The Unreasonable Effectiveness of Deep Learning

Yann LeCun
Facebook AI Research &
Center for Data Science, NYU
yann@cs.nyu.edu
http://yann.lecun.com

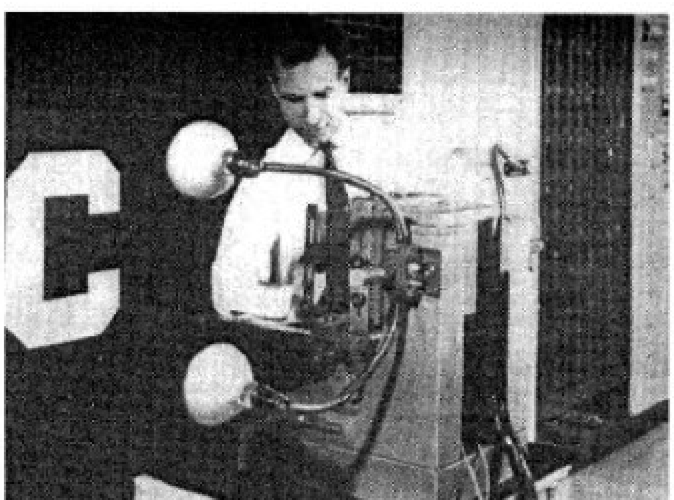
55 years of hand-crafted features

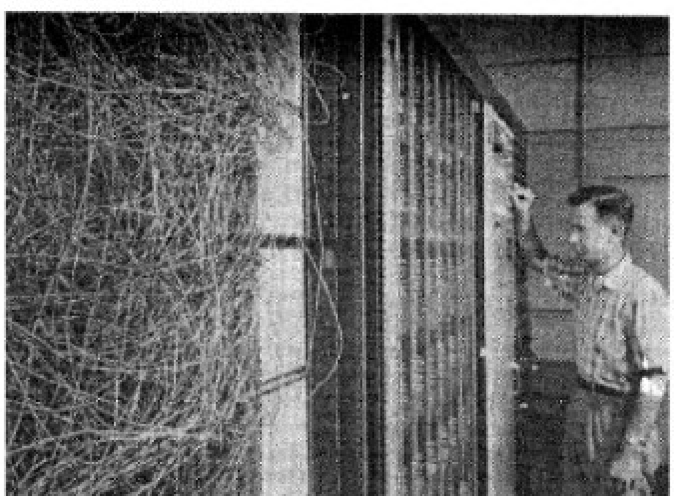
The traditional model of pattern recognition (since the late 50's)

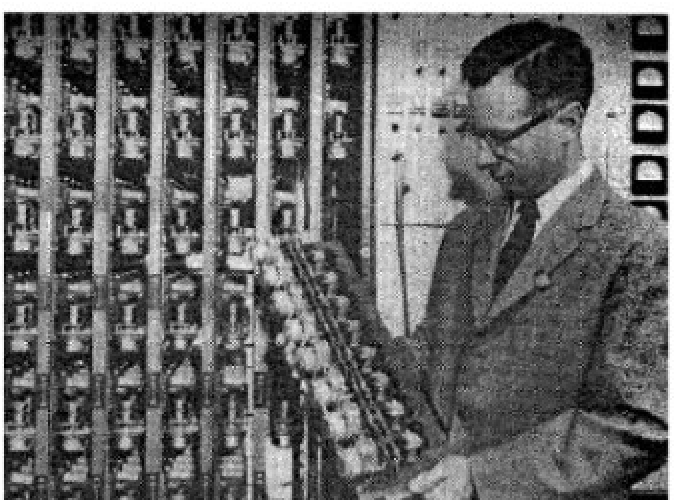
- Fixed/engineered features (or fixed kernel) + trainable classifier

hand-crafted Feature Extractor → "Simple" Trainable Classifier →

Perceptron

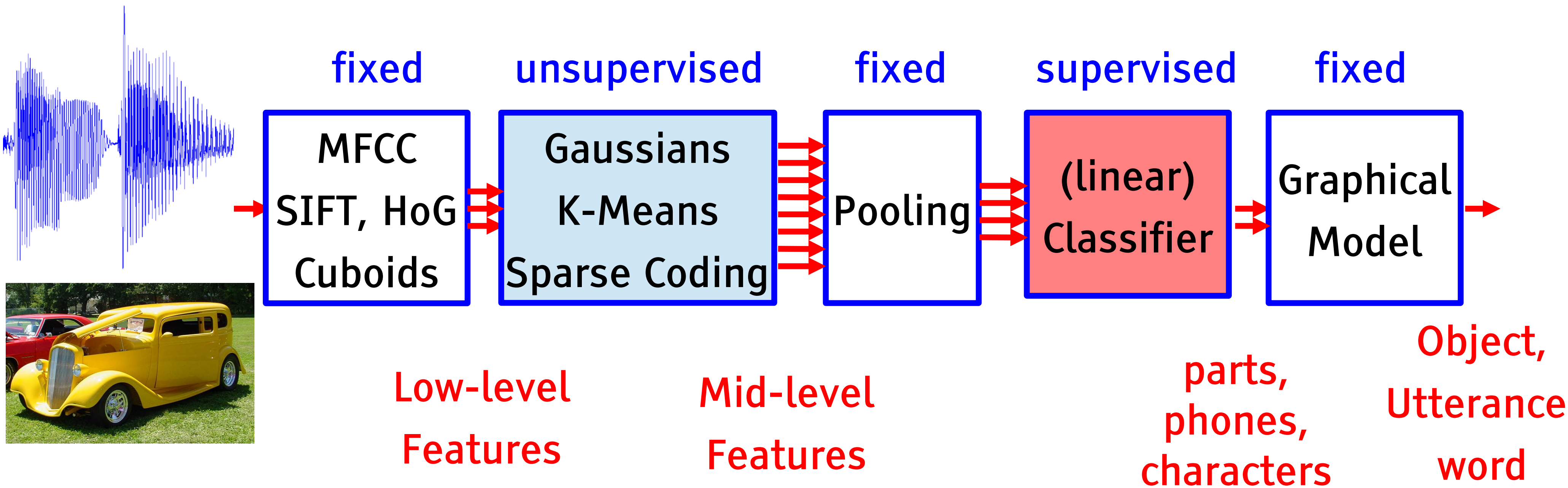






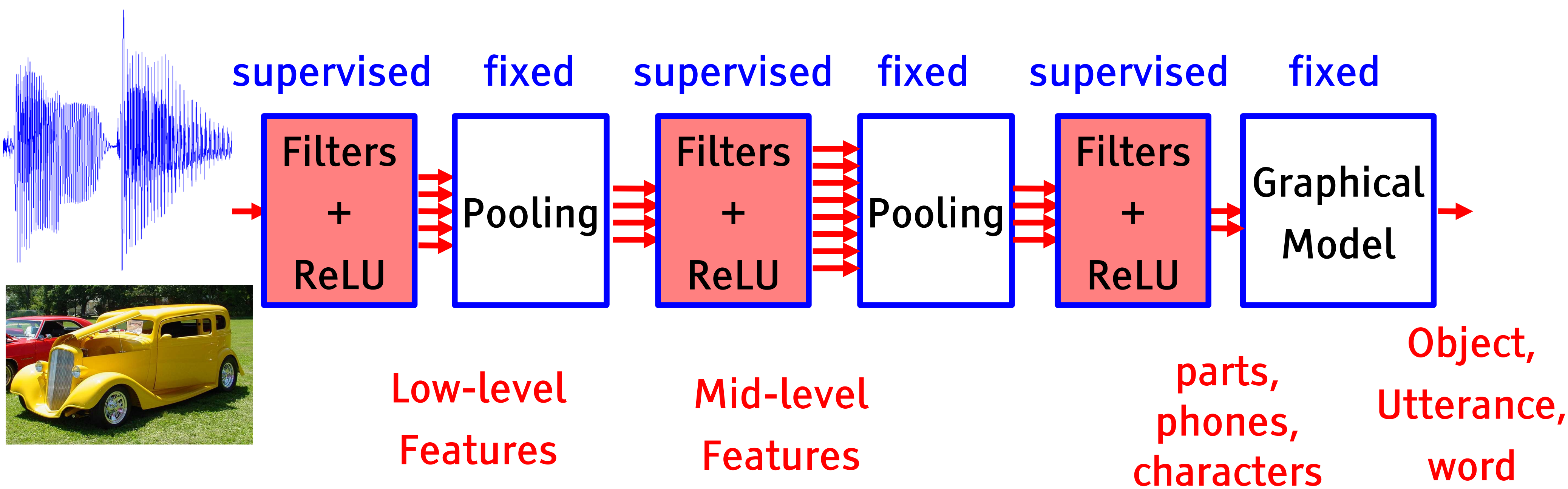
Architecture of "Classical" Recognition Systems

- "Classic" architecture for pattern recognition
  - Speech, and Object recognition (until recently)
  - Handwriting recognition (long ago)
  - Graphical model has latent variables (locations of parts)



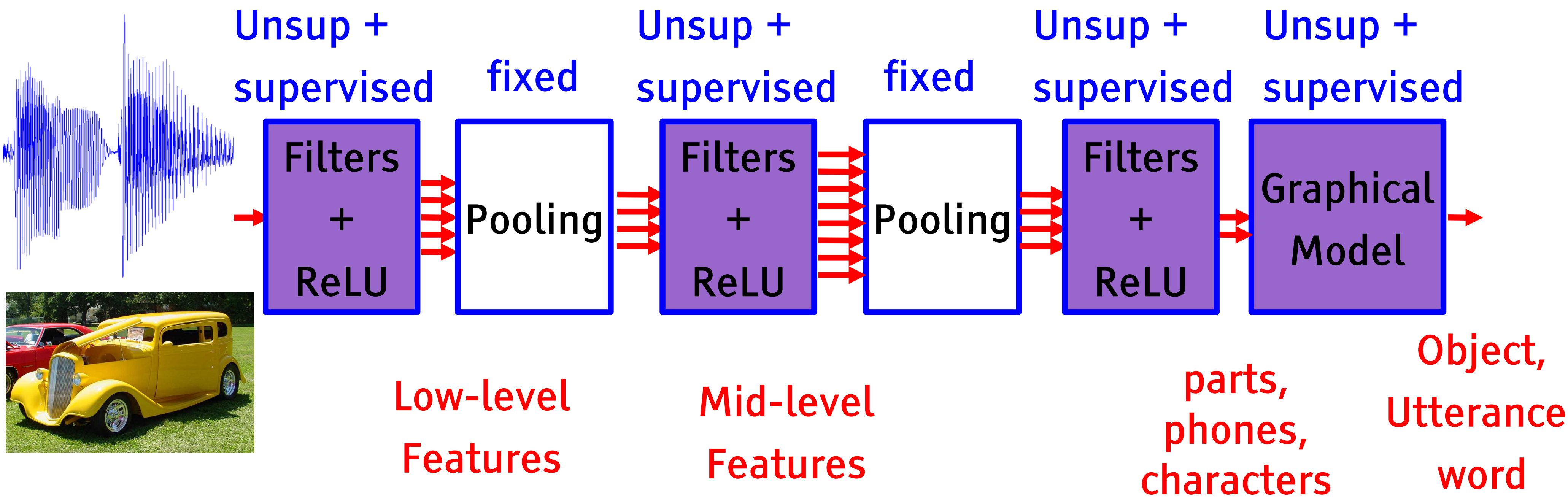
Architecture of Deep Learning-Based Recognition Systems

- **"Deep" architecture for pattern recognition**
  - Speech, and Object recognition (recently)
  - Handwriting recognition (since the 1990s)
  - Convolutional Net with optional Graphical Model on top
  - Trained purely supervised
  - Graphical model has latent variables (locations of parts)



Future Systems

- **Globally-trained deep architecture**
  - Handwriting recognition (since the mid 1990s)
  - All the modules are trained with a combination of unsupervised and supervised learning
  - **End-to-end training == deep structured prediction**

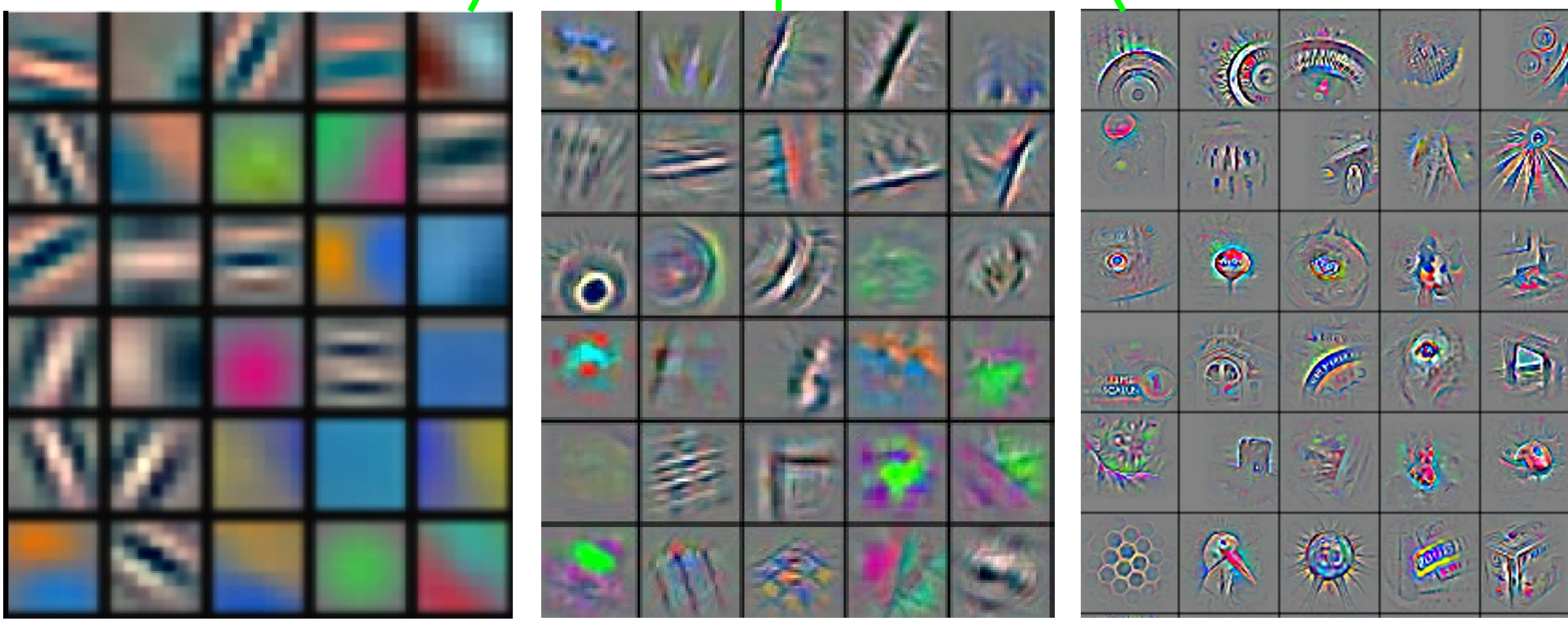


Deep Learning = Learning Hierarchical Representations

Y LeCun

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

Low-Level Feature → Mid-Level Feature → High-Level Feature → Trainable Classifier →

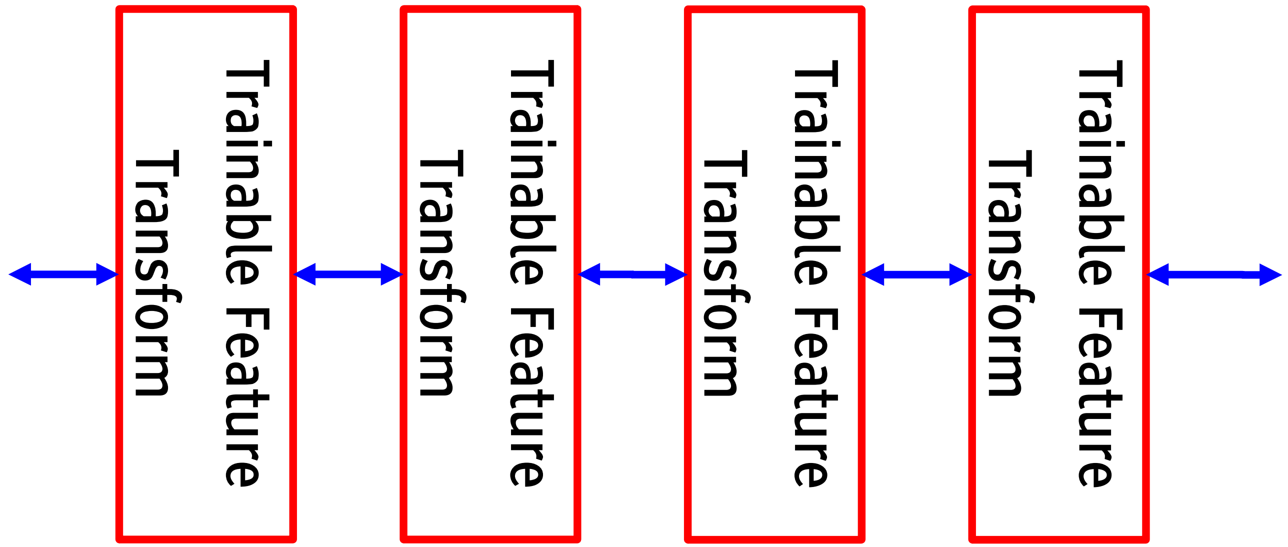


Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Trainable Feature Hierarchy

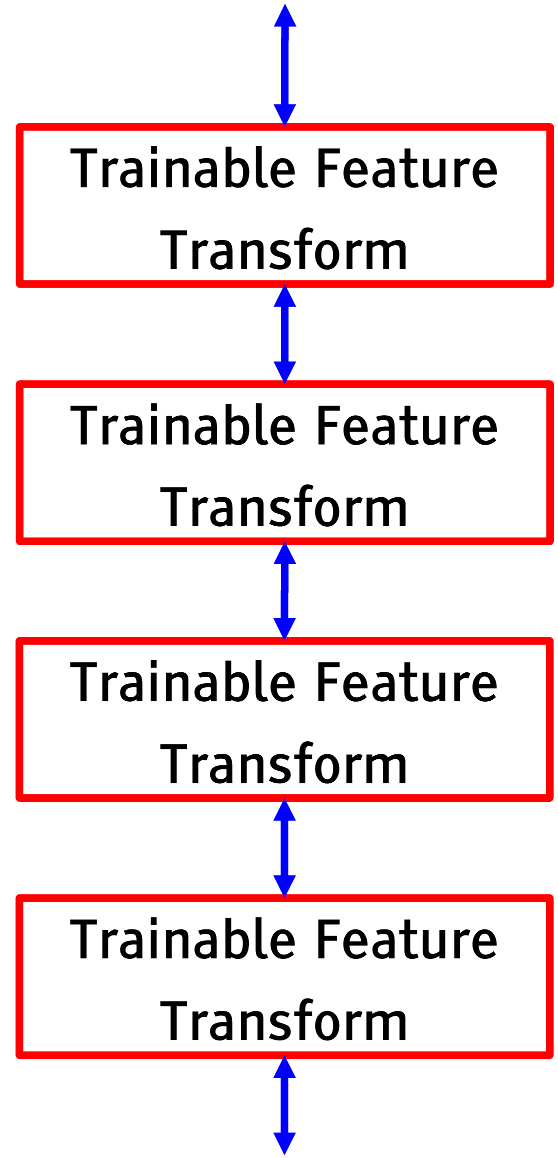
Y LeCun

- 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...

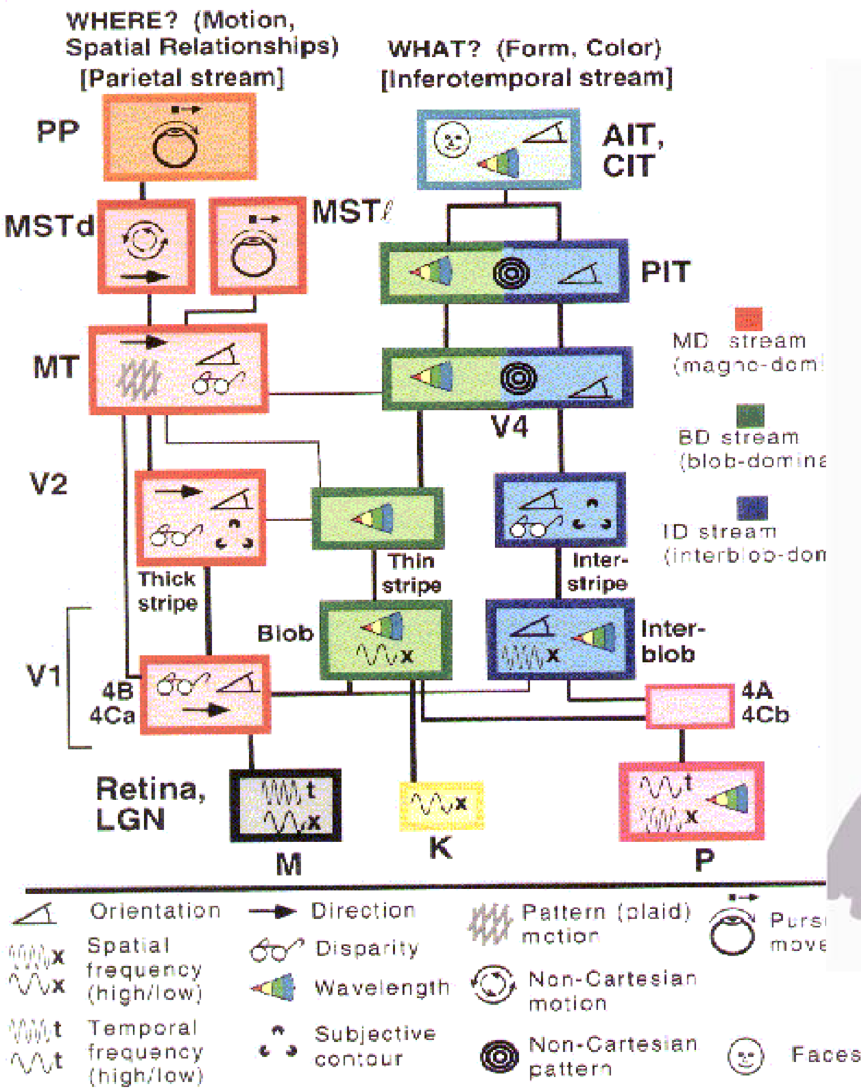
- **How do we learn representations of the perceptual world?**
  - How can a perceptual system build itself by looking at the world?
  - How much prior structure is necessary
- **ML/AI: how do we learn features or feature hierarchies?**
  - What is the fundamental principle? What is the learning algorithm? 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?**

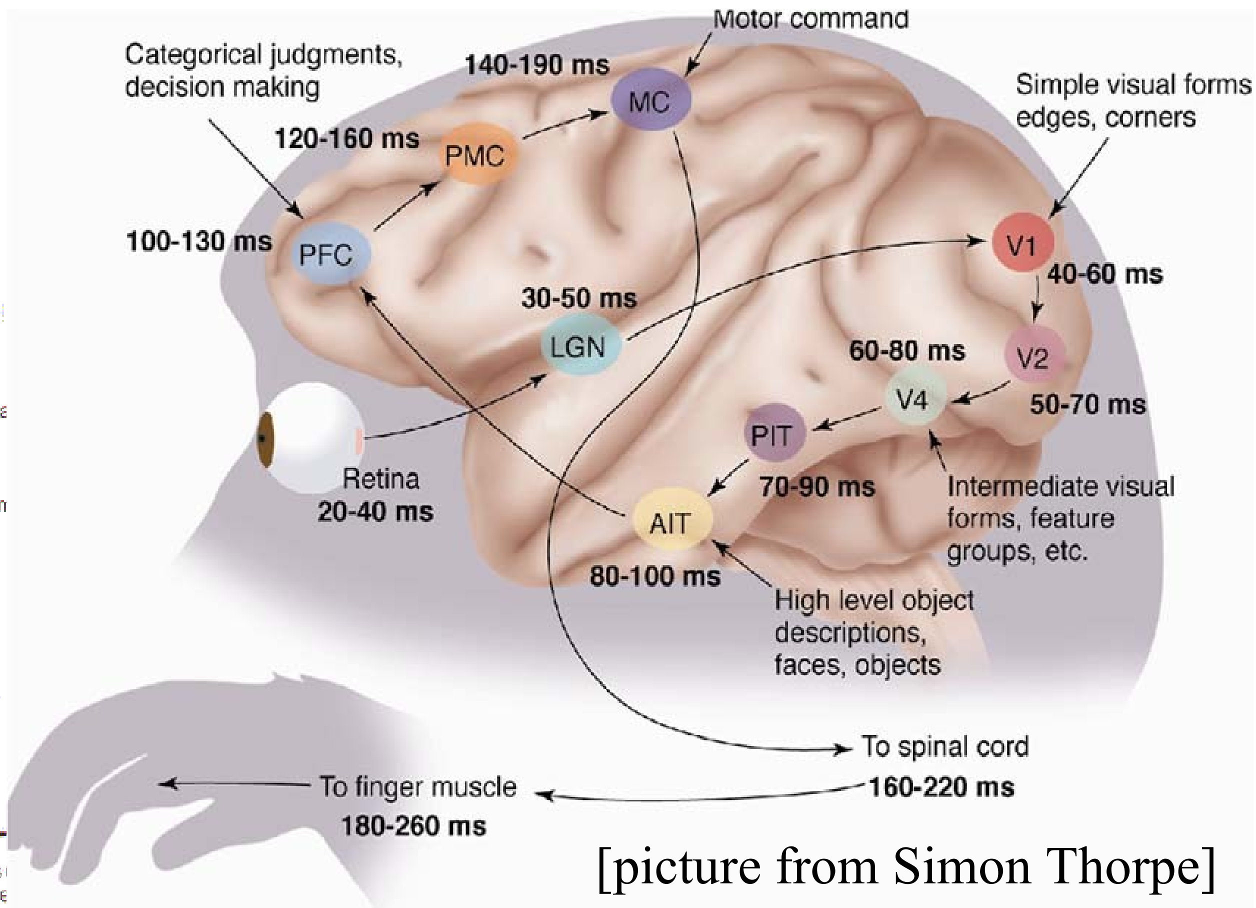


- **Deep Learning addresses the problem of learning hierarchical representations with a single algorithm**
  - or perhaps with a few algorithms

The Mammalian Visual Cortex is Hierarchical

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

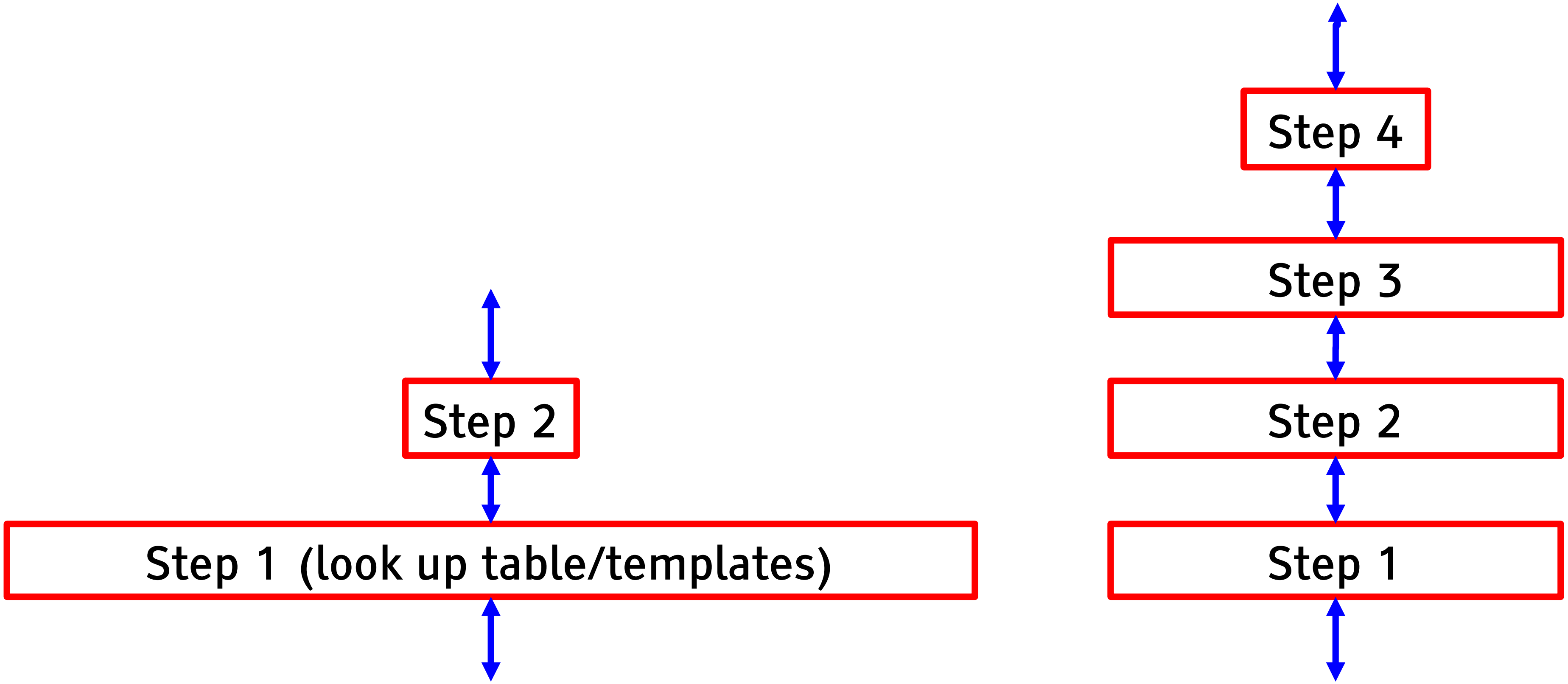




[Gallant & Van Essen]

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

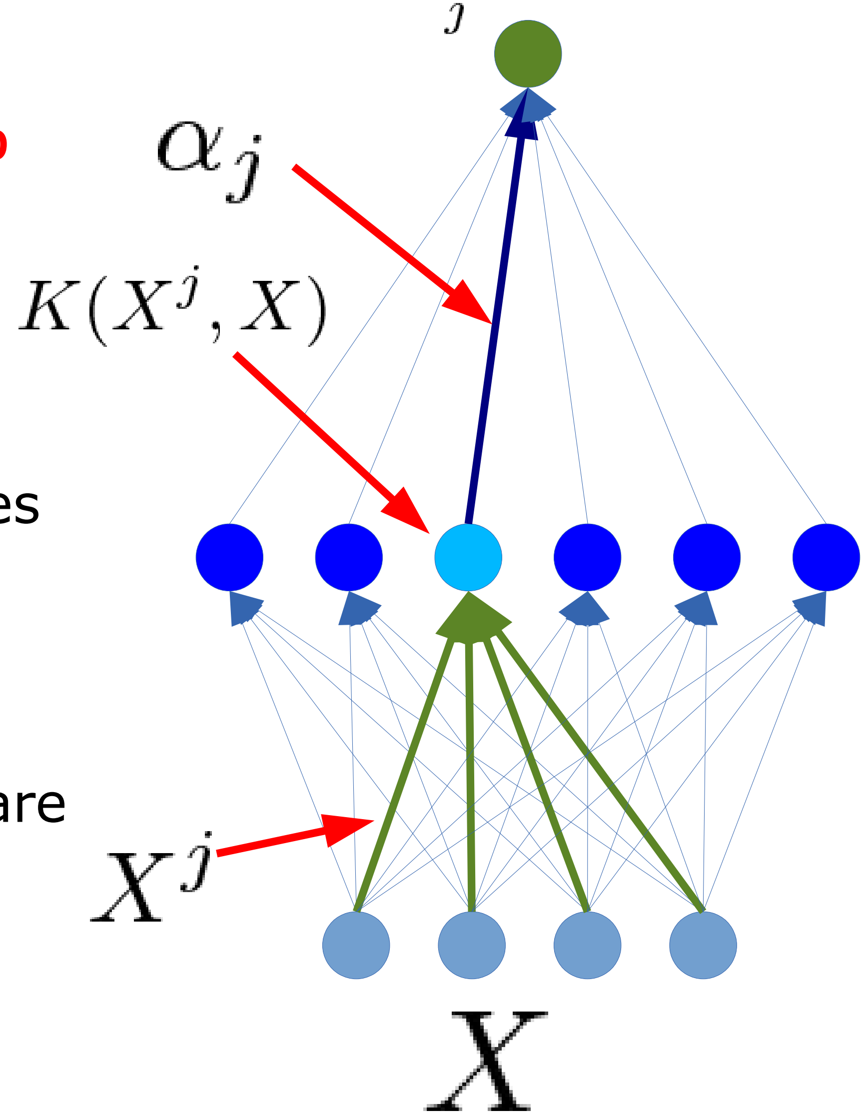
- "shallow & wide" vs "deep and narrow" == "more memory" vs "more time"
  - Look-up table vs algorithm
  - Few functions can be computed in two steps without an exponentially large lookup table
  - Using more than 2 steps can reduce the "memory" by an exponential factor.



Which Models are Deep?

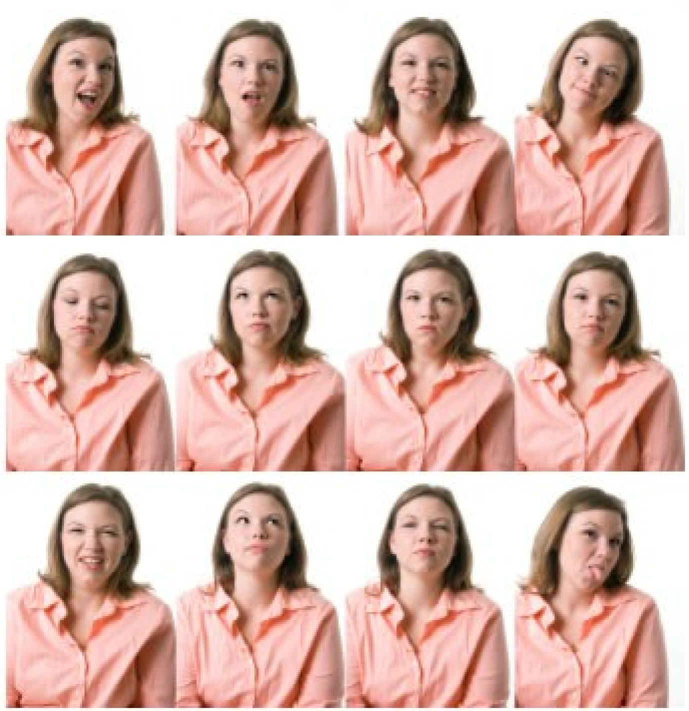
- **2-layer models are not deep (even if you train the first layer)**
  - Because there is no feature hierarchy
- **Neural nets with 1 hidden layer are not deep**
- **SVMs and Kernel methods are not deep**
  - Layer1: kernels; layer2: linear
  - The first layer is "trained" in with the simplest unsupervised 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

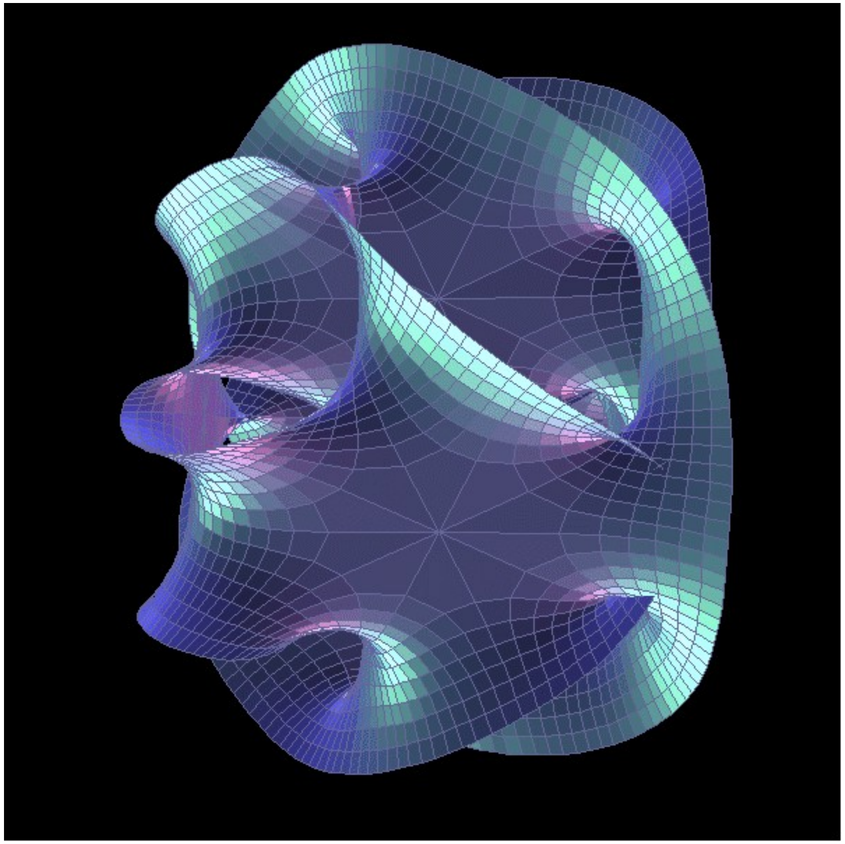
$$G(X, \alpha) = \sum_j \alpha_j K(X^j, X)$$



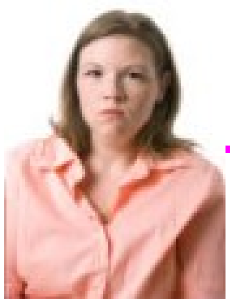
What Are
Good Feature?

- **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





- **Example: all face images of a person**
  - 1000x1000 pixels = 1,000,000 dimensions
  - But the face has 3 cartesian coordinates and 3 Euler angles
  - And humans have less than about 50 muscles in the face
  - Hence the manifold of face images for a person has <56 dimensions
- **The perfect representations of a face image:**
  - Its coordinates on the face manifold
  - Its coordinates away from the manifold
- **We do not have good and general methods to learn functions that turns an image into this kind of representation**



Ideal Feature Extractor

$$\begin{bmatrix} 1.2 \\ -3 \\ 0.2 \\ -2\ldots \end{bmatrix}$$

Face/not face

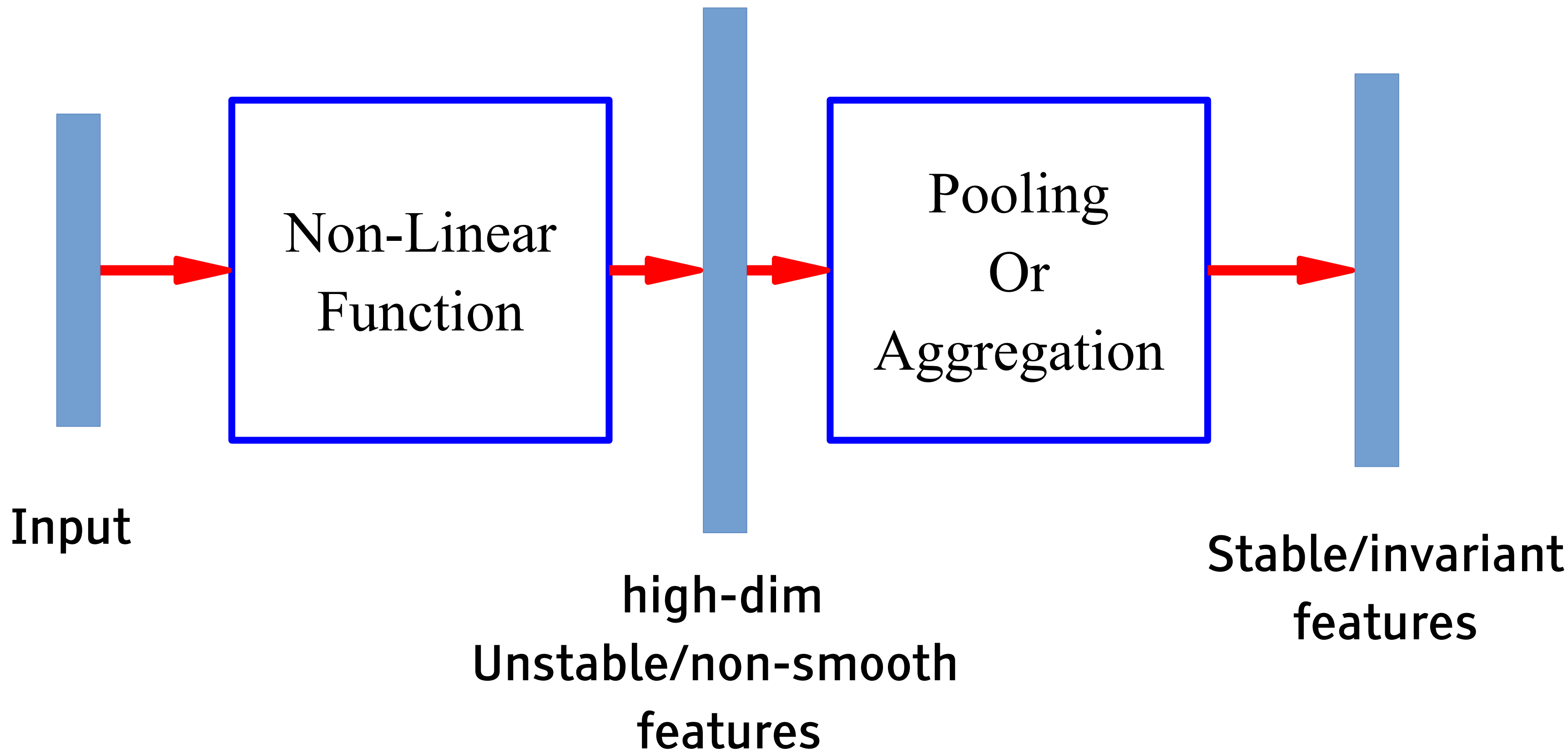
Pose

Lighting

Expression

Basic Idea for Invariant Feature Learning

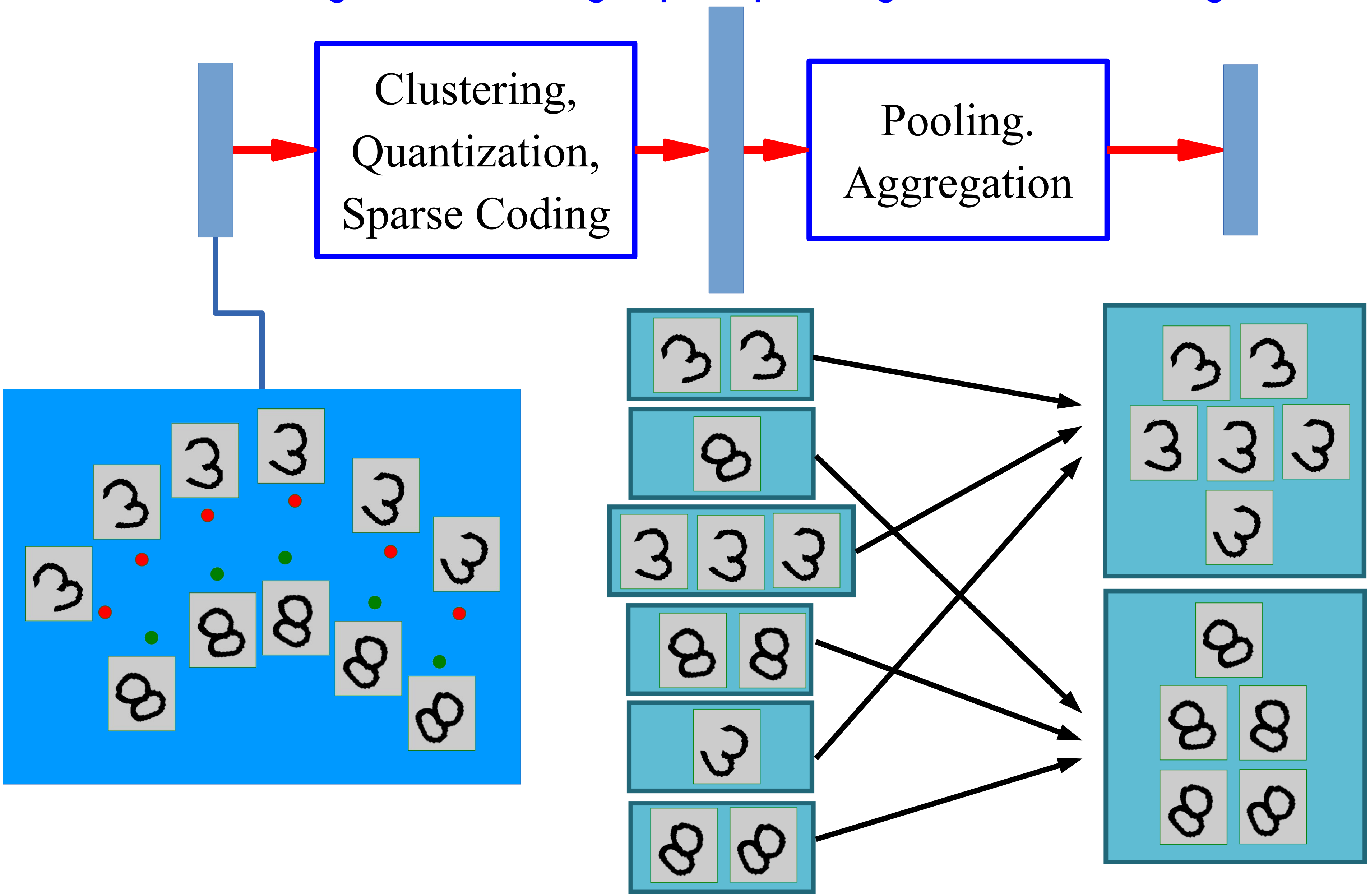
- Embed the input non-linearly into a high(er) dimensional space
  - In the new space, things that were non separable may become separable
- Pool regions of the new space together
  - Bringing together things that are semantically similar. Like pooling.



