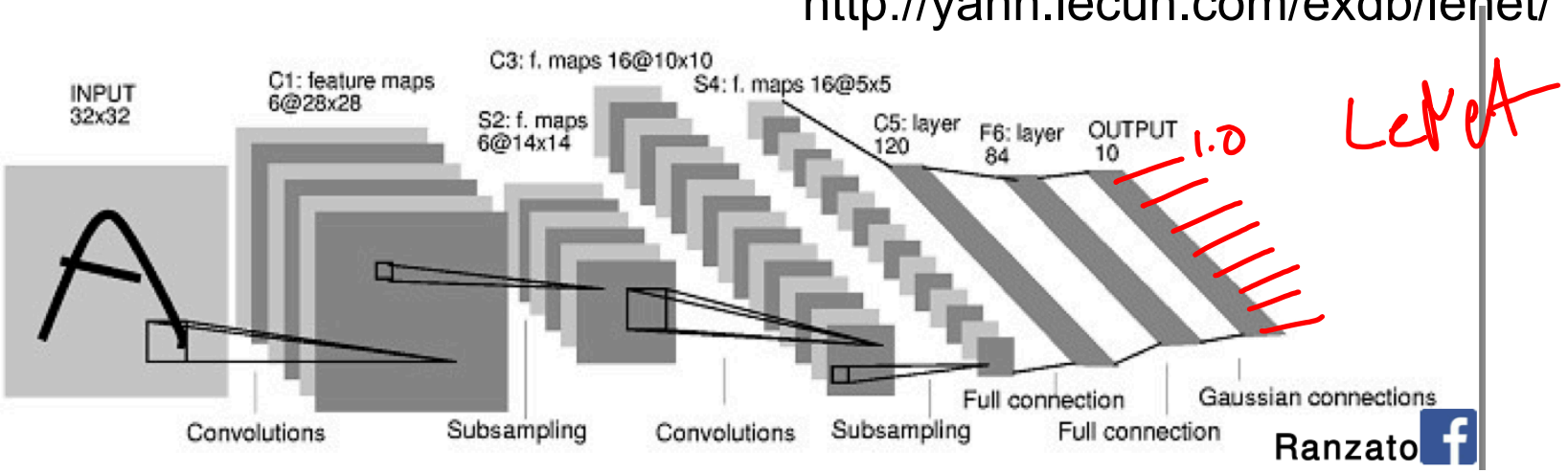
CNNs in vision



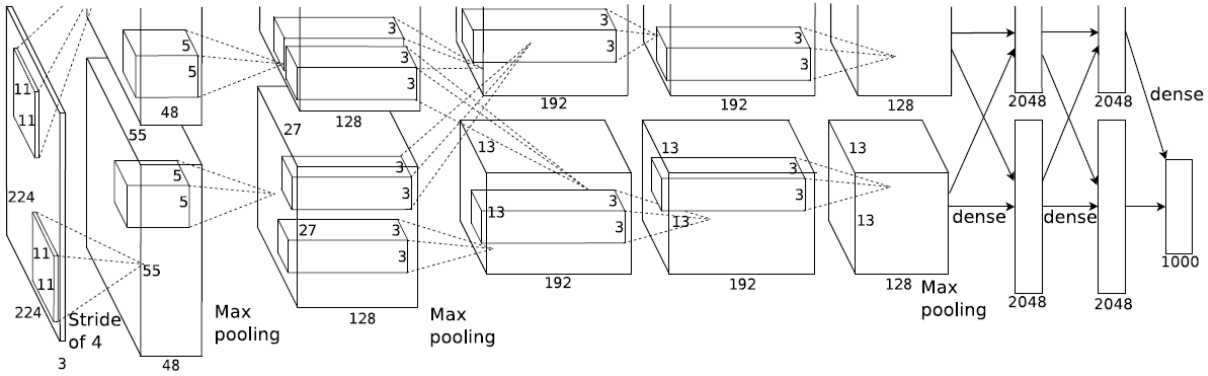
Slide: A. Vedaldi

Famous CNNs

http://yann.lecun.com/exdb/lenet/



Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, november 1998



A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. *NIPS*13

Latest & greatest CNNs (deeper and deeper)

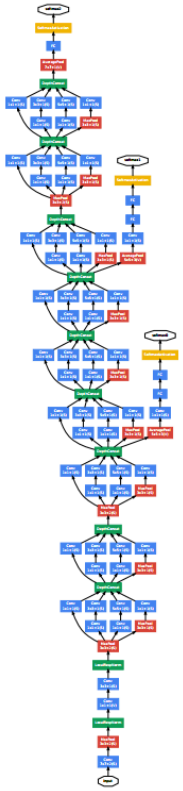


Figure 3: GoogLeNet network with all the bells and whistles

GoogLeNet

Table 1: **ConvNet configurations** (shown in columns). The depth of the configurations incre from the left (A) to the right (E), as more layers are added (the added layers are shown in bold). convolutional layer parameters are denoted as "conv⟨receptive field size⟩-⟨number of channe The ReLU activation function is not shown for brevity.

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

Table 2: **Number of parameters** (in millions).

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

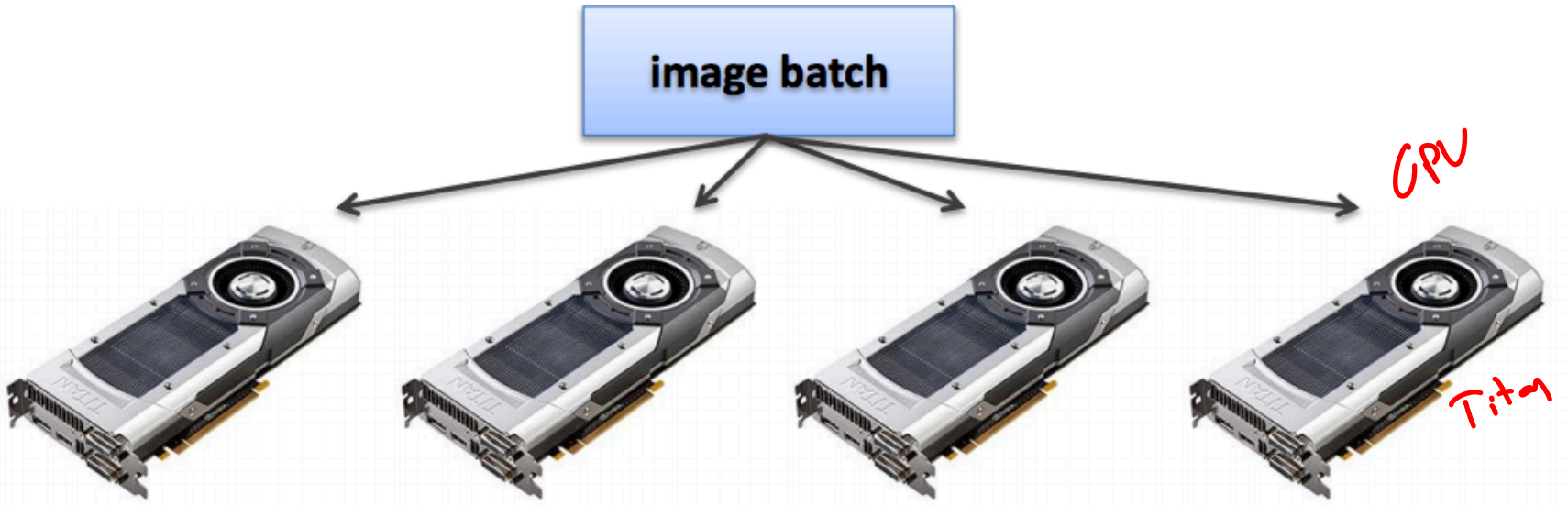
GoogLeNet
C. Szegedy et al.
Going deeper with convolutions
arXiv technical report, 2014

VGG net (Oxford):
K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, arXiv, 2014

Really large CNNs

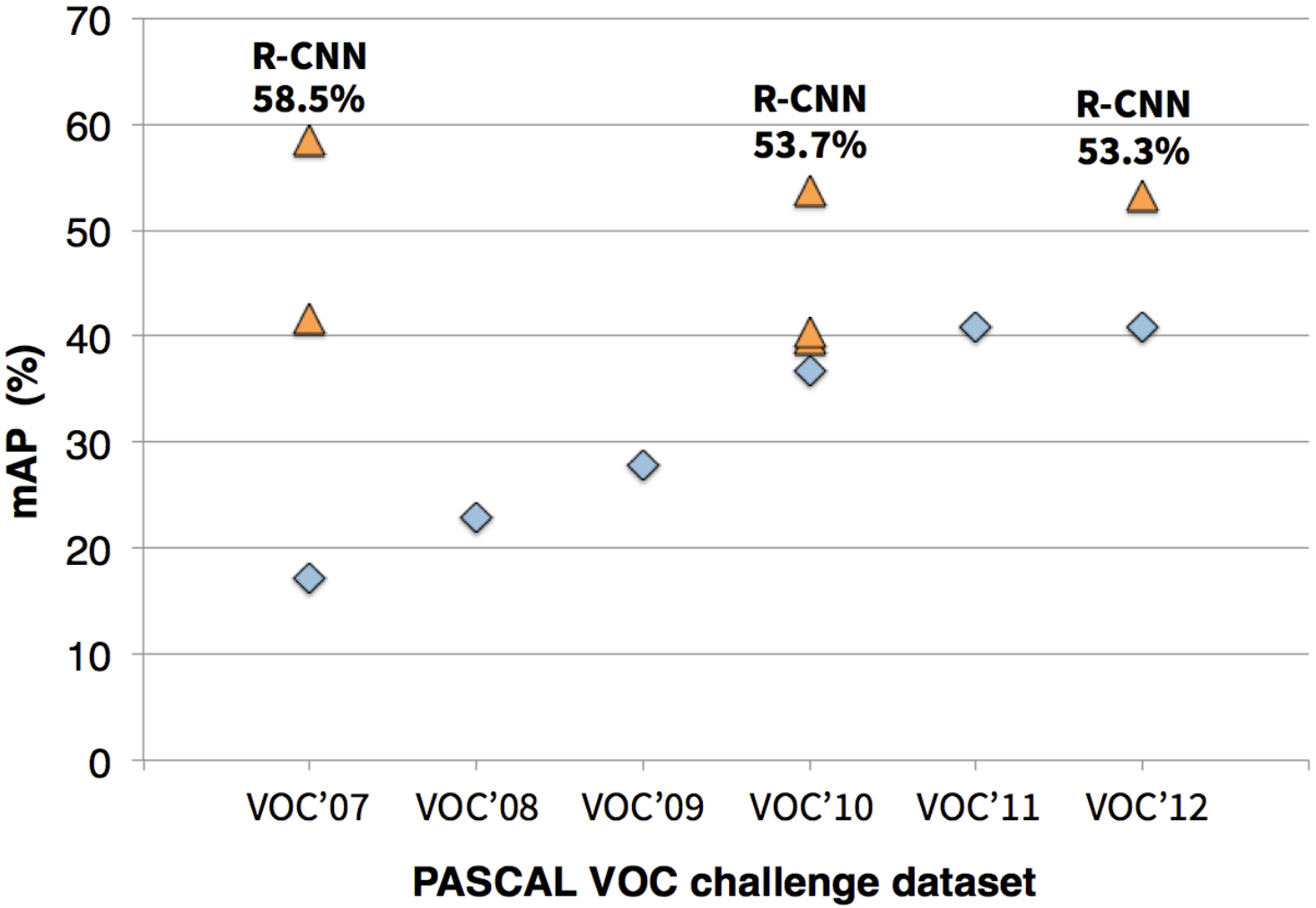
From VGG's network presentation:

- Heavily-modified Caffe C++ toolbox
- Multiple GPU support
  - 4 x NVIDIA Titan, off-the-shelf workstation
  - data parallelism for training and testing
  - ~3.75 times speed-up, 2-3 weeks for training



