

The Unreasonable Effectiveness of Deep Learning

Yann LeCun

Facebook AI Research &
Center for Data Science, NYU

yann@cs.nyu.edu

<http://yann.lecun.com>

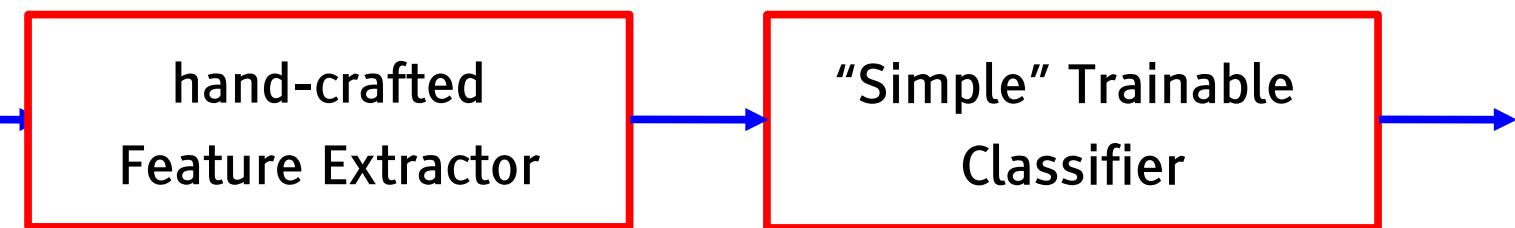


55 years of hand-crafted features

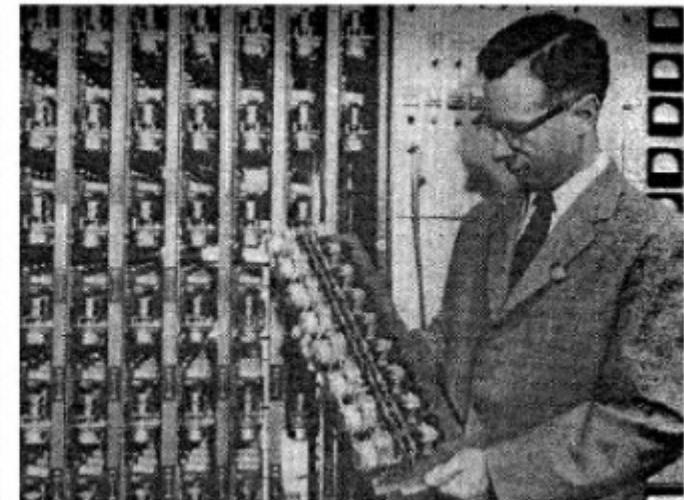
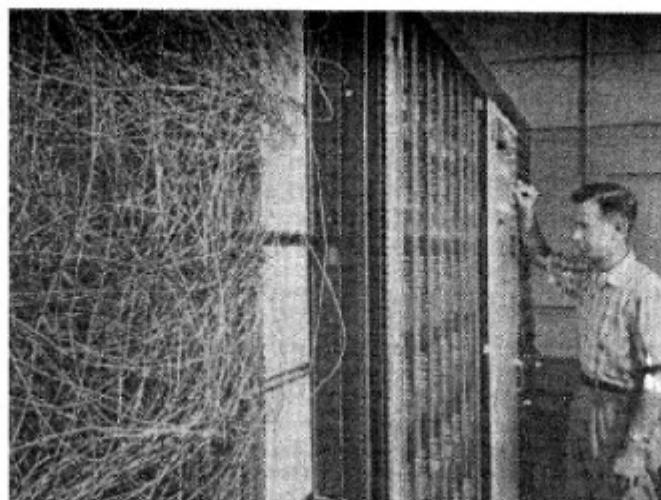
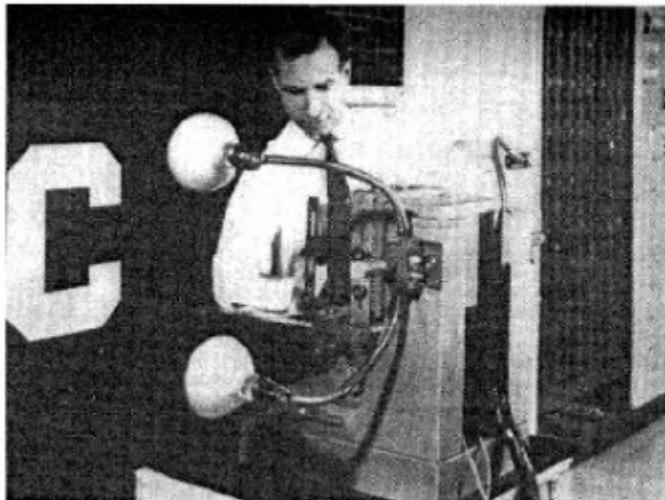
Y LeCun

■ The traditional model of pattern recognition (since the late 50's)

- ▶ Fixed/engineered features (or fixed kernel) + trainable classifier



■ Perceptron

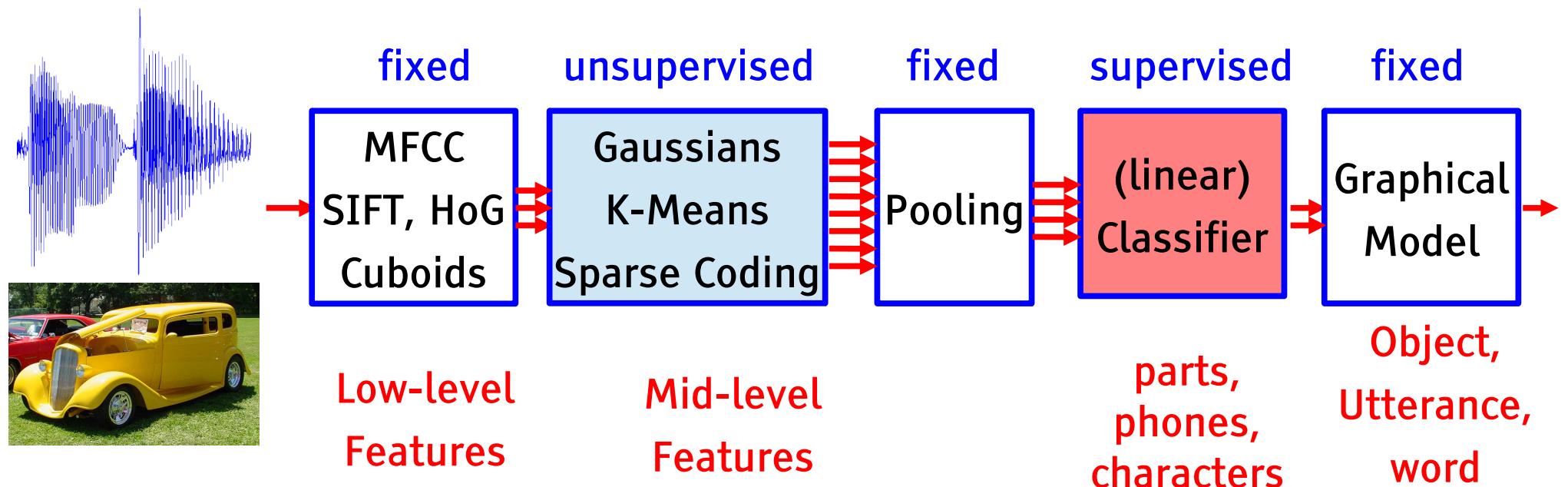


Architecture of “Classical” Recognition Systems

Y LeCun

“Classic” architecture for pattern recognition

- ▶ Speech recognition: 1990-2011
- ▶ Object Recognition: 2005-2012
- ▶ Handwriting recognition (long ago)
- ▶ Graphical model has latent variables (locations of parts)

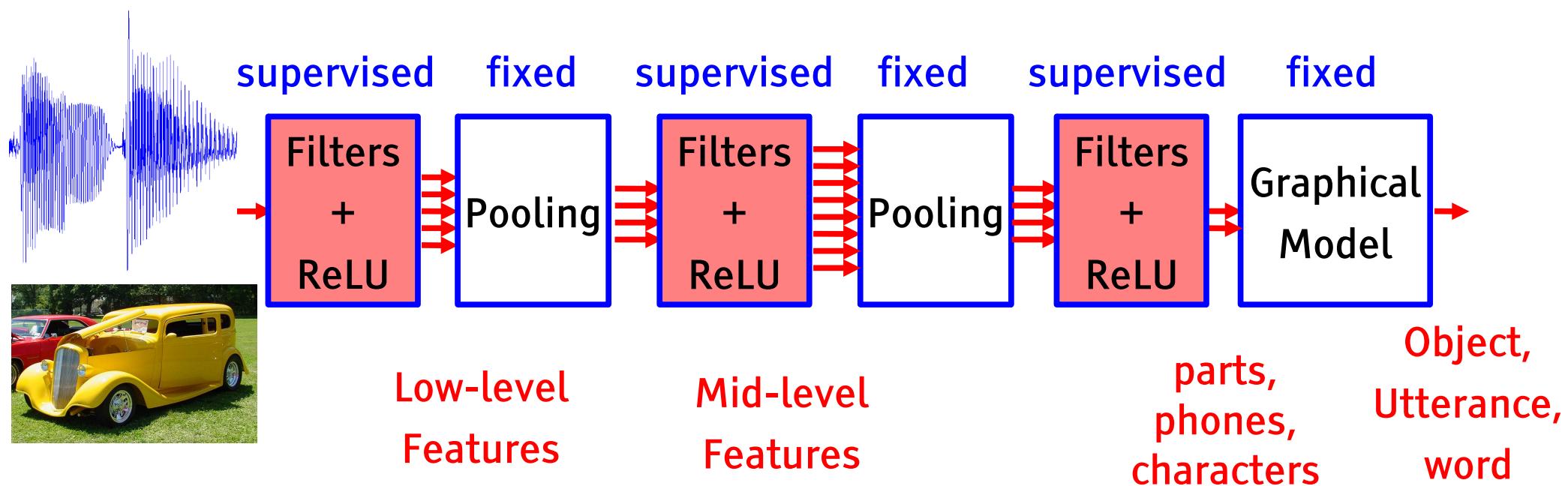


Architecture of Deep Learning-Based Recognition Systems

Y LeCun

“Deep” architecture for pattern recognition

- ▶ Speech, and Object recognition: since 2011/2012
- ▶ Handwriting recognition: since the early 1990s
- ▶ Convolutional Net with optional Graphical Model on top
- ▶ Trained purely supervised
- ▶ Graphical model has latent variables (locations of parts)

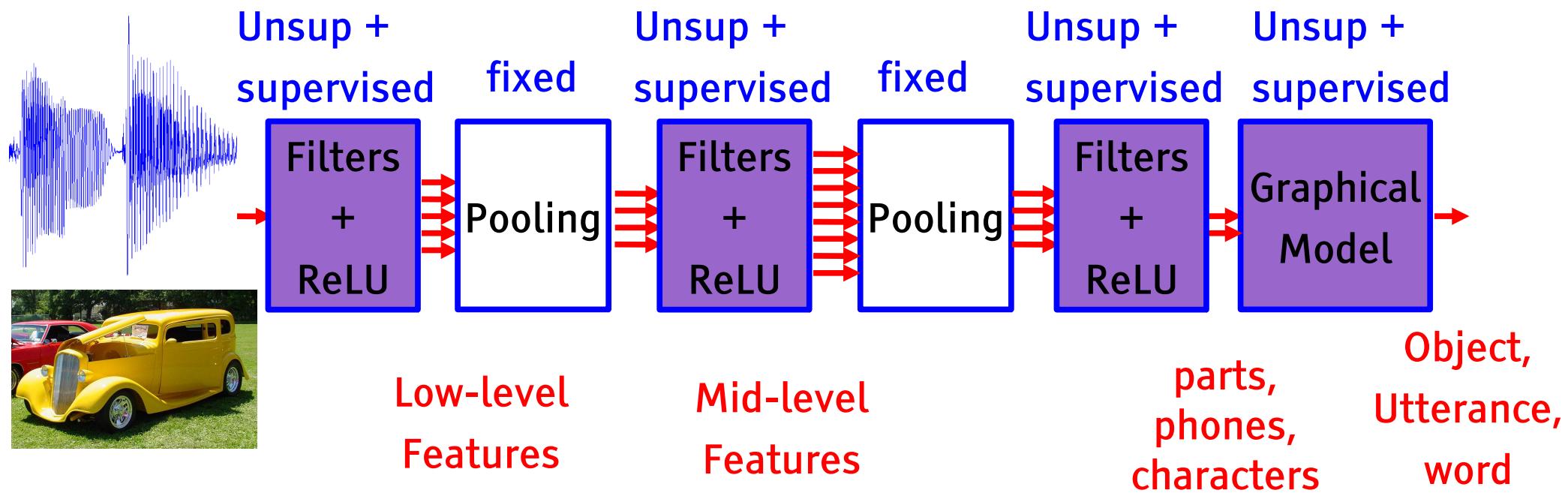


Future Systems: deep learning + structured prediction

Y LeCun

Globally-trained deep architecture

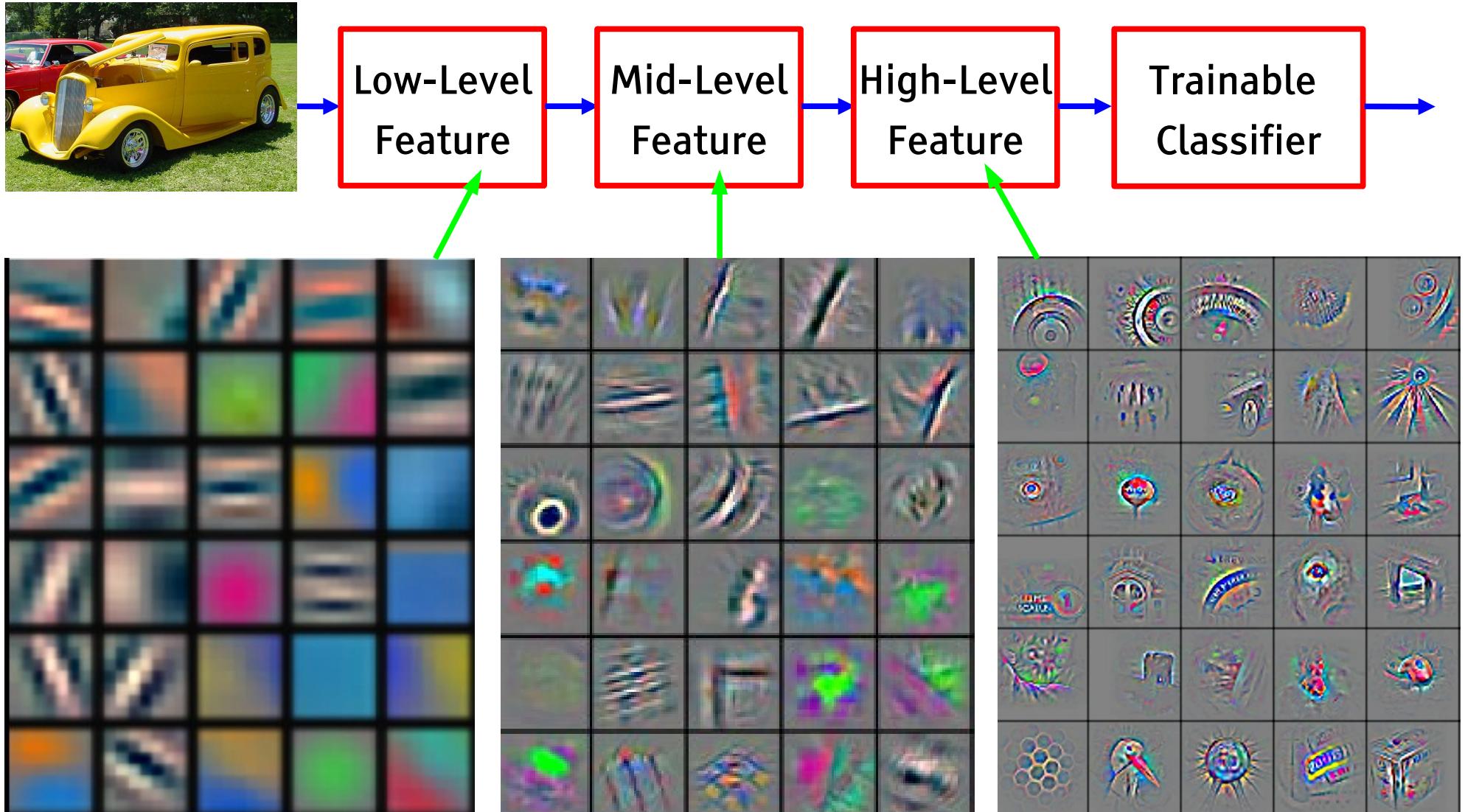
- ▶ Handwriting recognition: since the mid 1990s
- ▶ Speech Recognition: since 2011
- ▶ All the modules are trained with a combination of unsupervised and supervised learning
- ▶ **End-to-end training == deep structured prediction**



Deep Learning = Learning Hierarchical Representations

Y LeCun

■ It's deep if it has more than one stage of non-linear feature transformation



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Trainable Feature Hierarchy

■ Hierarchy of representations with increasing level of abstraction

■ Each stage is a kind of trainable feature transform

■ Image recognition

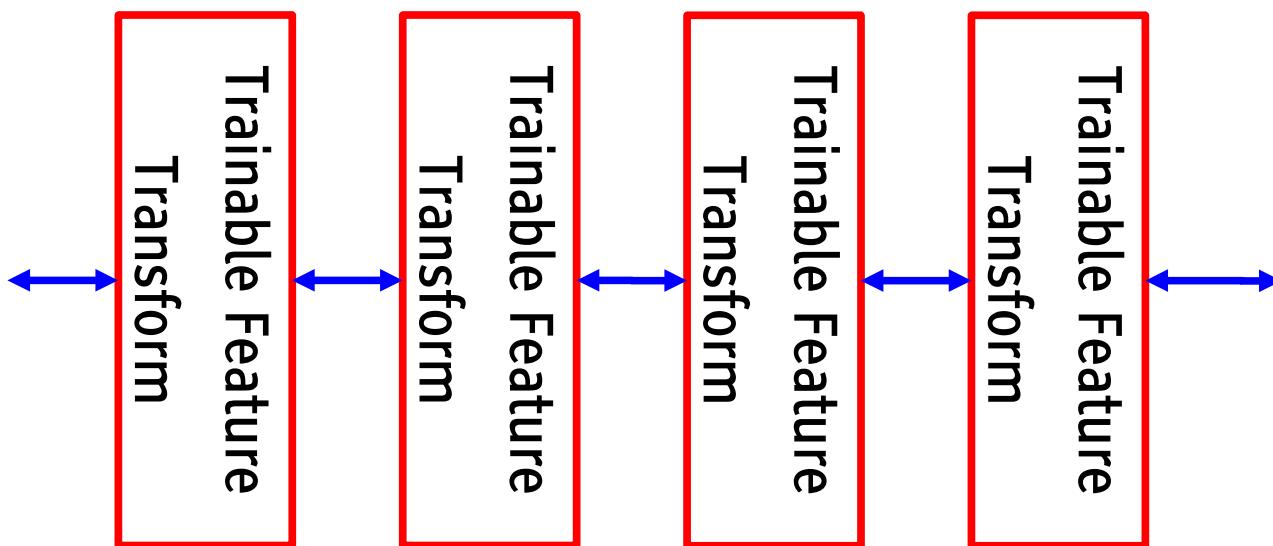
▶ Pixel → edge → texton → motif → part → object

■ Text

▶ Character → word → word group → clause → sentence → story

■ Speech

▶ Sample → spectral band → sound → ... → phone → phoneme → word



Learning Representations: a challenge for ML, CV, AI, Neuroscience, Cognitive Science...

Y LeCun

■ How do we learn representations of the perceptual world?

- ▶ How can a perceptual system build itself by looking at the world?
- ▶ How much prior structure is necessary

■ ML/AI: how do we learn features or feature hierarchies?

- ▶ What is the fundamental principle? What is the learning algorithm? What is the architecture?

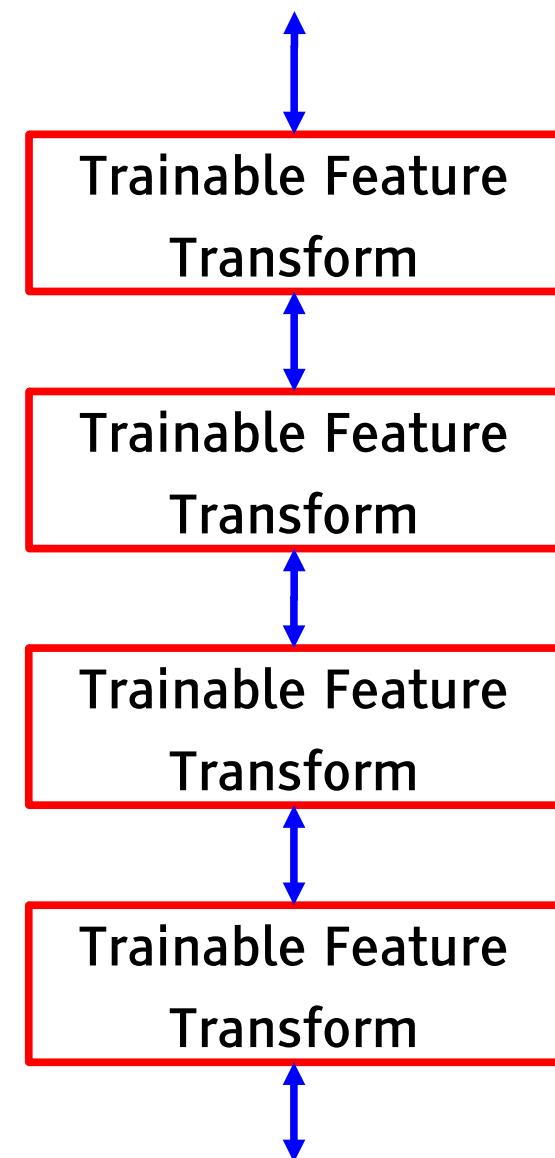
■ Neuroscience: how does the cortex learn perception?

- ▶ Does the cortex “run” a single, general learning algorithm? (or a small number of them)

■ CogSci: how does the mind learn abstract concepts on top of less abstract ones?

■ Deep Learning addresses the problem of learning hierarchical representations with a single algorithm

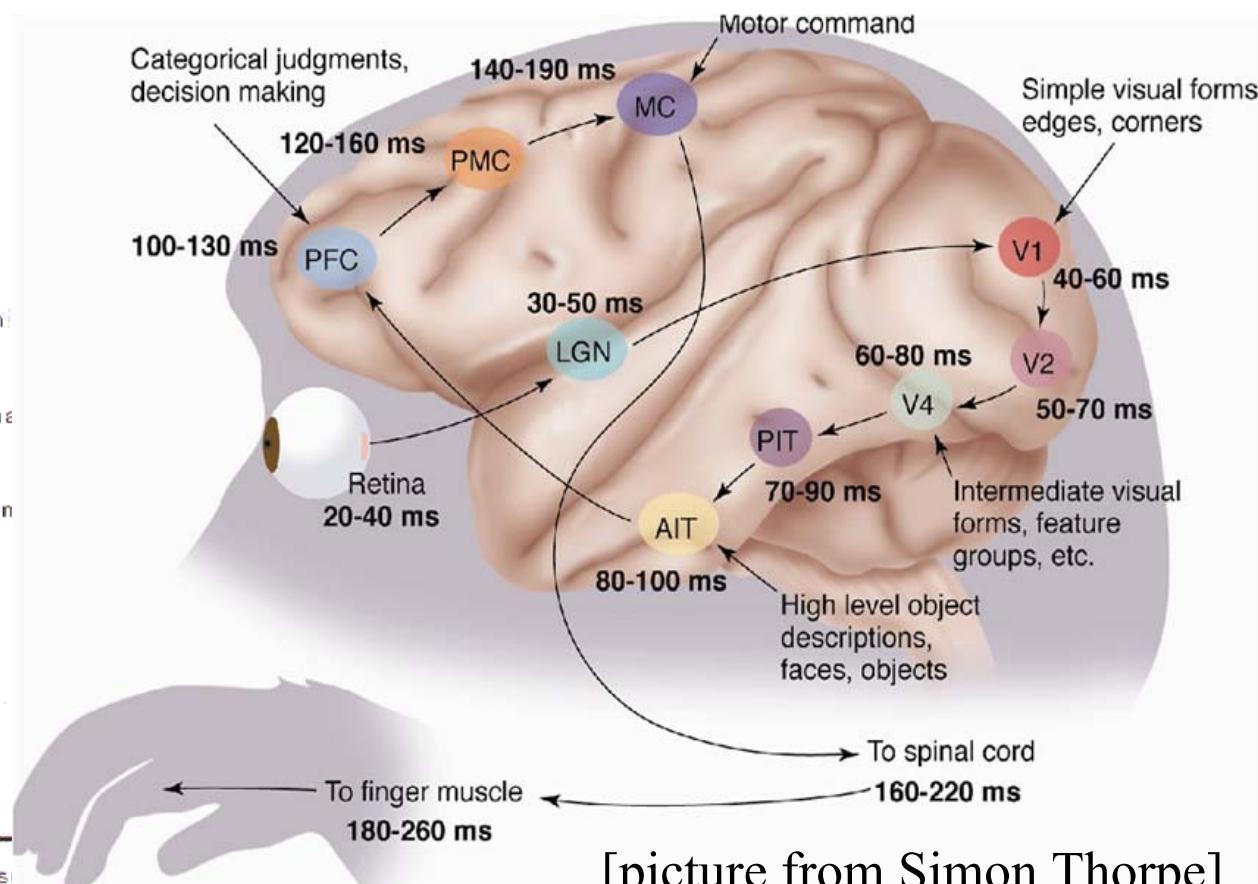
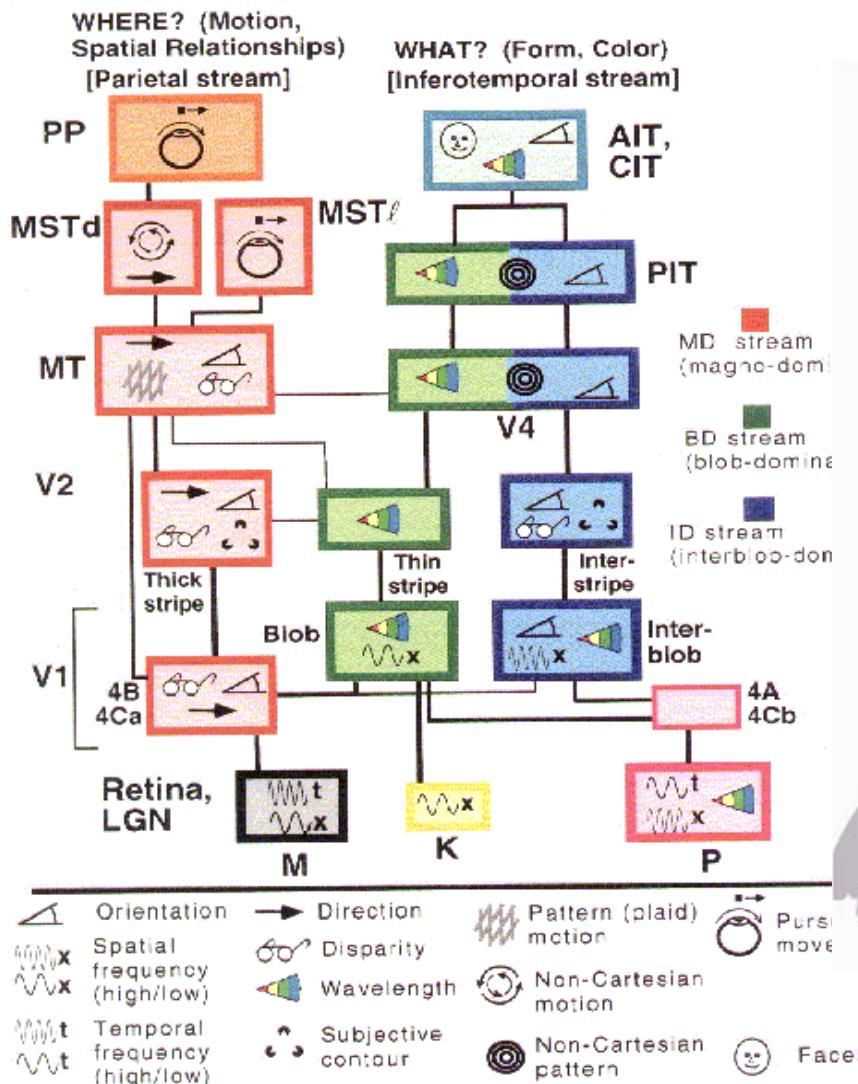
- ▶ or perhaps with a few algorithms



The Mammalian Visual Cortex is Hierarchical

Y LeCun

- The ventral (recognition) pathway in the visual cortex has multiple stages
- Retina - LGN - V1 - V2 - V4 - PIT - AIT
- Lots of intermediate representations



[picture from Simon Thorpe]

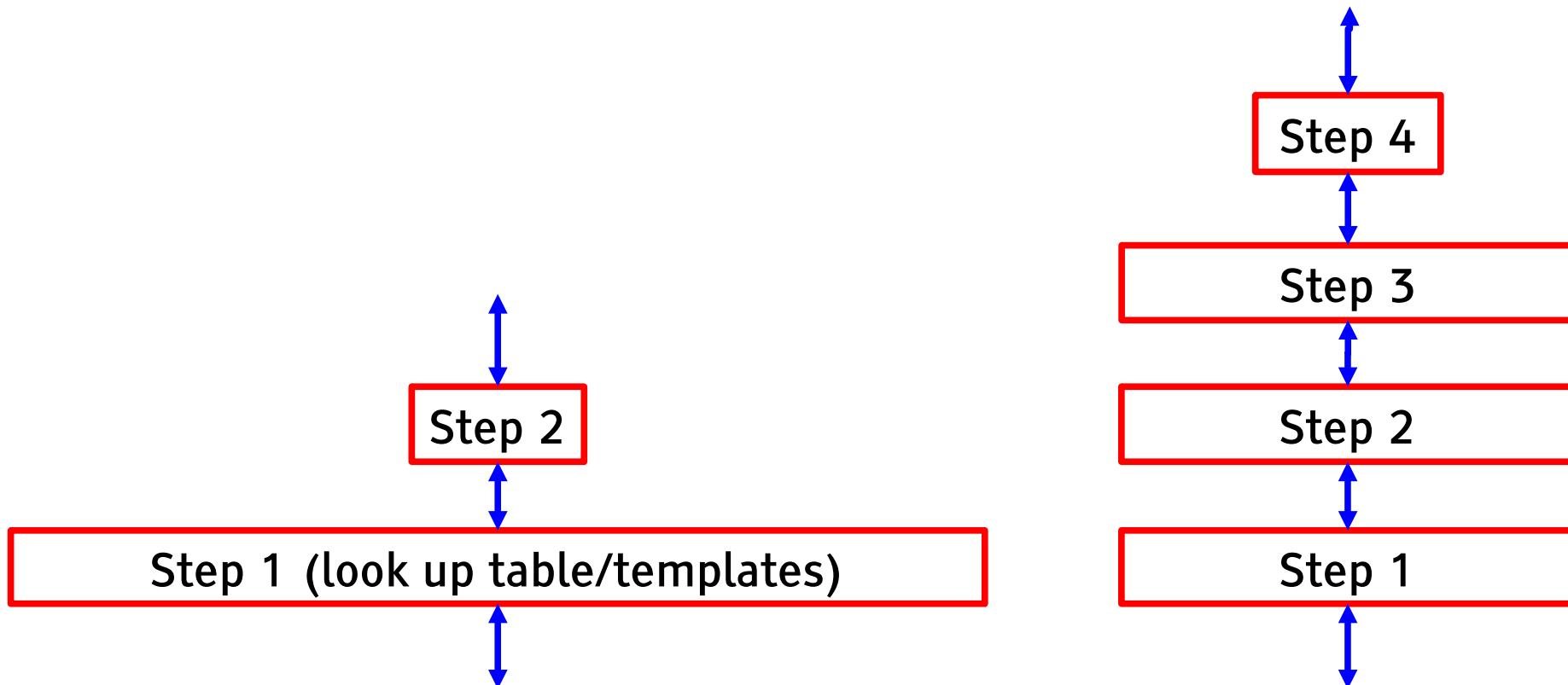
[Gallant & Van Essen]

Shallow vs Deep == lookup table vs multi-step algorithm

Y LeCun

■ “shallow & wide” vs “deep and narrow” == “more memory” vs “more time”

- ▶ Look-up table vs algorithm
- ▶ Few functions can be computed in two steps without an exponentially large lookup table
- ▶ Using more than 2 steps can reduce the “memory” by an exponential factor.



Which Models are Deep?

Y LeCun

- 2-layer models are not deep (even if you train the first layer)

▶ Because there is no feature hierarchy

- Neural nets with 1 hidden layer are not deep

- SVMs and Kernel methods are not deep

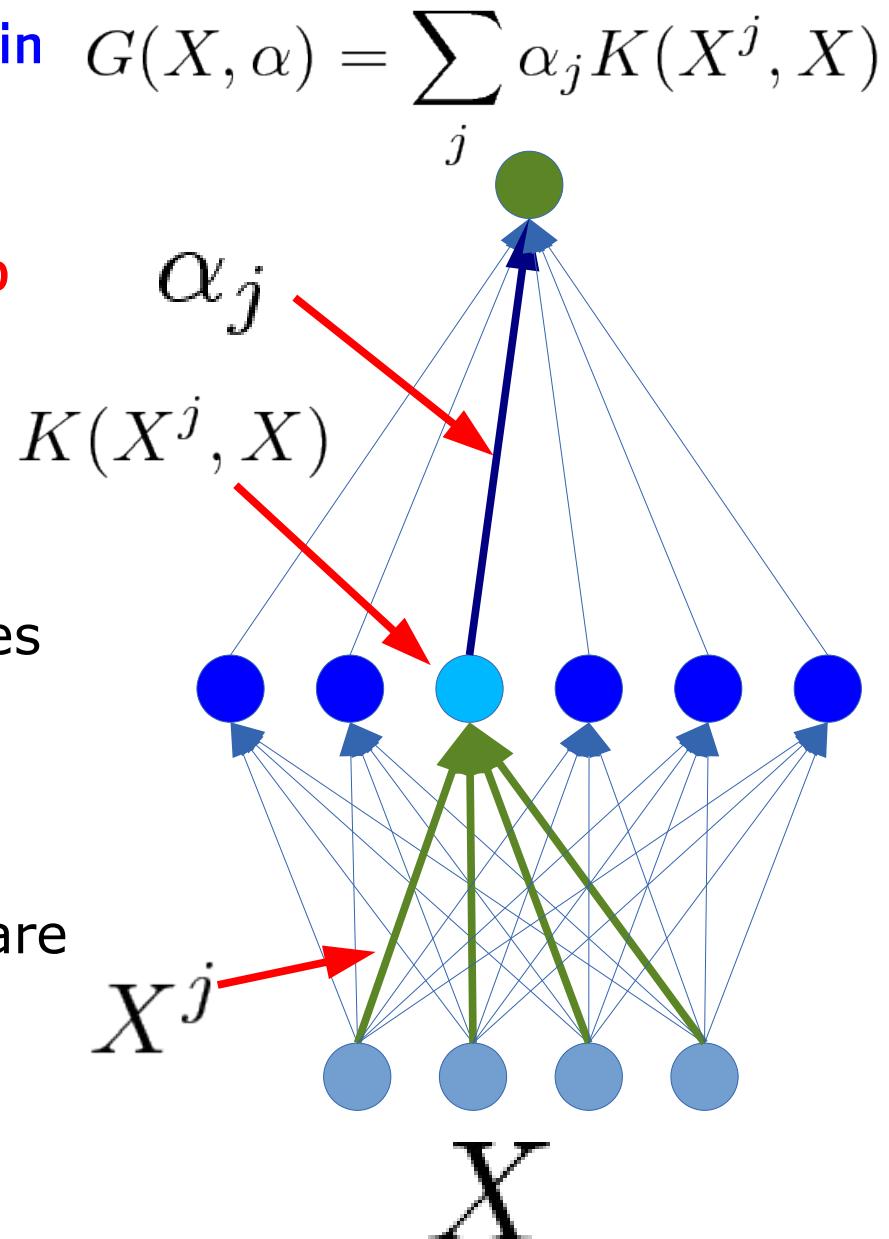
▶ Layer1: kernels; layer2: linear

▶ The first layer is “trained” in with the simplest unsupervised method ever devised: using the samples as templates for the kernel functions.

▶ “glorified template matching”

- Classification trees are not deep

▶ No hierarchy of features. All decisions are made in the input space





A dark blue rectangular overlay covers the center of the image, containing yellow text. The background consists of a complex arrangement of overlapping, translucent geometric shapes in shades of blue, red, and white, creating a sense of depth and motion.

What Are Good Feature?

Discovering the Hidden Structure in High-Dimensional Data

The manifold hypothesis

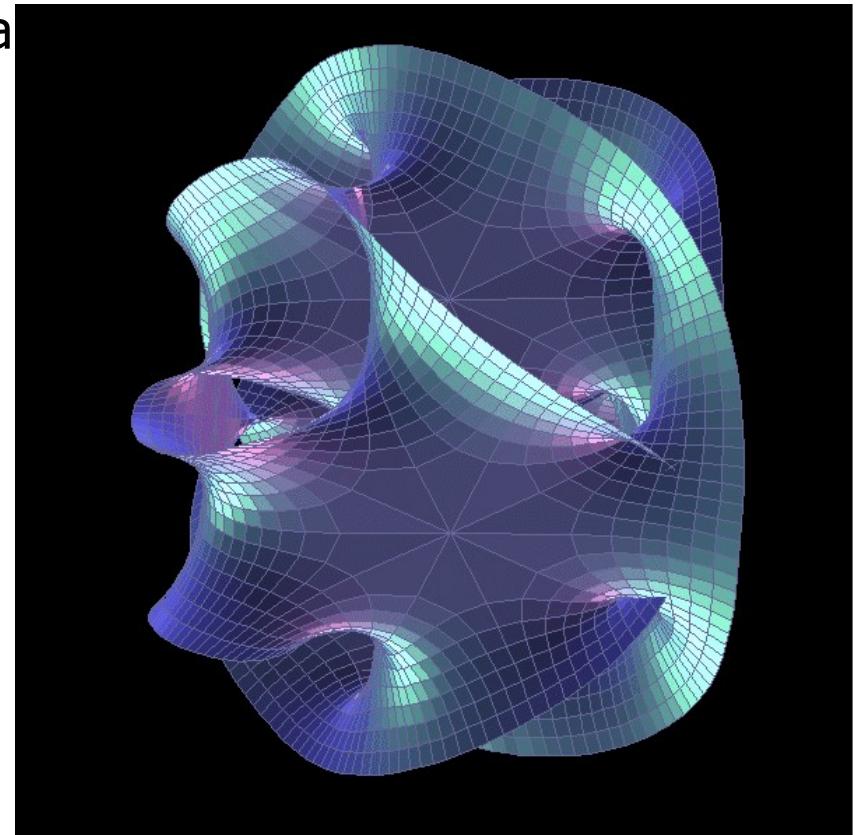
Y LeCun

■ Learning Representations of Data:

- ▶ **Discovering & disentangling the independent explanatory factors**

■ The Manifold Hypothesis:

- ▶ Natural data lives in a low-dimensional (non-linear) manifold
- ▶ Because variables in natural data



Basic Idea for Invariant Feature Learning

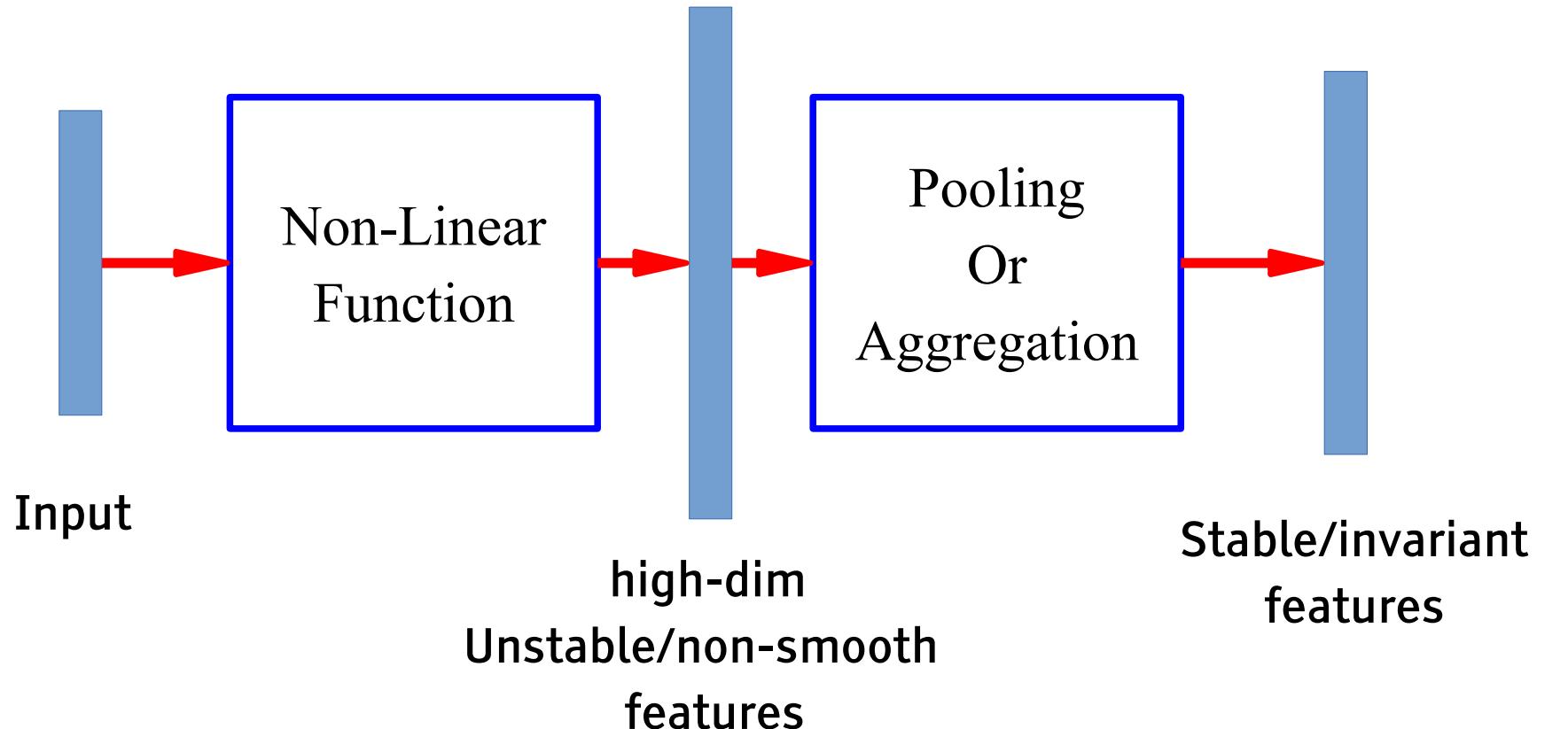
Y LeCun

■ Embed the input **non-linearly** into a high(er) dimensional space

- ▶ In the new space, things that were non separable may become separable

■ Pool regions of the new space together

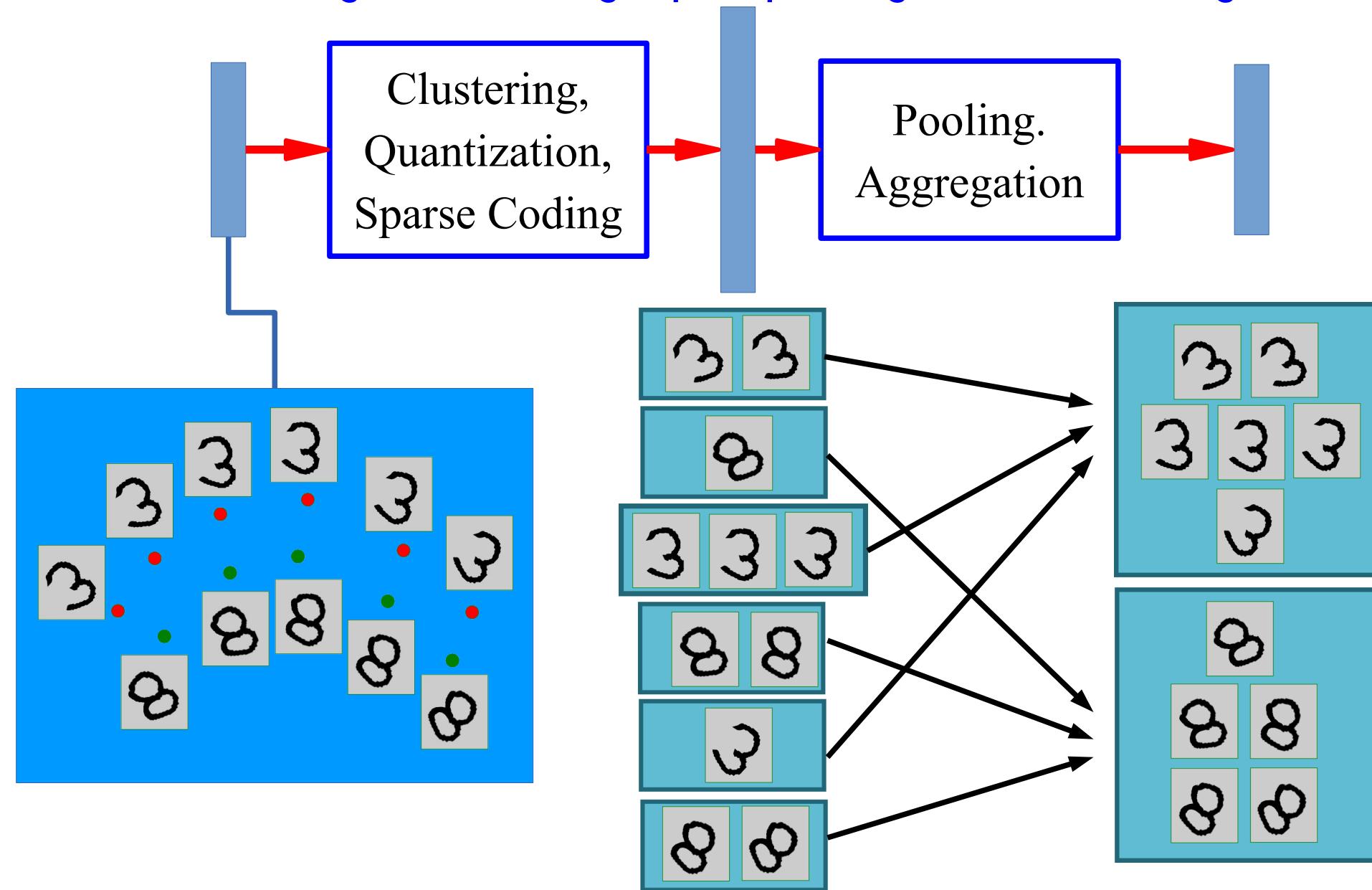
- ▶ Bringing together things that are semantically similar. Like pooling.



Sparse Non-Linear Expansion → Pooling

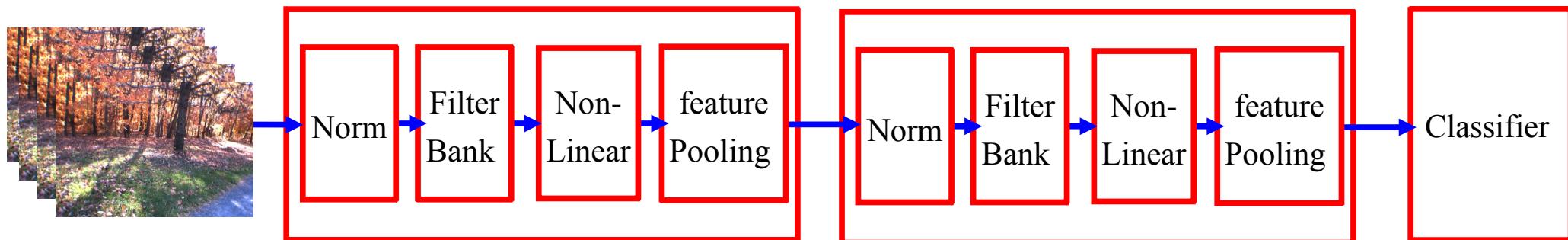
Y LeCun

- Use clustering to break things apart, pool together similar things



Overall Architecture: multiple stages of Normalization → Filter Bank → Non-Linearity → Pooling

Y LeCun



■ Normalization: variation on whitening (optional)

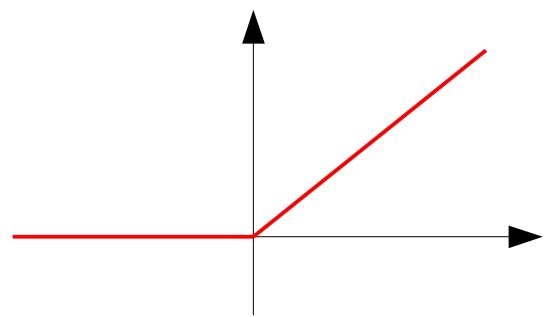
- Subtractive: average removal, high pass filtering
- Divisive: local contrast normalization, variance normalization

■ Filter Bank: dimension expansion, projection on overcomplete basis

■ Non-Linearity: sparsification, saturation, lateral inhibition....

- Rectification (ReLU), Component-wise shrinkage, tanh,..

$$ReLU(x) = \max(x, 0)$$



■ Pooling: aggregation over space or feature type

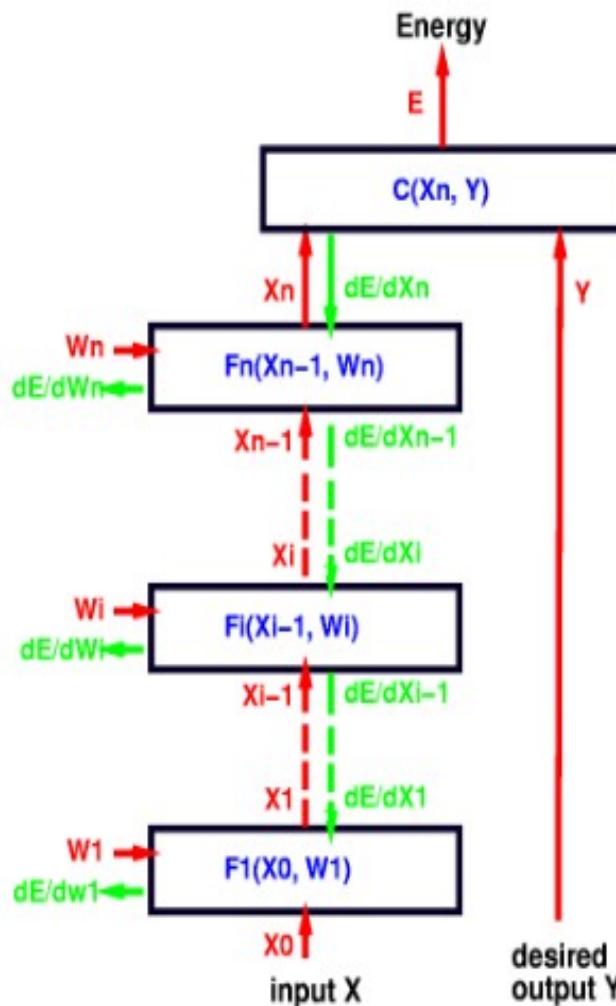
- Max, Lp norm, log prob.

$$MAX : Max_i(X_i); \quad L_p : \sqrt[p]{X_i^p}; \quad PROB : \frac{1}{b} \log \left(\sum_i e^{bX_i} \right)$$

Supervised Training: Stochastic (Sub)Gradient Optimization

Y LeCun

To compute all the derivatives, we use a backward sweep called the **back-propagation algorithm** that uses the recurrence equation for $\frac{\partial E}{\partial X_i}$



- $\frac{\partial E}{\partial X_n} = \frac{\partial C(X_n, Y)}{\partial X_n}$
- $\frac{\partial E}{\partial X_{n-1}} = \frac{\partial E}{\partial X_n} \frac{\partial F_n(X_{n-1}, W_n)}{\partial X_{n-1}}$
- $\frac{\partial E}{\partial W_n} = \frac{\partial E}{\partial X_n} \frac{\partial F_n(X_{n-1}, W_n)}{\partial W_n}$
- $\frac{\partial E}{\partial X_{n-2}} = \frac{\partial E}{\partial X_{n-1}} \frac{\partial F_{n-1}(X_{n-2}, W_{n-1})}{\partial X_{n-2}}$
- $\frac{\partial E}{\partial W_{n-1}} = \frac{\partial E}{\partial X_{n-1}} \frac{\partial F_{n-1}(X_{n-2}, W_{n-1})}{\partial W_{n-1}}$
-etc, until we reach the first module.
- we now have all the $\frac{\partial E}{\partial W_i}$ for $i \in [1, n]$.