Sparse Non-Linear Expansion → Pooling

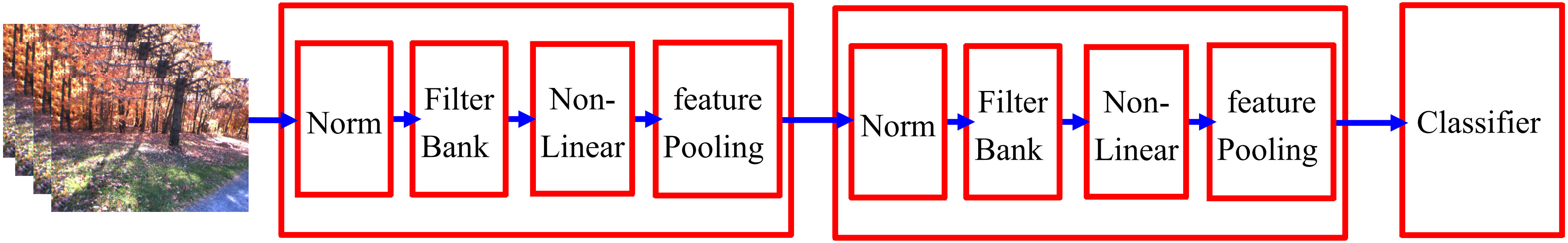
Y LeCun

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



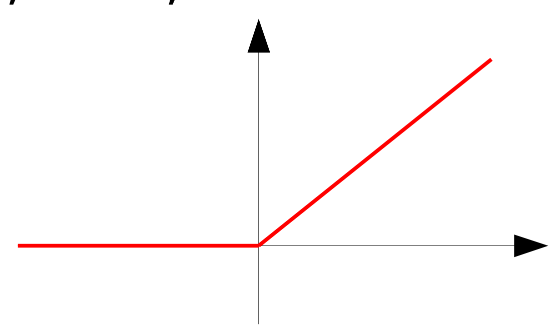
Overall Architecture: multiple stages of Normalization → Filter Bank → Non-Linearity → Pooling

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,..

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



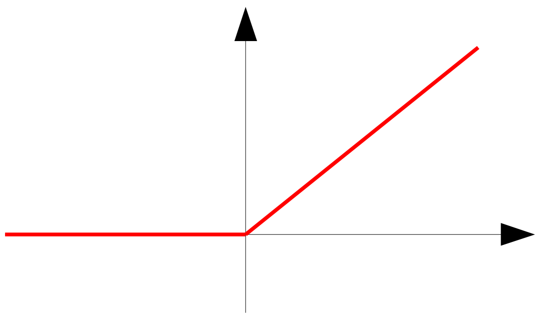
- **Pooling:** aggregation over space or feature type
  - Max, Lp norm, log prob.

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

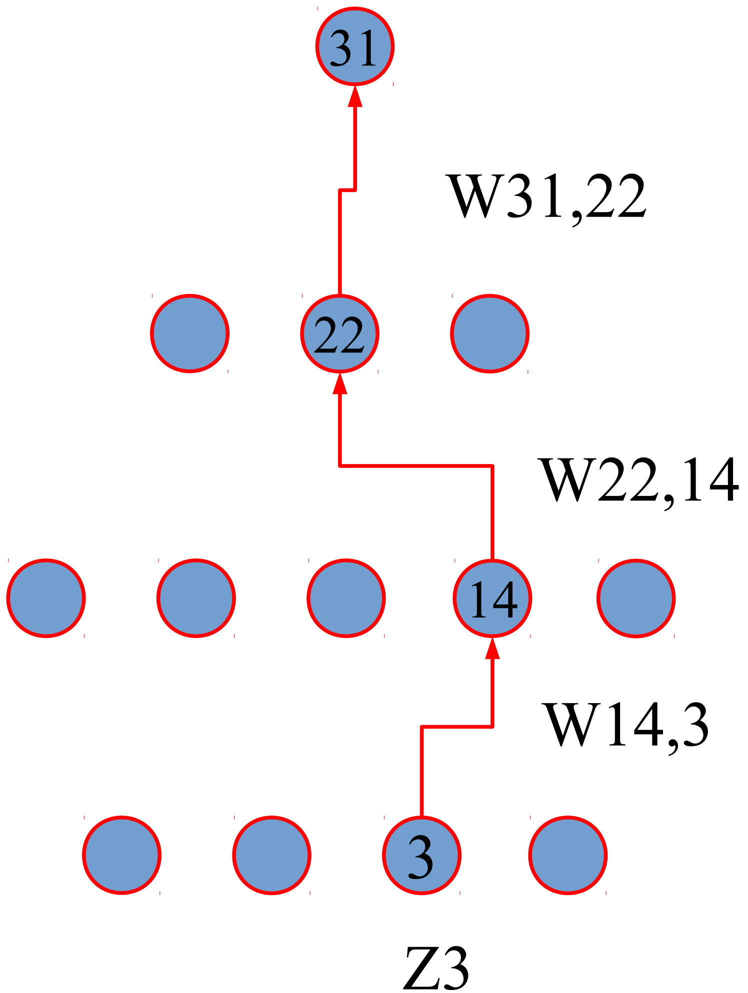
Deep Nets with ReLUs and Max Pooling

- Stack of linear transforms interspersed with Max operators
- Point-wise ReLUs:

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

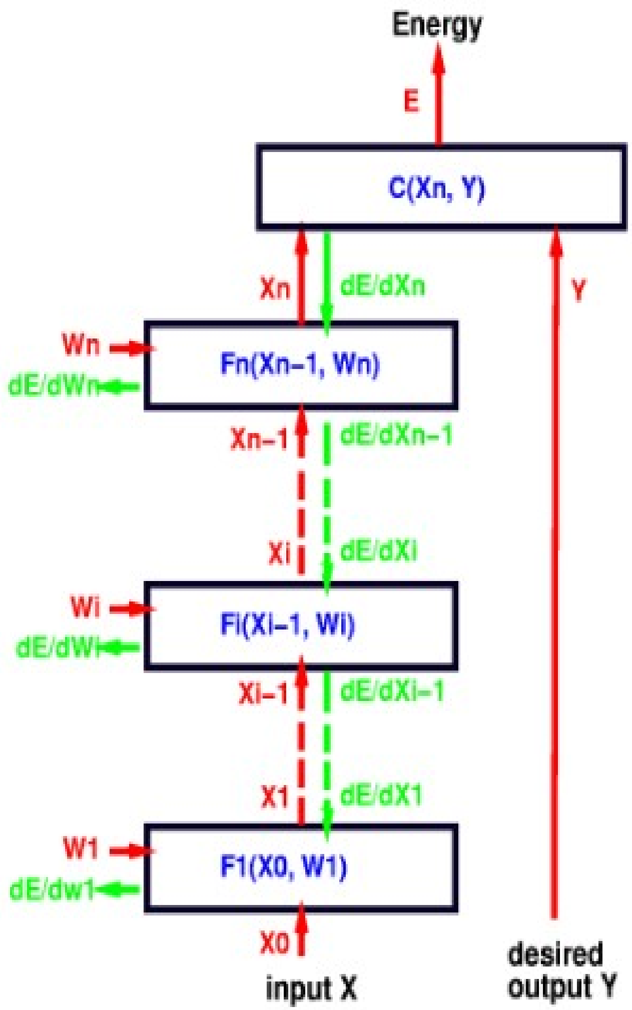


- Max Pooling
  - "switches" from one layer to the next



Supervised Training: Stochastic (Sub)Gradient Optimization

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 $i \in [1, n]$.

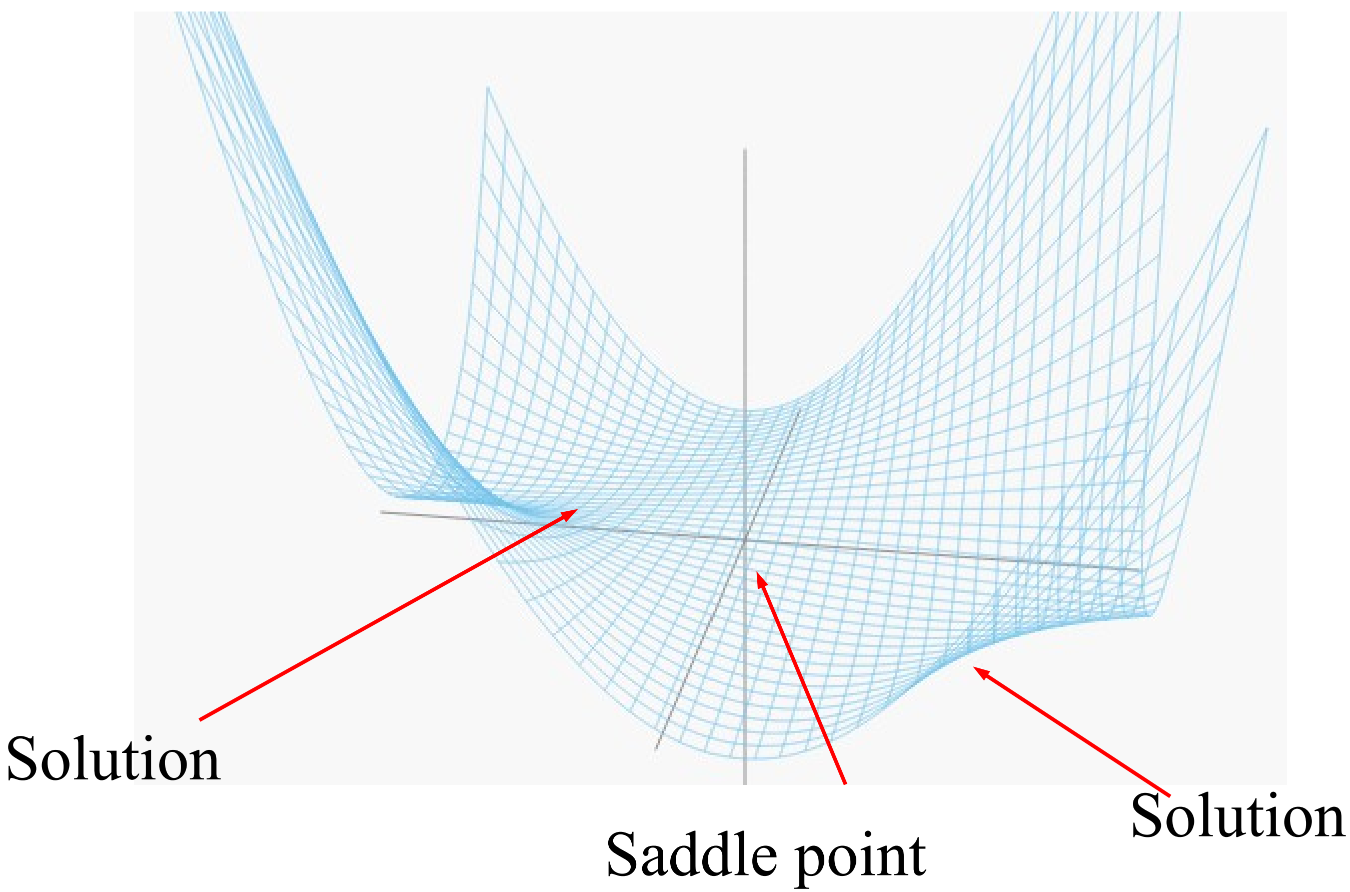
Loss Function for a simple network

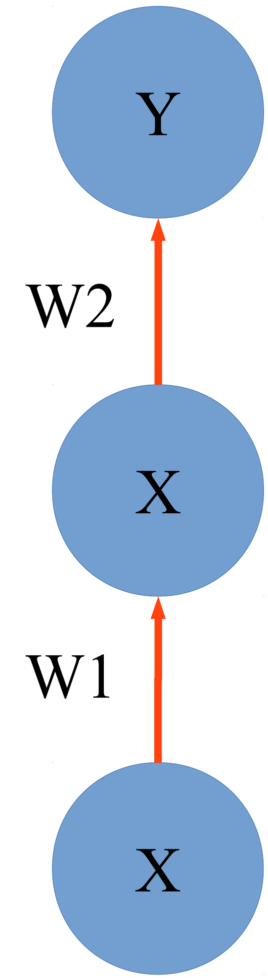
1-1-1 network

- Y = W1*W2*X

trained to compute the identity function with quadratic loss

- Single sample X=1, Y=1 L(W) = (1-W1*W2)^2





Deep Nets with ReLUs

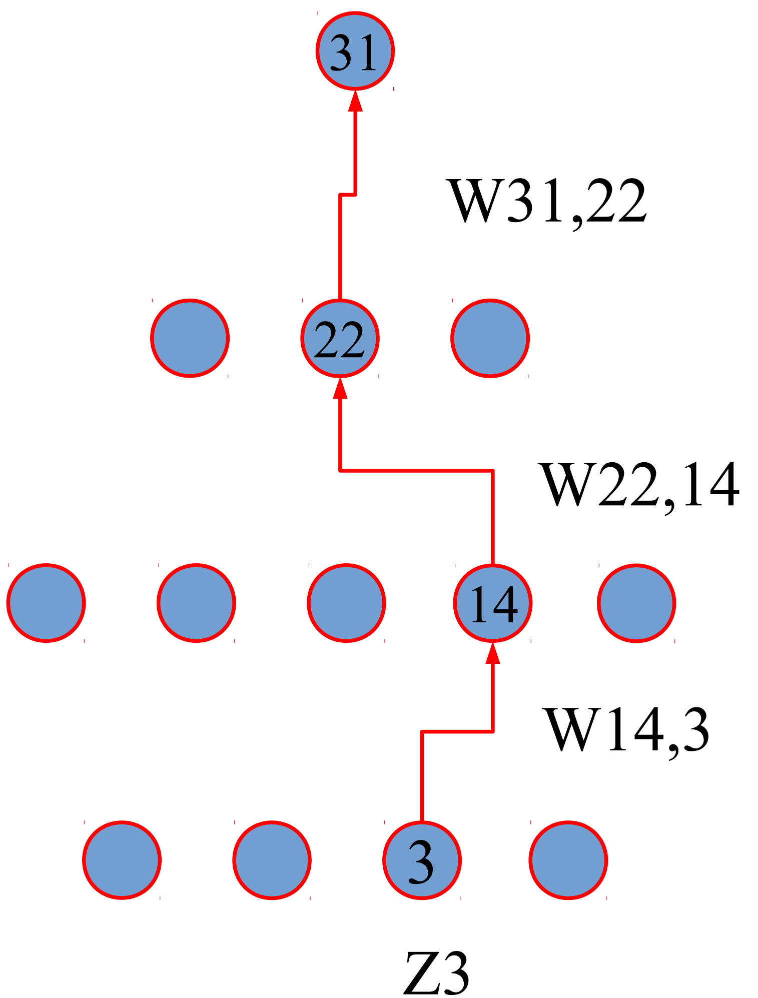
- Single output:

$$\hat{Y} = \sum_P \delta_P(W, X) \left(\prod_{(ij) \in P} W_{ij} \right) X_{P_{start}}$$

- Wij: weight from j to i
- P: path in network from input to output
  - P=(3,(14,3),(22,14),(31,22))
- di: 1 if ReLU i is linear, 0 if saturated.
- Xpstart: input unit for path P.

$$\hat{Y} = \sum_P \delta_P(W, X) \left(\prod_{(ij) \in P} W_{ij} \right) X_{P_{start}}$$

- Dp(W,X): 1 if path P is “active”, 0 if inactive
- Input-output function is piece-wise linear
- Polynomial in W with random coefficients



Deep Convolutional Nets (and other deep neural nets)

Y LeCun

- Training sample: (Xi,Yi) k=1 to K
- Objective function (with margin-type loss = ReLU)

$$L(W)=\sum_k ReLU(1-Y^k\sum_P \delta_P(W,X^k)(\prod_{(ij)\in P} W_{ij})X^k_{P_{start}})$$

$$L(W)=\sum_k \sum_P (X^k_{P_{start}}Y^k)\delta_P(W,X^k)(\prod_{(ij)\in P} W_{ij})$$

$$L(W)=\sum_P [\sum_k (X^k_{P_{start}}Y^k)\delta_P(W,X^k)](\prod_{(ij)\in P} W_{ij})$$

$$L(W)=\sum_P C_p(X,Y,W)(\prod_{(ij)\in P} W_{ij})$$

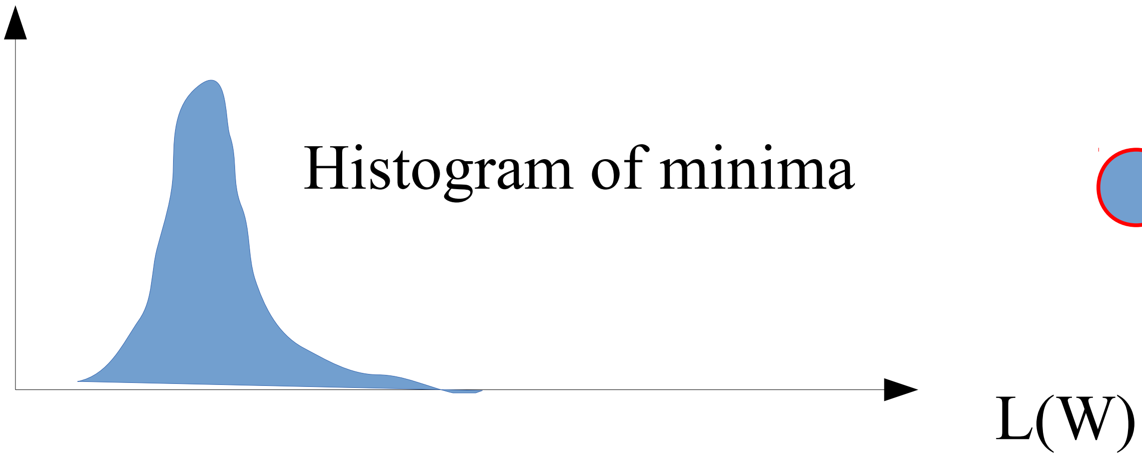
- Polynomial in W of degree l (number of adaptive layers)
- Continuous, piece-wise polynomial with “switched” and partially random coefficients
  - Coefficients are switched in an out depending on W

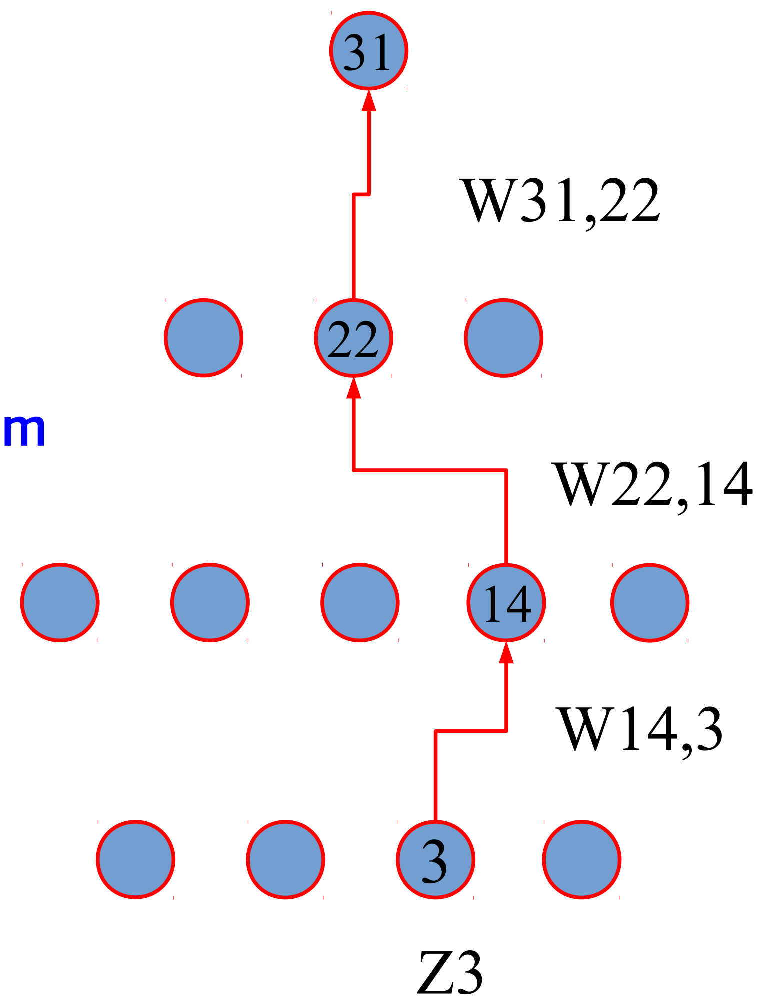
Deep Nets with ReLUs: Objective Function is Piecewise Polynomial

- If we use a hinge loss, delta now depends on label Yk:

$$L(W)=\sum_P C_p(X,Y,W)\Big(\prod_{(ij)\in P} W_{ij}\Big)$$

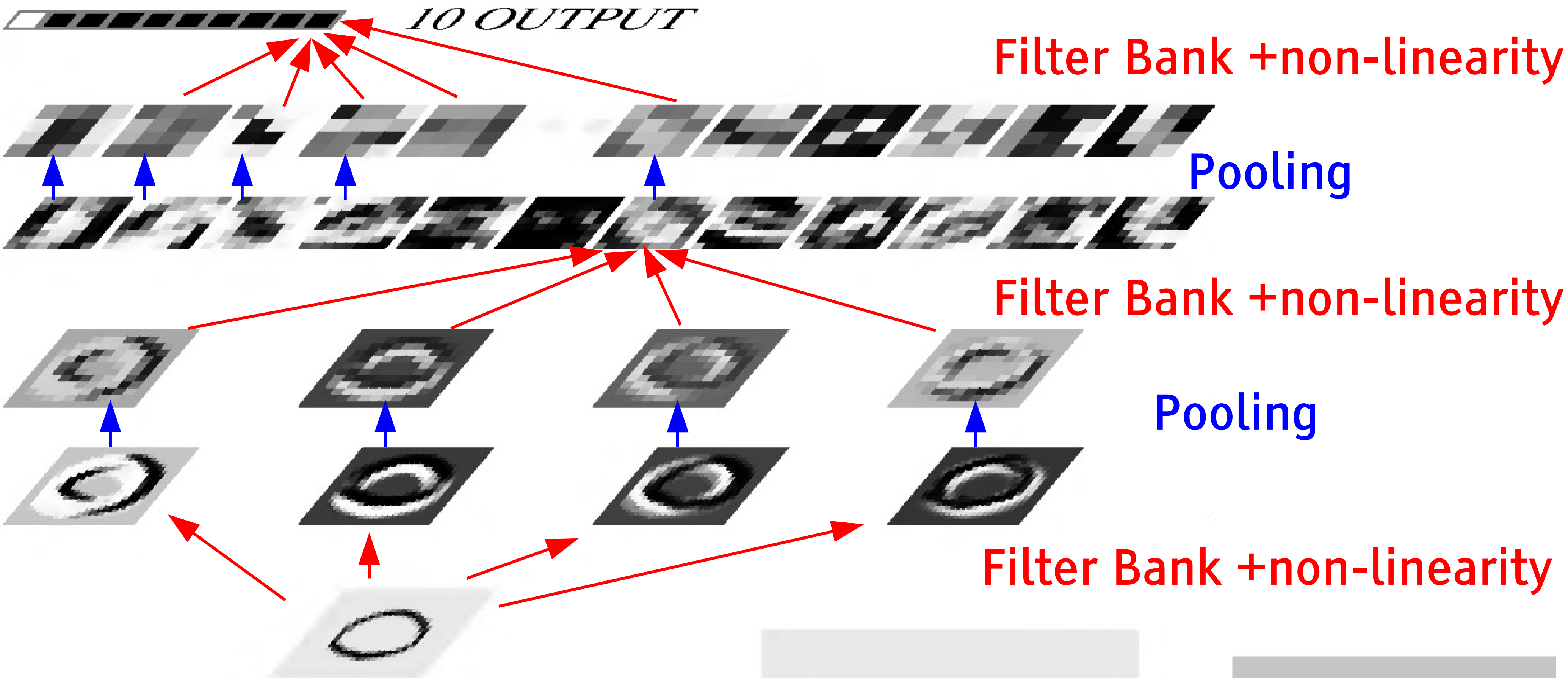
- Piecewise polynomial in W with random coefficients
- A lot is known about the distribution of critical points of polynomials on the sphere with random (Gaussian) coefficients [Ben Arous et al.]
  - High-order spherical spin glasses
  - Random matrix theory

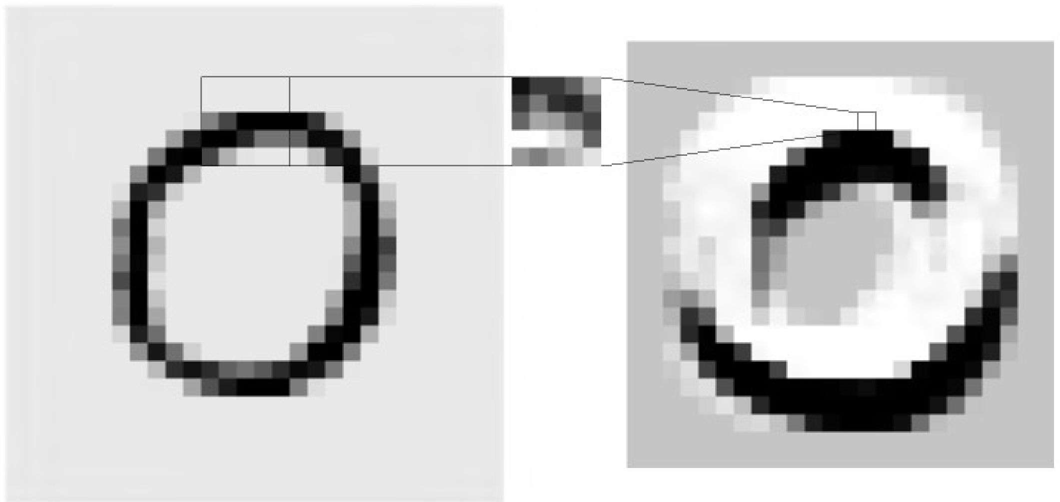




Convolutional Networks

Convolutional Network

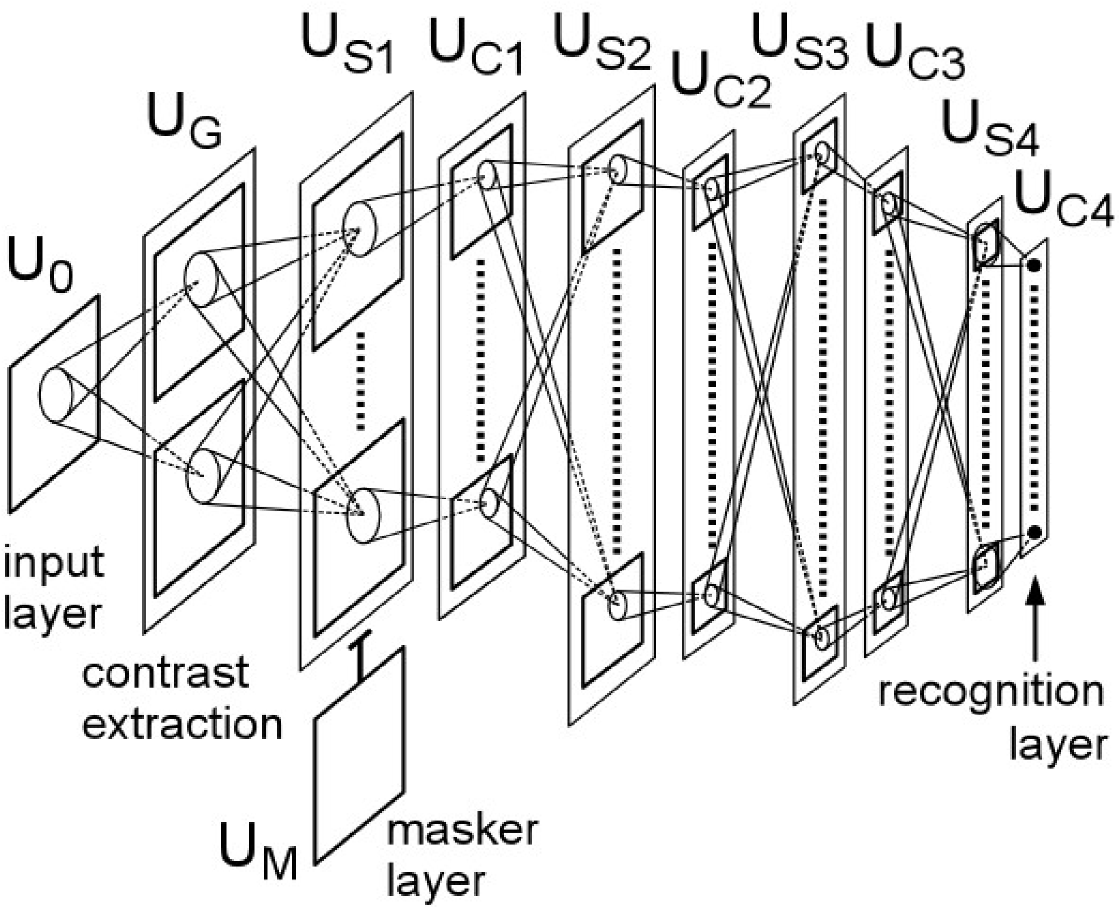


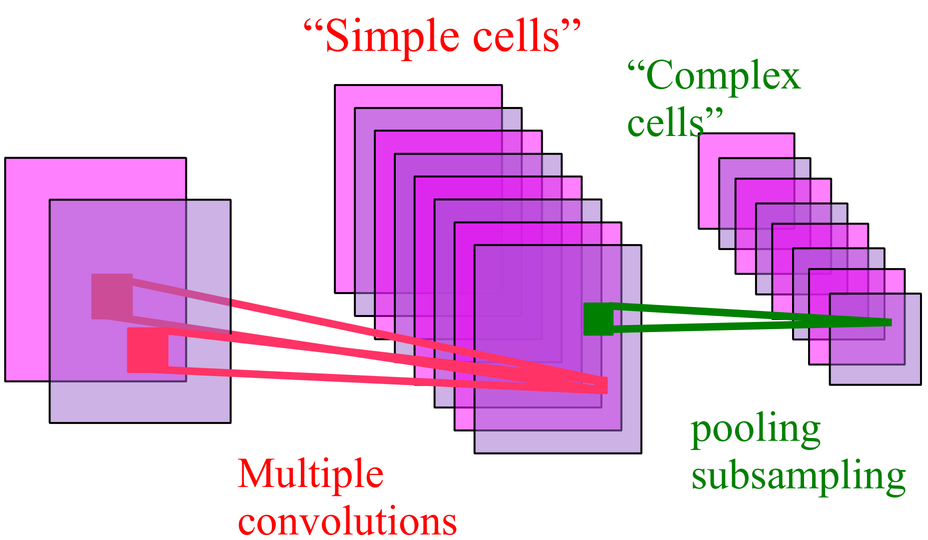


- [LeCun et al. NIPS 1989]

Early Hierarchical Feature Models for Vision

- [Hubel & Wiesel 1962]:
  - simple cells detect local features
  - complex cells "pool" the outputs of simple cells within a retinotopic neighborhood.

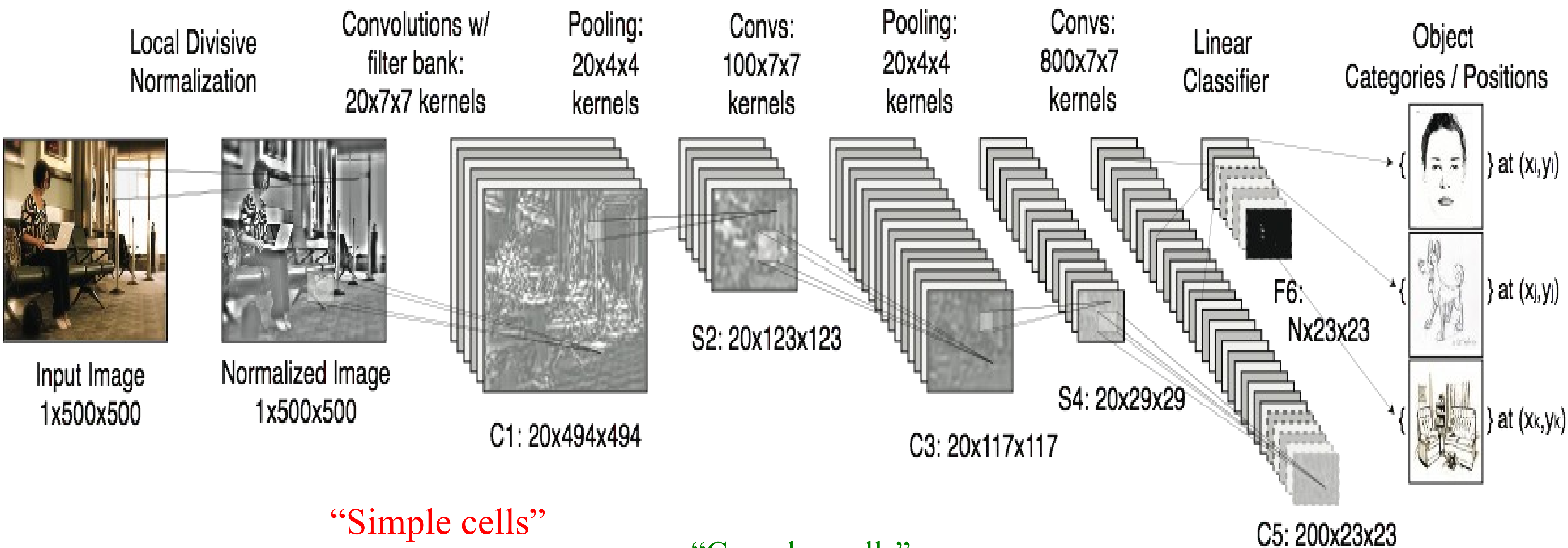


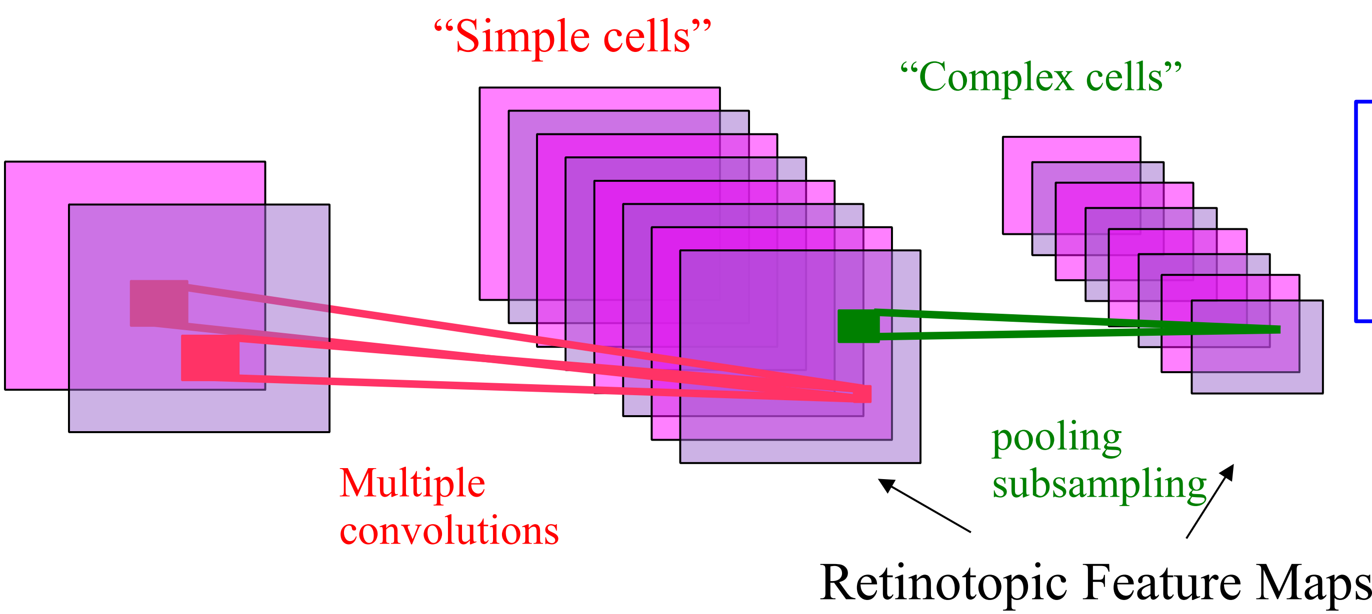


Cognitron & Neocognitron [Fukushima 1974-1982]

The Convolutional Net Model
(Multistage Hubel-Wiesel system)

Y LeCun

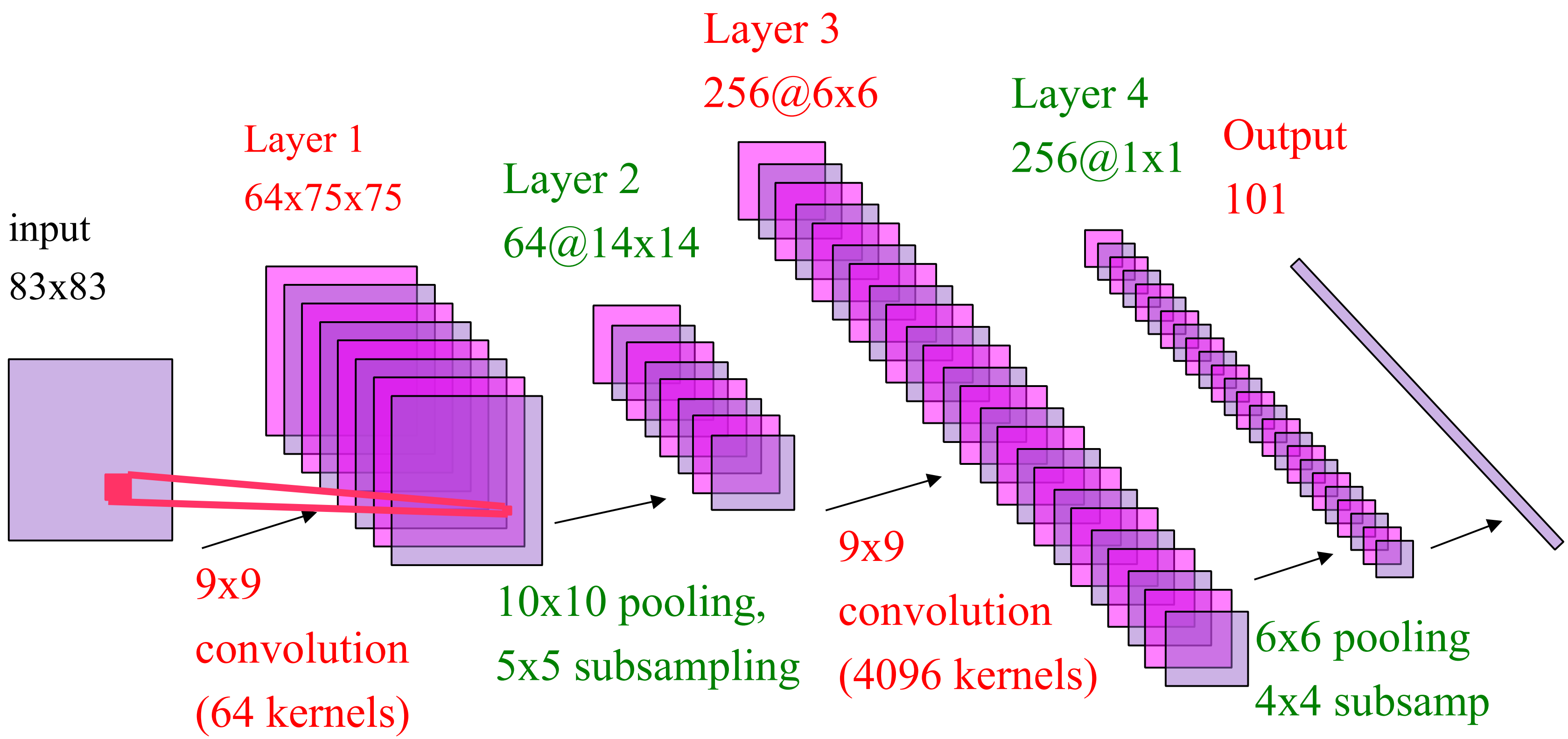




- Training is supervised
- With stochastic gradient descent

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

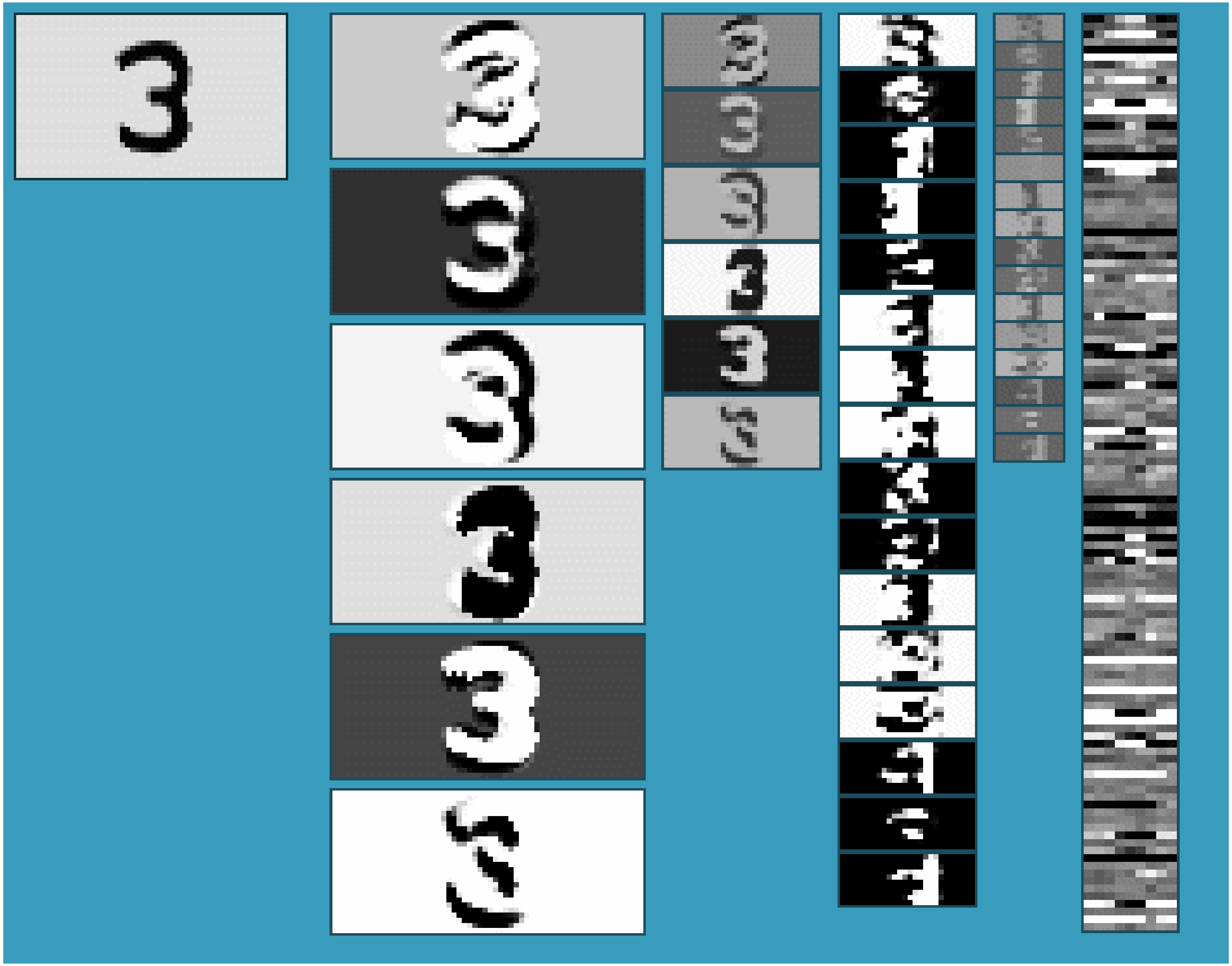
Convolutional Network (ConvNet)