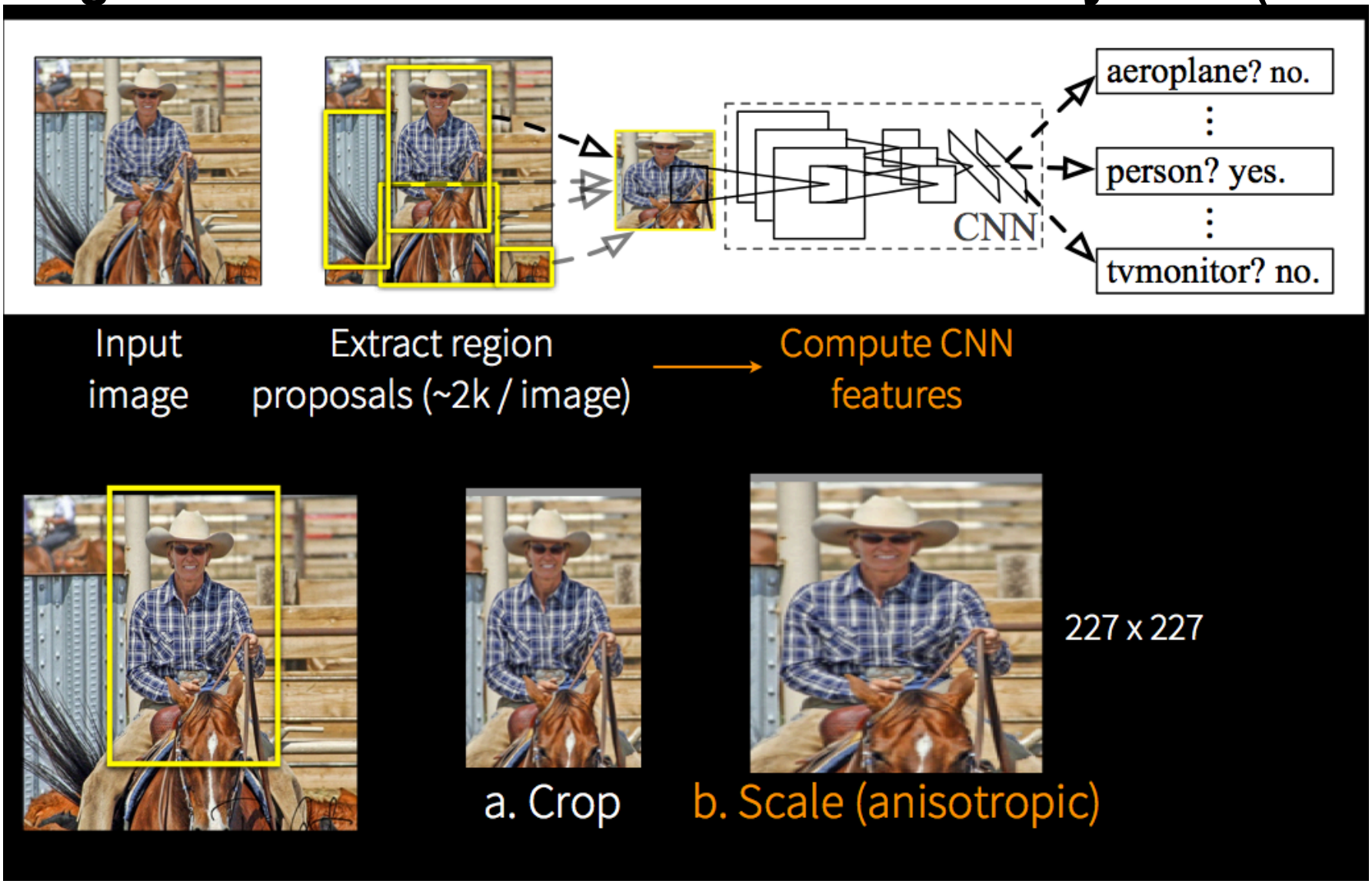
Imagenet top-5 error rate: 36%-> 18% (2012) -> 6% (2014)

Object recognition, 2013



A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. *NIPS*13
P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. ICLR 14.
R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CVPR* 14

Regions with Convolutional Neural Net.s system (RCNN)

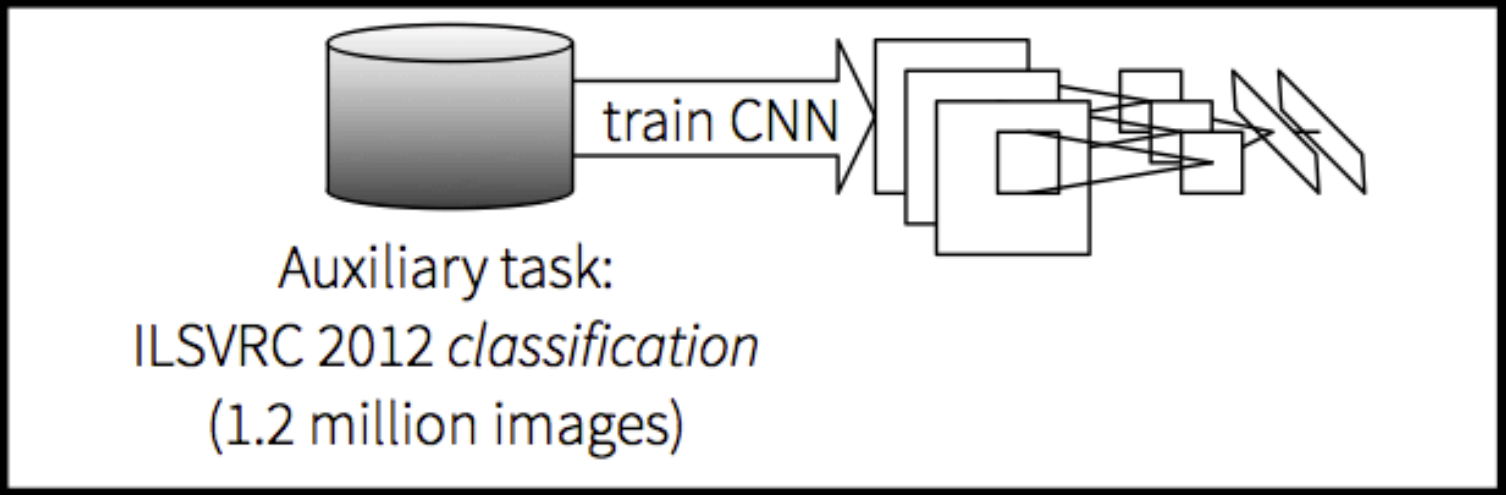


R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CVPR* 14

Regions with Convolutional Neural Net.s system (RCNN)

Supervised pre-training

Train a SuperVision CNN* for the 1000-way ILSVRC image classification task

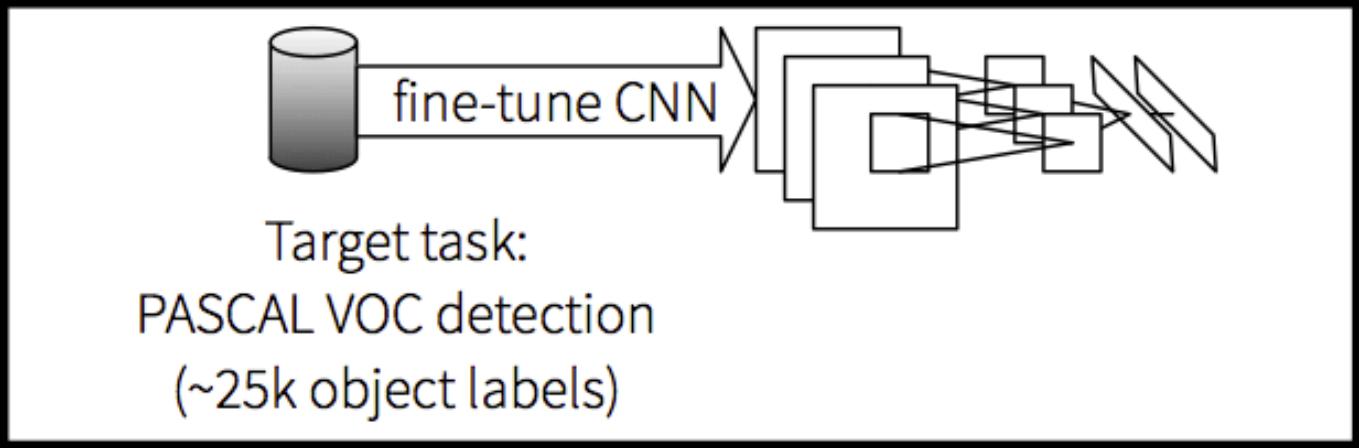


R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CVPR* 14

Regions with Convolutional Neural Net.s system (RCNN)

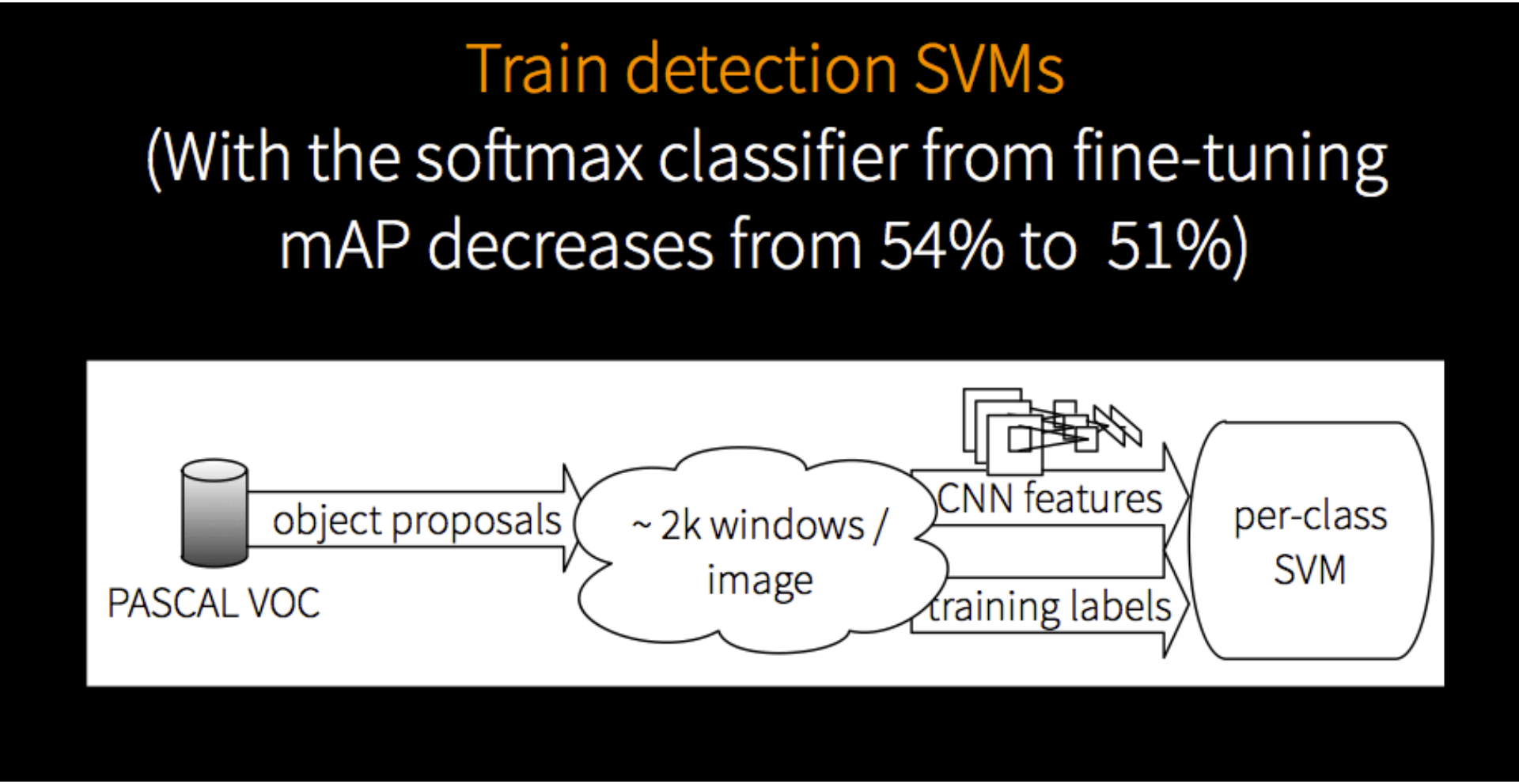
Fine-tune the CNN for detection

Transfer the representation learned for ILSVRC classification to PASCAL (or ImageNet detection)



R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CVPR* 14

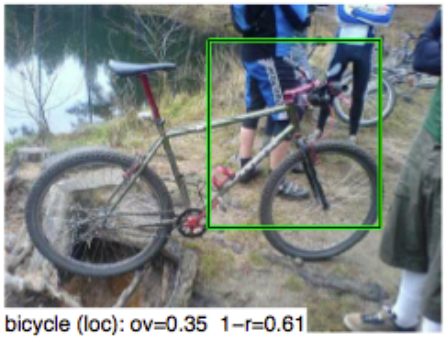
Regions with Convolutional Neural Net.s system (RCNN)



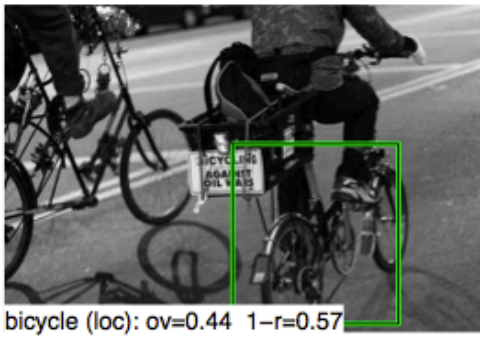
R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CVPR* 14