Loss Function for a simple network

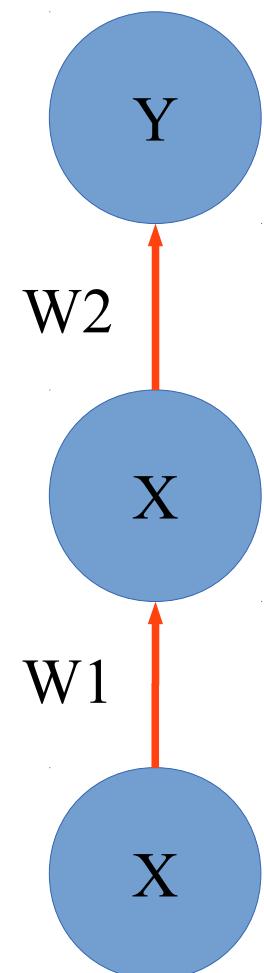
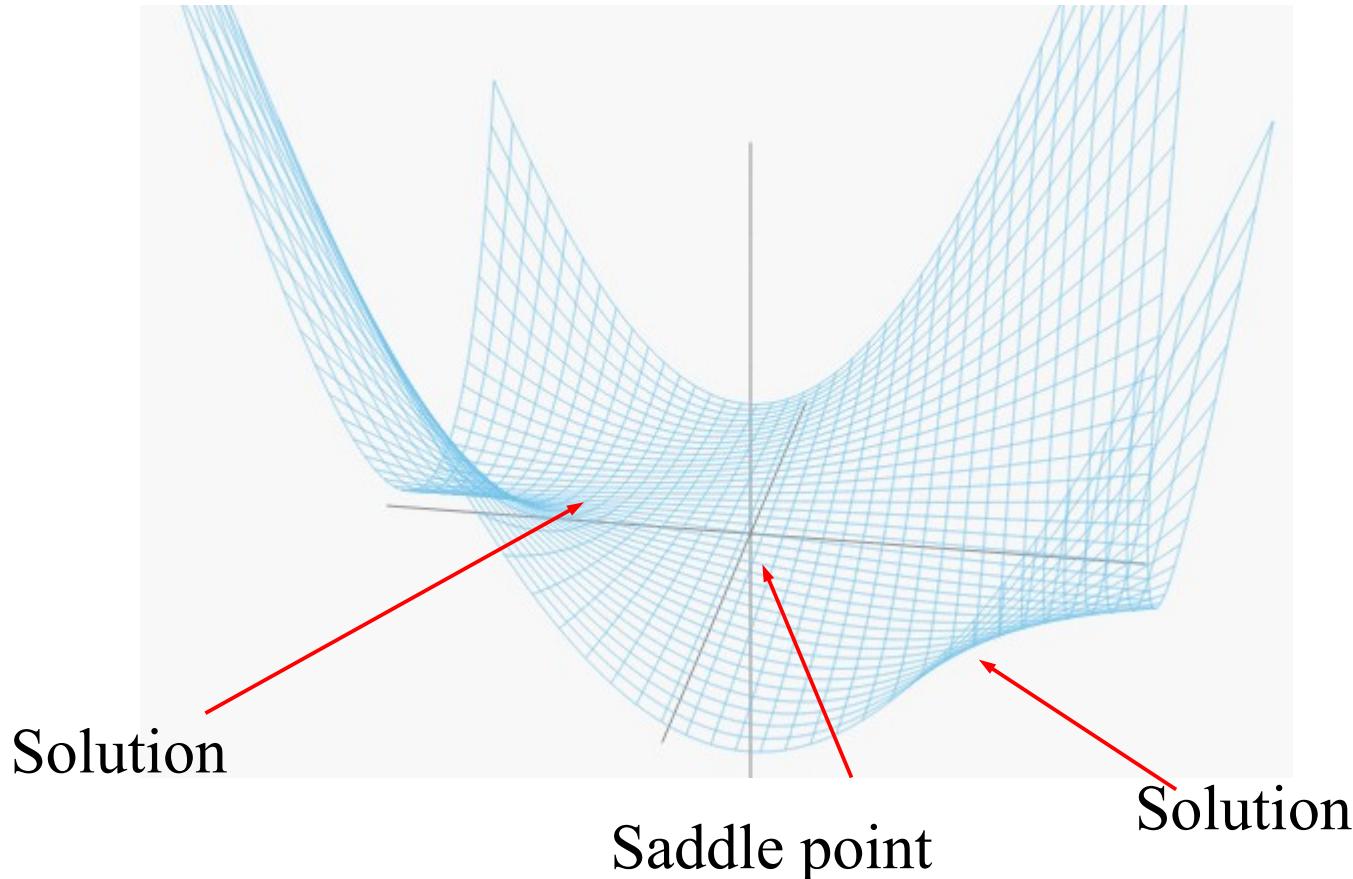
Y LeCun

1-1-1 network

- $Y = W_1 * W_2 * X$

trained to compute the identity function with quadratic loss

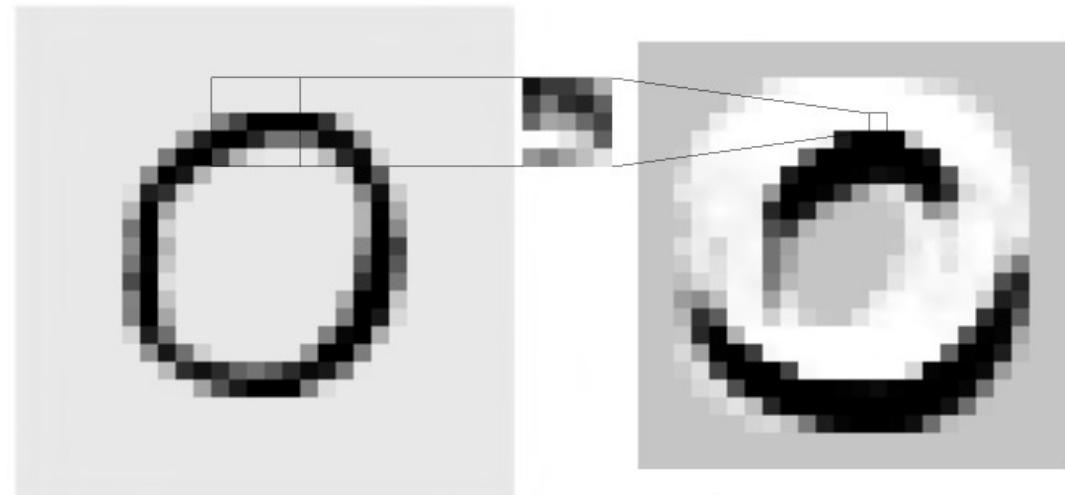
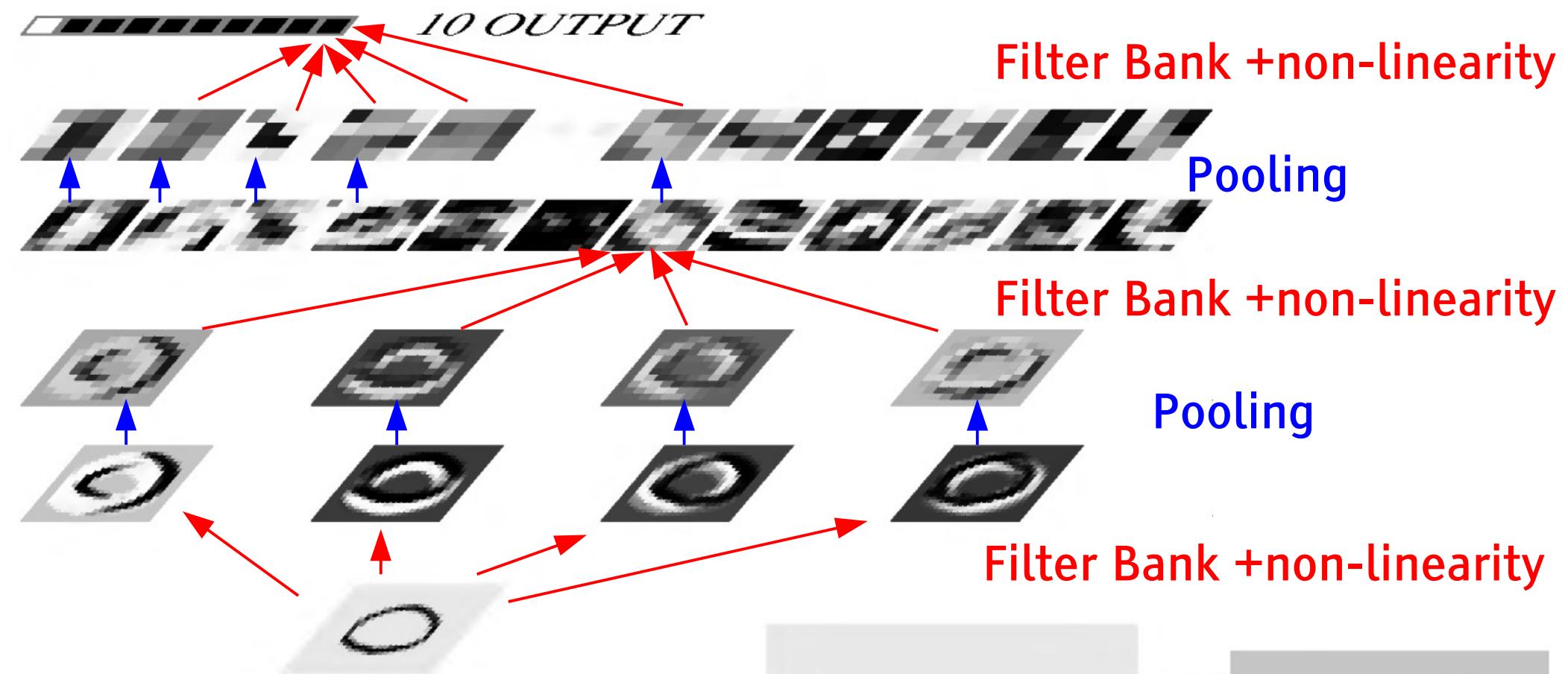
- Single sample $X=1, Y=1 \ L(W) = (1-W_1 * W_2)^2$



Convolutional Networks

Convolutional Network

Y LeCun



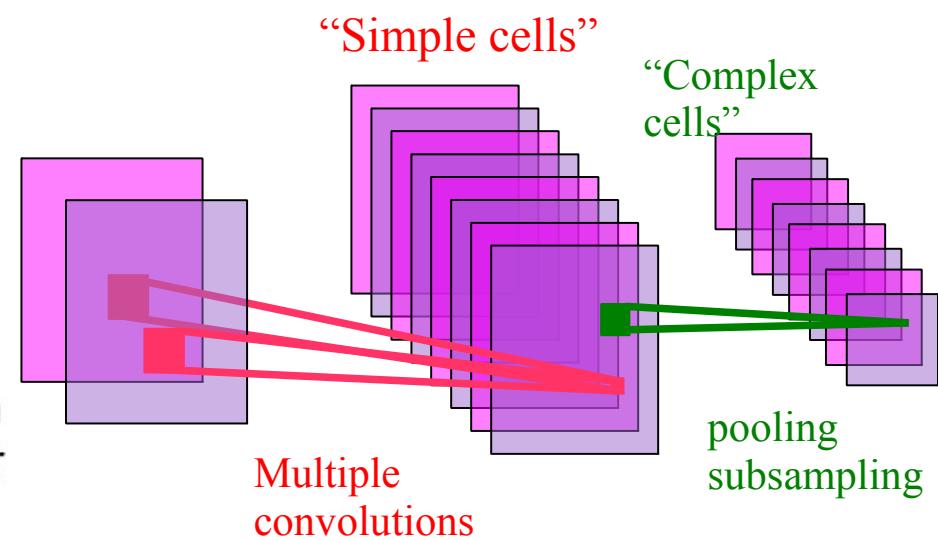
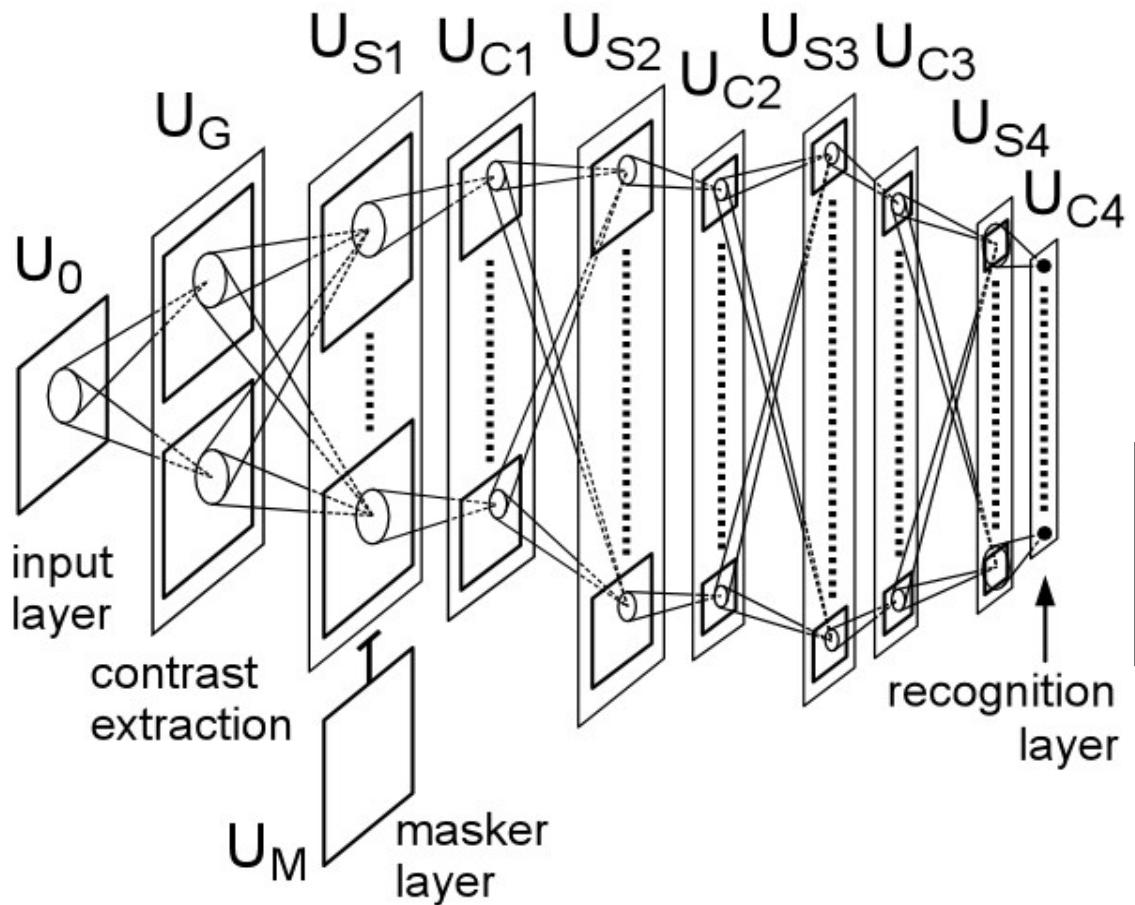
[LeCun et al. NIPS 1989]

Early Hierarchical Feature Models for Vision

Y LeCun

■ [Hubel & Wiesel 1962]:

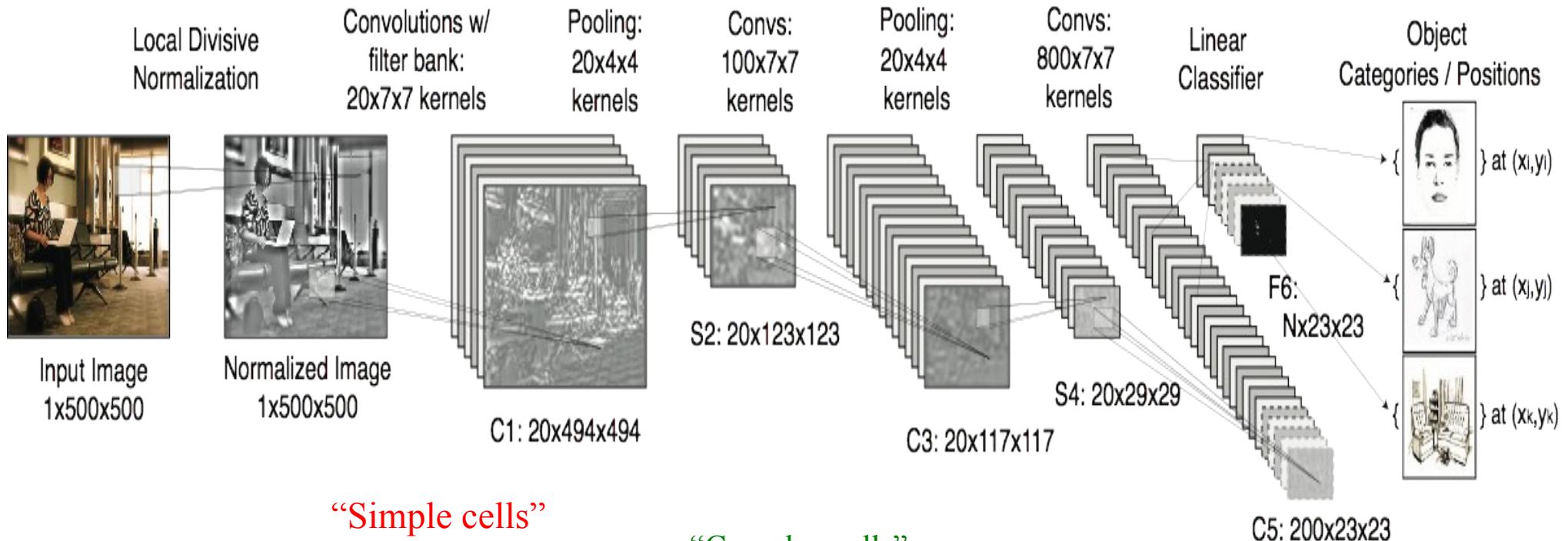
- ▶ simple cells detect local features
- ▶ complex cells “pool” the outputs of simple cells within a retinotopic neighborhood.



Cognitron & Neocognitron [Fukushima 1974-1982]

The Convolutional Net Model (Multistage Hubel-Wiesel system)

Y LeCun



“Simple cells”

“Complex cells”

Multiple convolutions

pooling
subsampling

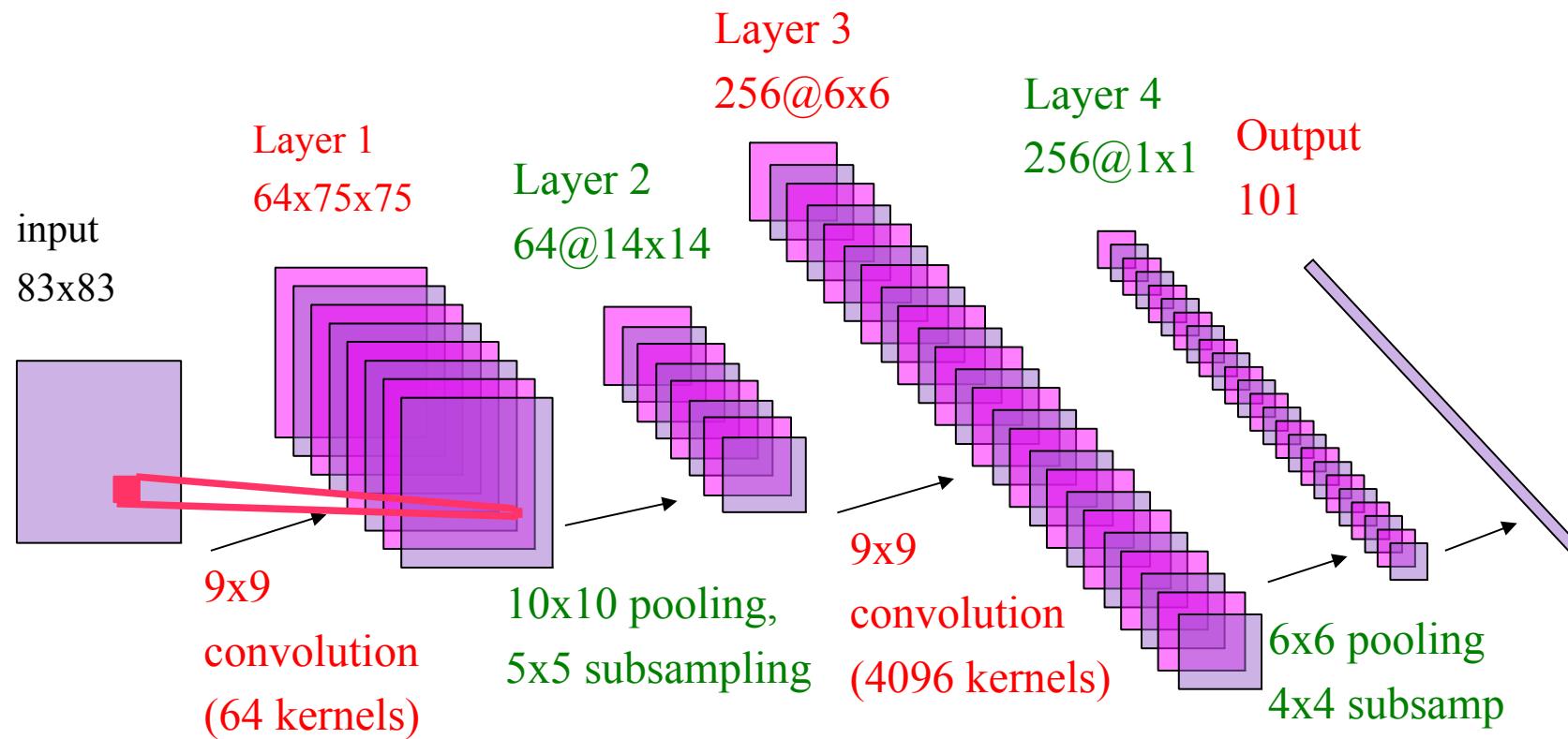
Retinotopic Feature Maps

Training is supervised
 With stochastic gradient descent

[LeCun et al. 89]
[LeCun et al. 98]

Convolutional Network (ConvNet)

Y LeCun



- **Non-Linearity:** half-wave rectification (ReLU), shrinkage function, sigmoid
- **Pooling:** max, average, L1, L2, log-sum-exp
- **Training:** Supervised (1988-2006), Unsupervised+Supervised (2006-now)

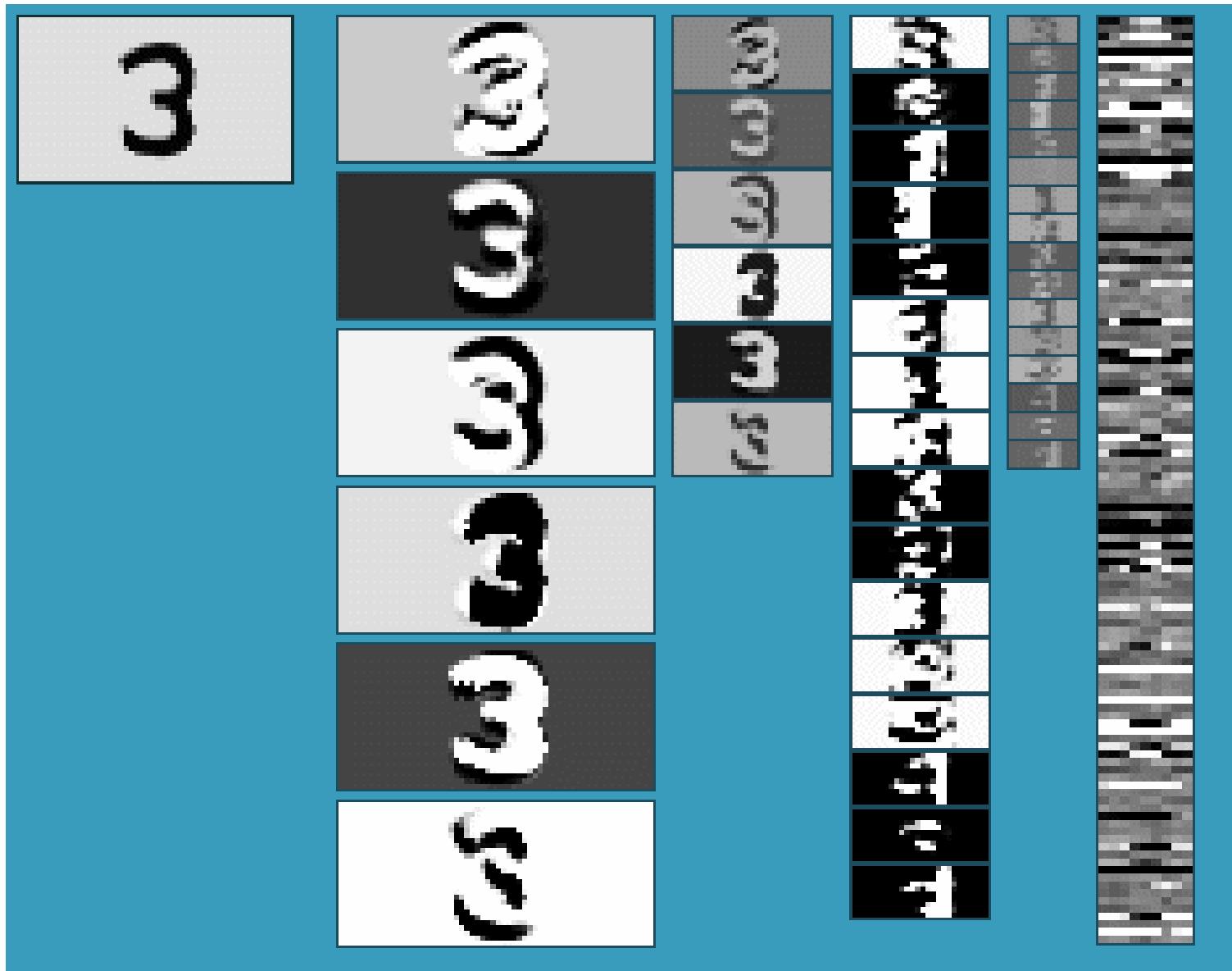
Convolutional Network (vintage 1990)

Y LeCun

filters → tanh → average-tanh → filters → tanh → average-tanh → filters → tanh

Curved
manifold

Flatter
manifold



LeNet1 Demo from 1993

Y LeCun

- Running on a 486 PC with an AT&T DSP32C add-on board (20 Mflops!)





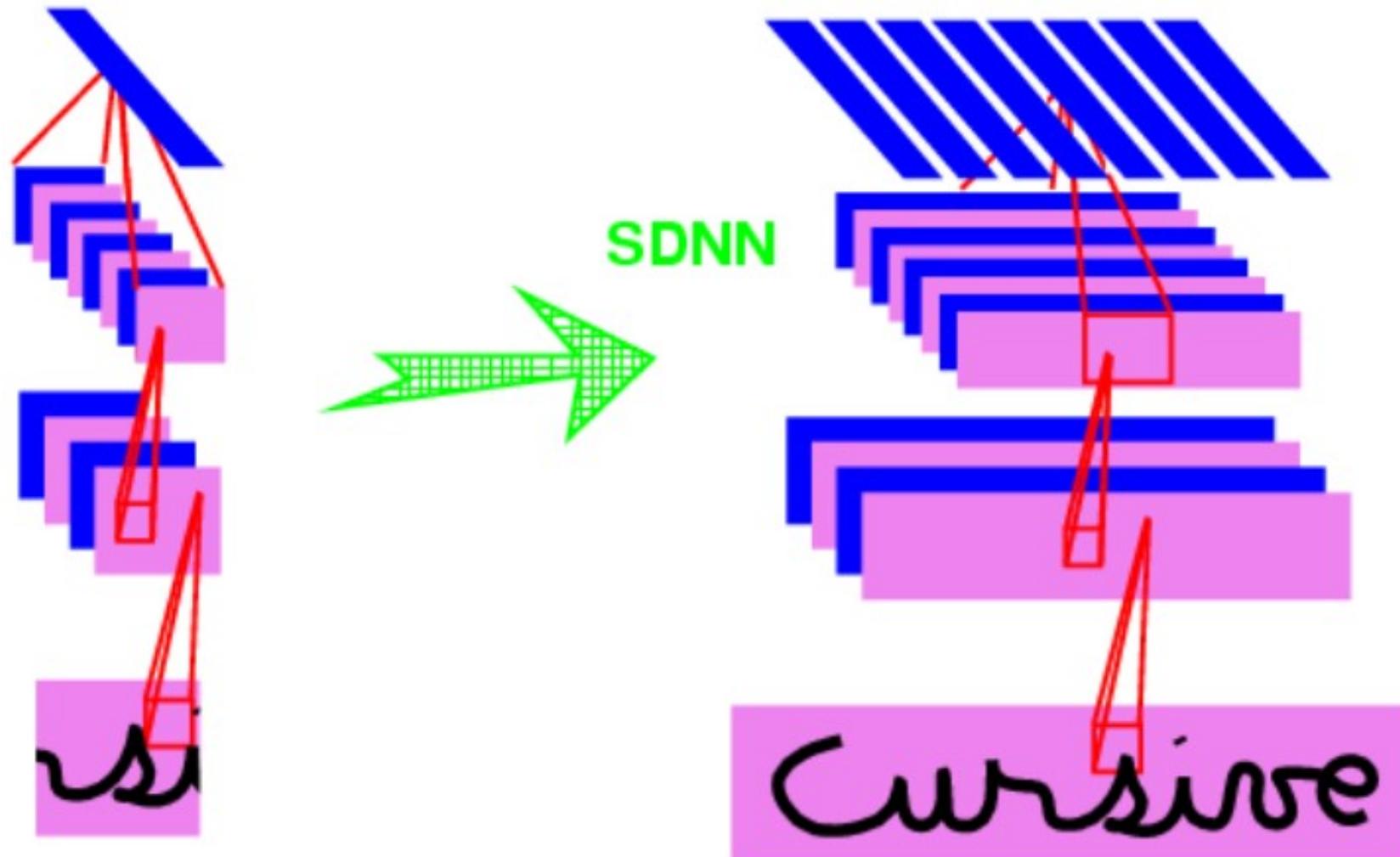
A dark blue rectangular overlay covers the center of the image, containing the title text.

Brute Force Approach To Multiple Object Recognition

Idea #1: Sliding Window ConvNet + Weighted FSM

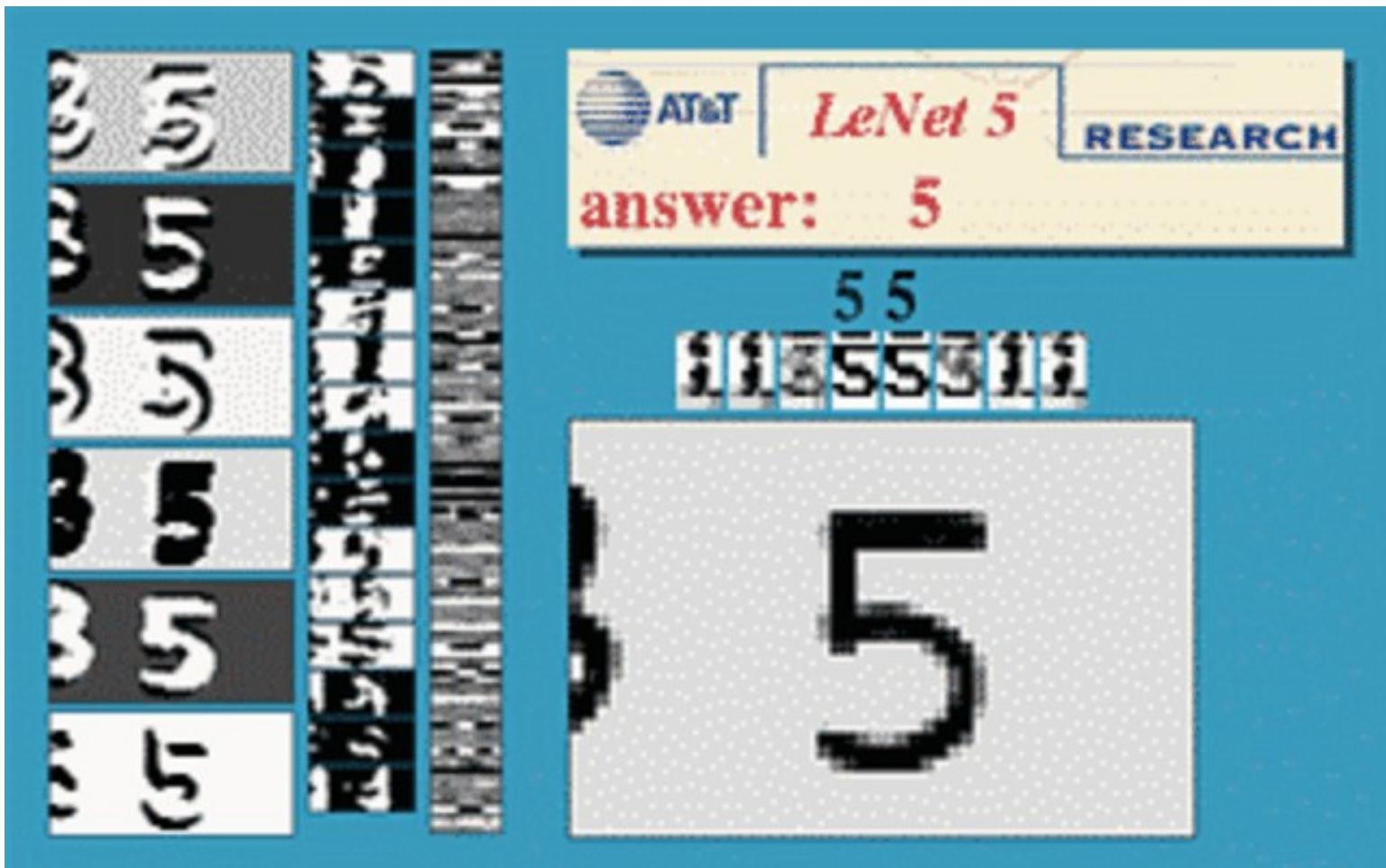
Y LeCun

- "Space Displacement Neural Net".
- Convolutions are applied to a large image
- Output and feature maps are extended/replicated accordingly



Idea #1: Sliding Window ConvNet + Weighted FSM

Y LeCun



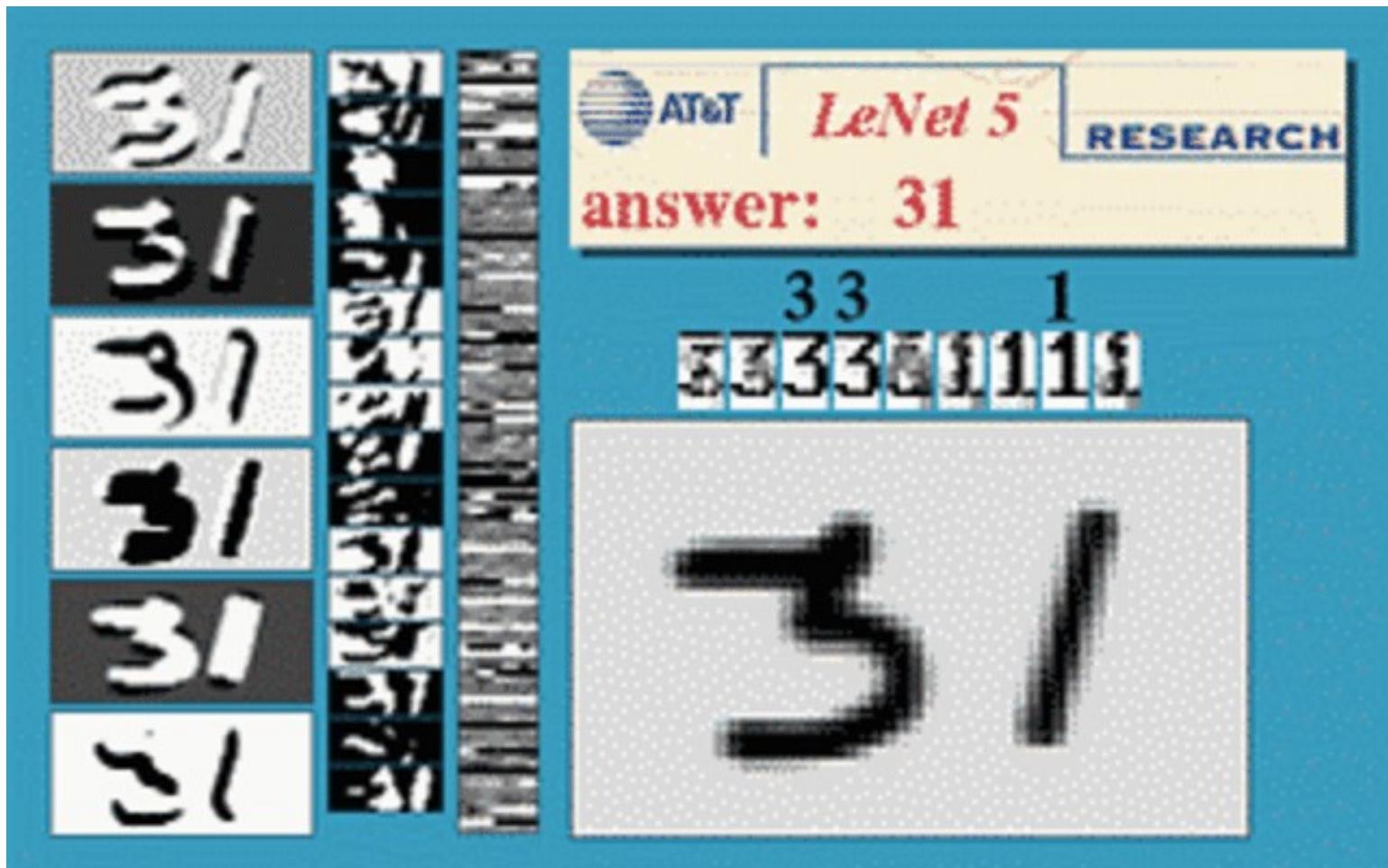
Idea #1: Sliding Window ConvNet + Weighted FSM

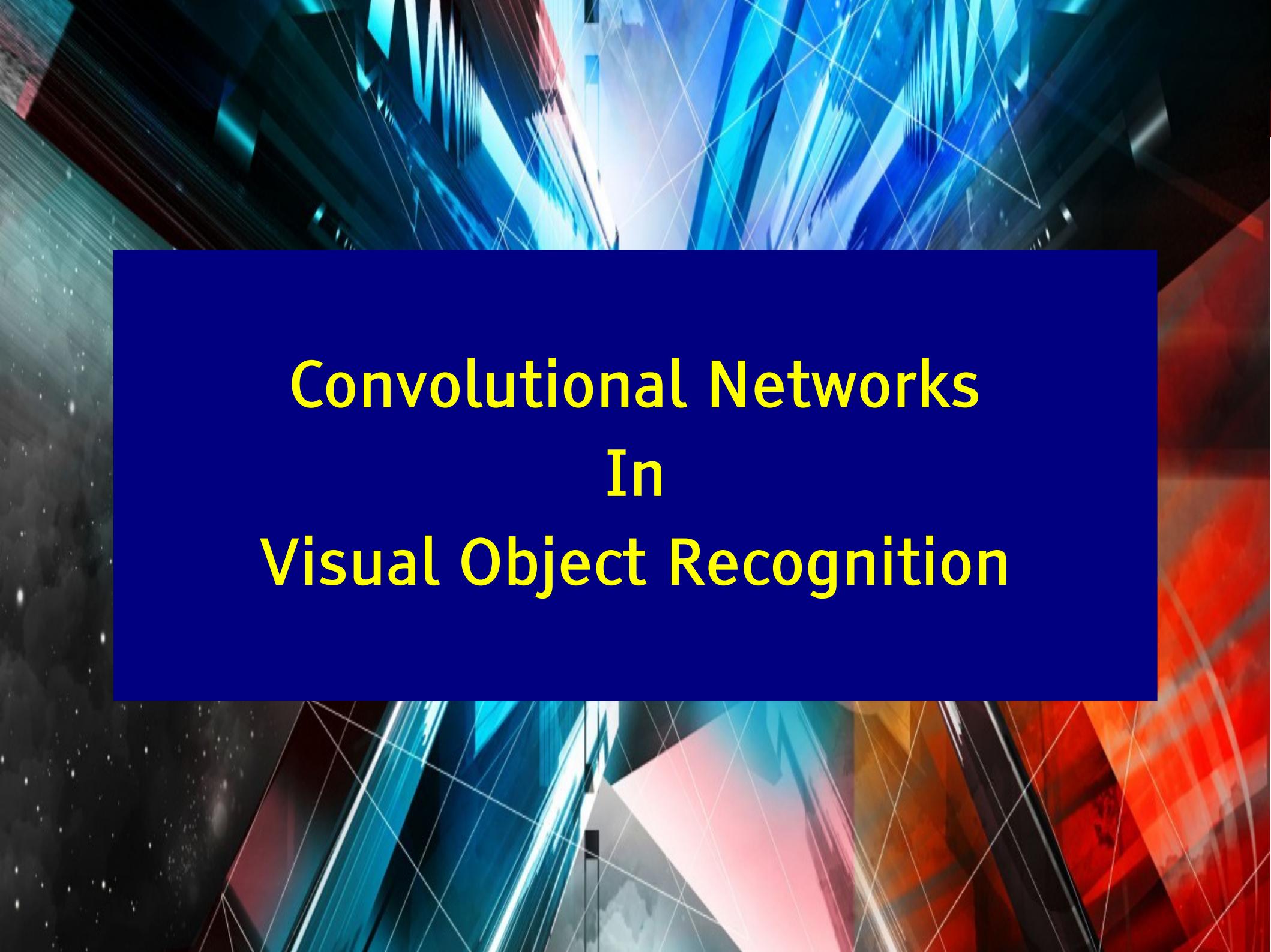
Y LeCun



Idea #1: Sliding Window ConvNet + Weighted FSM

Y LeCun





Convolutional Networks In Visual Object Recognition



We knew ConvNet worked well with characters and small images

Y LeCun

■ Traffic Sign Recognition (GTSRB)

- ▶ German Traffic Sign Reco Bench
- ▶ 99.2% accuracy (IDSIA)



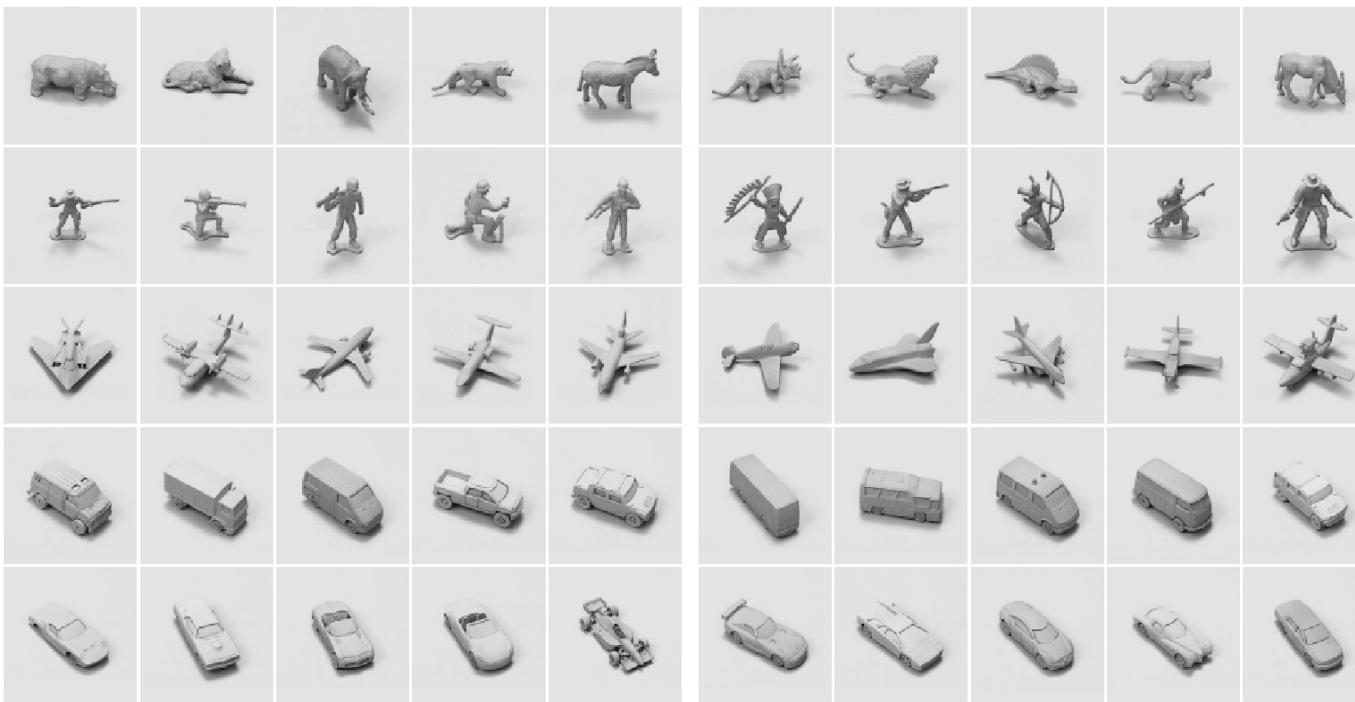
■ House Number Recognition (Google)

- ▶ Street View House Numbers
- ▶ 94.3 % accuracy (NYU)



NORB Dataset (2004): 5 categories, multiple views and illuminations

Y LeCun



■ Less than 6% error on test set with cluttered backgrounds

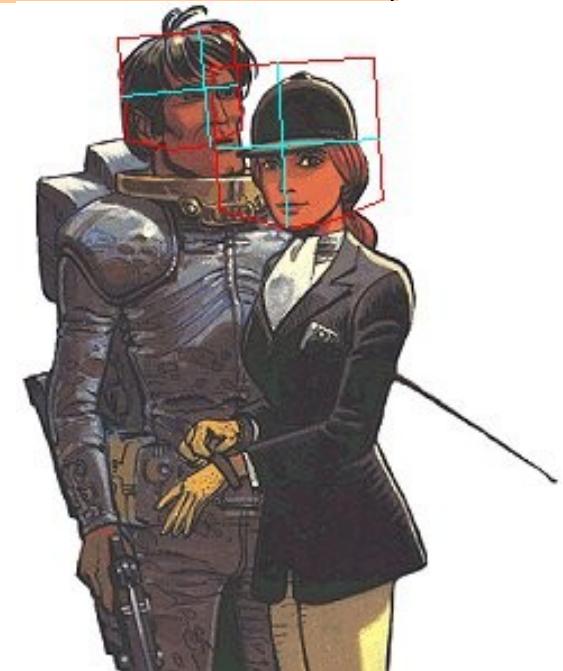
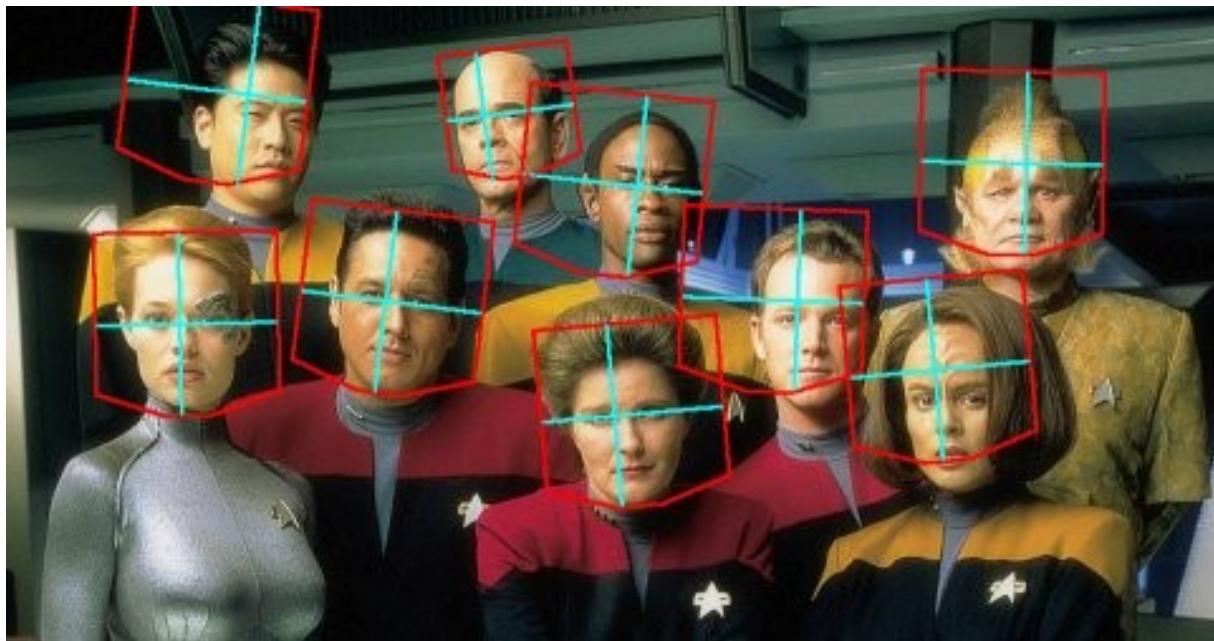
291,600 training samples,
58,320 test samples



mid 2000s: state of the art results on face detection

Y LeCun

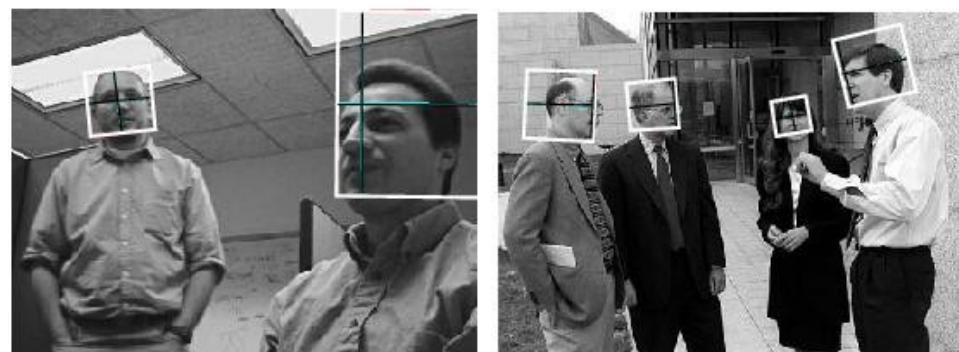
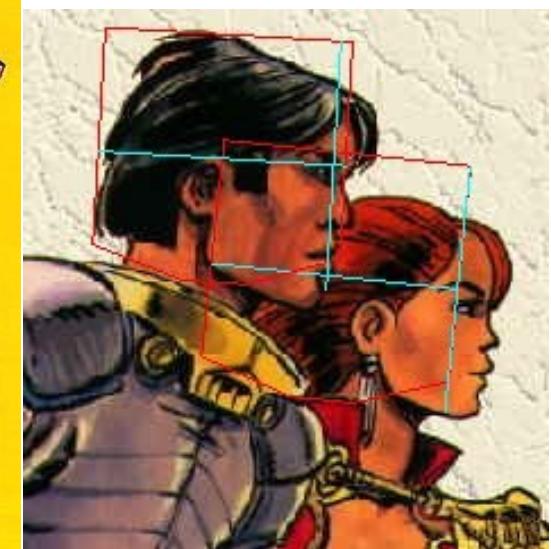
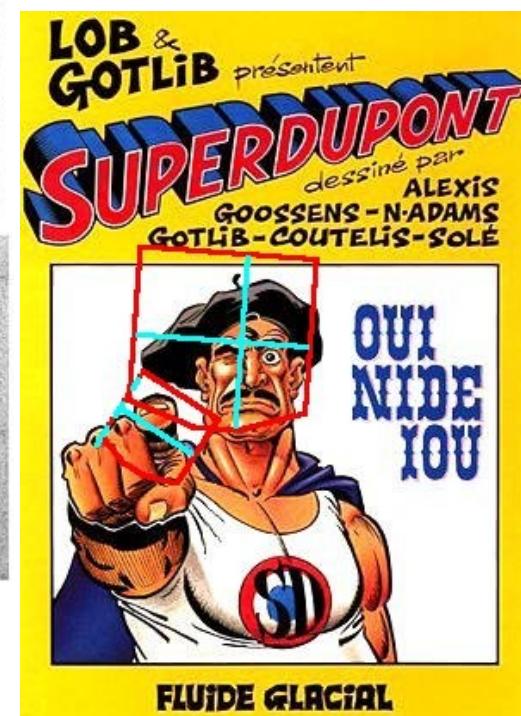
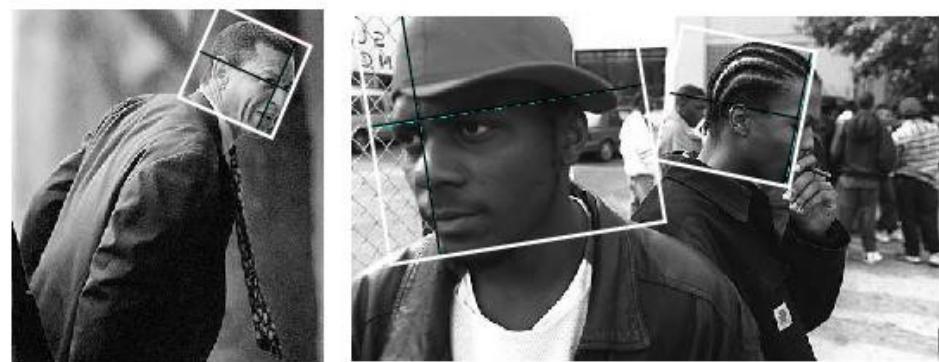
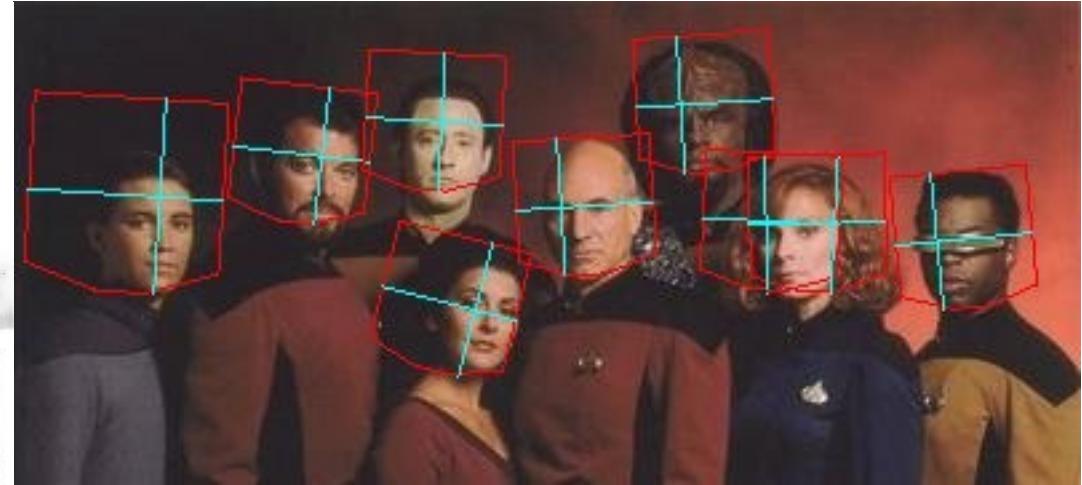
<i>Data Set-></i>	TILTED		PROFILE		MIT+CMU	
<i>False positives per image-></i>	4.42	26.9	0.47	3.36	0.5	1.28
Our Detector	90%	97%	67%	83%	83%	88%
Jones & Viola (tilted)	90%	95%	x		x	
Jones & Viola (profile)	x		70%	83%	x	



[Vaillant et al. IEE 1994] [Osadchy et al. 2004] [Osadchy et al, JMLR 2007]

Simultaneous face detection and pose estimation

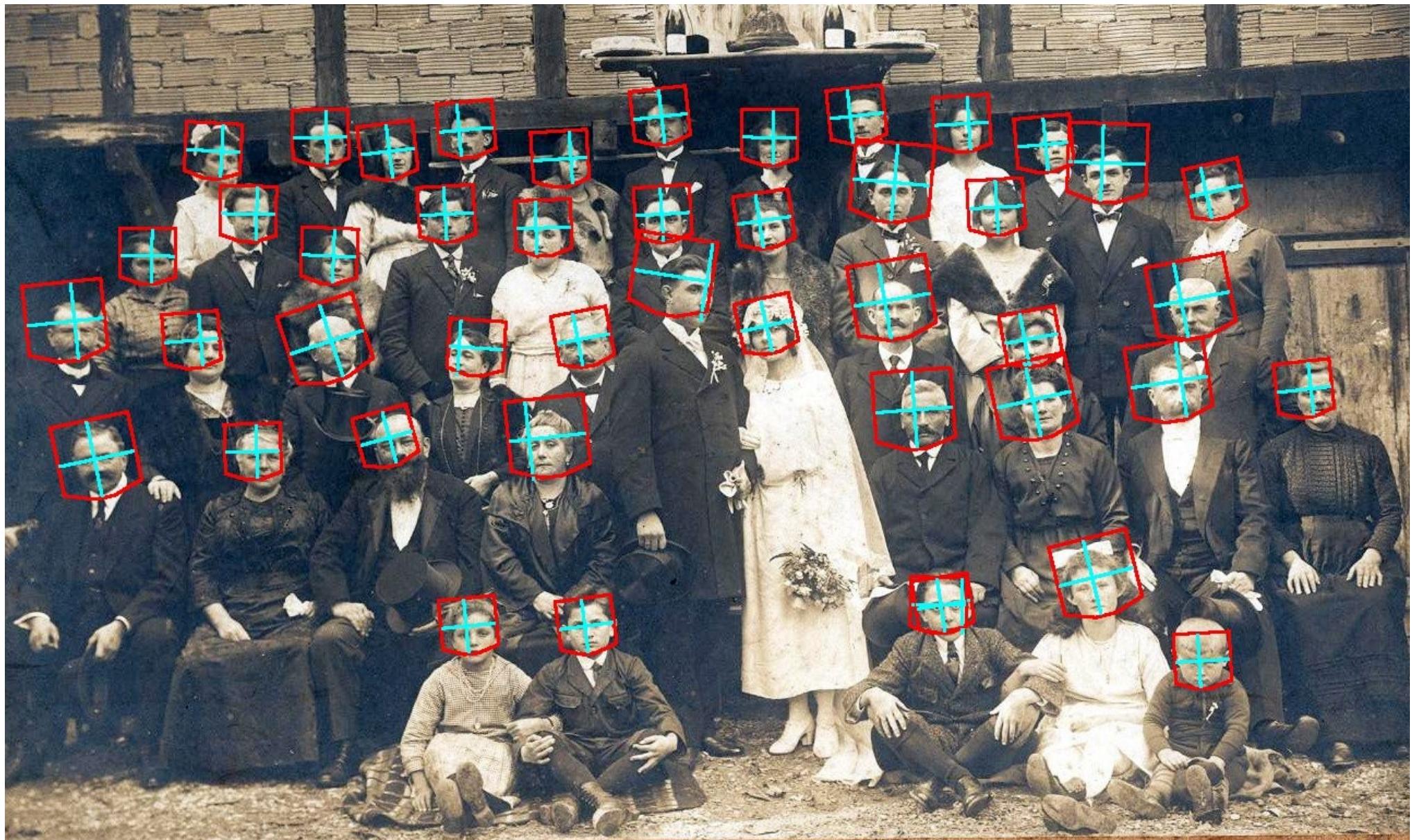
Y LeCun



[Vaillant et al. IEE 1994] [Osadchy et al. 2004] [Osadchy et al, JMLR 2007]

Simultaneous face detection and pose estimation

Y LeCun



Visual Object Recognition with Convolutional Nets

Y LeCun

■ In the mid 2000s, ConvNets were getting decent results on object classification

■ Dataset: "Caltech101":

- ▶ 101 categories
- ▶ 30 training samples per category

■ But the results were slightly worse than more "traditional" computer vision methods, because

- ▶ 1. the datasets were too small
- ▶ 2. the computers were too slow

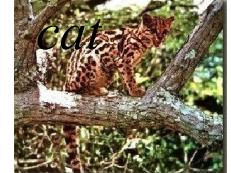
face



beaver



wild cat



lotus



an



background



dollar



metronome



w. chair



minar

cellphon



joshua



cougar body



Late 2000s: we could get decent results on object recognition

Y LeCun

- But we couldn't beat the state of the art because the datasets were too small
- Caltech101: 101 categories, 30 samples per category.
- But we learned that rectification and max pooling are useful! [Jarrett et al. ICCV 2009]

Single Stage System: $[64.F_{CSG}^{9 \times 9} - R/N/P^{5 \times 5}] - \text{log_reg}$

R/N/P	$R_{abs} - N - P_A$	$R_{abs} - P_A$	$N - P_M$	$N - P_A$	P_A
U^+	54.2%	50.0%	44.3%	18.5%	14.5%
R^+	54.8%	47.0%	38.0%	16.3%	14.3%
U	52.2%	43.3% (± 1.6)	44.0%	17.2%	13.4%
R	53.3%	31.7%	32.1%	15.3%	12.1% (± 2.2)
G	52.3%				

Two Stage System: $[64.F_{CSG}^{9 \times 9} - R/N/P^{5 \times 5}] - [256.F_{CSG}^{9 \times 9} - R/N/P^{4 \times 4}] - \text{log_reg}$

R/N/P	$R_{abs} - N - P_A$	$R_{abs} - P_A$	$N - P_M$	$N - P_A$	P_A
$U^+ U^+$	65.5%	60.5%	61.0%	34.0%	32.0%
$R^+ R^+$	64.7%	59.5%	60.0%	31.0%	29.7%
UU	63.7%	46.7%	56.0%	23.1%	9.1%
RR	62.9%	33.7% (± 1.5)	37.6% (± 1.9)	19.6%	8.8%
GT	55.8%	← like HMAX model			

Single Stage: $[64.F_{CSG}^{9 \times 9} - R/N/P^{5 \times 5}] - \text{PMK-SVM}$

U	64.0%	
Two Stages: $[64.F_{CSG}^{9 \times 9} - R/N/P^{5 \times 5}] - [256.F_{CSG}^{9 \times 9} - R/N] - \text{PMK-SVM}$	52.8%	



Then., two things happened...