- 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)

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



Curved manifold

Flatter manifold

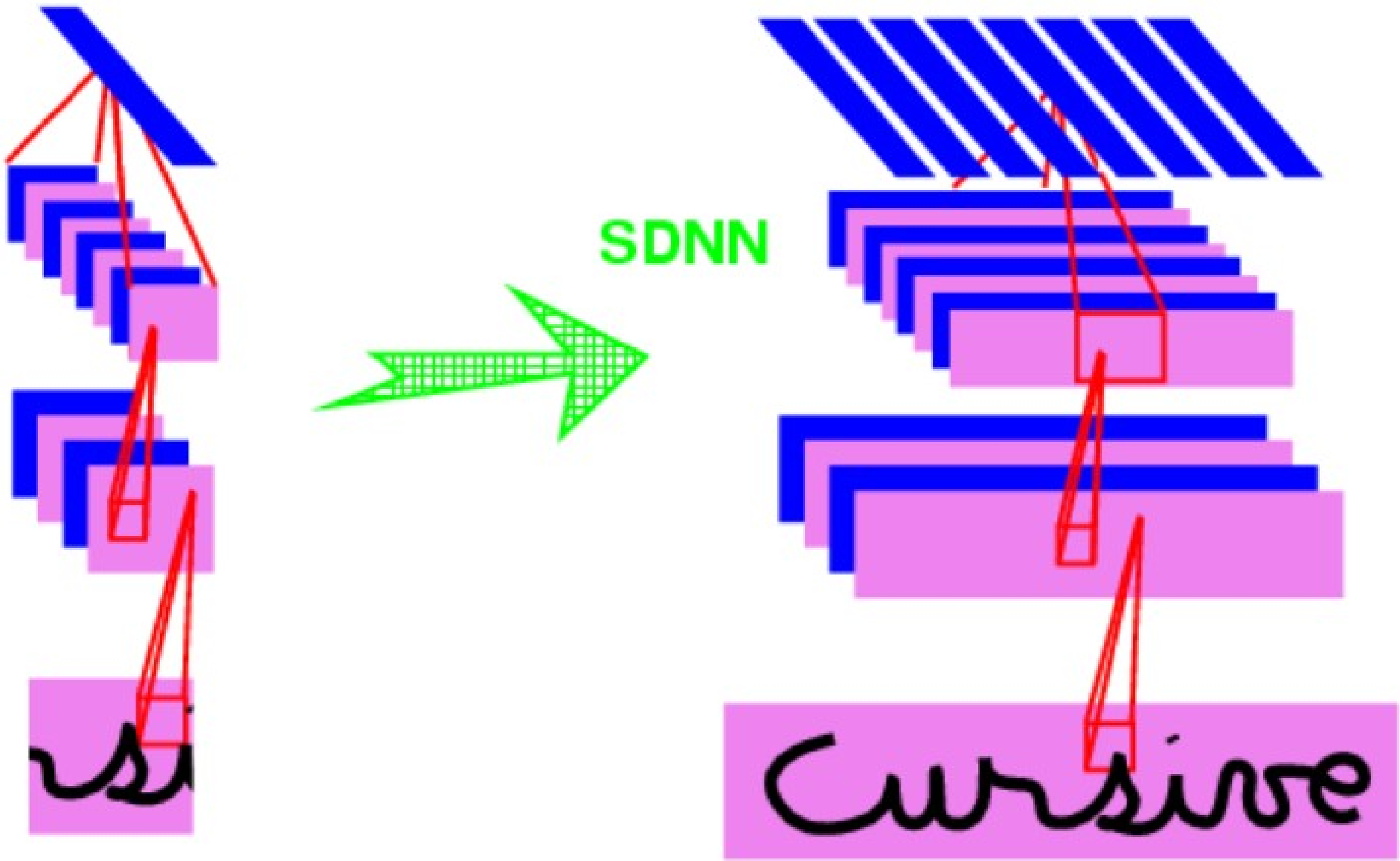
LeNet1 Demo from 1993

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

Brute Force Approach
To
Multiple Object Recognition

Idea #1: Sliding Window ConvNet + Weighted FSM

- "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

Idea #1: Sliding Window ConvNet + Weighted FSM

Y LeCun

Idea #1: Sliding Window ConvNet + Weighted FSM

Convolutional Networks
In
Visual Object Recognition

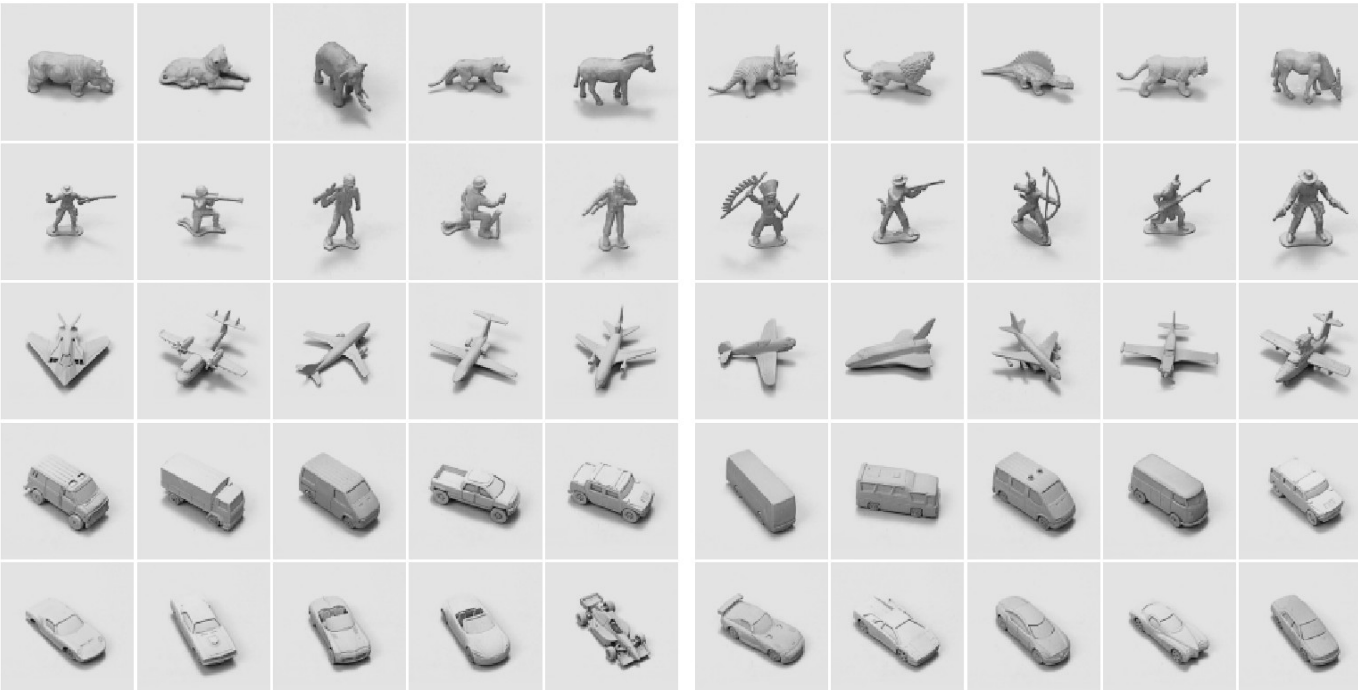
We knew ConvNet worked well with characters and small images

- **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

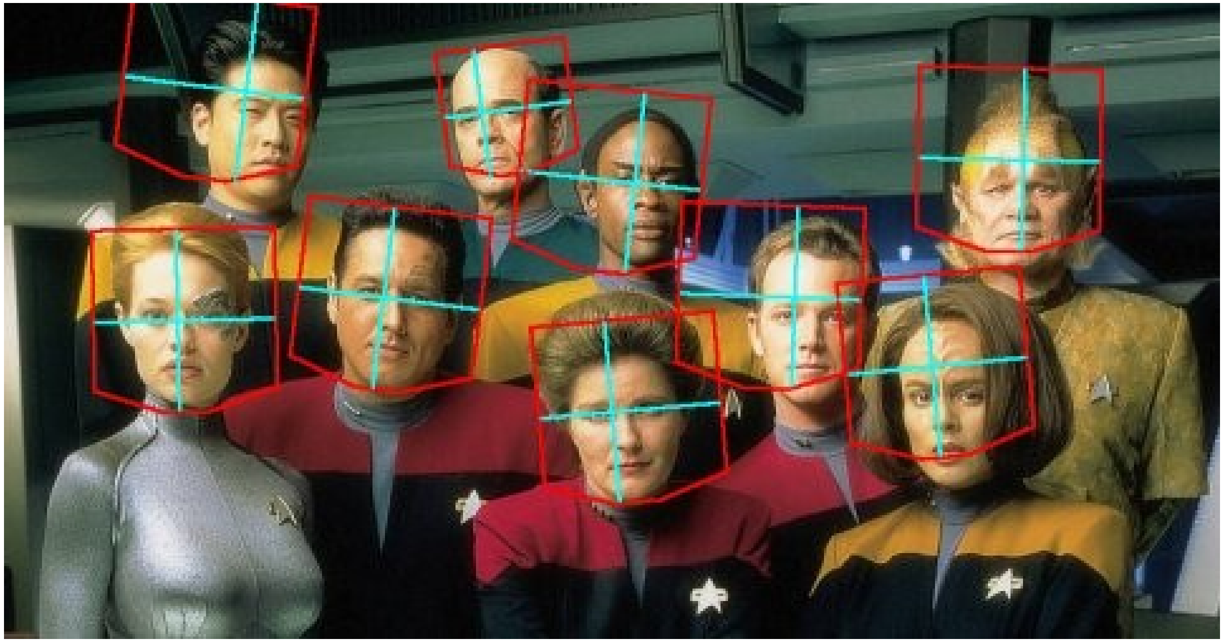


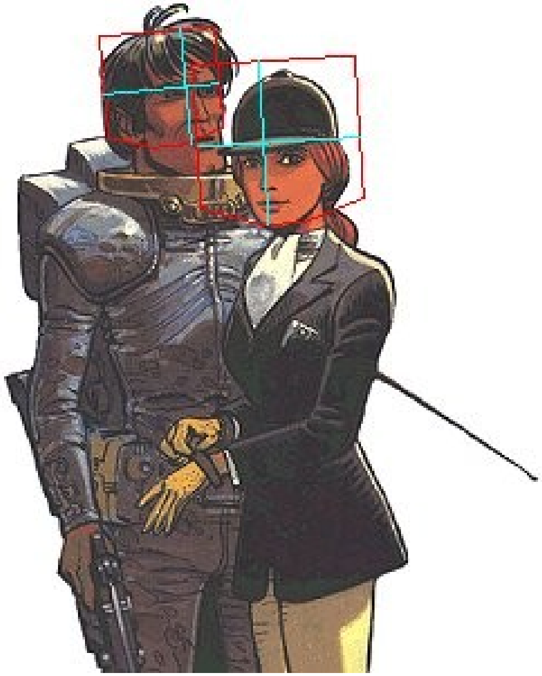
- Less than 6% error on test set with cluttered backgrounds

291,600 training samples,
58,320 test samples

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

Data Set->	TILTED		PROFILE		MIT+CMU	
False positives per image->	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	



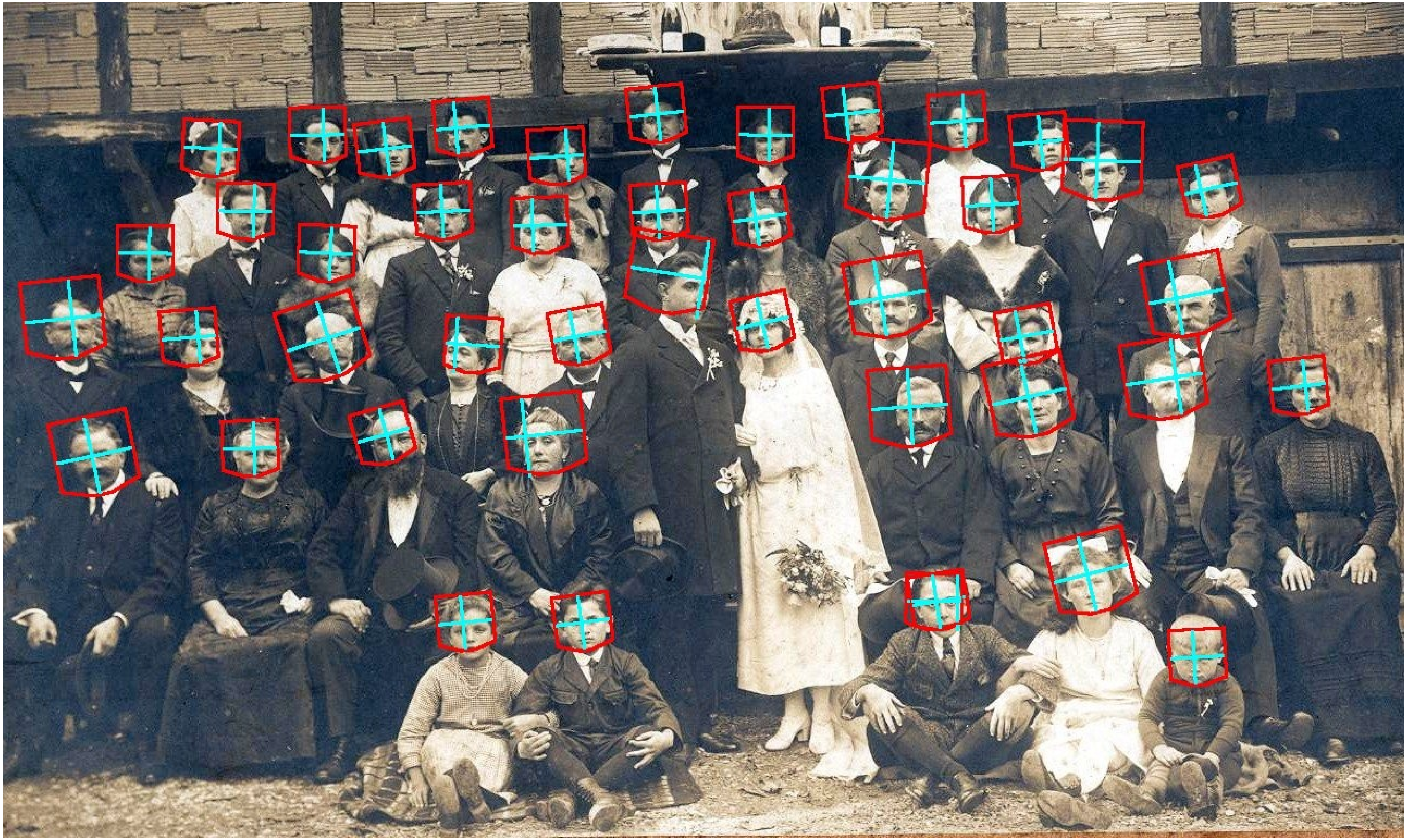


[Vaillant et al. IEE 1994][Osadchy et al. 2004] [Osadchy et al, JMLR 2007]

Simultaneous face detection and pose estimation

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

Simultaneous face detection and pose estimation



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

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

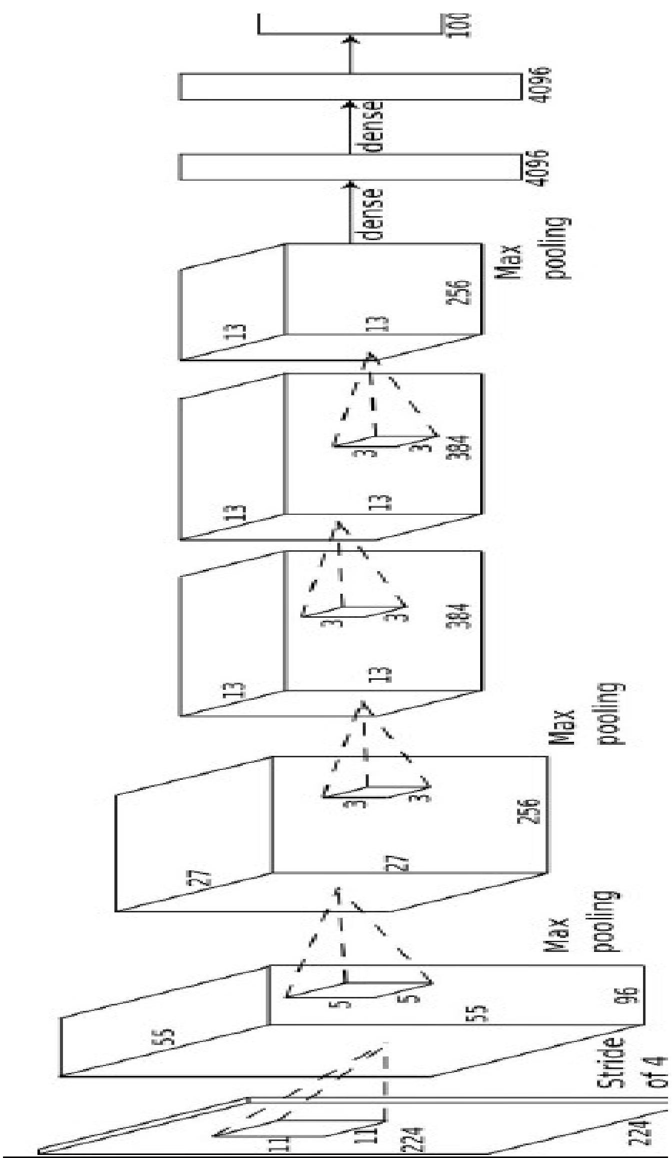
- But we couldn't beat the state of the art because the datasets were too small
- Caltech101: 101 categories, 30 samples per category.
- But we learned that rectification and max pooling are useful! [Jarrett et al. ICCV 2009]

Single Stage System: $[64.F^{9\times9}_{CSG} - R/N/P^{5\times5}]$ **- 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: $[64.F^{9\times9}_{CSG} - R/N/P^{5\times5}] - [256.F^{9\times9}_{CSG} - R/N/P^{4\times4}]$ **- 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: $[64.F^{9\times9}_{CSG} - R/N/P^{5\times5}]$ **- PMK-SVM**					
U	64.0%				
Two Stages: $[64.F^{9\times9}_{CSG} - R/N/P^{5\times5}] - [256.F^{9\times9}_{CSG} - R/N]$ **- PMK-SVM**					
UU	52.8%				

Object Recognition [Krizhevsky, Sutskever, Hinton 2012]

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

4M	FULL CONNECT	4Mflop
16M	FULL 4096/ReLU	16M
37M	FULL 4096/ReLU	37M
	MAX POOLING	
442K	CONV 3x3/ReLU 256fm	74M
1.3M	CONV 3x3ReLU 384fm	224M
884K	CONV 3x3/ReLU 384fm	149M
	MAX POOLING 2x2sub	
	LOCAL CONTRAST NORM	
307K	CONV 11x11/ReLU 256fm	223M
	MAX POOL 2x2sub	
	LOCAL CONTRAST NORM	
35K	CONV 11x11/ReLU 96fm	105M



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

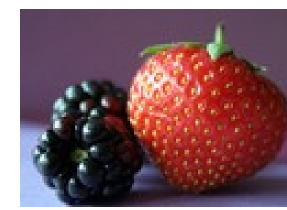
Matchstick

Flute

Backpack

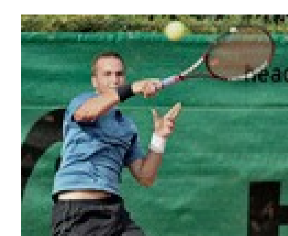
Sea lion

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....)

Object Recognition [Krizhevsky, Sutskever, Hinton 2012]

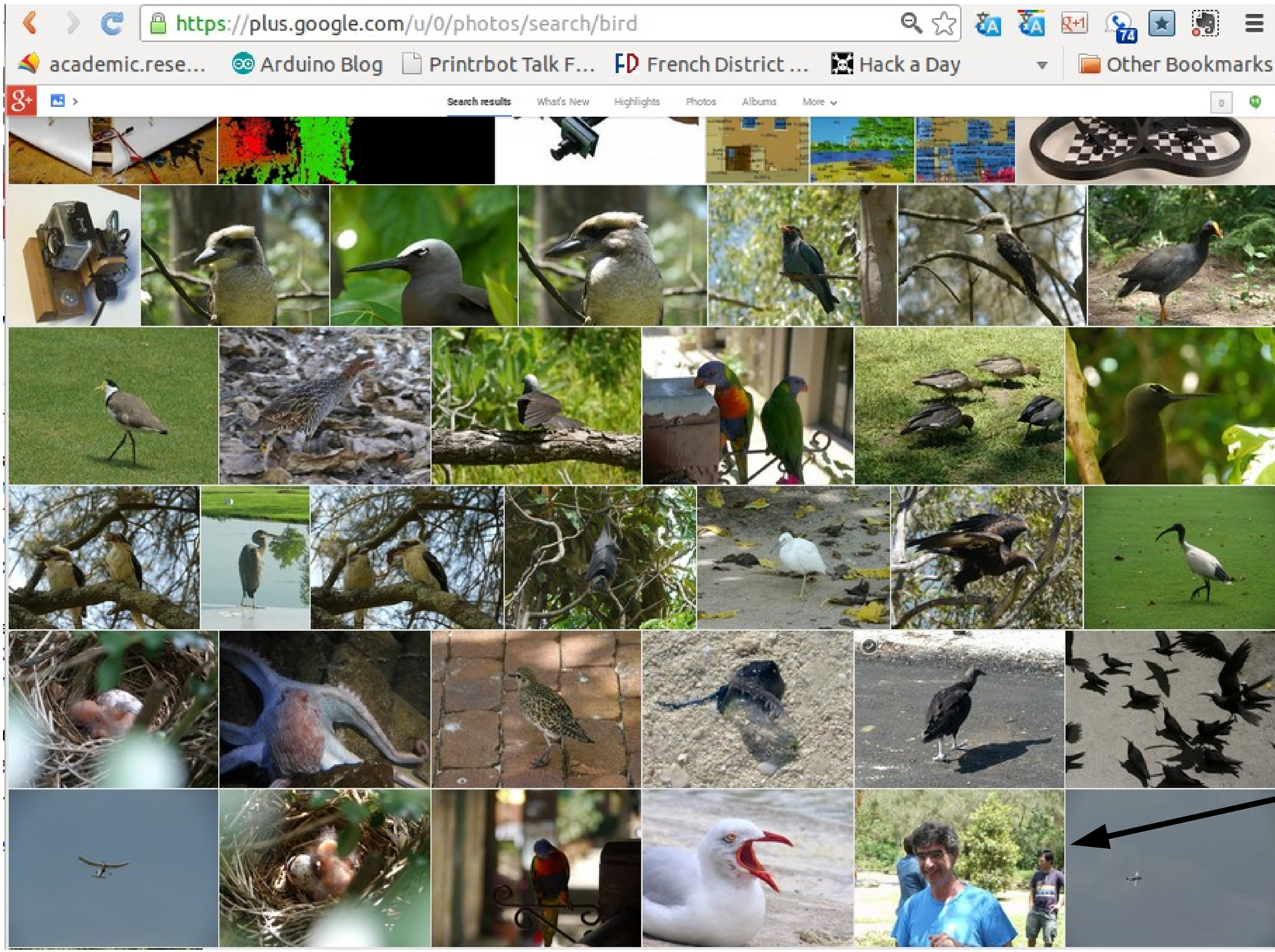
- Method: large convolutional net
  - 650K neurons, 832M synapses, 60M parameters
  - Trained with backprop on GPU
  - Trained “with all the tricks Yann came up with in the last 20 years, plus dropout” (Hinton, NIPS 2012)
  - Rectification, contrast normalization,...
- Error rate: 15% (whenever correct 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

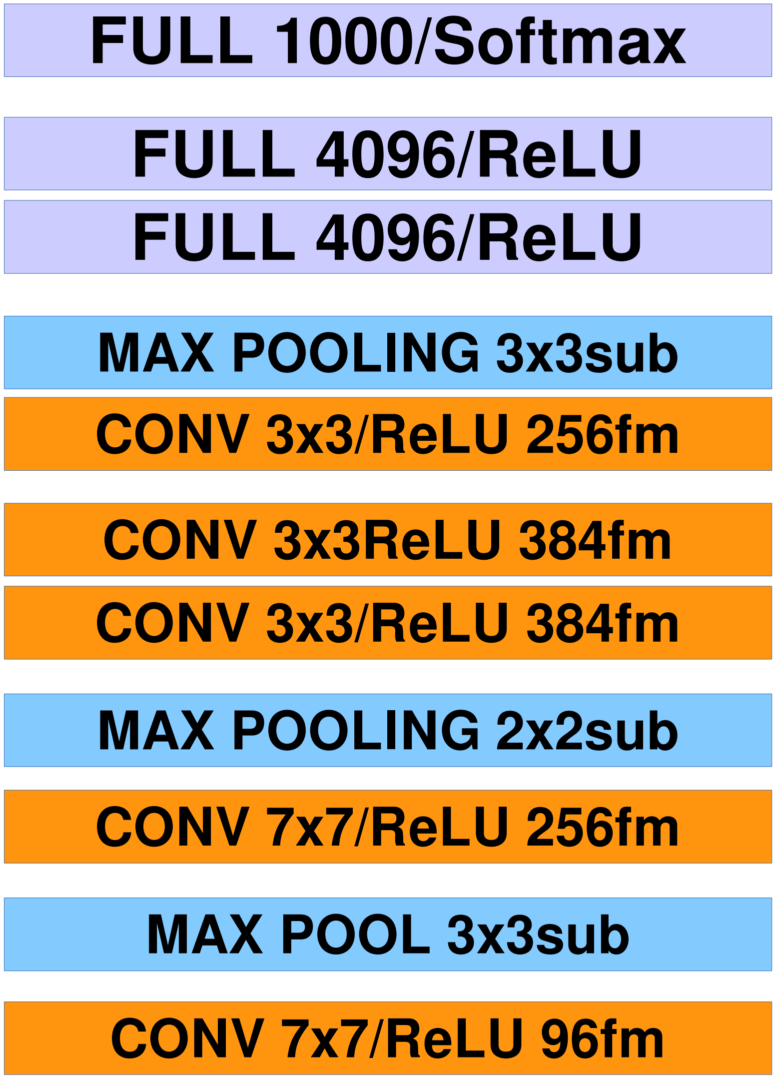
- Searched my personal collection for “bird”



Samy Bengio ???

NYU ConvNet Trained on ImageNet: OverFeat

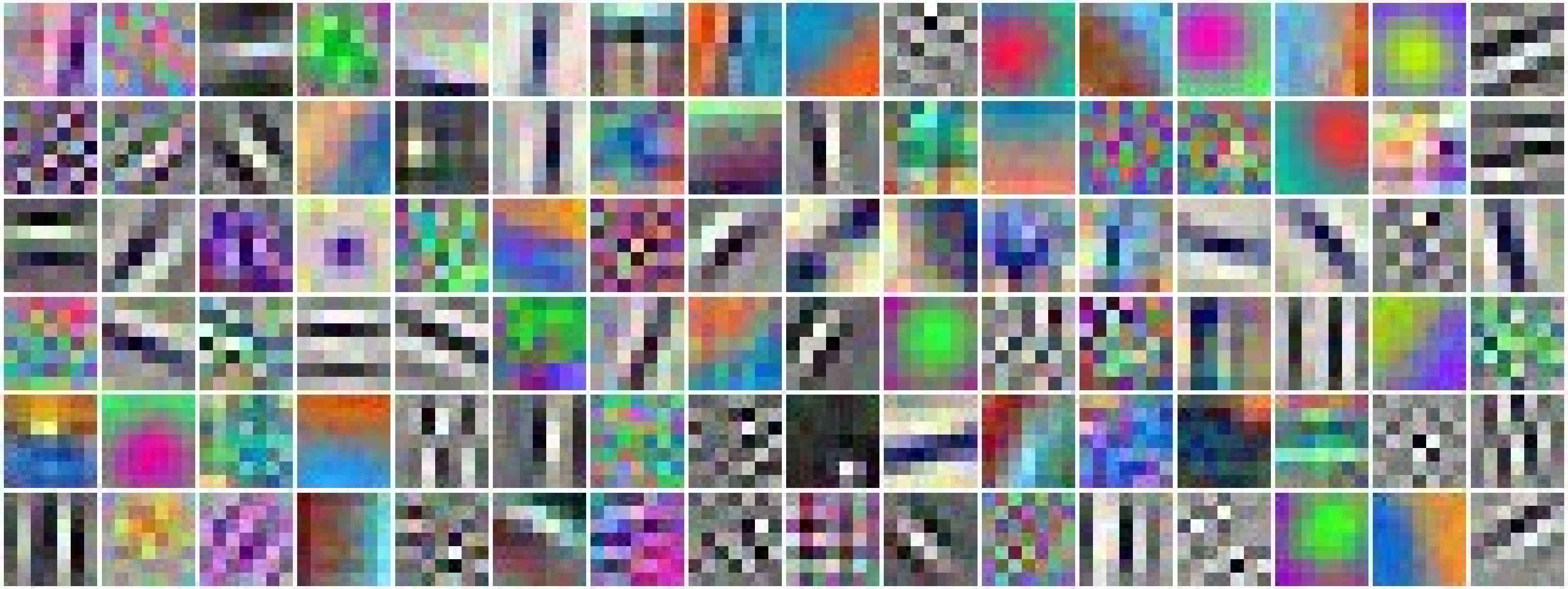
- [Sermanet et al. arXiv:1312.6229]
- Trained on GPU using Torch7
- Uses a number of new tricks
- Classification 1000 categories:
  - 13.8% error (top 5) with an ensemble of 7 networks (Krizhevsky: 15%)
  - 15.4% error (top 5) with a single network (Krizhevksy: 18.2%)
- Classification+Localization
  - 30% error (Krizhevsky: 34%)
- Detection (200 categories)
  - 19% correct
- Dowloadable code (running, no training)
  - **Search for "overfeat NYU" on Google**
  - **http://cilvr.nyu.edu → software**



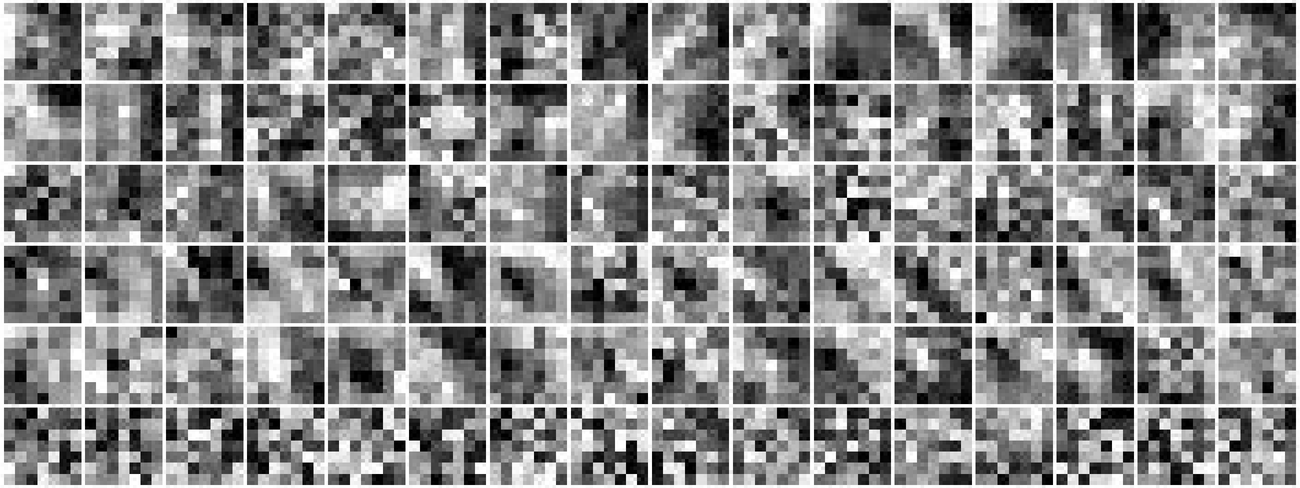
Kernels: Layer 1 (7x7)and Layer 2 (7x7)

Y LeCun

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

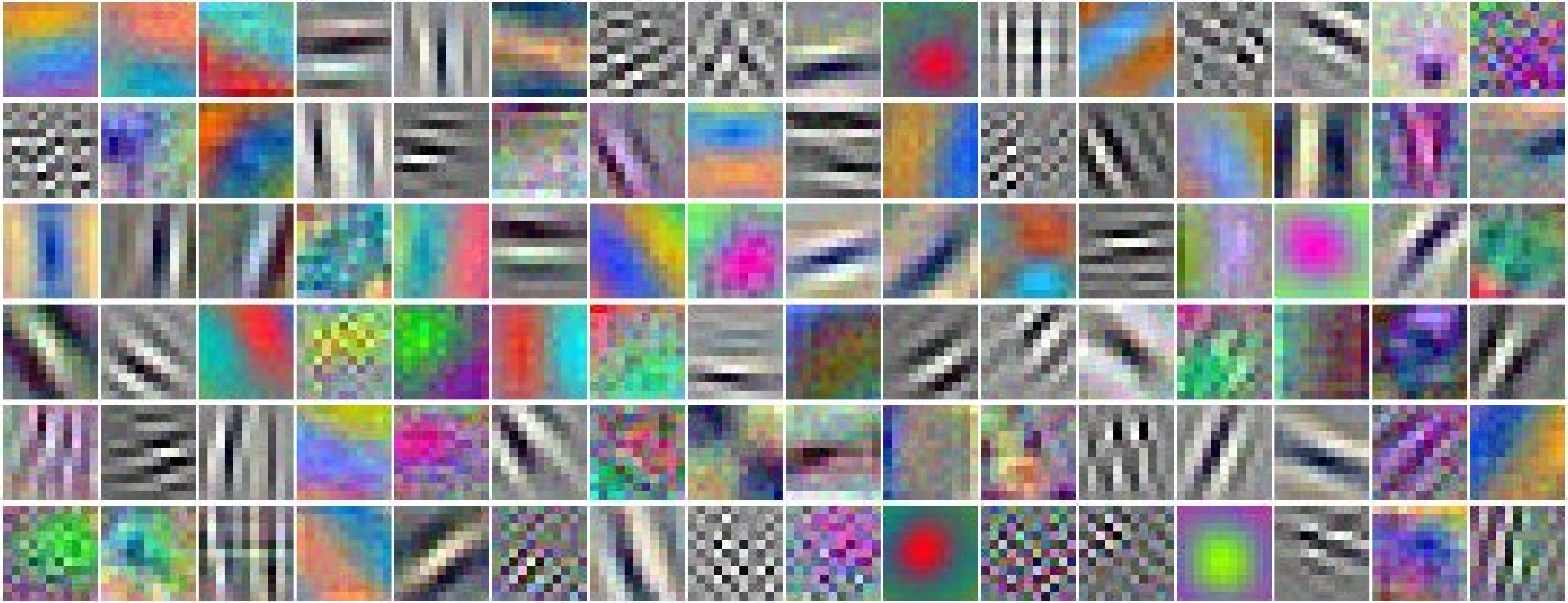


- Layer 2: 96x256 kernels, 7x7



Kernels: Layer 1 (11x11)

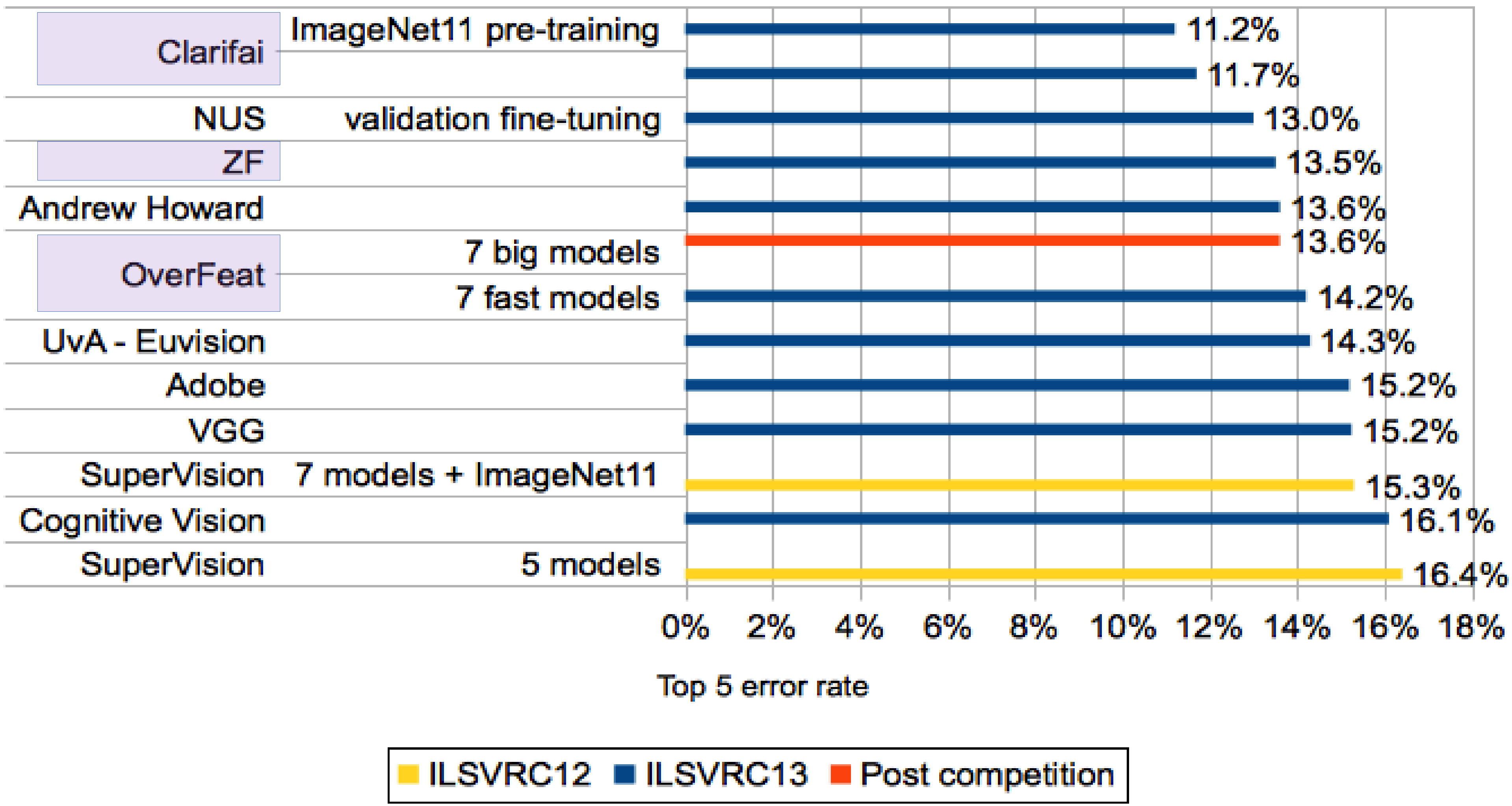
- Layer 1: 3x96 kernels, RGB->96 feature maps, 11x11 Kernels, stride 4



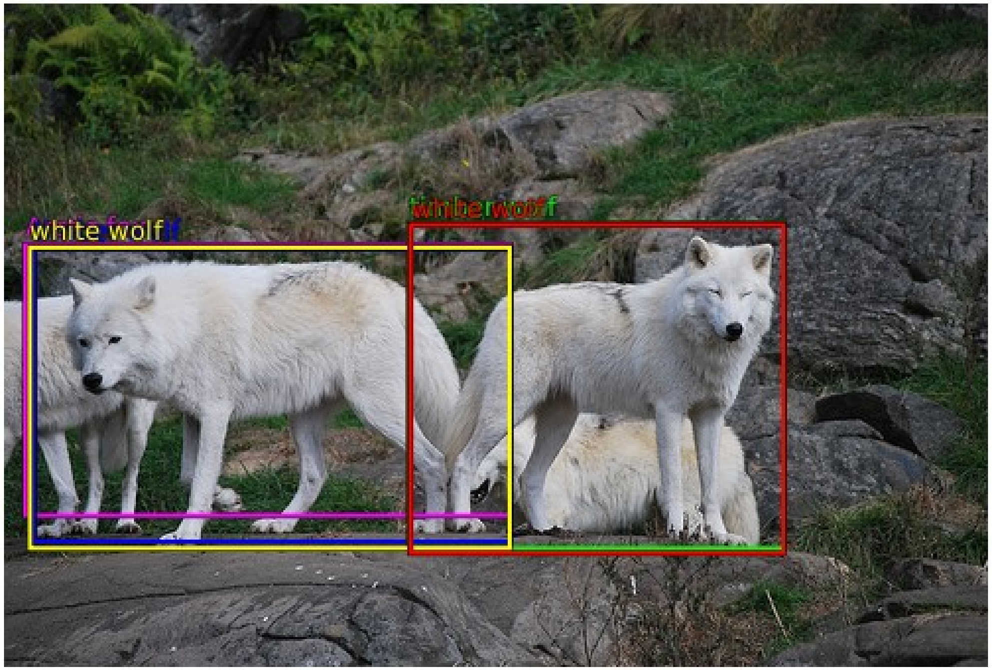
ImageNet 2013: Classification

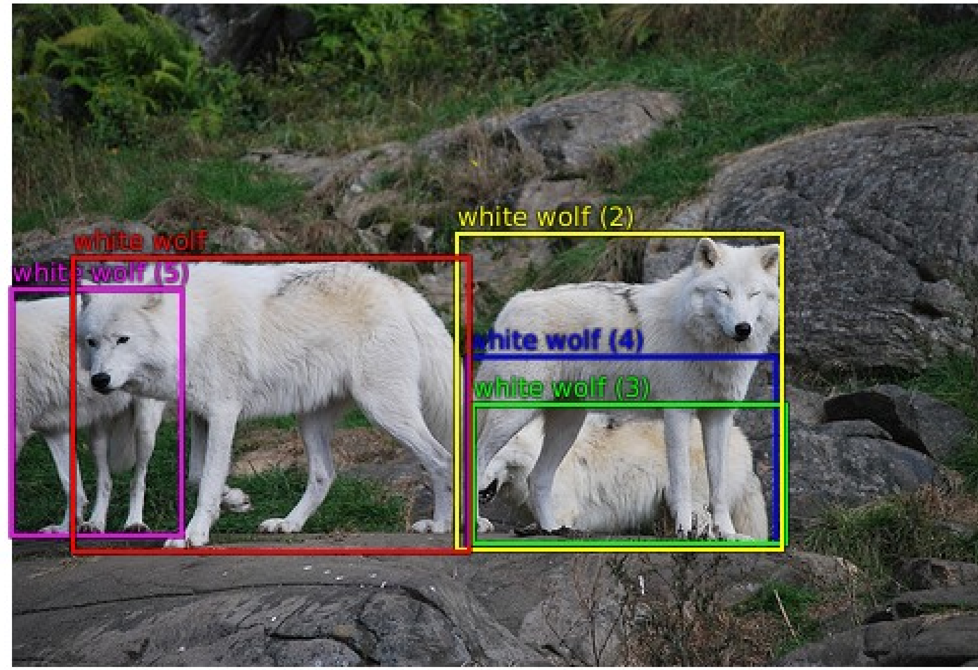
Y LeCun

- 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
- (NYU Teams in Purple)