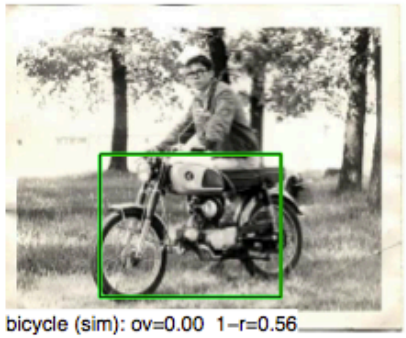
Detection results


bicycle (loc): ov=0.41 1−r=0.64


bicycle (loc): ov=0.35 1−r=0.61


bicycle (loc): ov=0.15 1−r=0.59

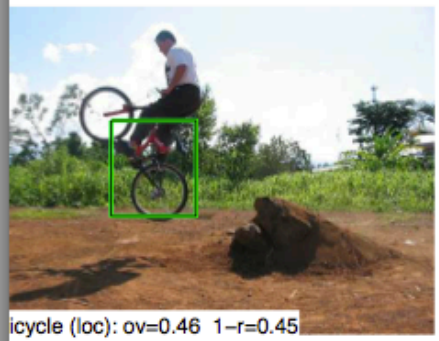

bicycle (loc): ov=0.44 1−r=0.57

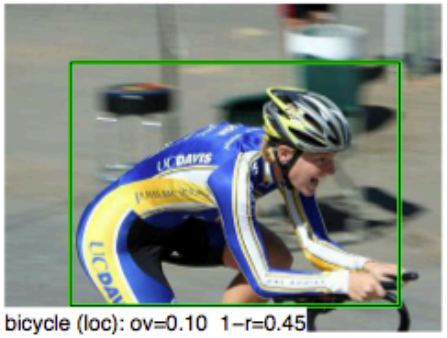

bicycle (sim): ov=0.00 1−r=0.56

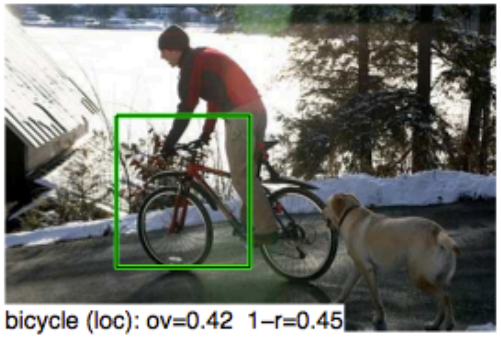

bicycle (bg): ov=0.00 1−r=0.52


bicycle (loc): ov=0.55 1−r=0.47


bicycle (bg): ov=0.00 1−r=0.47


icycle (loc): ov=0.46 1−r=0.45

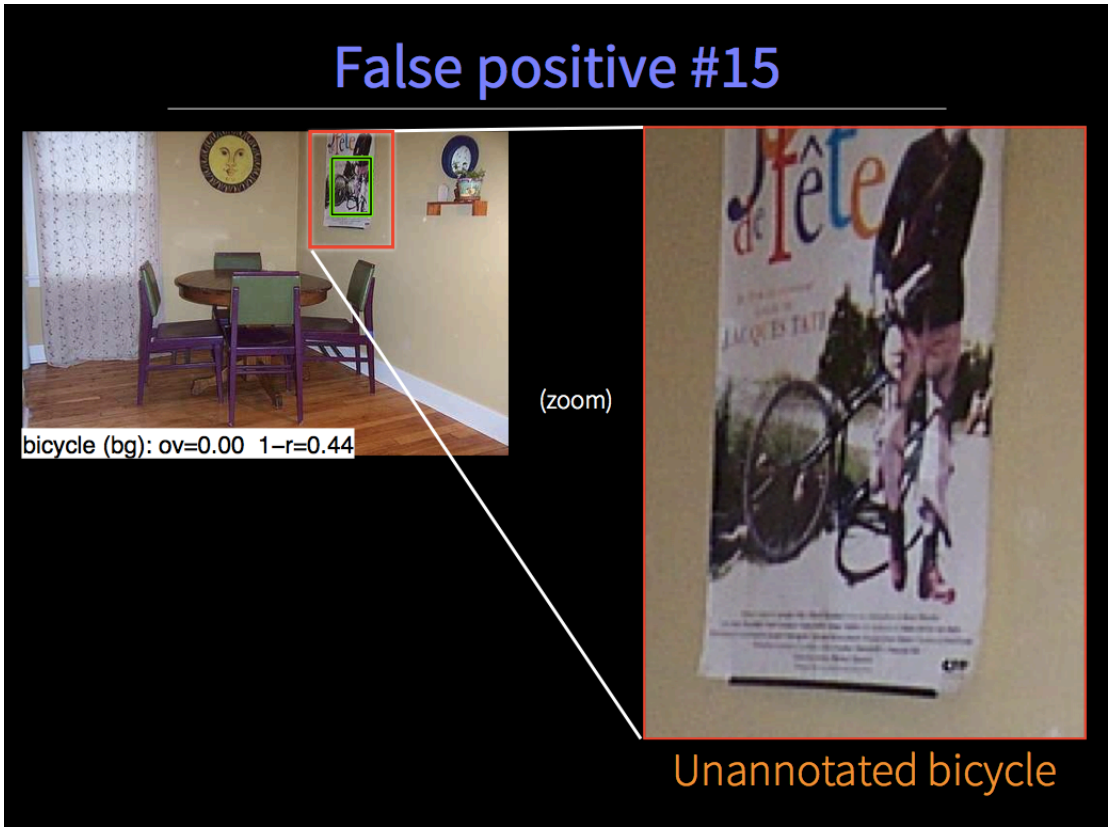

bicycle (loc): ov=0.10 1−r=0.45


bicycle (loc): ov=0.42 1−r=0.45


bicycle (bg): ov=0.00 1−r=0.44

R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CVPR* 14

Failures: mostly localization errors



R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CVPR* 14

Detection results

VOC 2007 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
R-CNN $pool_5$	51.8	60.2	36.4	27.8	23.2	52.8	60.6	49.2	18.3	47.8	44.3	40.8	56.6	58.7	42.4	23.4	46.1	36.7	51.3	55.7	44.2
R-CNN fc_6	59.3	61.8	43.1	34.0	25.1	53.1	60.6	52.8	21.7	47.8	42.7	47.8	52.5	58.5	44.6	25.6	48.3	34.0	53.1	58.0	46.2
R-CNN fc_7	57.6	57.9	38.5	31.8	23.7	51.2	58.9	51.4	20.0	50.5	40.9	46.0	51.6	55.9	43.3	23.3	48.1	35.3	51.0	57.4	44.7
R-CNN FT $pool_5$	58.2	63.3	37.9	27.6	26.1	54.1	66.9	51.4	26.7	55.5	43.4	43.1	57.7	59.0	45.8	28.1	50.8	40.6	53.1	56.4	47.3
R-CNN FT fc_6	63.5	66.0	47.9	37.7	29.9	62.5	70.2	60.2	32.0	57.9	47.0	53.5	60.1	64.2	52.2	31.3	55.0	50.0	57.7	63.0	53.1
R-CNN FT fc_7	64.2	69.7	50.0	41.9	32.0	62.6	71.0	60.7	32.7	58.5	46.5	56.1	60.6	66.8	54.2	31.5	52.8	48.9	57.9	64.7	54.2
R-CNN FT fc_7 BB	**68.1**	**72.8**	**56.8**	**43.0**	**36.8**	**66.3**	**74.2**	**67.6**	**34.4**	**63.5**	**54.5**	**61.2**	**69.1**	**68.6**	**58.7**	**33.4**	**62.9**	**51.1**	**62.5**	**64.8**	**58.5**
DPM v5 [20]	33.2	60.3	10.2	16.1	27.3	54.3	58.2	23.0	20.0	24.1	26.7	12.7	58.1	48.2	43.2	12.0	21.1	36.1	46.0	43.5	33.7
DPM ST [28]	23.8	58.2	10.5	8.5	27.1	50.4	52.0	7.3	19.2	22.8	18.1	8.0	55.9	44.8	32.4	13.3	15.9	22.8	46.2	44.9	29.1
DPM HSC [31]	32.2	58.3	11.5	16.3	30.6	49.9	54.8	23.5	21.5	27.7	34.0	13.7	58.1	51.6	39.9	12.4	23.5	34.4	47.4	45.2	34.3

Table 2: Detection average precision (%) on VOC 2007 test. Rows 1-3 show R-CNN performance without fine-tuning. Rows 4-6 show results for the CNN pre-trained on ILSVRC 2012 and then fine-tuned (FT) on VOC 2007 trainval. Row 7 includes a simple bounding-box regression (BB) stage that reduces localization errors (Section C). Rows 8-10 present DPM methods as a strong baseline. The first uses only HOG, while the next two use different feature learning approaches to augment or replace HOG.

VOC 2007 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
R-CNN T-Net	64.2	69.7	50.0	41.9	32.0	62.6	71.0	60.7	32.7	58.5	46.5	56.1	60.6	66.8	54.2	31.5	52.8	48.9	57.9	64.7	54.2
R-CNN T-Net BB	68.1	72.8	56.8	43.0	36.8	66.3	74.2	67.6	34.4	63.5	54.5	61.2	69.1	68.6	58.7	33.4	62.9	51.1	62.5	64.8	58.5
R-CNN O-Net	71.6	73.5	58.1	42.2	39.4	70.7	76.0	74.5	38.7	71.0	56.9	74.5	67.9	69.6	59.3	**35.7**	62.1	64.0	66.5	**71.2**	62.2
R-CNN O-Net BB	**73.4**	**77.0**	**63.4**	**45.4**	**44.6**	**75.1**	**78.1**	**79.8**	**40.5**	**73.7**	**62.2**	**79.4**	**78.1**	**73.1**	**64.2**	35.6	**66.8**	**67.2**	**70.4**	71.1	**66.0**

Table 3: Detection average precision (%) on VOC 2007 test for two different CNN architectures. The first two rows are results from Table 2 using Krizhevsky et al.'s architecture (T-Net). Rows three and four use the recently proposed 16-layer architecture from Simonyan and Zisserman (O-Net) [43].

R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CVPR* 14

G. Papandreou, I. Kokkinos and P. A. Savalle
Untangling Local and Global Deformations in Deep Convolutional Networks for Image Classification and Sliding Window Detection
arXiv:1412.0296, 2014

George Papandreou
Toyota Technological Institute at Chicago
(now at Google)

Pierre-Andre Savalle
Ecole Centrale Paris
(now at CISCO)

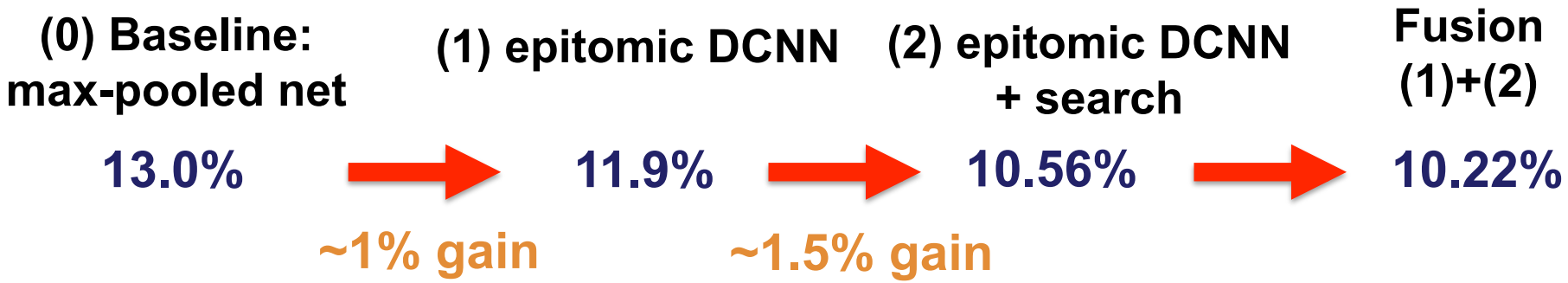
Earlier: ImageNet workshop, September 2014, G.P & I. K.

TTIC_ECP entry in a nutshell

Goal: Invariance in Deep CNNs

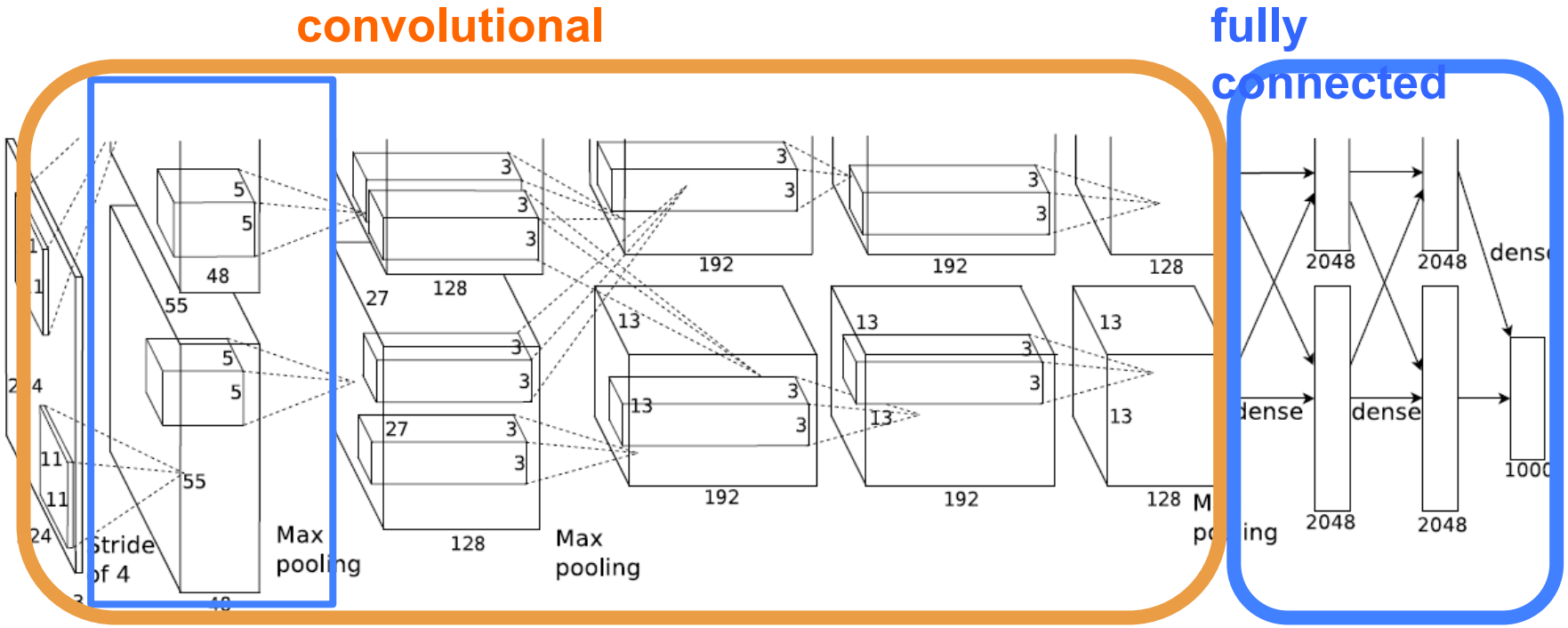
Part 1: Deep epitomic nets: local translation (deformation)

Part 2: Global scaling and translation



Top-5 error. All DCNNs have 6 convolutional and 2 fully-connected layers.