Y LeCun

■ The ImageNet dataset [Fei-Fei et al. 2012]

- ▶ 1.5 million training samples
- ▶ 1000 categories

■ Fast Graphical Processing Units (GPU)

- ▶ Capable of 1 trillion operations/second



Matchstick



Sea lion



Flute



Strawberry



Bathing



cap

Racket



ImageNet Large-Scale Visual Recognition Challenge

Y LeCun

The ImageNet dataset

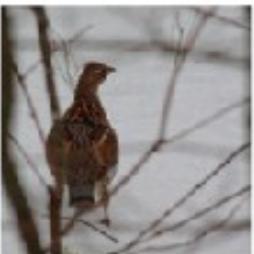
- ▶ 1.5 million training samples
- ▶ 1000 fine-grained categories (breeds of dogs....)



flamingo



cock



ruffed grouse



quail



partridge

...



pill bottle



beer bottle



wine bottle



water bottle



pop bottle

...



race car



wagon



minivan



jeep



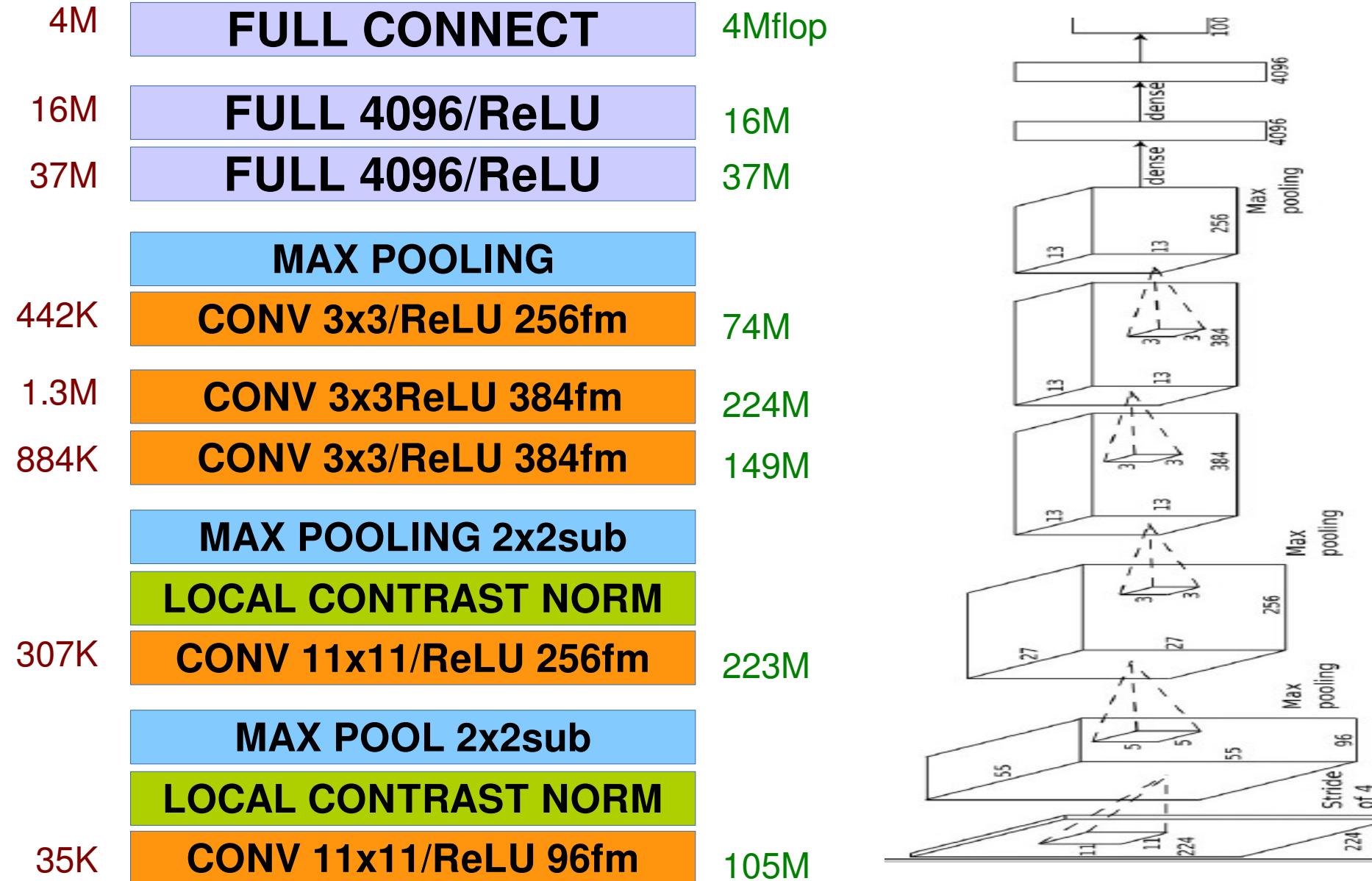
cab

...

Object Recognition [Krizhevsky, Sutskever, Hinton 2012]

Y LeCun

Won the 2012 ImageNet LSVRC. 60 Million parameters, 832M MAC ops



Object Recognition [Krizhevsky, Sutskever, Hinton 2012]

Y LeCun

■ Method: large convolutional net

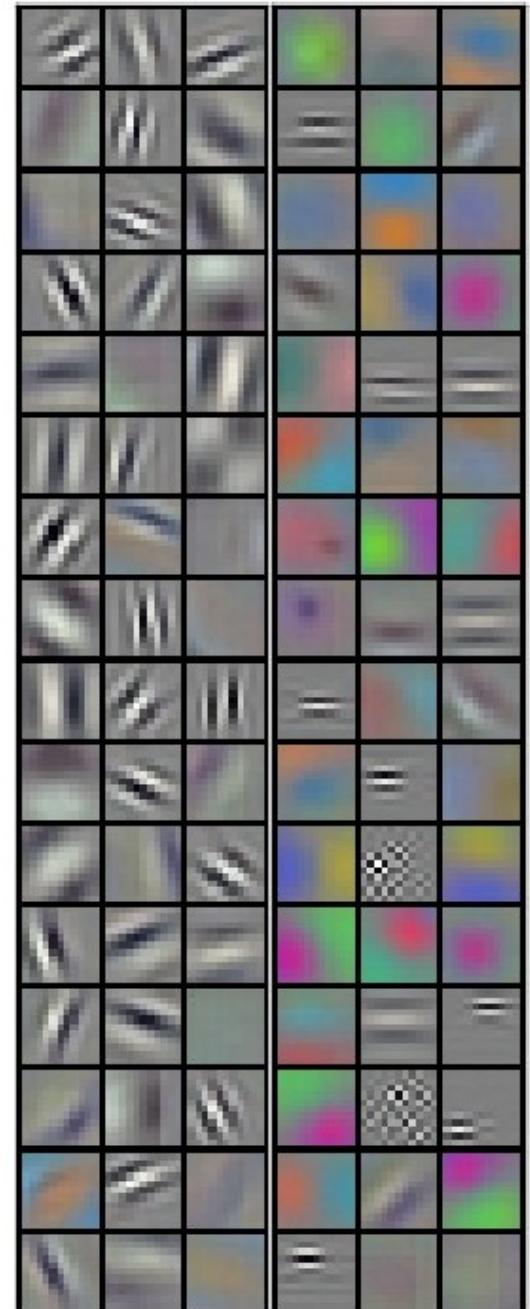
- ▶ 650K neurons, 832M synapses, 60M parameters
- ▶ Trained with backprop on GPU
- ▶ Trained “with all the tricks Yann came up with in the last 20 years, plus dropout” (Hinton, NIPS 2012)
- ▶ Rectification, contrast normalization,...

■ Error rate: 15% (whenever correct class isn't in top 5)

■ Previous state of the art: 25% error

■ A REVOLUTION IN COMPUTER VISION

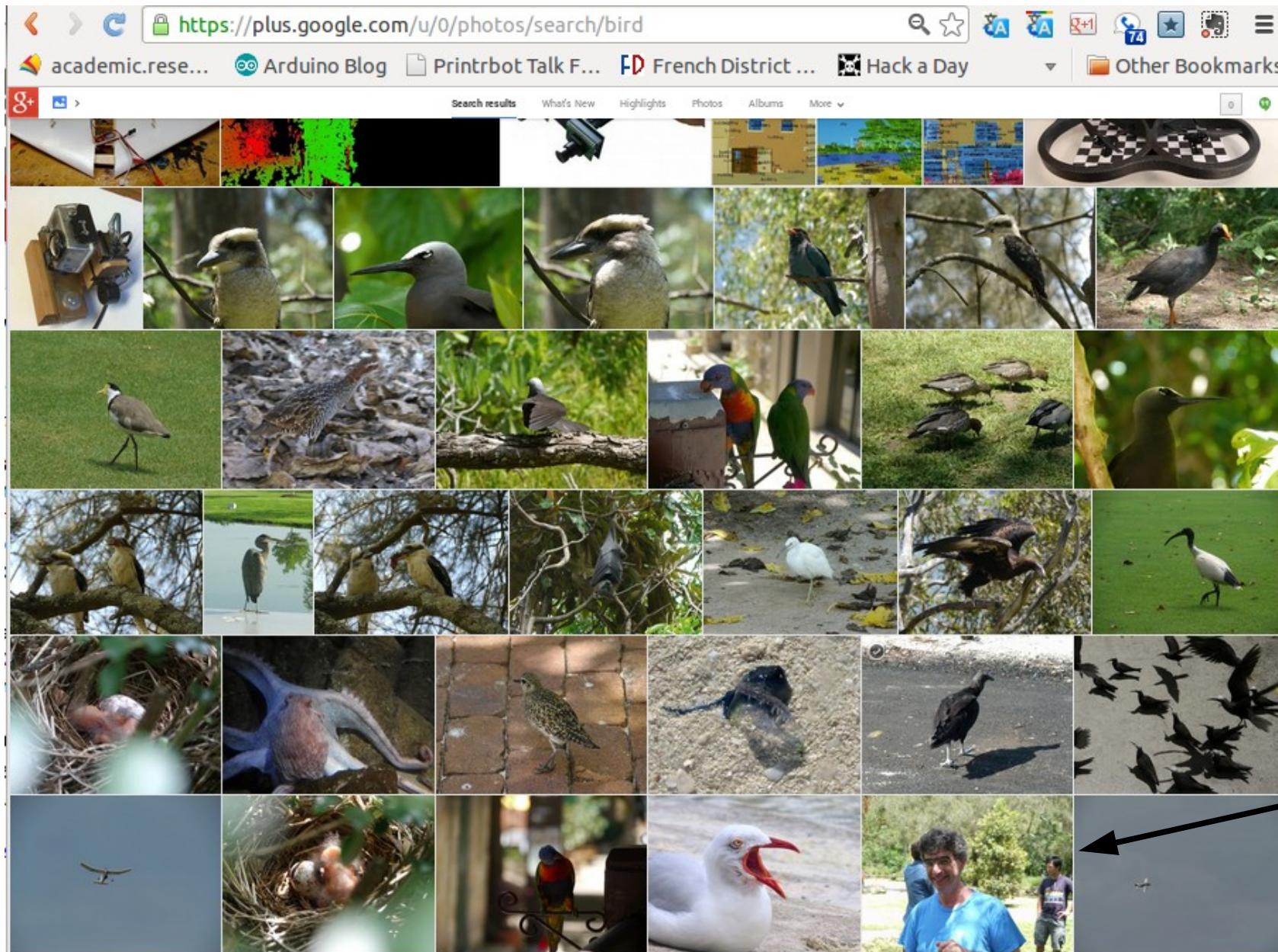
- Acquired by Google in Jan 2013
- Deployed in Google+ Photo Tagging in May 2013



ConvNet-Based Google+ Photo Tagger

Y LeCun

Searched my personal collection for “bird”



Samy
Bengio
???

NYU ConvNet Trained on ImageNet: OverFeat

Y LeCun

[Sermanet et al. arXiv:1312.6229]

Trained on GPU using Torch7

Uses a number of new tricks

Classification 1000 categories:

- ▶ 13.8% error (top 5) with an ensemble of 7 networks (Krizhevsky: 15%)
- ▶ 15.4% error (top 5) with a single network (Krizhevsky: 18.2%)

Classification+Localization

- ▶ 30% error (Krizhevsky: 34%)

Detection (200 categories)

- ▶ 19% correct

Downloadable code (running, no training)

- ▶ Search for “overfeat NYU” on Google
- ▶ <http://cilvr.nyu.edu> → software

FULL 1000/Softmax

FULL 4096/ReLU

FULL 4096/ReLU

MAX POOLING 3x3sub

CONV 3x3/ReLU 256fm

CONV 3x3ReLU 384fm

CONV 3x3/ReLU 384fm

MAX POOLING 2x2sub

CONV 7x7/ReLU 256fm

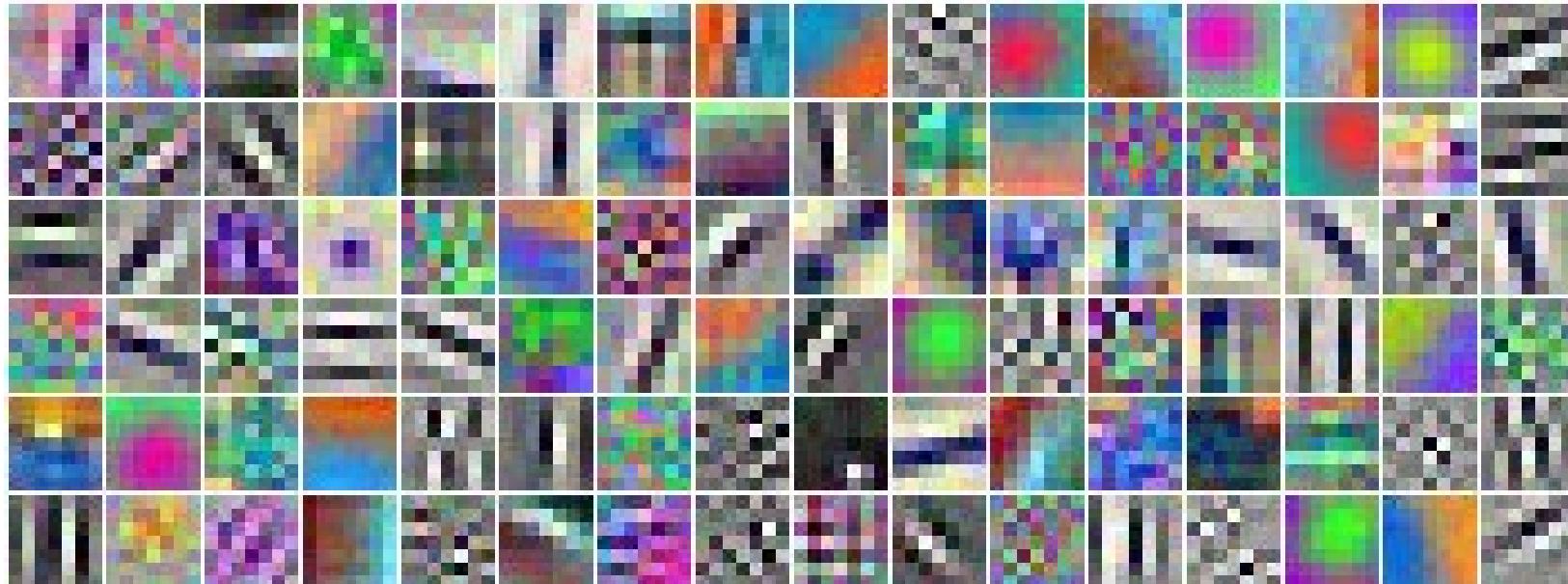
MAX POOL 3x3sub

CONV 7x7/ReLU 96fm

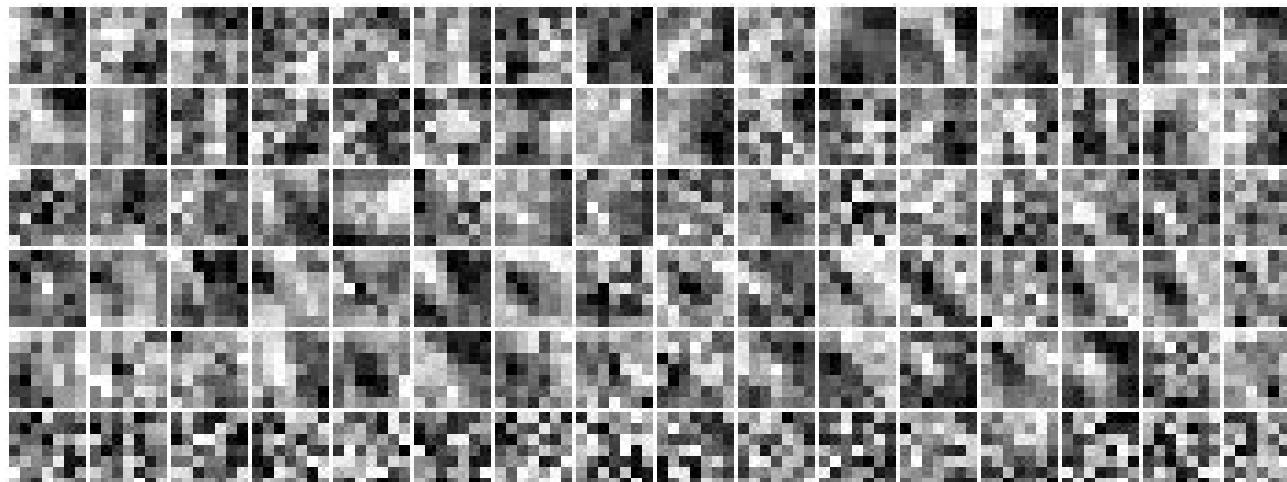
Kernels: Layer 1 (7x7)and Layer 2 (7x7)

Y LeCun

Layer 1: 3x96 kernels, RGB->96 feature maps, 7x7 Kernels, stride 2



Layer 2: 96x256 kernels, 7x7

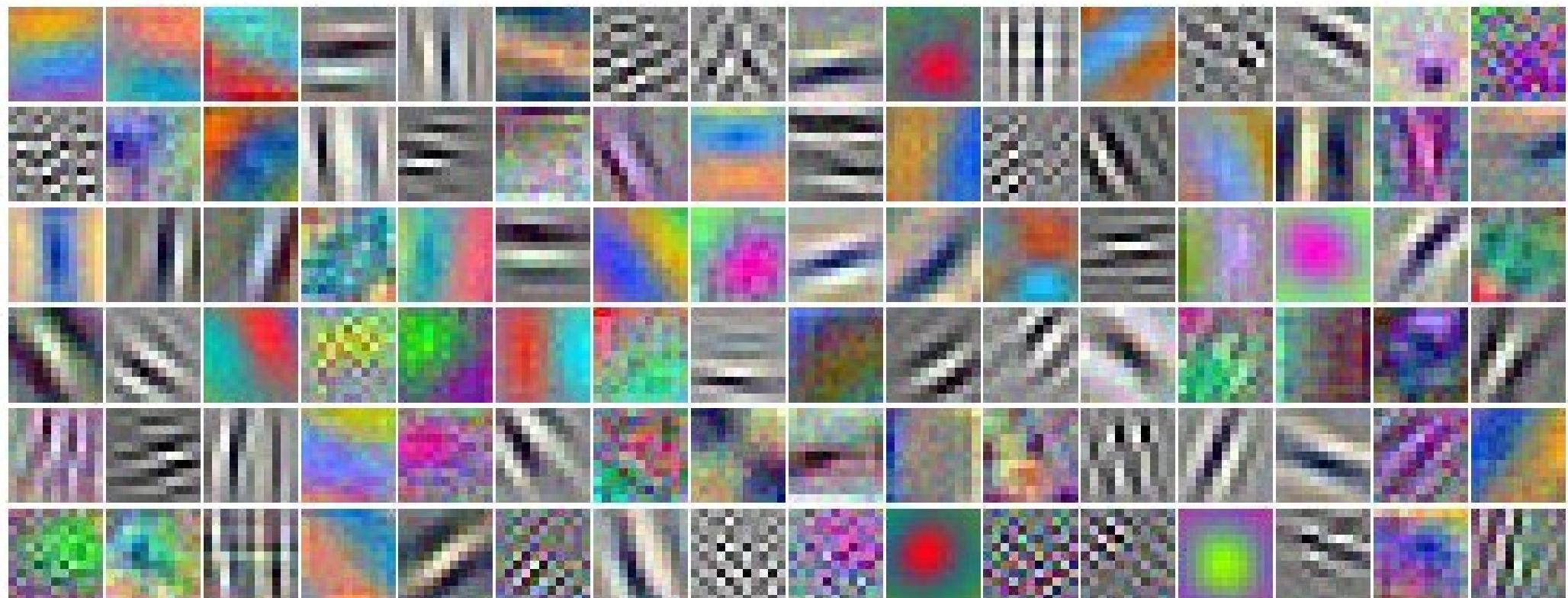




Kernels: Layer 1 (11x11)

Y LeCun

Layer 1: 3x96 kernels, RGB->96 feature maps, 11x11 Kernels, stride 4



ImageNet: Classification

Y LeCun

■ Give the name of the dominant object in the image

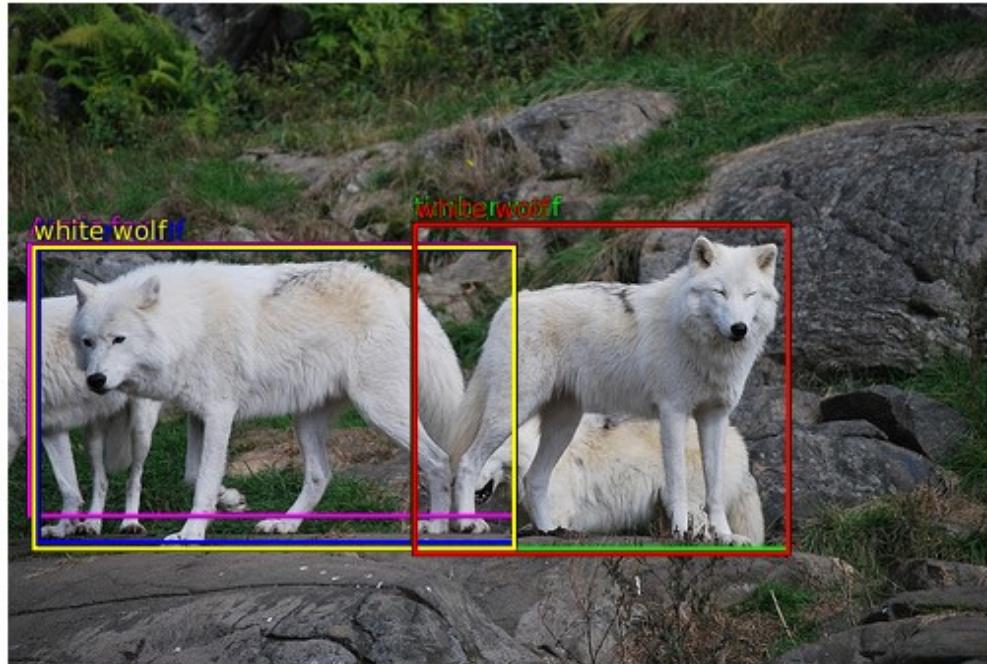
■ Top-5 error rates: if correct class is not in top 5, count as error

▶ Red:ConvNet, blue: no ConvNet

2012 Teams	%error	2013 Teams	%error	2014 Teams	%error
Supervision (Toronto)	15.3	Clarifai (NYU spinoff)	11.7	GoogLeNet	6.6
ISI (Tokyo)	26.1	NUS (singapore)	12.9	VGG (Oxford)	7.3
VGG (Oxford)	26.9	Zeiler-Fergus (NYU)	13.5	MSRA	8.0
XRCE/INRIA	27.0	A. Howard	13.5	A. Howard	8.1
UvA (Amsterdam)	29.6	OverFeat (NYU)	14.1	DeeperVision	9.5
INRIA/LEAR	33.4	UvA (Amsterdam)	14.2	NUS-BST	9.7
		Adobe	15.2	TTIC-ECP	10.2
		VGG (Oxford)	15.2	XYZ	11.2
		VGG (Oxford)	23.0	UvA	12.1

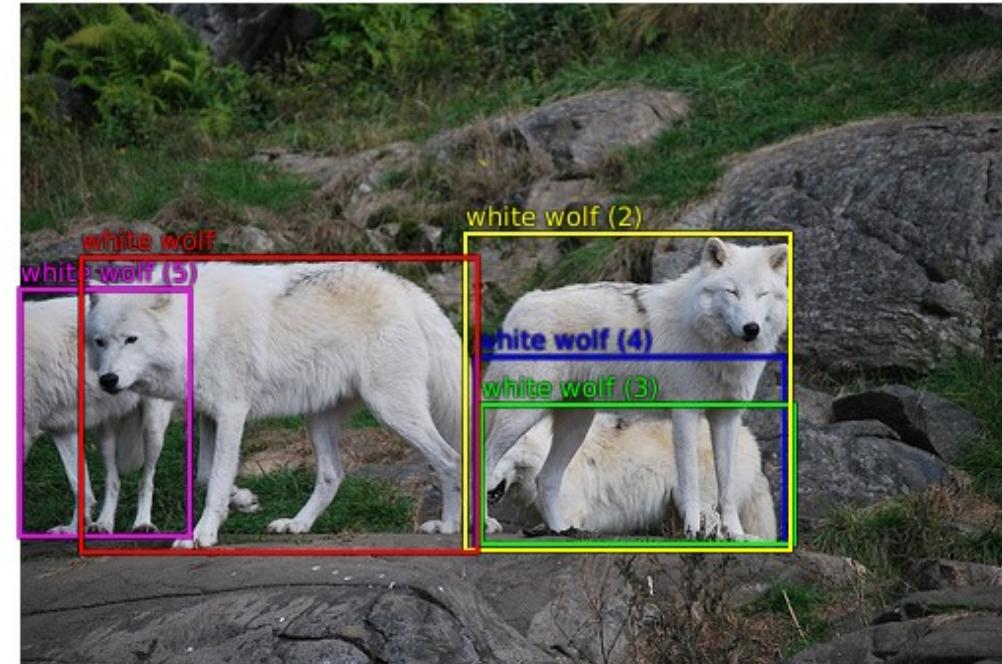
Classification+Localization. Results

Y LeCun



Top 5:

white wolf
white wolf
timber wolf
timber wolf
Arctic fox



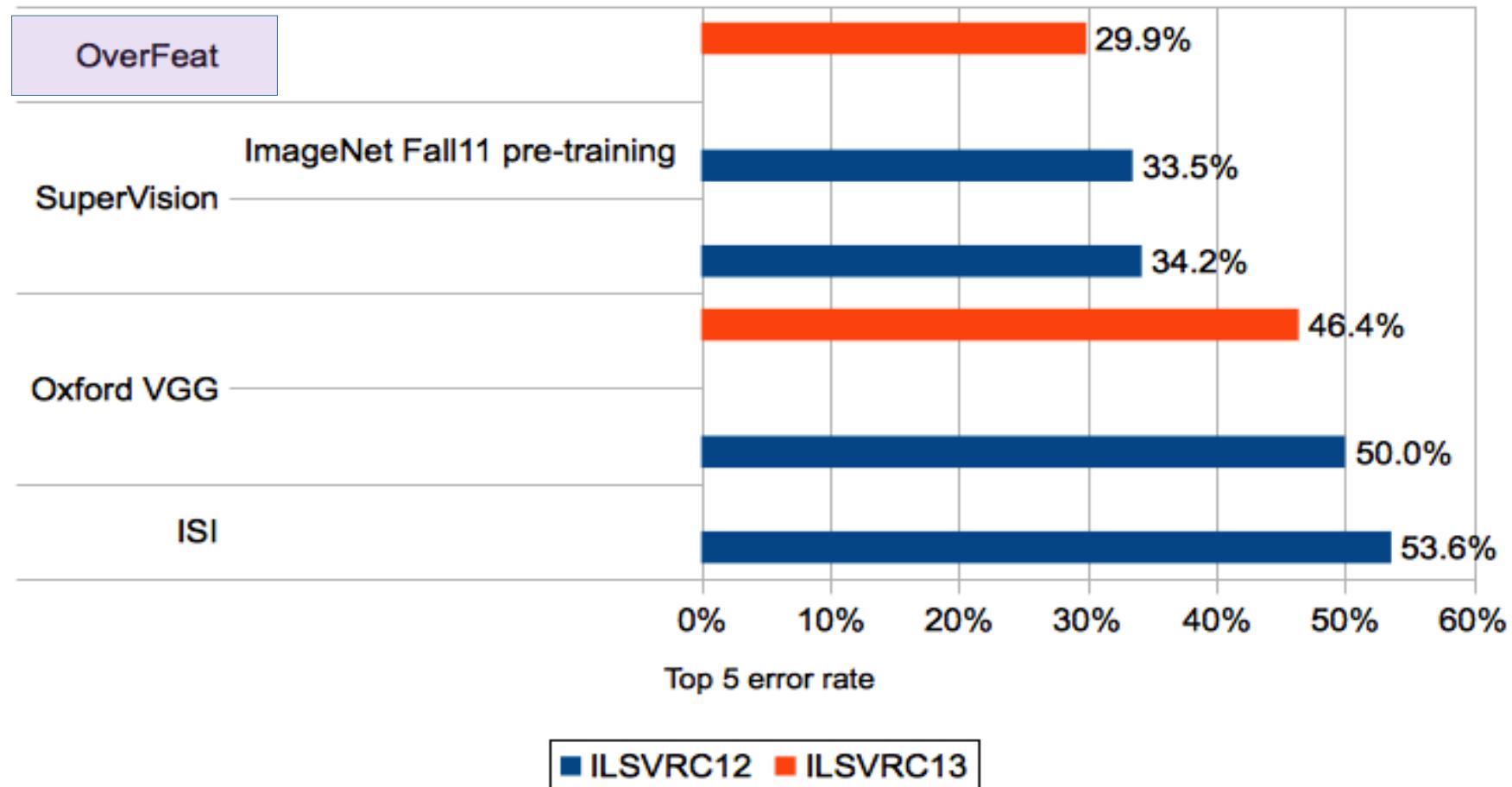
Groundtruth:

white wolf
white wolf (2)
white wolf (3)
white wolf (4)
white wolf (5)

Classification+Localization. Error Rates

Y LeCun

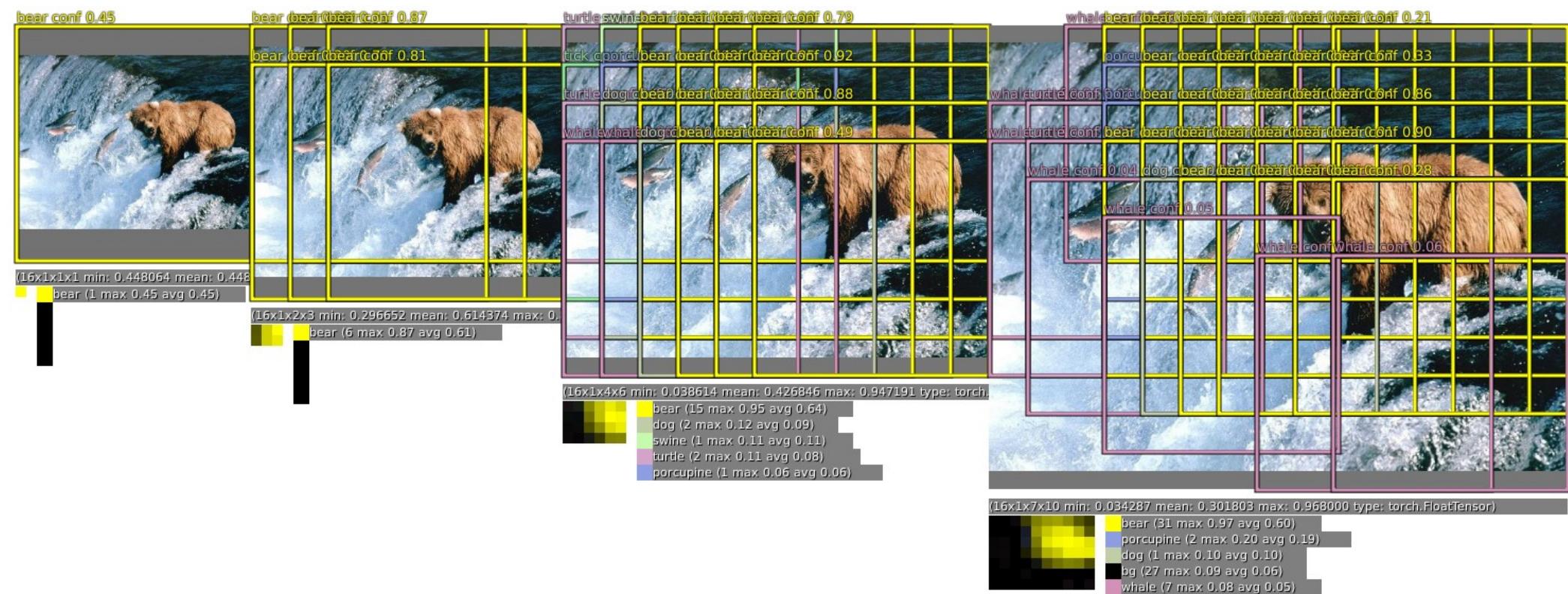
- It's best to propose several categories for the same window
 - ▶ One of them might be right
- 2014 results: 25.3% (VGG Oxford), 26.4% (GoogLeNet)



Classification + Localization: multiscale sliding window

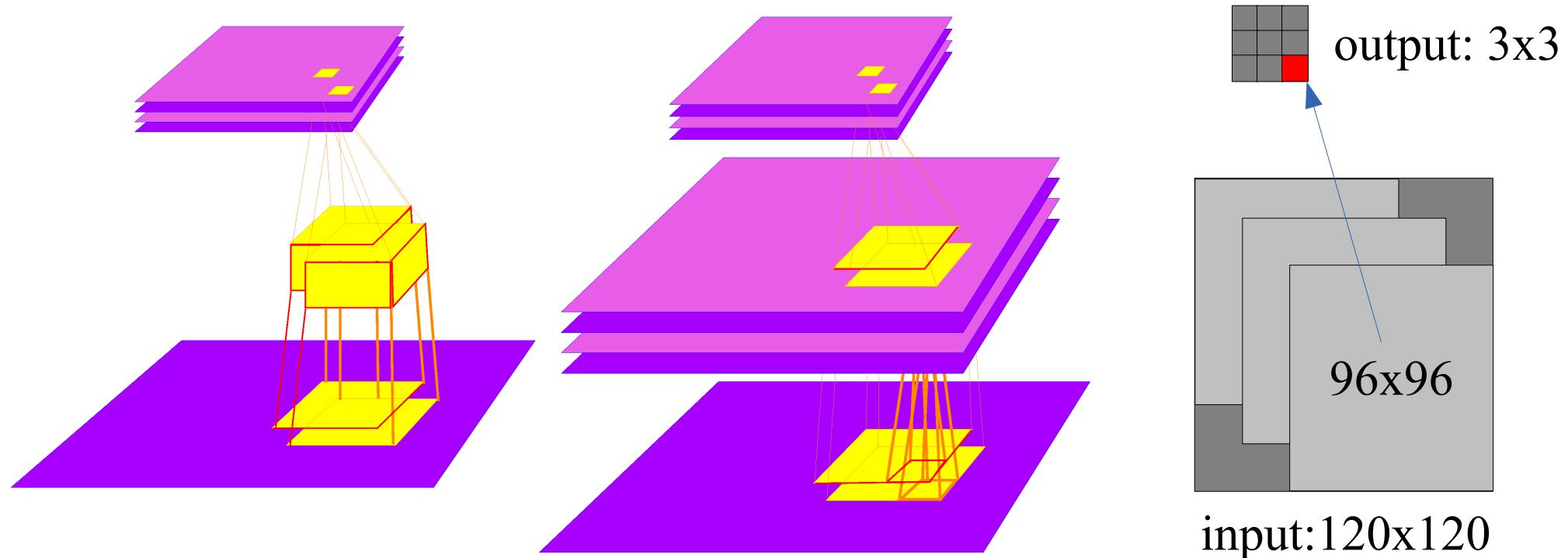
Y LeCun

- Apply convnet with a sliding window over the image at multiple scales
- Important note: it's very cheap to slide a convnet over an image
 - ▶ Just compute the convolutions over the whole image and replicate the fully-connected layers



Applying a ConvNet on Sliding Windows is Very Cheap!

Y LeCun

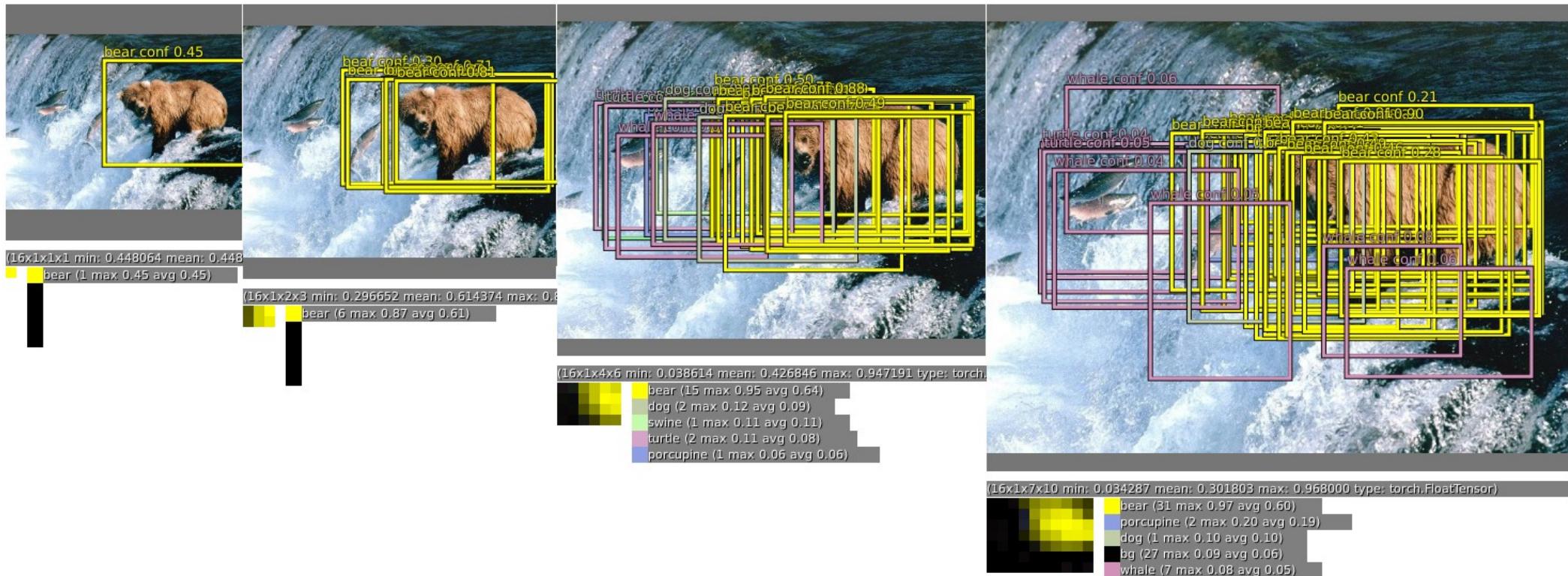


- Traditional Detectors/Classifiers must be applied to every location on a large input image, at multiple scales.
- Convolutional nets can replicated over large images very cheaply.
- Simply apply the convolutions to the entire image and spatially replicate the fully-connected layers**

Classification + Localization: sliding window + bounding box regression

Y LeCun

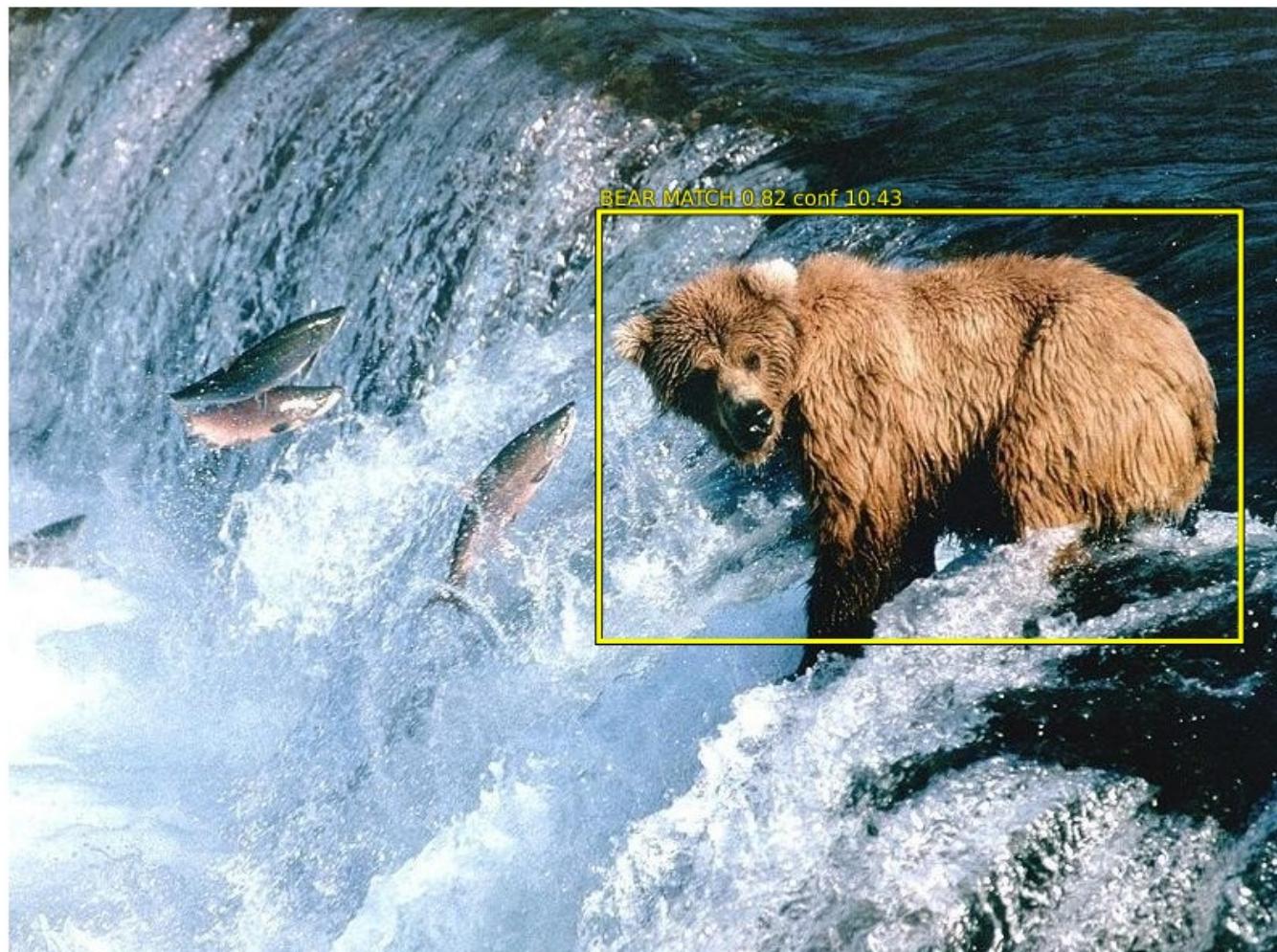
- Apply convnet with a sliding window over the image at multiple scales
- For each window, predict a class and bounding box parameters
 - ▶ Even if the object is not completely contained in the viewing window, the convnet can predict where it thinks the object is.



Classification + Localization: sliding window + bounding box regression + bbox voting

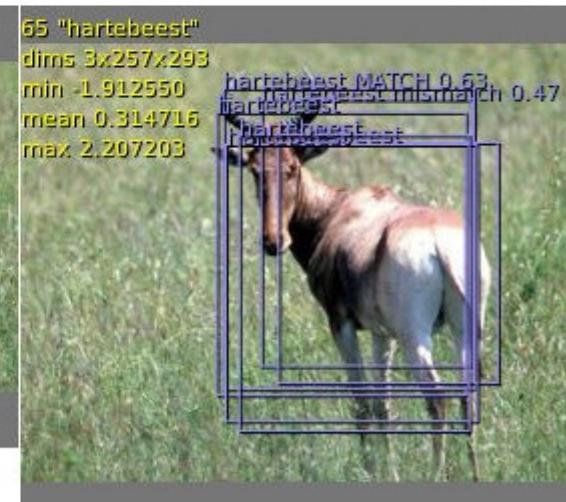
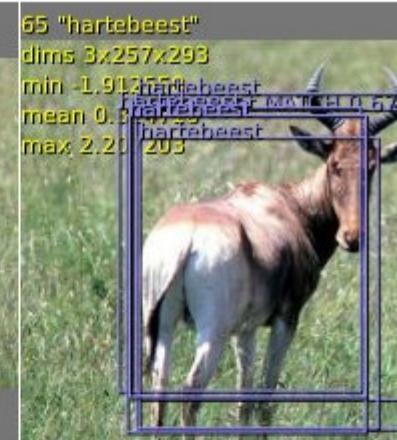
Y LeCun

- Apply convnet with a sliding window over the image at multiple scales
- For each window, predict a class and bounding box parameters
- Compute an “average” bounding box, weighted by scores

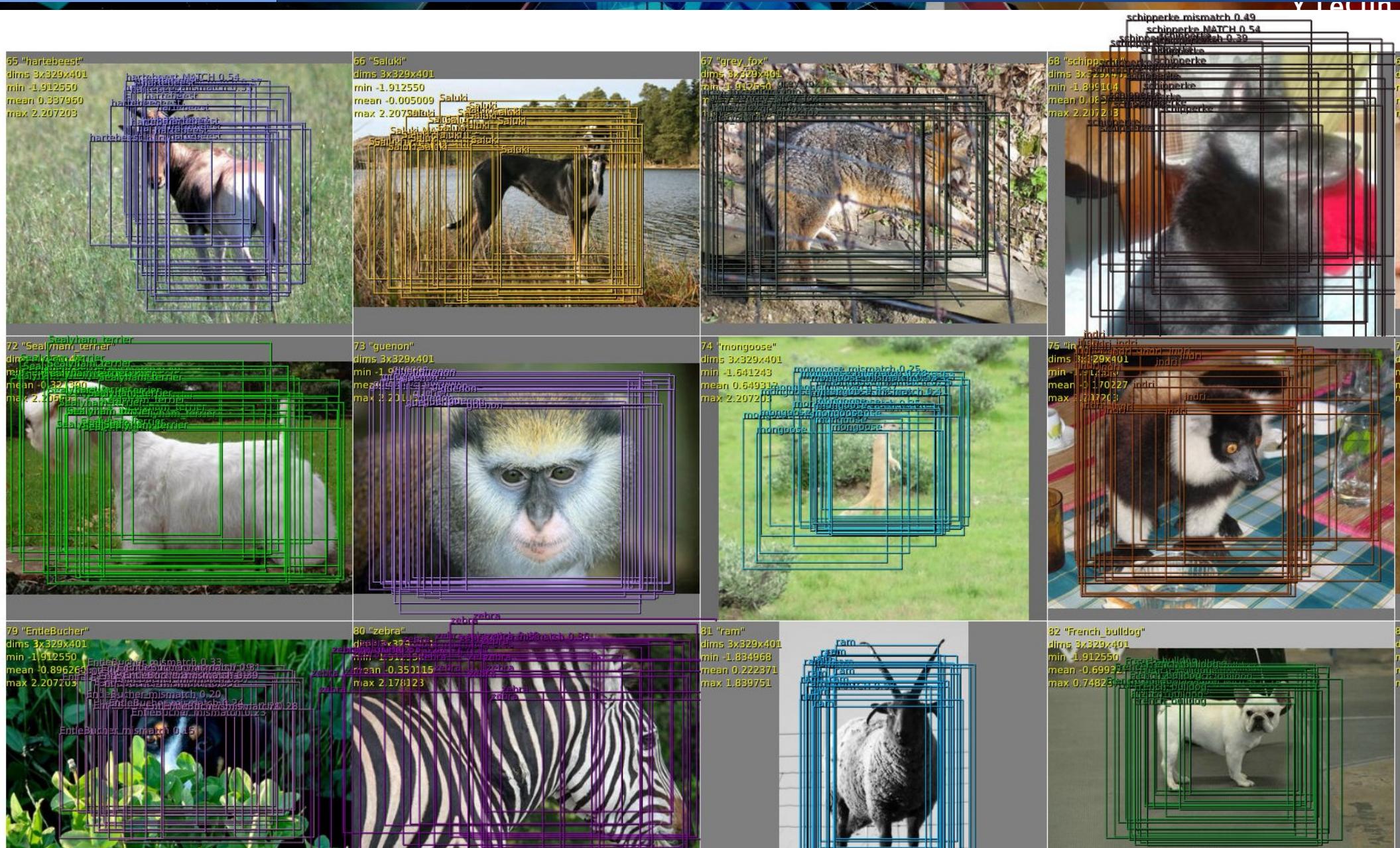


Localization: Sliding Window + bbox vote + multiscale

Y LeCun

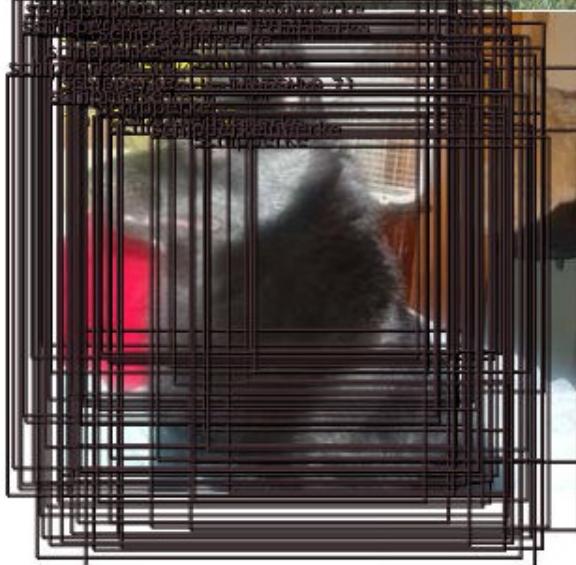
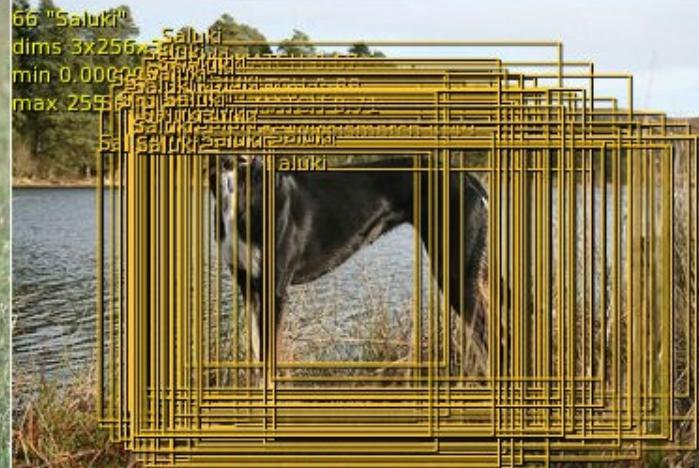


Detection / Localization

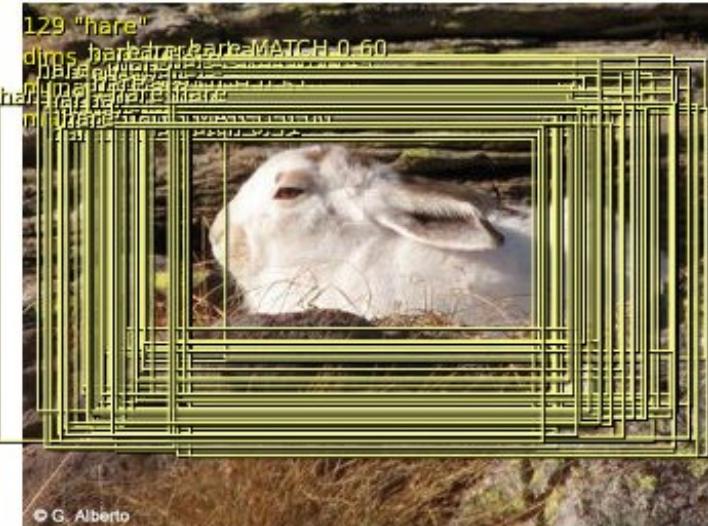


Detection / Localization

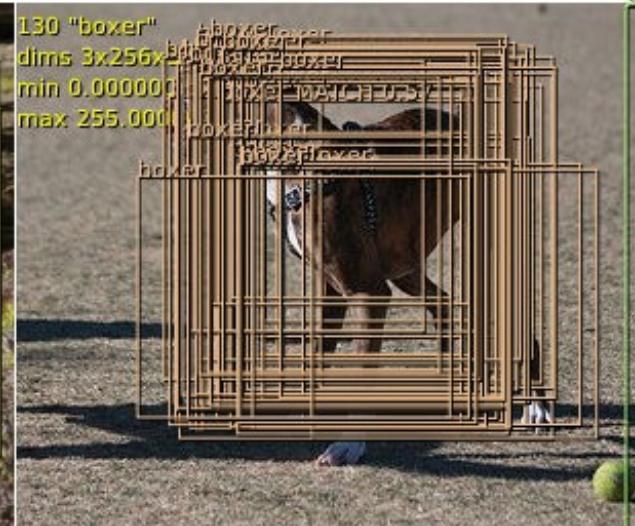
Y Lecun



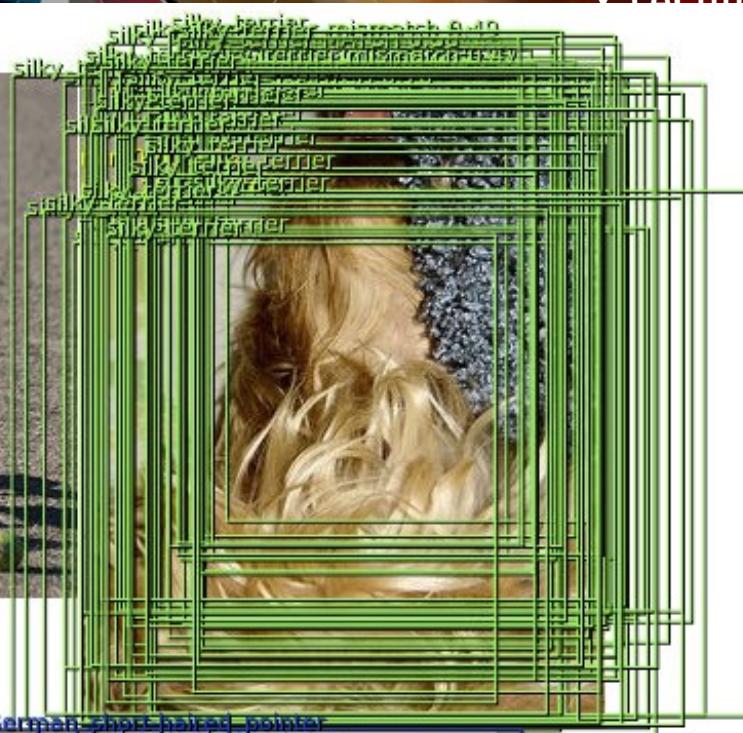
Detection / Localization



A close-up photograph of a light brown dog's face, looking slightly to the left. The dog has dark brown eyes and a white patch on its chest. The image is framed by a thick red rectangular border.



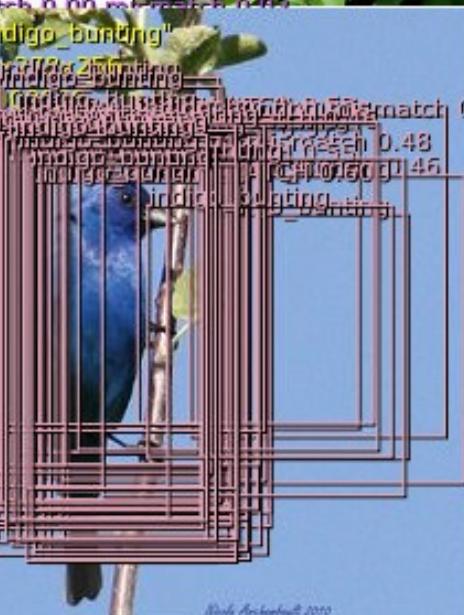
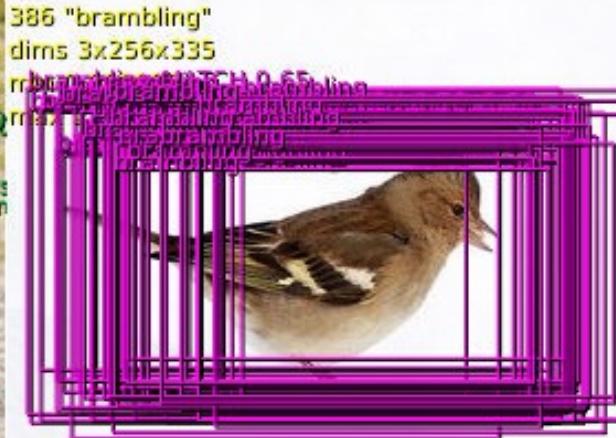
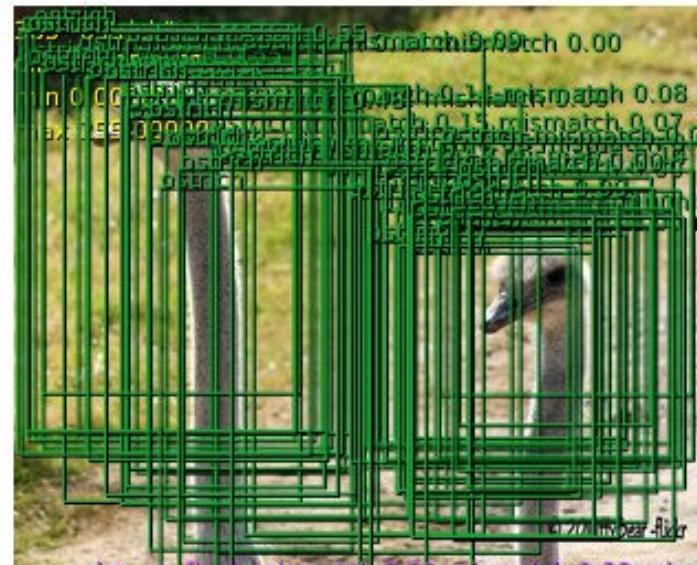
A close-up photograph of a Leonberger dog's face. The dog has a mix of brown and black fur, with darker areas around its eyes and ears. It is looking slightly to the right of the camera. The background is blurred, showing some greenery. The entire image is surrounded by a thick, decorative blue border.