Classification+Localization. Results





Top 5:

white wolf
white wolf
timber wolf
timber wolf
Arctic fox

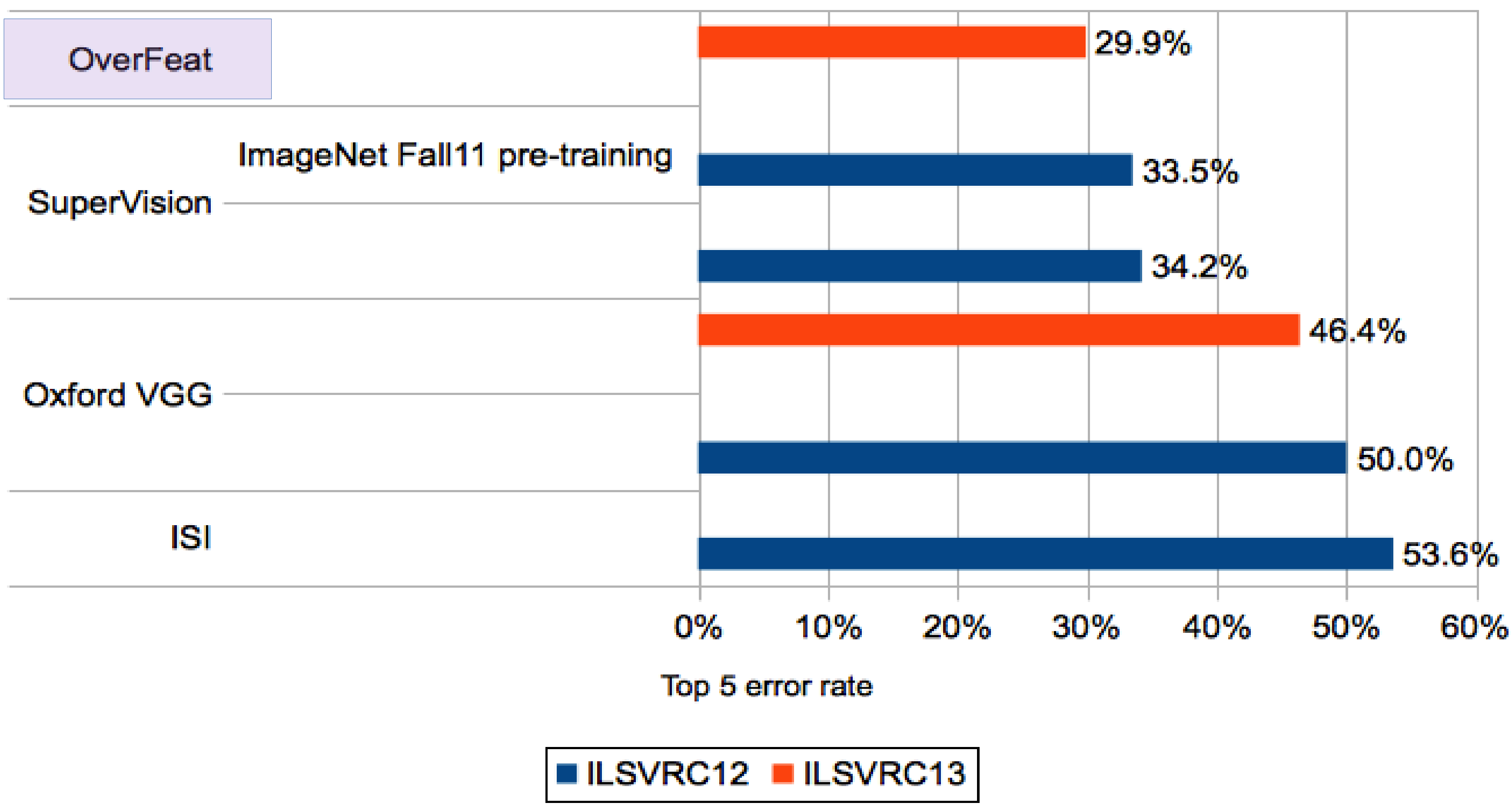
Groundtruth:

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

ILSVRC2012_val_00000027.JPEG

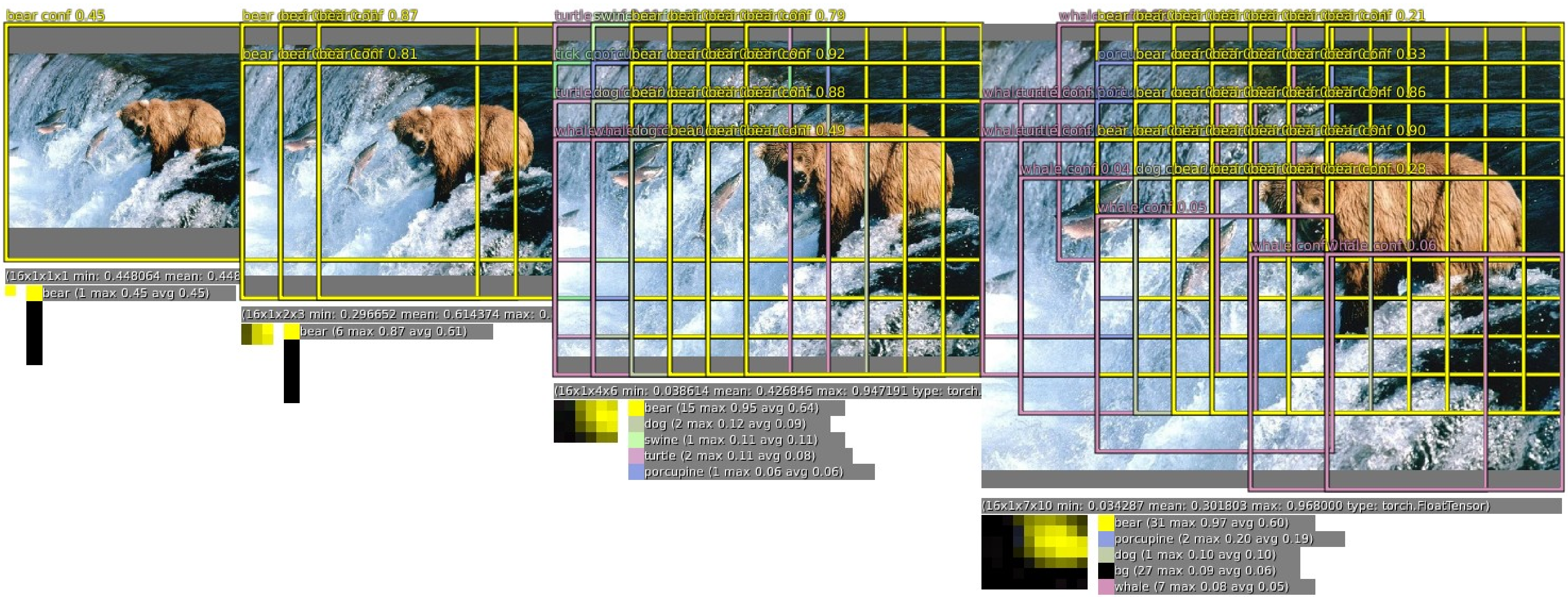
Classification+Localization. Error Rates

- **It's best to propose several categories for the same window**
  - One of them might be right



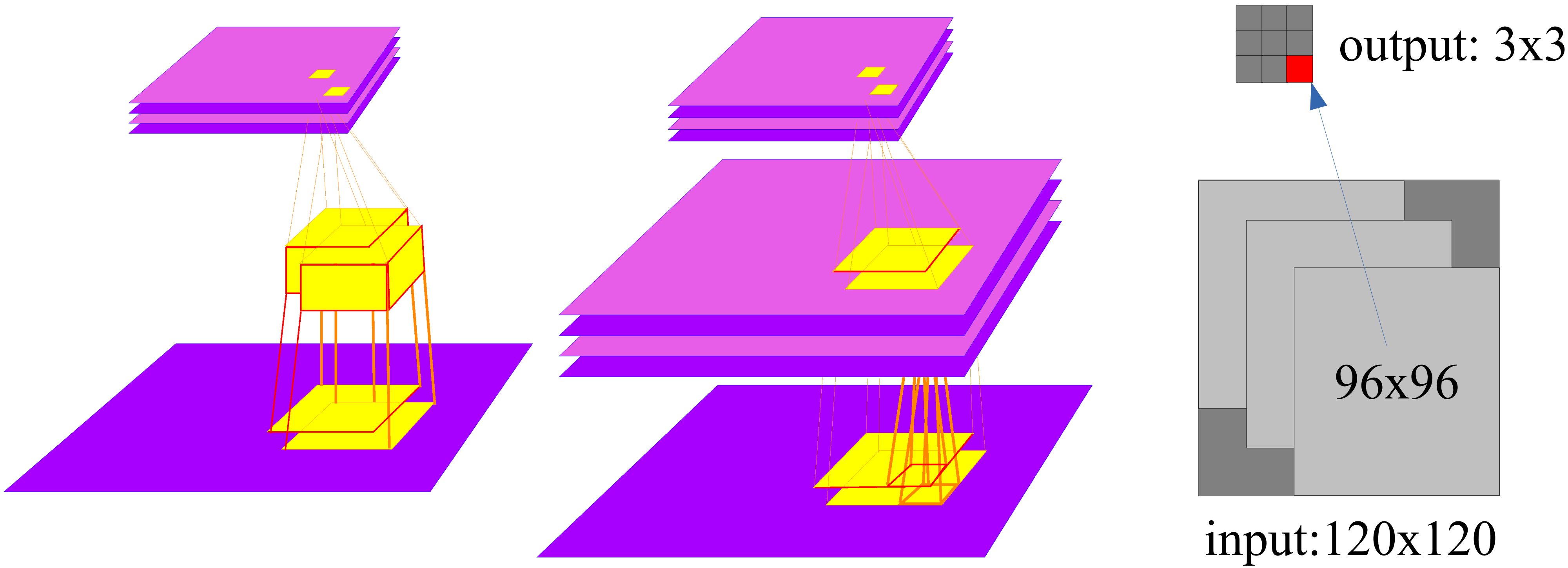
Classification + Localization: multiscale sliding window

- **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

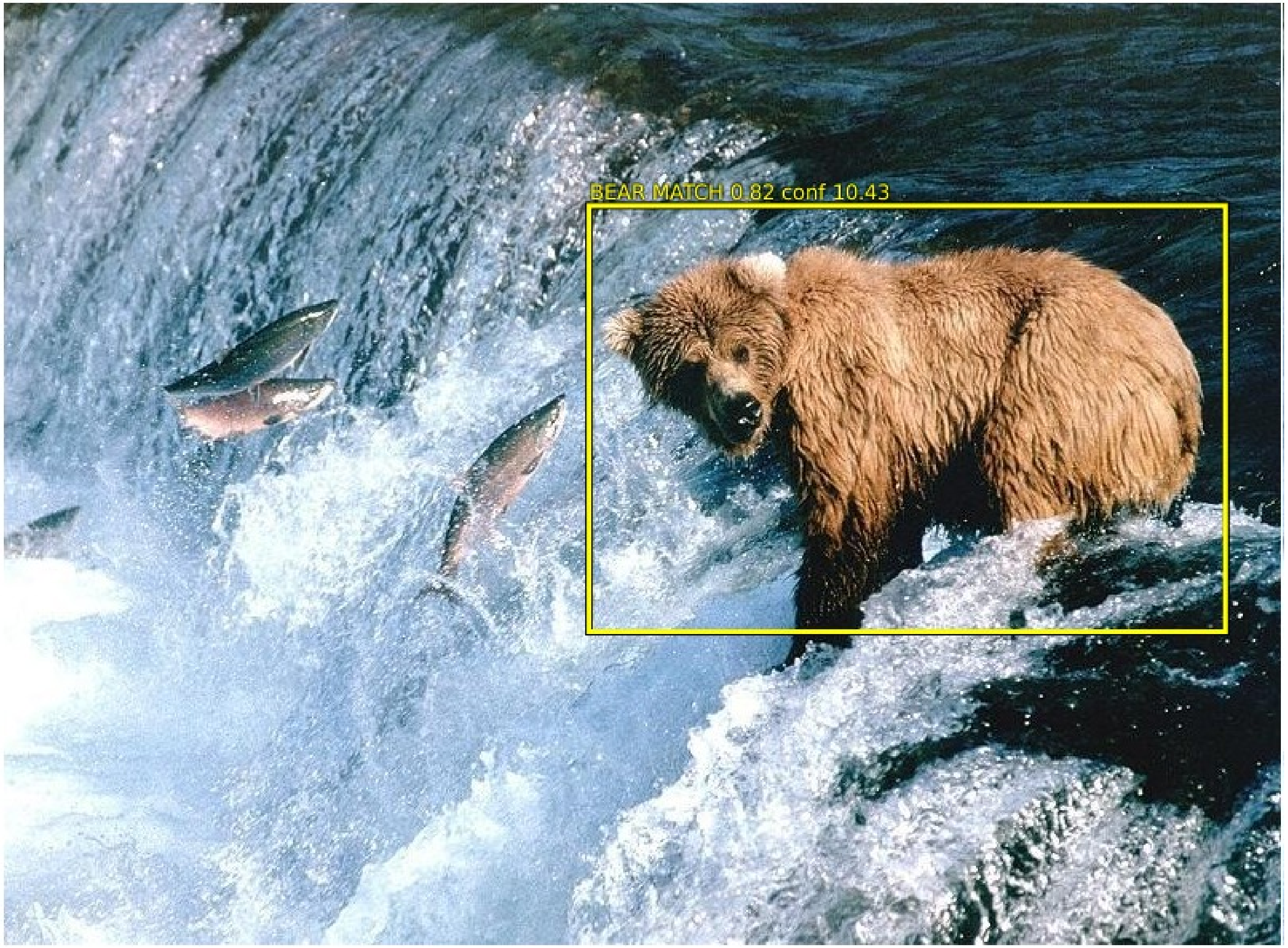
Classification + Localization: sliding window + bounding box regression

Y LeCun

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

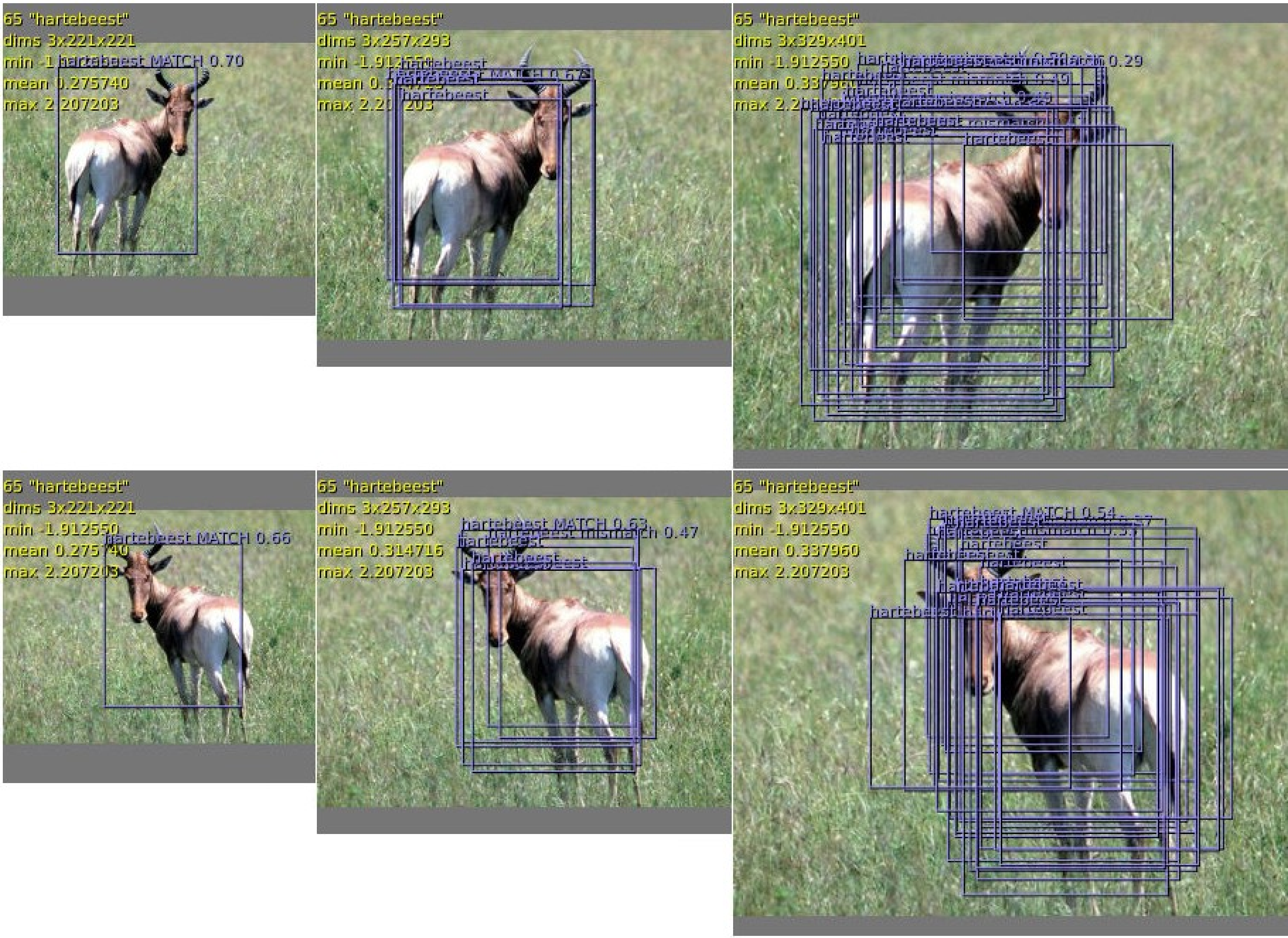
Classification + Localization: sliding window + bounding box regression + bbox voting

- 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

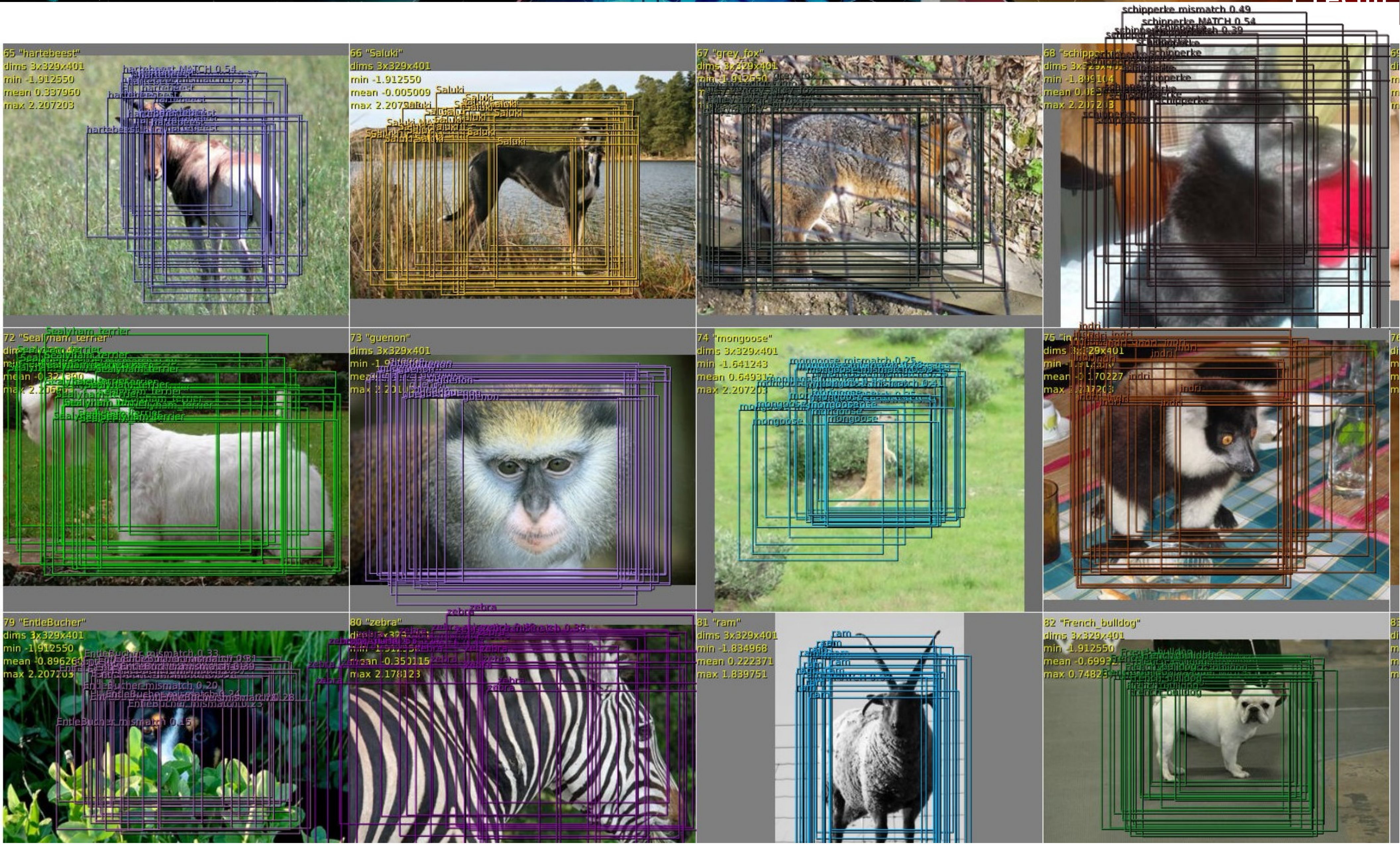


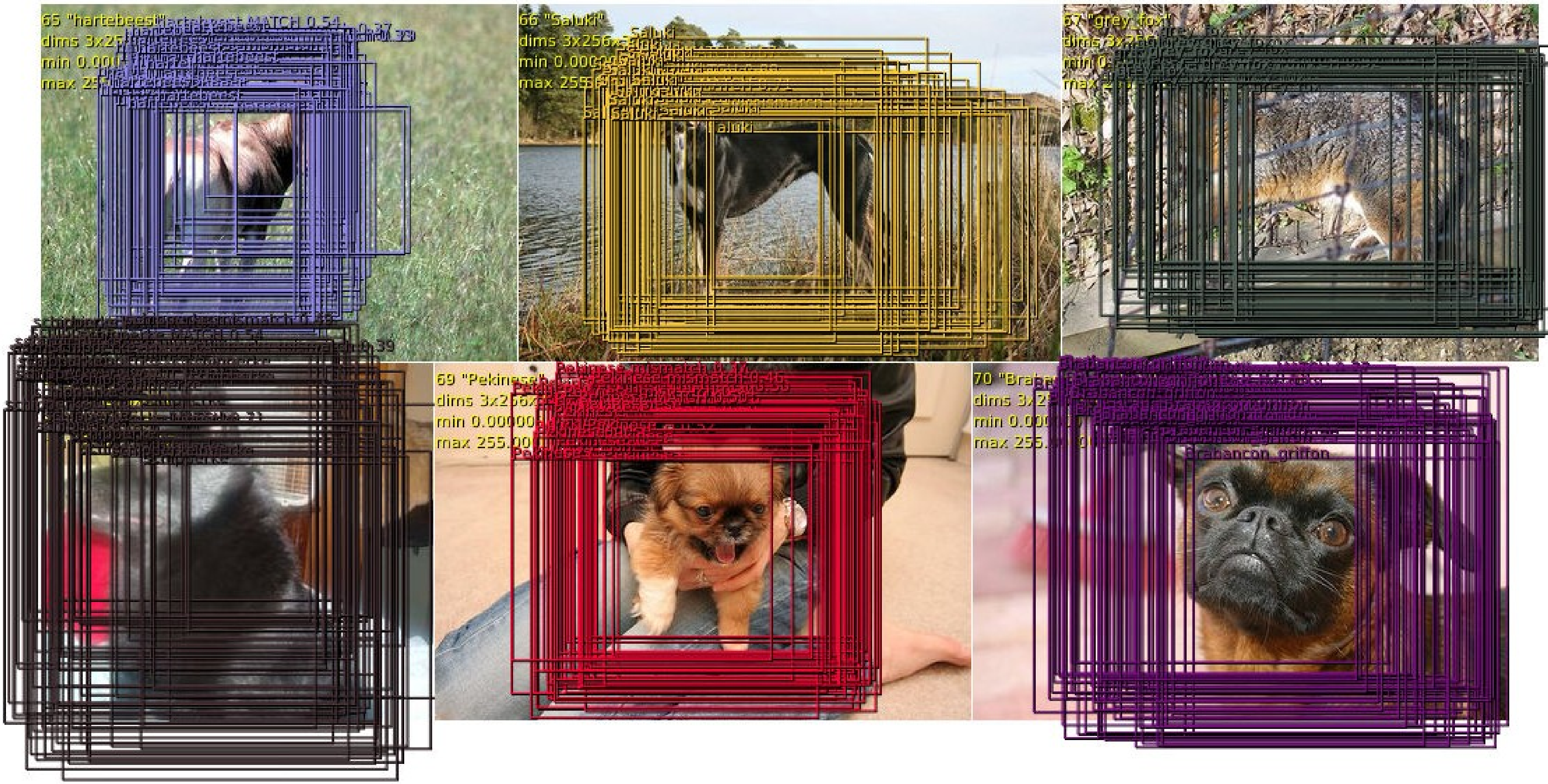
Localization: Sliding Window + bbox vote + multiscale

Y LeCun

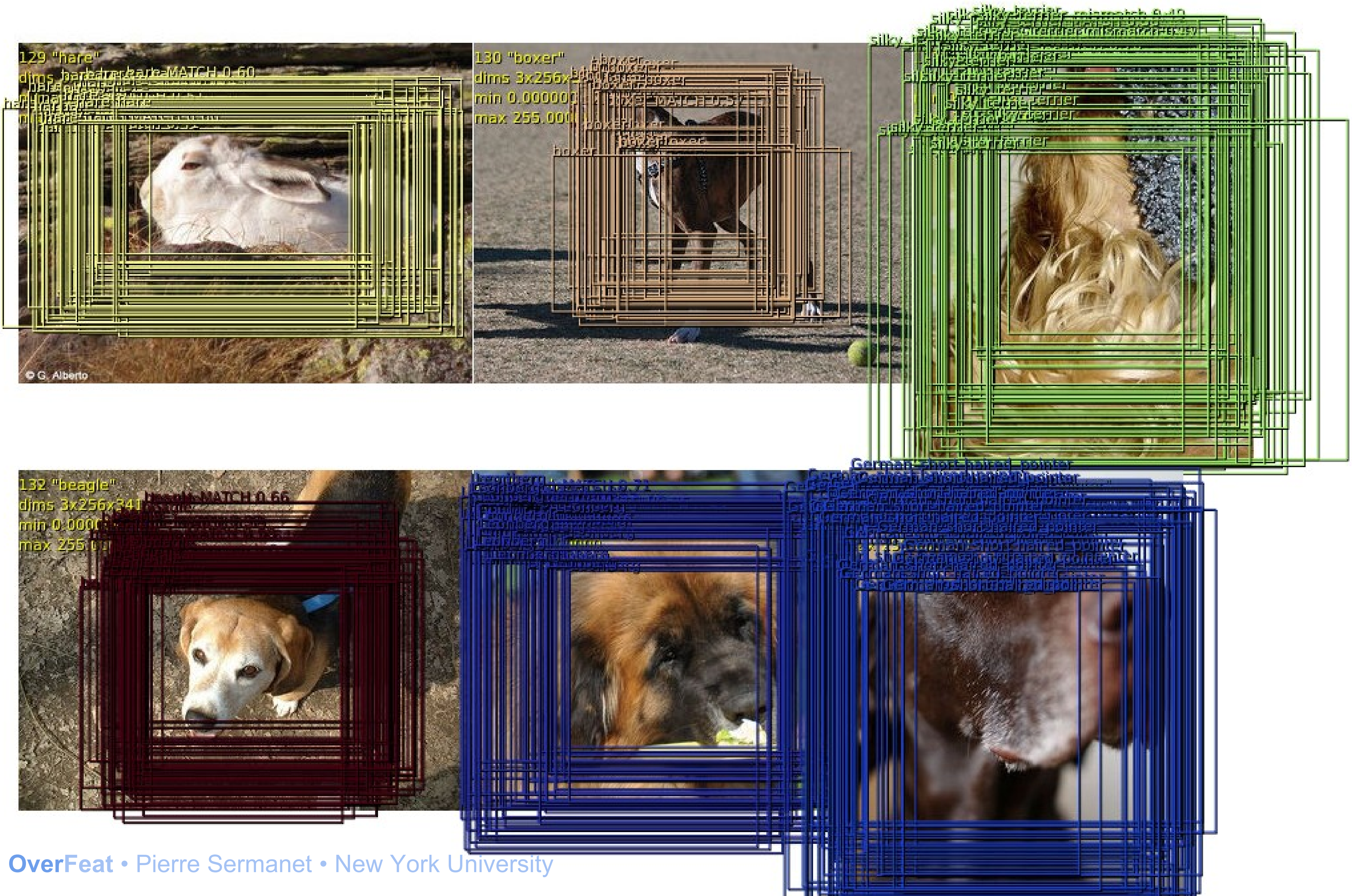


Detection / Localization

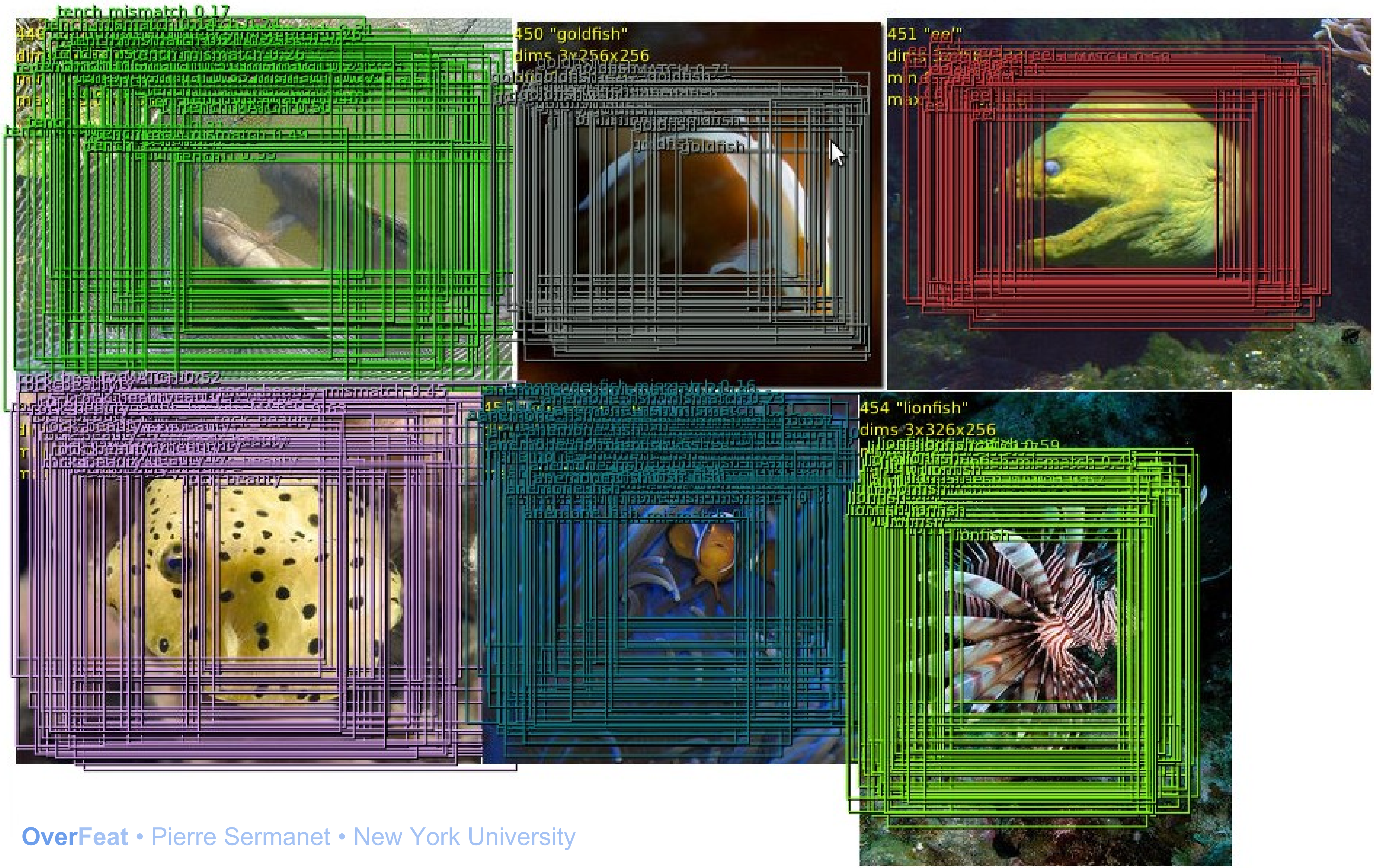



65 "hartebeest"
dims 3x25
min 0.000
max 25
66 "Saluki"
dims 3x256x
min 0.000
max 255
67 "grey_fox"
69 "Pekinese"
dims 3x256x
min 0.00000
max 255.000
70 "Braba
dims 3x25
min 0.000
max 255.
Brabancon_griffon

Detection / Localization



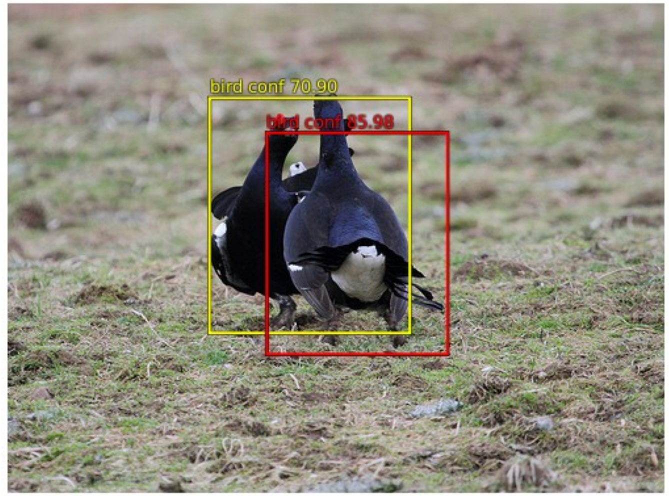
Detection / Localization

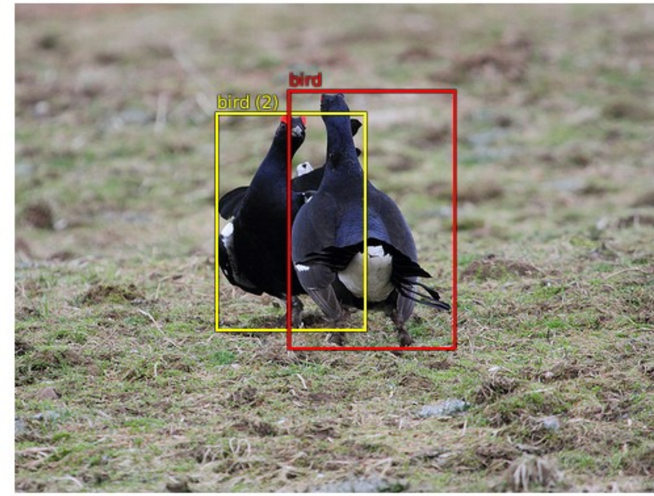

tench mismatch 0.17
450 "goldfish"
dims 3x256x256
451 "eel"
rock beauty mismatch 0.45
anemone fish mismatch 0.16
454 "lionfish"
dims 3x326x256

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?

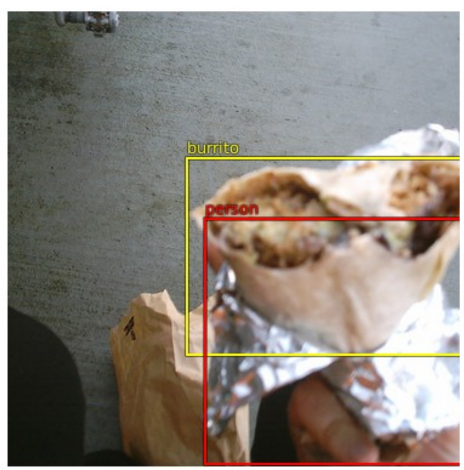




Top predictions:
bird (confidence 86.
bird (confidence 70.