Deep Convolutional Neural Networks (DCNNs)



Cascade of convolution + max-pooling blocks

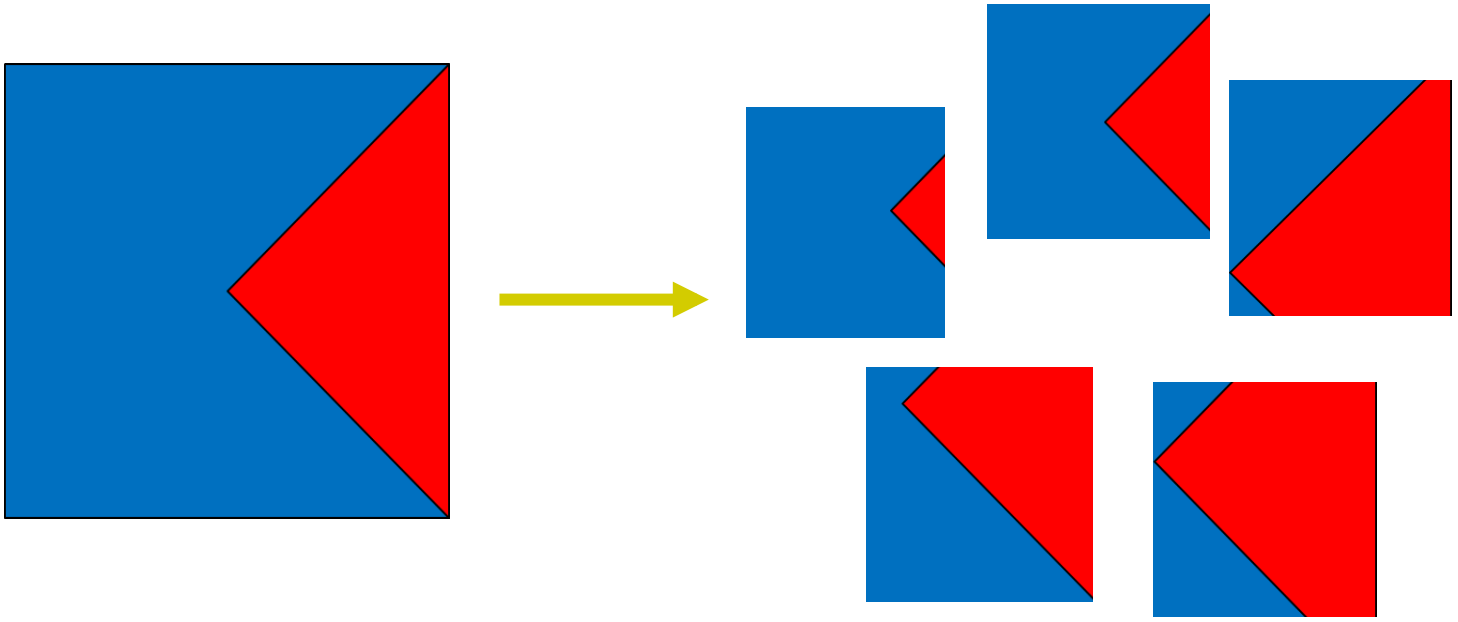
(deformation-invariant template matching)

Our work: different blocks (P1) & different architecture (P2)

LeCun et al.: Gradient-Based Learning Applied to Document Recognition, Proc. IEEE 1998

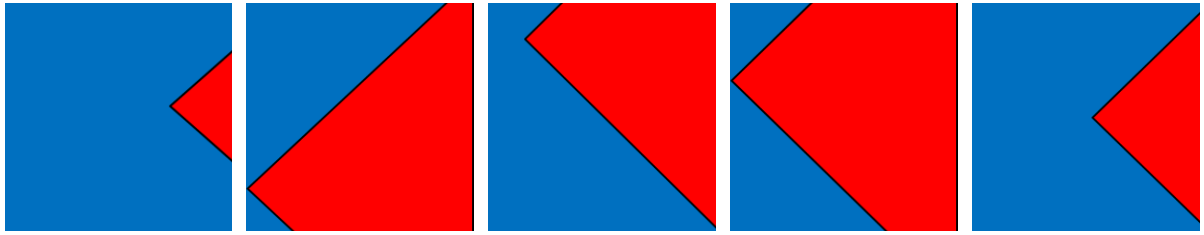
Krizhevsky et al.: ImageNet Classification with Deep CNNs, NIPS 2012

Part 1: Deep epitomic nets



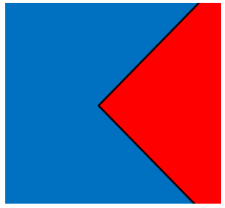
Epitomes: translation-invariant patch models

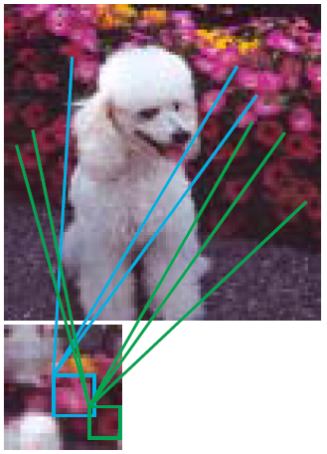
Patch Templates



Separate modeling: more data & less power per parameter

Epitomes: a lot more for just a bit more





EM-based training

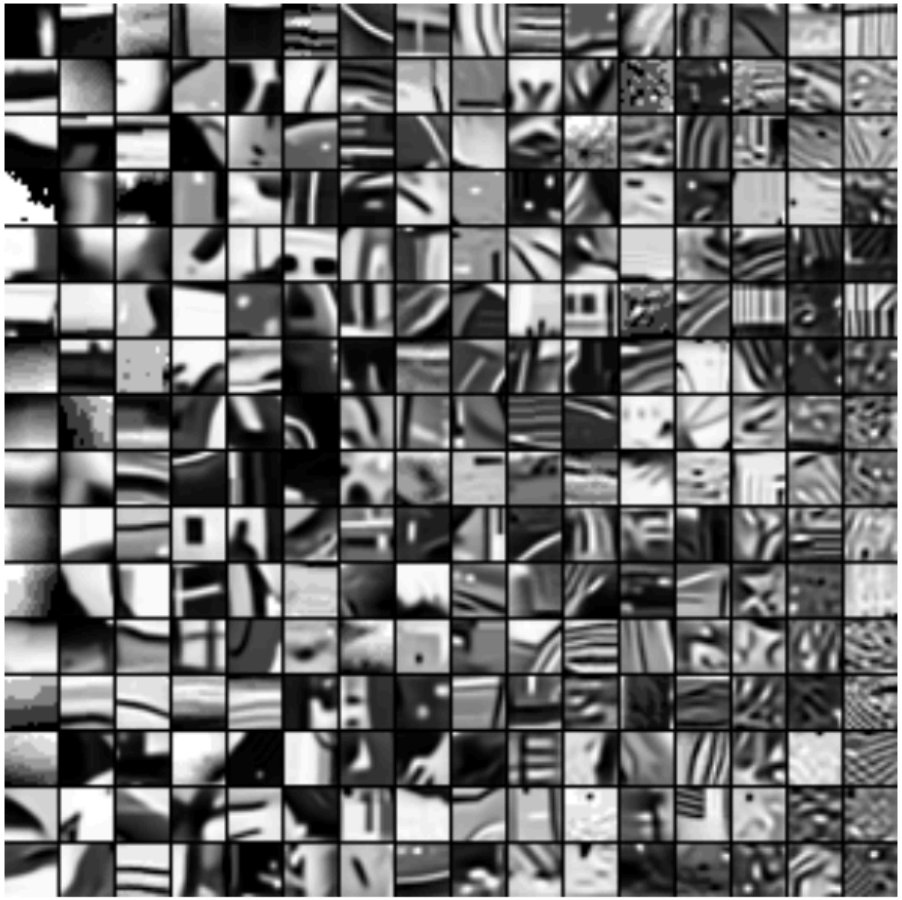
Jojic, Frey, Kannan: Epitomic analysis of appearance and shape, ICCV 2003

Benoit, Mairal, Bach, Ponce: Sparse image representation with epitomes, CVPR 2011

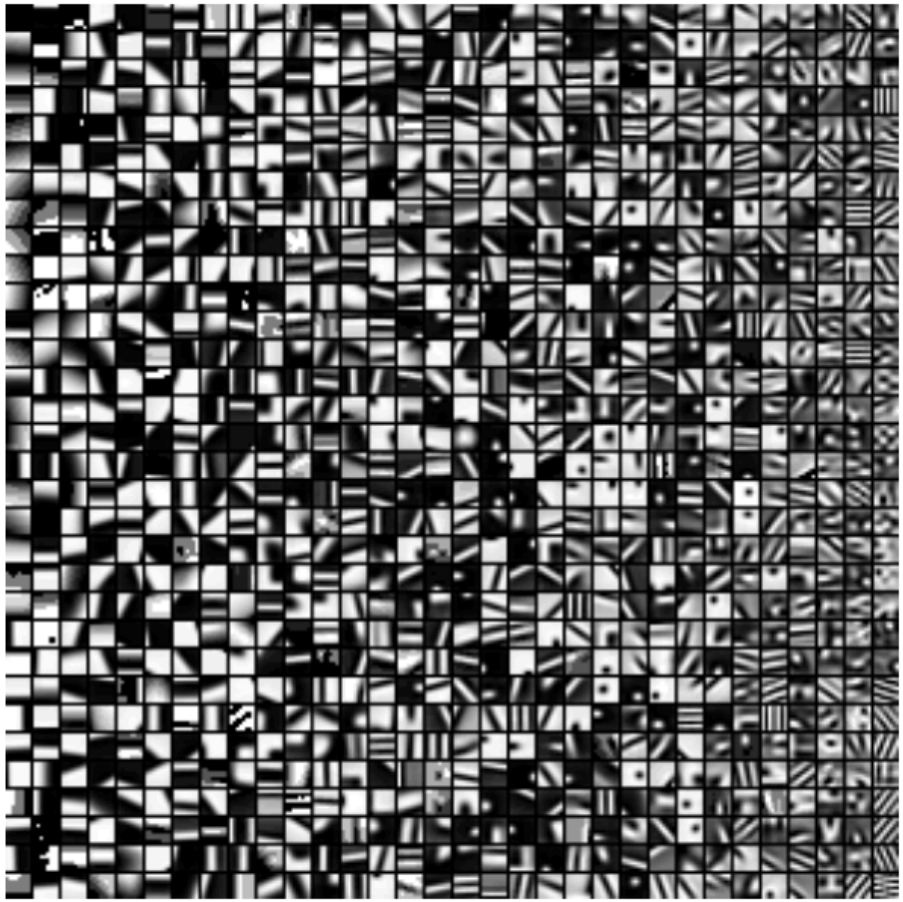
Grosse, Raina, Kwong, Ng: Shift-invariant sparse coding, UAI 2007

Mini-epitomes for image classification

Dictionary of mini-epitomes



Dictionary of patches (K-means)

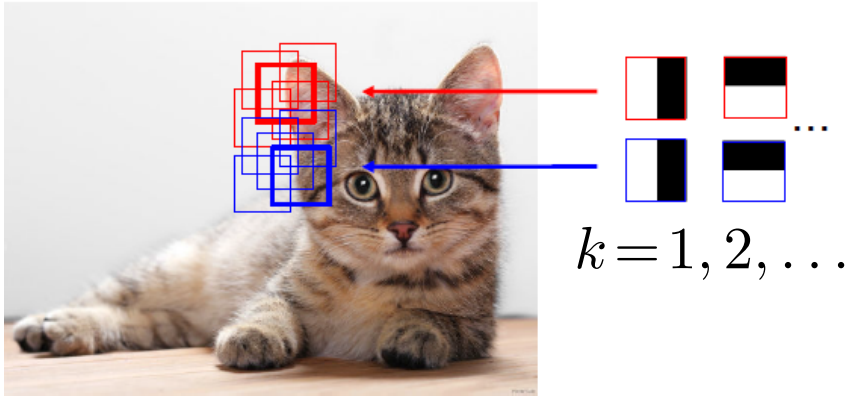


Gains in (flat) BoW classification

Papandreou, Chen, Yuille: Modeling Image Patches with a Dictionary of Mini-Epitomes, CVPR14