German short-haired pointer

Detection / Localization

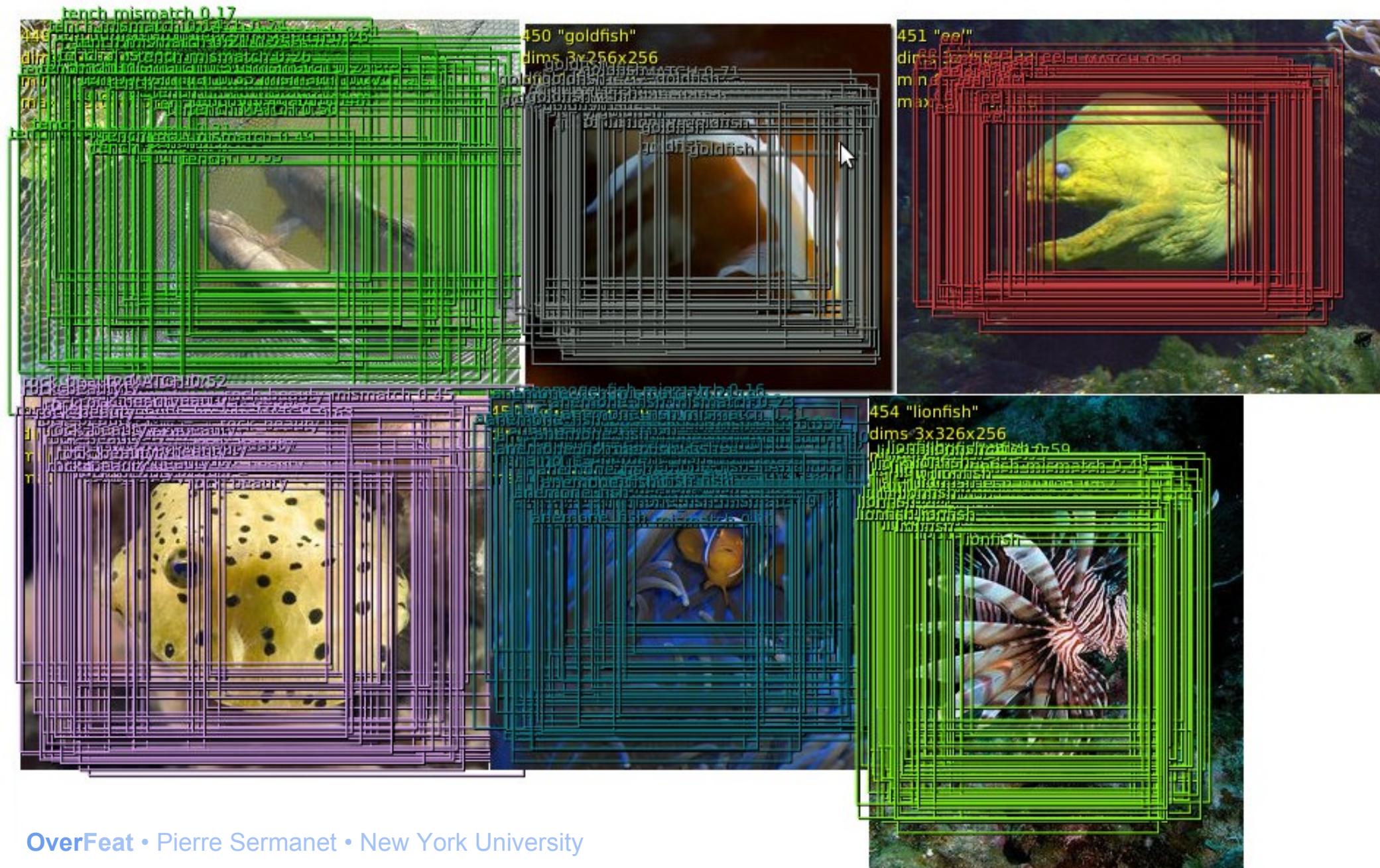
Y Lecun



16

Detection / Localization

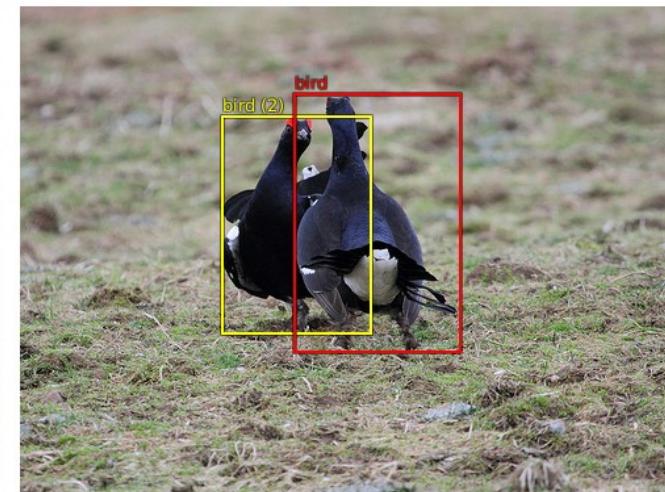
Y Lecun



Detection: Examples

Y LeCun

- 200 broad categories
 - There is a penalty for false positives
 - Some examples are easy some are impossible/ambiguous
 - Some classes are well detected
- Burritos?



Top predictions:

bird (confidence 86.0)
bird (confidence 70.9)

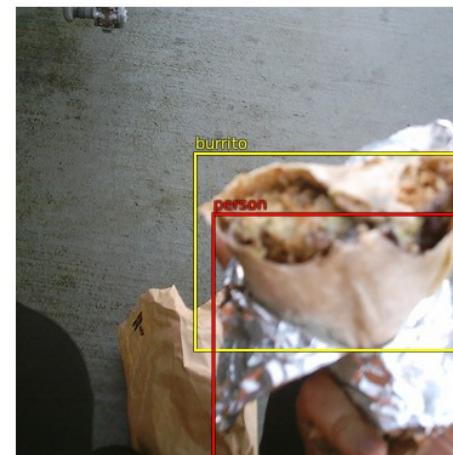
ILSVRC2012_val_00001136.jpeg

Groundtruth:



Top predictions:
burrito (confidence 28.9)

ILSVRC2012_val_00000572.jpeg



Groundtruth:
person
burrito



Top predictions:
burrito (confidence 17.4)

ILSVRC2012_val_00000606.jpeg



Groundtruth:
burrito
burrito (2)

Detection: Examples

Y LeCun

- Groundtruth is sometimes ambiguous or incomplete
- Large overlap between objects stops non-max suppression from working



Top predictions:

tv or monitor (confidence 11.5)
person (confidence 4.5)
miniskirt (confidence 3.1)

ILSVRC2012_val_00000119.JPG



Groundtruth:

tv or monitor
tv or monitor (2)
tv or monitor (3)
person
remote control
remote control (2)

ImageNet: Detection (200 categories)

Y LeCun

■ Give a bounding box and a category for all objects in the image

■ MAP = mean average precision

► Red:ConvNet, blue: no ConvNet

2013 Teams	mAP
UvA Amsterdam	22.6
NEC Labs-America	20.9
OverFeat NYU	19.4

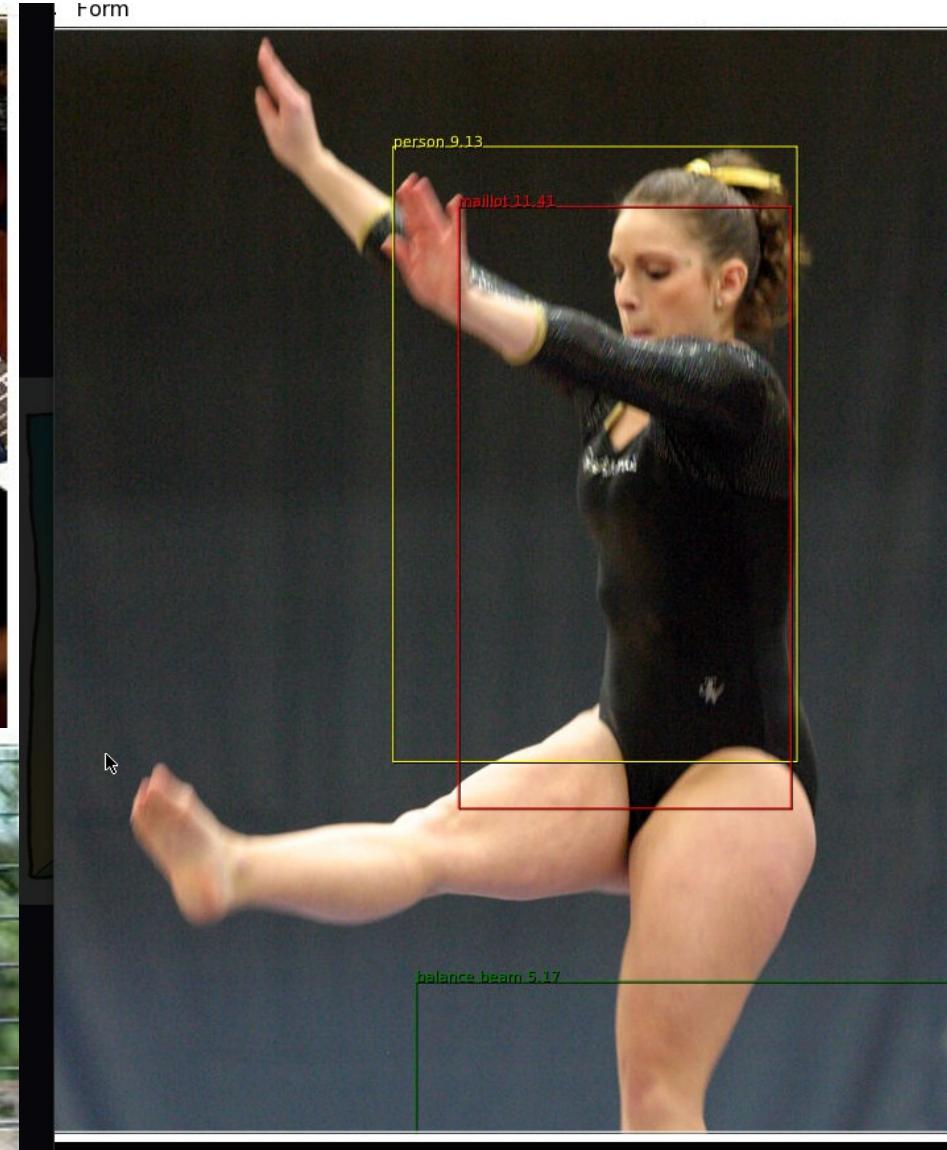
Off cycle results	mAP
Berkeley RCNN	34.5
OverFeat NYU	24.3

2014 Teams	mAP
GoogLeNet	43.9
CUHK-DeepID2	40.7
DeepInsight	40.4
NUS	37.2
UvA Amsterdam	35.4
MSRA	35.1
Berkeley RCNN	34.5

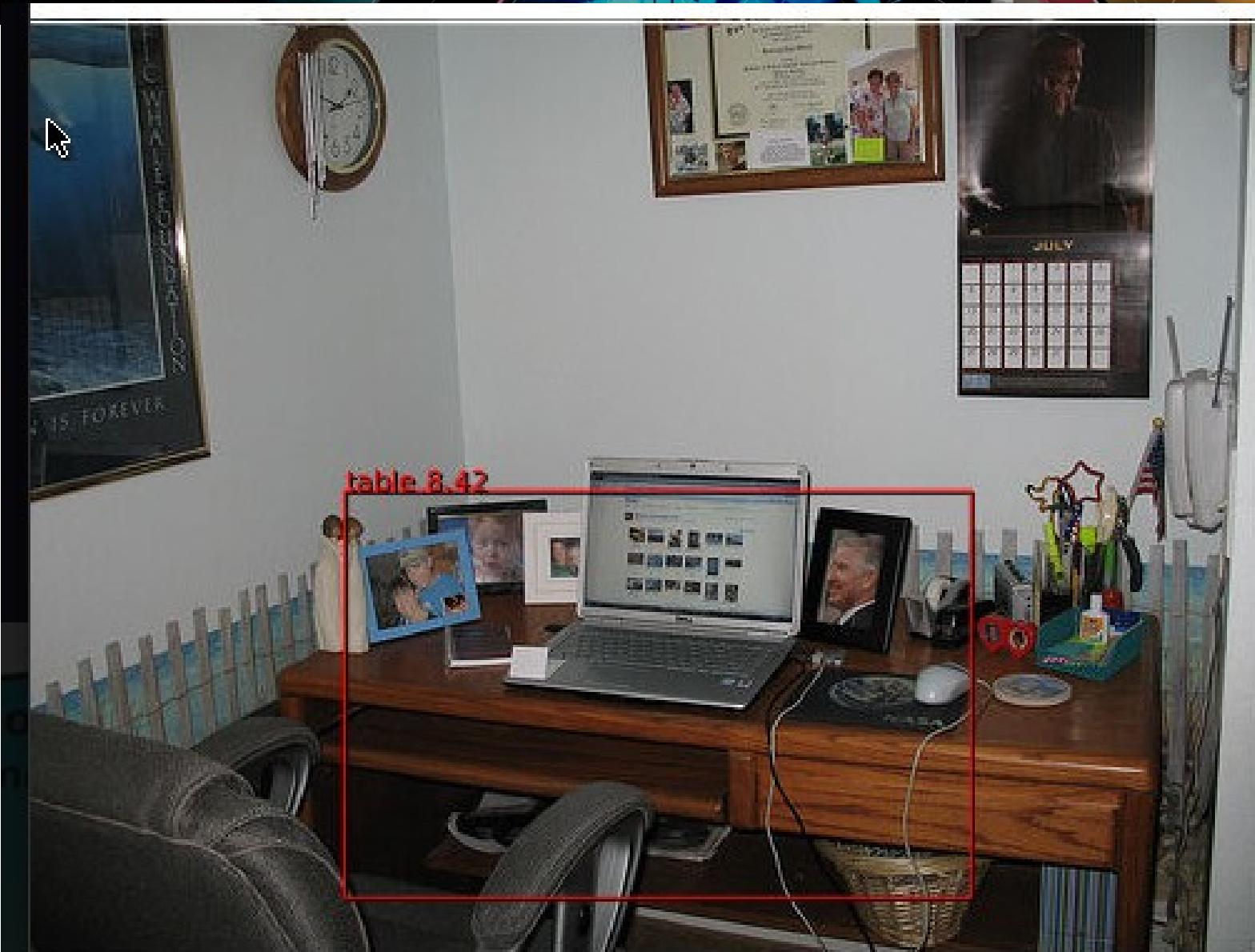


Results: pre-trained on ImageNet1K, fine-tuned on ImageNet Detection

Y LeCun



Detection Examples



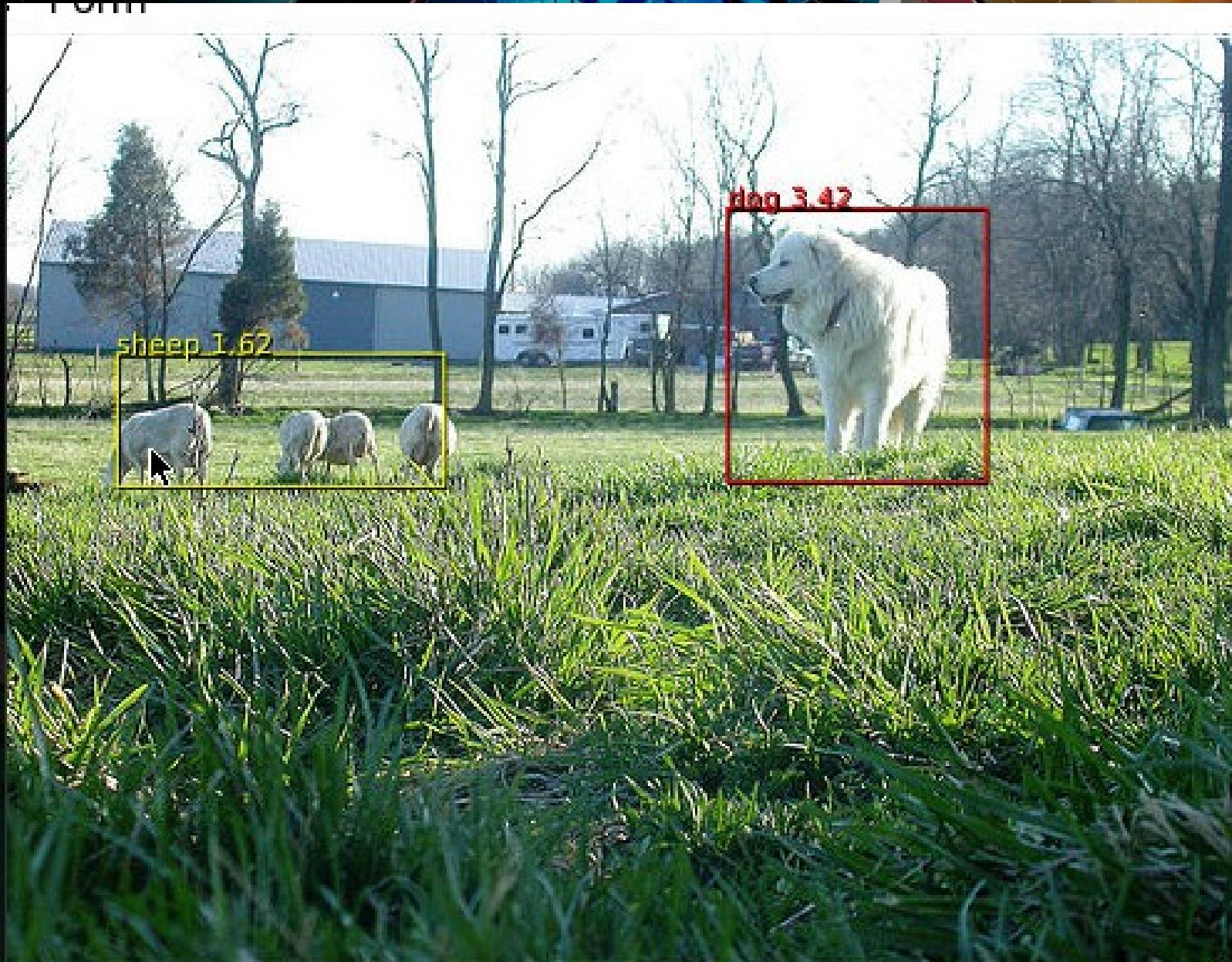
/home/snwiz/data/imagenet12/original/det/ILSVRC2013_DET_val/ILSVRC2012_val_00006665.JPG
table conf 8.416754

Detection Examples

Y Lecun



Detection Examples



/home/snwiz/data/imagenet12/original/det/ILSVRC2013_DET_test/ILSVRC2012_test_00090628.JPG
dog conf 3.419652
sheep conf 1.616341

Detection Examples

y Lecun

Form

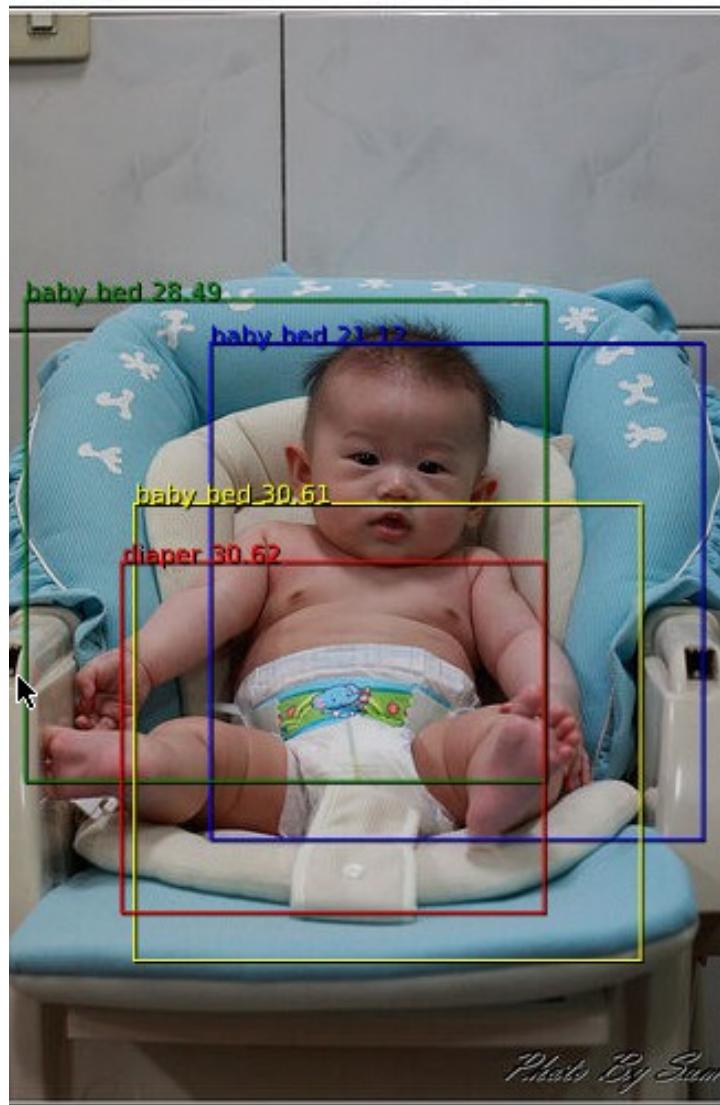


/home/snwiz/data/imagenet12/original/det/ILSVRC2013_DET_test/ILSVRC2012_test_00091048.JPG

person conf 17.898635

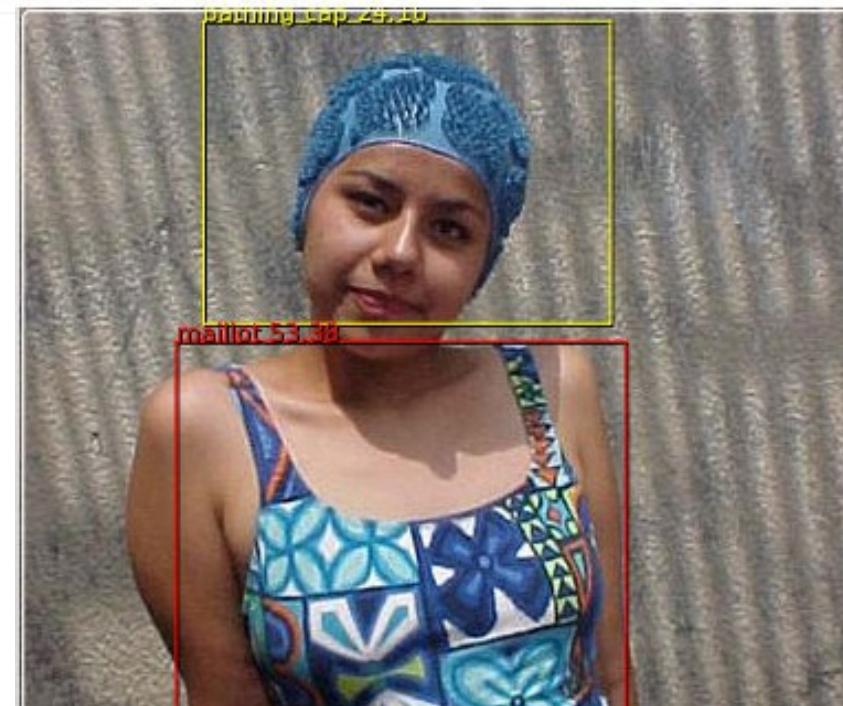
bow conf 15.628116

Detection Examples

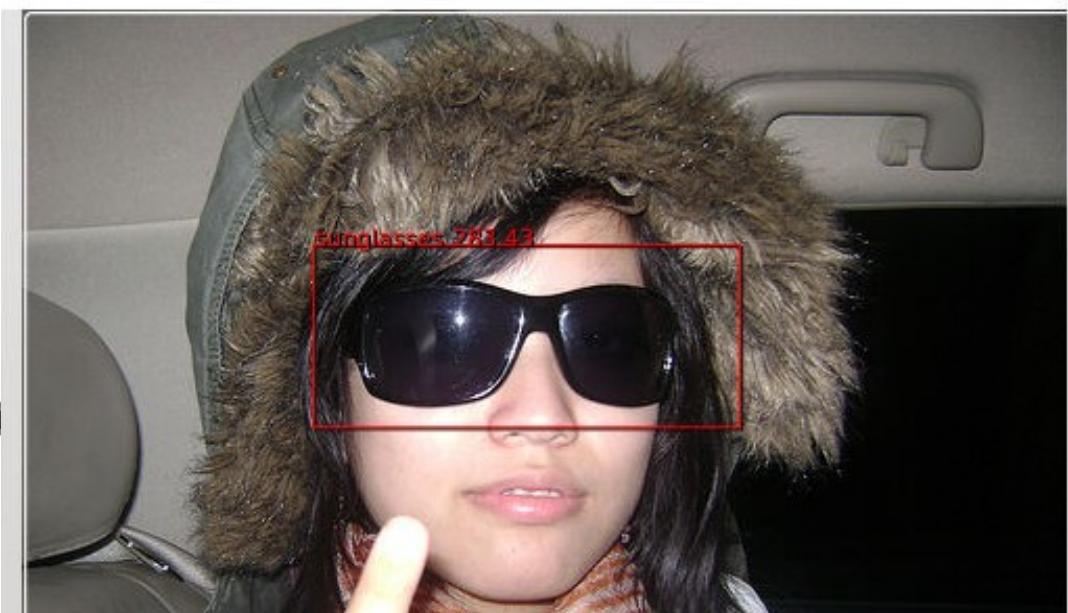


/home/snwiz/data/imagenet12/original/det/ILSVRC2013_DET_test/
diaper conf 30.616426
baby bed conf 30.607744
baby bed conf 28.493934
baby bed conf 21.117424

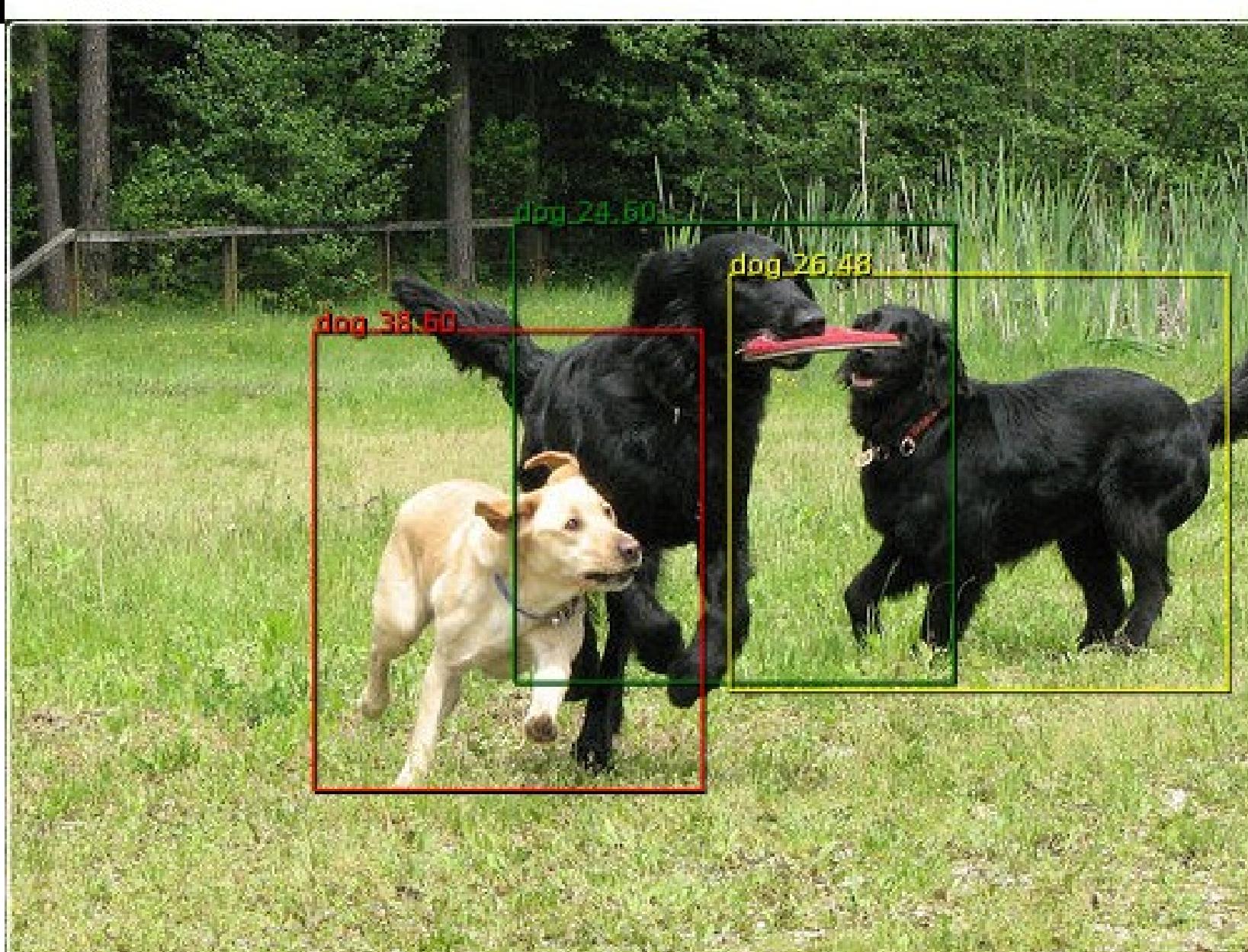
Form



Form



Detection Examples



/home/snwiz/data/imagenet12/original/det/ILSVRC2013_DET_test/ILSVRC2012_test_00000172.JPG
dog conf 38.603936

Detection Examples

Form



Detection: Difficult Examples

Y LeCun

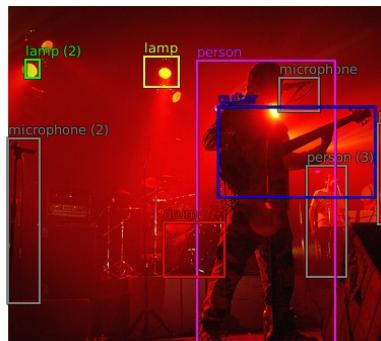
Groundtruth is sometimes ambiguous or incomplete



Top predictions:
microwave (confidence 5.6)
refrigerator (confidence 2.5)



Groundtruth:
bowl
microwave



Top predictions:
person (confidence 6.0)

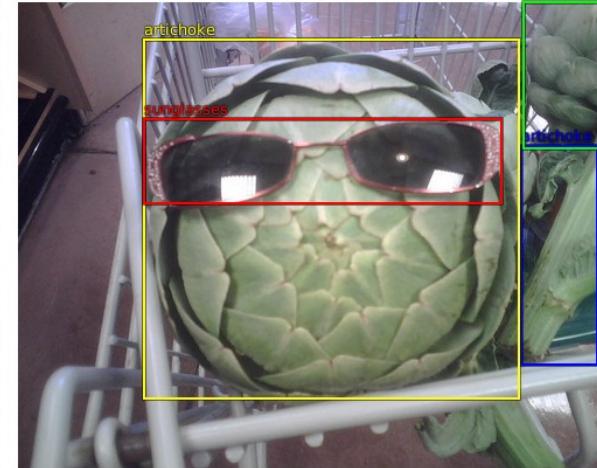
ILSVRC2012_val_00001273.jpeg

Groundtruth:
drum
lamp
lamp (2)
guitar
person
person (2)
person (3)
microphone
microphone (2)
microphone (3)



Top predictions:
artichoke (confidence 162.8)

ILSVRC2012_val_00001549.jpeg



Groundtruth:
sunglasses
artichoke
artichoke (2)
artichoke (3)

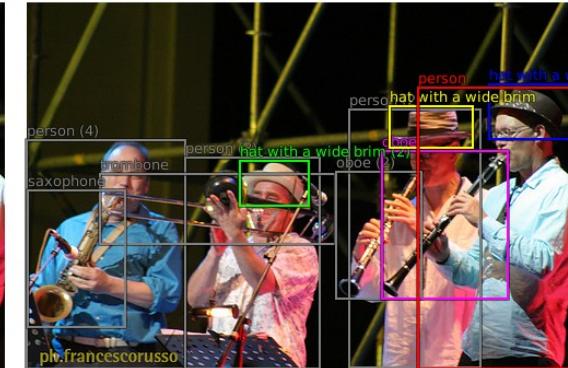
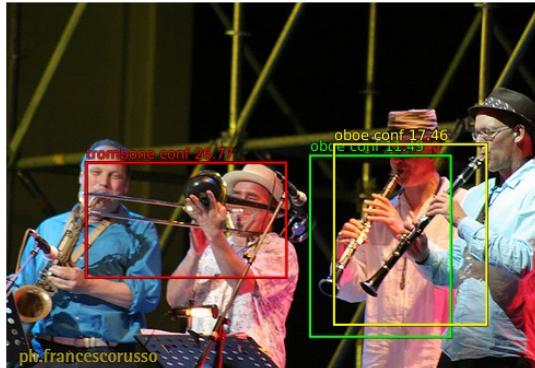
Detection: Difficult Examples

Y LeCun

■ Non-max suppression makes us miss many objects

▶ Person behind instrument

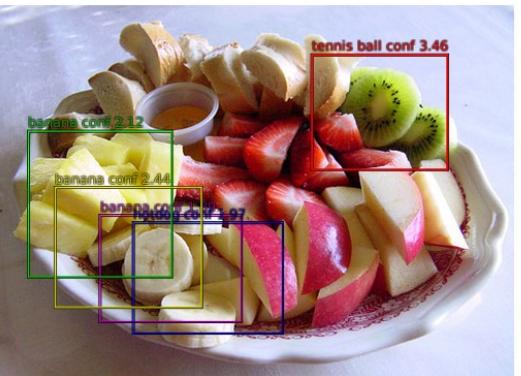
■ A bit of contextual post-processing would fix many errors



Top predictions:
trombone (confidence 26.8)
oboe (confidence 17.5)
oboe (confidence 11.5)

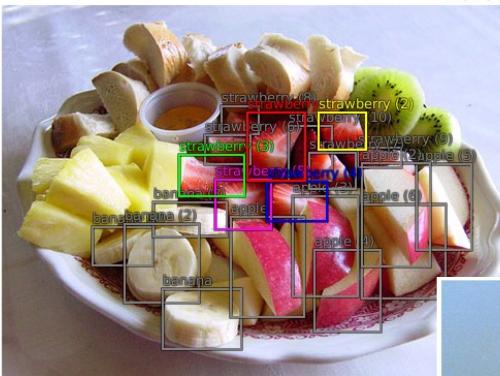
ILSVRC2012_val_00000614.jpeg

Groundtruth:
person
hat with a wide brim
hat with a wide brim (2)
hat with a wide brim (3)
oboe
oboe (2)
saxophone
trombone
person (2)
person (3)
person (4)



Top predictions:
tennis ball (confidence 3.5)
banana (confidence 2.4)
banana (confidence 2.1)
hotdog (confidence 2.0)
banana (confidence 1.9)

ILSVRC2012_val_00000320.jpeg

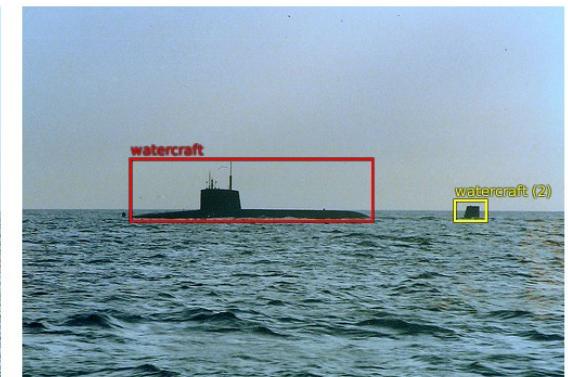


Groundtruth:
strawberry
strawberry (2)
strawberry (3)
strawberry (4)
strawberry (5)
strawberry (6)
strawberry (7)
strawberry (8)
strawberry (9)
strawberry (10)
apple
apple (2)
apple (3)



Top predictions:
watercraft (confidence 72.2)
watercraft (confidence 2.1)

ILSVRC2012_val_00000623.jpeg



Groundtruth:
watercraft
watercraft (2)

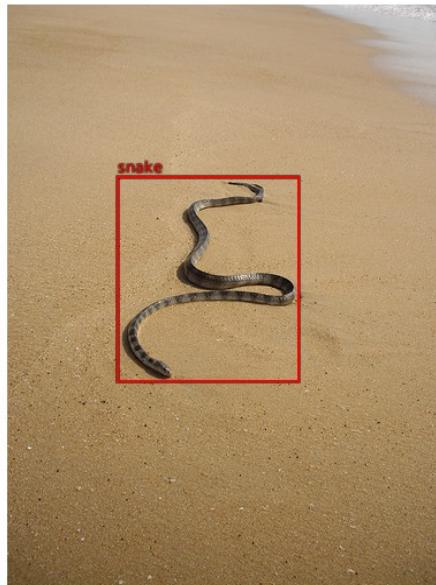
Detection: Interesting Failures

Y LeCun



Top predictions:
corkscrew (confidence 38.1)

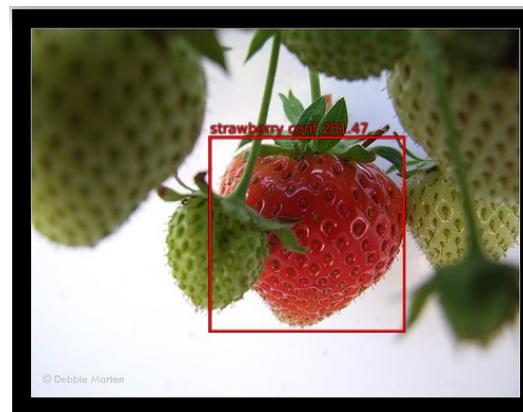
ILSVRC2012_val_00000324.jpeg



Groundtruth:
snake

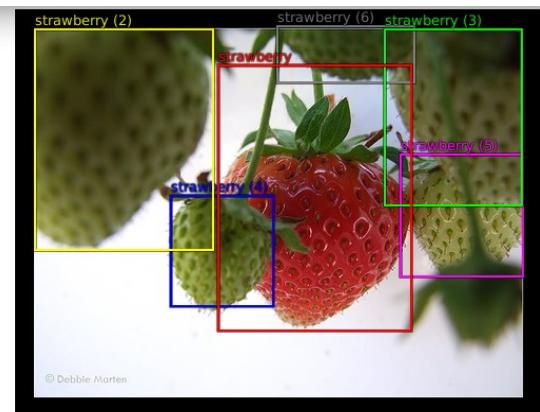


Snake → Corkscrew



Top predictions:
strawberry (confidence 201.5)

ILSVRC2012_val_00000099.jpeg



Groundtruth:
strawberry
strawberry (2)
strawberry (3)
strawberry (4)
strawberry (5)



Top predictions:
remote control (confidence 31.8)
filling cabinet (confidence 2.2)

ILSVRC2012_val_00000331.jpeg

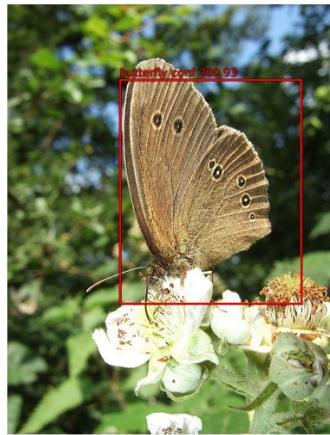


Groundtruth:
table
water bottle
water bottle (2)
water bottle (3)
water bottle (4)
refrigerator

Detection: Bad Groundtruth

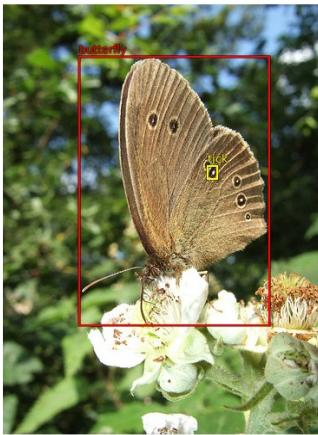
Y LeCun

One of the labelers likes ticks.....



Top predictions:
butterfly (confidence 200.9)

ILSVRC2012_val_00035074.jpeg

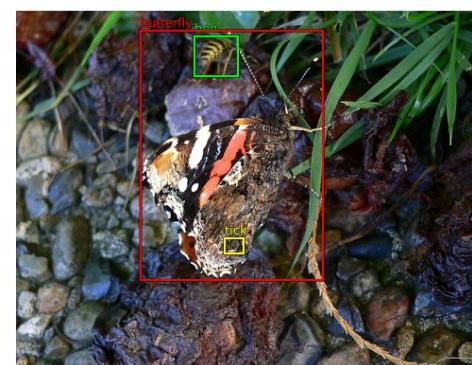


Groundtruth:
butterfly
tick



Top predictions:
butterfly (confidence 15.8)

ILSVRC2012_val_00012764.jpeg



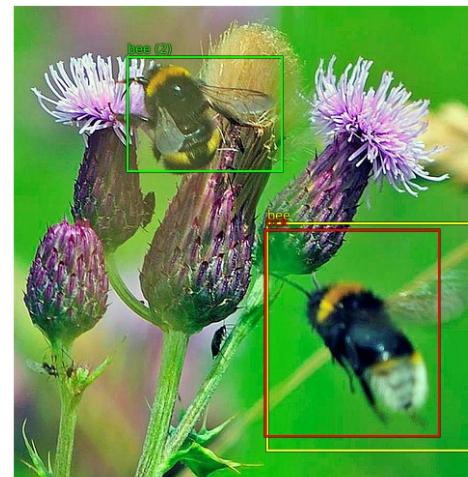
Groundtruth:
butterfly
tick
bee



Groundtruth:
butterfly
tick



Groundtruth:
tick
isopod



Groundtruth:
tick
bee
bee (2)



Top predictions:
snail (confidence 33.8)

ILSVRC2012_val_00023206.jpeg



Groundtruth:
snail
tick

ConvNets As Generic Feature Extractors

- Kaggle competition: Dog vs Cats



- Won by Pierre Sermanet (NYU):
- ImageNet network (OverFeat) last layers retrained on cats and dogs



Dogs vs. Cats

Finished

Wednesday, September 25, 2013

Swag • 215 teams

Saturday, February 1, 2014

Dashboard

Leaderboard - Dogs vs. Cats

This competition has completed. This leaderboard reflects the final standings.

See someone using multiple accounts?

[Let us know.](#)

#	Δ1w	Team Name * <small>in the money</small>	Score	Entries	Last Submission UTC	(Best - Last Submission)
1	-	Pierre Sermanet *	0.98914	5	Sat, 01 Feb 2014 21:43:19	(-1.2h)
2	↑26	orchid *	0.98309	17	Sat, 01 Feb 2014 23:52:30	
3	-	Owen	0.98171	15	Sat, 01 Feb 2014 17:04:40	(-1.3h)
4	new	Paul Covington	0.98171	3	Sat, 01 Feb 2014 23:05:20	
5	↓3	Maxim Milakov	0.98137	24	Sat, 01 Feb 2014 18:20:58	

OverFeat Features ->Trained Classifier on other datasets

Y LeCun

A. S. Razavian , H. Azizpour , J. Sullivan , S. Carlsson "CNN features off-the-shelf: An astounding baseline for recogniton", CVPR 2014, DeepVision Workshop.

<http://www.csc.kth.se/cvap/cvg/DL/ots/>

Comparing Best State of the Art Methods with Deep Represenations																				
	VOC07c	VOC12c	VOC12a	MIT67	SUN397	VOC07d	VOC10d	VOC11s	200Birds	102Flowers	H3Datt	UIUCatt	LFW	YTF	Paris6k	Oxford5k	Sculp6k	Holidays	UKB	
best non-CNN results	70.5	82.2	69.6	64.0	47.2	34.3	40.4	47.6	56.8	80.7	69.9	~90.0	96.3	89.4	78.2	81.7	45.4	82.2	89.3	
off-the-shelf ImageNet Model	80.13[10] 77.2[1]	82.7[10] 79.0[6]	-	69.0[1]	40.9[4]	46.2[2] 46.1[11]	44.1[11]	-	61.8[1] 58.8[4]	86.8[1]	73.0[1]	91.5[1]	-	-	79.5[1]	68.0[1]	42.3[1]	84.3[1]	91.1[1]	
off-the-shelf ImageNet Model + rep learning	-	-	-	68.9[3]	52.0[3]	-	-	-	65.0[4]	-	-	-	-	-	-	-	80.2[3]	-	-	
fine-tuned ImageNet Model	82.42[10] 77.7[5]	83.2[10] 82.8[5]	70.2[5]	-	-	58.5[2]	53.7[2]	47.9[2]	-	-	-	-	-	-	-	-	-	-	-	
Other Deep Learning Models	-	-	-	-	-	-	-	-	-	-	79.0[7]	-	97.35[8]	91.4[8]	-	-	-	-	-	

VOC07c:

Pascal VOC 2007 (Object Image Classification)

VOC12c:

Pascal VOC 2012 (Object Image Classification)

VOC12a:

Pascal VOC 2012 (Action Image Classification)

MIT67:

MIT 67 Indoor Scenes(Scene Image Classification)

VOC07d:

PASCAL VOC 2007 (Object Detection)

VOC10d:

PASCAL VOC 2010 (Object Detection)

VOC12d:

PASCAL VOC 2012 (Object Detection)

VOC11s:

PASCAL VOC 2011 (Object Category Segmentation)

200Birds:

UCSD-Caltech 2011-200 Birds dataset (Fine-grained Recognition)

102Flowers:

Oxford 102 Flowers (Fine-grained Recognition)

H3Datt:

H3D poselets Human 9 Attributes (Attribute Detection)

UIUCatt:

UIUC object attributes (Attribute Detection)

LFW:

Labelled Faces in the Wild (Metric Learning)

Oxford5k:

Oxford 5k Buildings Dataset (Instance Retrieval)

Paris6k:

Paris 6k Buildings Dataset (Instance Retrieval)

Sculp6k:

Oxford Sculptures Dataset (Instance Retrieval)

Holidays:

INRIA Holidays Scenes Dataset (Instance Retrieval)

UKB:

Uni. of Kentucky Retrieval Benchmark Dataset (Instance Retrieval)

OverFeat Features + Classifier on various datasets

Yi Li

Dataset	Performance	Score
[Sermanet et al 2014]: OverFeat (fine-tuned features for each task) (tasks are ordered by increasing difficulty)		
• image classification	ImageNet LSVRC 2013	competitive
• object localization	Dogs vs Cats Kaggle challenge 2014	state of the art
• object detection	ImageNet LSVRC 2013	state of the art
	ImageNet LSVRC 2013	state of the art
[Razavian et al, 2014]: public OverFeat library (no retraining) + SVM <u>(simplest approach possible on purpose, no attempt at more complex classifiers)</u> (tasks are ordered by “distance” from classification task on which OverFeat was trained)		
• image classification	Pascal VOC 2007	competitive
• scene recognition	MIT-67	competitive
• fine grained recognition	Caltech-UCSD Birds 200-2011	competitive
• attribute detection	Oxford 102 Flowers	competitive
• image retrieval (search by image similarity)	UIUC 64 object attributes	state of the art
	H3D Human Attributes	state of the art
	Oxford 5k buildings	?
	Paris 6k buildings	?
	Sculp6k	?
	Holidays	competitive
	UKBench	relatively poor

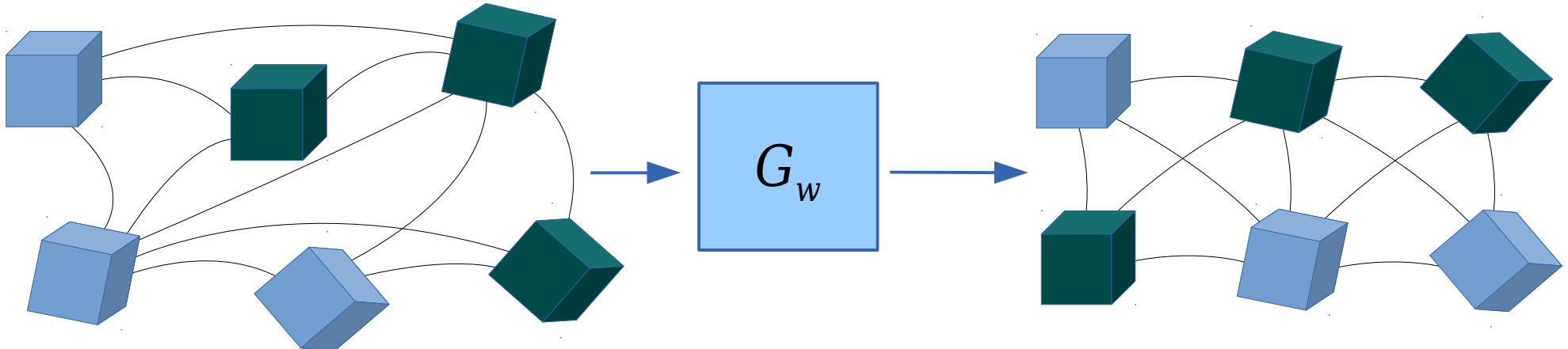
Image Similarity Matching With Siamese Networks Embedding, DrLIM

DrLIM: Metric Learning

Y LeCun

Dimensionality Reduction by Learning an Invariant Mapping

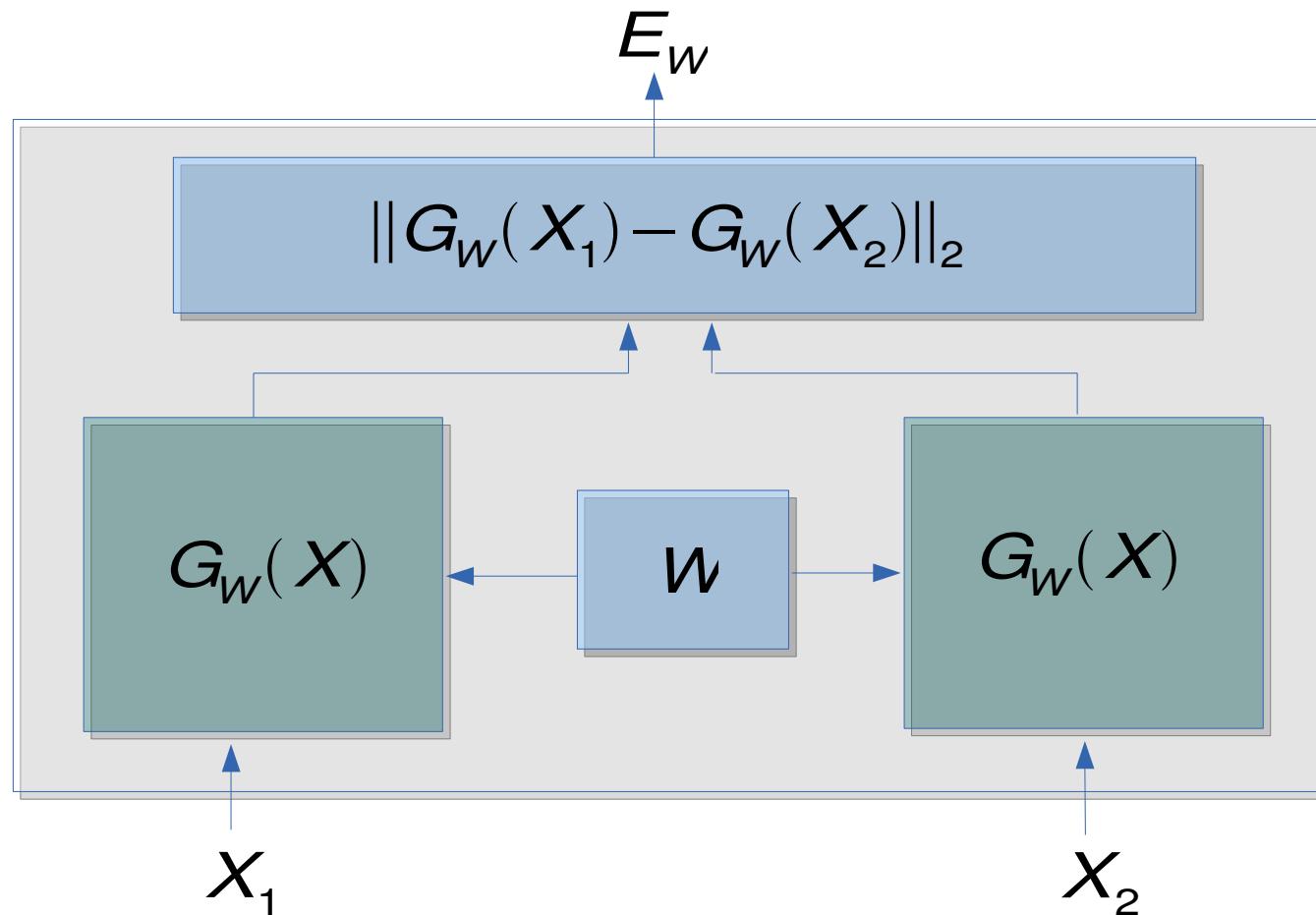
- **Step 1:** Construct neighborhood graph.
- **Step 2:** Choose a parameterized family of functions.
- **Step 3:** Optimize the parameters such that:
 - Outputs for **similar samples** are pulled closer.
 - Outputs for **dissimilar samples** are pushed away.



joint work with Sumit Chopra: Hadsell et al. CVPR 06; Chopra et al., CVPR 05

Siamese Architecture

Y LeCun



Siamese Architecture [Bromley, Sackinger, Shah, LeCun 1994]

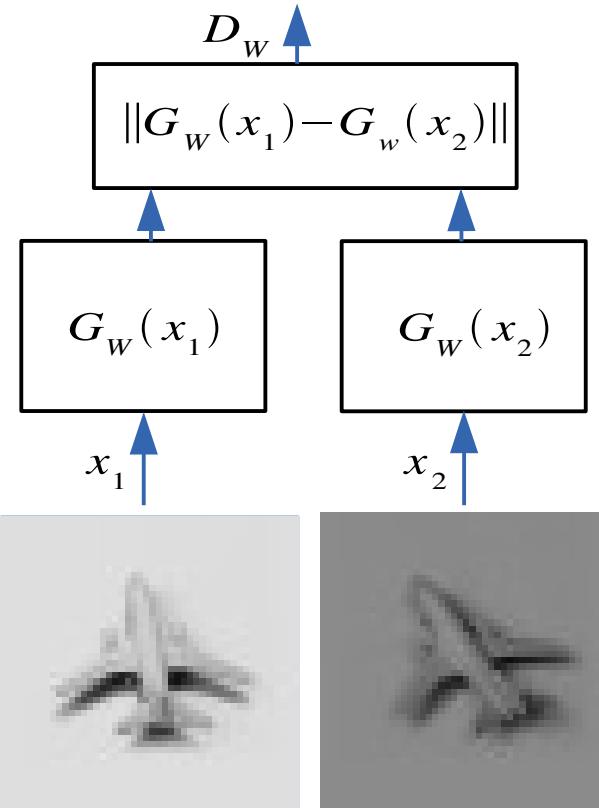
Siamese Architecture and loss function

Y LeCun

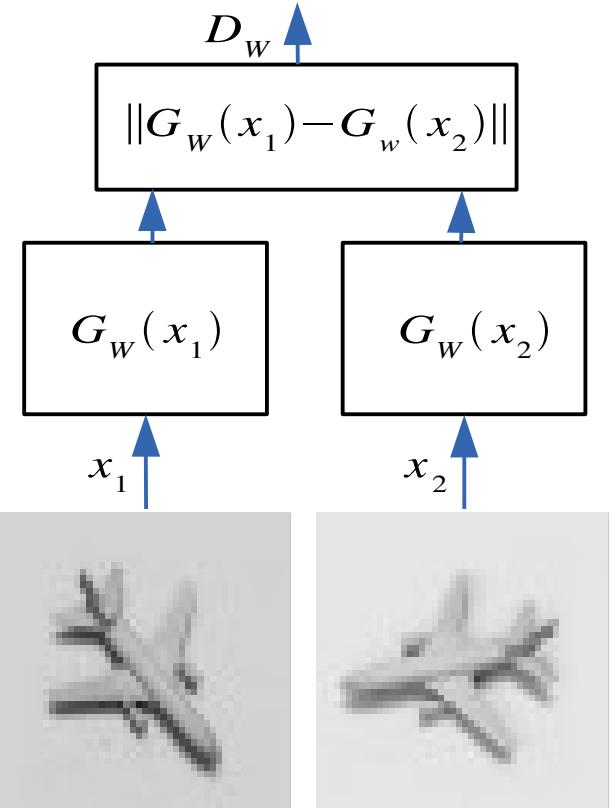
Loss function:

- Outputs corresponding to input samples that are neighbors in the neighborhood graph should be nearby
- Outputs for input samples that are not neighbors should be far away from each other

Make this small



Make this large



Similar images (neighbors
in the neighborhood graph)

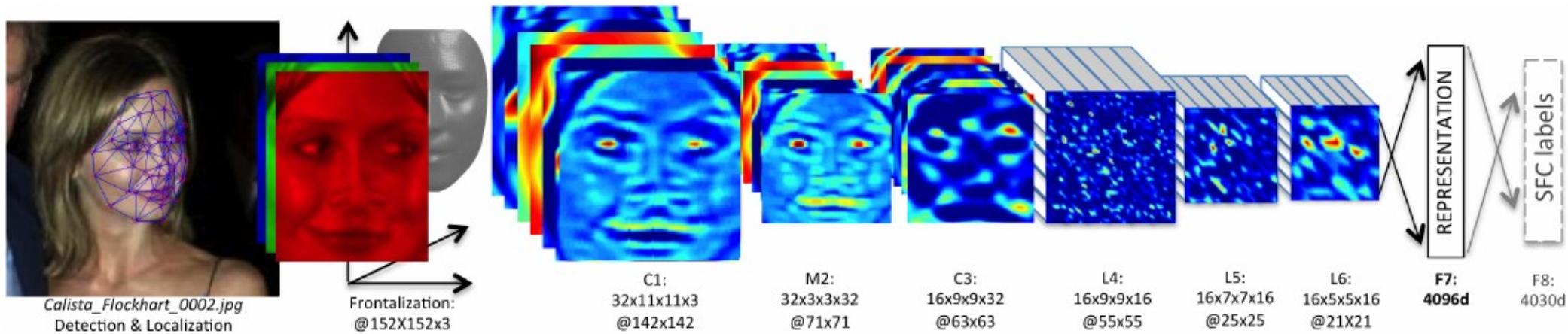
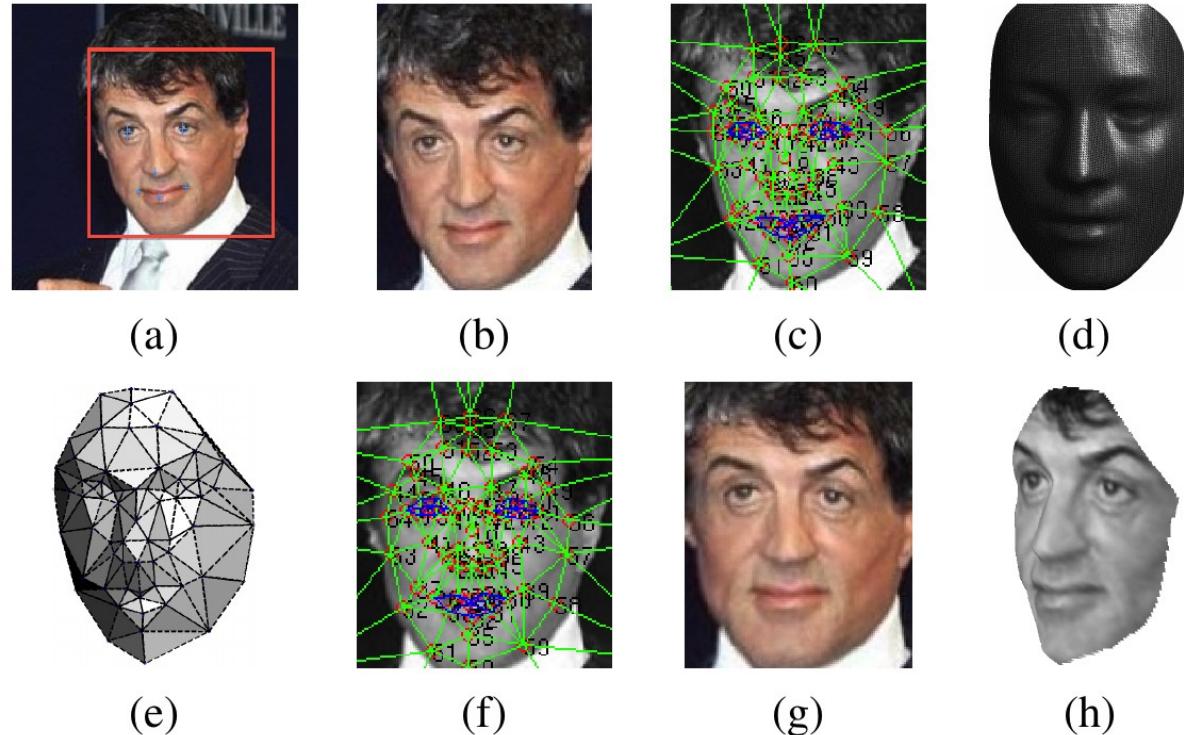
Dissimilar images
(non-neighbors in the
neighborhood graph)

Face Recognition:DeepFace (Facebook AI Research)

Y LeCun

[Taigman et al. CVPR 2014]

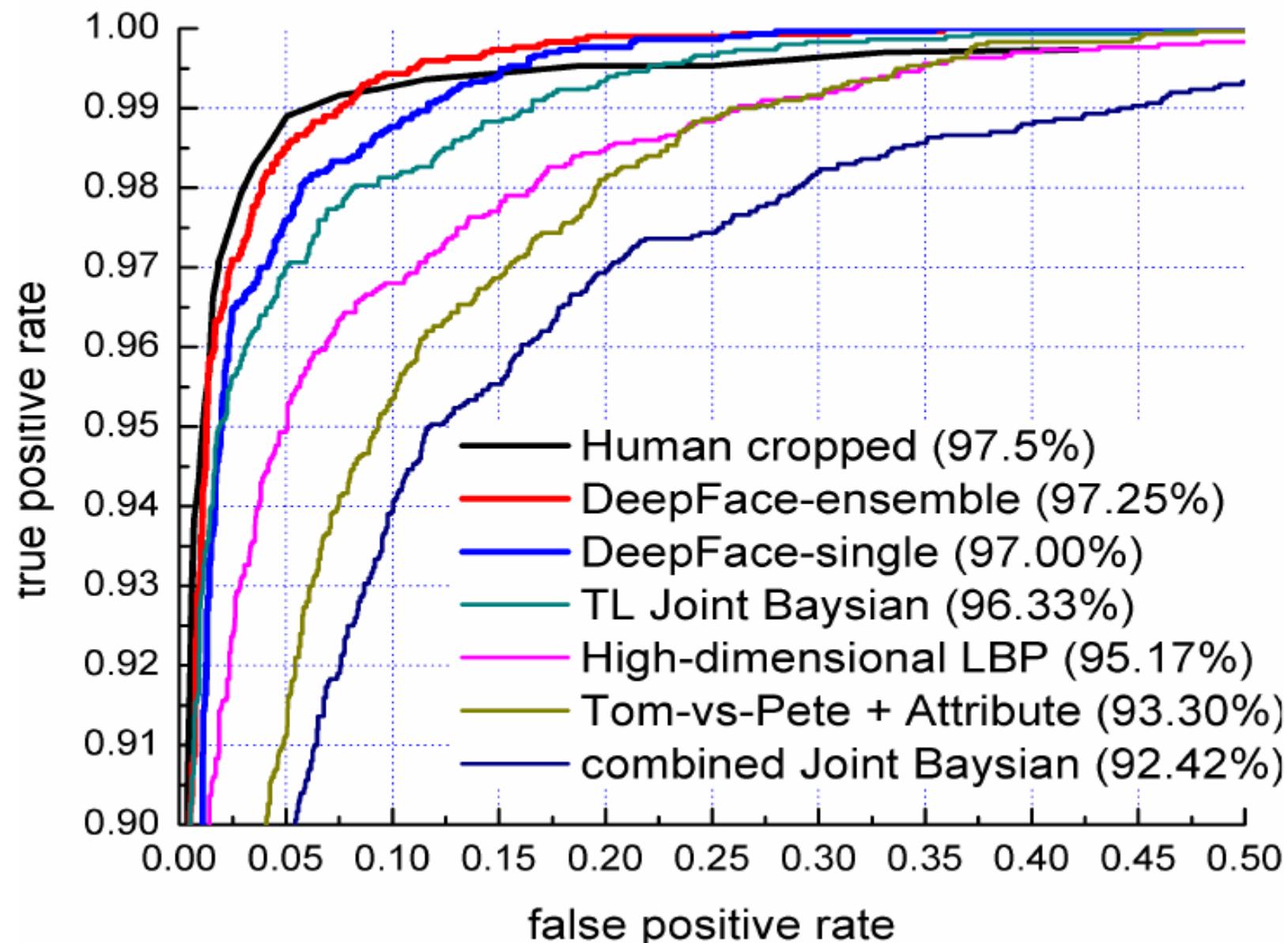
- Alignment
- Convnet



Face Recognition:DeepFace (Facebook AI Research)

Y LeCun

Performance on Labeled Face in the Wild dataset (LFW)



DeepFace: performance

Y LeCun

Method	Accuracy	Protocol
Joint Bayesian [6]	0.9242 ± 0.0108	restricted
Tom-vs-Pete [4]	0.9330 ± 0.0128	restricted
High-dim LBP [7]	0.9517 ± 0.0113	restricted
TL Joint Bayesian [5]	0.9633 ± 0.0108	restricted
DeepFace-single	0.9592 ± 0.0092	unsupervised
DeepFace-single	0.9700 ± 0.0087	restricted
DeepFace-ensemble	0.9715 ± 0.0084	restricted
DeepFace-ensemble	0.9725 ± 0.0081	unrestricted
Human, cropped	0.9753	

Table 3. Comparison with the state-of-the-art on the *LFW* dataset.