ILSVRC2012_val_00001136.JPEG

Groundtruth:



Top predictions:
burrito (confidence 28.9)

ILSVRC2012_val_00000572.JPEG

Groundtruth:
person
burrito

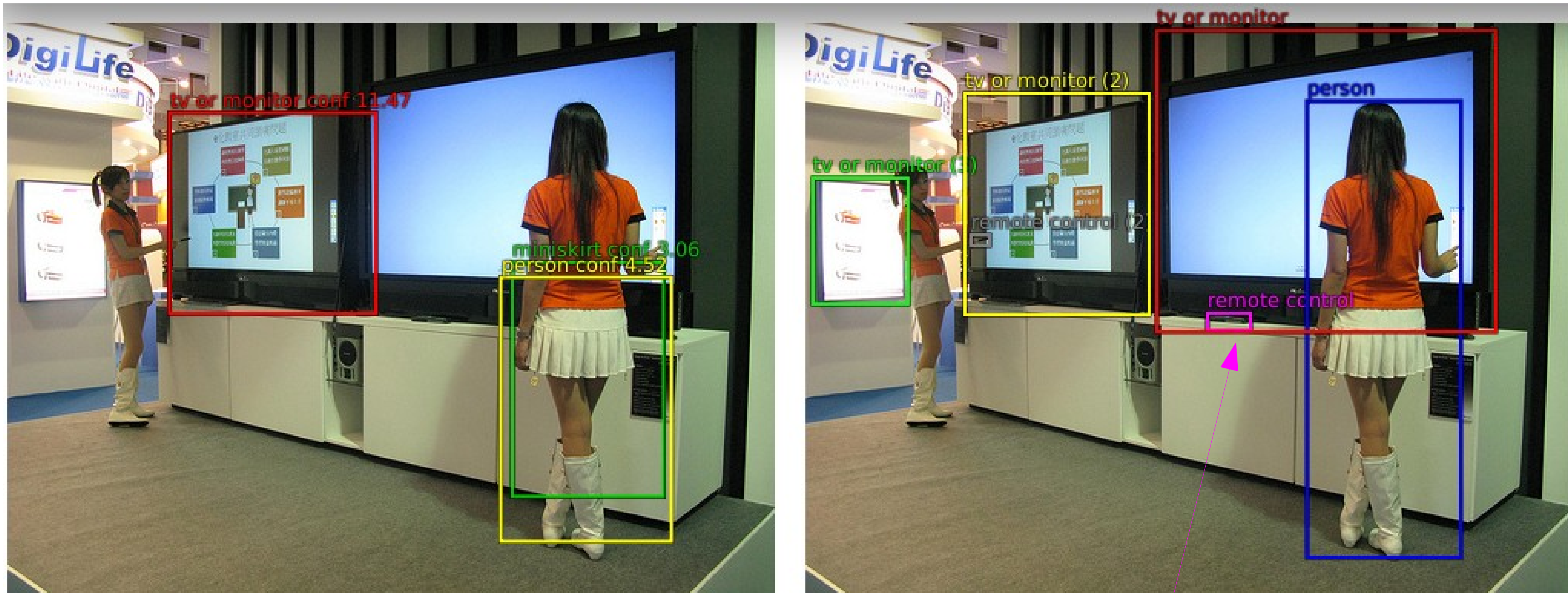
Top predictions:
burrito (confidence 17.4)

ILSVRC2012_val_00000606.JPEG

Groundtruth:
burrito
burrito (2)

Detection: Examples

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



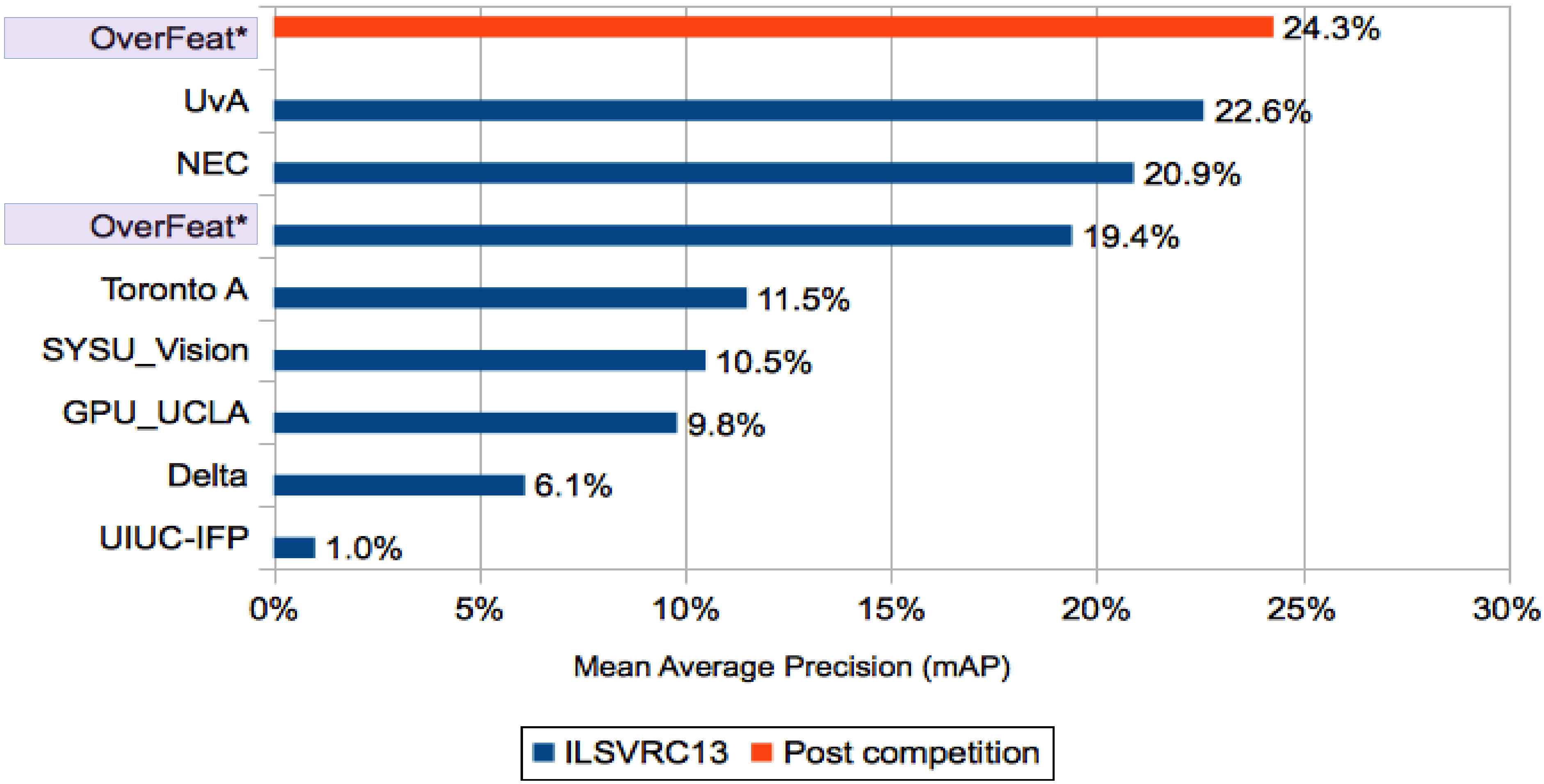
Top predictions:
tv or monitor (confidence 11.5)
person (confidence 4.5)
miniskirt (confidence 3.1)

ILSVRC2012_val_00000119.JPEG

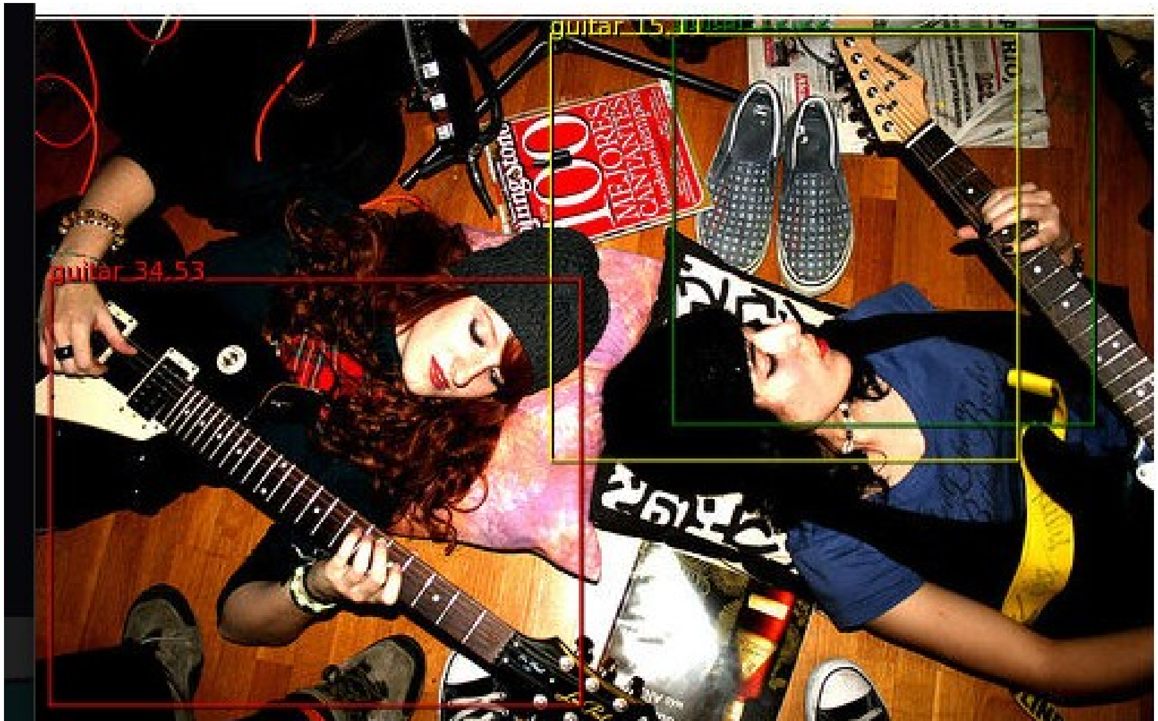
Groundtruth:
tv or monitor
tv or monitor (2)
tv or monitor (3)
person
remote control
remote control (2)

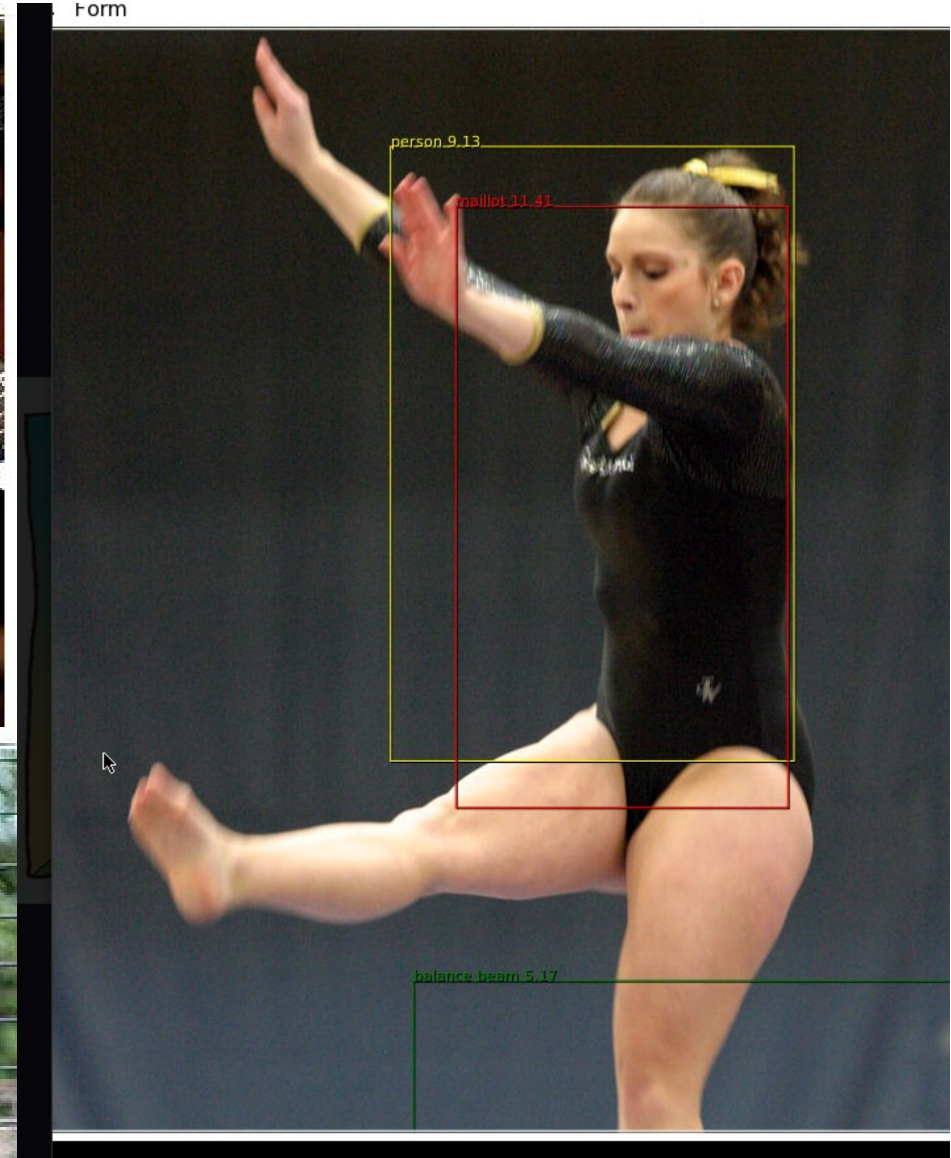
ImageNet 2013: Detection

- 200 categories. System must give 5 bounding boxes with categories
- OverFeat: pre-trained on ImageNet 1K, fine-tuned on ImageNet Detection.
- Post-deadline result: 0.243 mean average precision

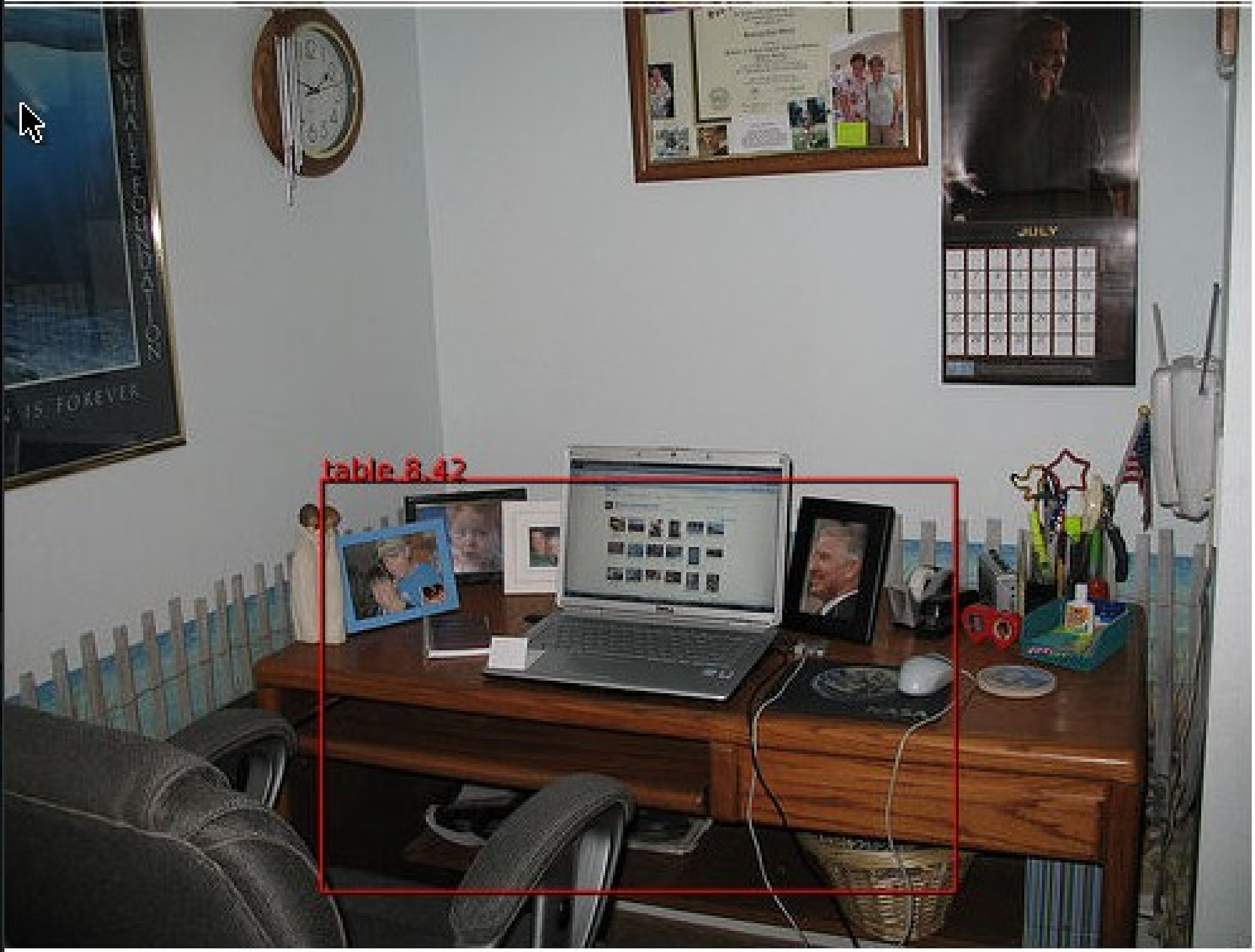


Results: pre-trained on ImageNet1K, fine-tuned on ImageNet Detection





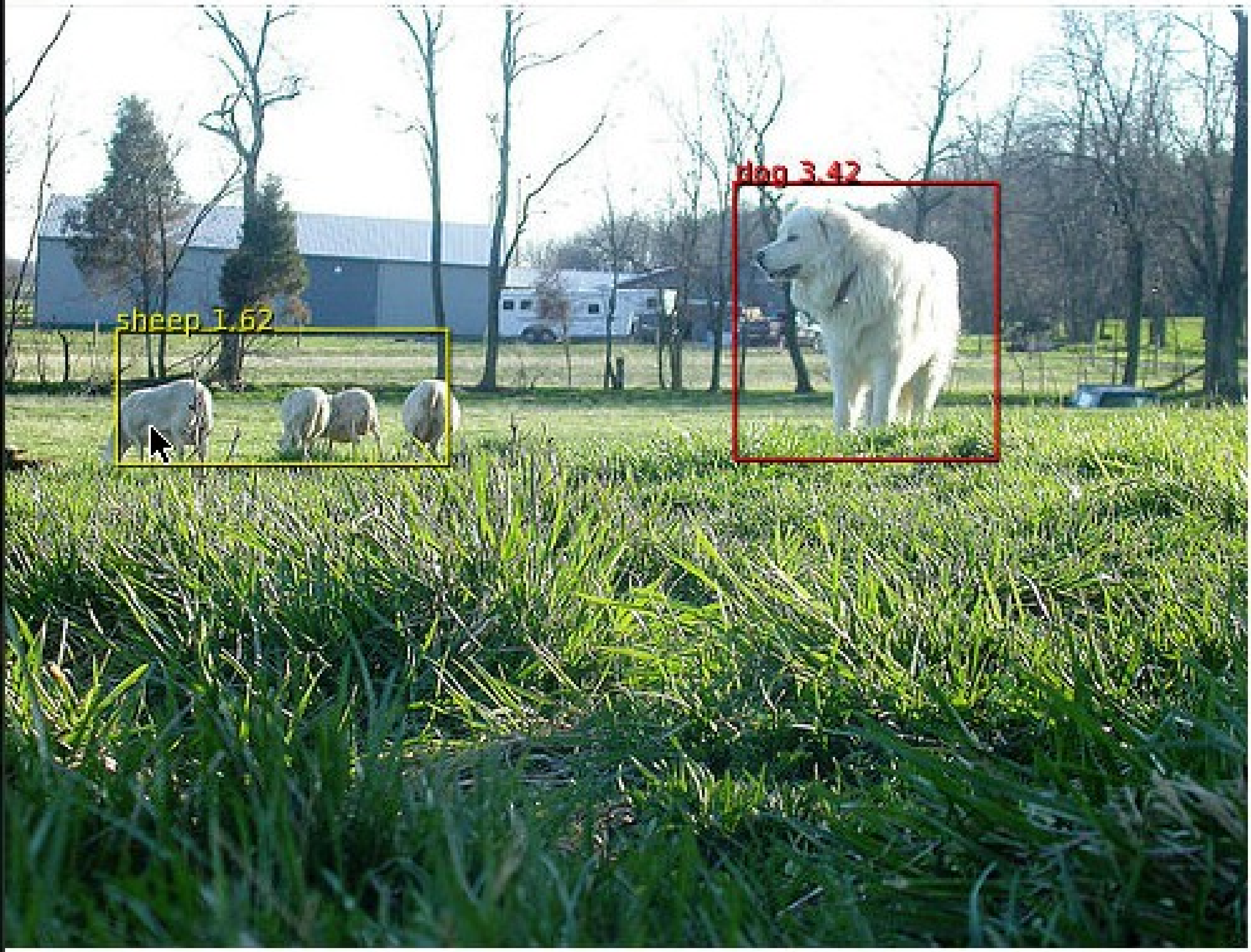
Detection Examples



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

Detection Examples

Detection Examples

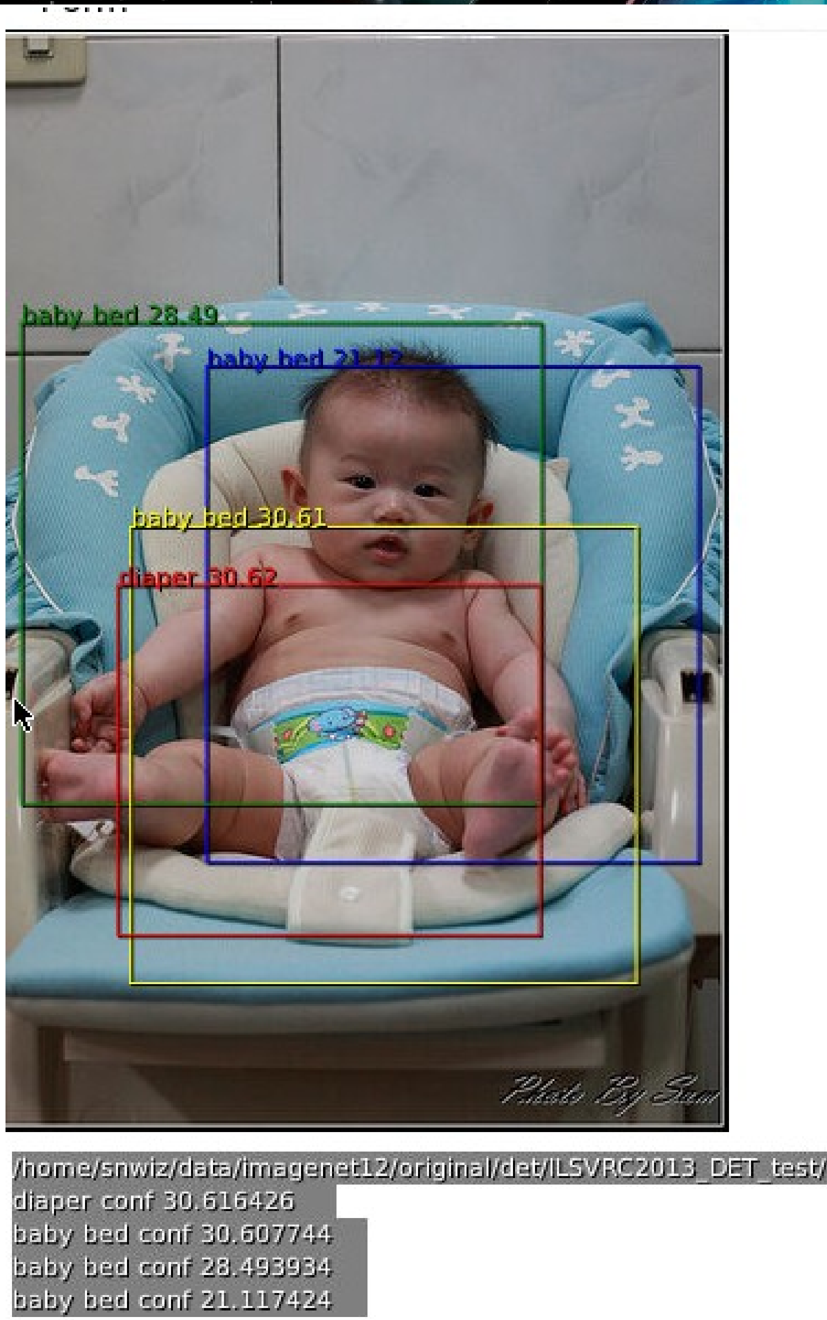


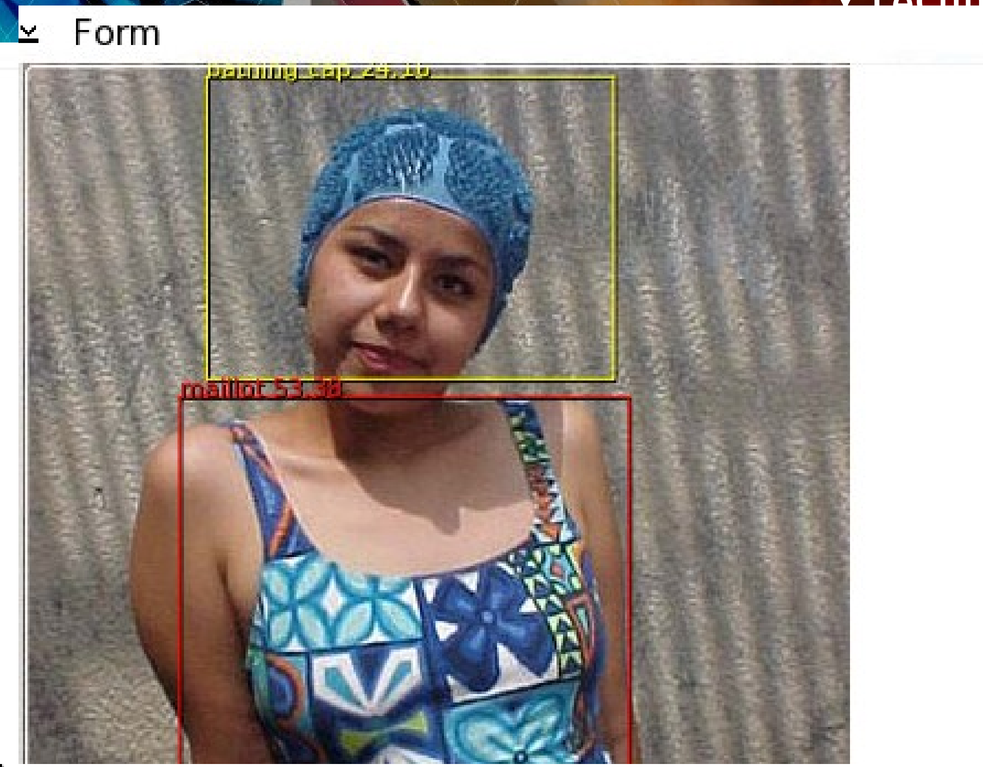
/home/snwiz/data/imagenet12/original/det/ILSVRC2013_DET_test/ILSVRC2012_test_00090628.JPEG
dog conf 3.419652
sheep conf 1.616341

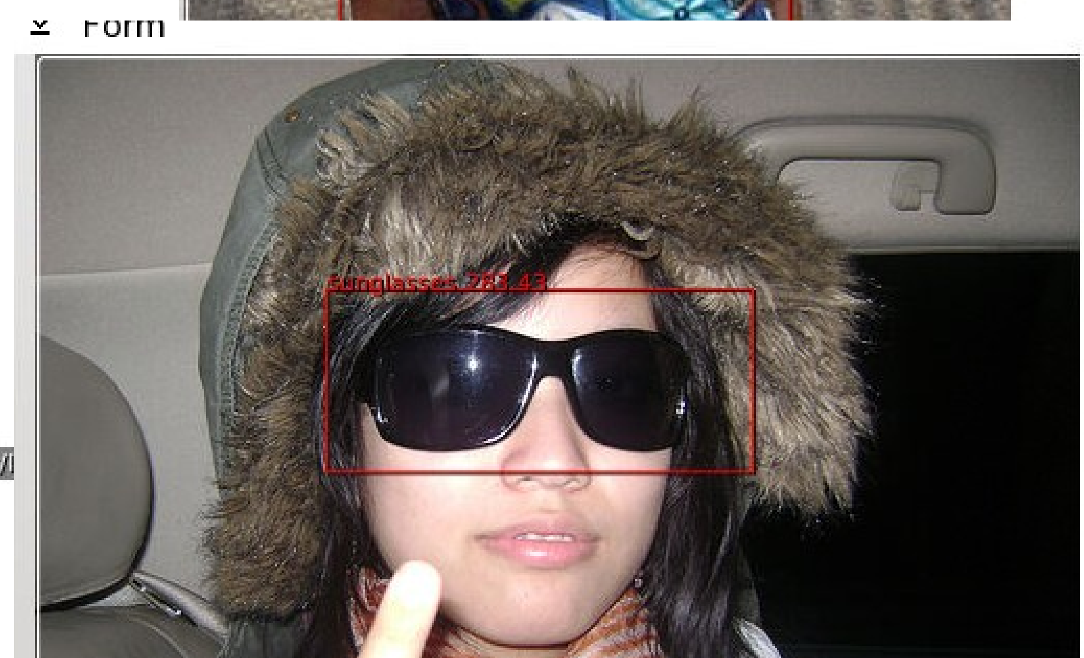
Detection Examples

/home/snwiz/data/imagenet12/original/det/ILSVRC2013_DET_test/ILSVRC2012_test_00091048.JPEG
person conf 17.898635
bow conf 15.628116

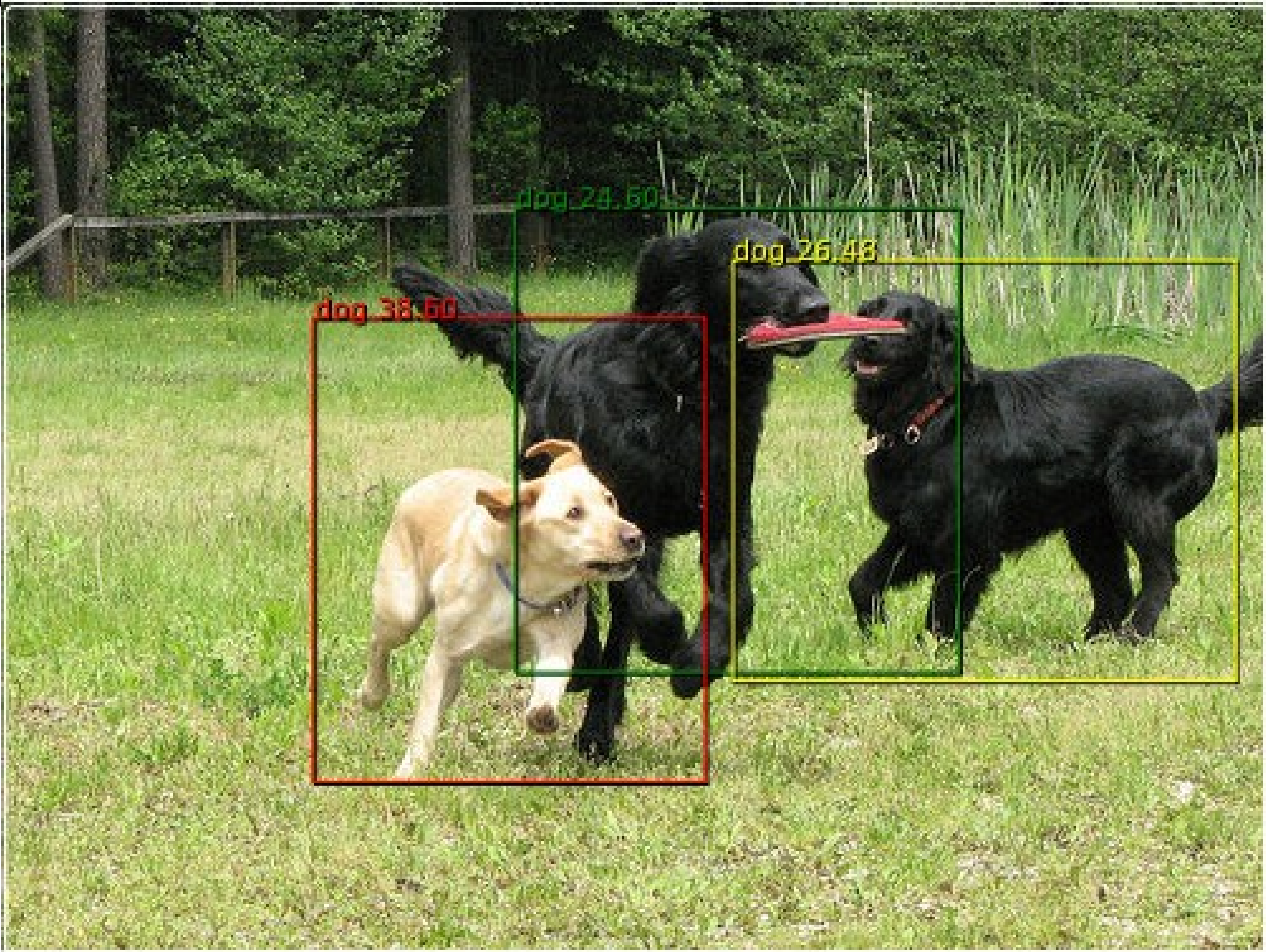
Detection Examples







Detection Examples



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

Detection Examples

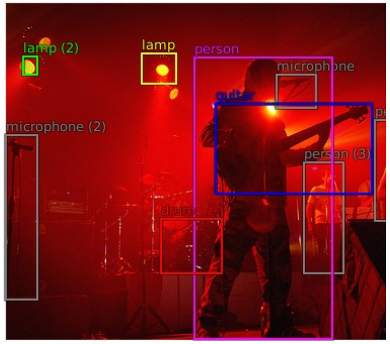
Detection: Difficult Examples

- Groundtruth is sometimes ambiguous or incomplete

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

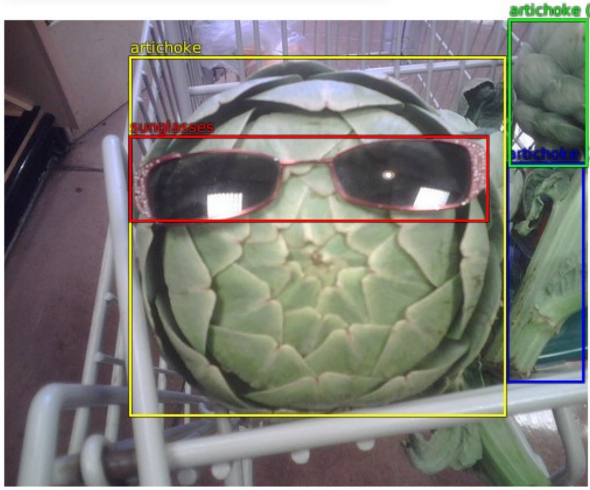
Groundtruth:
bowl
microwave

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)

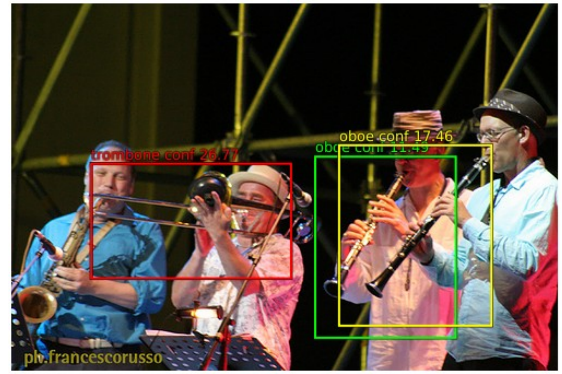
Top predictions:
artichoke (confidence 162.8)
ILSVRC2012_val_00001549.JPEG



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

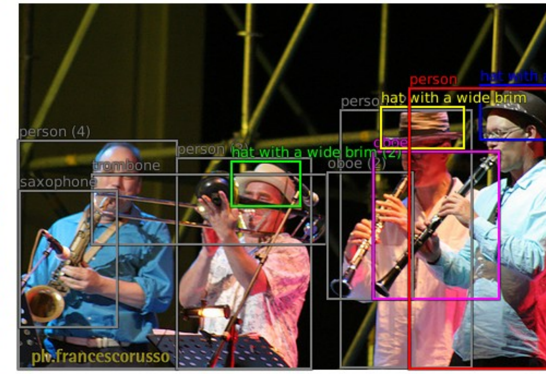
Detection: Difficult Examples

- Non-max suppression makes us miss many objects
  - Person behind instrument
- A bit of contextual post-processing would fix many errors

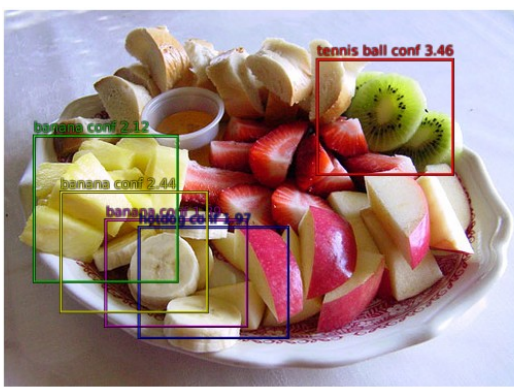


Top predictions:
trombone (confidence 26.8)
oboe (confidence 17.5)
oboe (confidence 11.5)

ILSVRC2012_val_00000614.JPEG

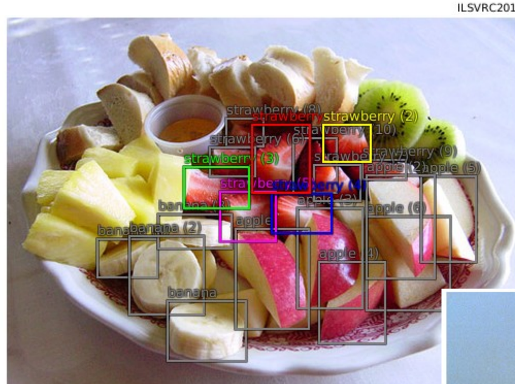


Groundtruth:
person
hat with a wide brim
hat with a wide brim (2)
hat with a wide brim (3)
oboe
oboe (2)
saxophone
trombone
person (2)
person (3)
person (4)

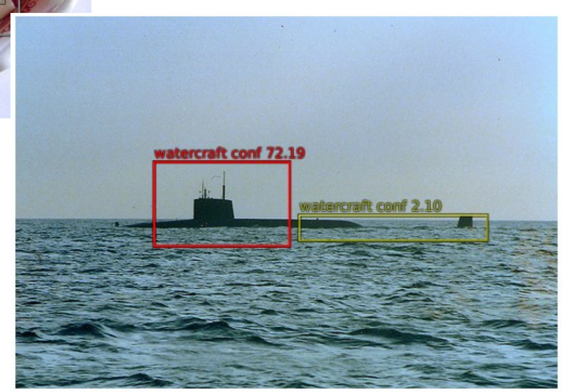


Top predictions:
tennis ball (confidence 3.5)
banana (confidence 2.4)
banana (confidence 2.1)
hotdog (confidence 2.0)
banana (confidence 1.9)

ILSVRC2012_val_00000320.JPEG

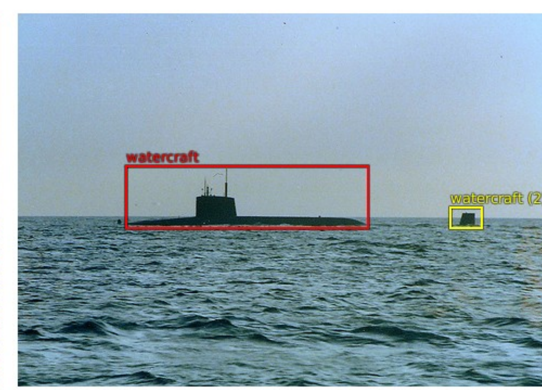


Groundtruth:
strawberry
strawberry (2)
strawberry (3)
strawberry (4)
strawberry (5)
strawberry (6)
strawberry (7)
strawberry (8)
strawberry (9)
strawberry (10)
apple
apple (2)
apple (3)



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

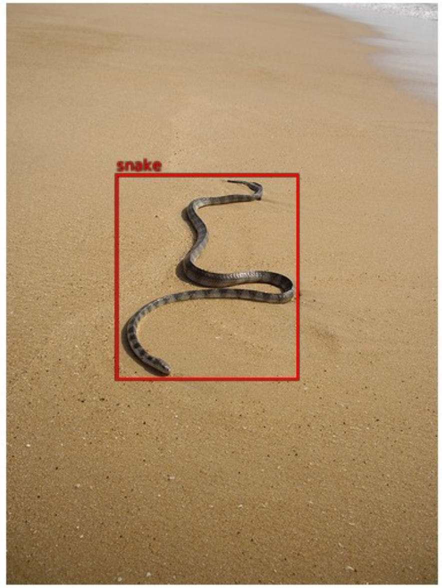
ILSVRC2012_val_00000623.JPEG



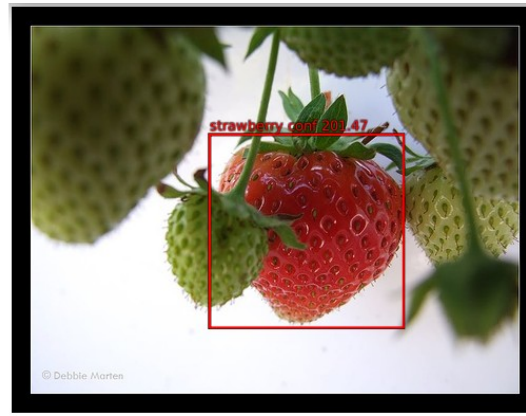
Groundtruth:
watercraft
watercraft (2)

Detection: Interesting Failures

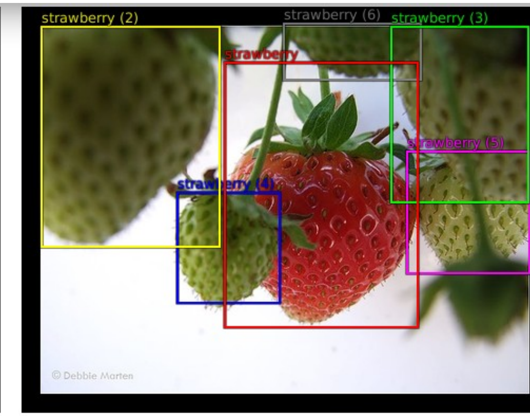
Top predictions:
corkscrew (confidence 38.1)
ILSVRC2012_val_00000324.JPEG



Groundtruth:
snake



Top predictions:
strawberry (confidence 201.5)
ILSVRC2012_val_00000099.JPEG



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

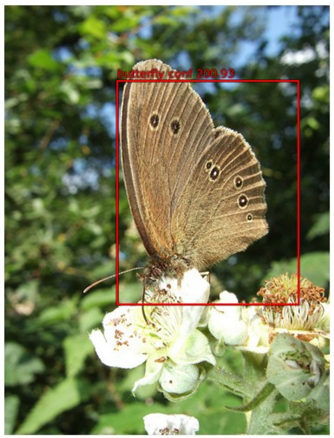
- Snake → Corkscrew

Top predictions:
remote control (confidence 31.8)
filing cabinet (confidence 2.2)
ILSVRC2012_val_00000331.JPEG

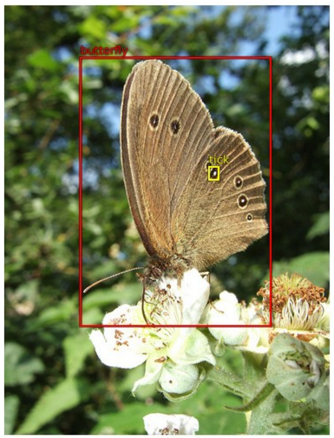
Groundtruth:
table
water bottle
water bottle (2)
water bottle (3)
water bottle (4)
refrigerator

Detection: Bad Groundtruth

One of the labelers likes ticks.....