From flat to deep: Epitomic convolution

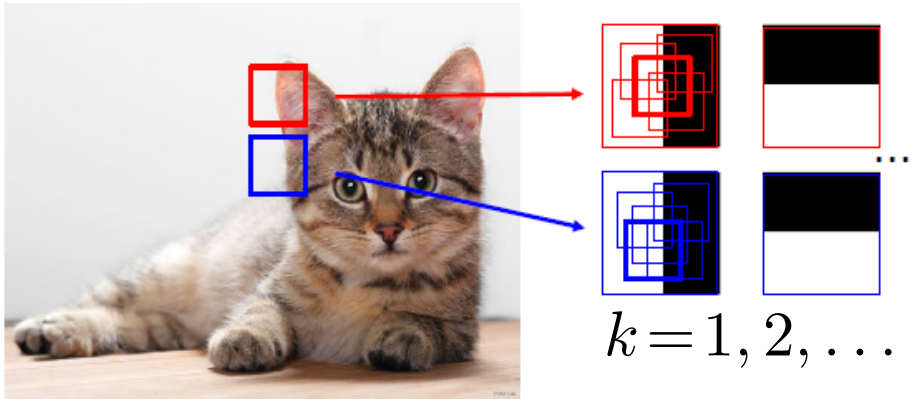
Max-Pooling



Max over image positions

$$y_{i,k} = \max_{p \in \mathcal{N}_{image}} \mathbf{x}_{i+p}^T \mathbf{w}_k$$

Epitomic Convolution

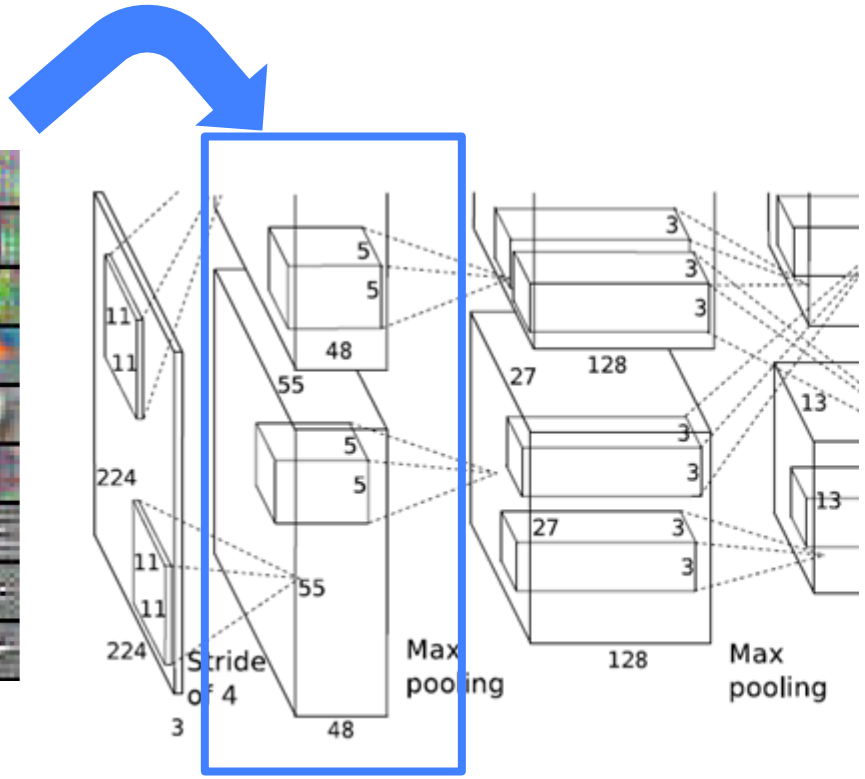


Max over epitome positions

$$y_{i,k} = \max_{p \in \mathcal{N}_{epitome}} \mathbf{x}_i^T \mathbf{w}_{k,p}$$

Deep Epitomic Convolutional Nets

Epitomic convolution

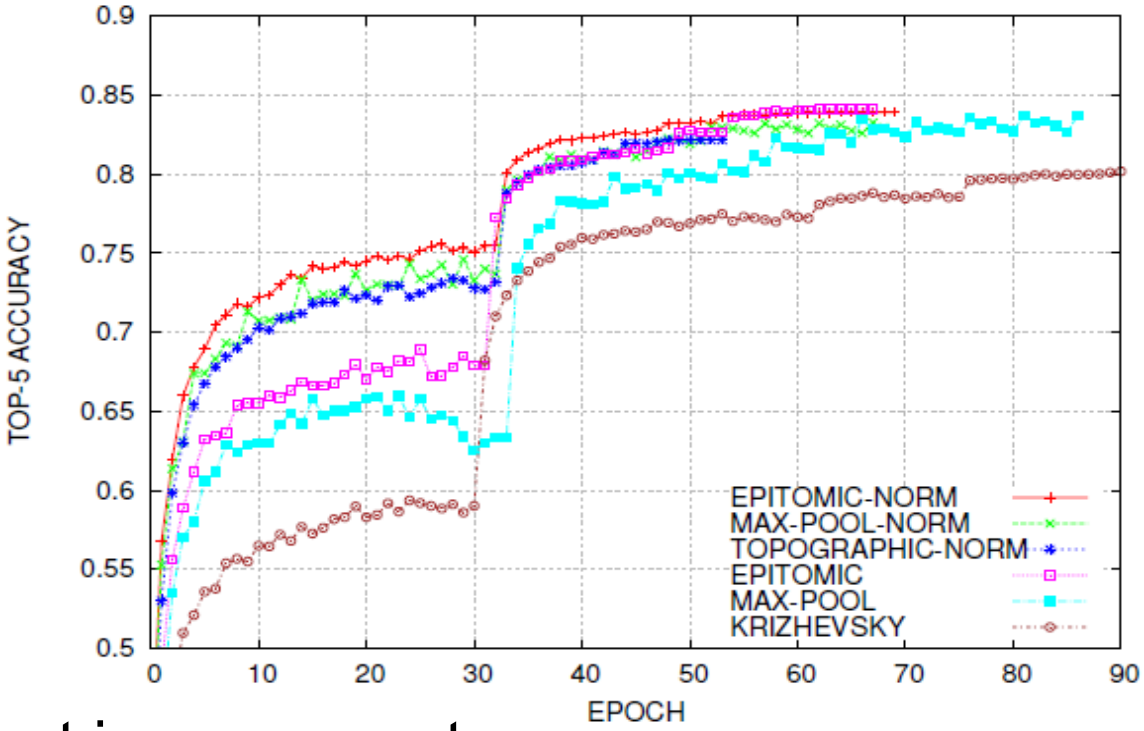


Convolution + max-pooling

Supervised dictionary learning by back-propagation

Deep Epitomic Convolutional Nets

Parameter sharing: faster and more reliable model learning



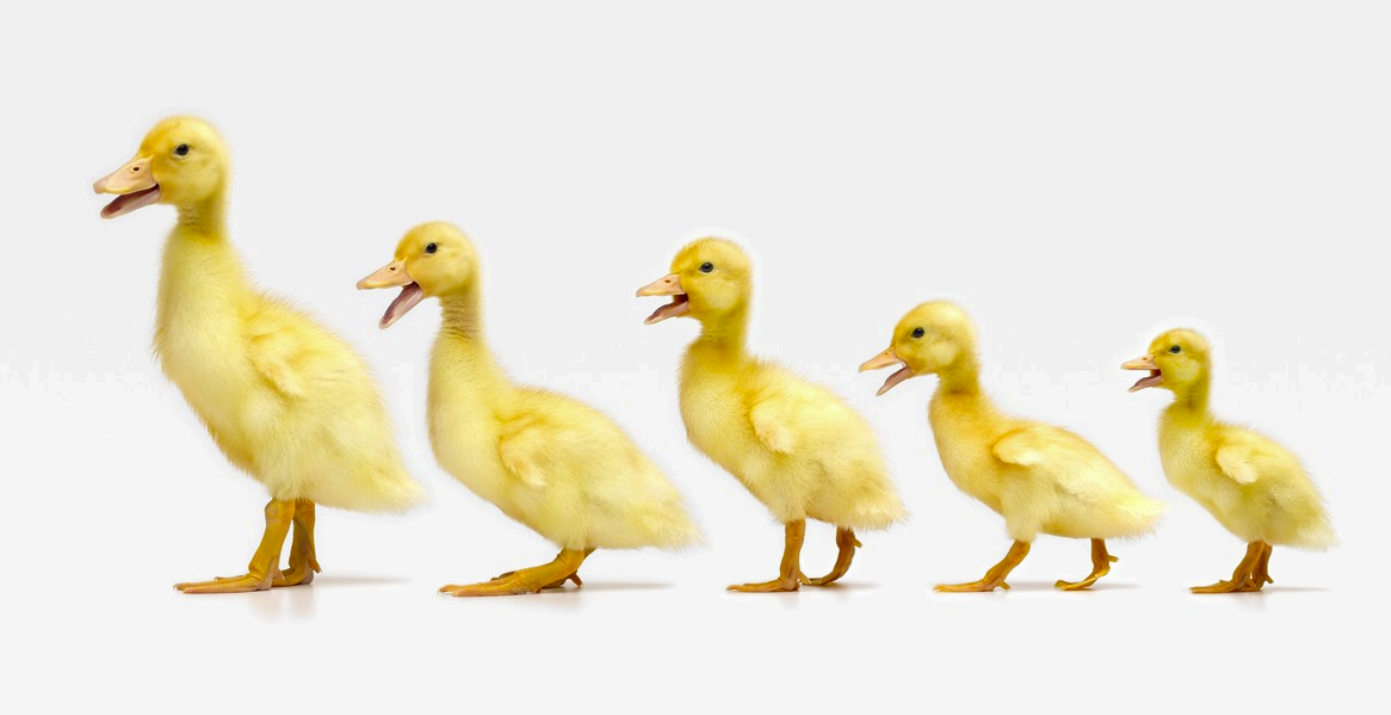
Consistent improvements

(0) Baseline: max-pooled net **13.0%**

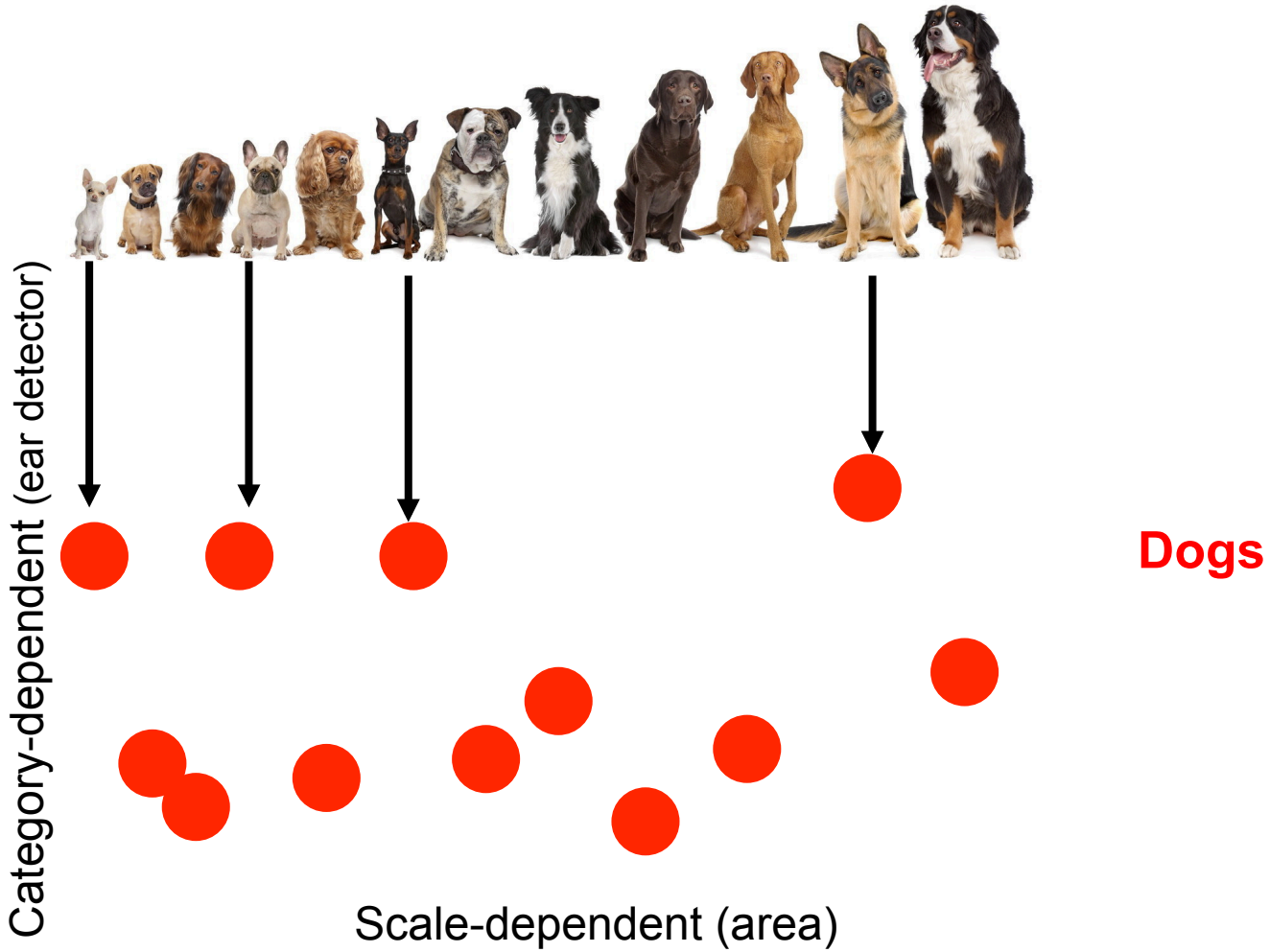
(1) epitomic DCNN **11.9%**

~1% gain

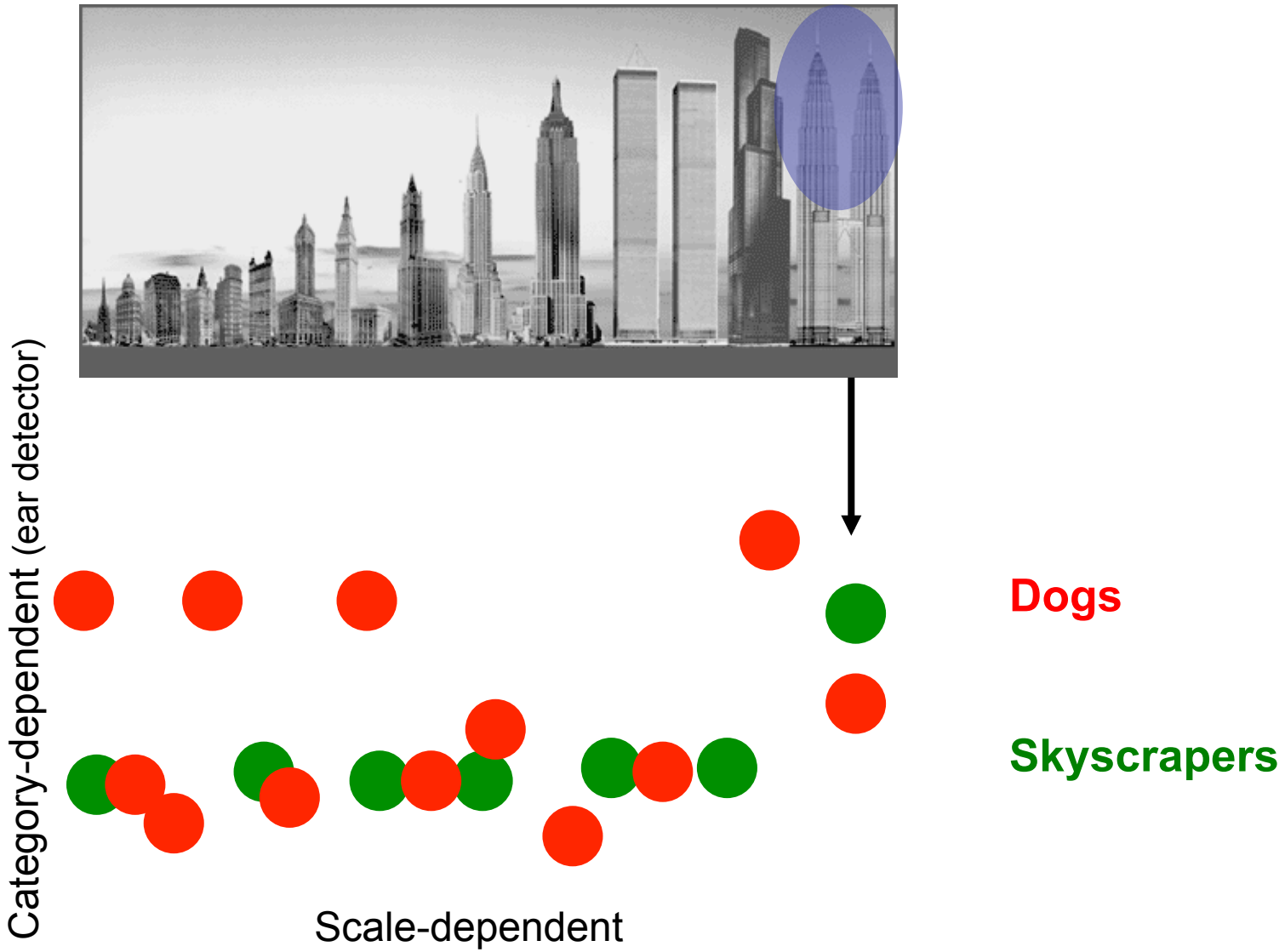
Part 2: Global scaling and translation