Method	Accuracy (%)	AUC	EER
MBGS+SVM- [31]	78.9 ± 1.9	86.9	21.2
APEM+FUSION [22]	79.1 ± 1.5	86.6	21.4
STFRD+PMML [9]	79.5 ± 2.5	88.6	19.9
VSOF+OSS [23]	79.7 ± 1.8	89.4	20.0
DeepFace-single	91.4 ± 1.1	96.3	8.6

Table 4. Comparison with the state-of-the-art on the *YTF* dataset.

Network	Error (<i>SFC</i>)	Accuracy (<i>LFW</i>)
<i>DeepFace-gradient</i>	8.9%	0.9582 ± 0.0118
<i>DeepFace-align2D</i>	9.5%	0.9430 ± 0.0136
<i>DeepFace-Siamese</i>	NA	0.9617 ± 0.0120

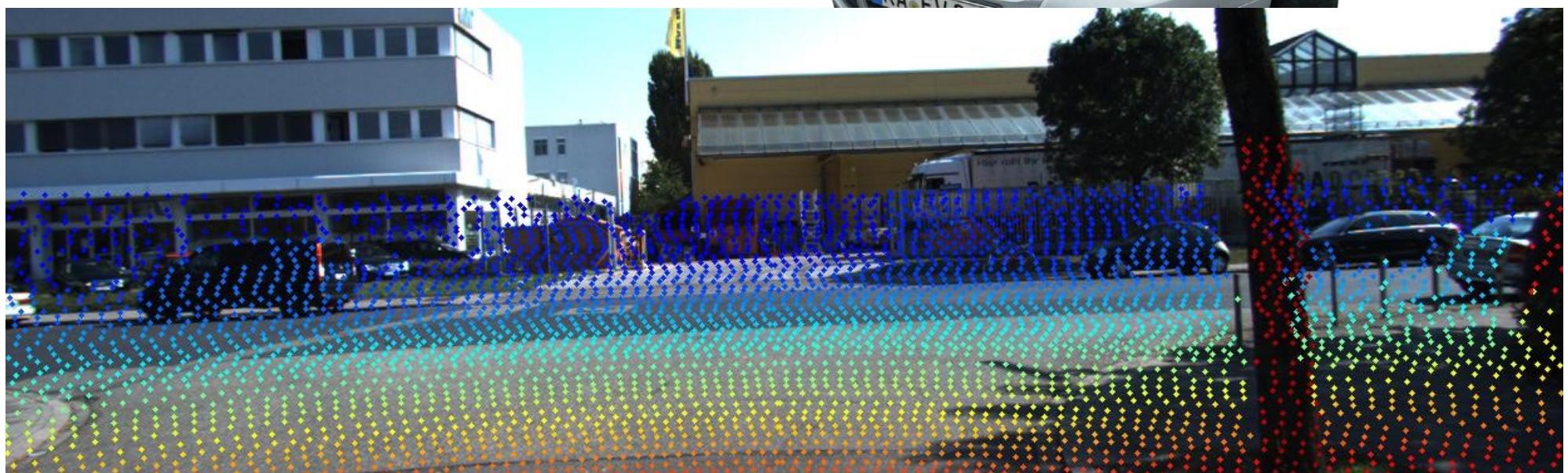
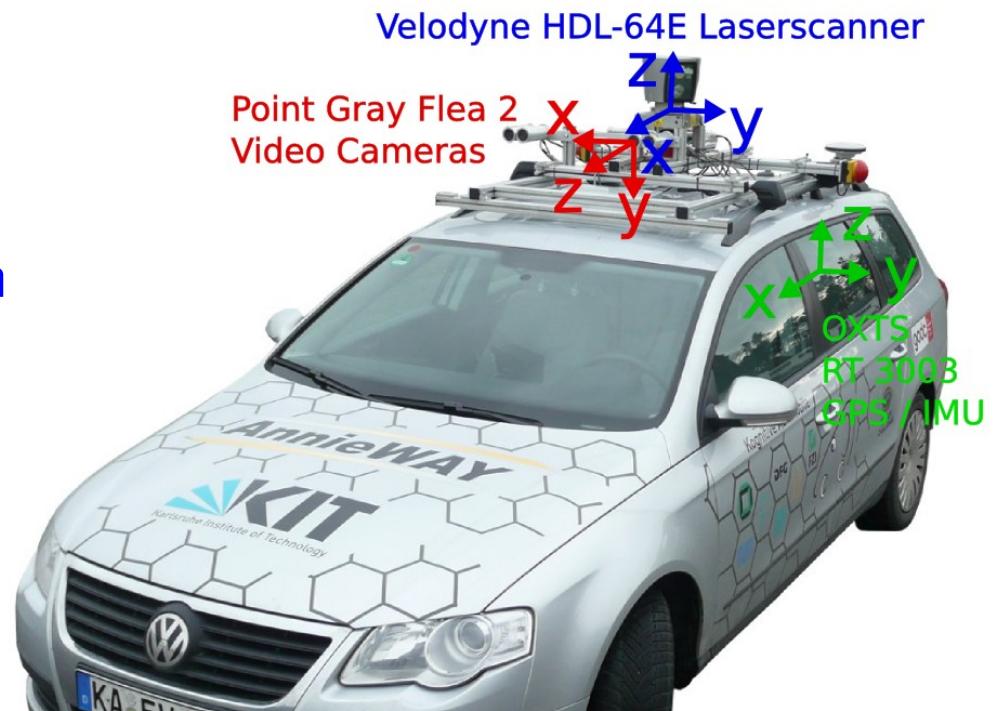
Table 2. The performance of various individual *DeepFace* networks and the Siamese network.

Accurate Depth Estimation from Stereo

KITTI Dataset (RGB registered with LIDAR)

Y LeCun

- Stereo Cameras + Velodyne LIDAR:
aligned data.
- Collected from a car
- Lots of results with many methods from
many teams around the world
- Stereo images sparsely labeled with
depth values
- supervised learning



KITTI Dataset (RGB registered with LIDAR)

Y LeCun

- supervised learning of a patch matcher (as a binary classifier)



Left patch

Right patch

Label



Good match



Bad match

ConvNet for Stereo Matching

Using a ConvNet to learn a similarity measure between image patches

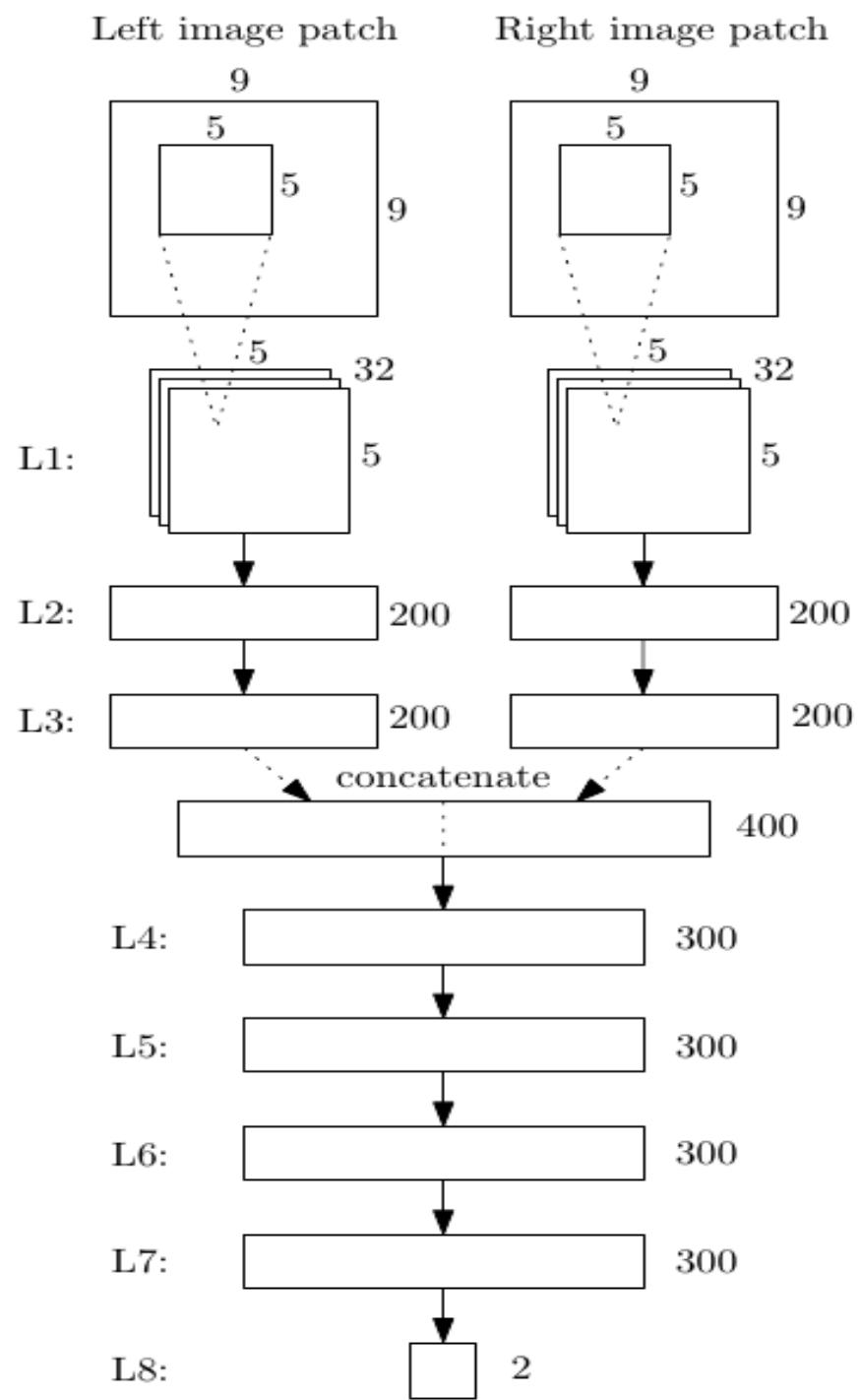
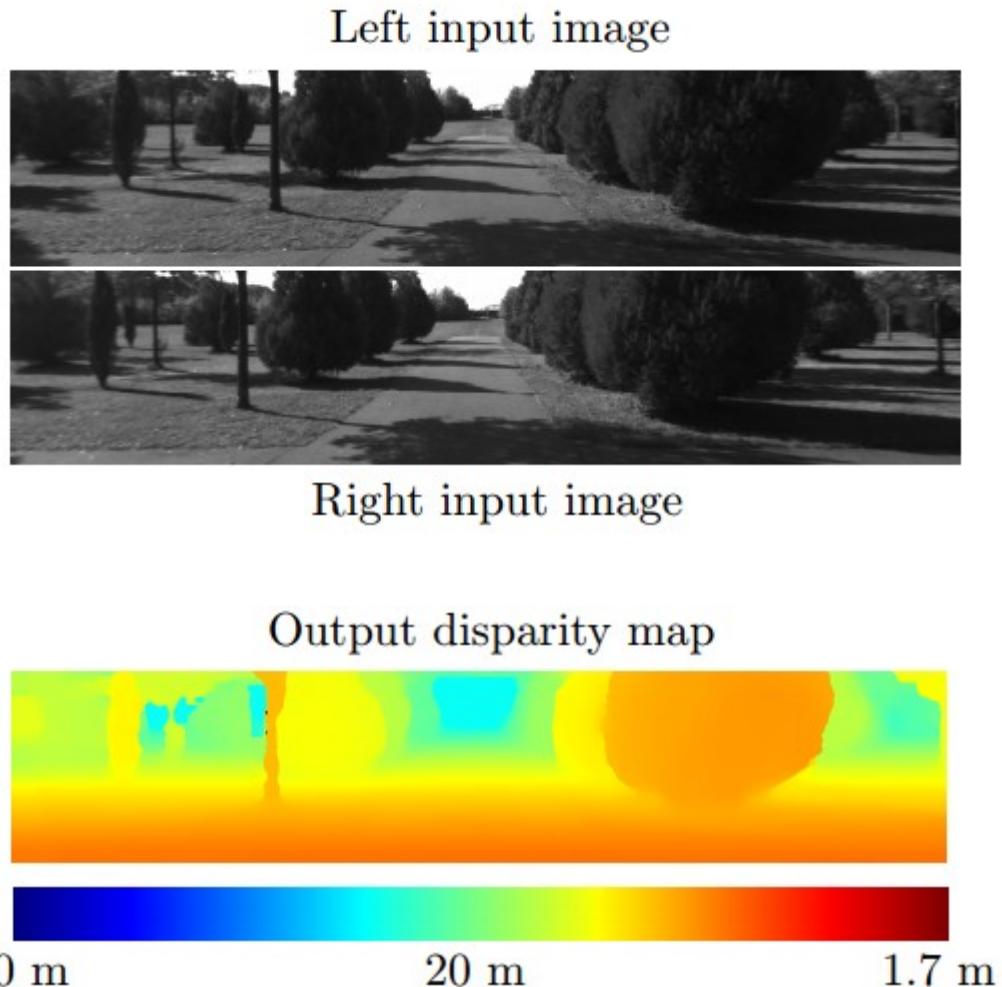


Figure 2: Network architecture

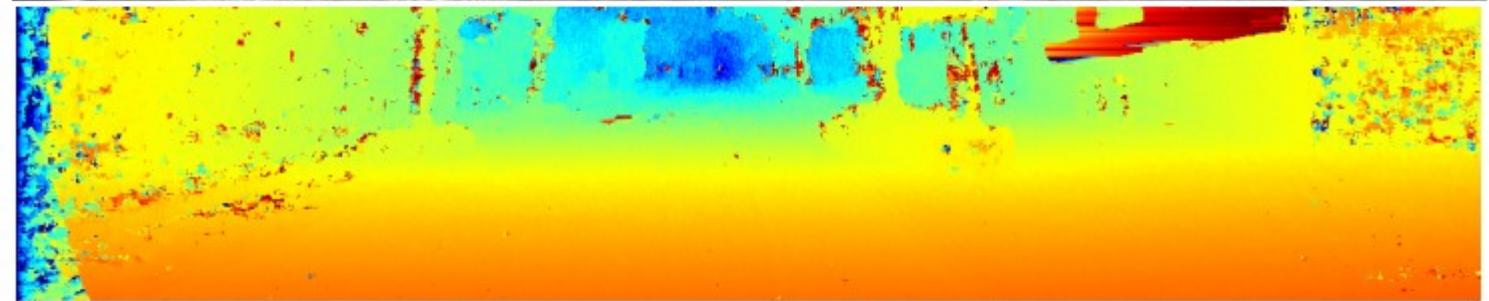
ConvNet for Stereo Matching

Y LeCun

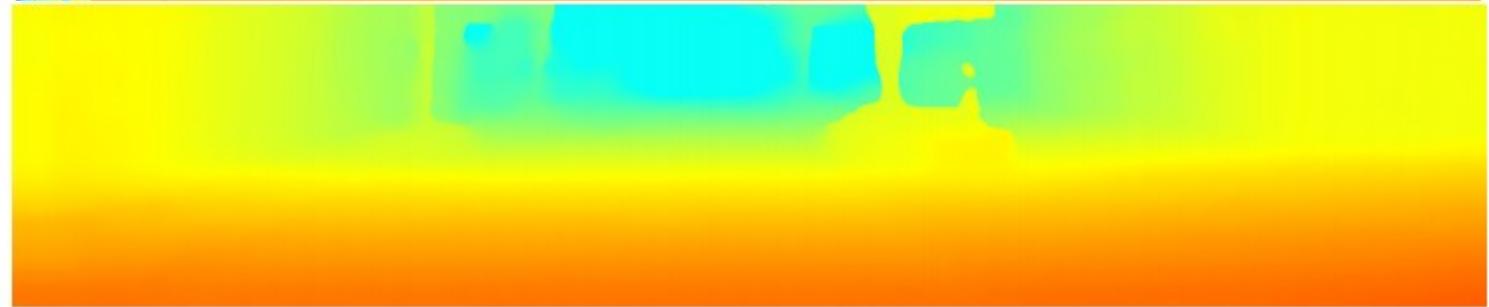
Image



Winner-take-all output
(14.55% error >3 pix)



Result after cleanup
(2.61% error >3 pix)



Depth Estimation from Stereo Pairs

Using a ConvNet to learn a similarity measure
between image patches/

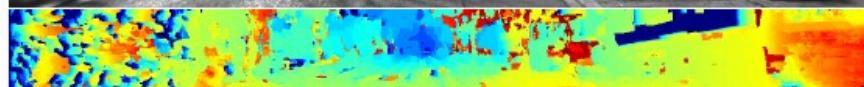
Record holder on KITTI dataset (Sept 2014):

Rank	Method		Error
1	Our method		2.61 %
2	SPS-StF1	Yamaguchi et al. (2014)	2.83 %
3	VC-SF	Vogel et al. (2014)	3.05 %
4	PCBP-SS	Yamaguchi et al. (2013)	3.40 %
5	DDS-SS		3.83 %

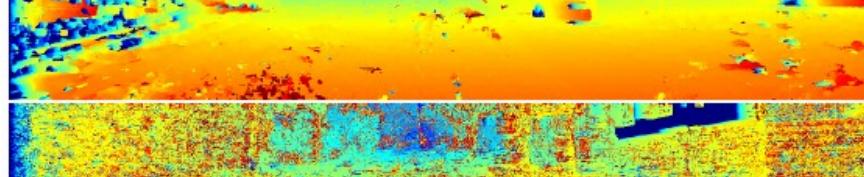
Original image



SADD



Census



ConvNet

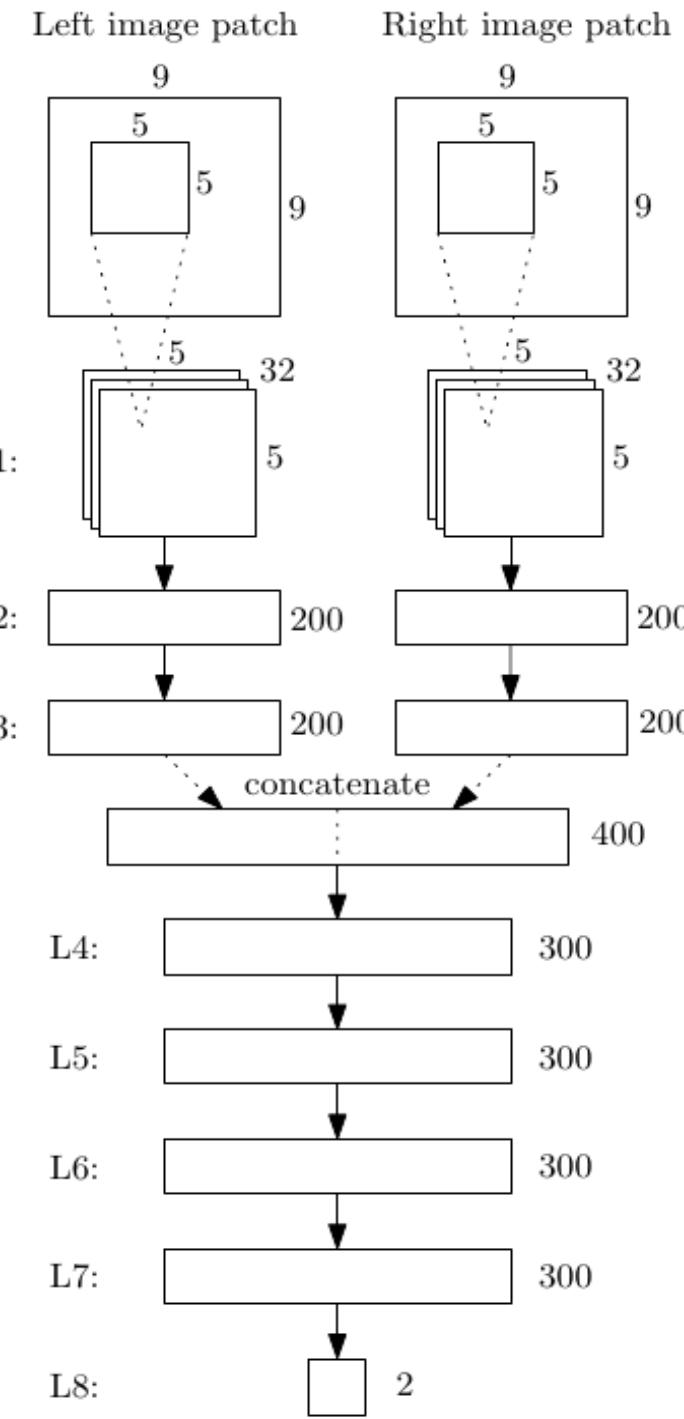
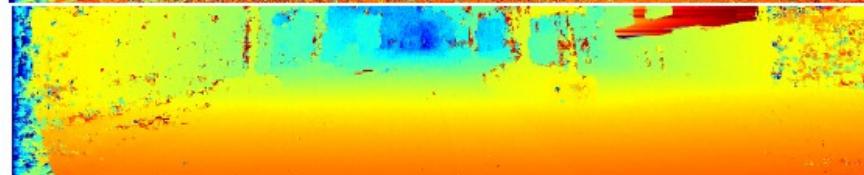


Figure 2: Network architecture



KITTI Stereo Leaderboard (Sept 2014)

Y LeCun

Metric: Percentage of pixels with more than 3 pixel error on disparity

Our ConvNet method is first. Doesn't use optical flow nor multiple images.

#2 uses optical flow and 2 frames

#3 uses multiple frames

#4 is at 3.39% vs our 2.61%

Rank	Method	Setting	Code	Out-Noc	Out-All	Avg-Noc	Avg-All	Density	Runtime
1	<u>MC-CNN</u>			2.61 %	3.84 %	0.8 px	1.0 px	100.00 %	100 s

J. Zbontar and Y. LeCun: [Computing the Stereo Matching Cost with a Convolutional Neural Network](#). 2014.

2	<u>SPS-StFl</u>		2.83 %	3.64 %	0.8 px	0.9 px	100.00 %	35 s	
---	-----------------	--	--------	--------	--------	--------	----------	------	--

K. Yamaguchi, D. McAllester and R. Urtasun: [Efficient Joint Segmentation, Occlusion Labeling, Stereo and Flow Estimation](#). ECCV 2012.

3	<u>VC-SF</u>	 	3.05 %	3.31 %	0.8 px	0.8 px	100.00 %	300 s	
---	--------------	---	--------	--------	--------	--------	----------	-------	--

C. Vogel, S. Roth and K. Schindler: [View-Consistent 3D Scene Flow Estimation over Multiple Frames](#). Proceedings of European Conference on Computer Vision (ECCV), 2012.

4	<u>SPS-St</u>			3.39 %	4.41 %	0.9 px	1.0 px	100.00 %	5 s	
---	---------------	--	--	--------	--------	--------	--------	----------	-----	--

K. Yamaguchi, D. McAllester and R. Urtasun: [Efficient Joint Segmentation, Occlusion Labeling, Stereo and Flow Estimation](#). ECCV 2012.

5	<u>PCBP-SS</u>		3.40 %	4.72 %	0.8 px	1.0 px	100.00 %	5 min
---	----------------	--	--------	--------	--------	--------	----------	-------

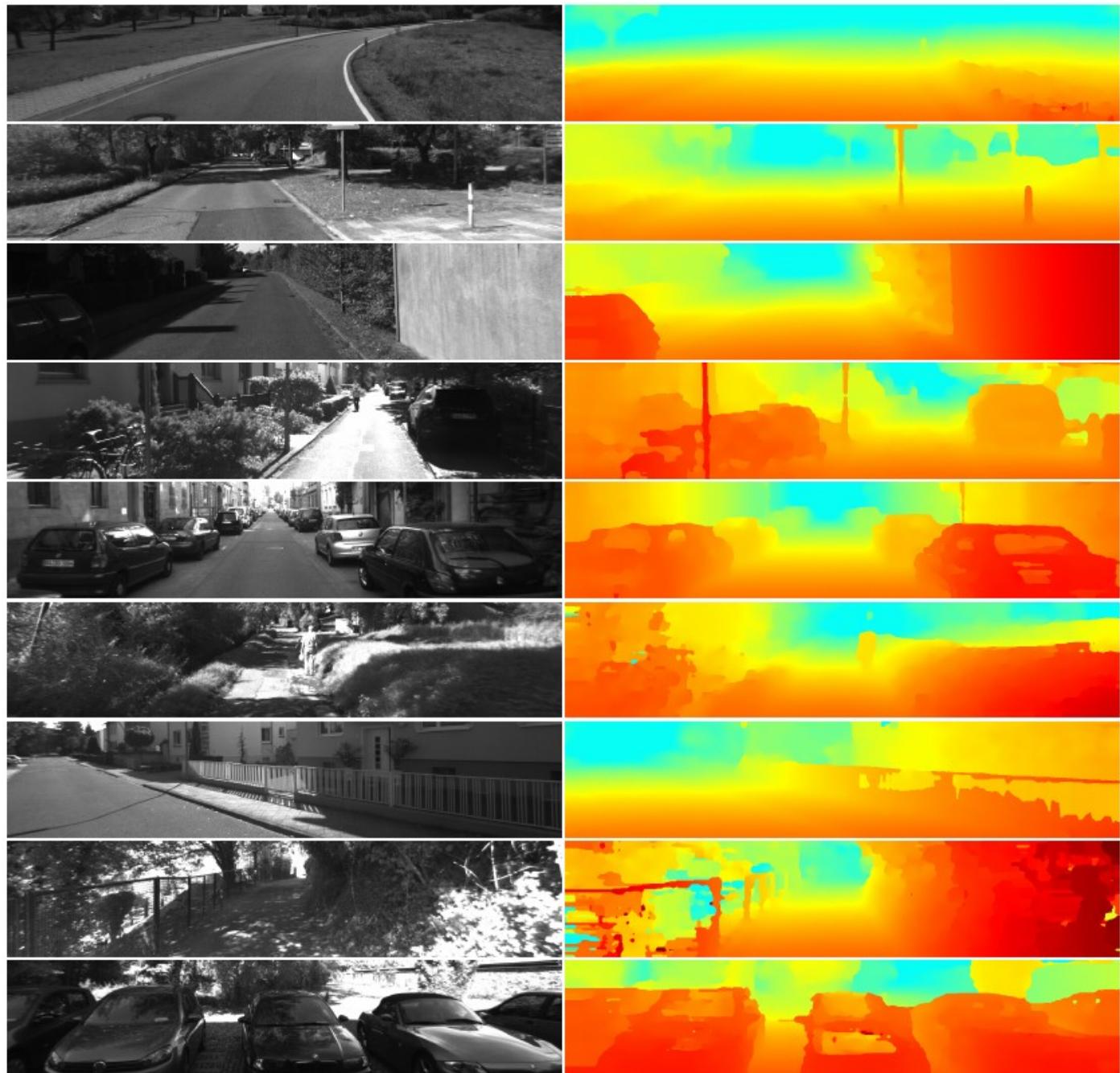
K. Yamaguchi, D. McAllester and R. Urtasun: [Robust Monocular Epipolar Flow Estimation](#). CVPR 2013.

Depth Estimation from Stereo Pairs: Results

Y LeCun

[Zbontar & LeCun Arxiv '14]

Presentation at ECCV
workshop 2014/9/6



Body Pose Estimation

Pose Estimation and Attribute Recovery with ConvNets

Y LeCun

Pose-Aligned Network for Deep Attribute Modeling

[Zhang et al. CVPR 2014] (Facebook AI Research)



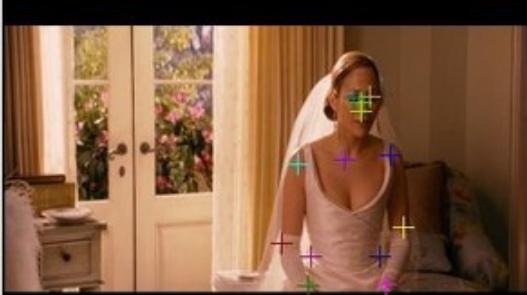
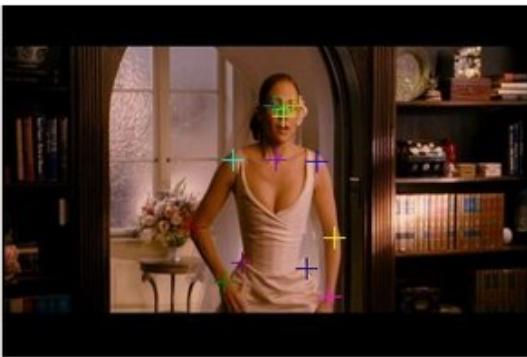
(a) Highest scoring results for people wearing glasses.



(b) Highest scoring results for people wearing a hat.

Real-time hand pose recovery

[Tompson et al. Trans. on Graphics 14]



Body pose estimation [Tompson et al. ICLR, 2014]



Other Tasks for Which Deep Convolutional Nets are the Best

Y LeCun

- Handwriting recognition MNIST (many), Arabic HWX (IDSIA)
 - OCR in the Wild [2011]: StreetView House Numbers (NYU and others)
 - Traffic sign recognition [2011] GTSRB competition (IDSIA, NYU)
 - Asian handwriting recognition [2013] ICDAR competition (IDSIA)
 - Pedestrian Detection [2013]: INRIA datasets and others (NYU)
 - Volumetric brain image segmentation [2009] connectomics (IDSIA, MIT)
 - Human Action Recognition [2011] Hollywood II dataset (Stanford)
 - Object Recognition [2012] ImageNet competition (Toronto)
 - Scene Parsing [2012] Stanford bgd, SiftFlow, Barcelona datasets (NYU)
 - Scene parsing from depth images [2013] NYU RGB-D dataset (NYU)
 - Speech Recognition [2012] Acoustic modeling (IBM and Google)
 - Breast cancer cell mitosis detection [2011] MITOS (IDSIA)
-
- The list of perceptual tasks for which ConvNets hold the record is growing.
 - Most of these tasks (but not all) use purely supervised convnets.

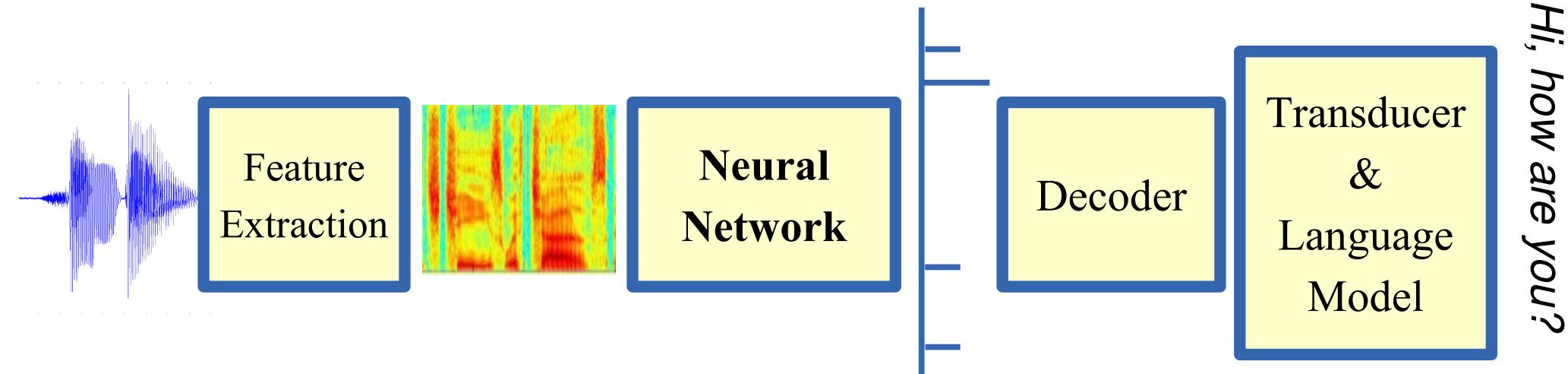
Deep Learning and Convolutional Networks in Speech, Audio, and Signals

Acoustic Modeling in Speech Recognition (Google)

Y LeCun

A typical speech recognition architecture with DL-based acoustic modeling

- ▶ Features: log energy of a filter bank (e.g. 40 filters)
- ▶ Neural net acoustic modeling (convolutional or not)
- ▶ Input window: typically 10 to 40 acoustic frames
- ▶ Fully-connected neural net: **10 layers, 2000-4000 hidden units/layer**
- ▶ But convolutional nets do better....
- ▶ Predicts phone state, typically 2000 to 8000 categories

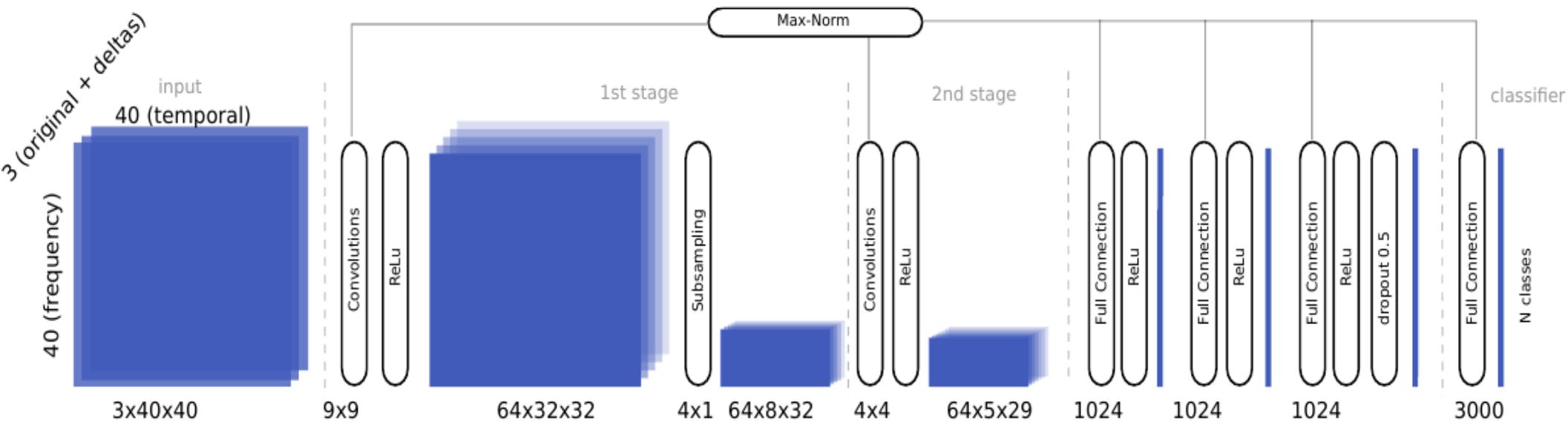


Mohamed et al. “DBNs for phone recognition” NIPS Workshop 2009

Zeiler et al. “On rectified linear units for speech recognition” ICASSP 2013

Speech Recognition with Convolutional Nets (NYU/IBM)

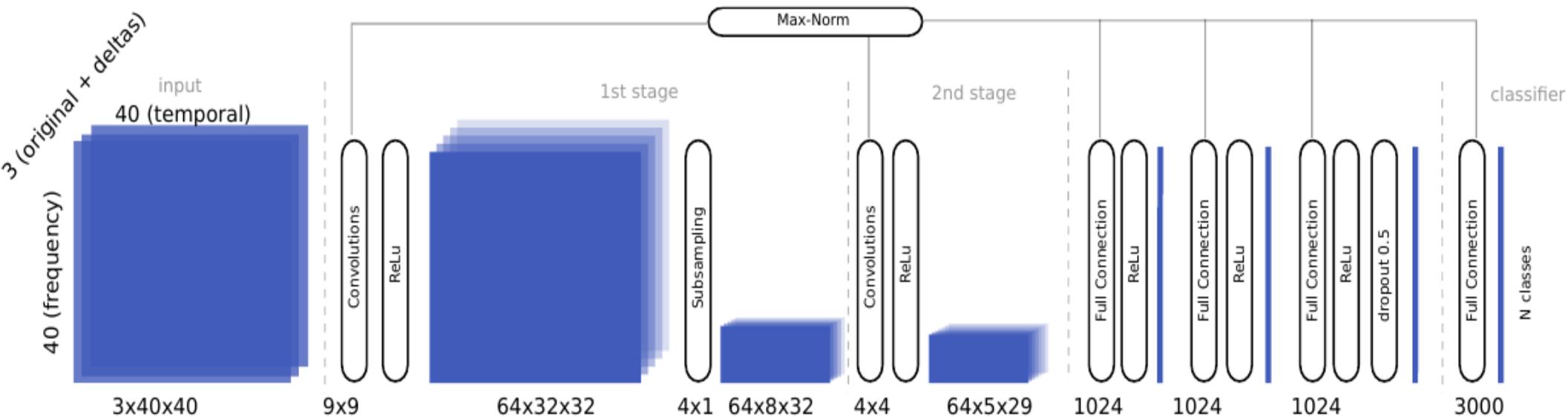
Y LeCun



- Acoustic Model: ConvNet with 7 layers. 54.4 million parameters.
- Classifies acoustic signal into 3000 context-dependent subphones categories
- ReLU units + dropout for last layers
- Trained on GPU. 4 days of training

Speech Recognition with Convolutional Nets (NYU/IBM)

Y LeCun



Subphone-level classification error (sept 2013):

- ▶ Cantonese: phone: 20.4% error; subphone: 33.6% error (IBM DNN: 37.8%)

Subphone-level classification error (march 2013)

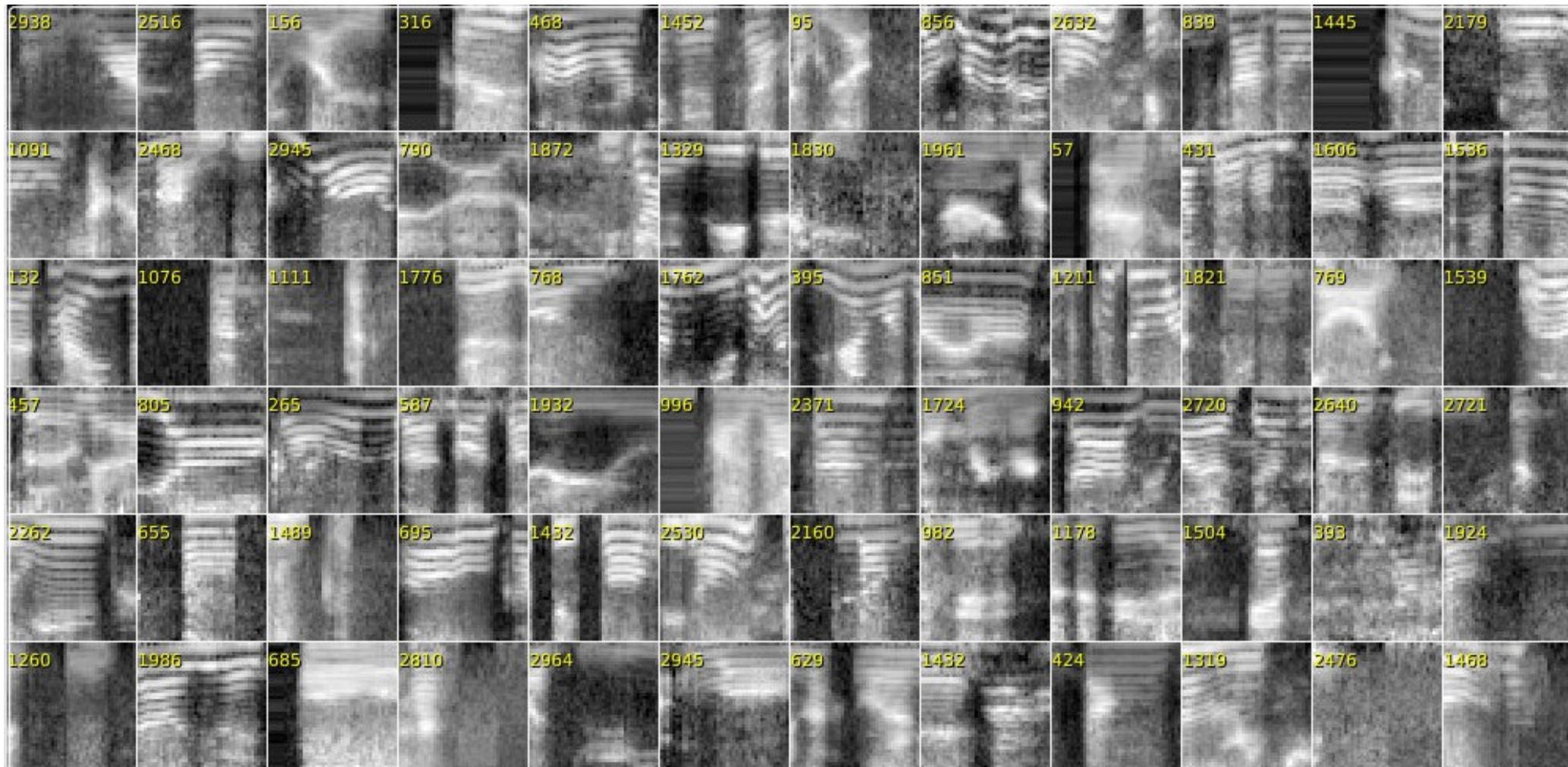
- ▶ Cantonese: subphone: 36.91%
- ▶ Vietnamese: subphone 48.54%
- ▶ Full system performance (token error rate on conversational speech):
 - ▶ 76.2% (52.9% substitution, 13.0% deletion, 10.2% insertion)

Speech Recognition with Convolutional Nets (NYU/IBM)

Y LeCun

Training samples.

- ▶ 40 MEL-frequency Cepstral Coefficients
- ▶ Window: 40 frames, 10ms each



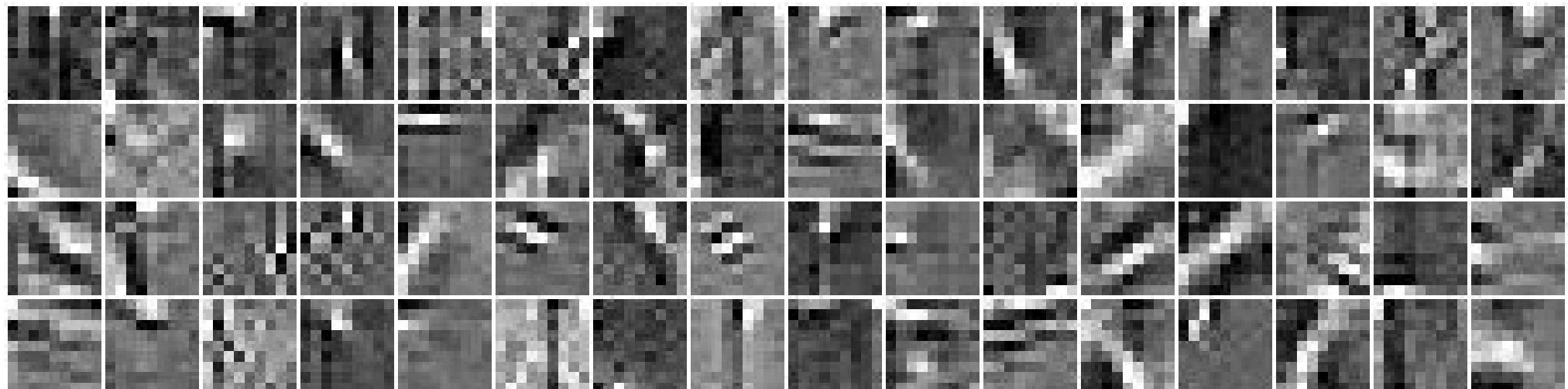


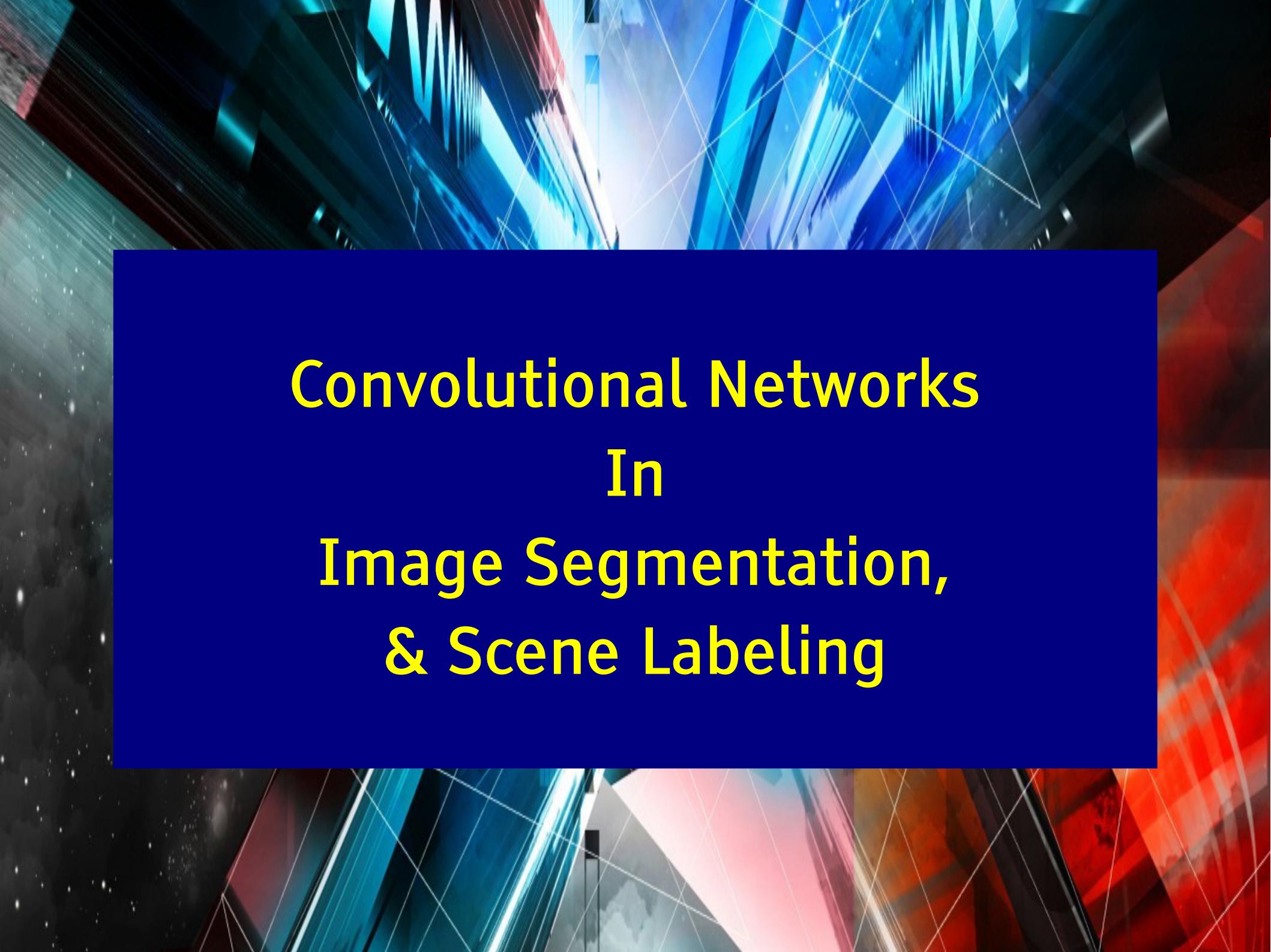
Speech Recognition with Convolutional Nets (NYU/IBM)

Y LeCun

■ Convolution Kernels at Layer 1:

- ▶ 64 kernels of size 9x9





Convolutional Networks In Image Segmentation, & Scene Labeling

ConvNets for Image Segmentation

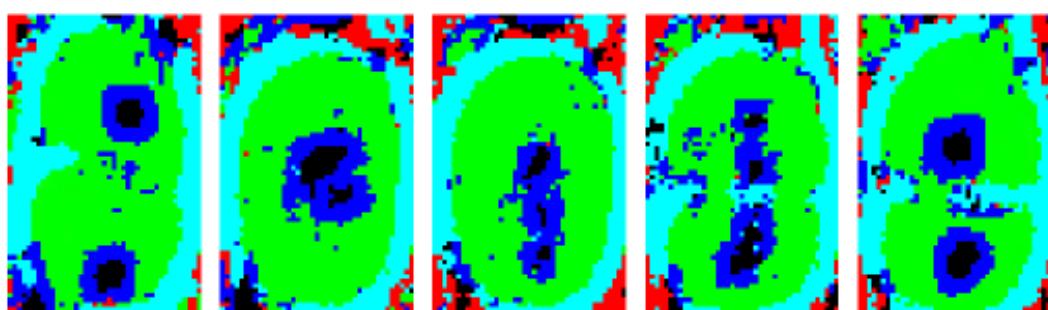
Y LeCun

- Biological Image Segmentation

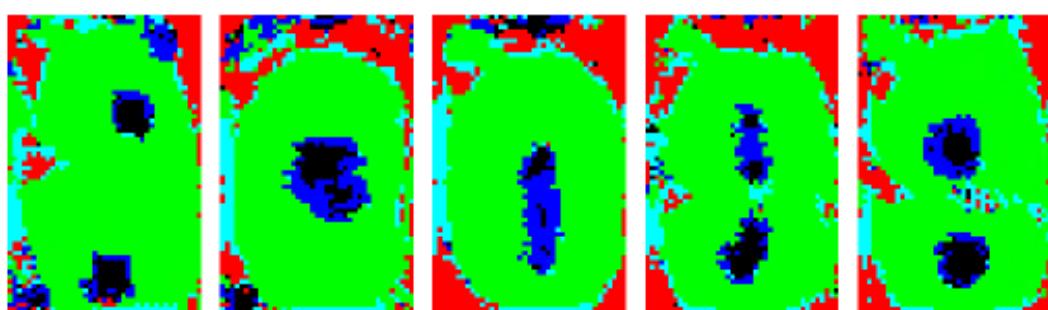
- ▶ [Ning et al. IEEE-TIP 2005]



- Pixel labeling with large context using a convnet

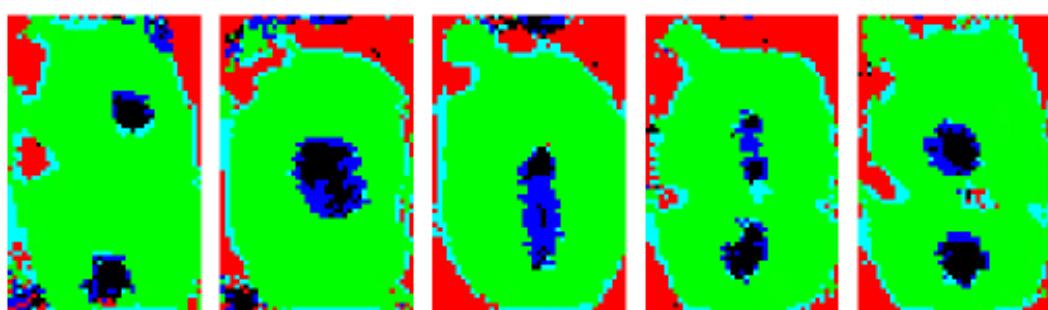


- ConvNet takes a window of pixels and produces a label for the central pixel



- Cleanup using a kind of conditional random field (CRF)