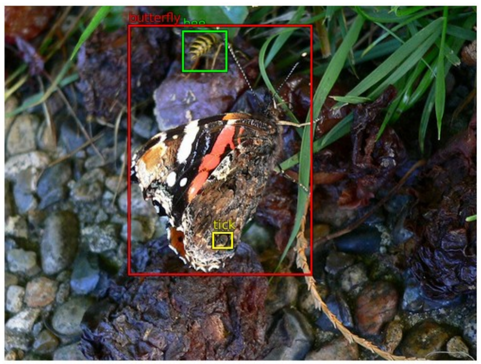


Top predictions:
butterfly (confidence 200.9)
ILSVRC2012_val_00035074.JPEG

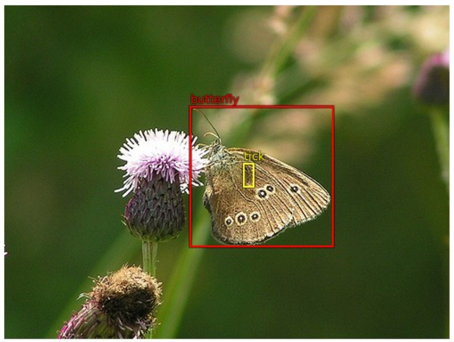


Groundtruth:
butterfly
tick

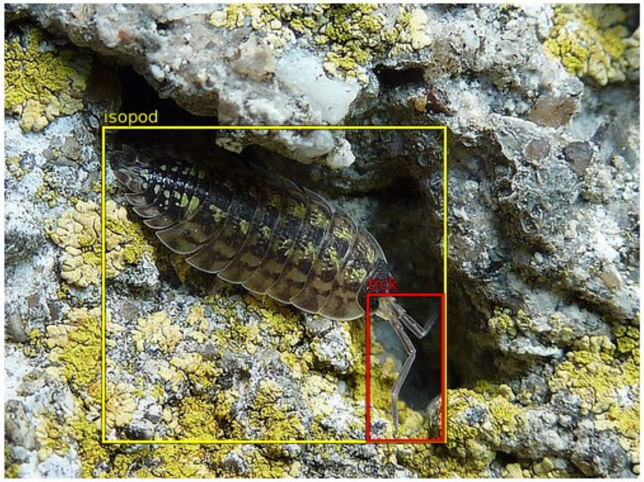
Top predictions:
butterfly (confidence 15.8)
ILSVRC2012_val_00012764.JPEG



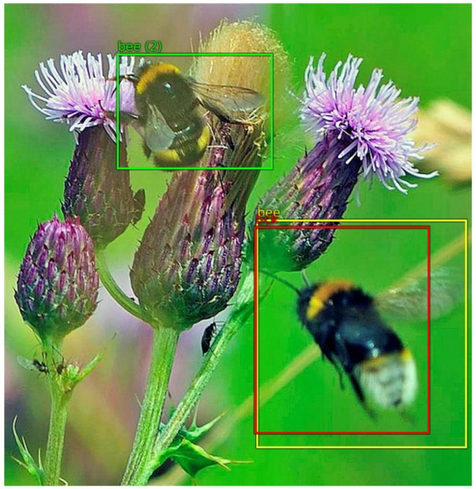
Groundtruth:
butterfly
tick
bee



Groundtruth:
butterfly
tick



Groundtruth:
tick
isopod



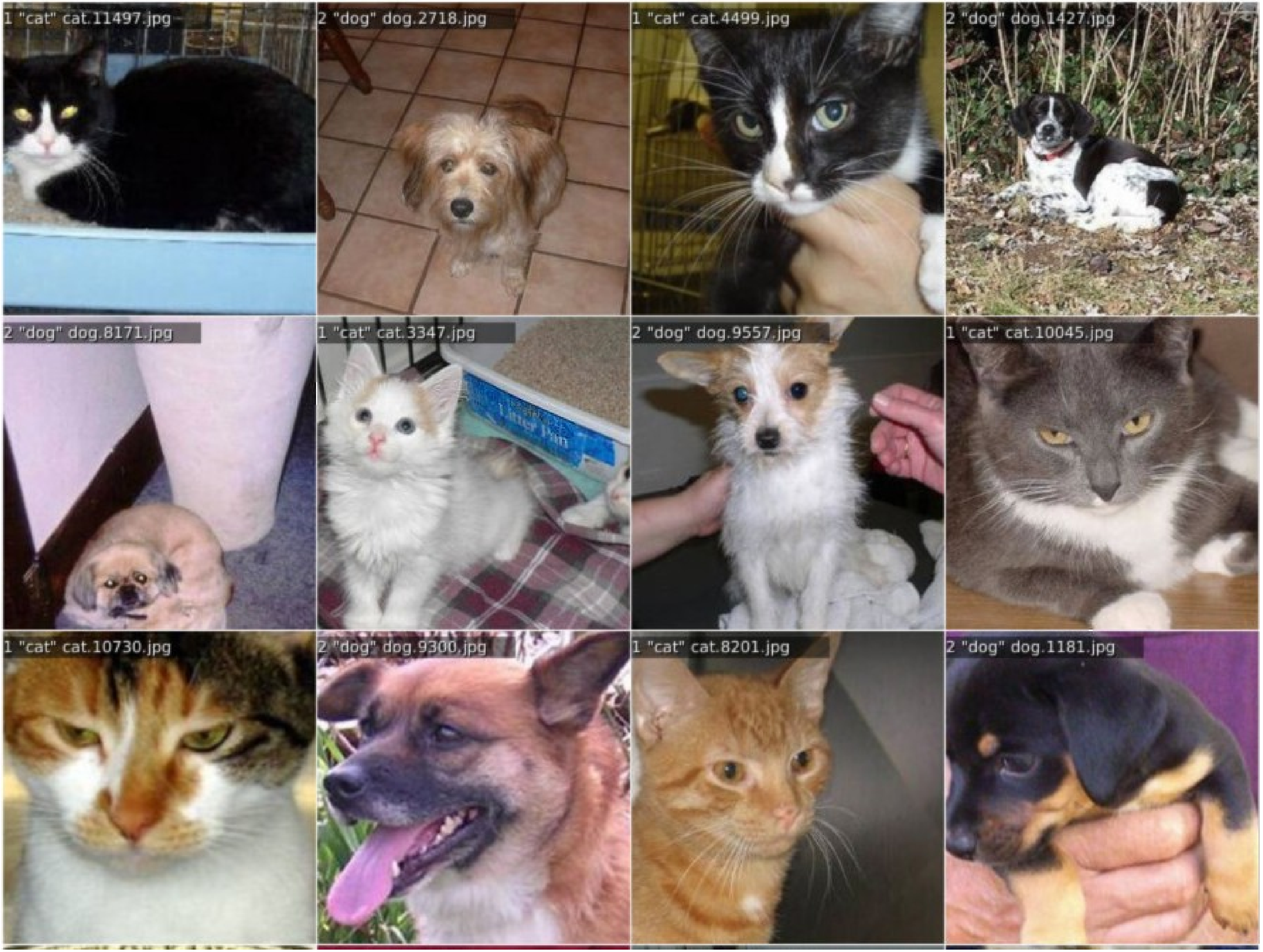
Groundtruth:
tick
bee
bee (2)

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

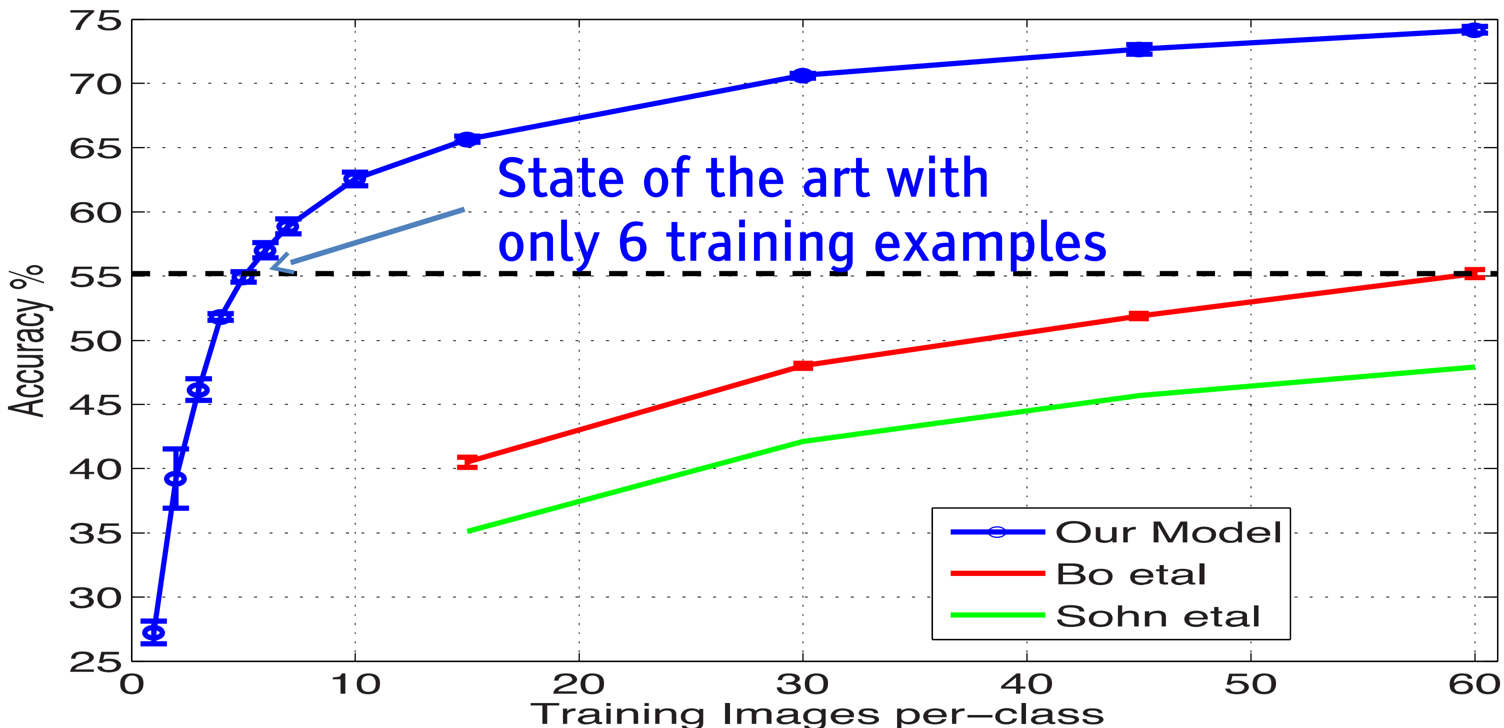
This competition has completed. This leaderboard reflects the final standings.

See someone using multiple accounts? Let us know.

#	Δ1w	Team Name * in the money	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

Features are generic: Caltech 256

- Network first trained on ImageNet.
- Last layer chopped off
- Last layer trained on Caltech 256,
- first layers N-1 kept fixed.
- State of the art accuracy with only 6 training samples/class



# Train	Acc % 15/class	Acc % 30/class	Acc % 45/class	Acc % 60/class
Sohn *et al.* [16]	35.1	42.1	45.7	47.9
Bo *et al.* [3]	40.5 ± 0.4	48.0 ± 0.2	51.9 ± 0.2	55.2 ± 0.3
Non-pretr.	9.0 ± 1.4	22.5 ± 0.7	31.2 ± 0.5	38.8 ± 1.4
ImageNet-pretr.	**65.7 ± 0.2**	**70.6 ± 0.2**	**72.7 ± 0.4**	**74.2 ± 0.3**

3: [Bo, Ren, Fox. CVPR, 2013]   16: [Sohn, Jung, Lee, Hero ICCV 2011]

OverFeat Features ->Trained Classifier on other datasets

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 Segmentatic

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 + Classifer on various datasets

	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	13.6 % error
	Dogs vs Cats Kaggle challenge 2014	**state of the art**	98.9%
• object localization	ImageNet LSVRC 2013	**state of the art**	29.9% error
• object detection	ImageNet LSVRC 2013	**state of the art**	24.3% mAP
[Razavian et al, 2014]: public OverFeat library (no retraining) + SVM **(simplest approach possible on purpose, no attempt at more complex classifiers)** (tasks are ordered by "distance" from classification task on which OverFeat was trained)			
• image classification	Pascal VOC 2007	competitive	73.9% mAP
• scene recognition	MIT-67	competitive	58.4% mAP
• fine grained recognition	Caltech-UCSD Birds 200-2011	competitive	53.3% mAP
	Oxford 102 Flowers	competitive	74.70% mAP
• attribute detection	UIUC 64 object attributes	**state of the art**	89.0% mAUC
	H3D Human Attributes	**state of the art**	70.78% mAP
• image retrieval (search by image similarity)	Oxford 5k buildings	?	0.52
	Paris 6k buildings	?	0.676
	Sculp6k	?	0.269
	Holidays	competitive	0.646
	UKBench	relatively poor	3.05

Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, Yann LeCun, **OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks**, http://arxiv.org/abs/1312.6229, ICLR 2014

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, Stefan Carlsson, **CNN Features off-the-shelf: an Astounding Baseline for Recognition**, http://arxiv.org/abs/1403.6382, DeepVision CVPR 2014 workshop

Other ConvNet Results

	Dataset	Performance	Score
[Zeiler et al 2013]			
• image classification	ImageNet LSVRC 2013	**state of the art**	11.2% error
	Caltech-101 (15, 30 samples per class)	competitive	83.8%, 86.5%
	Caltech-256 (15, 60 samples per class)	**state of the art**	65.7%, 74.2%
	Pascal VOC 2012	competitive	79% mAP
[Donahue et al, 2014]: DeCAF+SVM			
• image classification	Caltech-101 (30 classes)	**state of the art**	86.91%
• domain adaptation	Amazon -> Webcam, DSLR -> Webcam	**state of the art**	82.1%, 94.8%
• fine grained recognition	Caltech-UCSD Birds 200-2011	**state of the art**	65.0%
• scene recognition	SUN-397	competitive	40.9%
[Girshick et al, 2013]			
• image detection	Pascal VOC 2007	**state of the art**	48.0% mAP
	Pascal VOC 2010 (comp4)	**state of the art**	43.5% mAP
• image segmentation	Pascal VOC 2011 (comp6)	**state of the art**	47.9% mAP
[Oquab et al, 2013]			
• image classification	Pascal VOC 2007	**state of the art**	77.7% mAP
	Pascal VOC 2012	**state of the art**	82.8% mAP
	Pascal VOC 2012 (action classification)	**state of the art**	70.2% mAP

M.D. Zeiler, R. Fergus, **Visualizing and Understanding Convolutional Networks,** Arxiv 1311.2901 http://arxiv.org/abs/1311.2901

J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. **Decaf: A deep convolutional activation feature for generic visual recognition.** In ICML, 2014, http://arxiv.org/abs/1310.1531

R. B. Girshick, J. Donahue, T. Darrell, and J. Malik. **Rich feature hierarchies for accurate object detection and semantic segmentation**. arxiv:1311.2524 [cs.CV], 2013, http://arxiv.org/abs/1311.2524

M. Oquab, L. Bottou, I. Laptev, and J. Sivic. **Learning and transferring mid-level image representations using convolutional neural networks.** Technical Report HAL-00911179, INRIA, 2013. http://hal.inria.fr/hal-00911179

Other ConvNet Results

	Dataset	Performance	Score
[Khan et al 2014] • shadow detection	UCF CMU UIUC	**state of the art** **state of the art** **state of the art**	90.56% 88.79% 93.16%
[Sander Dieleman, 2014] • image attributes	Kaggle Galaxy Zoo challenge	**state of the art**	0.07492

S. H. Khan, M. Bennamoun, F. Sohel, R. Togneri. **Automatic Feature Learning for Robust Shadow Detection,** CVPR 2014
Sander Dieleman, Kaggle Galaxy Zoo challenge 2014 http://benanne.github.io/2014/04/05/galaxy-zoo.html

- **Results compiled by Pierre Sermanet**
  - http://cs.nyu.edu/~sermanet/papers/Deep_ConvNets_for_Vision-Results.pdf

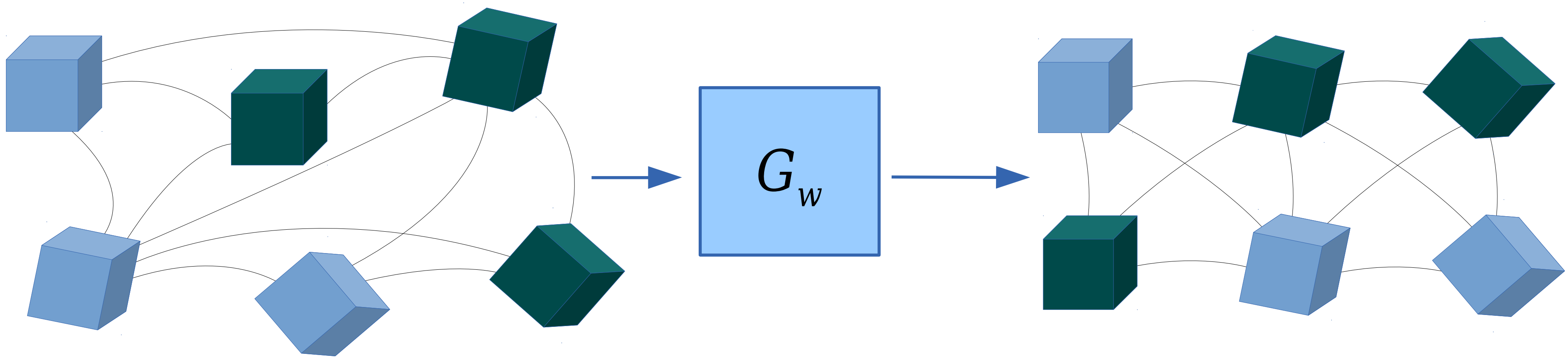
Image Similarity Matching With Siamese Networks Embedding, DrLIM

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

DrLIM: Contrative Loss function

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.

- Loss function for inputs X_1 and X_2 with binary label Y and $D_W = ||G_W(X_1) - G_W(X_2)||_2$:

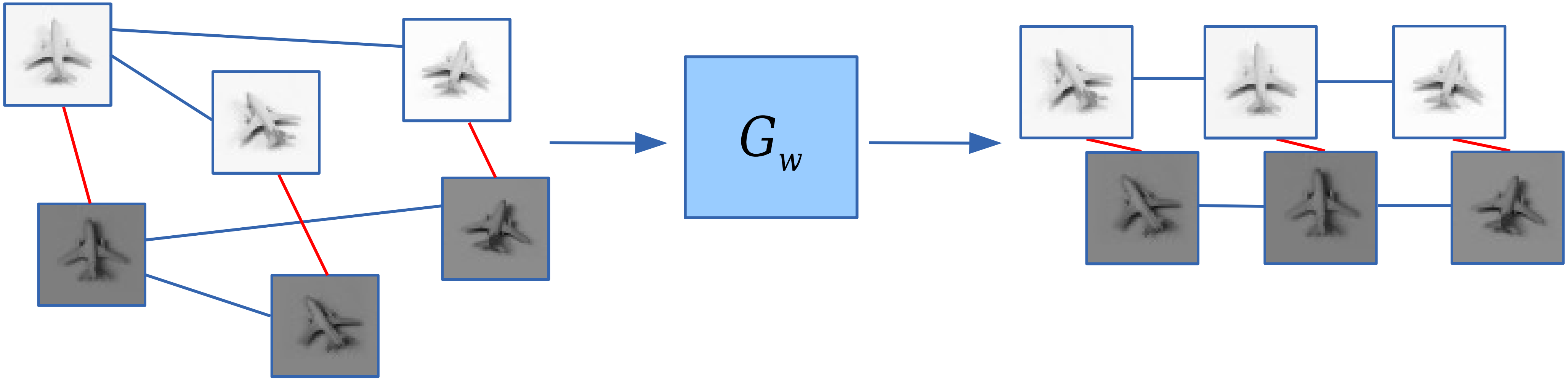
$$L(W, Y, \vec{X_1}, \vec{X_2}) = (1 - Y)\frac{1}{2}(D_W)^2 + (Y)\frac{1}{2}\{max(0, m - D_W)\}^2$$

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

DrLIM: Contrative Loss function

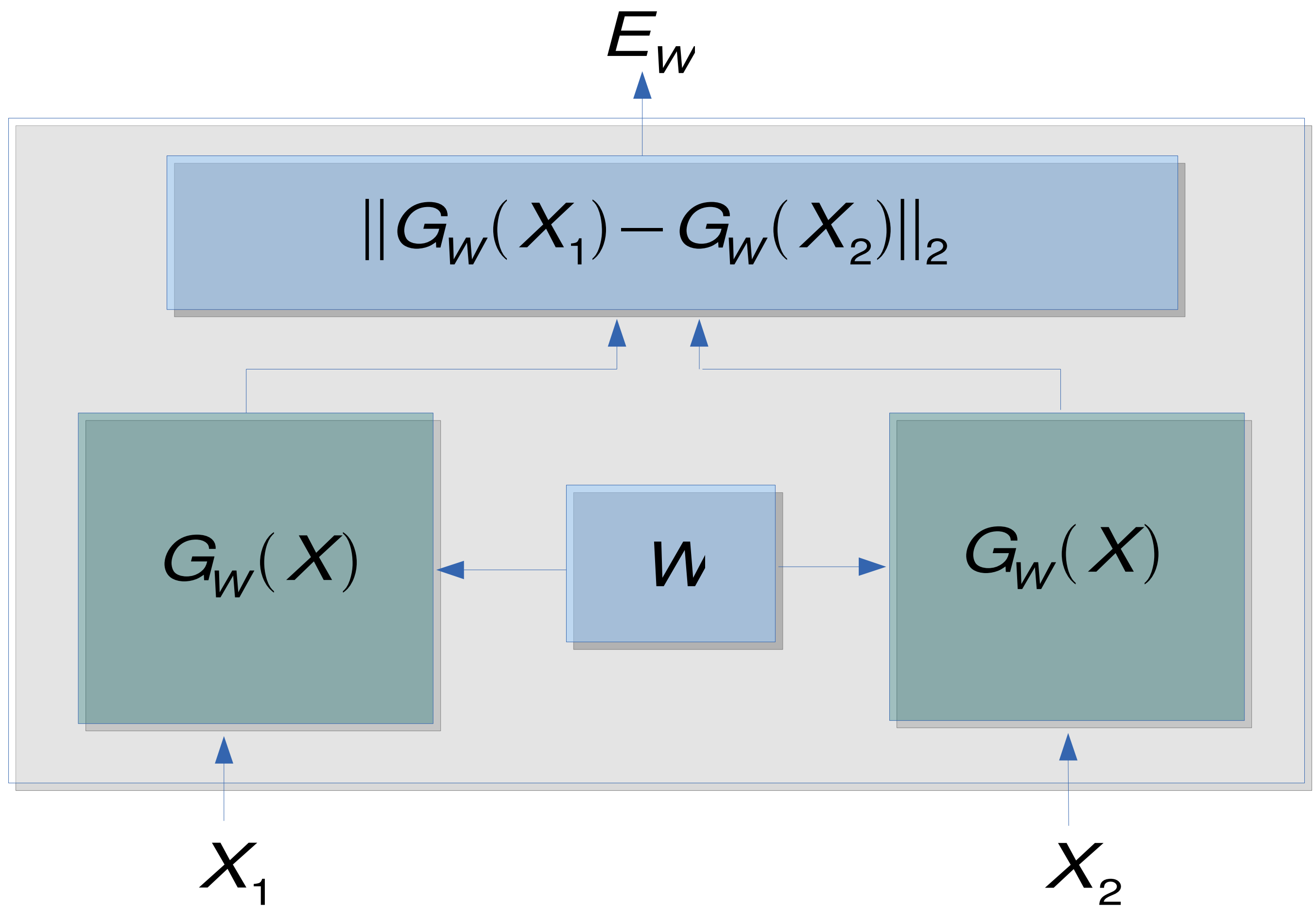
Dimensionality Reduction by Learning an Invariant Mapping

- **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

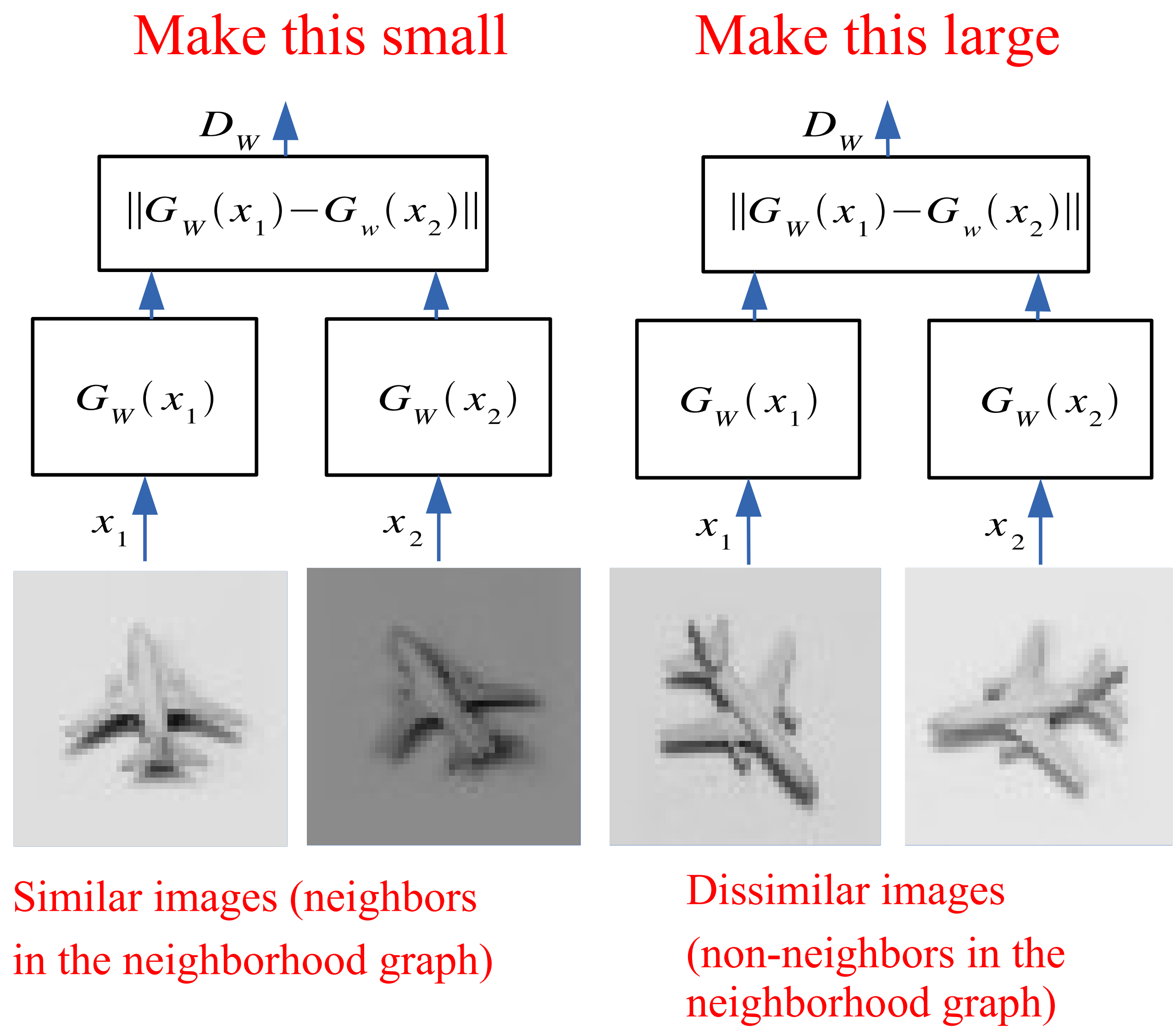


Siamese Architecture [Bromley, Sackinger, Shah, LeCun 1994]

Siamese Architecture and loss function

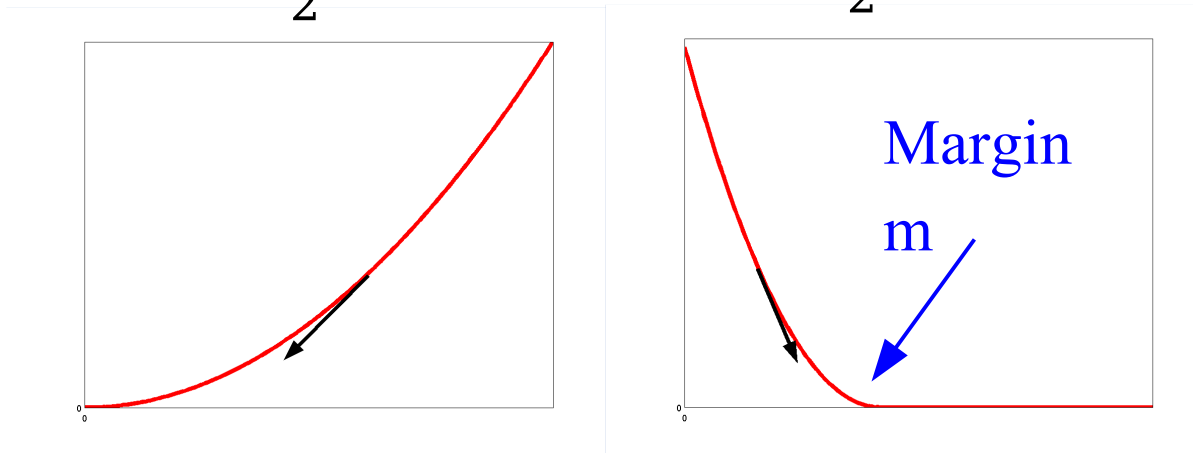
Loss function:

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



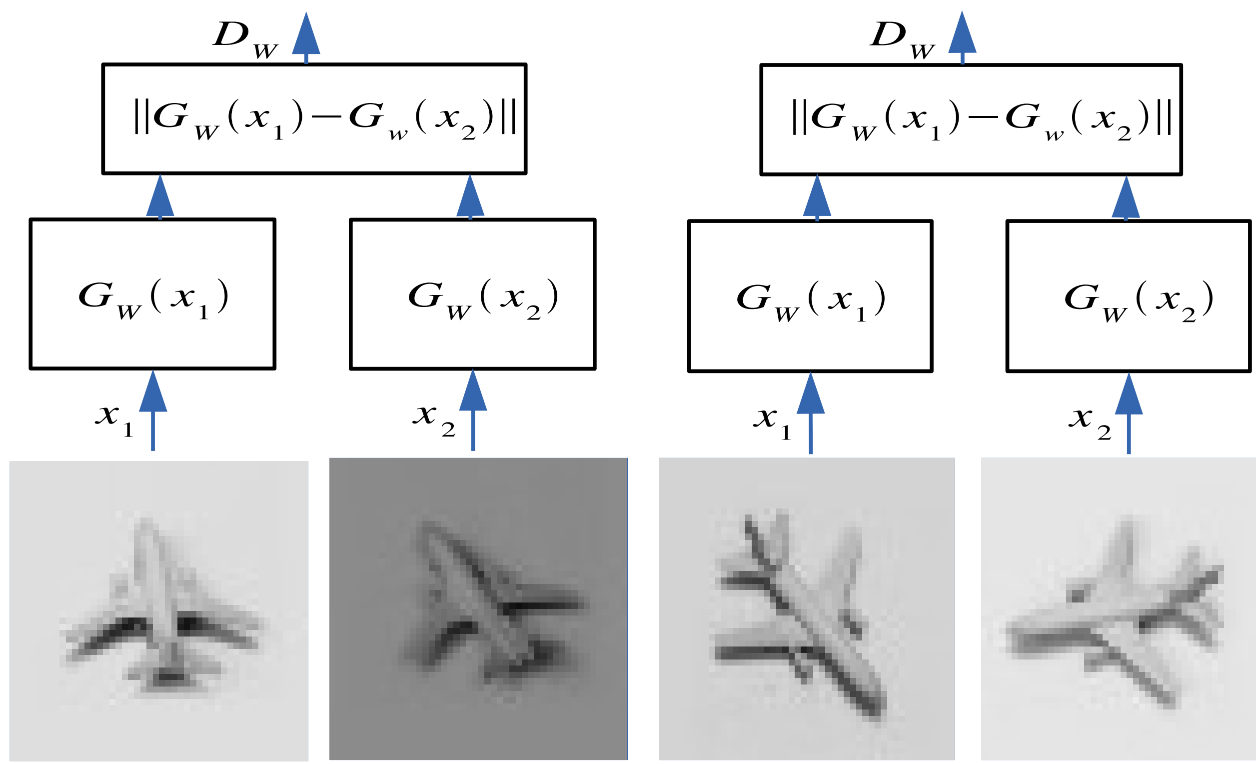
Loss function

$$L_{similar} = \frac{1}{2} D_W^2 \qquad L_{dissimilar} = \frac{1}{2} \{max(0, m - D_W)\}^2$$



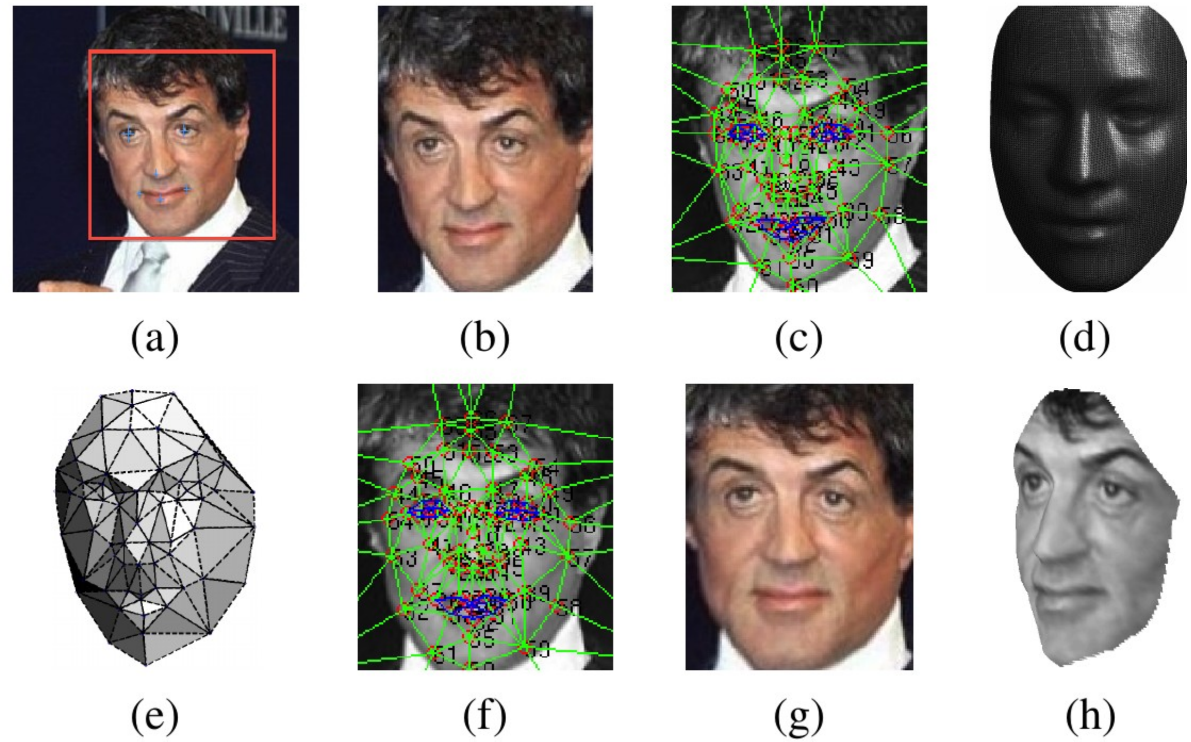
Loss function:

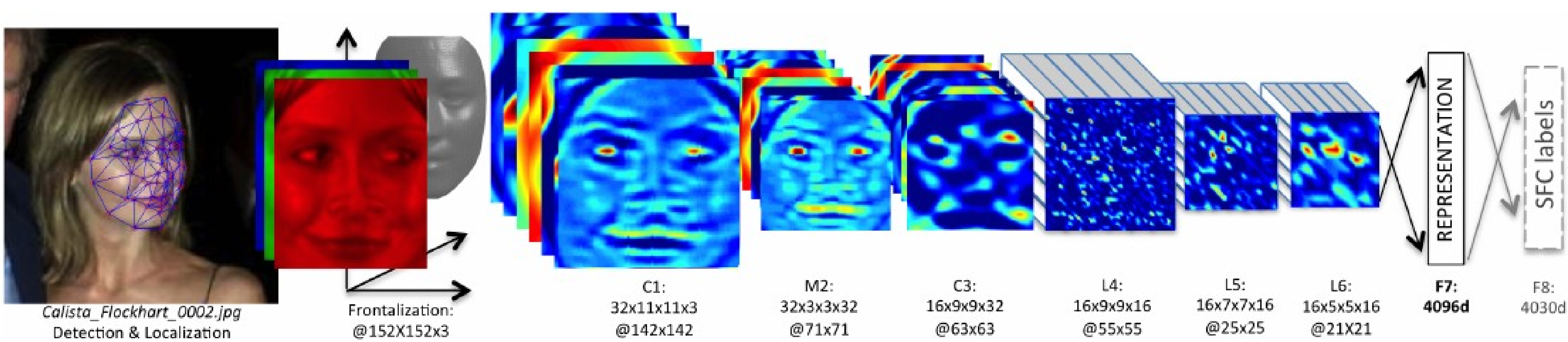
- Pay quadratically for making outputs of neighbors far apart
- Pay quadratically for making outputs of non-neighbors smaller than a margin m



Face Recognition:DeepFace (Facebook AI Research)

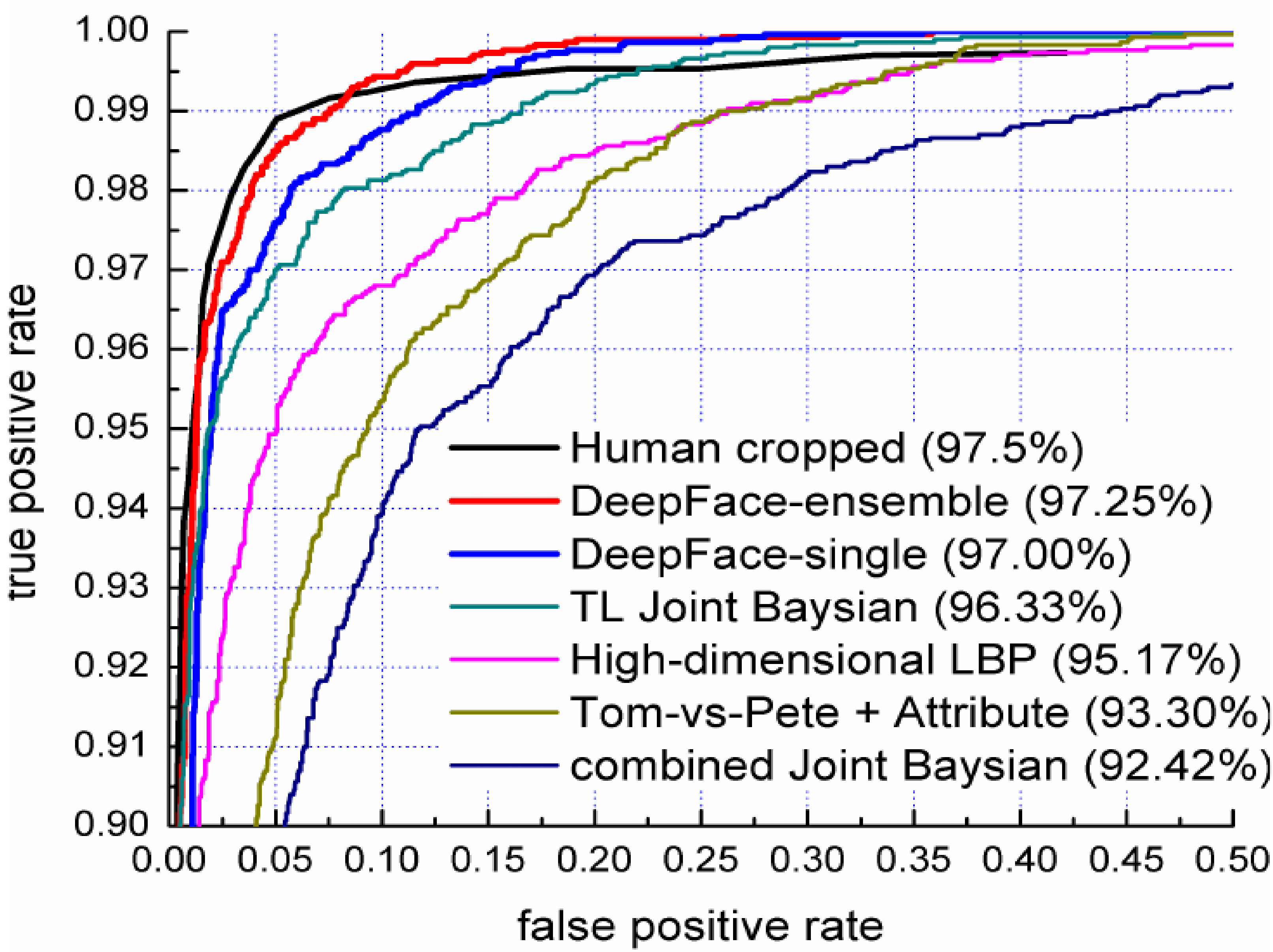
- [Taigman et al. CVPR 2014]
  - Alignment
  - Convnet





Face Recognition:DeepFace (Facebook AI Research)

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 (*SFC*)	Accuracy (*LFW*)
DeepFace-gradient	8.9%	0.9582 ±0.0118
DeepFace-align2D	9.5%	0.9430 ±0.0136
DeepFace-Siamese	NA	0.9617 ±0.0120

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

Depth Estimation from Stereo Pairs

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

Record holder on KITTI dataset (Sept 2014):

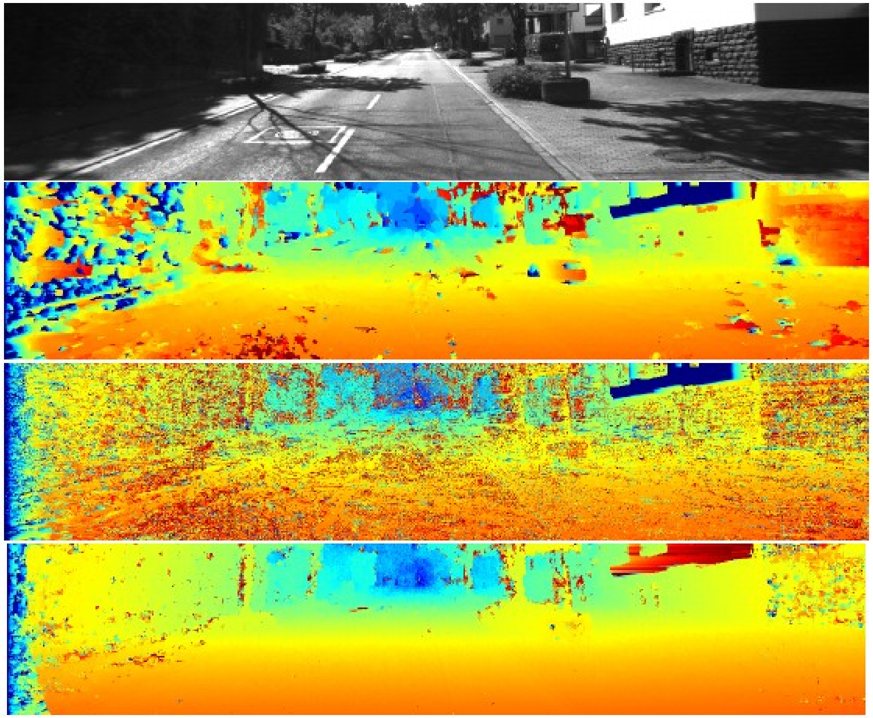
Rank	Method		Error
1	Our method		2.61 %
2	SPS-StFl	Yamaguchi et al. (2014)	2.83 %
3	VC-SF	Vogel et al. (2014)	3.05 %
4	PCBP-SS	Yamaguchi et al. (2013)	3.40 %
5	DDS-SS		3.83 %

Original image

SADD

Census

ConvNet



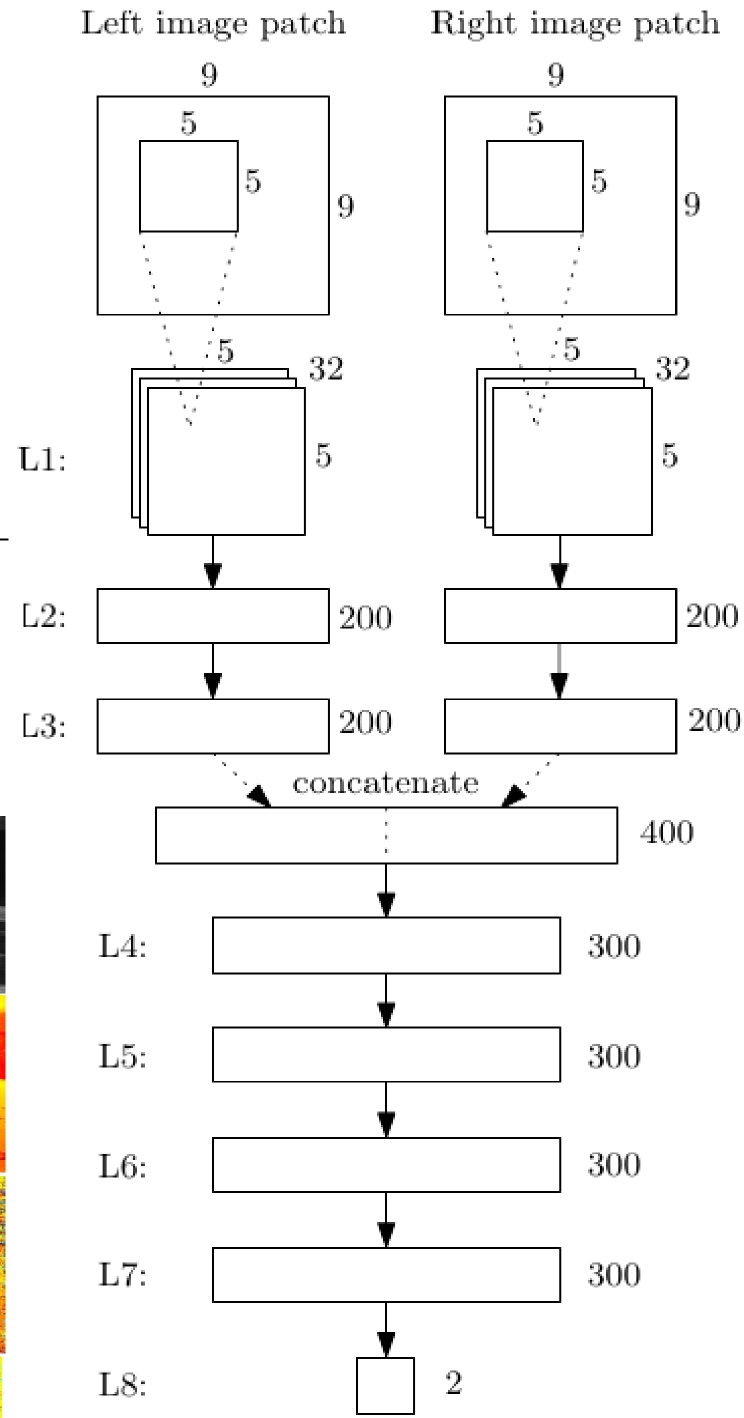
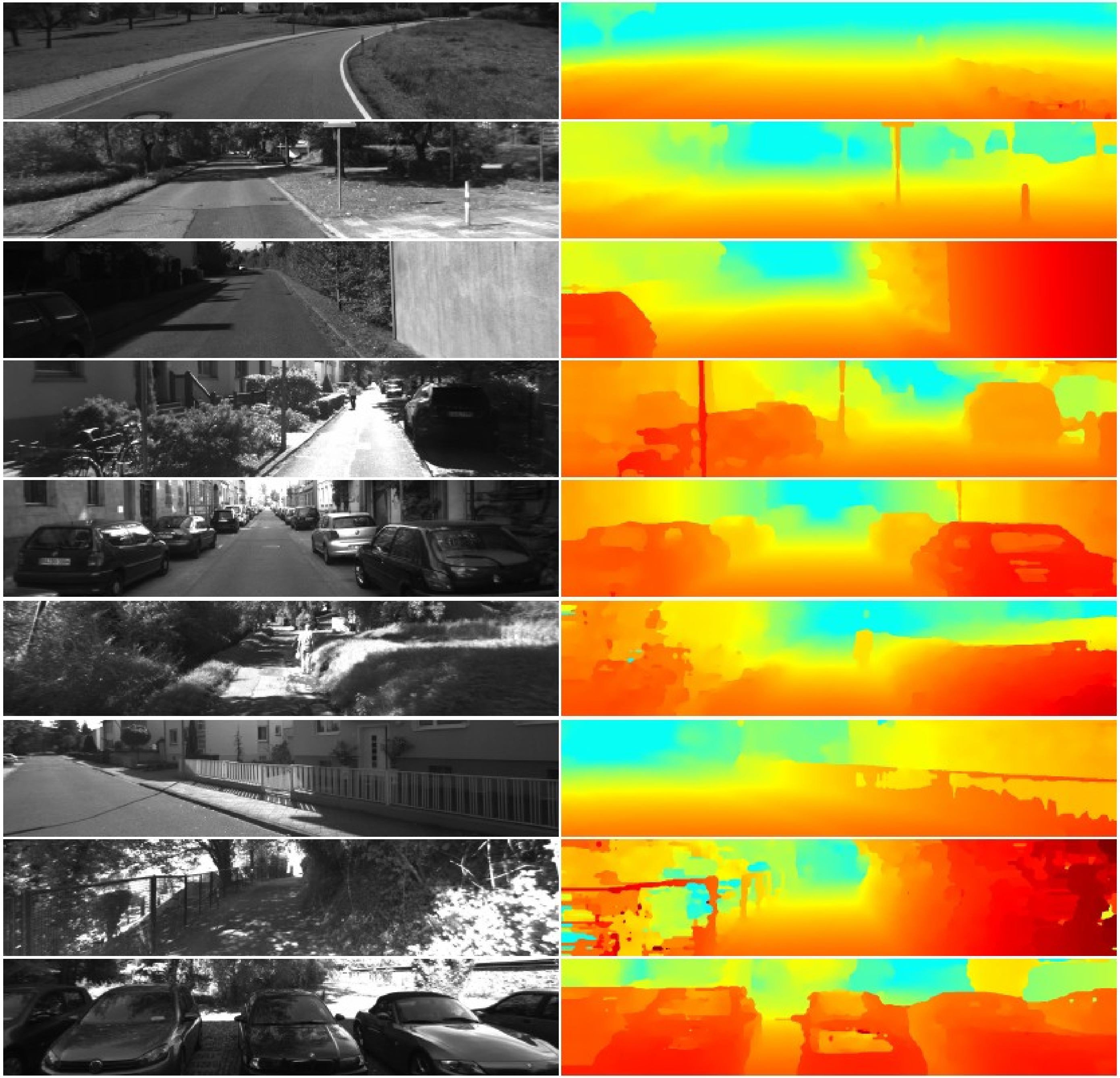