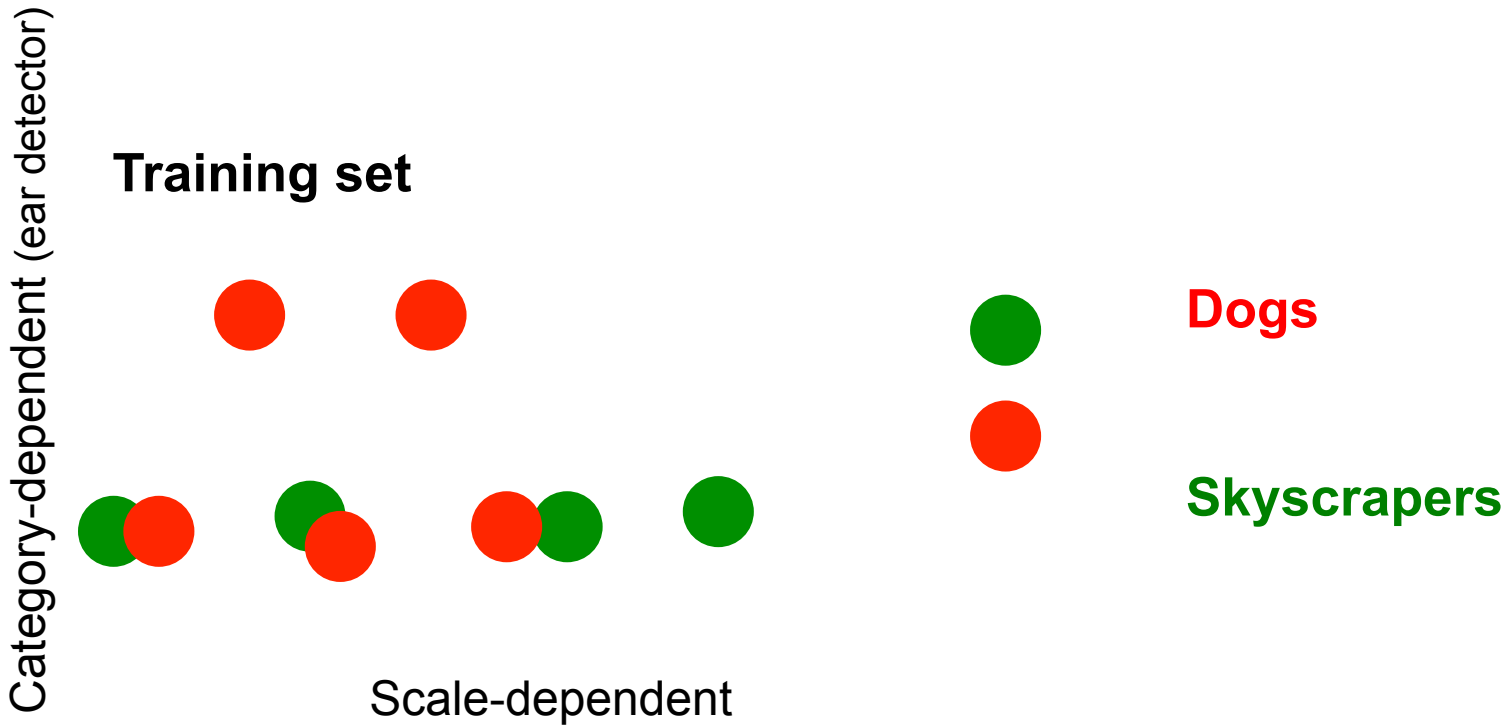
Scale Invariance challenge



Scale Invariance challenge

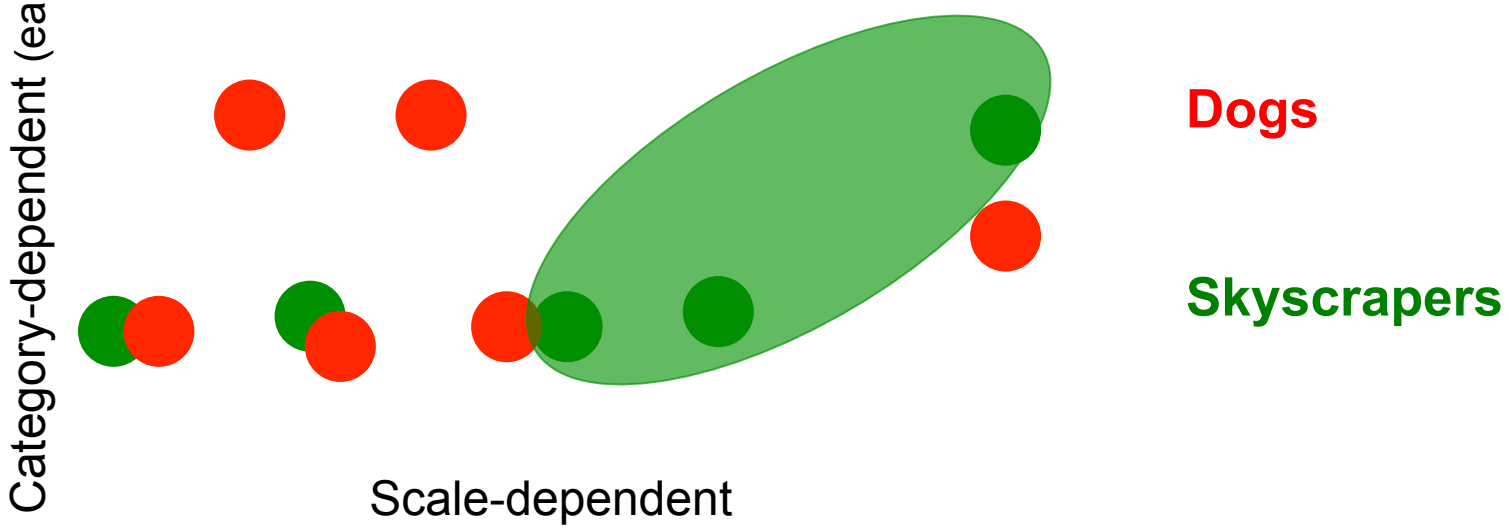


Scale Invariance challenge



Scale Invariance challenge

Rule: Large skyscrapers have ears, large dogs don't



Scale Invariant classification

Category-dependent

MIL: End-to-end training!

Scale-dependent

feature $x \rightarrow \{x_{s_1}, \ldots, x_{s_K}\}$ 'bag' of features

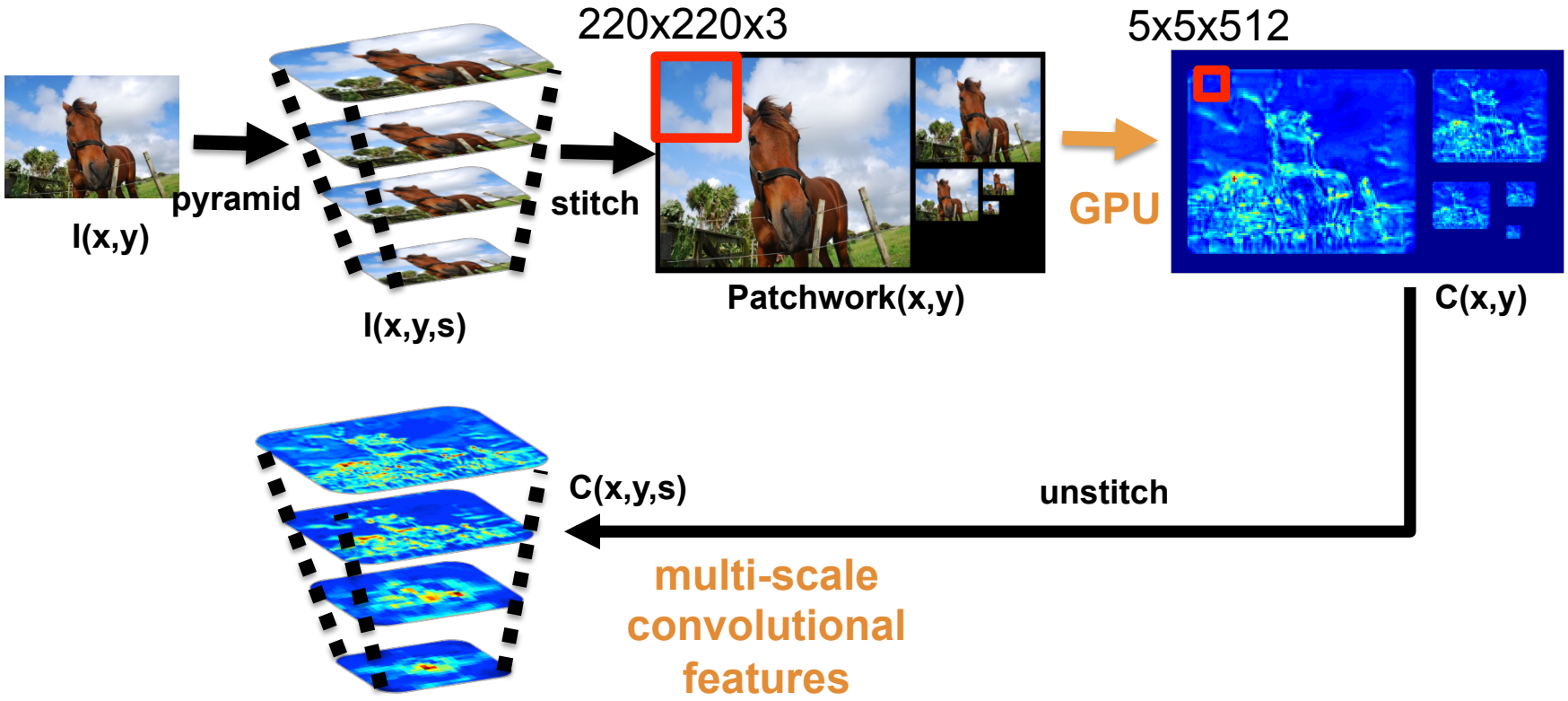
$$F(x) \rightarrow \{F(x_{s_1}), \ldots, F(x_{s_K})\}$$

$$F'(x) = \frac{1}{K}\sum_{k=1}^{K} F(x_{s_k})$$

This work: $F'(x) = \max_k F(x_{s_k})$

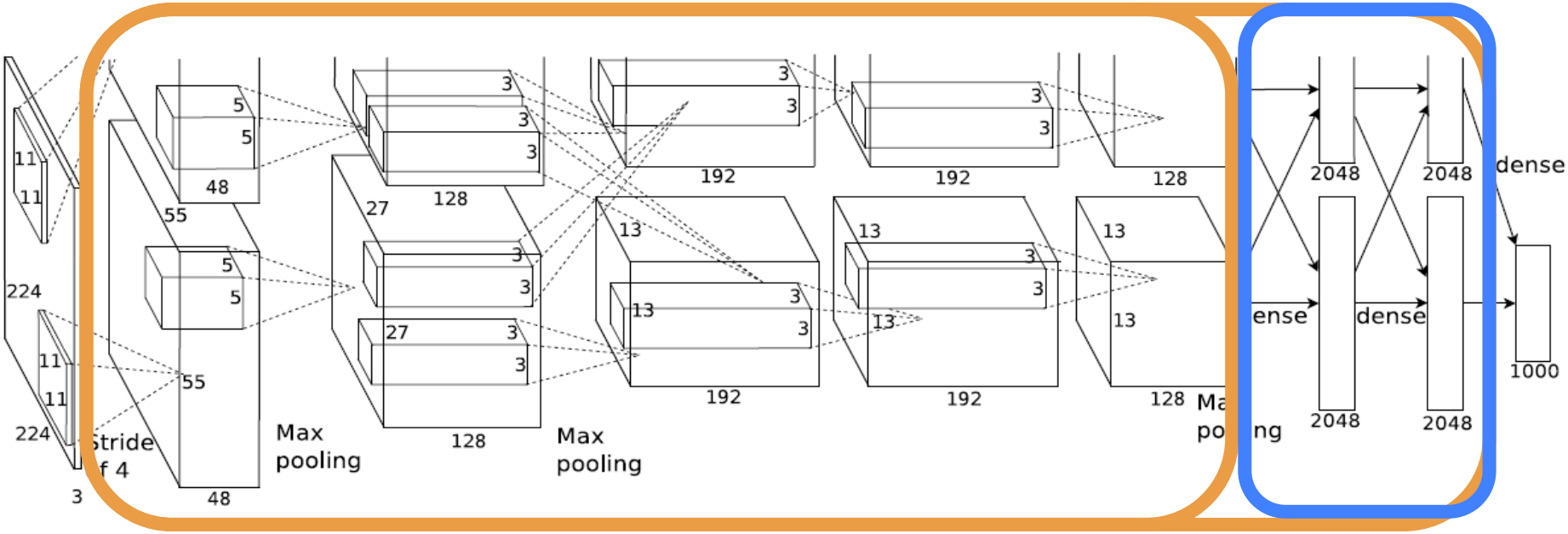
A. Howard. Some improvements on deep convolutional neural network based image classification, 2013.
K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.
T. Dietterich et al. Solving the multiple-instance problem with axis-parallel rectangles. Artificial Intelligence, 1997.