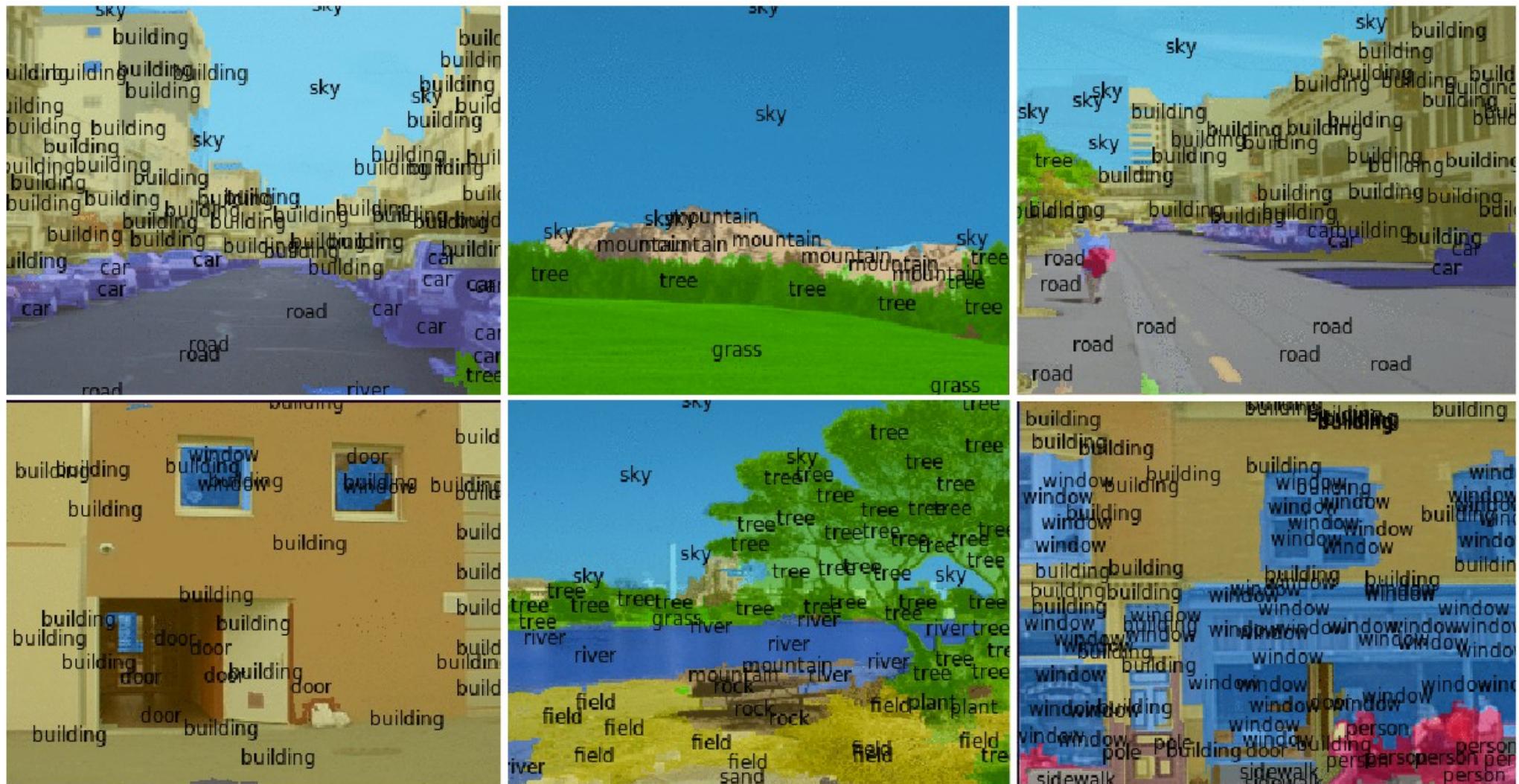
- ▶ Similar to a field of expert



Semantic Labeling / Scene Parsing: Labeling every pixel with the object it belongs to

Y LeCun

- Would help identify obstacles, targets, landing sites, dangerous areas
- Would help line up depth map with edge maps

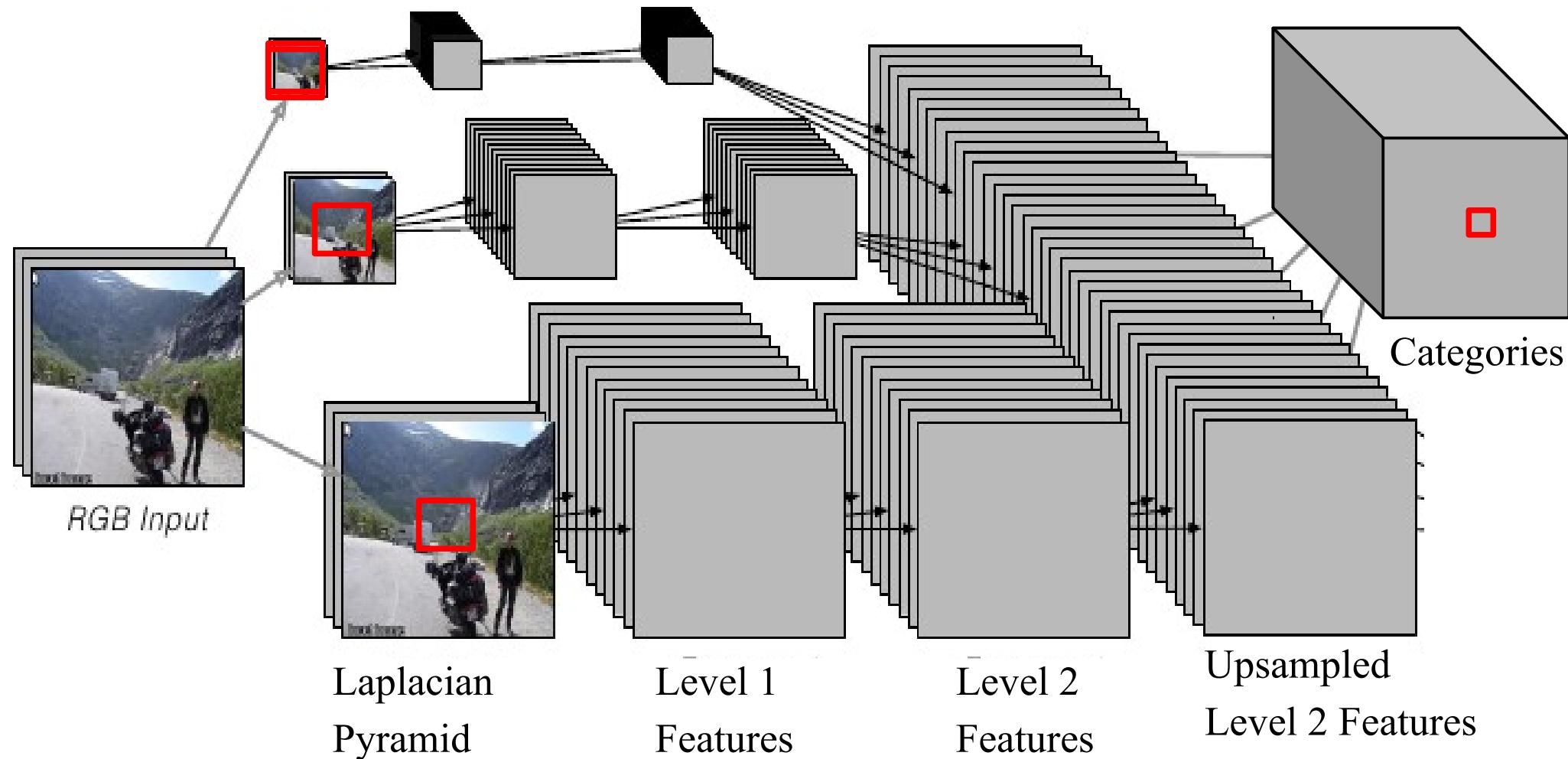


Scene Parsing/Labeling: ConvNet Architecture

Y LeCun

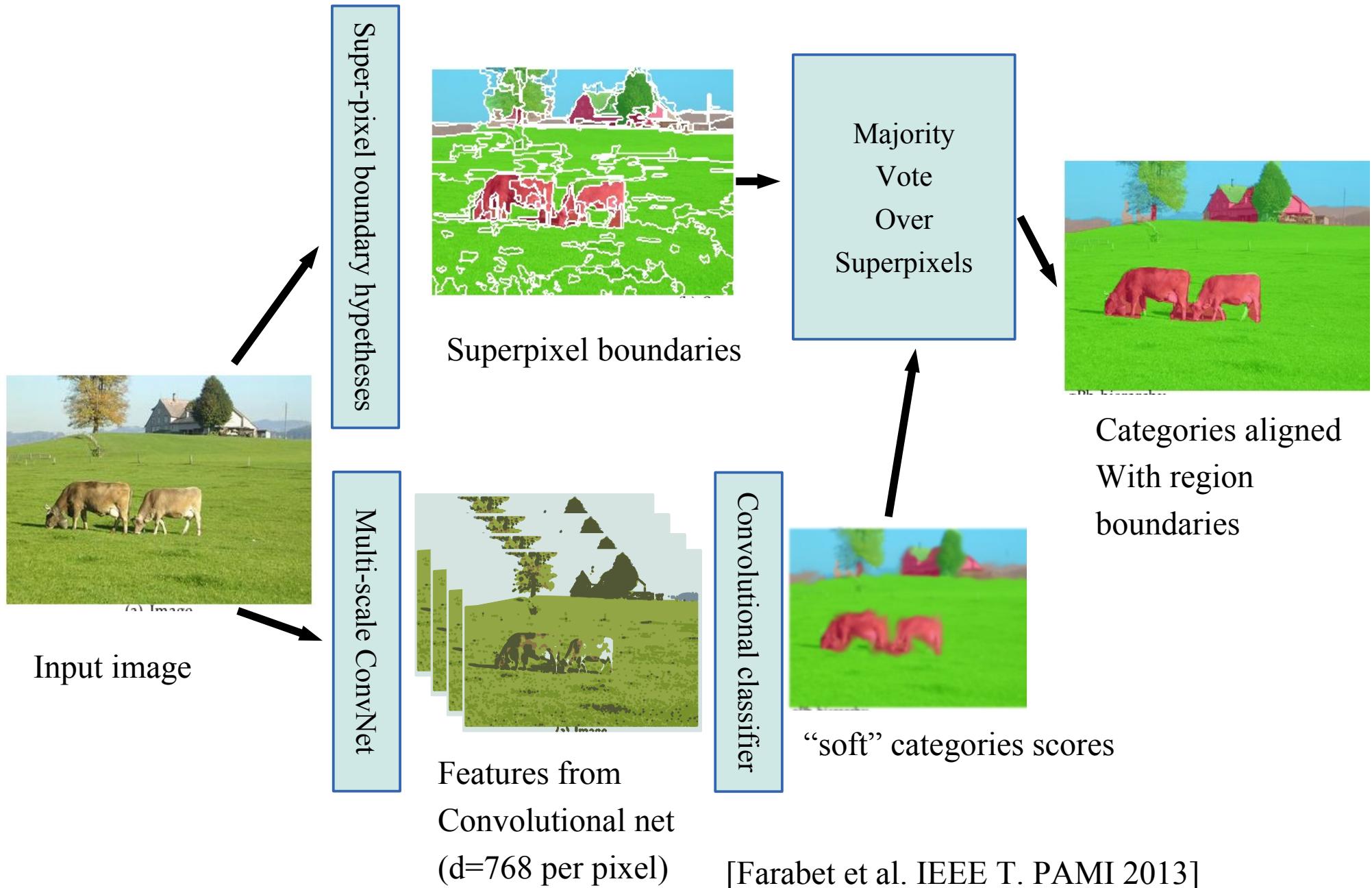
- Each output sees a large input context:

- ▶ **46x46** window at full rez; **92x92** at $\frac{1}{2}$ rez; **184x184** at $\frac{1}{4}$ rez
- ▶ [7x7conv]->[2x2pool]->[7x7conv]->[2x2pool]->[7x7conv]->
- ▶ Trained supervised on fully-labeled images



Method 1: majority over super-pixel regions

Y LeCun



Scene Parsing/Labeling: Performance

Y LeCun

■ Stanford Background Dataset [Gould 1009]: 8 categories

	Pixel Acc.	Class Acc.	CT (sec.)
Gould <i>et al.</i> 2009 [14]	76.4%	-	10 to 600s
Munoz <i>et al.</i> 2010 [32]	76.9%	66.2%	12s
Tighe <i>et al.</i> 2010 [46]	77.5%	-	10 to 300s
Socher <i>et al.</i> 2011 [45]	78.1%	-	?
Kumar <i>et al.</i> 2010 [22]	79.4%	-	< 600s
Lempitzky <i>et al.</i> 2011 [28]	81.9%	72.4%	> 60s
singlescale convnet	66.0 %	56.5 %	0.35s
multiscale convnet	78.8 %	72.4%	0.6s
multiscale net + superpixels	80.4%	74.56%	0.7s
multiscale net + gPb + cover	80.4%	75.24%	61s
multiscale net + CRF on gPb	81.4%	76.0%	60.5s

Scene Parsing/Labeling: Performance

Y LeCun

	Pixel Acc.	Class Acc.
Liu <i>et al.</i> 2009 [31]	74.75%	-
Tighe <i>et al.</i> 2010 [44]	76.9%	29.4%
raw multiscale net ¹	67.9%	45.9%
multiscale net + superpixels ¹	71.9%	50.8%
multiscale net + cover ¹	72.3%	50.8%
multiscale net + cover ²	78.5%	29.6%

- SIFT Flow Dataset
- [Liu 2009]:
33 categories

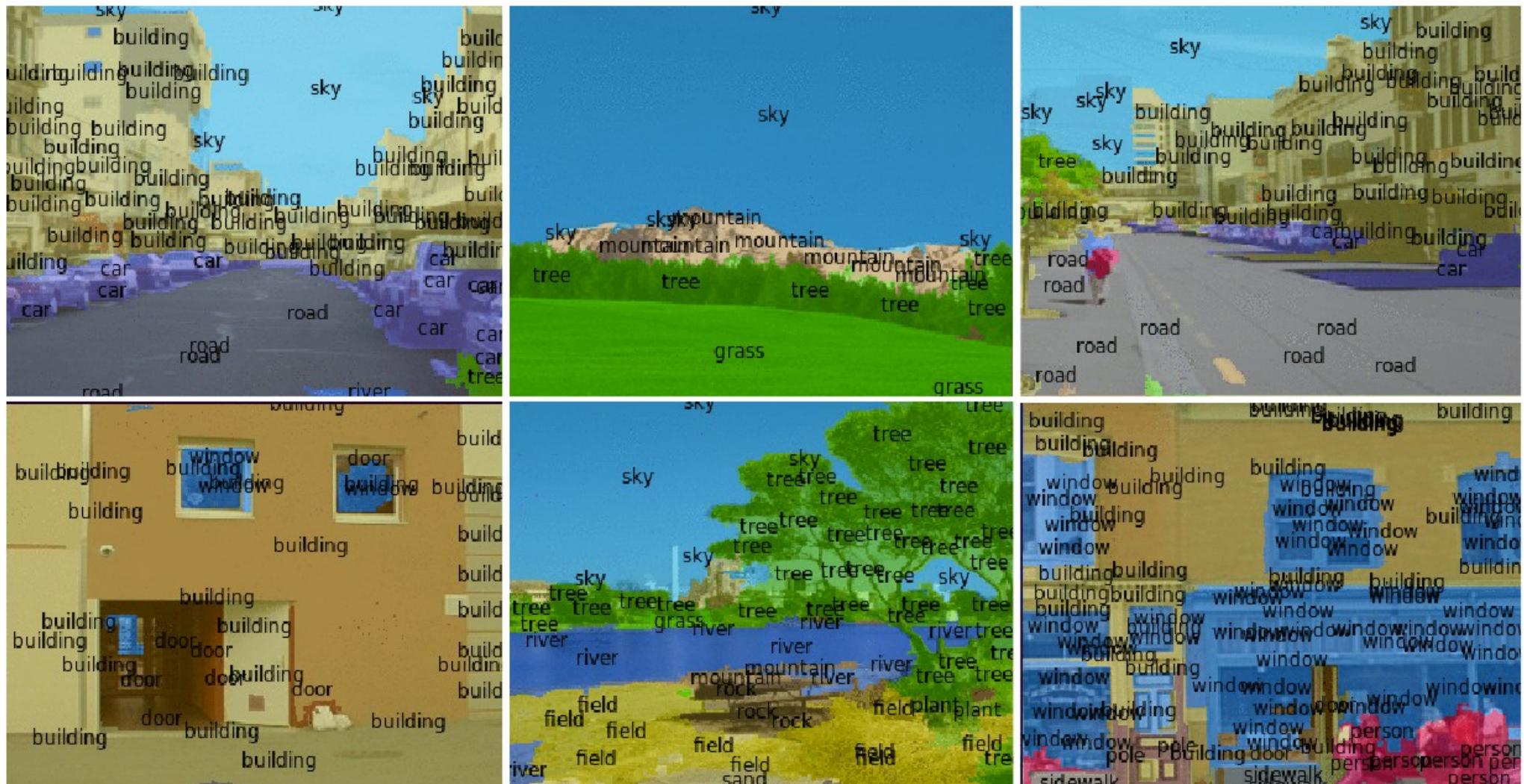
- Barcelona dataset
- [Tighe 2010]:
170 categories.

	Pixel Acc.	Class Acc.
Tighe <i>et al.</i> 2010 [44]	66.9%	7.6%
raw multiscale net ¹	37.8%	12.1%
multiscale net + superpixels ¹	44.1%	12.4%
multiscale net + cover ¹	46.4%	12.5%
multiscale net + cover ²	67.8%	9.5%

Scene Parsing/Labeling: SIFT Flow dataset (33 categories)

Y LeCun

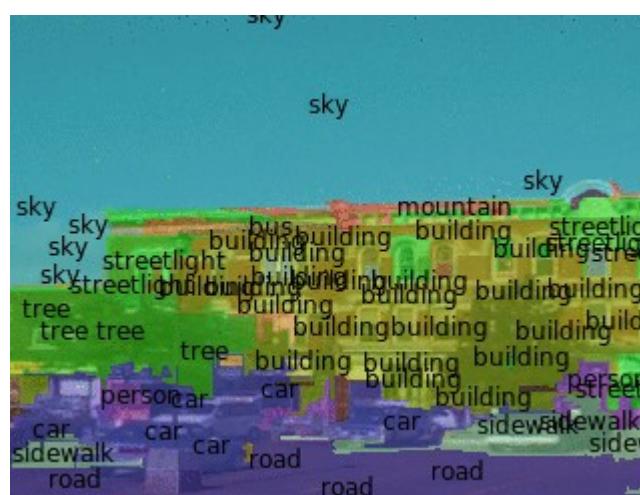
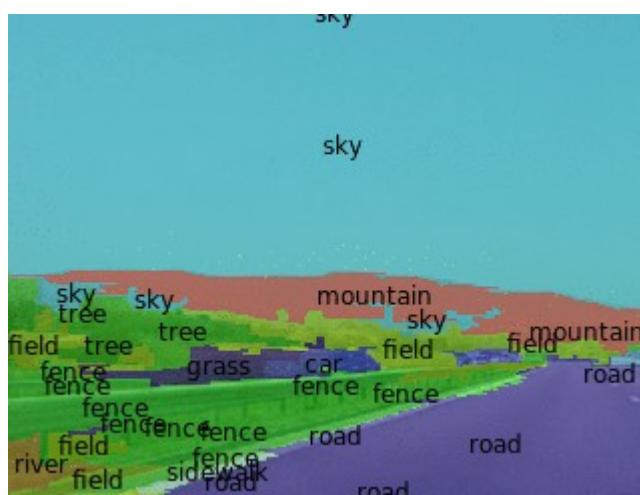
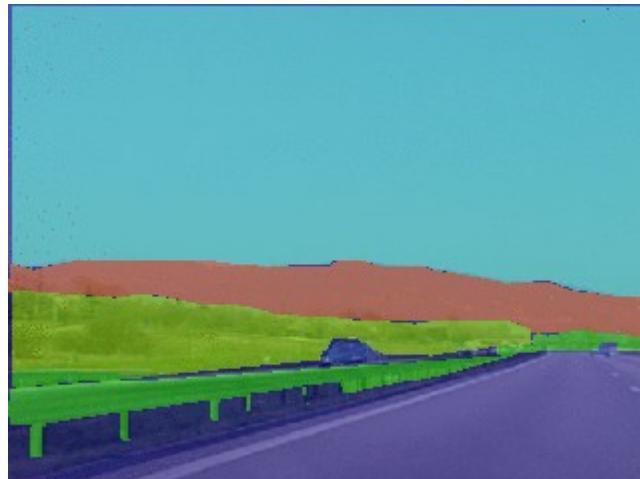
Samples from the SIFT-Flow dataset (Liu)



[Farabet et al. ICML 2012, PAMI 2013]

Scene Parsing/Labeling: SIFT Flow dataset (33 categories)

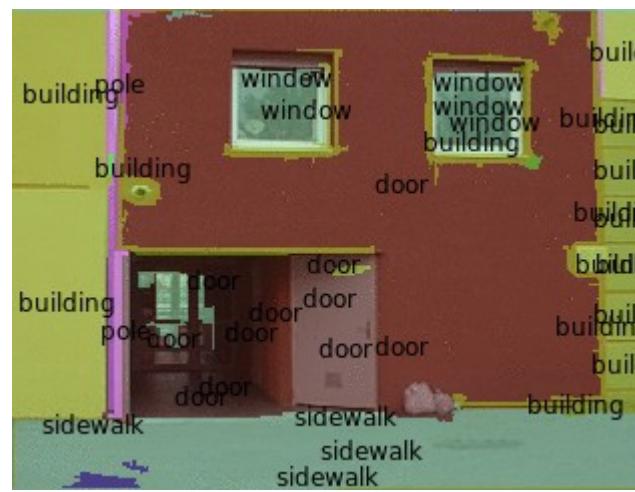
Y LeCun



[Farabet et al. ICML 2012, PAMI 2013]

Scene Parsing/Labeling

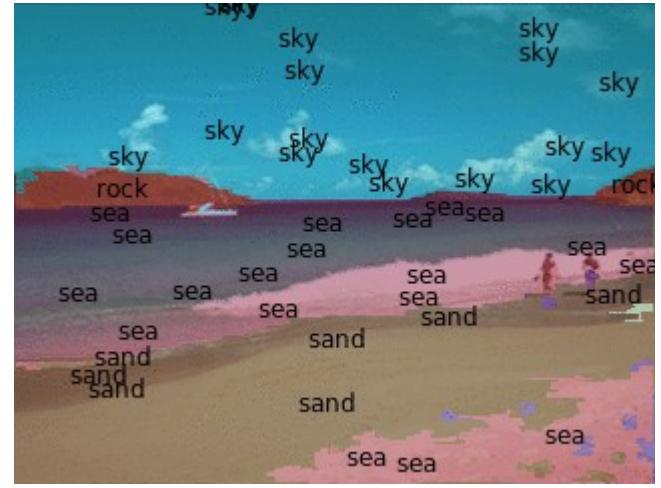
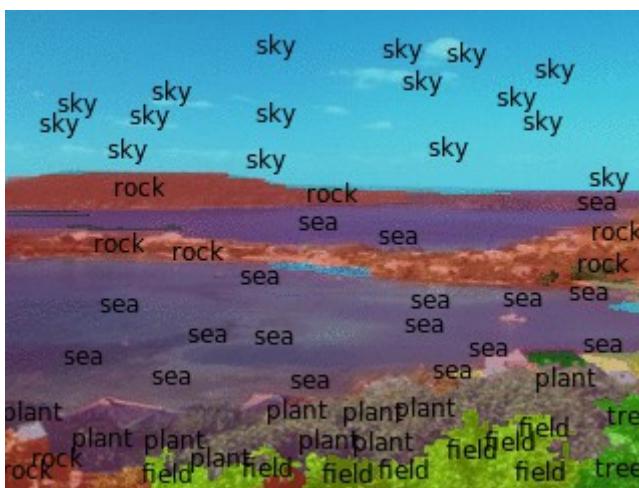
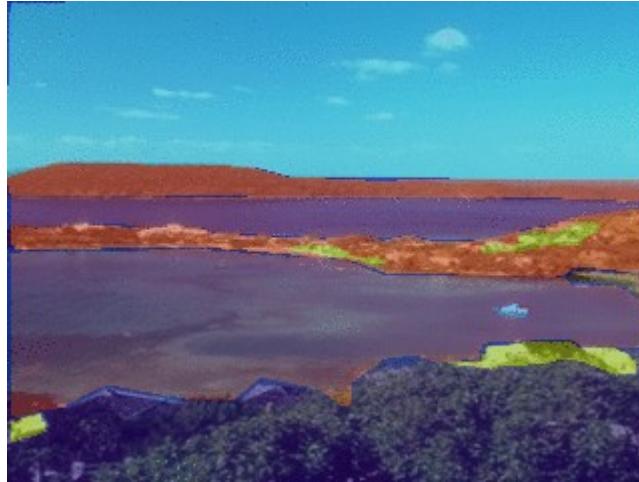
Y LeCun



[Farabet et al. ICML 2012, PAMI 2013]

Scene Parsing/Labeling

Y LeCun



[Farabet et al. ICML 2012, PAMI 2013]

Scene Parsing/Labeling

Y LeCun



[Farabet et al. ICML 2012, PAMI 2013]

Scene Parsing/Labeling

Y LeCun



[Farabet et al. ICML 2012, PAMI 2013]

Scene Parsing/Labeling

Y LeCun



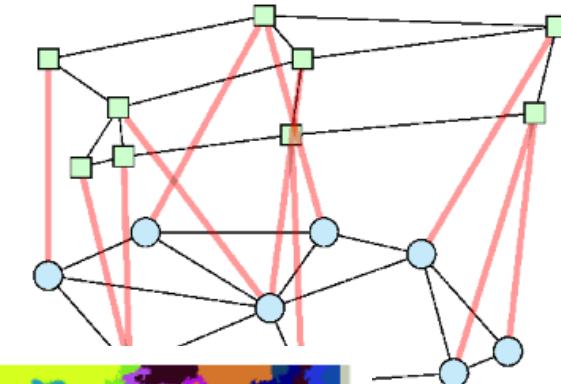
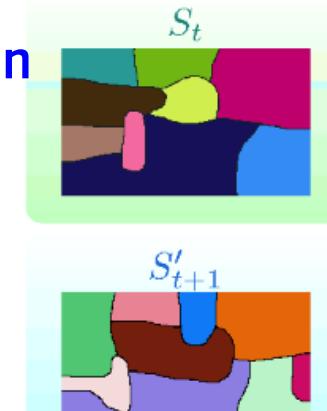
- No post-processing
- Frame-by-frame
- ConvNet runs at 50ms/frame on Virtex-6 FPGA hardware
 - ▶ But communicating the features over ethernet limits system performance

Temporal Consistency

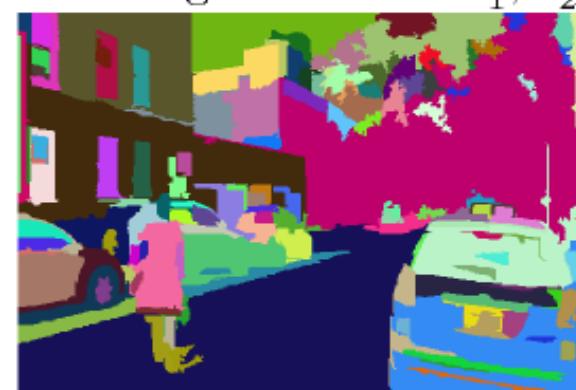
Y LeCun

Spatio-Temporal Super-Pixel segmentation

- [Couprie et al ICIP 2013]
- [Couprie et al JMLR under review]
- Majority vote over super-pixels



Independent segmentations S'_1 , S'_2 and S'_3



Temporally consistent segmentations $S_1 (= S'_1)$, S_2 , and S_3

Scene Parsing/Labeling: Temporal Consistency

Y LeCun



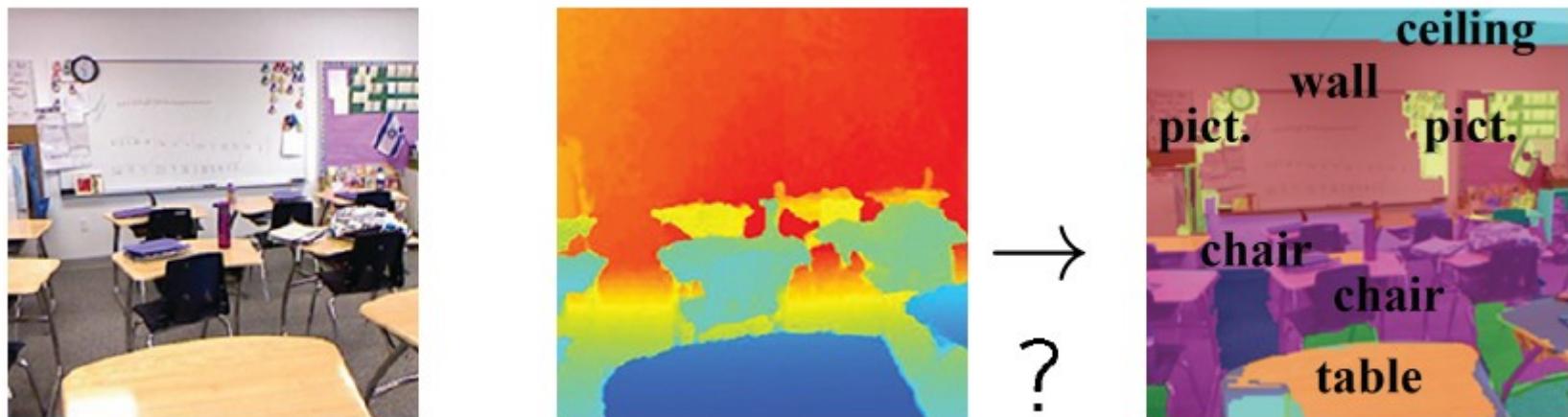
- Causal method for temporal consistency

[Couprie, Farabet, Najman, LeCun ICLR 2013, ICIP 2013]

NYU RGB-D Dataset

Y LeCun

Captured with a Kinect on a steadycam



Results

Y LeCun

Depth helps a bit

- ▶ Helps a lot for floor and props
- ▶ Helps surprisingly little for structures, and hurts for furniture

	Ground	Furniture	Props	Structure	Class Acc.	Pixel Acc.	Comput. time (s)
Silberman et al. (2012)	68	70	42	59	59.6	58.6	>3
Cadena and Kosecka (2013)	87.9	64.1	31.0	77.8	65.2	66.9	1.7
Multiscale convnet	68.1	51.1	29.9	87.8	59.2	63.0	0.7
Multiscale+depth convnet	87.3	45.3	35.5	86.1	63.5	64.5	0.7

[C. Cadena, J. Kosecka "Semantic Parsing for Priming Object Detection in RGB-D Scenes"
Semantic Perception Mapping and Exploration (SPME), Karlsruhe 2013]

Scene Parsing/Labeling on RGB+Depth Images

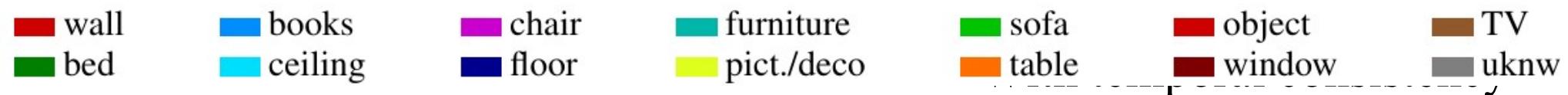
Y LeCun



Ground truths



Our results



[Couprie, Farabet, Najman, LeCun ICLR 2013, ICIP 2013]

Scene Parsing/Labeling on RGB+Depth Images

Y LeCun

wall	books	chair	furniture	sofa	object	TV
bed	ceiling	floor	pict./deco	table	window	uknw



Ground truths



Our results

Temporal consistency



(a) Output of the Multiscale convnet trained using depth information - frame by frame



(b) Results smoothed temporally using Couprie et al. (2013a)

[Couprie, Farabet, Najman, LeCun ICLR 2013]

[Couprie, Farabet, Najman, LeCun ICIP 2013]

[Couprie, Farabet, Najman, LeCun submitted to JMLR]



Semantic Segmentation on RGB+D Images and Videos

Y LeCun



[Couprie, Farabet, Najman, LeCun ICLR 2013, ICIP 2013]

Self-Learning ConvNet for Scene Labeling

**Vision-Based Navigation
for Off-Road Robots**

LAGR project: vision-based navigation for off-road robot

Y LeCun

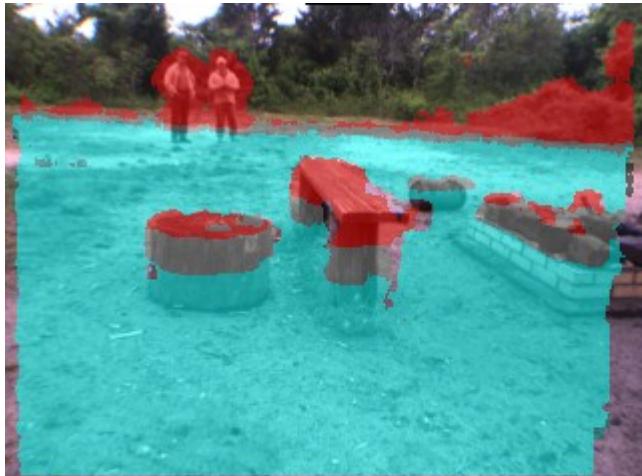
- Getting a robot to drive autonomously in unknown terrain solely from vision (camera input).
- Our team (NYU/Net-Scale Technologies Inc.) was one of 8 participants funded by DARPA
- All teams received identical robots and can only modify the software (not the hardware)
- The robot is given the GPS coordinates of a goal, and must drive to the goal as fast as possible. The terrain is unknown in advance. The robot is run 3 times through the same course.
- Long-Range Obstacle Detection with on-line, self-trained ConvNet
- Uses temporal consistency!



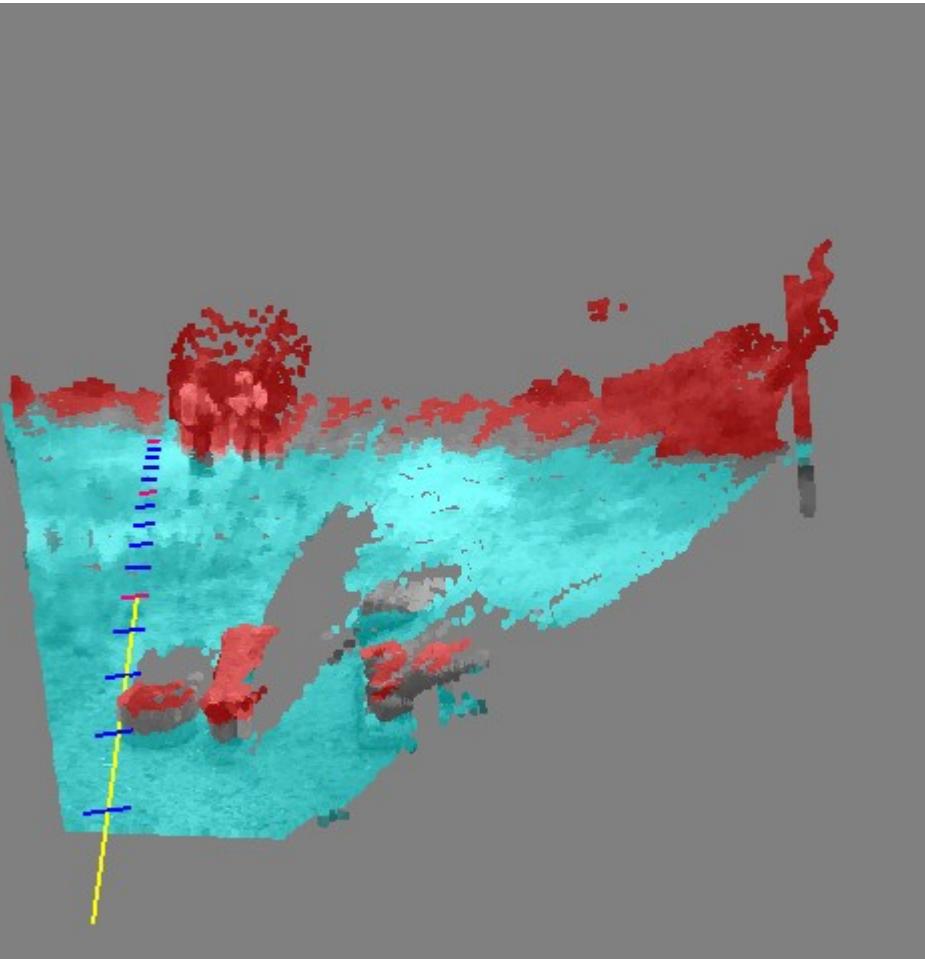
Obstacle Detection at Short Range: Stereovision

Y LeCun

Obstacles overlaid with camera image



Camera image

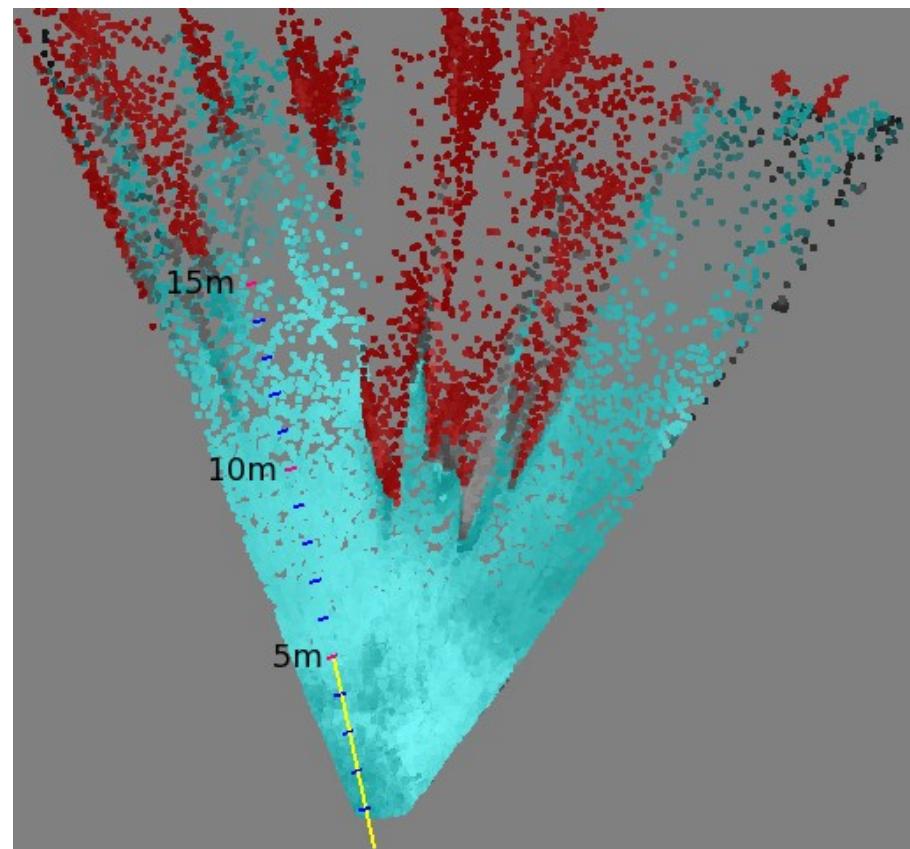


Detected obstacles (red)

But Stereovision Doesn't work at long range

Y LeCun

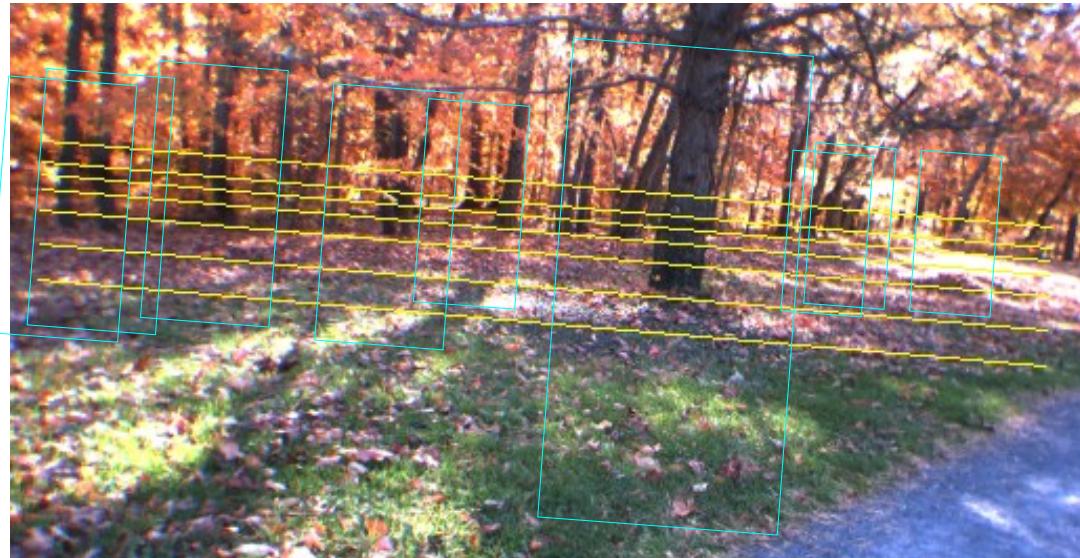
- Stereo is only good up to about 10 meters.
- But not seeing past 10 meters is like driving in a fog or a snowstorm!



Ground Blizzard Negaunee, MI

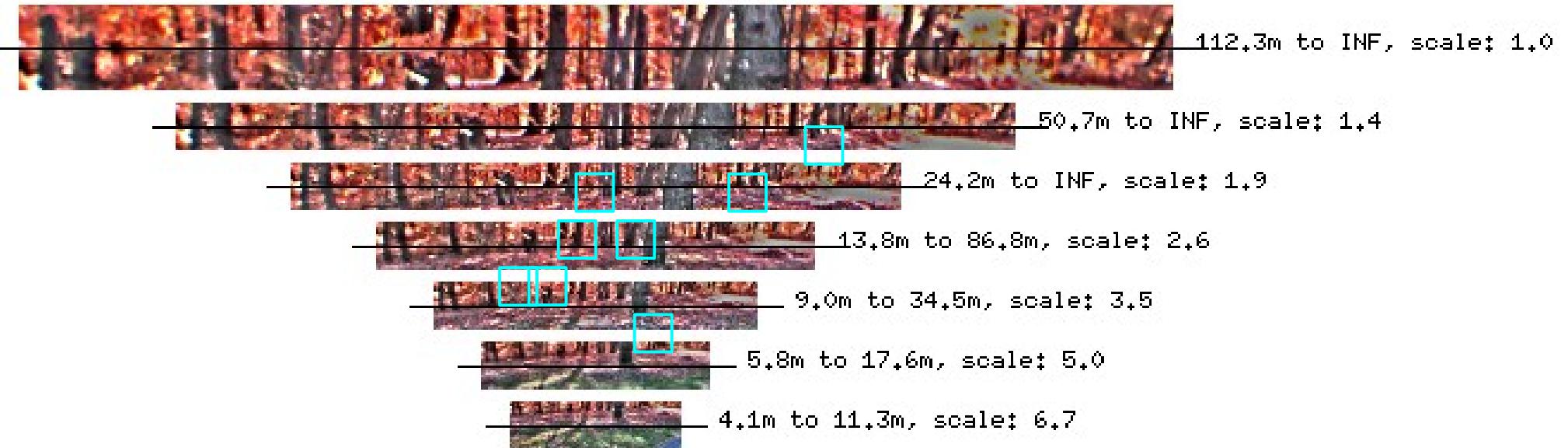
Long Range Vision with a Convolutional Net

Y LeCun



Pre-processing (125 ms)

- Ground plane estimation
- Horizon leveling
- Conversion to YUV + local contrast normalization
- Scale invariant pyramid of distance-normalized image “bands”



Convolutional Net Architecture

Y LeCun

100 features per
3x12x25 input window

YUV image band
20-36 pixels tall,
36-500 pixels wide

20@30x484

3@36x484

YUV input

100@25x121

20@30x125

CONVOLUTIONS (6x5)

MAX SUBSAMPLING (1x4)

CONVOLUTIONS (7x6)





Scene Labeling with ConvNet + online learning

Y LeCun

Image Labeling for Off-Road Robots [Hadsell JFR 2008]

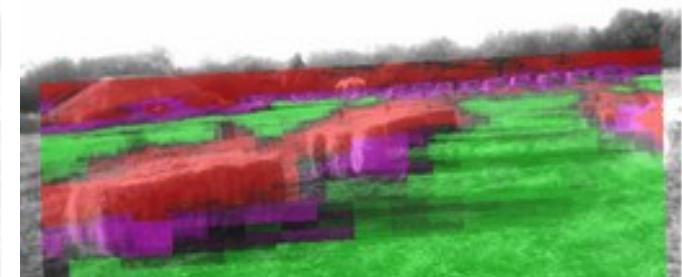
- ▶ ConvNet labels pixels as one of 3 categories
- ▶ Traversable/flat (green), non traversible (red), foot of obstacle (purple)
- ▶ Labels obtained from stereo vision and SLAM



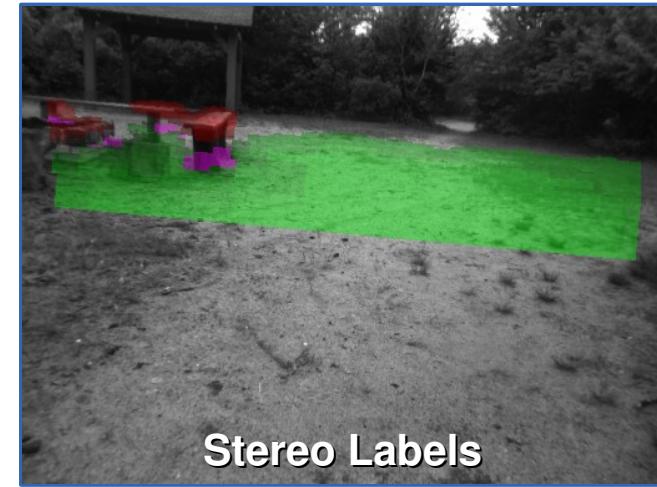
Input image



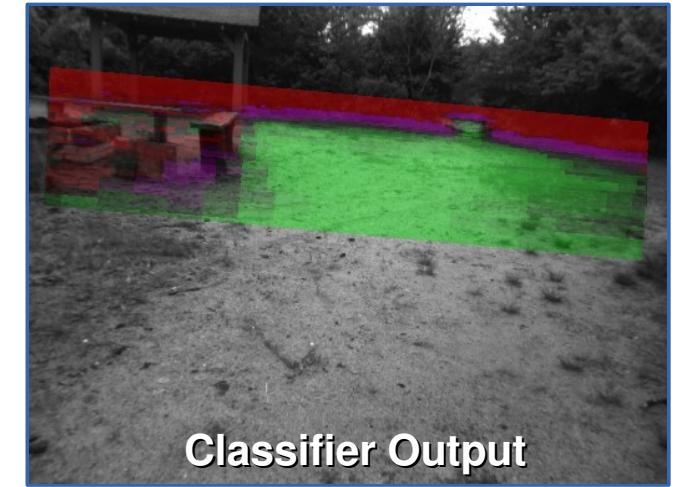
Stereo Labels



Input image



Stereo Labels



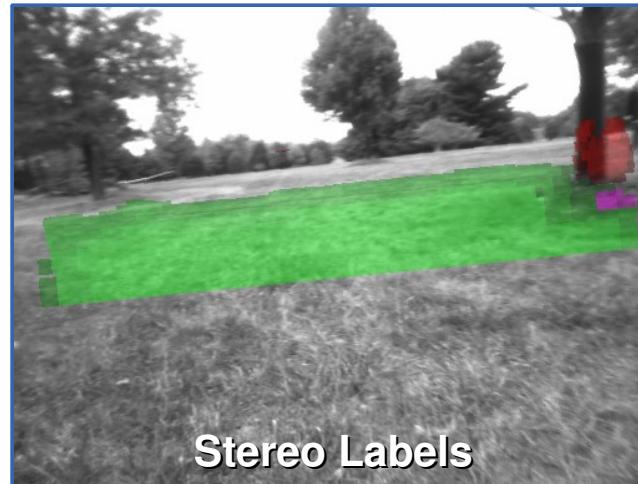
Classifier Output

Long Range Vision Results

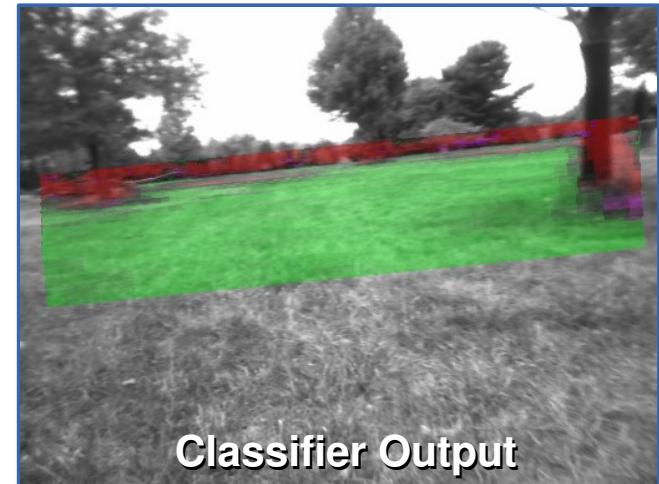
Y LeCun



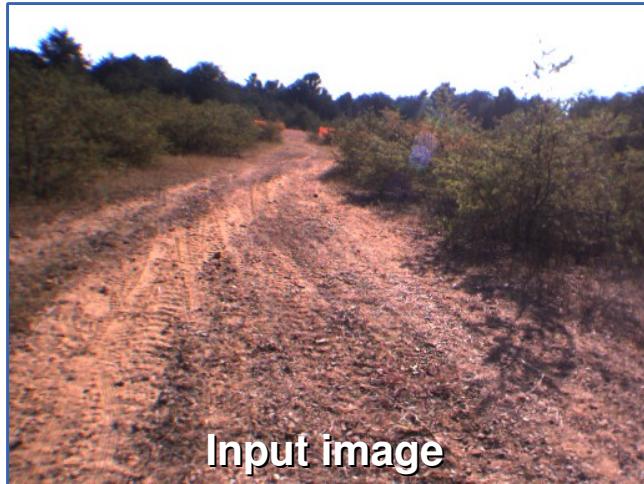
Input image



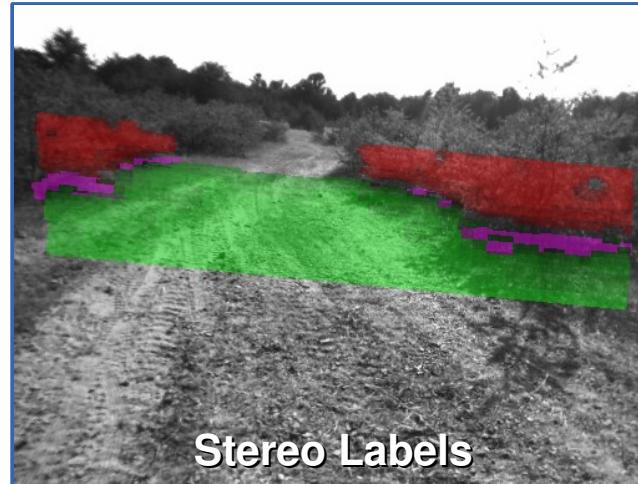
Stereo Labels



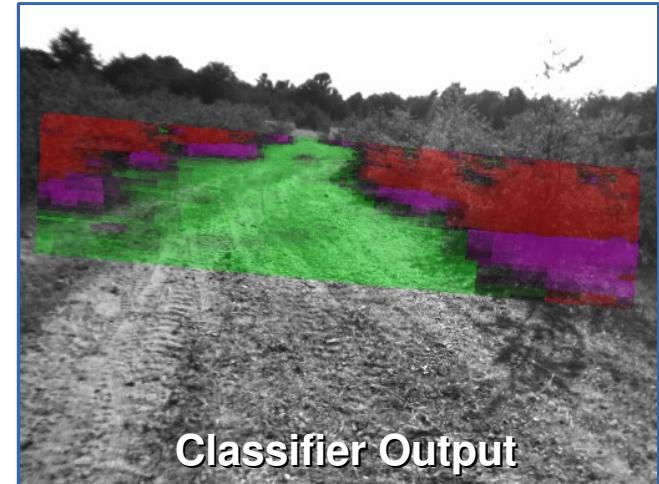
Classifier Output



Input image



Stereo Labels



Classifier Output

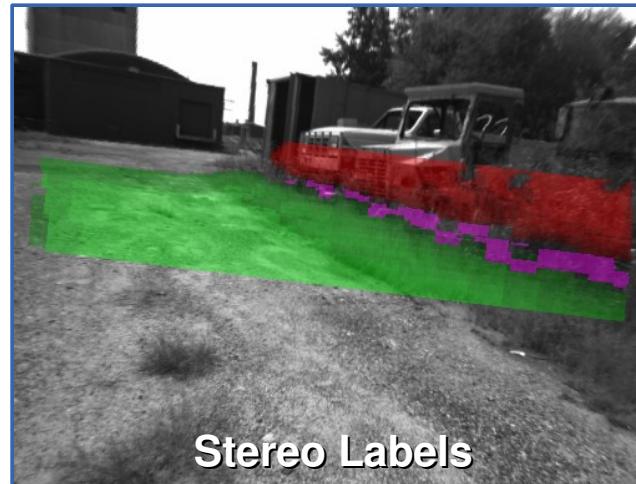


Long Range Vision Results

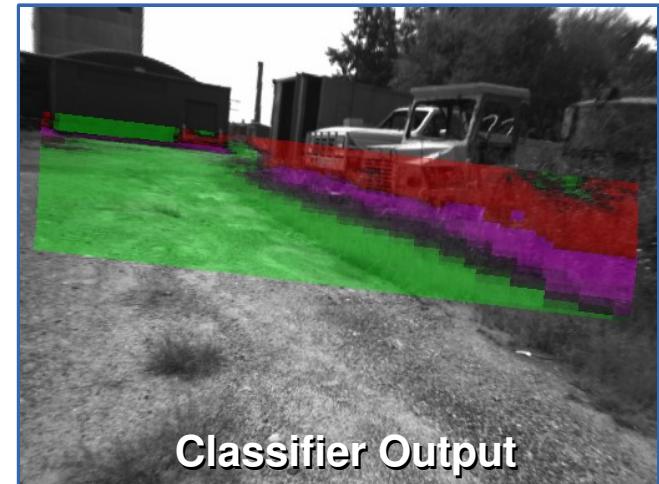
Y LeCun



Input image



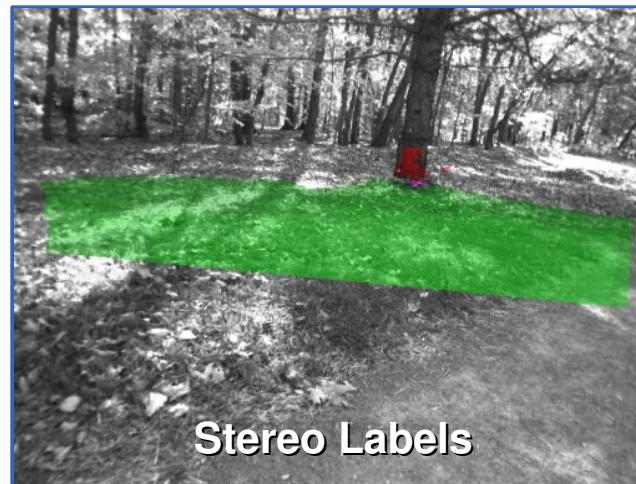
Stereo Labels



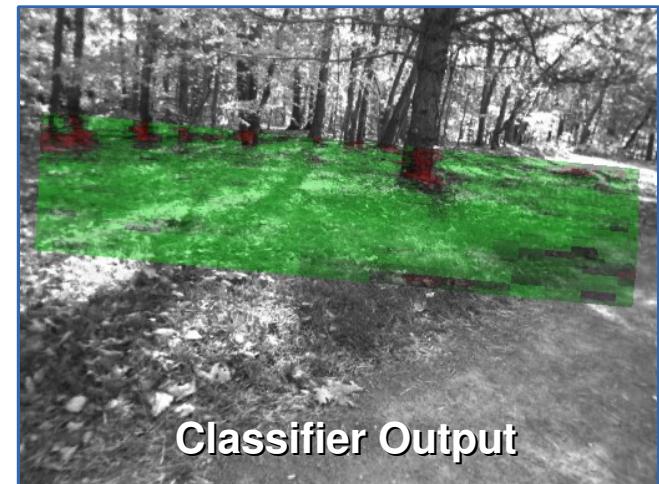
Classifier Output



Input image



Stereo Labels



Classifier Output



Commercial Applications of Convolutional Nets

Y LeCun

- Form Reading: AT&T 1994
- Check reading: AT&T/NCR 1996 (read 10-20% of all US checks in 2000)
- Handwriting recognition: Microsoft early 2000
- Face and person detection: NEC 2005, France Telecom late 2000s.
- Gender and age recognition: NEC 2010 (vending machines)
- OCR in natural images: Google 2013 (StreetView house numbers)
- Photo tagging: Google 2013
- Image Search by Similarity: Baidu 2013
- Since early 2014, the number of deployed applications of ConvNets has exploded
- Many applications at Facebook, Google, Baidu, Microsoft, IBM, NEC, Yahoo.....
 - ▶ Speech recognition, face recognition, image search, content filtering/ranking,....
- Tens of thousands of servers run ConvNets continuously every day.