Figure 2: Network architecture

Depth Estimation from Stereo Pairs: Results

[Zbontar & LeCun Arxiv '14]

Presentation at ECCV workshop Saturday 9/6



Body Pose Estimation

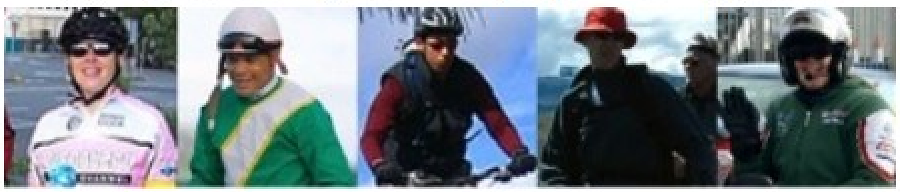
Pose Estimation and Attribute Recovery with ConvNets

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

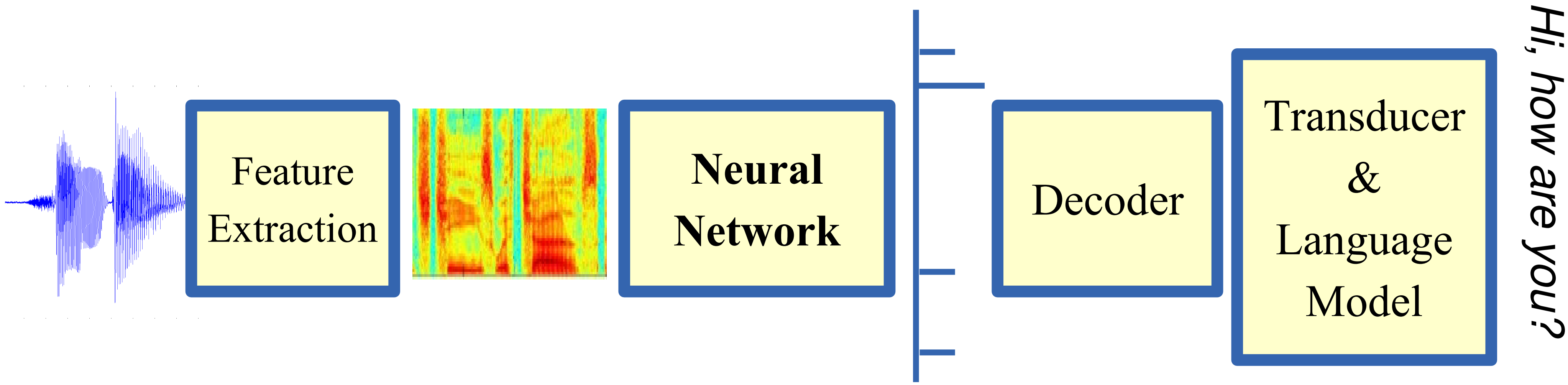
- Handwriting recognition MNIST (many), Arabic HWX (IDSIA)
- OCR in the Wild [2011]: StreetView House Numbers (NYU and others)
- Traffic sign recognition [2011] GTSRB competition (IDSIA, NYU)
- Asian handwriting recognition [2013] ICDAR competition (IDSIA)
- Pedestrian Detection [2013]: INRIA datasets 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)

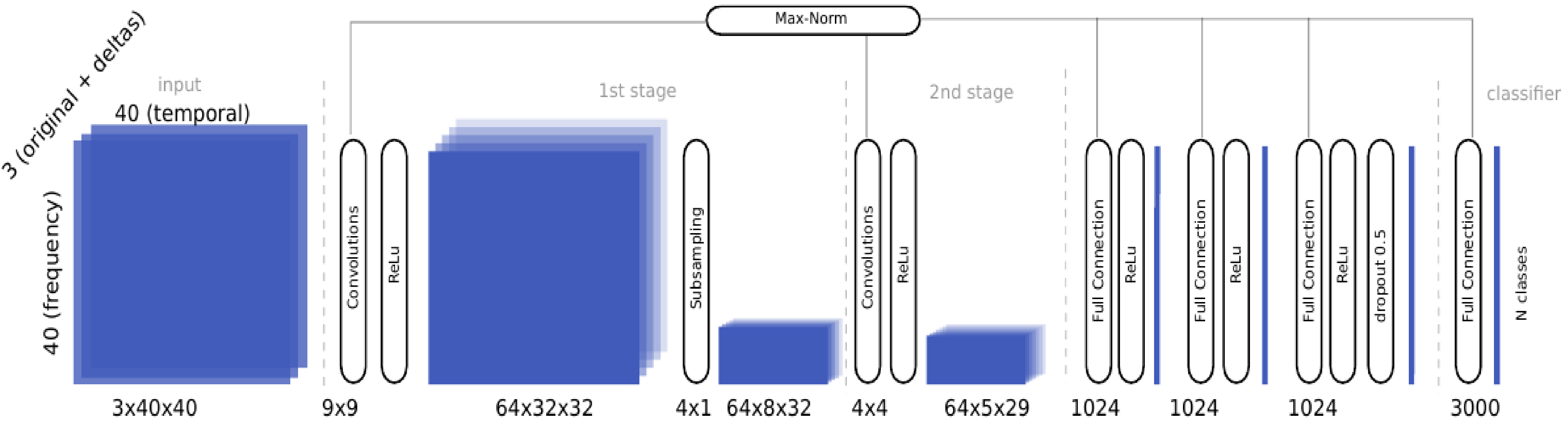
- **A typical speech recognition architecture with DL-based acoustic modeling**
  - Features: log energy of a filter bank (e.g. 40 filters)
  - Neural net acoustic modeling (convolutional or not)
  - Input window: typically 10 to 40 acoustic frames
  - Fully-connected neural net: 10 layers, 2000-4000 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)

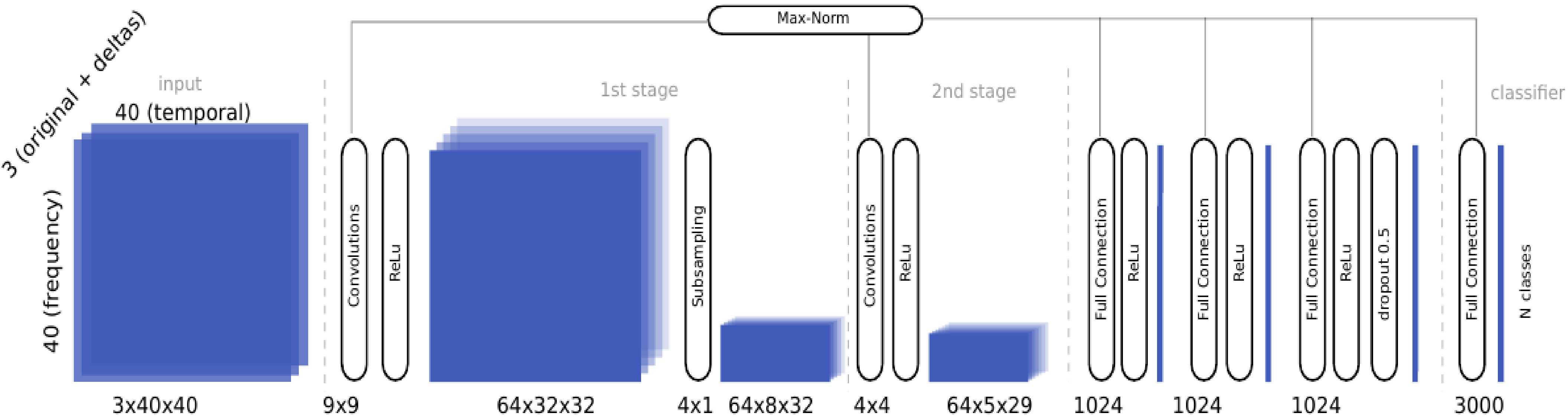
Y LeCun



- 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

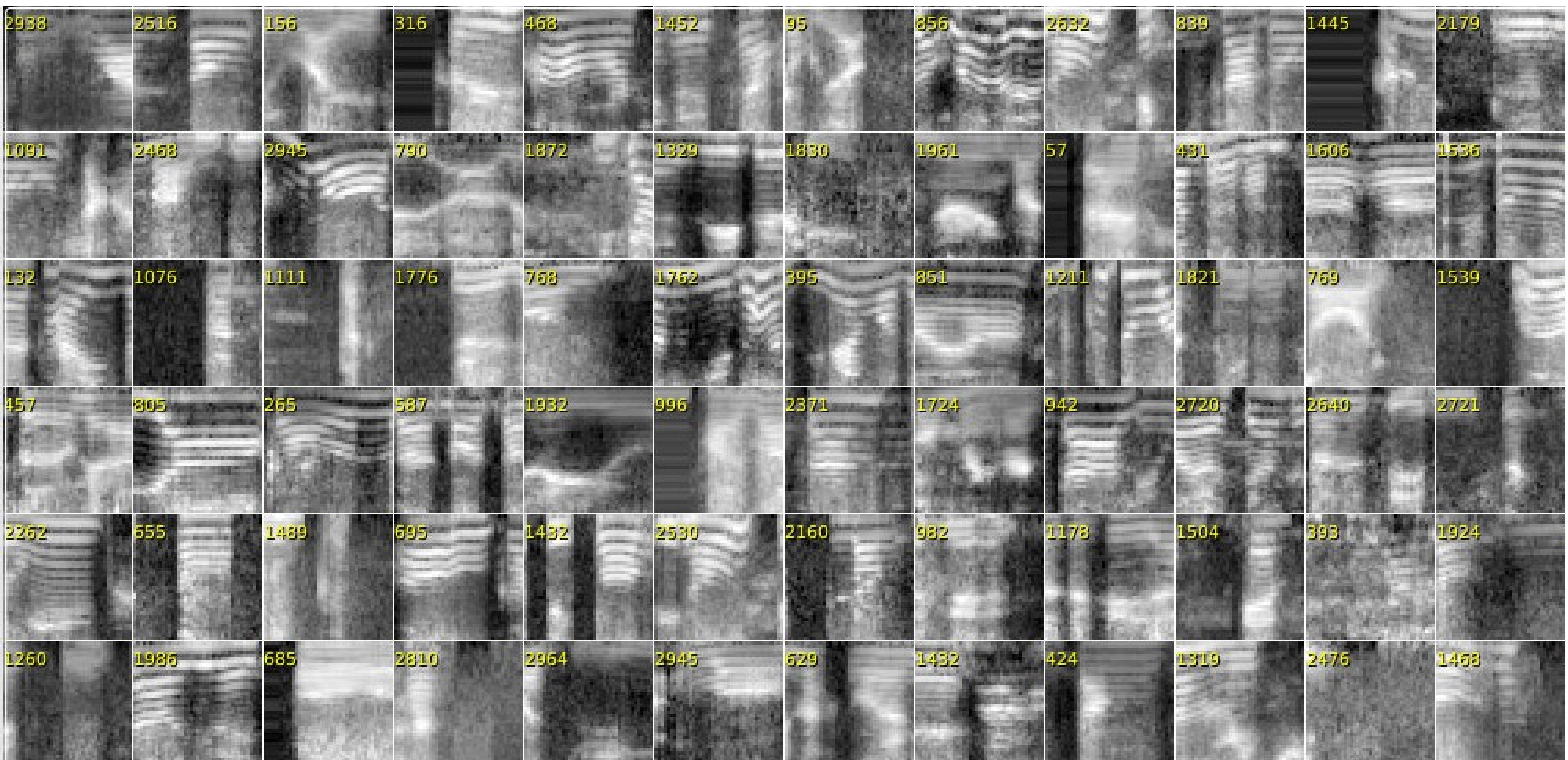


- **Subphone-level classification error (sept 2013):**
  - Cantonese: phone: 20.4% error; subphone: 33.6% error (IBM DNN: 37.8%)
- **Subphone-level classification error (march 2013)**
  - Cantonese: subphone: 36.91%
  - Vietnamese: subphone 48.54%
  - Full system performance (token error rate on conversational speech):
    - 76.2% (52.9% substitution, 13.0% deletion, 10.2% insertion)

Speech Recognition with Convolutional Nets (NYU/IBM)

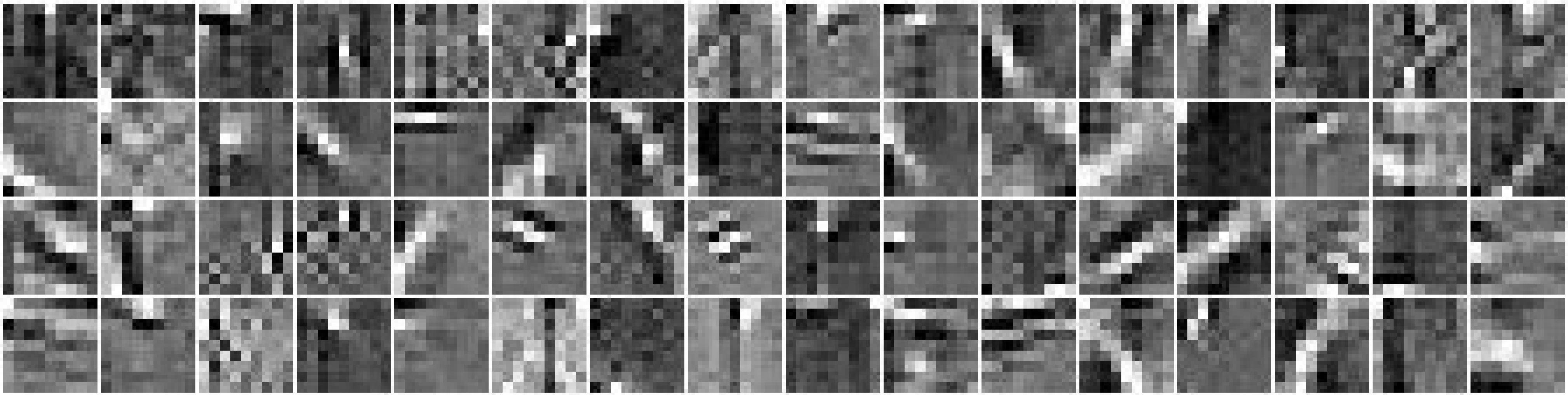
Training samples.

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



Speech Recognition with Convolutional Nets (NYU/IBM)

- **Convolution Kernels at Layer 1:**
  - 64 kernels of size 9x9

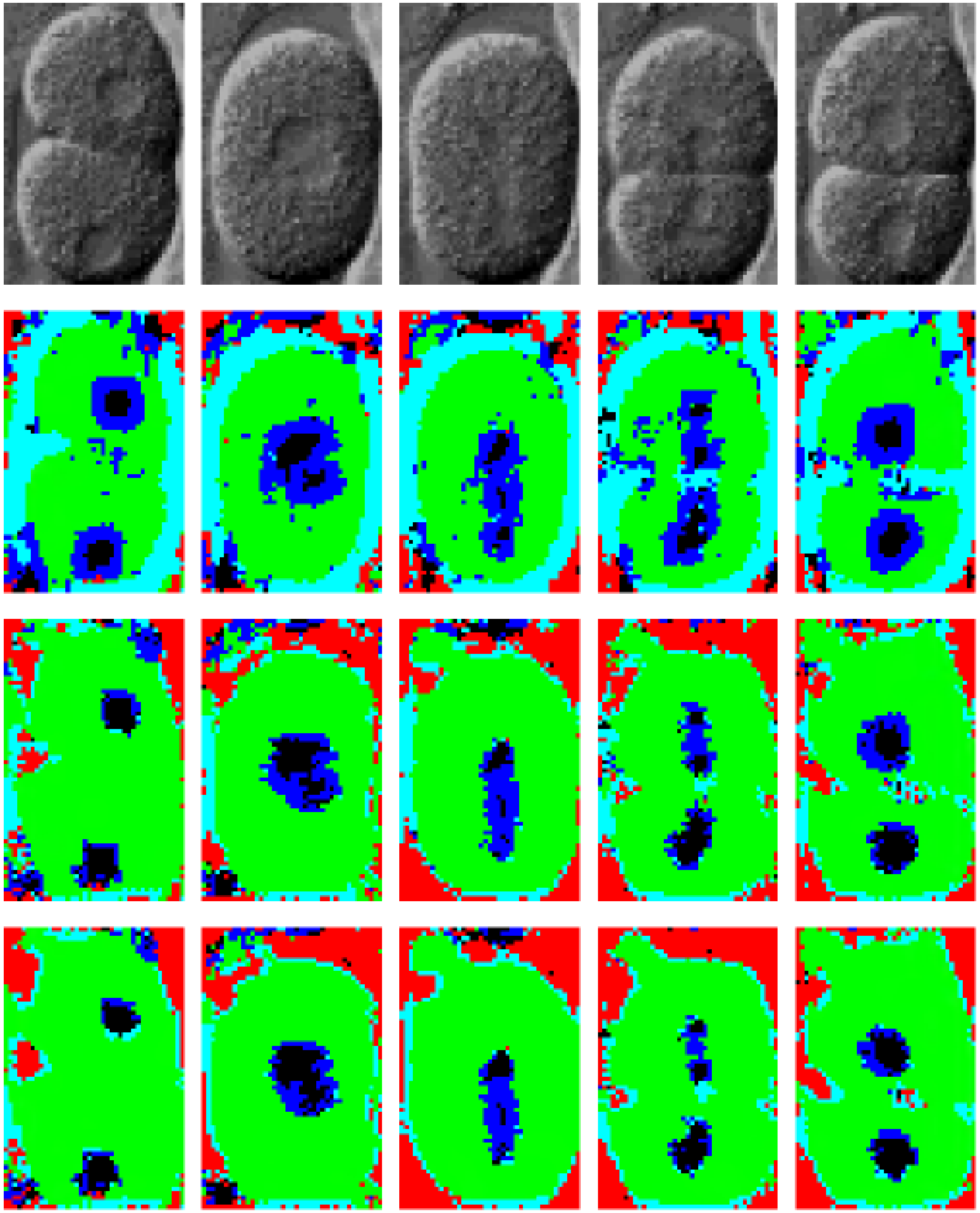


Convolutional Networks
In
Image Segmentation,
& Scene Labeling

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



ConvNet in Connectomics [Jain, Turaga, Seung 2007-present]

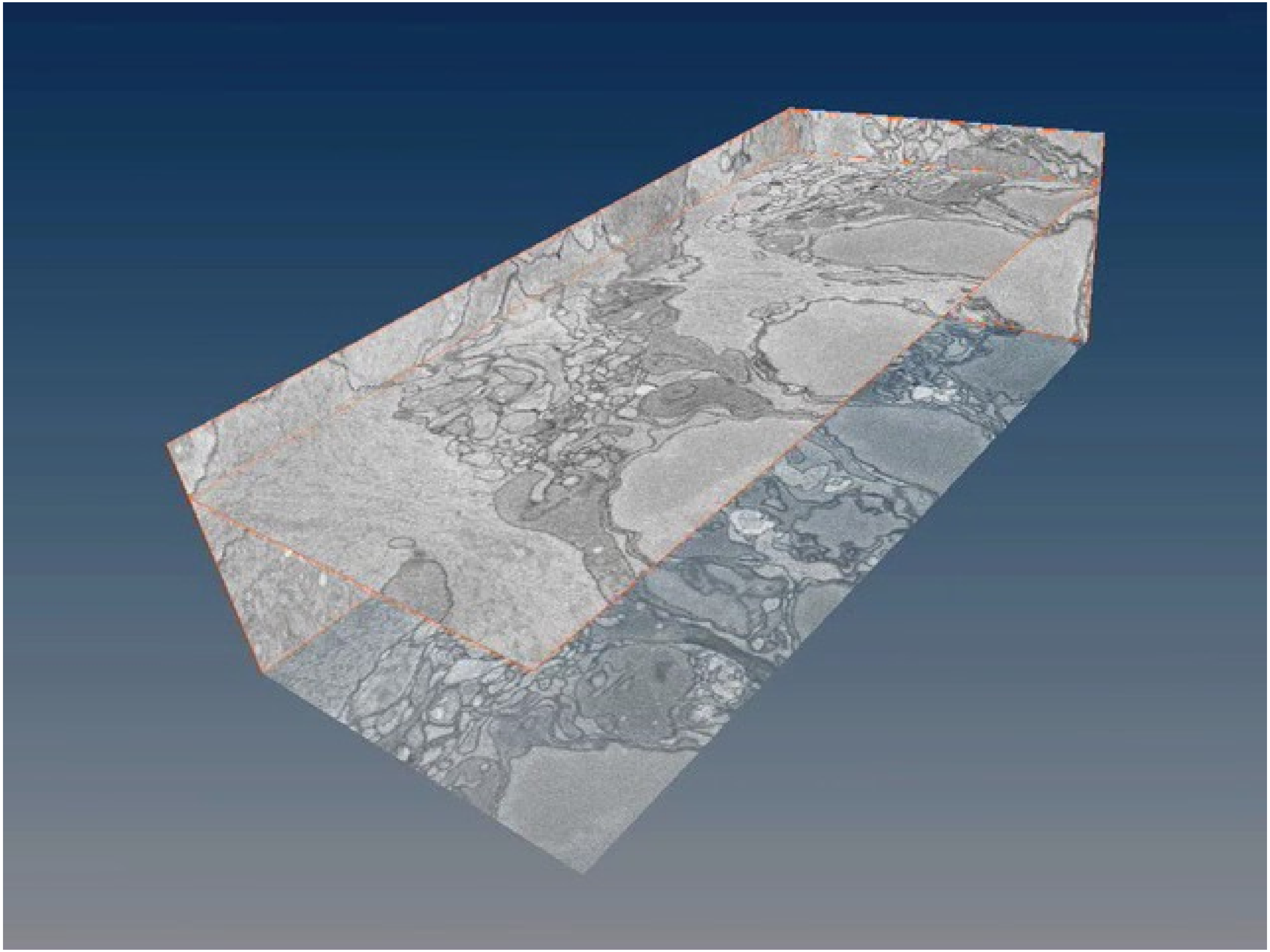
- 3D ConvNet

  Volumetric

  Images

- Each voxel labeled as "membrane" or "non-membrane using a 7x7x7 voxel neighborhood

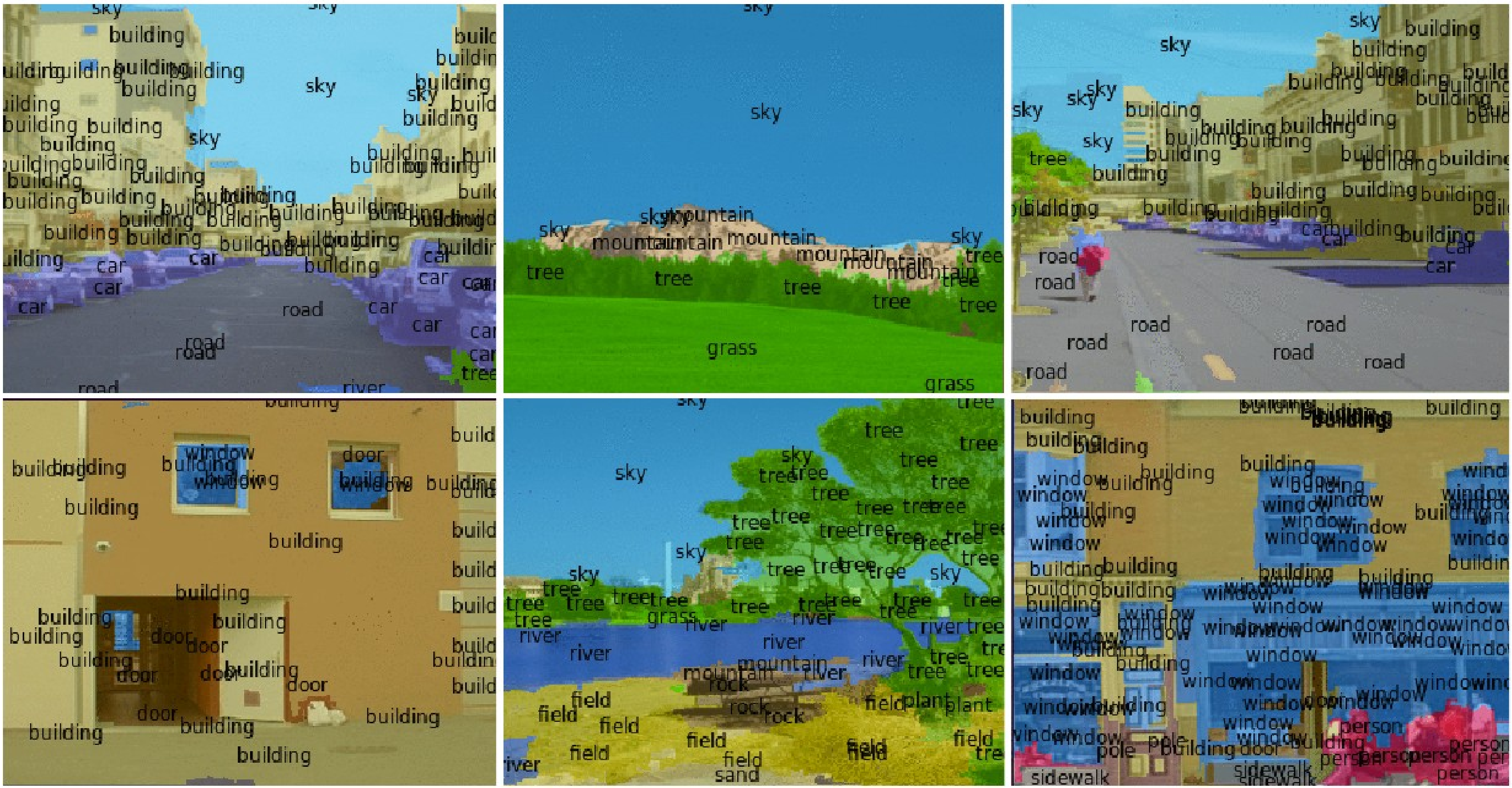
- Has become a standard method in connectomics



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

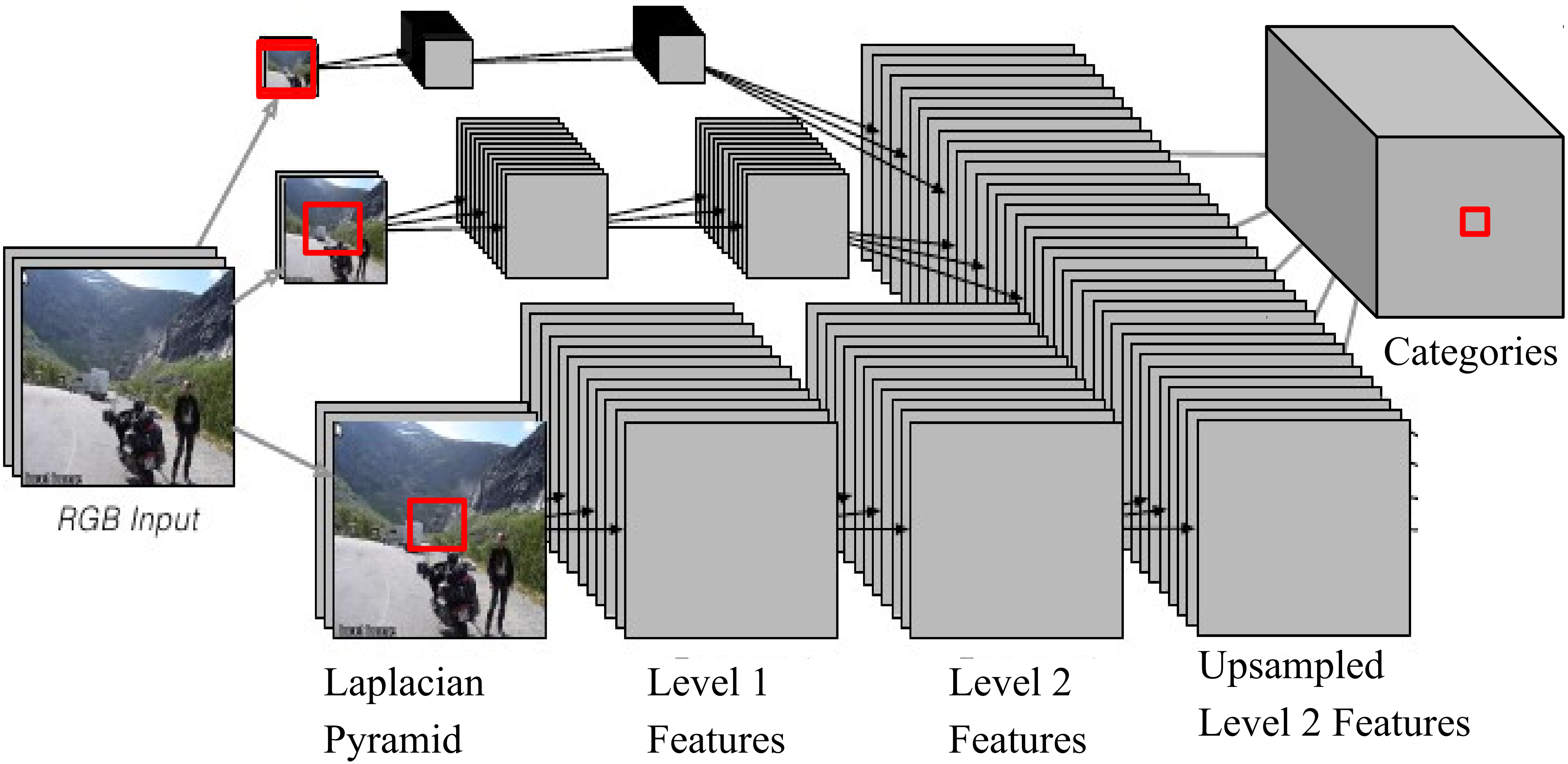


[Farabet et al. ICML 2012, PAMI 2013]

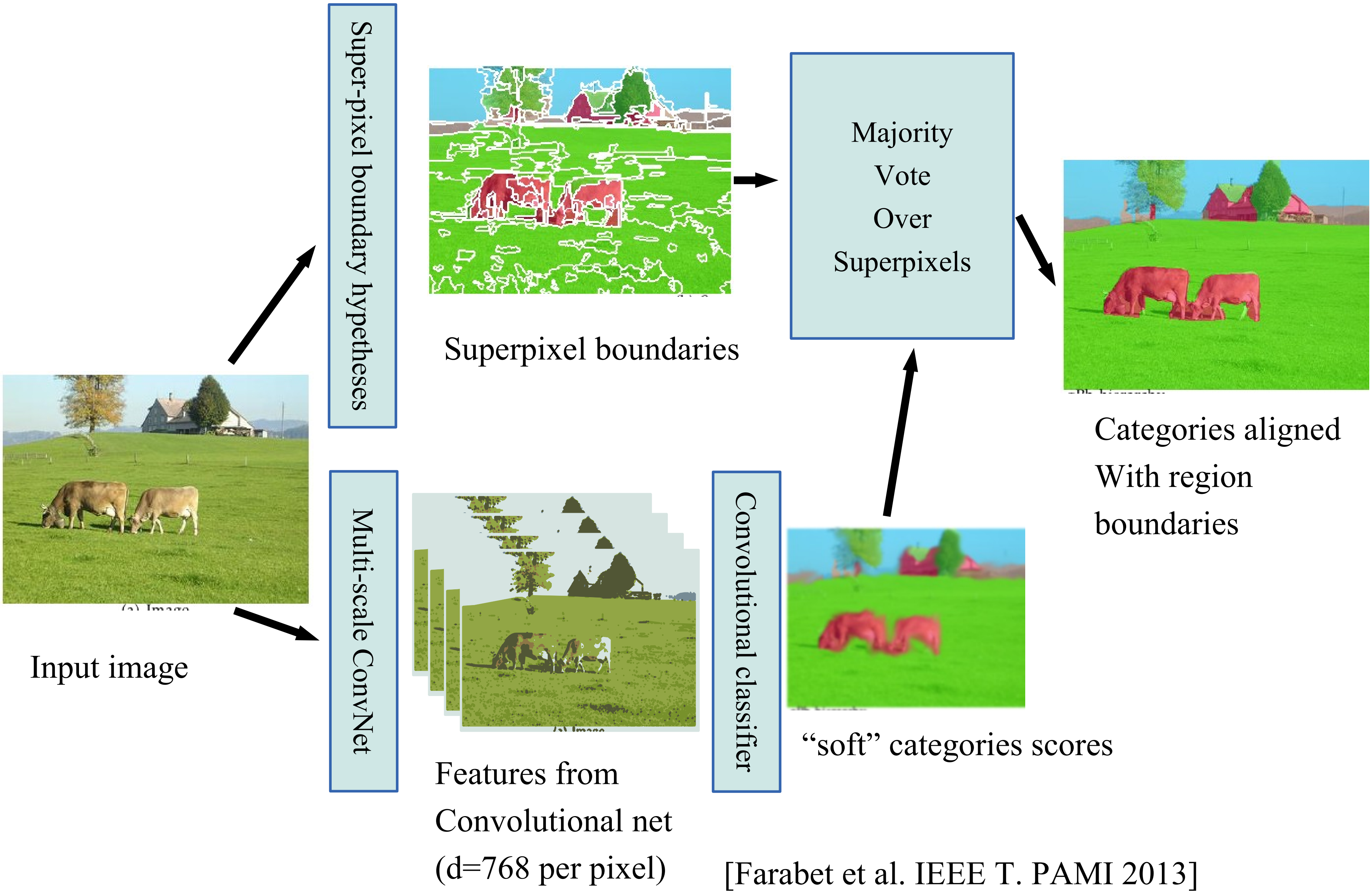
Scene Parsing/Labeling: ConvNet Architecture

Y LeCun

- **Each output sees a large input context:**
  - **46x46** window at full rez; **92x92** at ½ rez; **184x184** at ¼ rez
  - [7x7conv]->[2x2pool]->[7x7conv]->[2x2pool]->[7x7conv]->
  - Trained supervised on fully-labeled images



Method 1: majority over super-pixel regions



[Farabet et al. IEEE T. PAMI 2013]

Scene Parsing/Labeling: Performance

- Stanford Background Dataset [Gould 1009]: 8 categories

	Pixel Acc.	Class Acc.	CT (sec.)
Gould *et al.* 2009 [14]	76.4%	-	10 to 600s
Munoz *et al.* 2010 [32]	76.9%	66.2%	12s
Tighe *et al.* 2010 [46]	77.5%	-	10 to 300s
Socher *et al.* 2011 [45]	78.1%	-	?
Kumar *et al.* 2010 [22]	79.4%	-	$< 600s$
Lempitzky *et al.* 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

[Farabet et al. IEEE T. PAMI 2013]

Scene Parsing/Labeling: Performance

Y LeCun

	Pixel Acc.	Class Acc.
Liu *et al.* 2009 [31]	74.75%	-
Tighe *et al.* 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

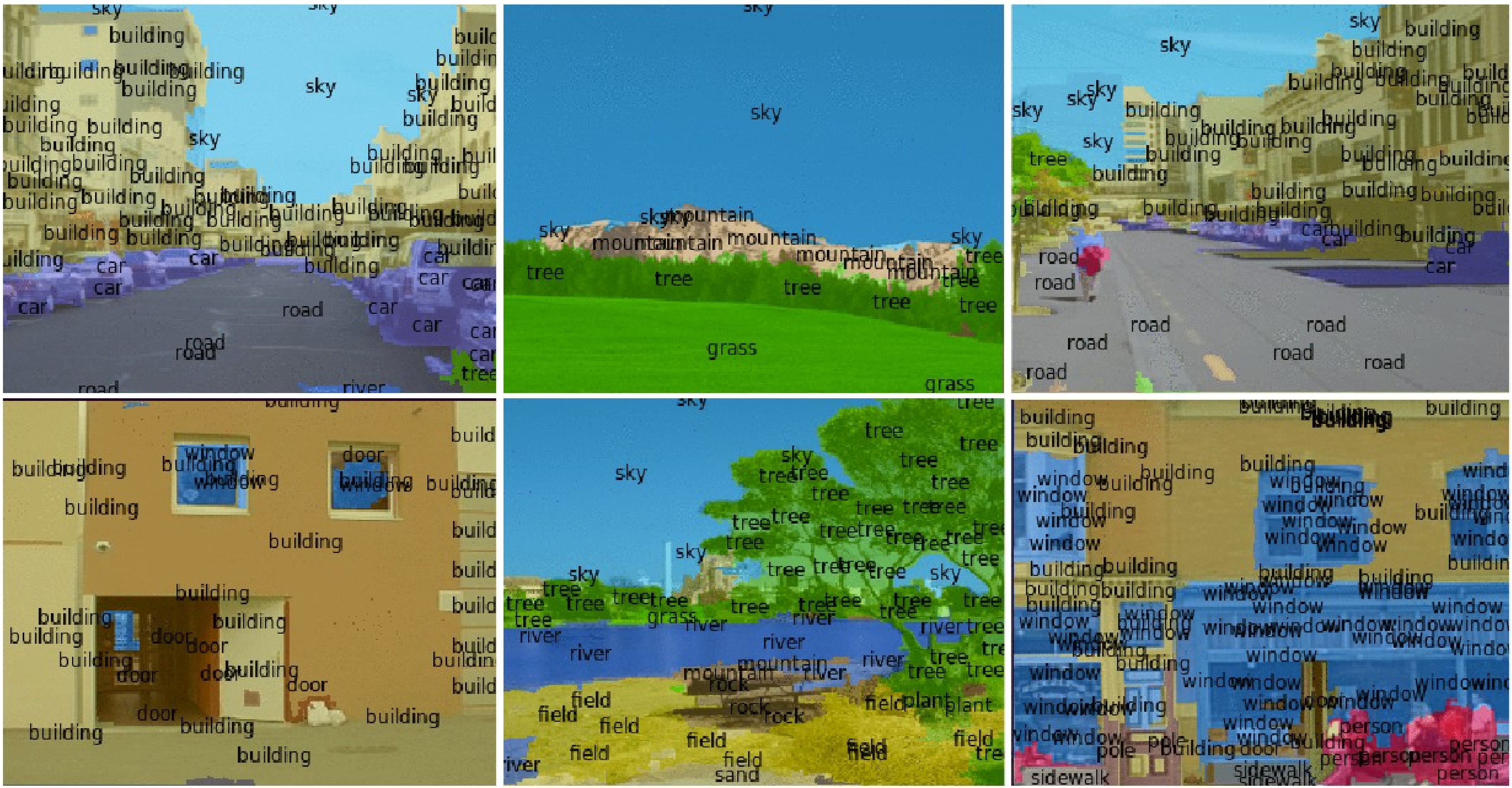
- Barcelona dataset
- [Tighe 2010]:
- 170 categories.