Step 1: Efficient multi-scale convolutional features



Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: Overfeat : ICLR 2014
Dubout, C., Fleuret, F.: Exact acceleration of linear object detectors. ECCV 2012
Iandola, F., Moskewicz, M., Karayev, S., Girshick, R., Darrell, T., Keutzer, K.: Densenet. arXiv 2014

Step 2: From fully connected to fully convolutional



Step 3: Global max-pooling

$$G_c = \max_{x,y} F_c(x,y)$$

learned class-specific bias

Consistent, explicit position and scale search during training and testing

For free: argmax yields 48% localization error

(0) Baseline: max-pooled net	(1) epitomic DCNN	(2) epitomic DCNN + search	Fusion (1)+(2)
13.0%	11.9%	10.56%	10.22%
	~1% gain	~1.5% gain	

How about sliding window detectors?

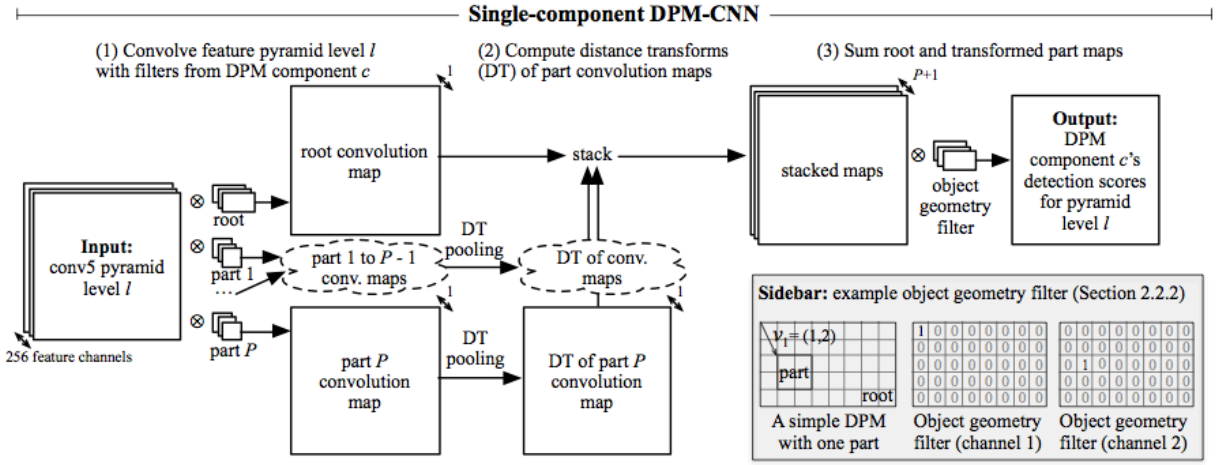


Figure 2. **CNN equivalent to a single-component DPM.** DPM can be written as an equivalent CNN by unrolling the DPM detection algorithm into a network. We present the construction for a single-component DPM-CNN here and then show how several of these CNNs can be composed into a multi-component DPM-CNN using a maxout layer (Figure 3). A single-component DPM-CNN operates on a feature pyramid level. (1) The pyramid level is convolved with the root filter and P part filters, yielding $P + 1$ convolution maps. (2) The part convolution maps are then processed with a distance transform, which we show is a generalization of max pooling. (3) The root convolution map and the DT transformed part convolution maps are stacked into a single feature map with $P + 1$ channels and then convolved with a sparse object geometry filter (see sidebar diagram and Section 2.2.2). The output is a single-channel score map for the DPM component.

Ross Girshick, Forrest Iandola, Trevor Darrell, Jitendra Malik, Deformable Part Models are Convolutional Neural Networks, http://arxiv.org/abs/1409.5403

P.-A. Savalle, S. Tsogkas, G. Papandreou and I. Kokkinos, Deformable Part Models with CNN Features, 3rd Parts and Attributes Workshop, in conjunction with ECCV 2014.

Substantially worse than RCNN

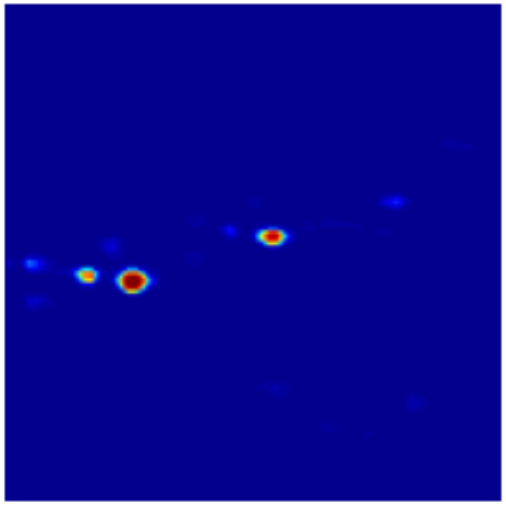
Our guess: because of ***not*** warping images

Explicit search over aspect ratio, scale & position

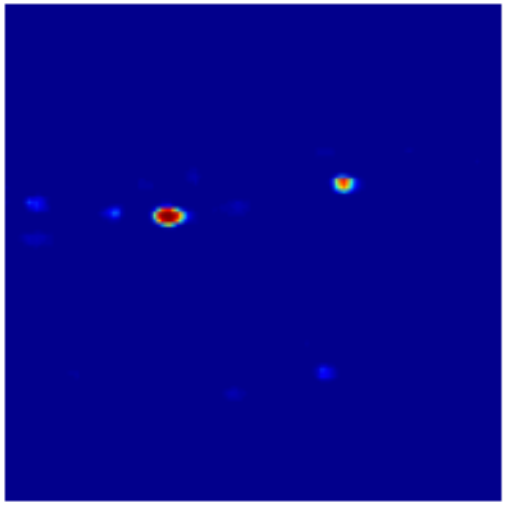
Input image

Patchwork, $\alpha = 0.33$

Patchwork, $\alpha = 0.57$



'car' score, $\alpha = 0.33$

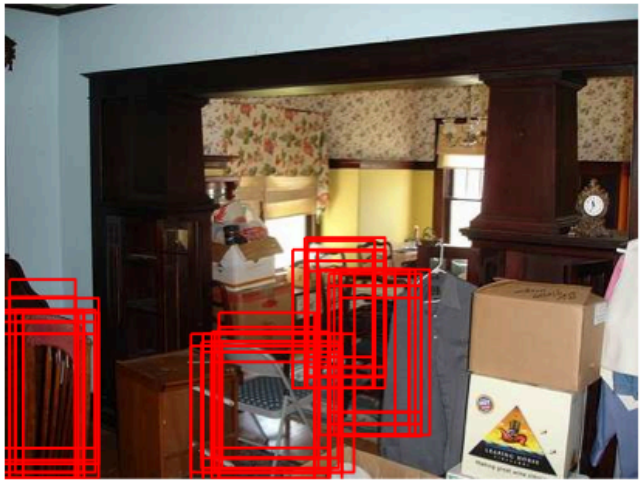


'car' score, $\alpha = 0.57$

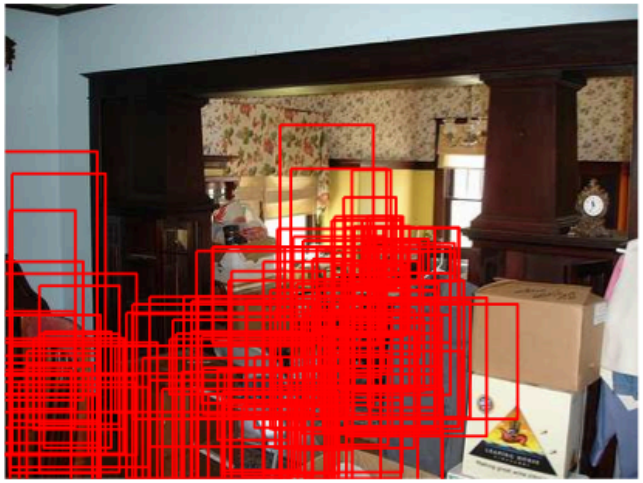
Explicit search over aspect ratio, scale & position

Figure 6. Patchwork images (even rows) and car detector scores (odd rows) used for explicit position, scale, and aspect ratio ('*alpha*') search. We observe that the score is maximized at patchwork positions (i.e. image positions, scale, and aspect ratio combinations) corresponding to square-shaped cars.

Training for localization



a



b

Our
positive & negatives

c

d

RCNN
positive & negatives

RCNN is limited by segmentation front-end

Performance

VOC 2007 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
Our work (VGG)	64.4	72.1	54.6	40.4	46.5	66.2	72.9	58.2	31.8	69.8	31.8	59.3	71.1	68.3	64.7	31.0	55.0	49.8	55.3	64.4	56.4
RCNN7 [9] (VGG)	71.6	73.5	58.1	42.2	39.4	70.7	76.0	74.5	38.7	71.0	56.9	74.5	67.9	69.6	59.3	35.7	62.1	64.0	66.5	71.2	62.2
RCNN7 [9] (UoT)	64.2	69.7	50.0	41.9	32.0	62.6	71.0	60.7	32.7	58.5	46.5	56.1	60.6	66.8	54.2	31.5	52.8	48.9	57.9	64.7	54.2
DPM [37] (UoT)	49.3	69.5	31.9	28.7	40.4	61.5	61.5	41.5	25.5	44.5	47.8	32.0	67.5	61.8	46.7	25.9	40.5	46.0	57.1	58.2	46.9

Table 5. Detection average precision (%) on the PASCAL VOC 2007 test set, using the proposed CNN sliding window detector that performs explicit position, scale, and aspect ratio search. We compare to the RCNN architecture of [9] and the end-to-end trained DPMs of [37]. In parenthesis we indicate the DCNN used for detection: UoT is the University of Toronto DCNN [18] and VGG is the DCNN of Oxford's Visual Geometry Group [33].

1st row: us – sliding windows, 56.4 mAP (5-6 seconds)

2nd row: RCNN + VGG network, 62.2 mAP (60 seconds)

3rd row: RCNN + AlexNet, 54.2 mAP (10 seconds)

4th row: sliding windows – 46.9

56.4 mAP: first shot

-no hinge loss

-no hard negative mining

-smaller (100x100) inputs, smaller network

lots of room for improvement!