Software Tools and Hardware Acceleration for Convolutional Networks



Software Platform for Deep Learning: Torch7

Y LeCun

Torch7

- ▶ based on the **LuaJIT** language
- ▶ Simple and lightweight dynamic language (widely used for games)
- ▶ Multidimensional array library with CUDA and OpenMP backends
- ▶ FAST: Has a native just-in-time compiler
- ▶ Has an unbelievably nice foreign function interface to call C/C++ functions from Lua

Torch7 is an extension of Lua with

- ▶ Multidimensional array engine
- ▶ A machine learning library that implements multilayer nets, convolutional nets, unsupervised pre-training, etc
- ▶ Various libraries for data/image manipulation and computer vision
- ▶ Used at Facebook Ai Research, Google (Deep Mind, Brain), Intel, and many academic groups and startups

Single-line installation on Ubuntu and Mac OSX:

- ▶ <http://torch.ch>

Torch7 Cheat sheet (with links to libraries and tutorials):

- <https://github.com/torch/torch7/wiki/Cheatsheet>

Example: building a Neural Net in Torch7

Net for SVHN digit recognition

10 categories

Input is 32x32 RGB (3 channels)

1500 hidden units

Creating a 2-layer net

Make a cascade module

Reshape input to vector

Add Linear module

Add tanh module

Add Linear Module

Add log softmax layer

```
Noutputs = 10;  
nfeats = 3; Width = 32; height = 32  
ninputs = nfeats*width*height  
nhiddens = 1500
```

```
-- Simple 2-layer neural network  
model = nn.Sequential()  
model:add(nn.Reshape(ninputs))  
model:add(nn.Linear(ninputs,nhiddens))  
model:add(nn.Tanh())  
model:add(nn.Linear(nhiddens,noutputs))  
model:add(nn.LogSoftMax())
```

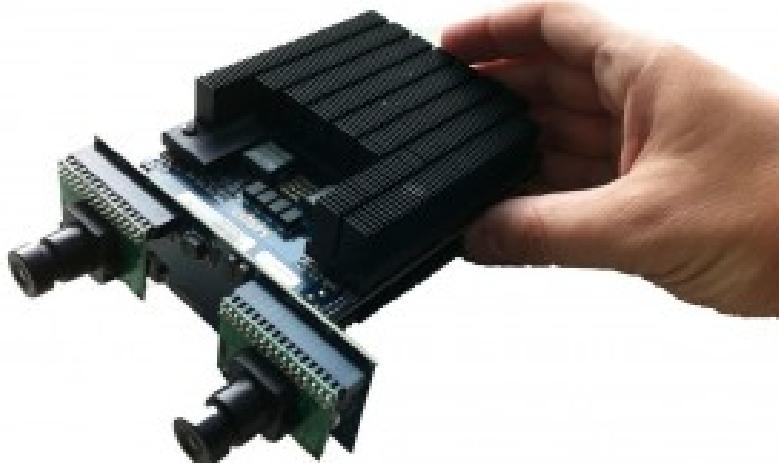
```
criterion = nn.ClassNLLCriterion()
```

See Torch7 example at <http://bit.ly/16tyLAX>

NeuFlow architecture (NYU + Purdue)

Y LeCun

- Collaboration NYU-Purdue: Eugenio Culurciello's e-Lab.
- Running on Picocomputing 8x10cm high-performance FPGA board
 - ▶ Virtex 6 LX240T: 680 MAC units, 20 neuflow tiles
- Full scene labeling at 20 frames/sec (50ms/frame) at 320x240



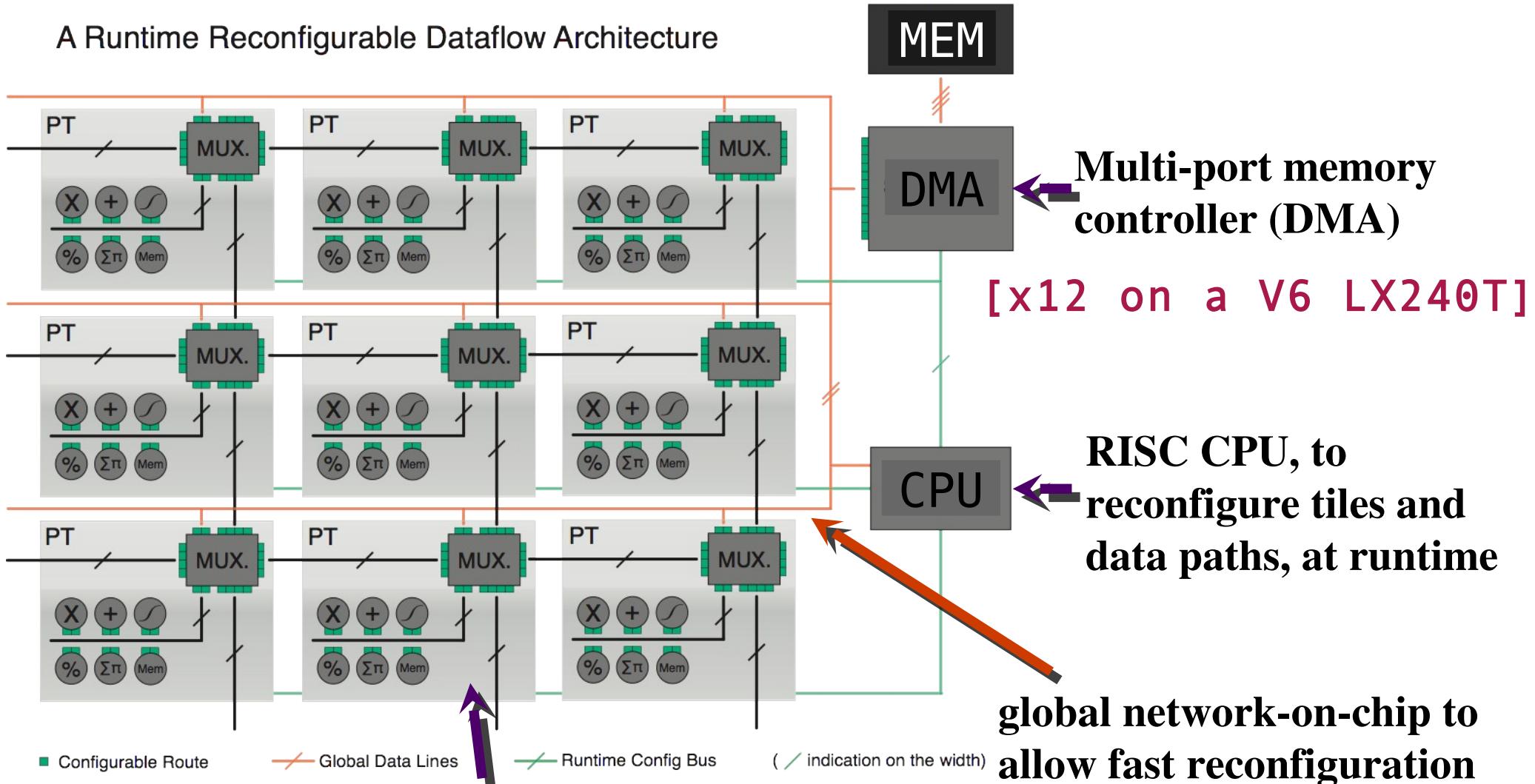
board with Virtex-6



NewFlow: Architecture

Y LeCun

A Runtime Reconfigurable Dataflow Architecture



grid of passive processing tiles (PTs)
[x20 on a Virtex6 LX240T]

Multi-port memory controller (DMA)
[x12 on a V6 LX240T]

RISC CPU, to reconfigure tiles and data paths, at runtime

global network-on-chip to allow fast reconfiguration

NewFlow: Processing Tile Architecture

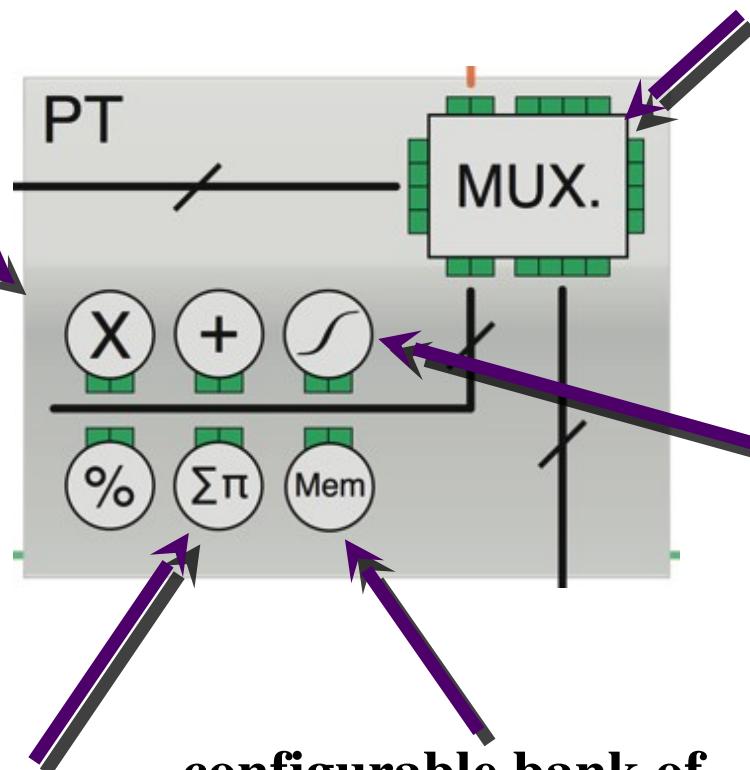
Y LeCun

Term-by-term
streaming operators
(MUL,DIV,ADD,SU
B,MAX)

[x8, 2 per tile]

full 1/2D parallel convolver
with 100 MAC units

[x4]



configurable router,
to stream data in
and out of the tile, to
neighbors or DMA
ports

[x20]

configurable piece-wise
linear or quadratic
mapper

[x4]

configurable bank of
FIFOs , for stream
buffering, up to 10kB
per PT

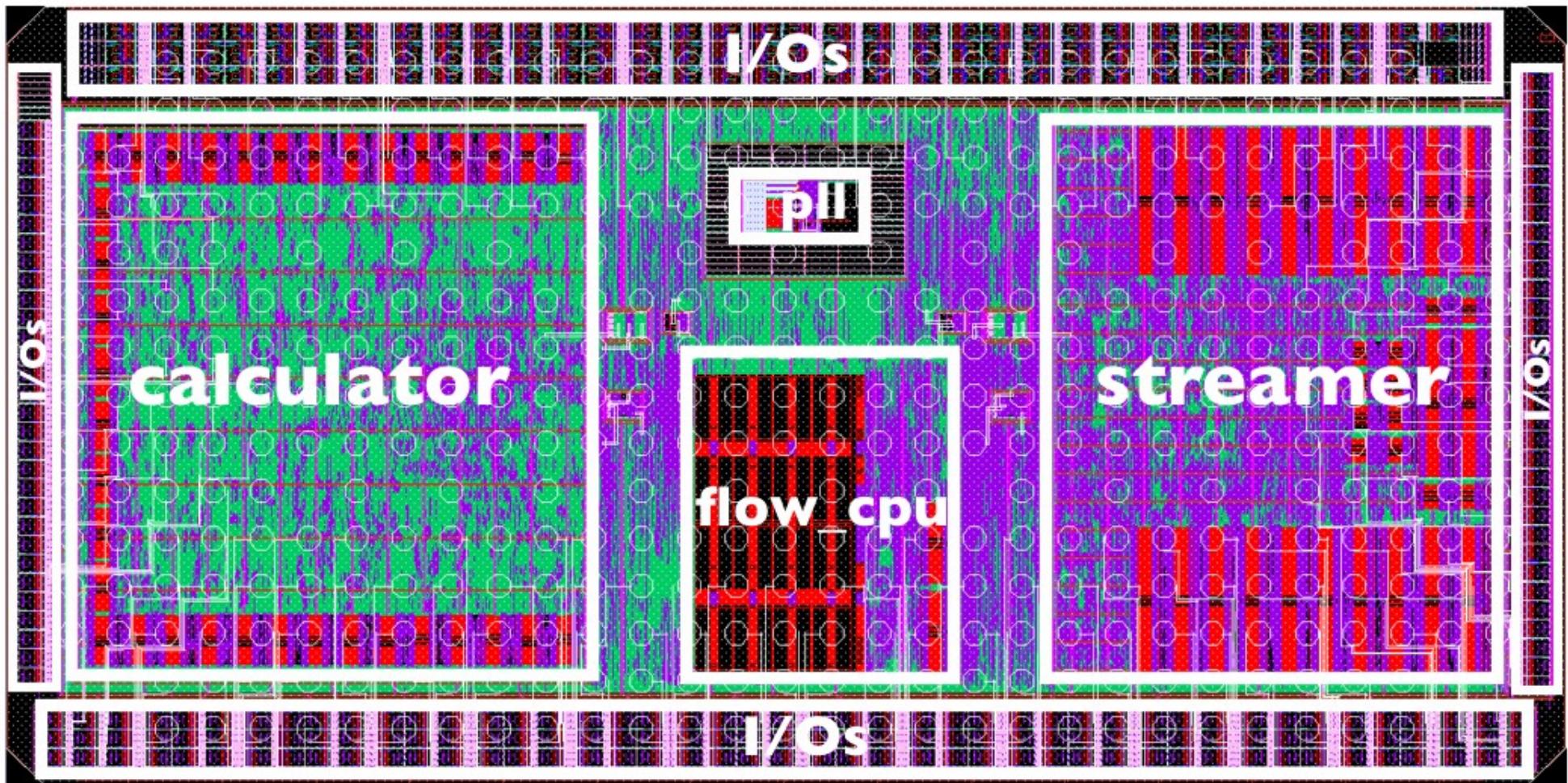
[x8]

[Virtex6 LX240T]

NewFlow ASIC: 2.5x5 mm, 45nm, 0.6Watts, >300GOPS

Y LeCun

- Collaboration Purdue-NYU: Eugenio Culurciello's e-Lab
- Suitable for vision-enabled embedded and mobile devices
- (but the fabrication was botched...)



[Pham, Jelaca, Farabet, Martini, LeCun, Culurciello 2012]

NewFlow: Performance

Y LeCun

	Intel I7 4 cores	neuFlow Virtex4	neuFlow Virtex 6	nVidia GT335m	NeuFlow ASIC 45nm	nVidia GTX480*
Peak GOP/sec	40	40	160	182	320	1350
Actual GOP/sec	12	37	147	54	300	294
FPS	14	46	182	67	364	374
Power (W)	50	10	10	30	0.6	220
Embed? (GOP/s/W)	0.24	3.7	14.7	1.8	490	1.34

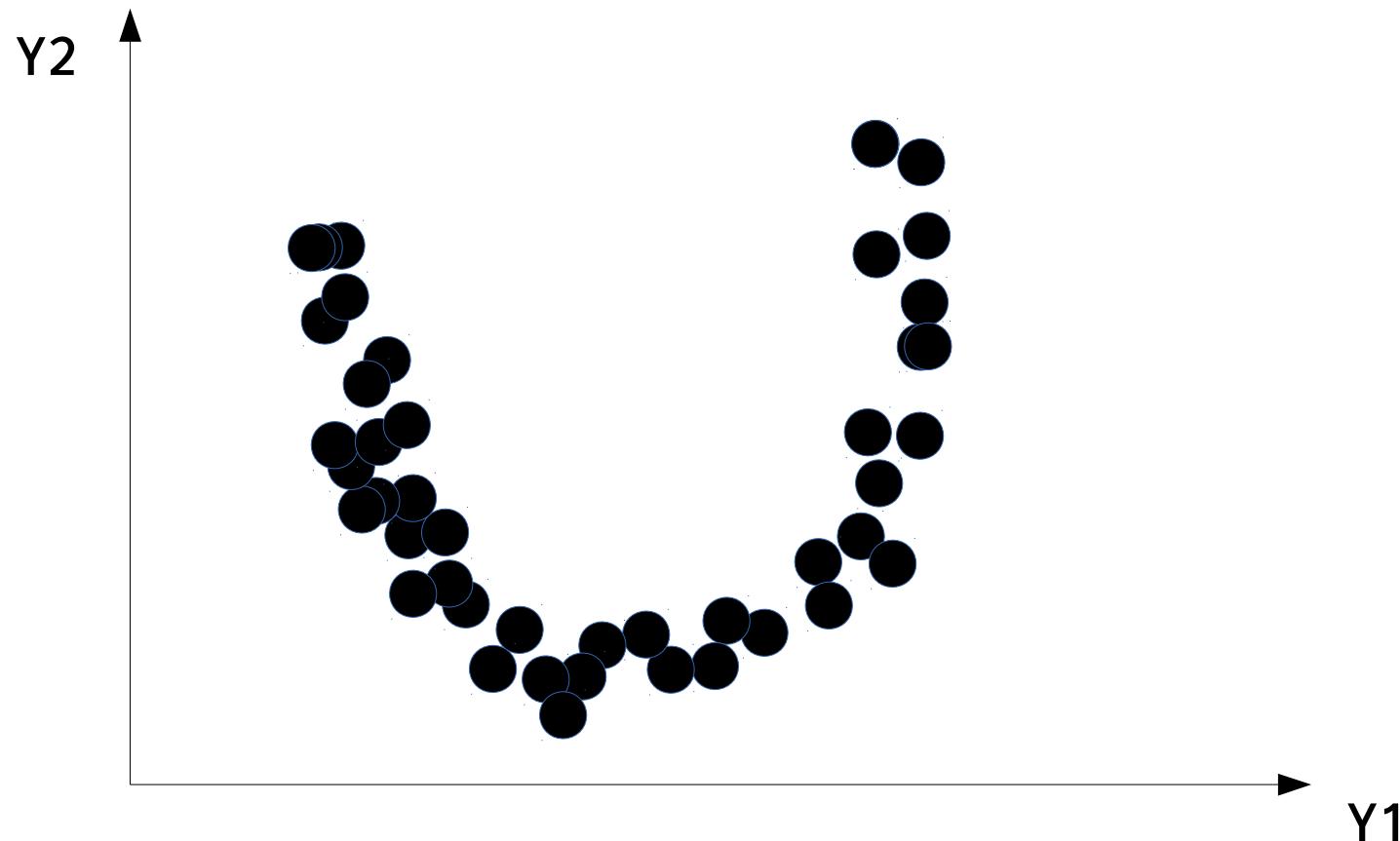
- NeuFlow Virtex6 can run the semantic labeling system at 50ms/frame
- * performance of Nvidia GPU is higher when using minibatch training

Unsupervised Learning

Energy-Based Unsupervised Learning

Y LeCun

- Learning an energy function (or contrast function) that takes
 - ▶ Low values on the data manifold
 - ▶ Higher values everywhere else

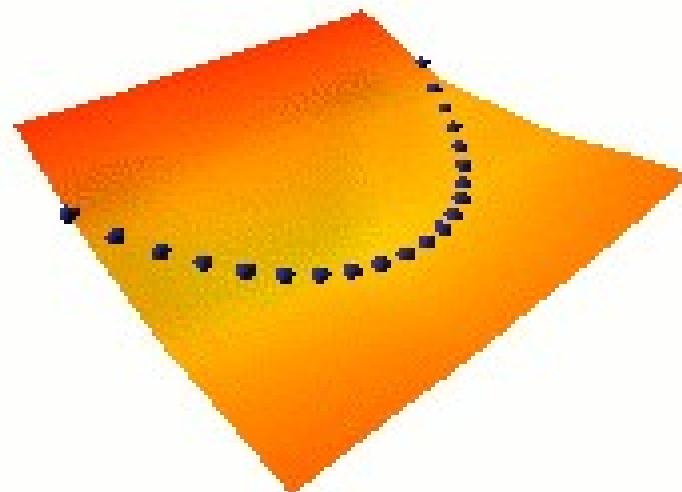
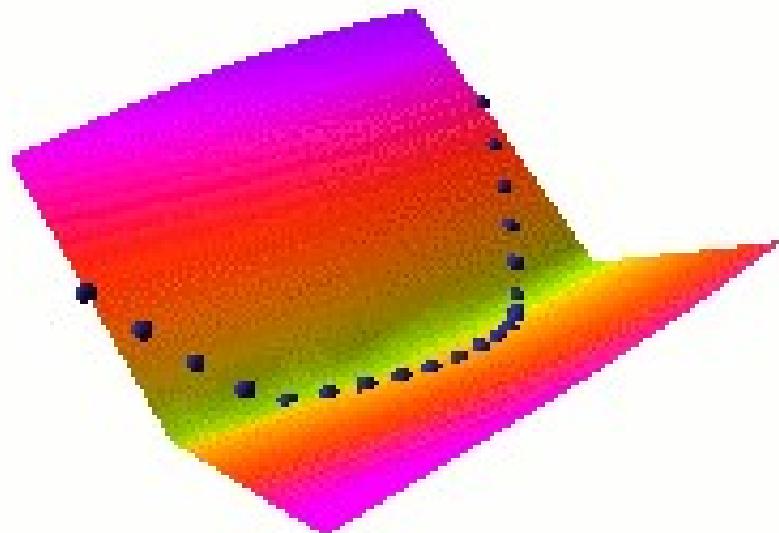


Learning the Energy Function

Y LeCun

parameterized energy function $E(Y,W)$

- ▶ Make the energy low on the samples
- ▶ Make the energy higher everywhere else
- ▶ Making the energy low on the samples is easy
- ▶ But how do we make it higher everywhere else?



Sparse Modeling: Sparse Coding + Dictionary Learning

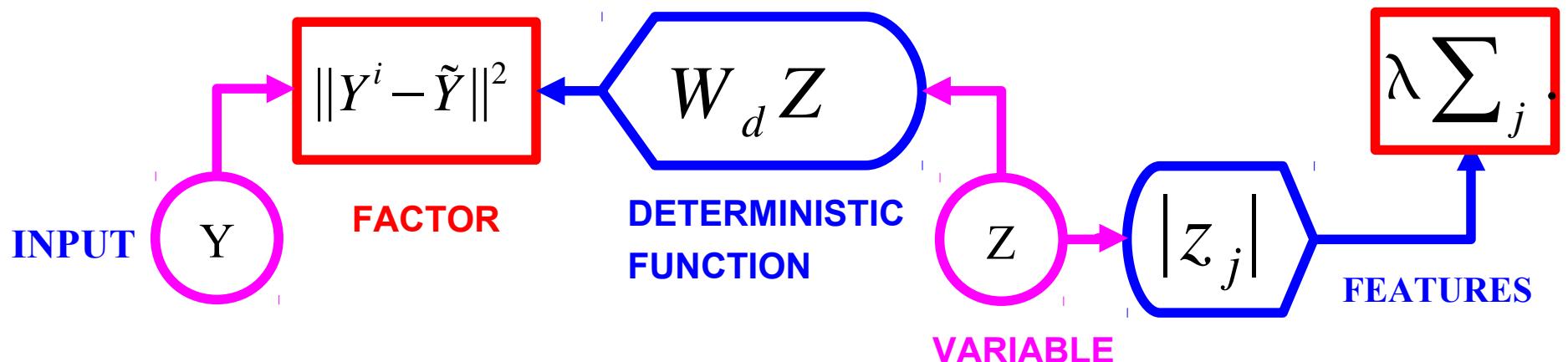
Y LeCun

[Olshausen & Field 1997]

■ Sparse linear reconstruction

■ Energy = reconstruction_error + code_prediction_error + code_sparsity

$$E(Y^i, Z) = \|Y^i - W_d Z\|^2 + \lambda \sum_j |z_j|$$



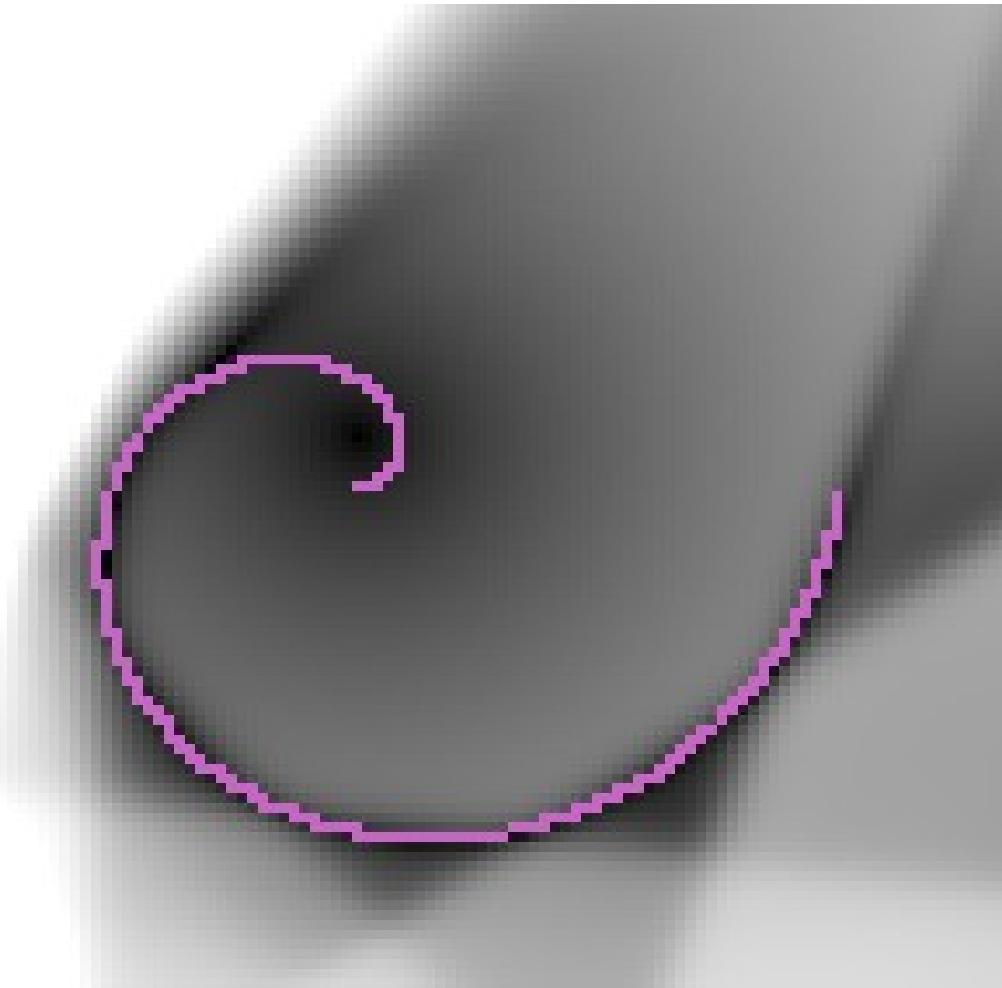
■ Inference is expensive: ISTA/FISTA, CGIHT, coordinate descent....

$$Y \rightarrow \hat{Z} = \operatorname{argmin}_Z E(Y, Z)$$

#6. use a regularizer that limits the volume of space that has low energy

Y LeCun

Sparse coding, sparse auto-encoder, Predictive Sparse Decomposition

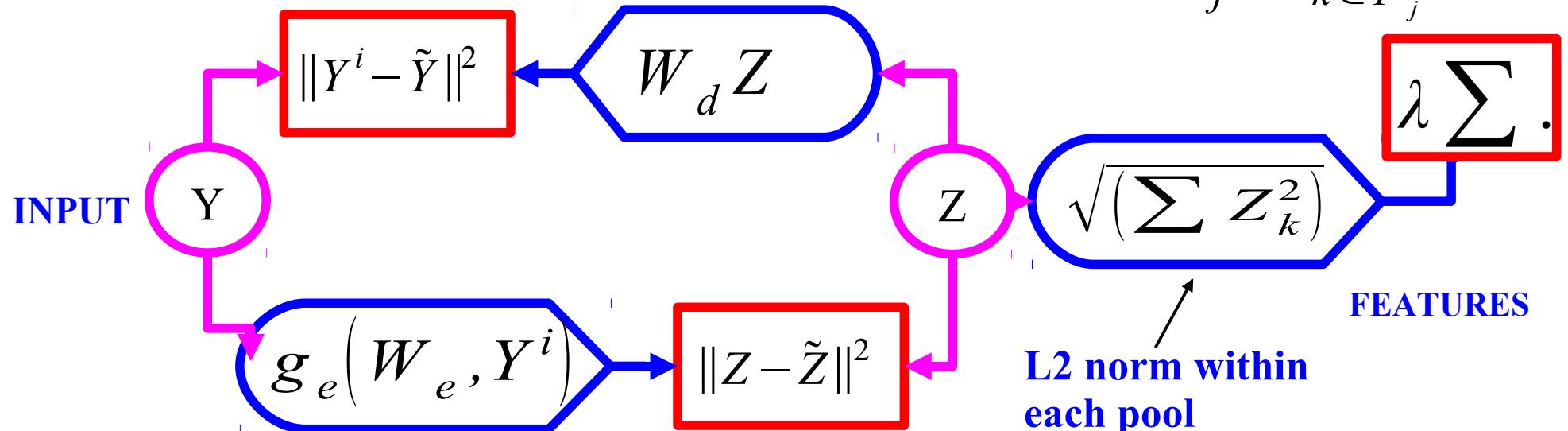


Regularized Encoder-Decoder Model (auto-Encoder) for Unsupervised Feature Learning

Y LeCun

- Encoder: computes feature vector Z from input X
- Decoder: reconstructs input X from feature vector Z
- Feature vector: high dimensional and regularized (e.g. sparse)
- Factor graph with energy function $E(X,Z)$ with 3 terms:
 - ▶ Linear decoding function and reconstruction error
 - ▶ Non-Linear encoding function and prediction error term
 - ▶ Pooling function and regularization term (e.g. sparsity)

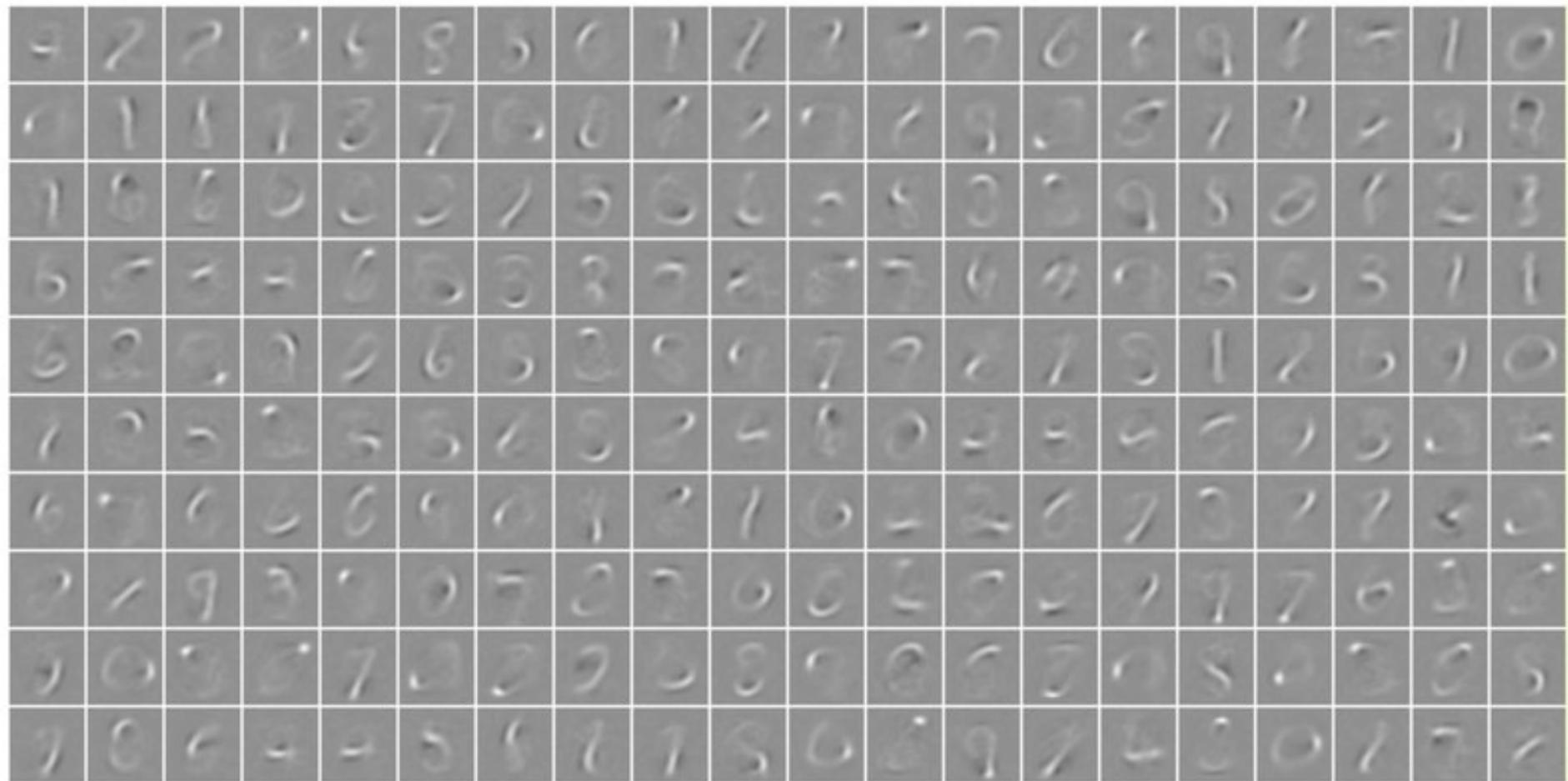
$$E(Y, Z) = \|Y - W_d Z\|^2 + \|Z - g_e(W_e, Y)\|^2 + \sum_j \sqrt{\sum_{k \in P_j} Z_k^2}$$



PSD: Basis Functions on MNIST

Y LeCun

- Basis functions (and encoder matrix) are digit parts

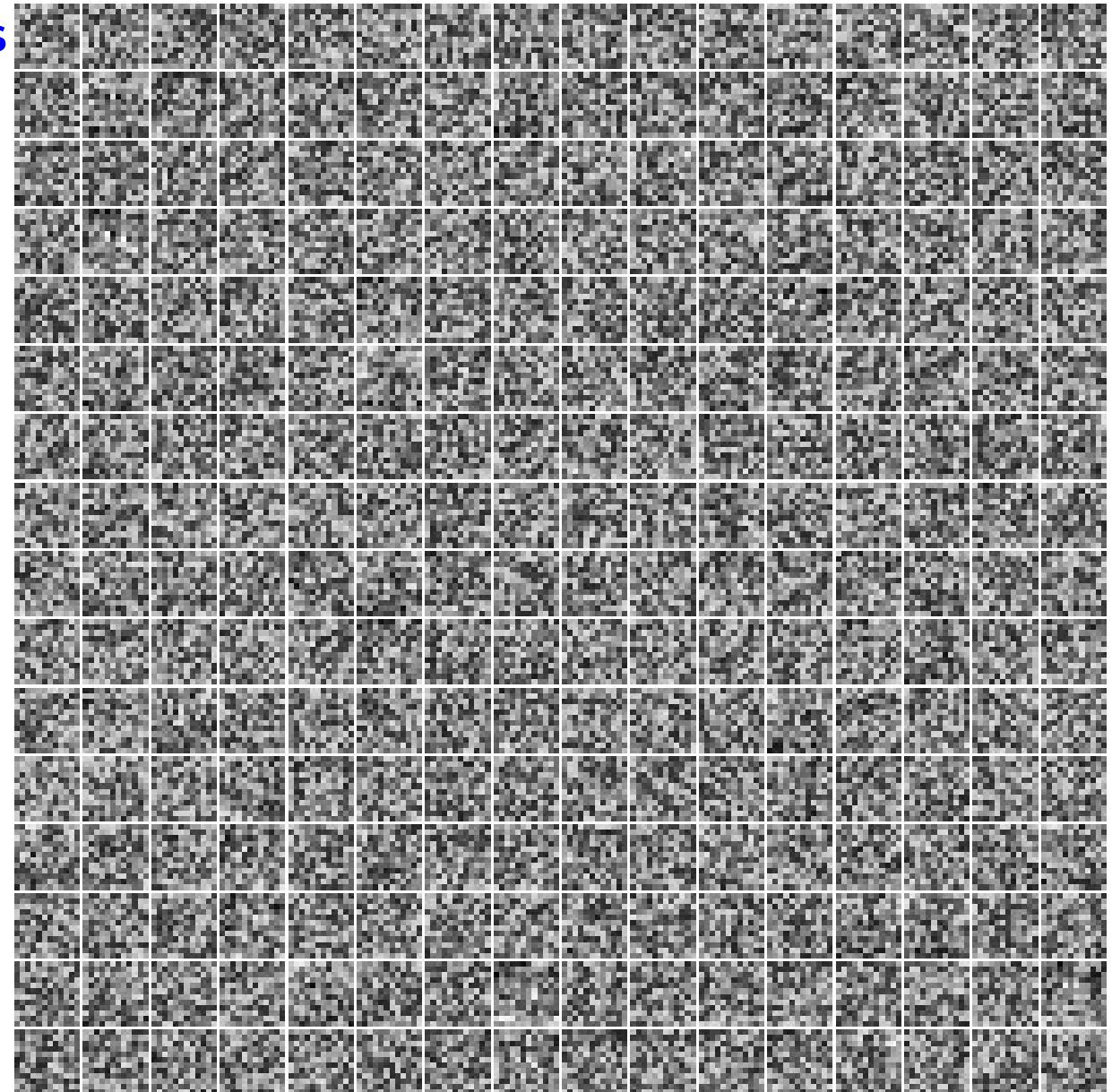


Predictive Sparse Decomposition (PSD): Training

Y LeCun

- Training on natural images patches.

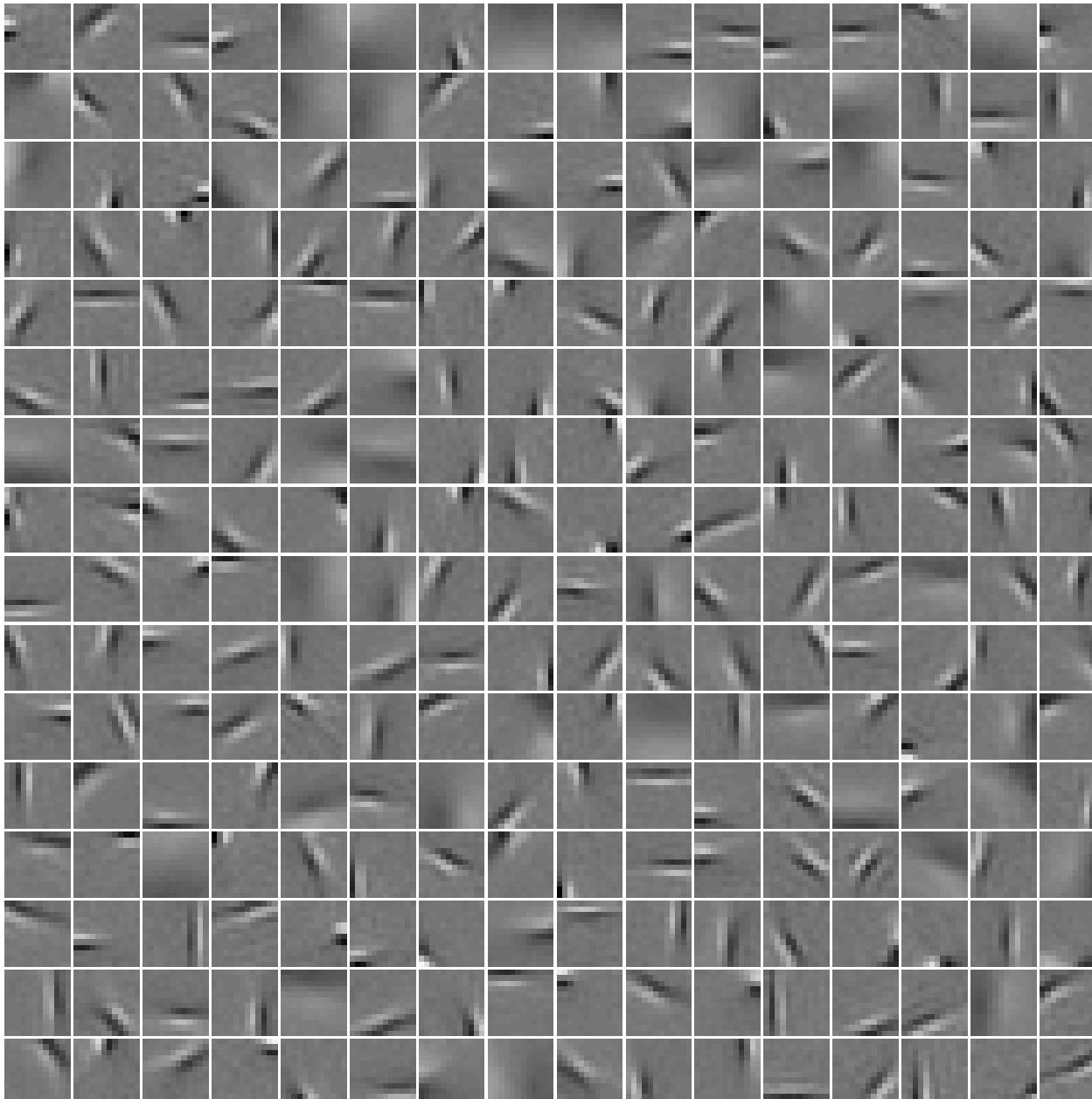
- ▶ 12X12
- ▶ 256 basis functions



iteration no 0

Learned Features on natural patches: V1-like receptive fields

Y LeCun



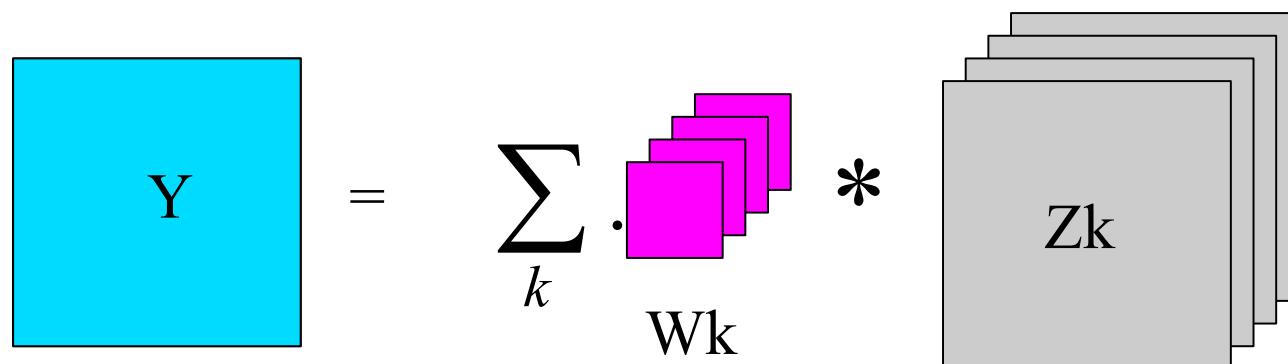
Convolutional Sparse Coding

- Replace the dot products with dictionary element by convolutions.

- Input Y is a full image
- Each code component Z_k is a feature map (an image)
- Each dictionary element is a convolution kernel

- Regular sparse coding $E(Y, Z) = \left\| Y - \sum_k W_k Z_k \right\|^2 + \alpha \sum_k |Z_k|$

- Convolutional S.C. $E(Y, Z) = \left\| Y - \sum_k W_k * Z_k \right\|^2 + \alpha \sum_k |Z_k|$



“deconvolutional networks” [Zeiler, Taylor, Fergus CVPR 2010]

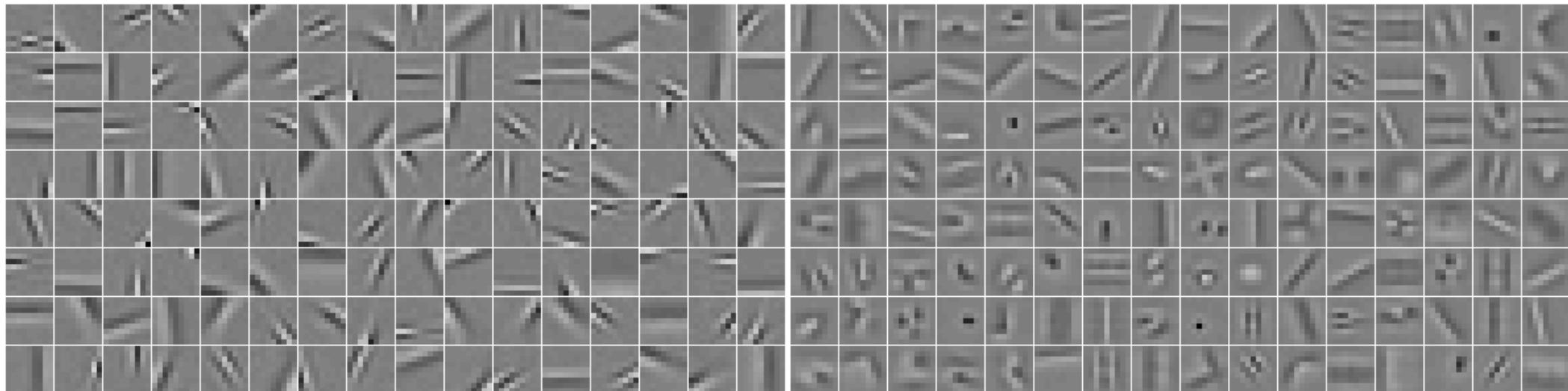
Convolutional PSD: Encoder with a soft sh() Function

Y LeCun

Convolutional Formulation

- Extend sparse coding from **PATCH** to **IMAGE**

$$\mathcal{L}(x, z, \mathcal{D}) = \frac{1}{2} \|x - \sum_{k=1}^K \mathcal{D}_k * z_k\|_2^2 + \sum_{k=1}^K \|z_k - f(W^k * x)\|_2^2 + |z|_1$$



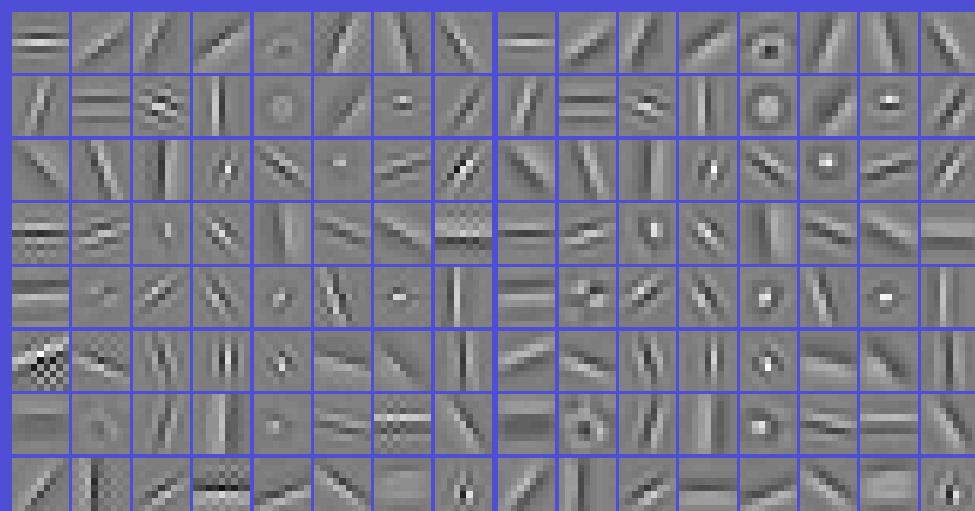
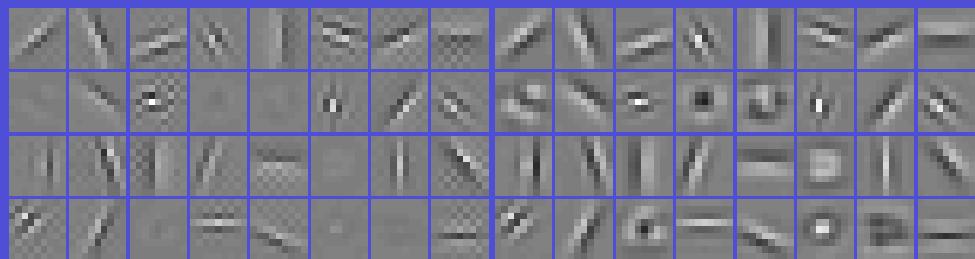
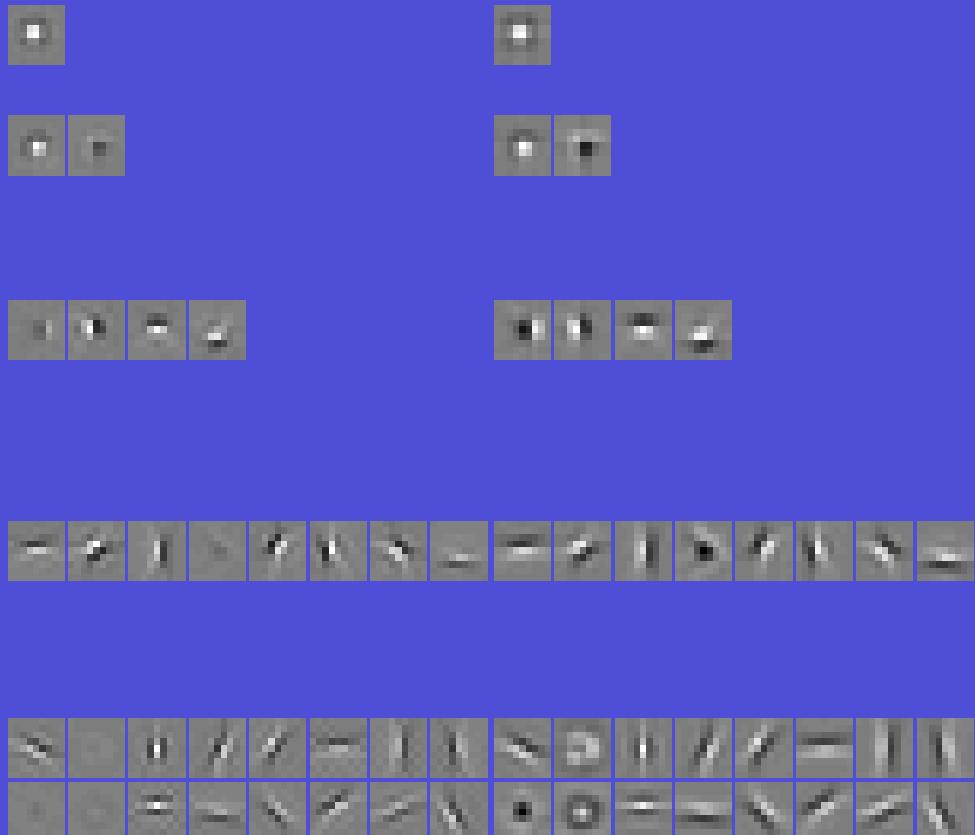
► **PATCH** based learning

► **CONVOLUTIONAL** learning

Convolutional Sparse Auto-Encoder on Natural Images

Y LeCun

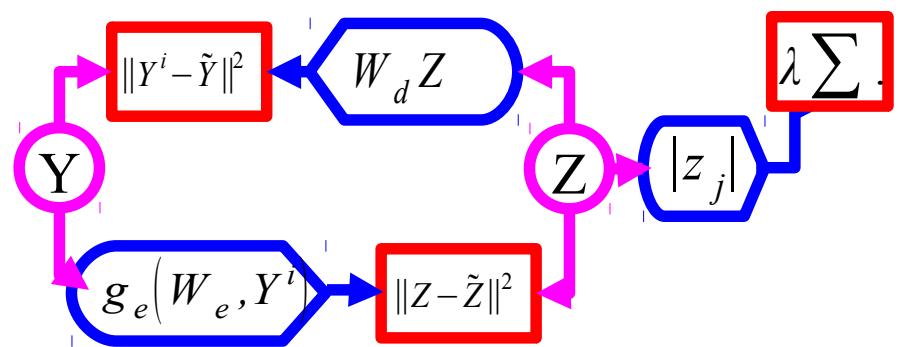
- Filters and Basis Functions obtained with 1, 2, 4, 8, 16, 32, and 64 filters.



Using PSD to Train a Hierarchy of Features

Y LeCun

Phase 1: train first layer using PSD

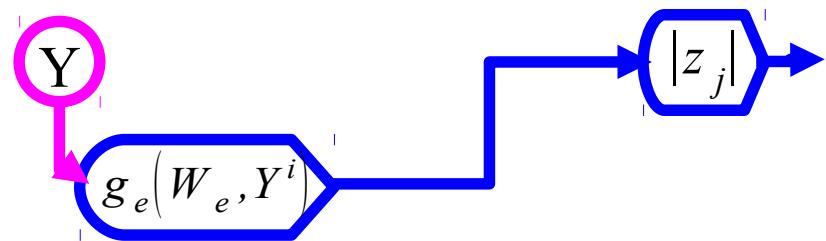


FEATURES

Using PSD to Train a Hierarchy of Features

Y LeCun

- Phase 1: train first layer using PSD
- Phase 2: use encoder + absolute value as feature extractor

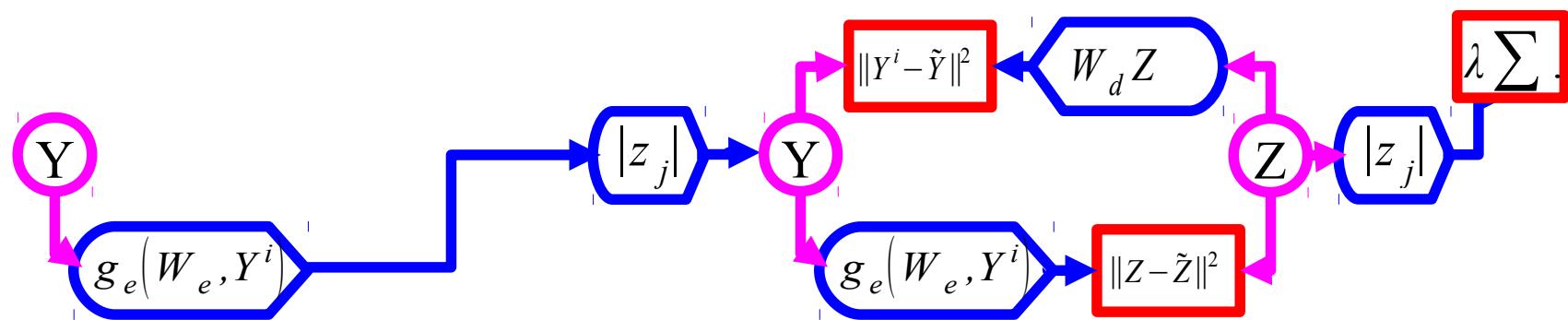


FEATURES

Using PSD to Train a Hierarchy of Features

Y LeCun

- Phase 1: train first layer using PSD
- Phase 2: use encoder + absolute value as feature extractor
- Phase 3: train the second layer using PSD

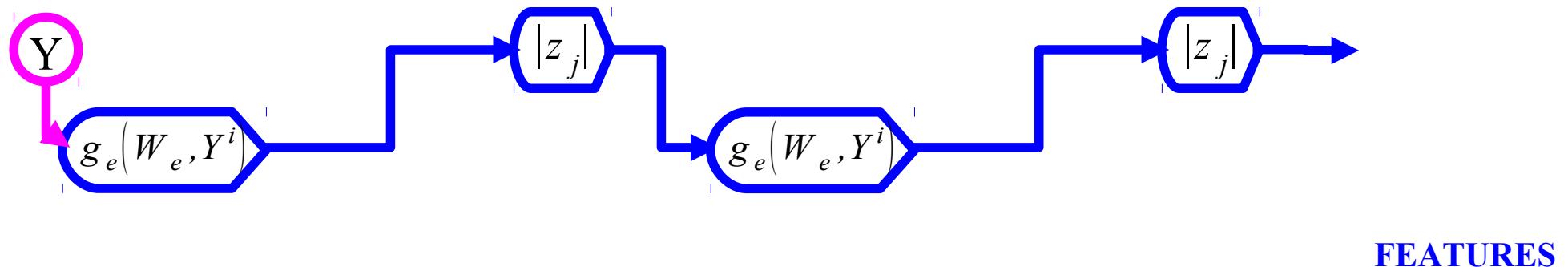


FEATURES

Using PSD to Train a Hierarchy of Features

Y LeCun

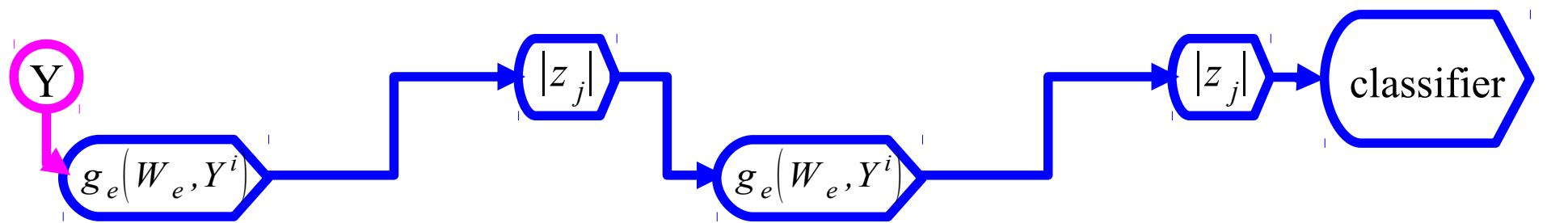
- Phase 1: train first layer using PSD
- Phase 2: use encoder + absolute value as feature extractor
- Phase 3: train the second layer using PSD
- Phase 4: use encoder + absolute value as 2nd feature extractor



Using PSD to Train a Hierarchy of Features

Y LeCun

- Phase 1: train first layer using PSD
- Phase 2: use encoder + absolute value as feature extractor
- Phase 3: train the second layer using PSD
- Phase 4: use encoder + absolute value as 2nd feature extractor
- Phase 5: train a supervised classifier on top
- Phase 6 (optional): train the entire system with supervised back-propagation



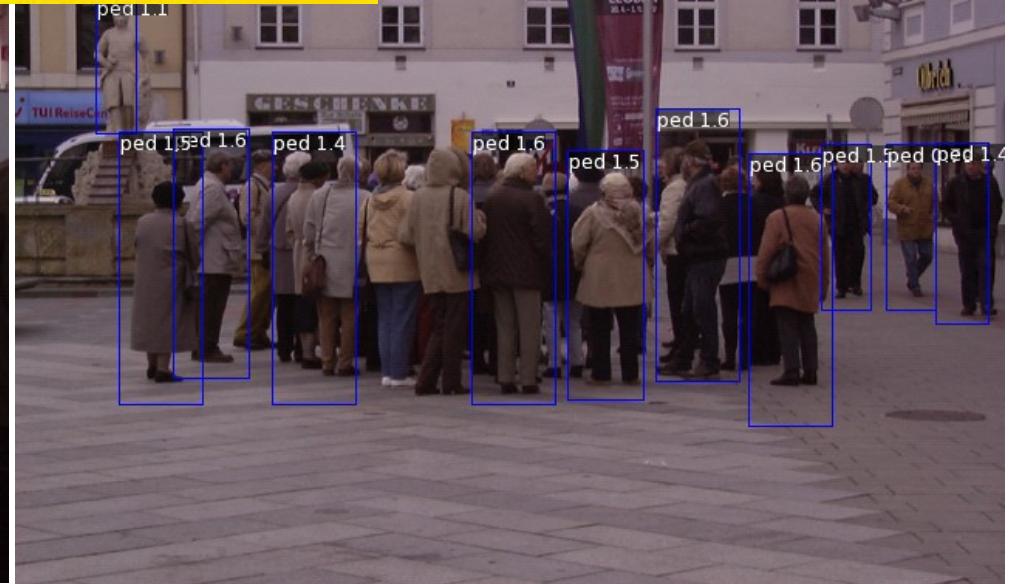
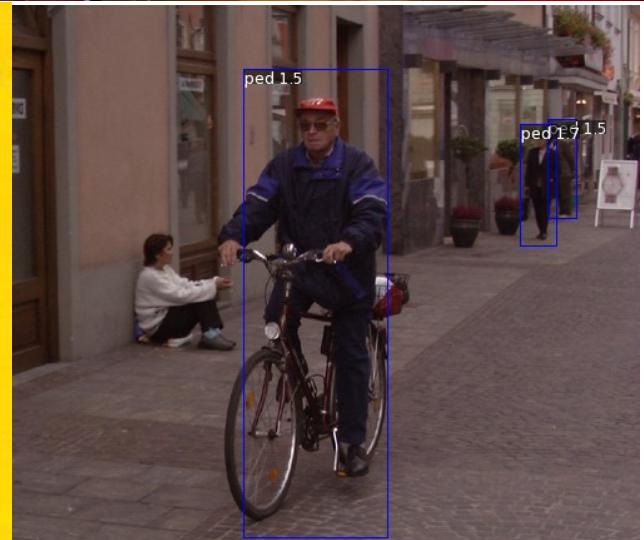
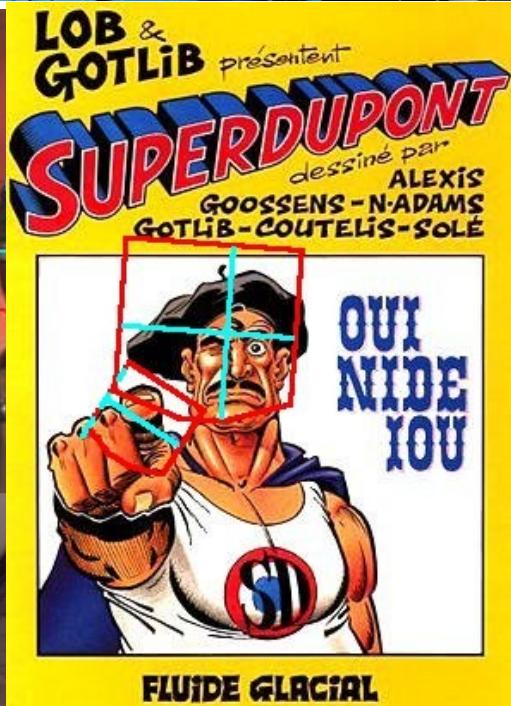
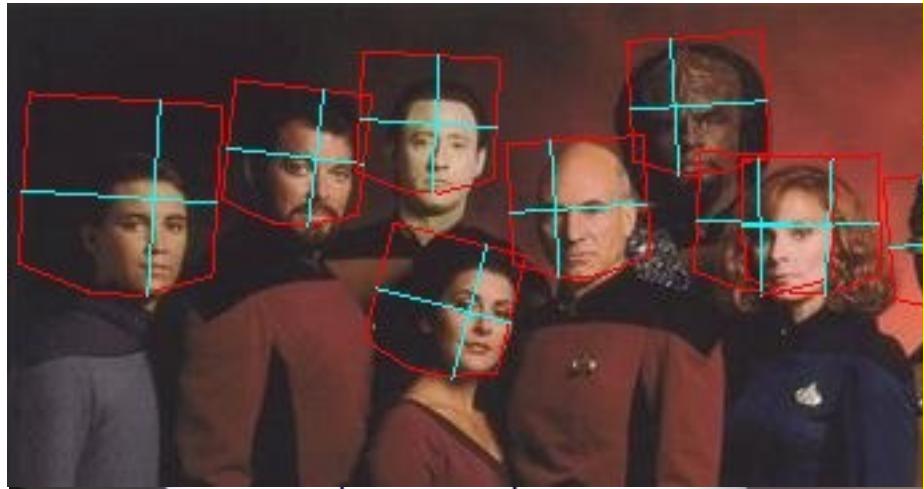
FEATURES



**Unsupervised + Supervised
For
Pedestrian Detection**

Pedestrian Detection, Face Detection

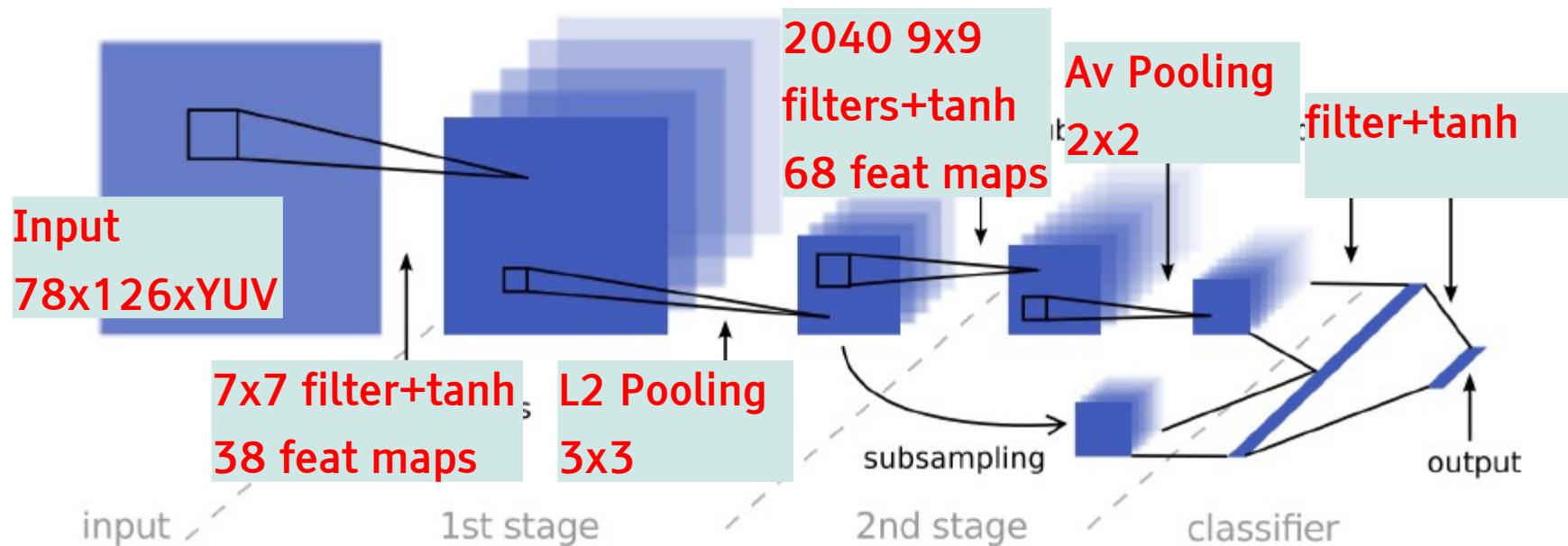
Y LeCun



ConvNet Architecture with Multi-Stage Features

Y LeCun

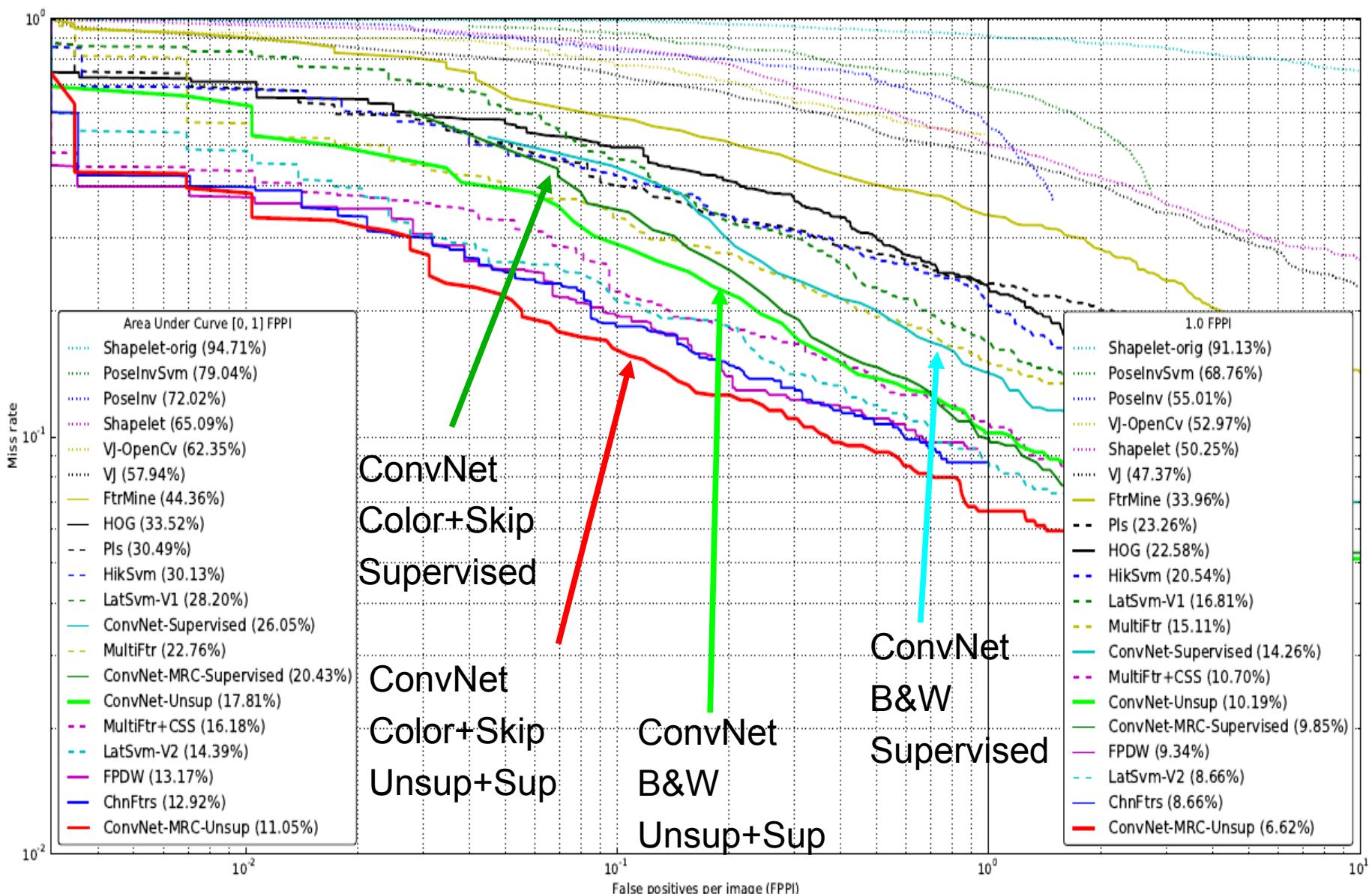
- Feature maps from all stages are pooled/subsampled and sent to the final classification layers
 - Pooled low-level features: good for textures and local motifs
 - High-level features: good for “gestalt” and global shape



Task	Single-Stage features	Multi-Stage features	Improvement %
Pedestrians detection (INRIA)	14.26%	9.85%	31%
Traffic Signs classification (GTSRB) [33]	1.80%	0.83%	54%
House Numbers classification (SVHN) [32]	5.54%	5.36%	3.2%

Pedestrian Detection: INRIA Dataset. Miss rate vs false positives

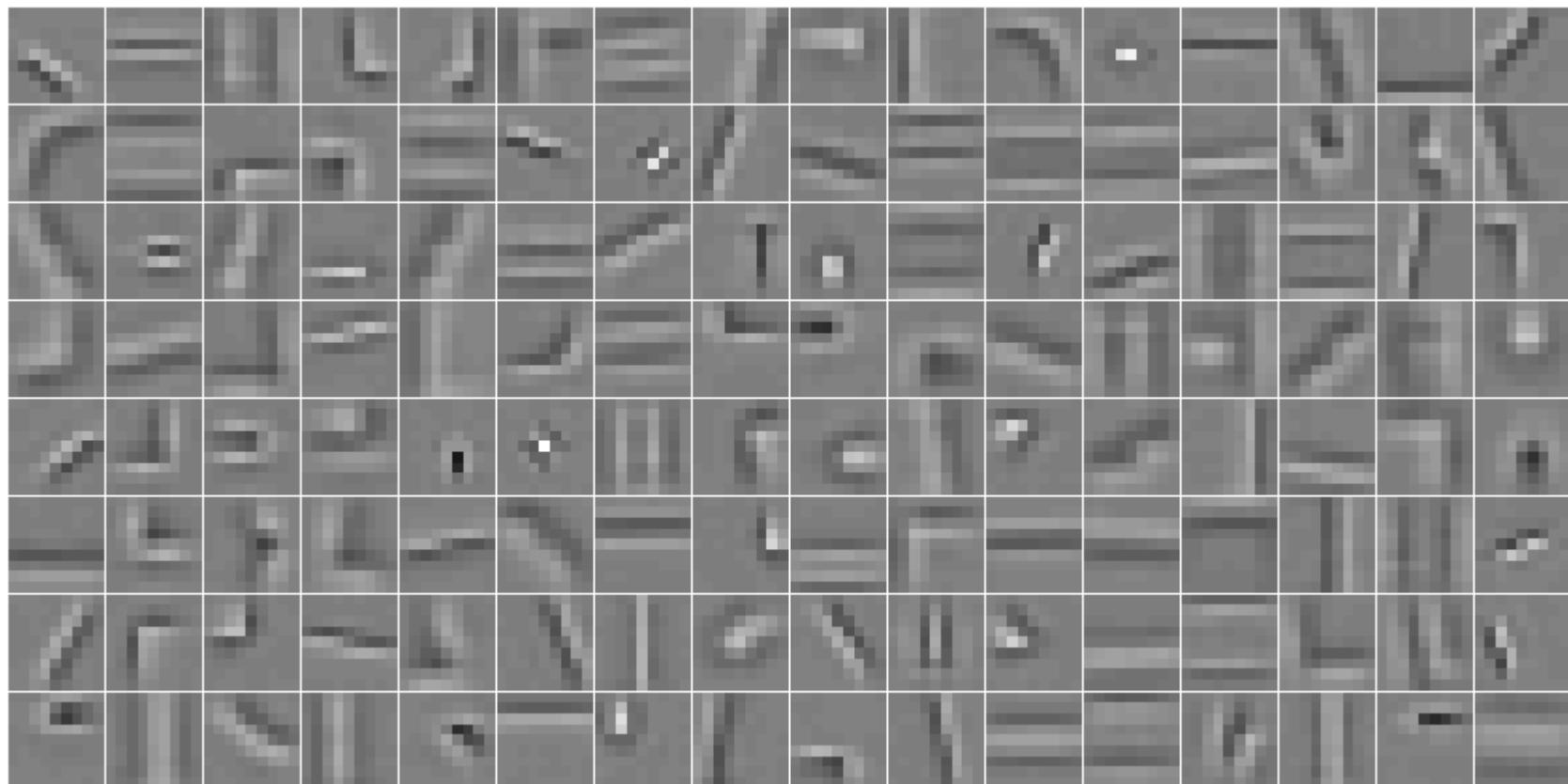
Y LeCun



Unsupervised pre-training with convolutional PSD

Y LeCun

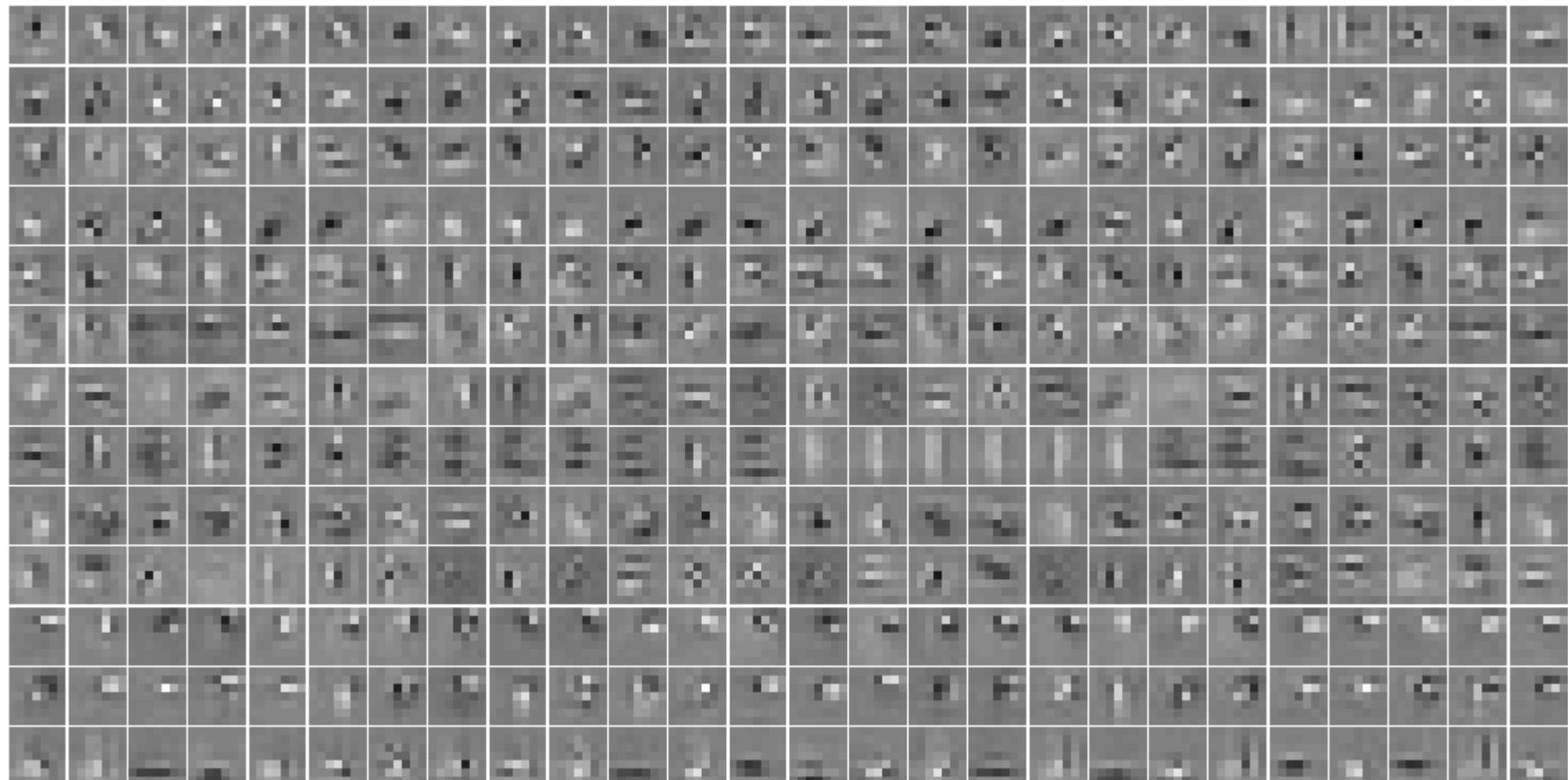
- ➊ 128 stage-1 filters on Y channel.
- ➋ Unsupervised training with convolutional predictive sparse decomposition



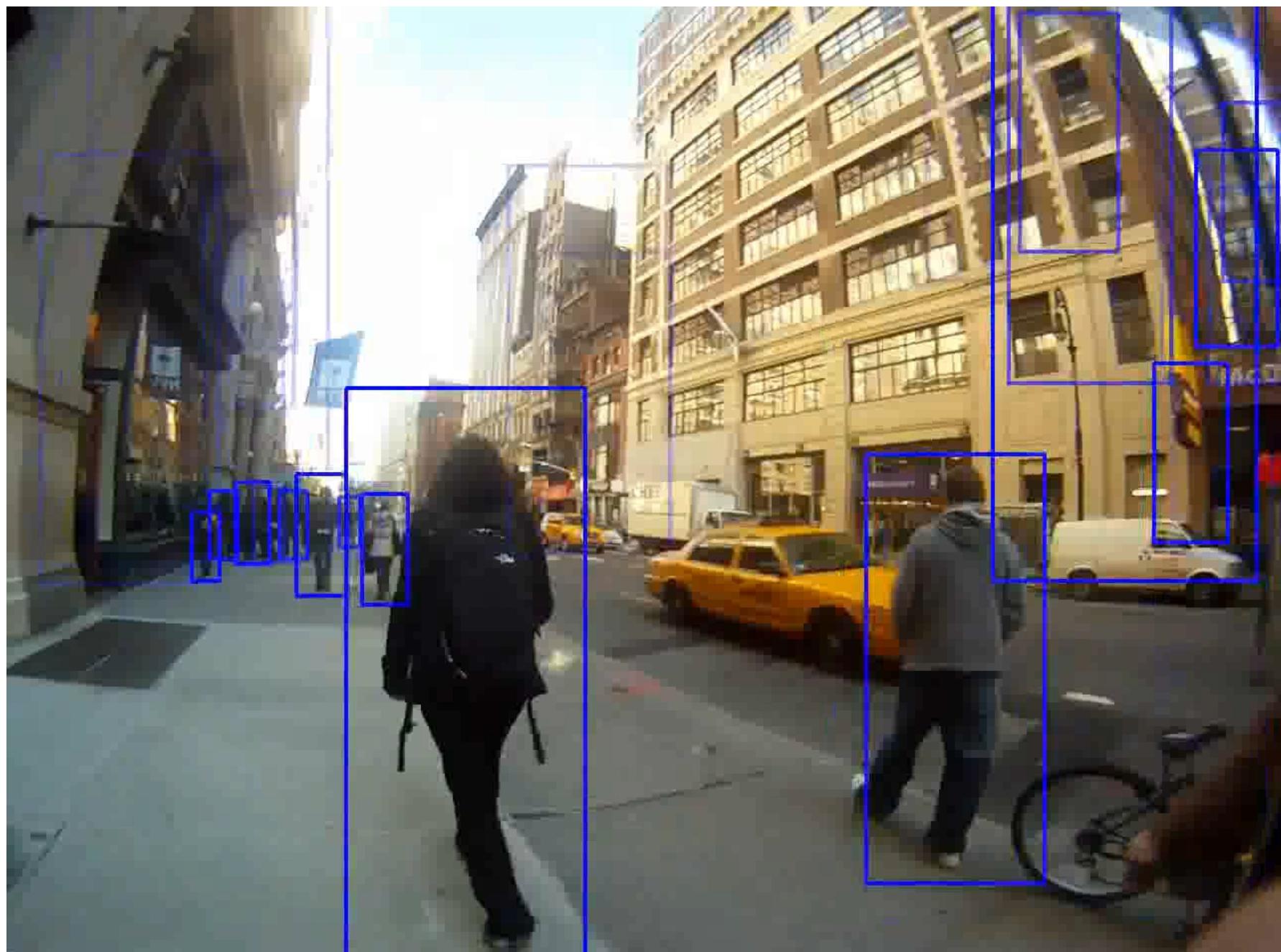
Unsupervised pre-training with convolutional PSD

Y LeCun

- Stage 2 filters.
- Unsupervised training with convolutional predictive sparse decomposition

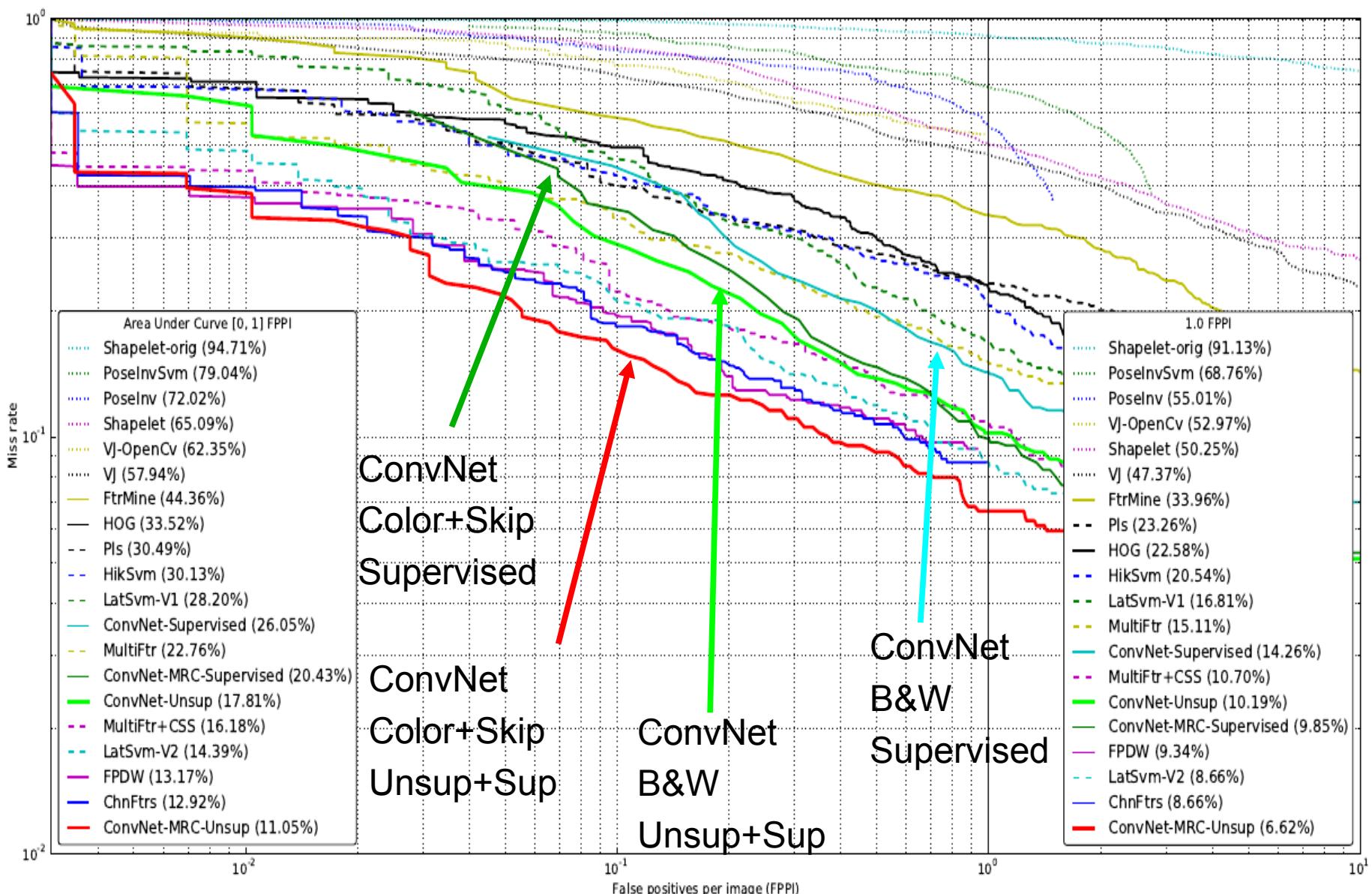






Pedestrian Detection: INRIA Dataset. Miss rate vs false positives

Y LeCun



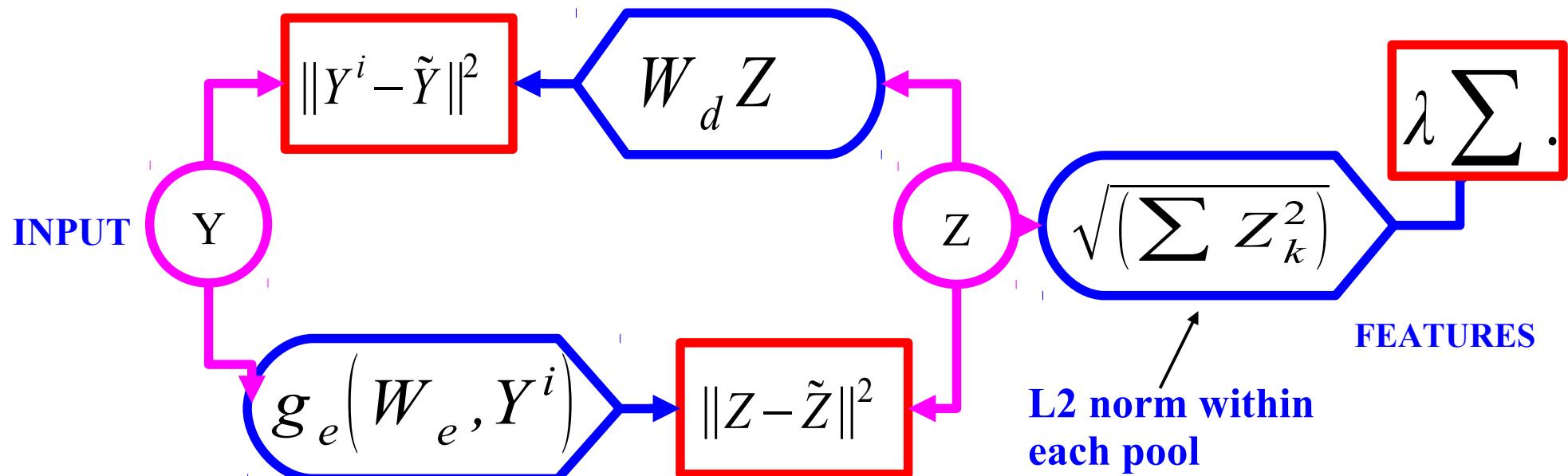
Unsupervised Learning: Invariant Features

Learning Invariant Features with L2 Group Sparsity

Y LeCun

- Unsupervised PSD ignores the spatial pooling step.
- Could we devise a similar method that learns the pooling layer as well?
- Idea [Hyvarinen & Hoyer 2001]: group sparsity on pools of features
 - ▶ Minimum number of pools must be non-zero
 - ▶ Number of features that are on within a pool doesn't matter
 - ▶ Pools tend to regroup similar features

$$E(Y, Z) = \|Y - W_d Z\|^2 + \|Z - g_e(W_e, Y)\|^2 + \sum_j \sqrt{\sum_{k \in P_j} Z_k^2}$$



Learning Invariant Features with L2 Group Sparsity

Y LeCun

Idea: features are pooled in group.

- Sparsity: sum over groups of L2 norm of activity in group.

[Hyvärinen Hoyer 2001]: "subspace ICA"

- decoder only, square

[Welling, Hinton, Osindero NIPS 2002]: pooled product of experts

- encoder only, overcomplete, log student-T penalty on L2 pooling

[Kavukcuoglu, Ranzato, Fergus LeCun, CVPR 2010]: Invariant PSD

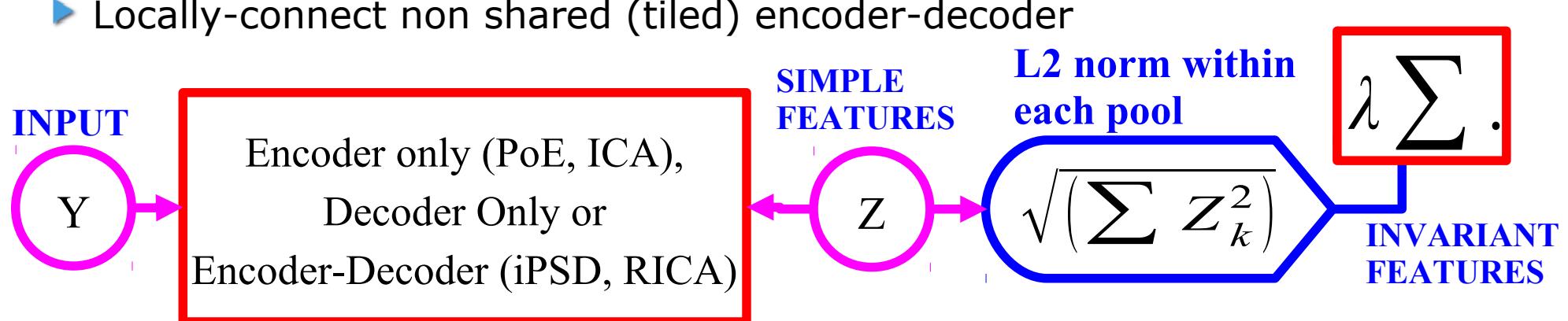
- encoder-decoder (like PSD), overcomplete, L2 pooling

[Le et al. NIPS 2011]: Reconstruction ICA

- Same as [Kavukcuoglu 2010] with linear encoder and tied decoder

[Gregor & LeCun arXiv:1006:0448, 2010] [Le et al. ICML 2012]

- Locally-connect non shared (tiled) encoder-decoder



Groups are local in a 2D Topographic Map

Y LeCun

- The filters arrange themselves spontaneously so that similar filters enter the same pool.
- The pooling units can be seen as complex cells
- Outputs of pooling units are invariant to local transformations of the input
 - ▶ For some it's translations, for others rotations, or other transformations.

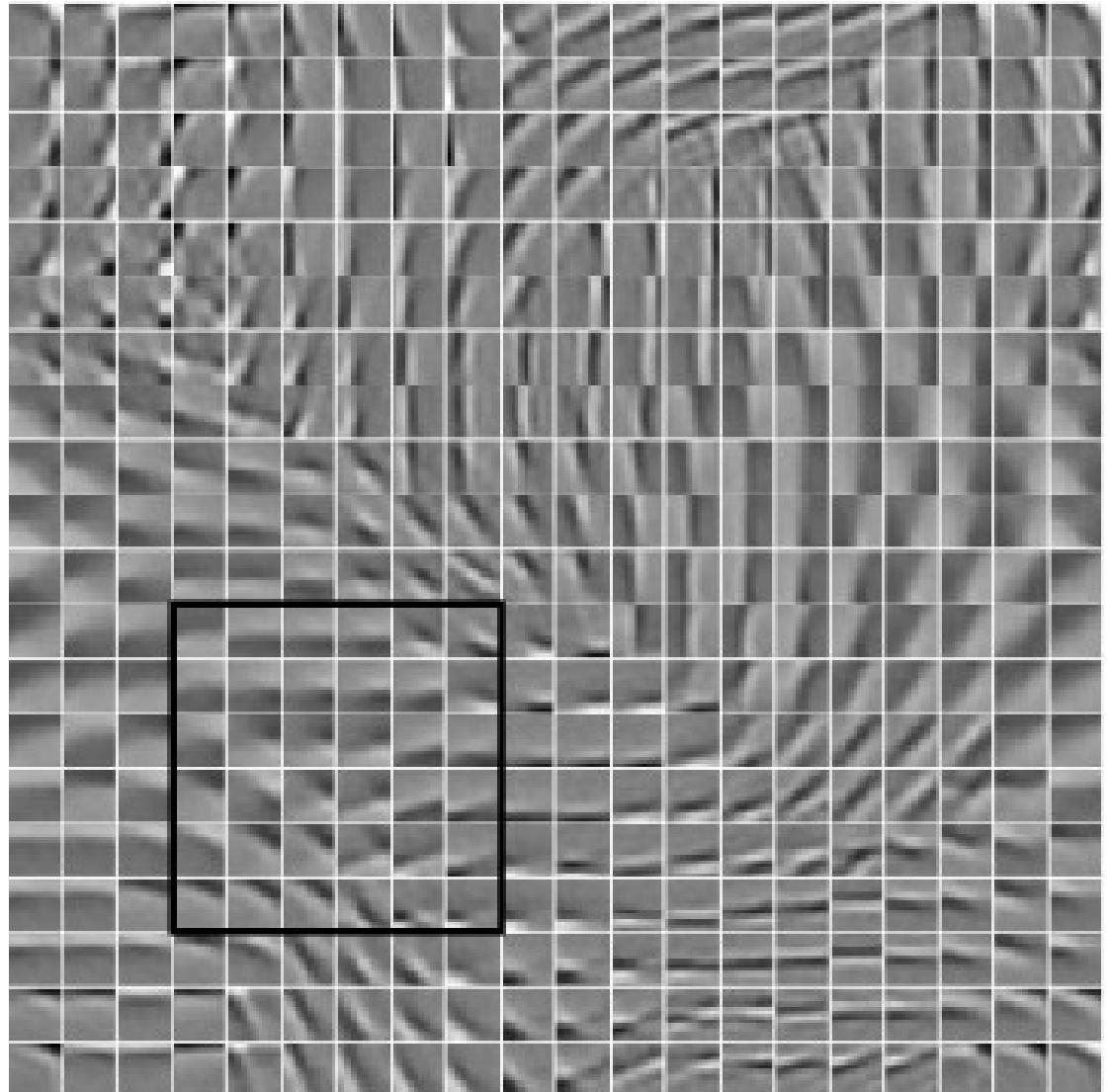


Image-level training, local filters but no weight sharing

Y LeCun

- Training on 115x115 images. Kernels are 15x15 (not shared across space!)

- ▶ [Gregor & LeCun 2010]
- ▶ Local receptive fields
- ▶ No shared weights
- ▶ 4x overcomplete
- ▶ L2 pooling
- ▶ Group sparsity over pools

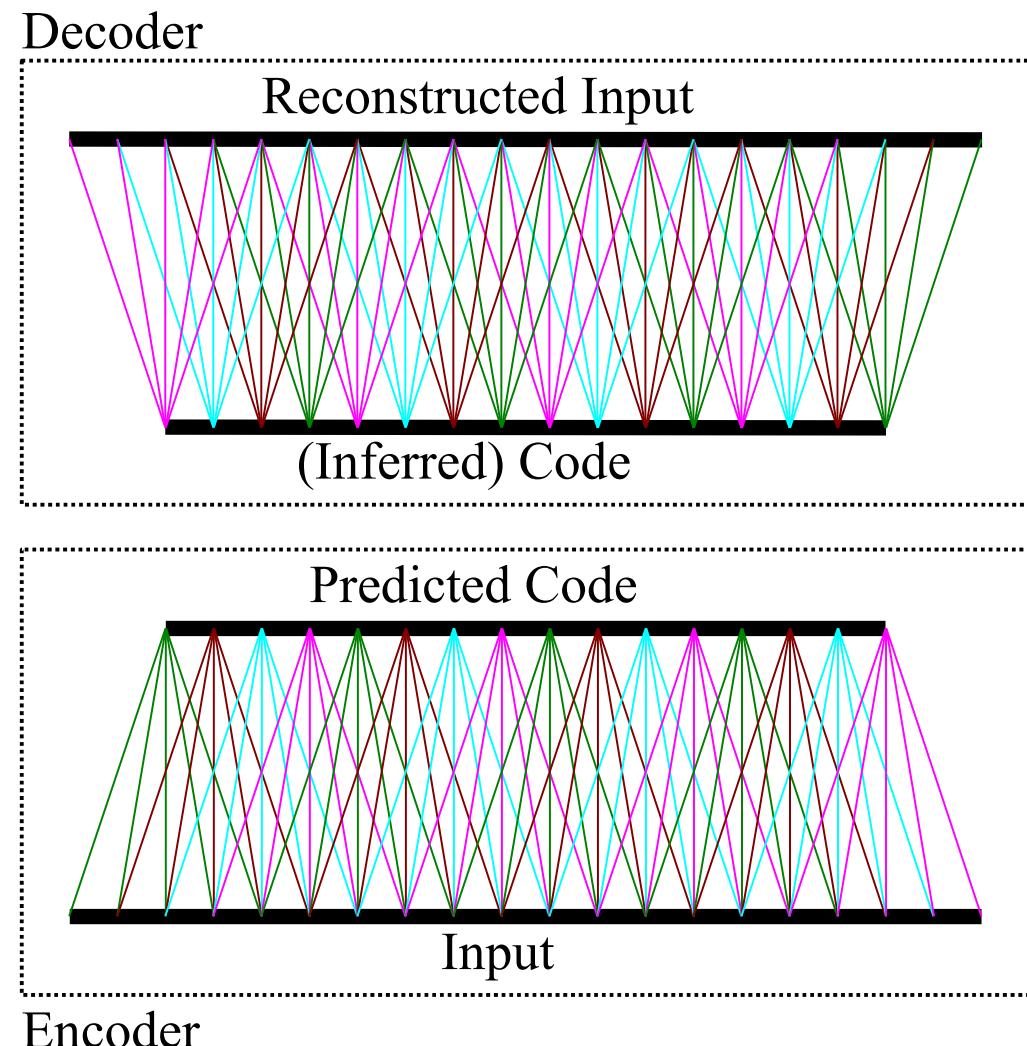
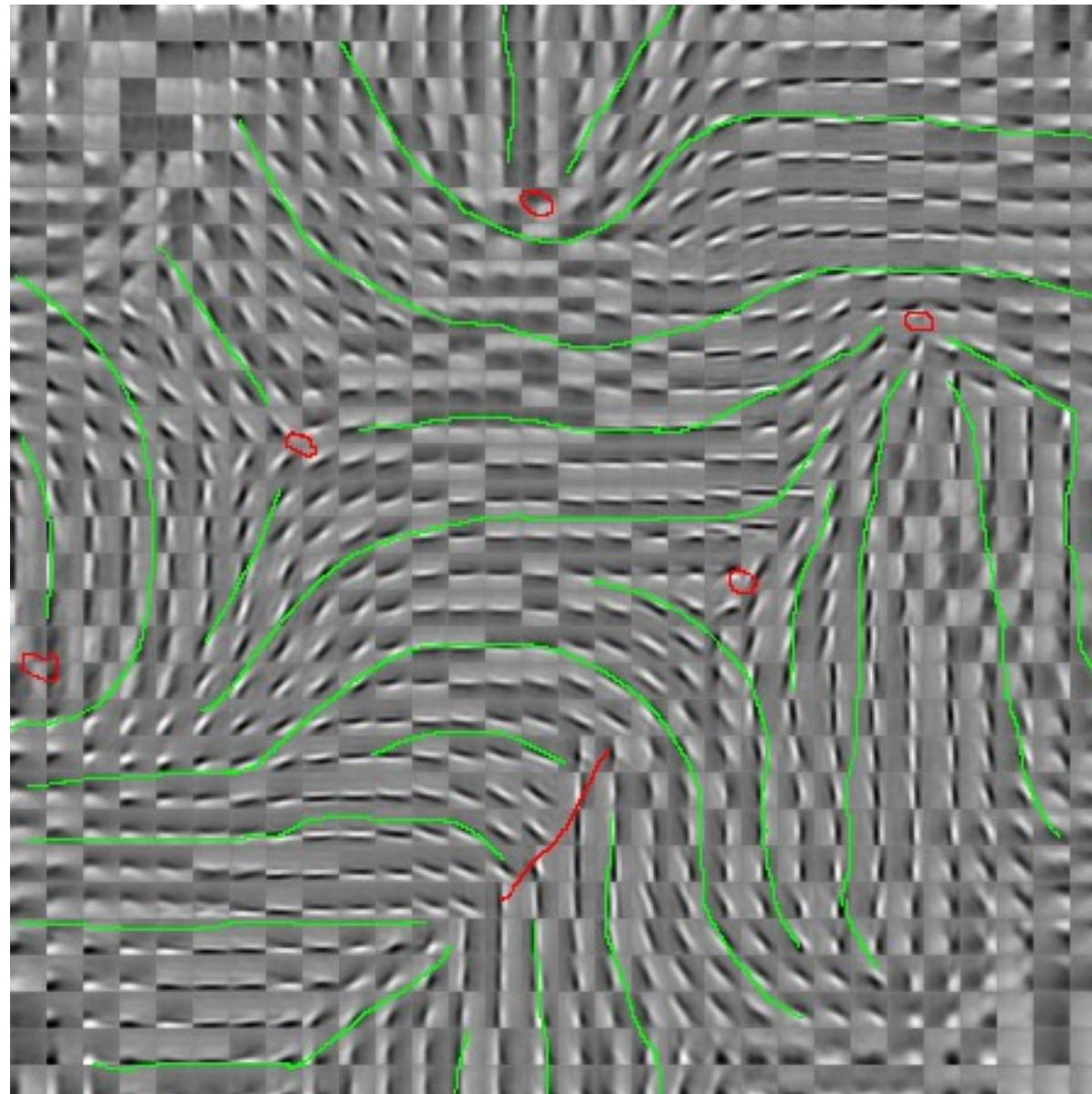


Image-level training, local filters but no weight sharing

Y LeCun

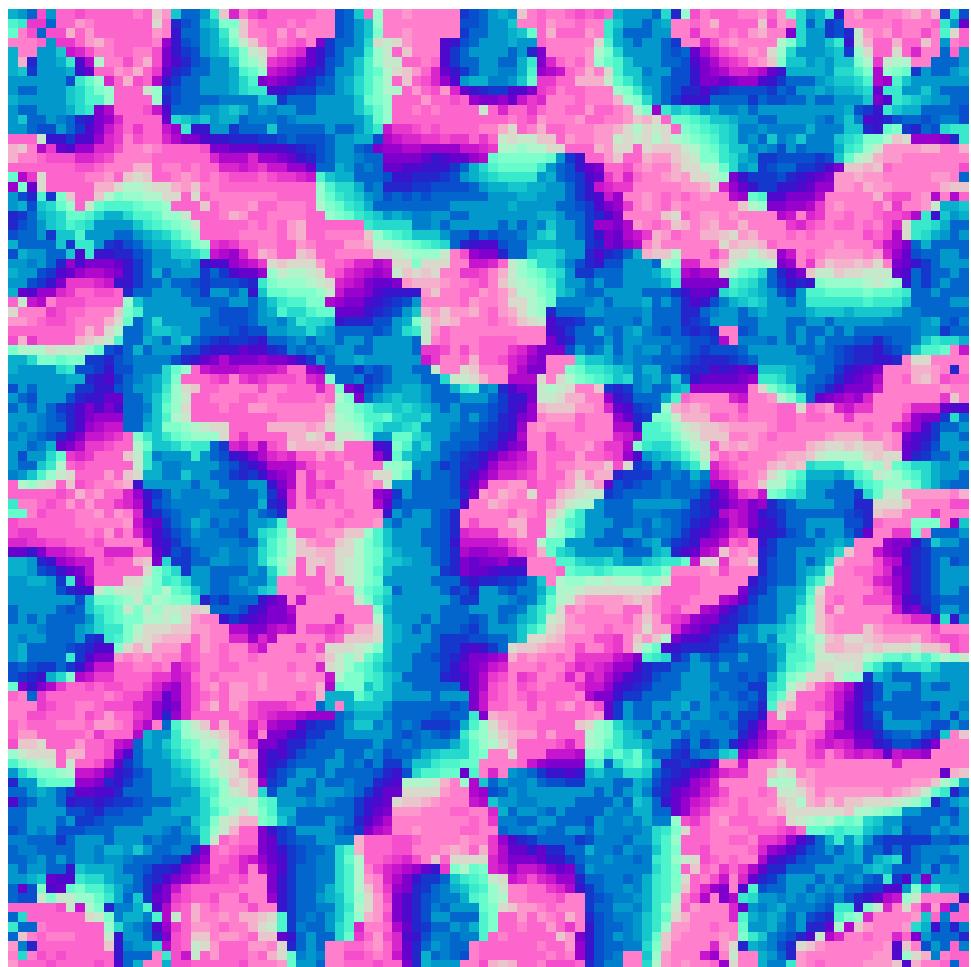
- Training on 115x115 images. Kernels are 15x15 (not shared across space!)



Topographic Maps

K Obermayer and GG Blasdel, Journal of Neuroscience, Vol 13, 4114-4129 (Monkey)

Y LeCun

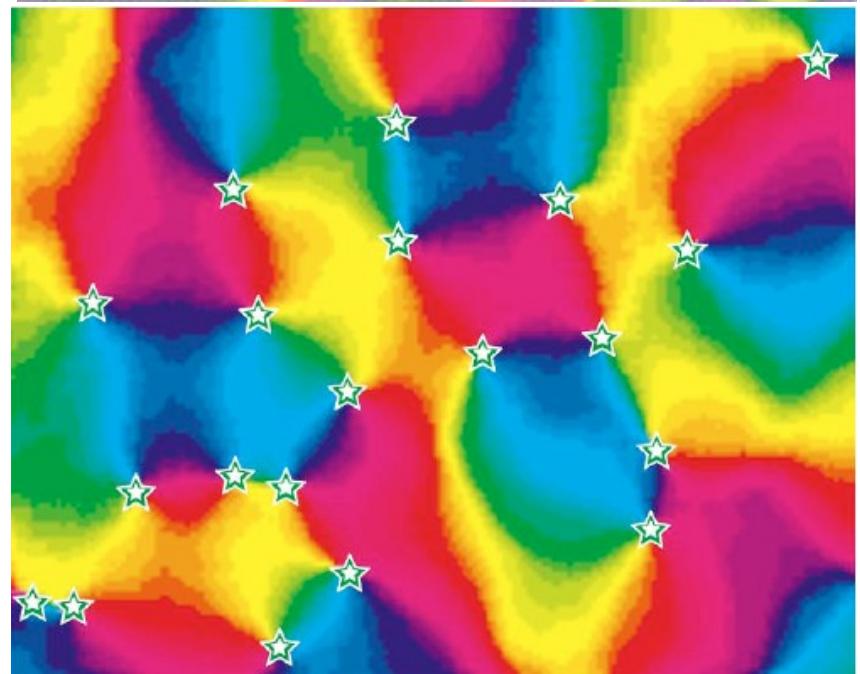
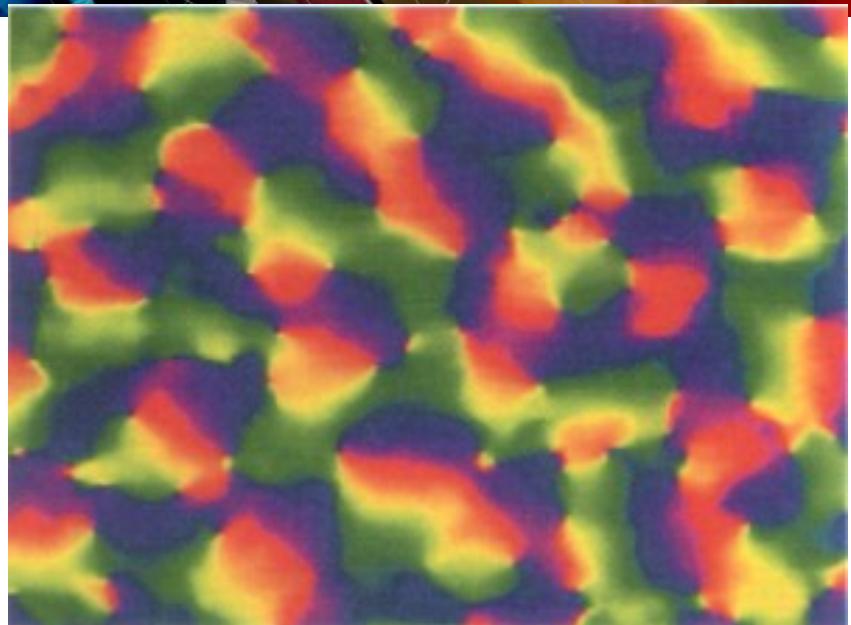


119x119 Image Input

100x100 Code

20x20 Receptive field size

$\sigma=5$

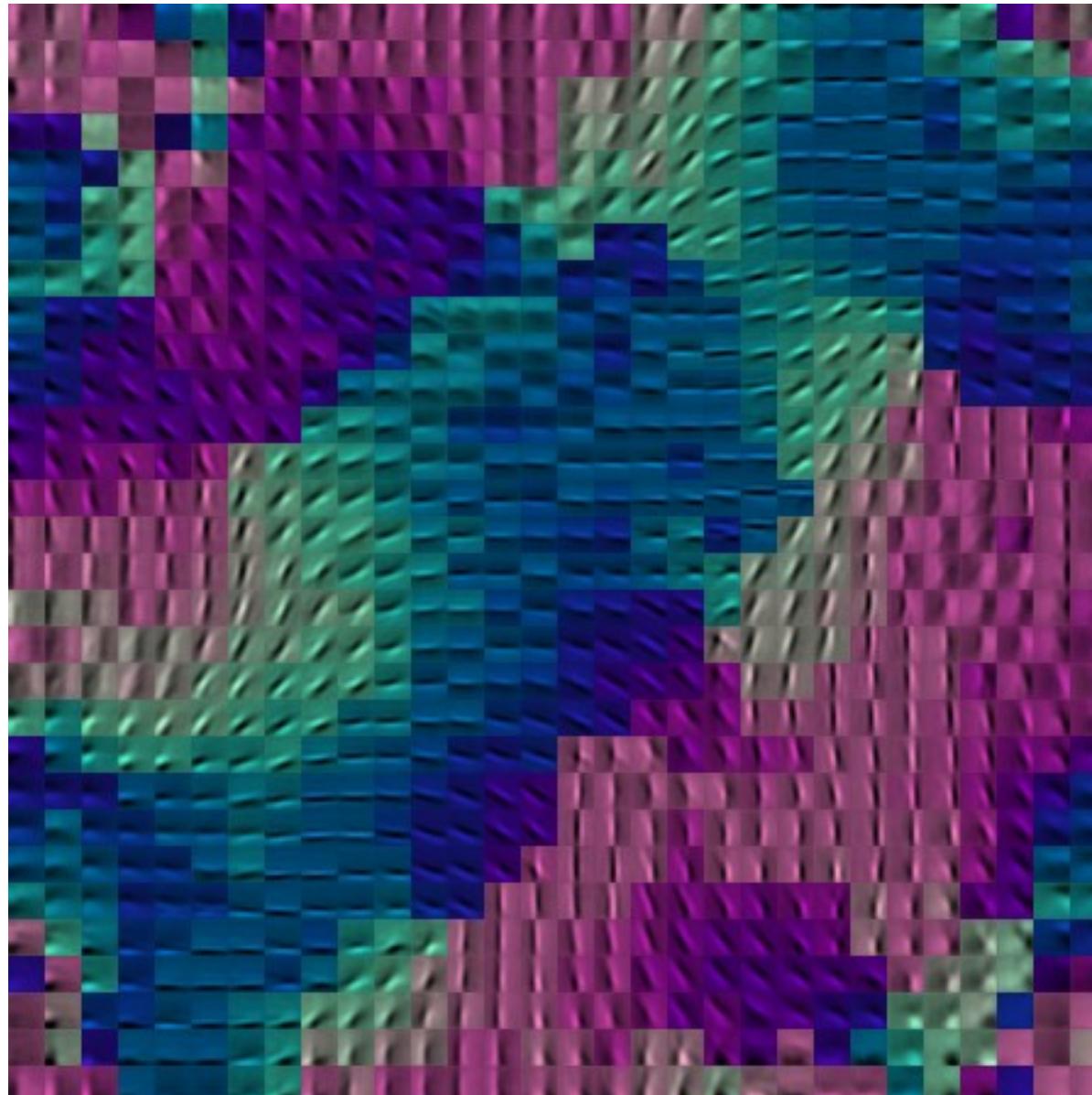


Michael C. Crair, et. al. The Journal of Neurophysiology Vol. 77 No. 6 June 1997, pp. 3381-3385 (Cat)

Image-level training, local filters but no weight sharing

Y LeCun

- Color indicates orientation (by fitting Gabors)



Future Challenges



Future Challenges

Y LeCun

- **Integrated feed-forward and feedback**

- ▶ Deep Boltzmann machine do this, but there are issues of scalability.

- **Integrating supervised and unsupervised learning in a single algorithm**

- ▶ Again, deep Boltzmann machines do this, but....

- **Integrating deep learning and structured prediction (“reasoning”)**

- ▶ This has been around since the 1990's but needs to be revived

- **Learning representations for complex reasoning**

- ▶ “recursive” networks that operate on vector space representations of knowledge [Pollack 90's] [Bottou 2010] [Socher, Manning, Ng 2011]

- **Representation learning in natural language processing**

- ▶ [Y. Bengio 01],[Collobert Weston 10], [Mnih Hinton 11] [Socher 12]

- **Better theoretical understanding of deep learning and convolutional nets**

- ▶ e.g. Stephane Mallat's “scattering transform”, work on the sparse representations from the applied math community....



Towards Practical AI: Challenges

Y LeCun

- Applying deep learning to NLP (requires “structured prediction”)
 - Video analysis/understanding (requires unsupervised learning)
 - High-performance/low power embedded systems for ConvNets (FPGA/ASIC?)
 - Very-large-scale deep learning (distributed optimization)
 - Integrating reasoning with DL (“energy-based models”, recursive neural nets)
-
- Then we can have
 - ▶ Automatically-created high-performance data analytics systems
 - ▶ Vector-space embedding of everything (language, users,...)
 - ▶ Multimedia content understanding, search and indexing
 - ▶ Multilingual speech dialog systems
 - ▶ Driver-less cars
 - ▶ Autonomous maintenance robots / personal care robots

The Future: Unification Feed-Forward & Feedback; Supervised & Unsupervised

Y LeCun

- Marrying feed-forward convolutional nets with generative “deconvolutional nets”
 - ▶ Deconvolutional networks
 - [Zeiler-Graham-Fergus ICCV 2011]

- Feed-forward/Feedback networks allow reconstruction, multimodal prediction, restoration, etc...
 - ▶ Deep Boltzmann machines can do this, but there are scalability issues with training

- Finding a single rule for supervised and unsupervised learning
 - ▶ Deep Boltzmann machines can also do this, but there are scalability issues with training