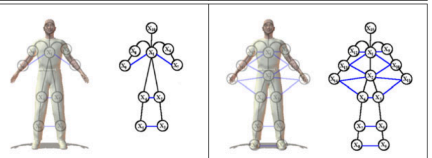
G. Papandreou, I. Kokkinos and P. A. Savalle, Untangling Local and Global Deformations in Deep Convolutional Networks for Image Classification and Sliding Window Detection, arXiv:1412.0296, 2014

Internship opportunities

6-9 months, with financial support

Possibility of extension to PhD

1) Deformable Part Models for 3D/6D: optimization & pose estimation

	10-part model	15-part model
Graph		
Number of nodes	10	15
Number of edges		
kinematic	9	14
interpenetration	4	13
Max node degree	7	8
Avg. node degree	2.6	3.6

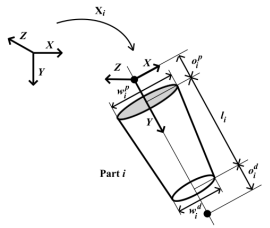
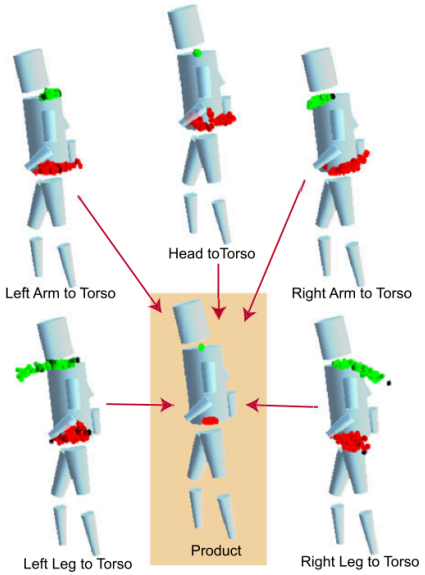
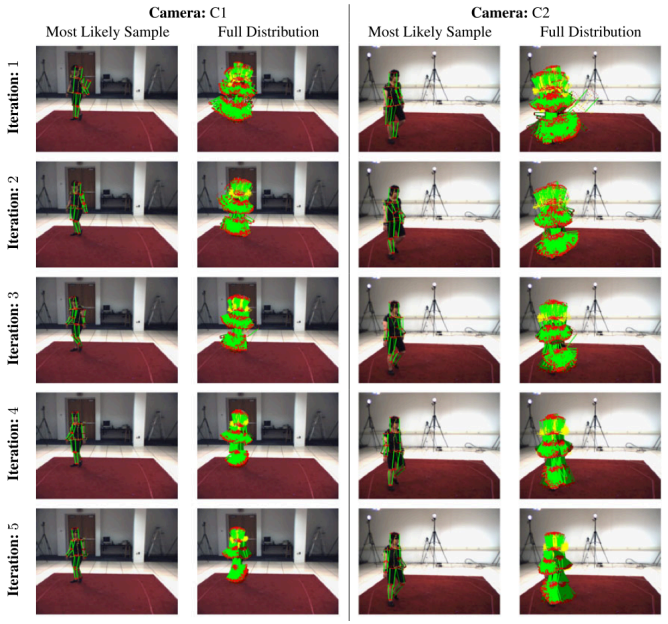


Fig. 2 Parametrization of a 3D body part





Loose-limbed People: Estimating 3D Human Pose and Motion Using Non-parametric Belief PropagationLeonid Sigal · Michael Isard · Horst Haussecker ·Michael J. Black, IJCV 2012

Alternative: 3D Medical Image Analysis with 3D DPMs

Internship opportunities

6-9 months, with financial support

Possibility of extension to PhD

2) Deep Learning for Object Detection

- Structured prediction
- Cascades
- Branch & bound
- …

3) Deep Learning and Low-level Vision

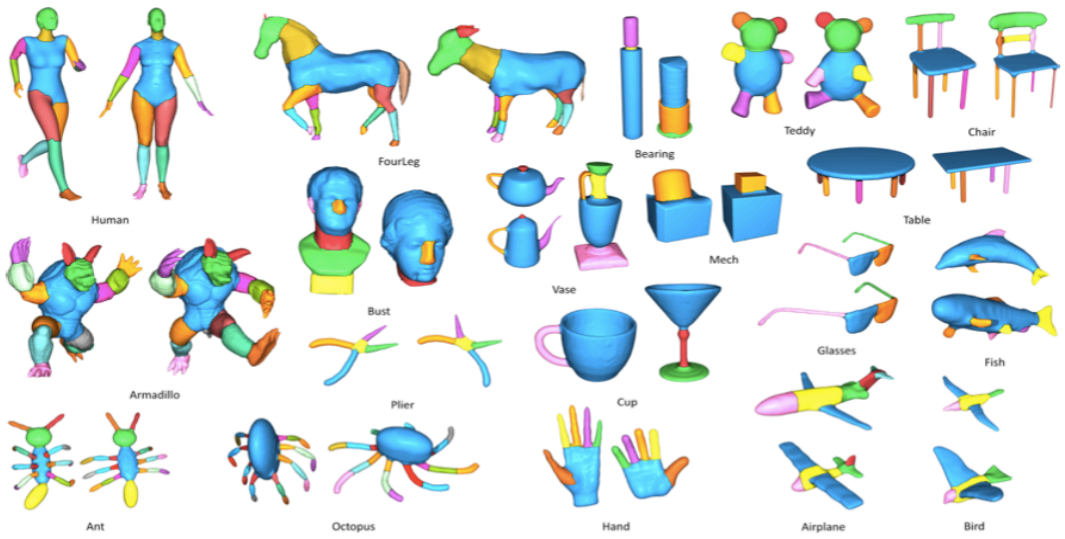
- Boundary detection
- Matching/Optical Flow
- Image Descriptors
- …

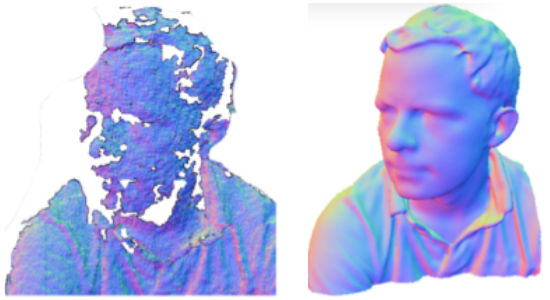
Internship opportunities

6-9 months, with financial support

Possibility of extension to PhD

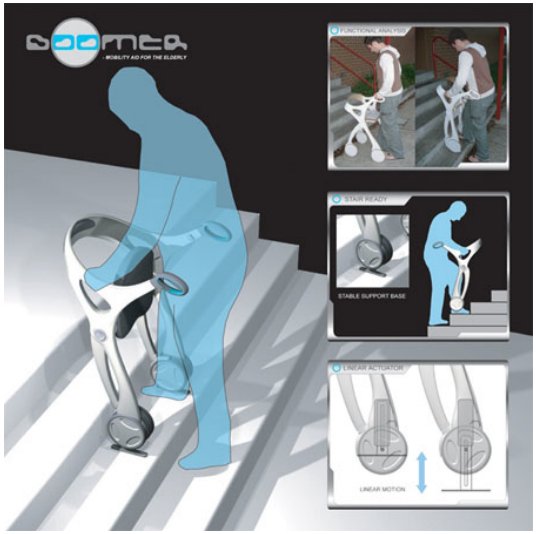
4) 3D Statistical Shape Models & RGB-D sensors





Seeral other topics possible
-talk to me/email me
-and/or check http://cvn.ecp.fr/personnel/iasonas/publications.html

Depth-based 3D pose estimation (2013-2016)



Project MOBOT: Robotic assistants for the elderly

Partners: TU Munich (DE), ICCS (GR), U Heidelberg (DE)

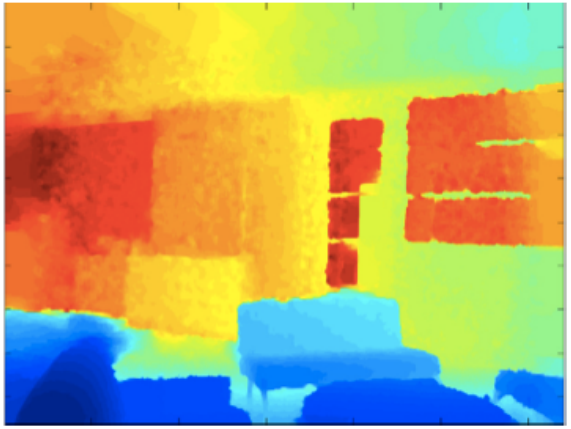
Rapid pose estimation using depth data (DTBB, Depth features, pose estimation)
Action recognition from multiple cues (depth, motion, intensity, detectors)

Fully funded for 3 years, starting as soon as February

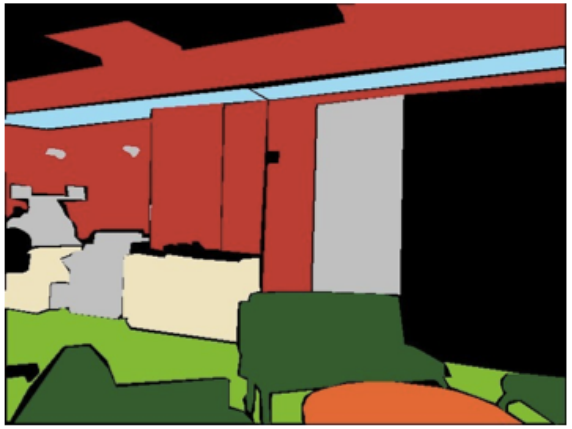
Depth-based 3D object recognition (2013-2016)

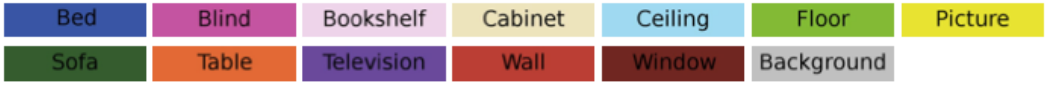
RGB input

Depth input



Desired symbolic output



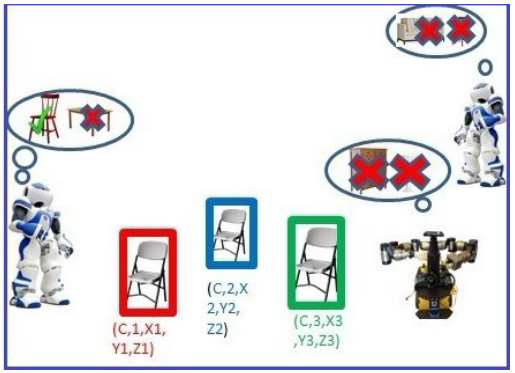


Project RECONFIG: Reconfigurable, heterogeneous multi-agent systems

Partners: KTH (SE), U Aalto (FI) , NTUA (GR)

Multi-view object recognition using depth data
Segmentation, registration, tracking of object surfaces

Fully funded for 3 years, starting as soon as February



Daily Living Activities assistance (2015-2018)

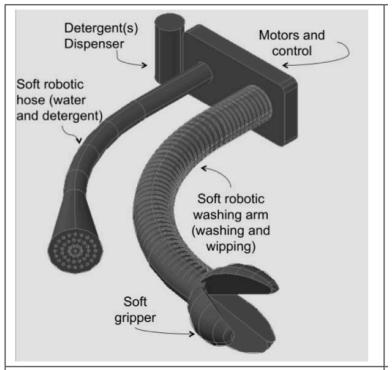


Figure 1. The two robotic devices: (i) robotic hose and (i) robotic washing arm. The length of the robotic arm can vary depending the shower space and the ergonomic design. The concept is modular, self contained and easily interfaced with a conventional tab shower infrastructure.

Project ISUPPORT: Robotic assistants for the elderly

REFERENCES

Convolutional Nets

– LeCun, Bottou, Bengio and Haffner: Gradient-Based Learning Applied to Document Recognition, Proceedings of the IEEE, 86(11):2278-2324, November 1998

- Krizhevsky, Sutskever, Hinton "ImageNet Classification with deep convolutional neural networks" NIPS 2012

– Jarrett, Kavukcuoglu, Ranzato, LeCun: What is the Best Multi-Stage Architecture for Object Recognition?, Proc. International Conference on Computer Vision (ICCV'09), IEEE, 2009

- Kavukcuoglu, Sermanet, Boureau, Gregor, Mathieu, LeCun: Learning Convolutional Feature Hierachies for Visual Recognition, Advances in Neural Information Processing Systems (NIPS 2010), 23, 2010

– see yann.lecun.com/exdb/publis for references on many different kinds of convnets.

– see http://www.cmap.polytechnique.fr/scattering/ for scattering networks (similar to convnets but with less learning and stronger mathematical foundations)

– see http://www.idsia.ch/~juergen/ for other references to ConvNets and LSTMs.

REFERENCES

Applications of Convolutional Nets

– Farabet, Couprie, Najman, LeCun. Scene Parsing with Multiscale Feature Learning, Purity Trees, and Optimal Covers", ICML 2012

– Pierre Sermanet, Koray Kavukcuoglu, Soumith Chintala and Yann LeCun: Pedestrian Detection with Unsupervised Multi-Stage Feature Learning, CVPR 2013

- D. Ciresan, A. Giusti, L. Gambardella, J. Schmidhuber. Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images. NIPS 2012

- Raia Hadsell, Pierre Sermanet, Marco Scoffier, Ayse Erkan, Koray Kavackuoglu, Urs Muller and Yann LeCun. Learning Long-Range Vision for Autonomous Off-Road Driving, Journal of Field Robotics, 26(2):120-144, 2009

– Burger, Schuler, Harmeling. Image Denoisng: Can Plain Neural Networks Compete with BM3D?, CVPR 2012

– Hadsell, Chopra, LeCun. Dimensionality reduction by learning an invariant mapping, CVPR 2006

– Bergstra et al. Making a science of model search: hyperparameter optimization in hundred of dimensions for vision architectures, ICML 2013

REFERENCES

Deep Learning in general

– deep learning tutorial @ CVPR 2014 https://sites.google.com/site/deeplearningcvpr2014/

– deep learning tutorial slides at ICML 2013: icml.cc/2013/?page_id=39

– Yoshua Bengio, Learning Deep Architectures for AI, Foundations and Trends in Machine Learning, 2(1), pp.1-127, 2009.

– LeCun, Chopra, Hadsell, Ranzato, Huang: A Tutorial on Energy-Based Learning, in Bakir, G. and Hofman, T. and Schölkopf, B. and Smola, A. and Taskar, B. (Eds), Predicting Structured Data, MIT Press, 2006

"Theory" of Deep Learning

– Mallat: Group Invariant Scattering, Comm. In Pure and Applied Math. 2012

– Pascanu, Montufar, Bengio: On the number of inference regions of DNNs with piece wise linear activations, ICLR 2014

– Pascanu, Dauphin, Ganguli, Bengio: On the saddle-point problem for non-convex optimization, arXiv 2014

- Delalleau, Bengio: Shallow vs deep Sum-Product Networks, NIPS 2011

SOFTWARE

Torch7: learning library that supports neural net training

http://www.torch.ch
http://code.cogbits.com/wiki/doku.php (tutorial with demos by C. Farabet)
https://github.com/sermanet/OverFeat

Python-based learning library (U. Montreal)

- http://deeplearning.net/software/theano/ (does automatic differentiation)

Efficient CUDA kernels for ConvNets (Krizhevsky)

– code.google.com/p/cuda-convnet

Caffe (Yangqing Jia)

– http://caffe.berkeleyvision.org

The end! ☺

Getting started

- http://www.vlfeat.org/matconvnet/