	Pixel Acc.	Class Acc.
Tighe *et al.* 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%

[Farabet et al. IEEE T. PAMI 2012]

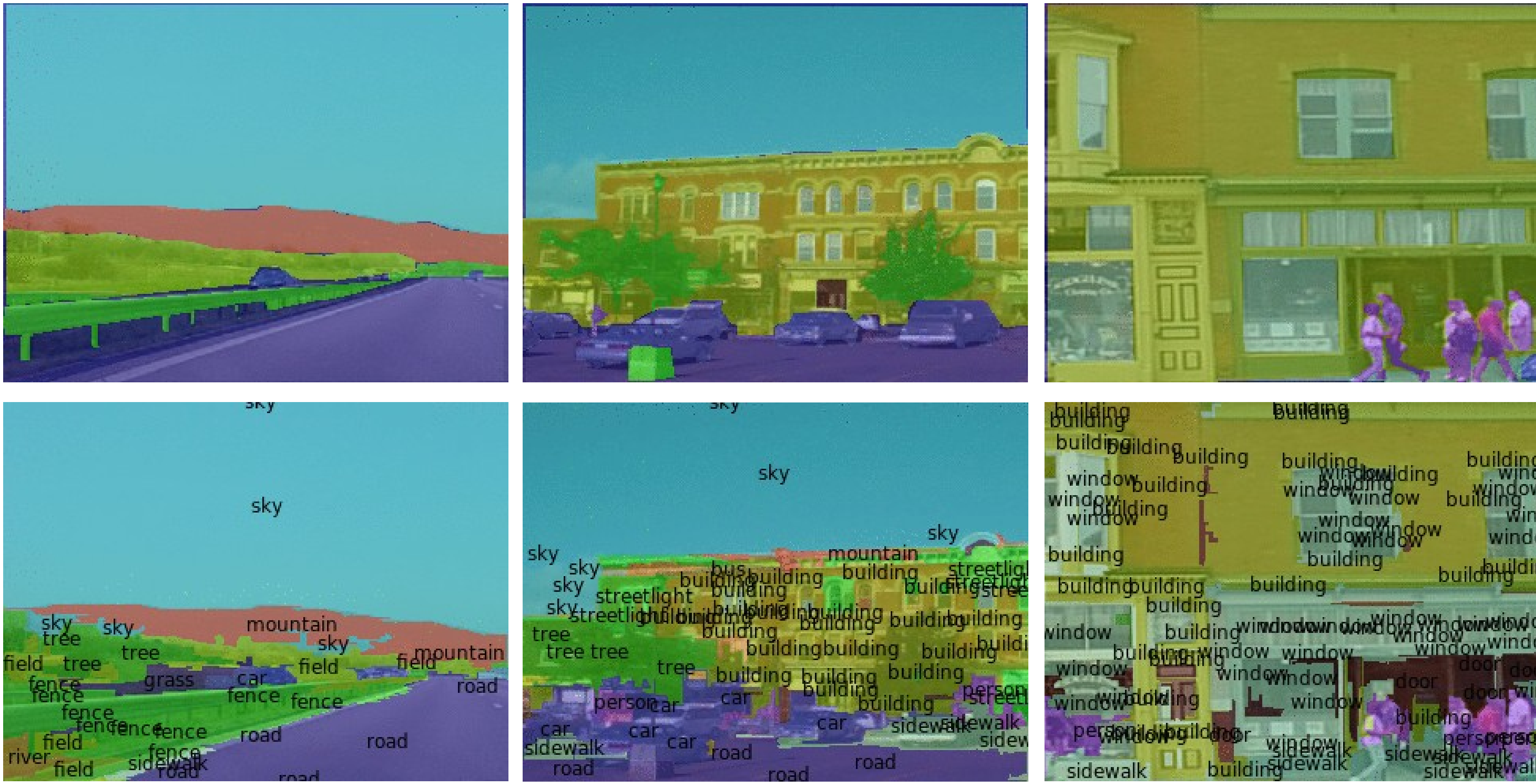
Scene Parsing/Labeling: SIFT Flow dataset (33 categories)

- Samples from the SIFT-Flow dataset (Liu)



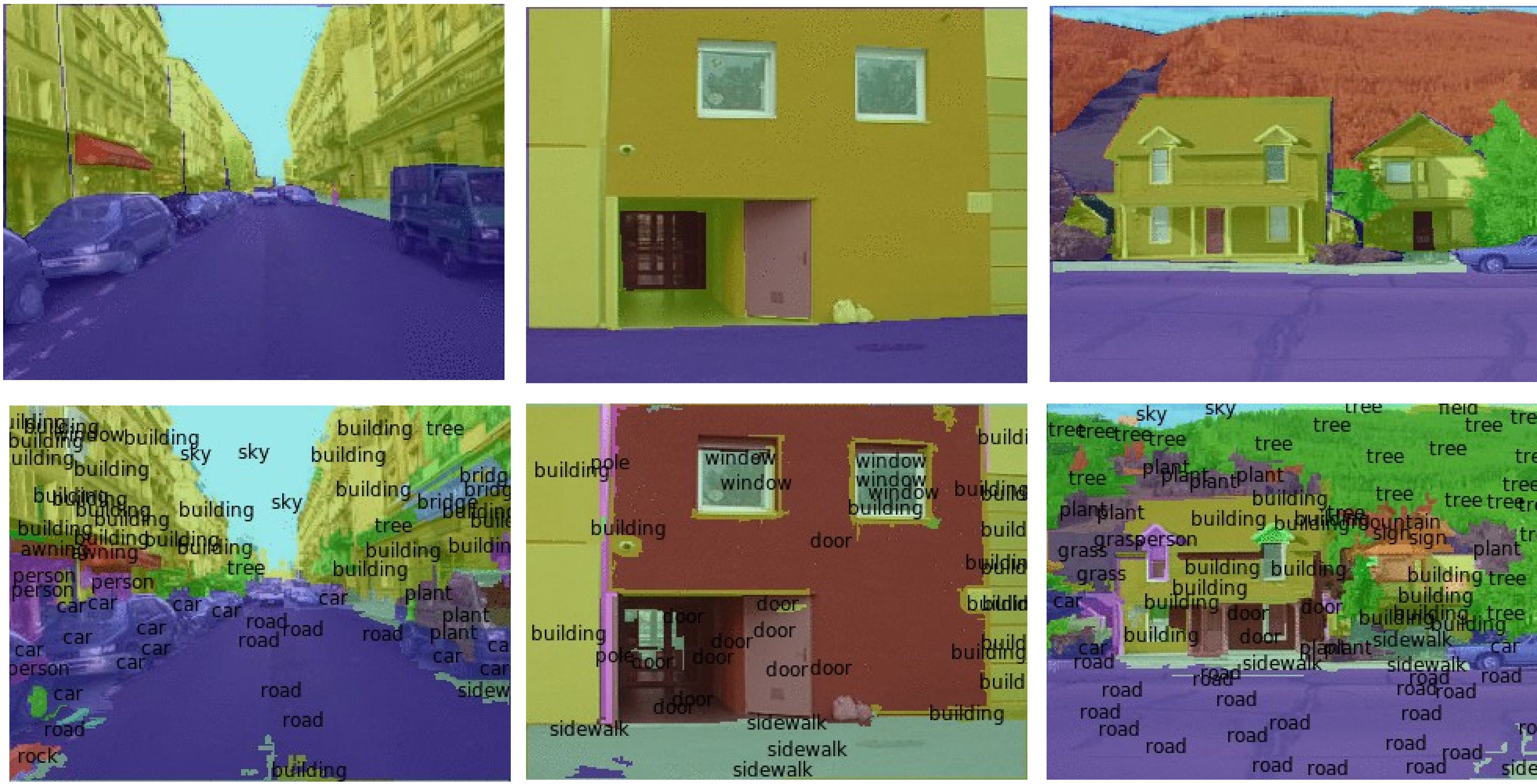
[Farabet et al. ICML 2012, PAMI 2013]

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



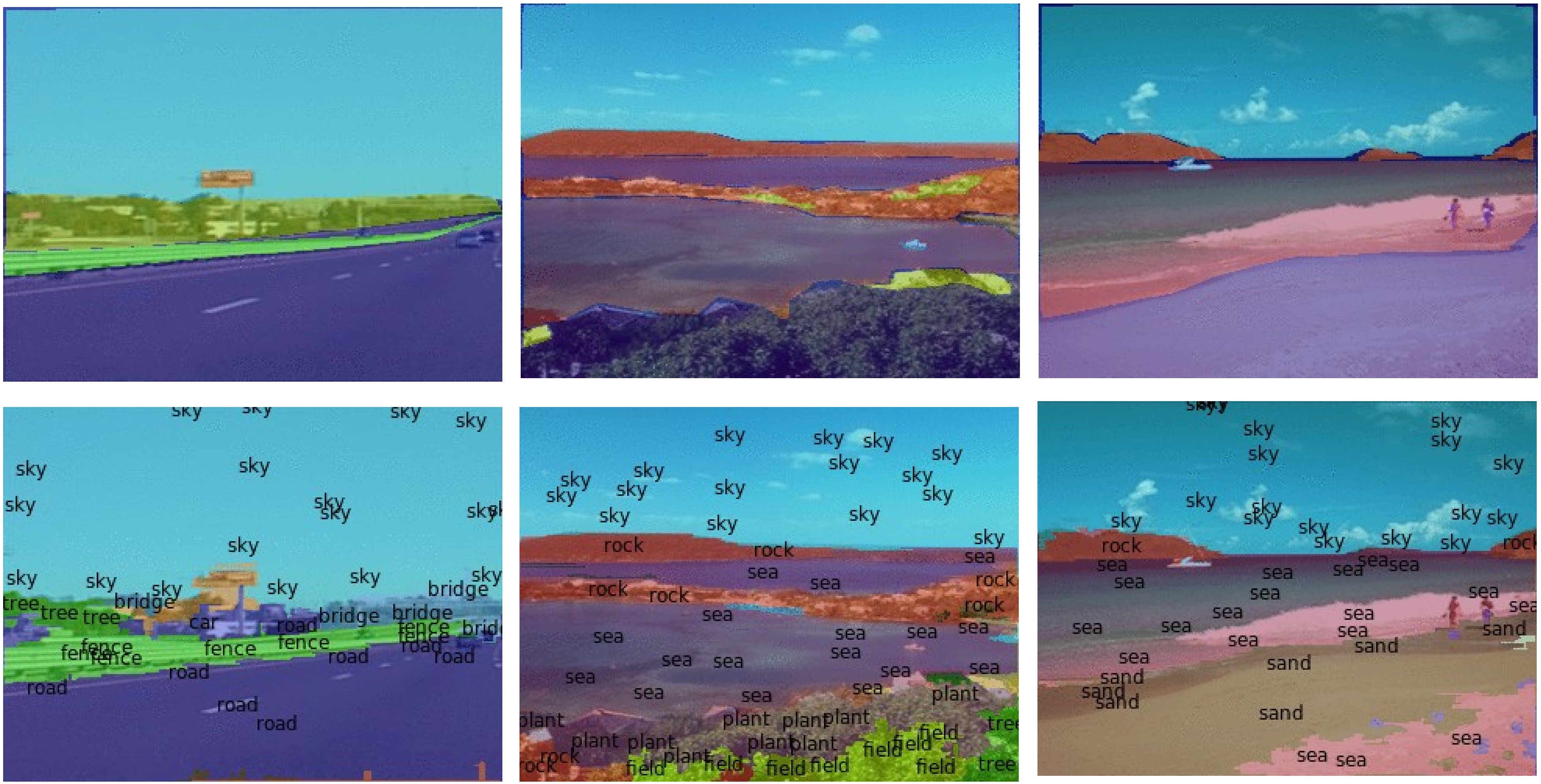
[Farabet et al. ICML 2012, PAMI 2013]

Scene Parsing/Labeling



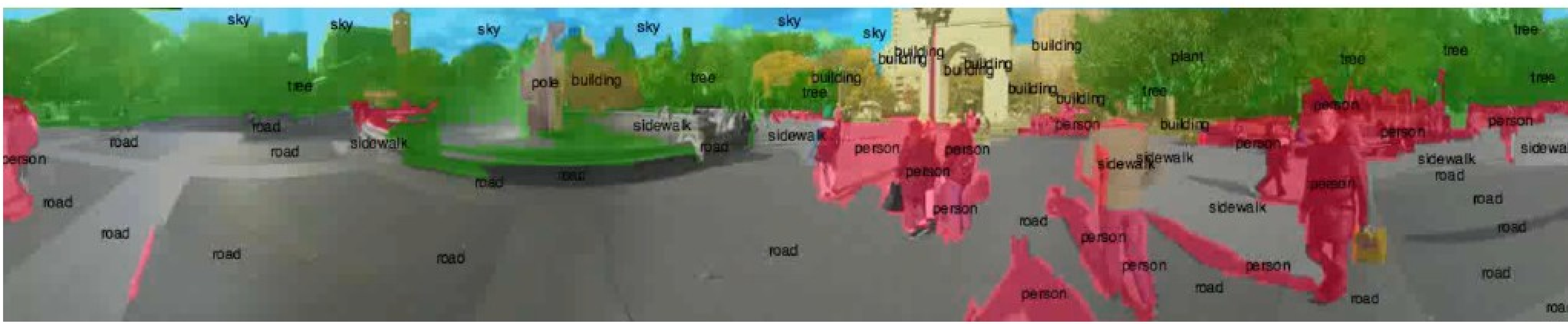
[Farabet et al. ICML 2012, PAMI 2013]

Scene Parsing/Labeling



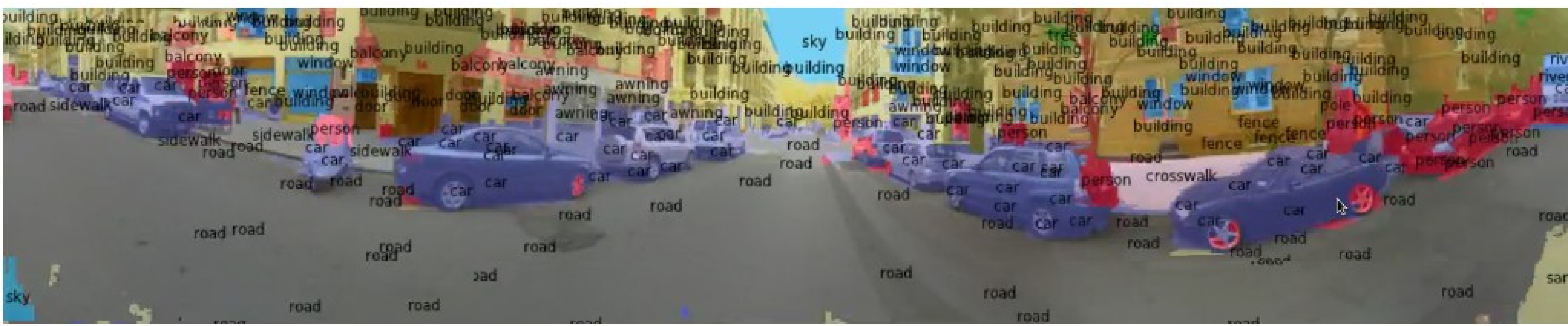
[Farabet et al. ICML 2012, PAMI 2013]

Scene Parsing/Labeling



[Farabet et al. ICML 2012, PAMI 2013]

Scene Parsing/Labeling



[Farabet et al. ICML 2012, PAMI 2013]

Scene Parsing/Labeling

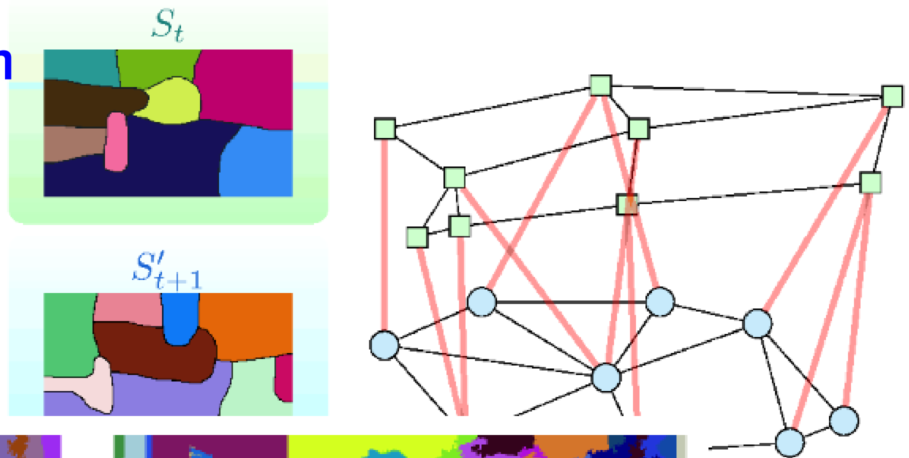
- No post-processing
- Frame-by-frame
- ConvNet runs at 50ms/frame on Virtex-6 FPGA hardware
  - But communicating the features over ethernet limits system performance

Temporal Consistency

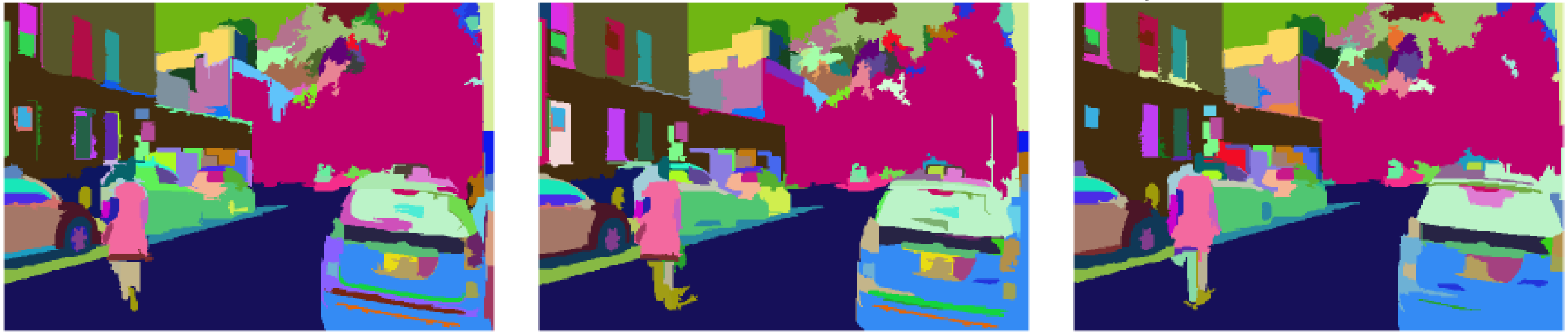
Y LeCun

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

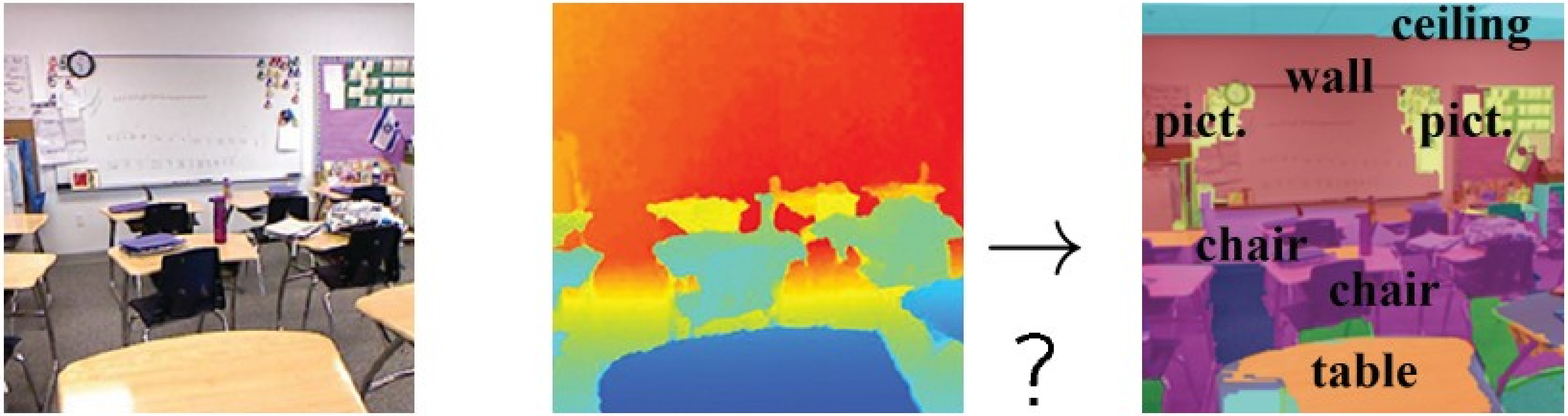
Scene Parsing/Labeling: Temporal Consistency

- Causal method for temporal consistency

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

NYU RGB-D Dataset

- Captured with a Kinect on a steadycam



Results

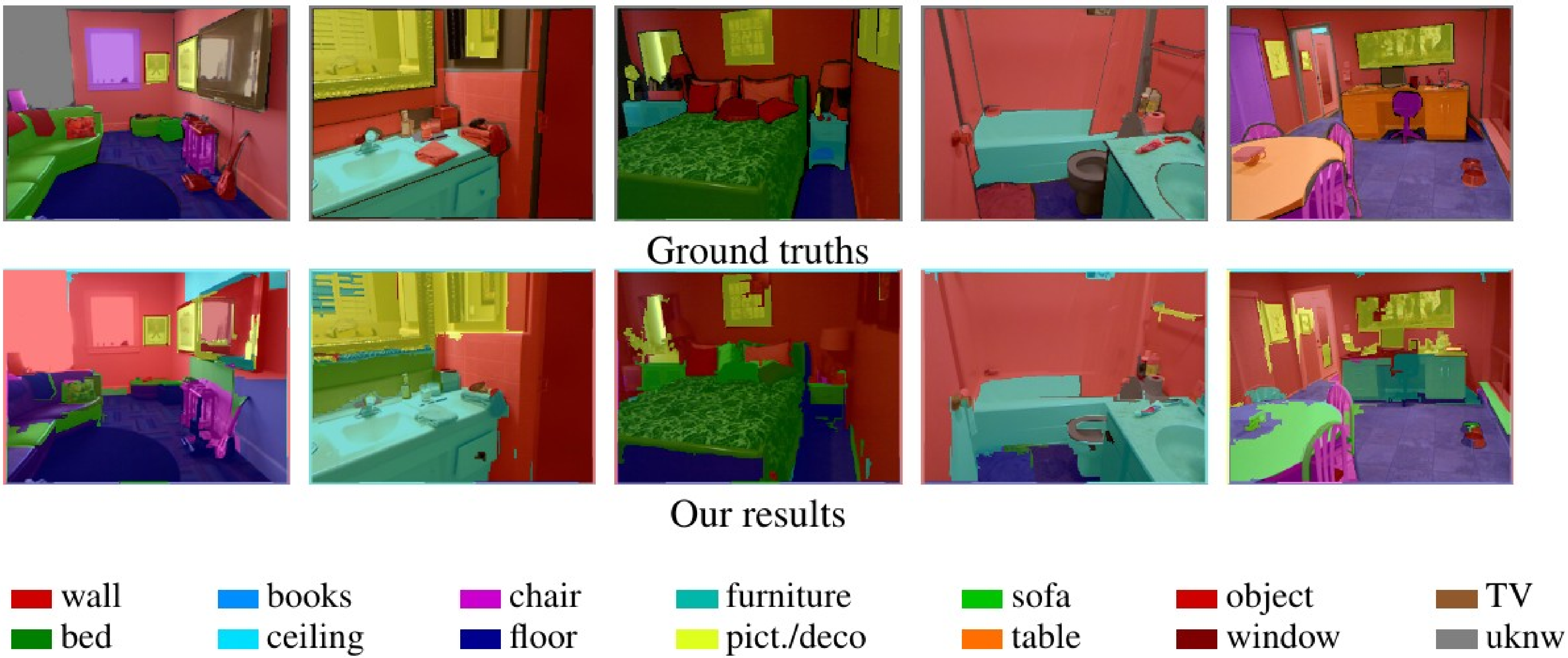
Depth helps a bit

- 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**

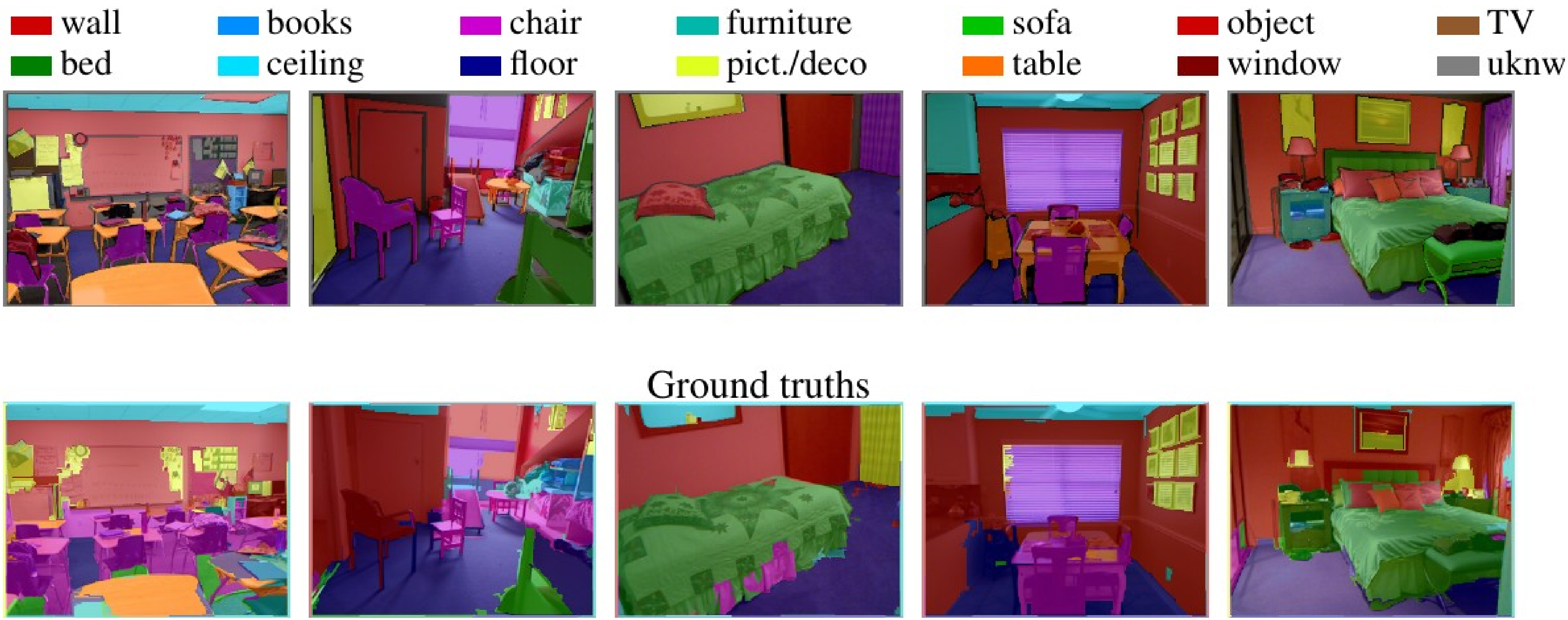
[C. Cadena, J. Kosecka "Semantic Parsing for Priming Object Detection in RGB-D Scenes" Semantic Perception Mapping and Exploration (SPME), Karlsruhe 2013]

Scene Parsing/Labeling on RGB+Depth Images



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

Scene Parsing/Labeling on RGB+Depth Images



Ground truths

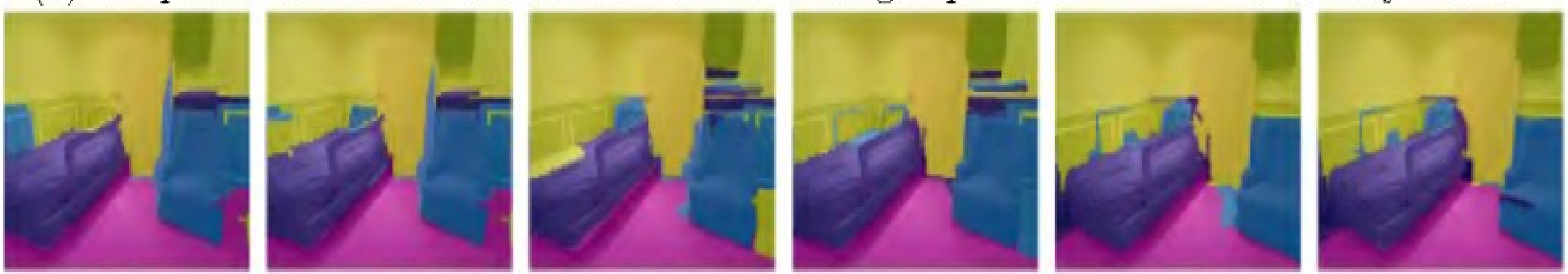
Our results

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

Labeling Videos

- 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

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

Commercial Applications of Convolutional Nets

- 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 Platform for Deep Learning: Torch7

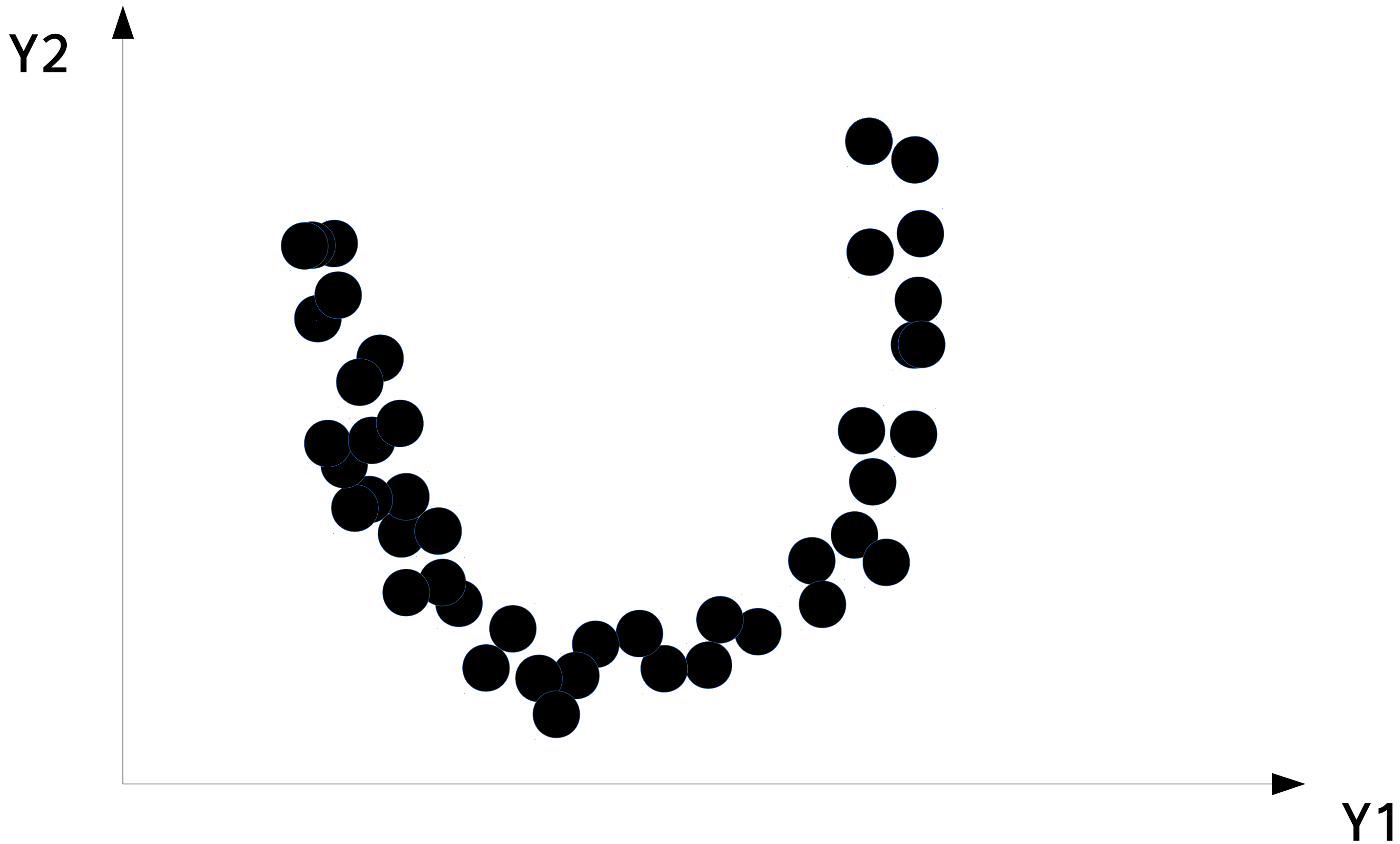
Y LeCun

- **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

Unsupervised Learning

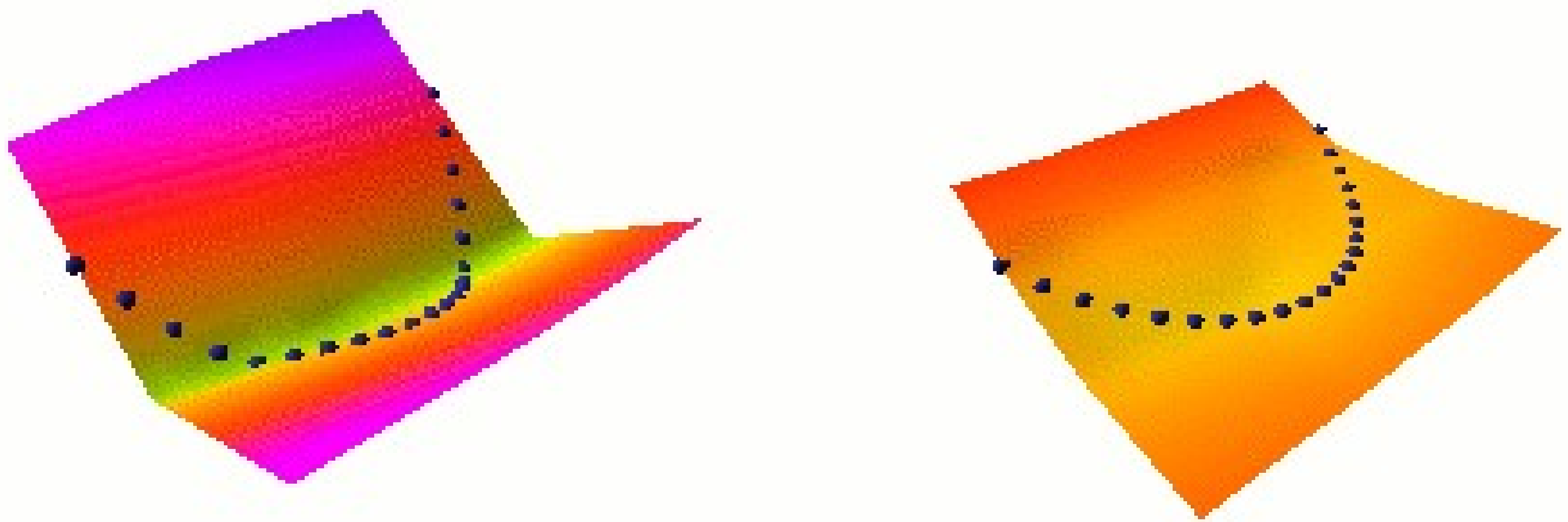
Energy-Based Unsupervised Learning

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



Learning the Energy Function

- 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?



Seven Strategies to Shape the Energy Function

Y LeCun

- **1. build the machine so that the volume of low energy stuff is constant**
  - PCA, K-means, GMM, square ICA
- **2. push down of the energy of data points, push up everywhere else**
  - Max likelihood (needs tractable partition function)
- **3. push down of the energy of data points, push up on chosen locations**
  - contrastive divergence, Ratio Matching, Noise Contrastive Estimation, Minimum Probability Flow
- **4. minimize the gradient and maximize the curvature around data points**
  - score matching
- **5. train a dynamical system so that the dynamics goes to the manifold**
  - denoising auto-encoder
- **6. use a regularizer that limits the volume of space that has low energy**
  - Sparse coding, sparse auto-encoder, PSD
- **7. if E(Y) = ||Y - G(Y)||^2, make G(Y) as "constant" as possible.**
  - Contracting auto-encoder, saturating auto-encoder

#1: constant volume of low energy
Energy surface for PCA and K-means

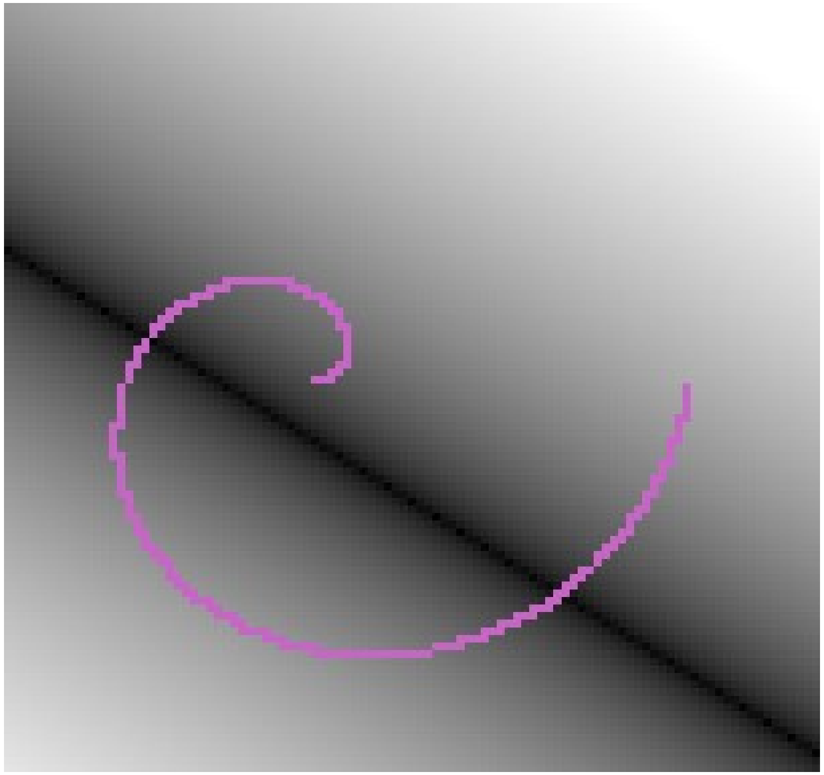
- **1. build the machine so that the volume of low energy stuff is constant**
  - PCA, K-means, GMM, square ICA...

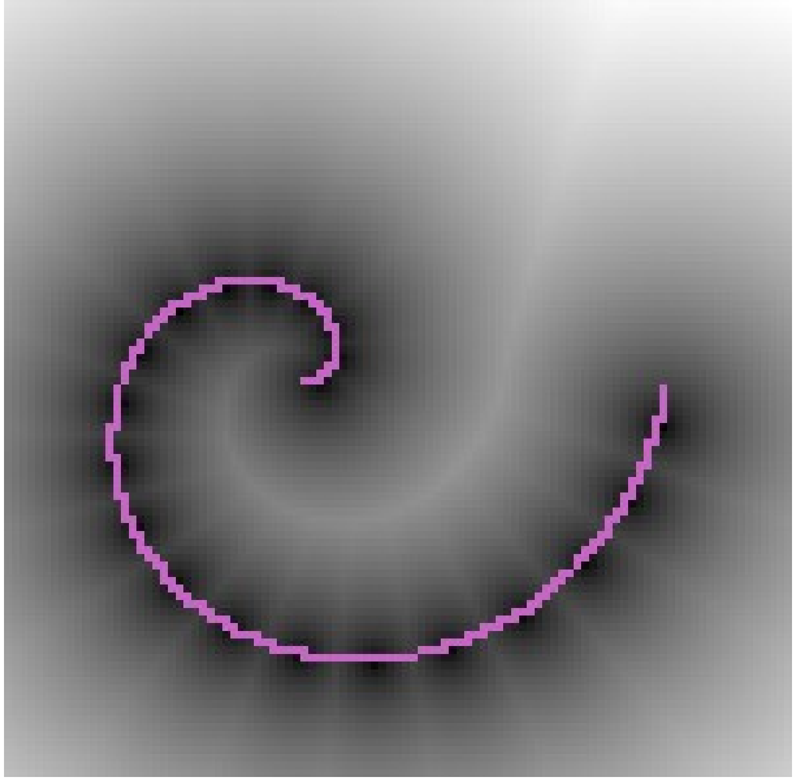
PCA

$$E(Y) = \| W^T W Y - Y \|^2$$

K-Means,
Z constrained to 1-of-K code

$$E(Y) = min_z \sum_i \| Y - W_i Z_i \|^2$$





#2: push down of the energy of data points, push up everywhere else

Y LeCun

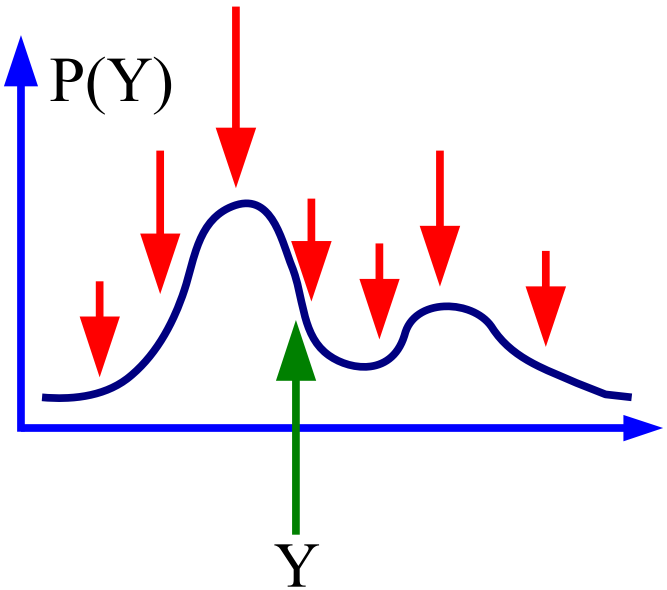
Max likelihood (requires a tractable partition function)

Maximizing P(Y|W) on training samples

make this big

$$P(Y|W) = \frac{e^{-\beta E(Y,W)}}{\int_y e^{-\beta E(y,W)}}$$

make this small

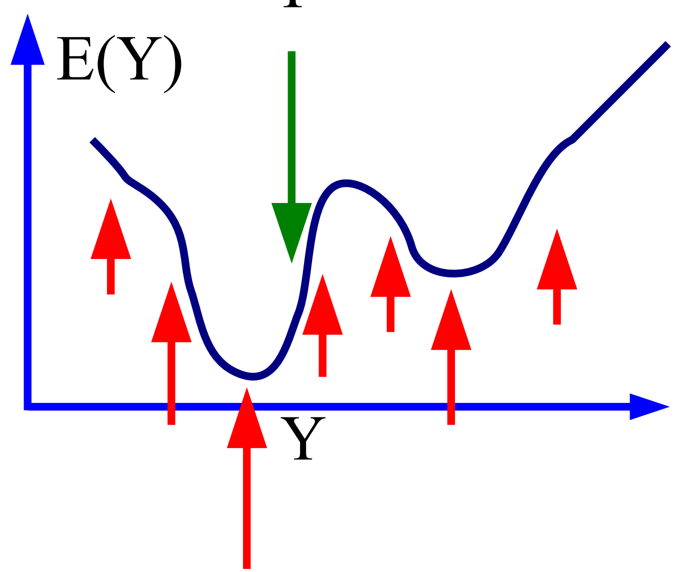


Minimizing -log P(Y,W) on training samples

$$L(Y,W) = E(Y,W) + \frac{1}{\beta}\log\int_y e^{-\beta E(y,W)}$$

make this small

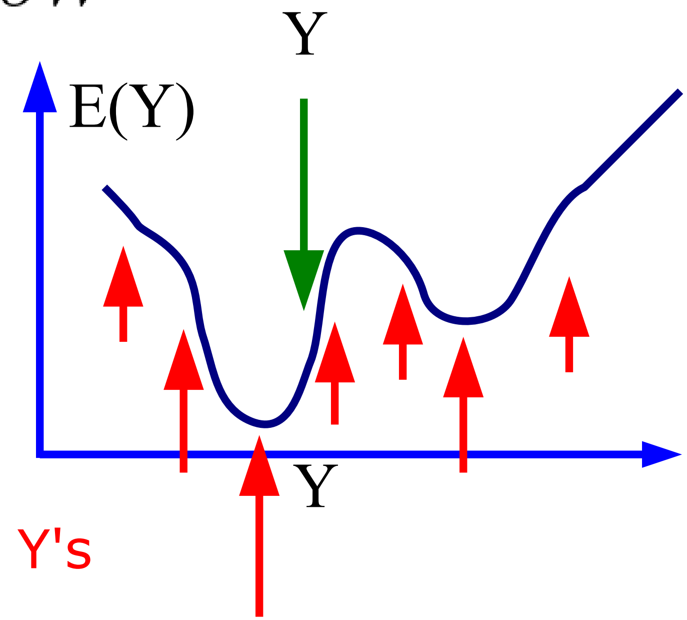
make this big



#2: push down of the energy of data points, push up everywhere else

Gradient of the negative log-likelihood loss for one sample Y:

$$\frac{\partial L(Y,W)}{\partial W} = \frac{\partial E(Y,W)}{\partial W} - \int_y P(y|W)\frac{\partial E(y,W)}{\partial W}$$



Gradient descent:

$$W \leftarrow W - \eta \frac{\partial L(Y,W)}{\partial W}$$

Pushes down on the energy of the samples

Pulls up on the energy of low-energy Y's

$$W \leftarrow W - \eta \frac{\partial E(Y,W)}{\partial W} + \eta \int_y P(y|W)\frac{\partial E(y,W)}{\partial W}$$

#3. push down of the energy of data points, push up on chosen locations

Y LeCun

- **contrastive divergence, Ratio Matching, Noise Contrastive Estimation, Minimum Probability Flow**
- **Contrastive divergence: basic idea**
  - Pick a training sample, lower the energy at that point
  - From the sample, move down in the energy surface with noise
  - Stop after a while
  - Push up on the energy of the point where we stopped
  - This creates grooves in the energy surface around data manifolds
  - CD can be applied to any energy function (not just RBMs)
- **Persistent CD: use a bunch of "particles" and remember their positions**
  - Make them roll down the energy surface with noise
  - Push up on the energy wherever they are
  - Faster than CD
- **RBM**

$$E(Y,Z) = -Z^T W Y \qquad E(Y) = -\log \sum_z e^{Z^T W Y}$$

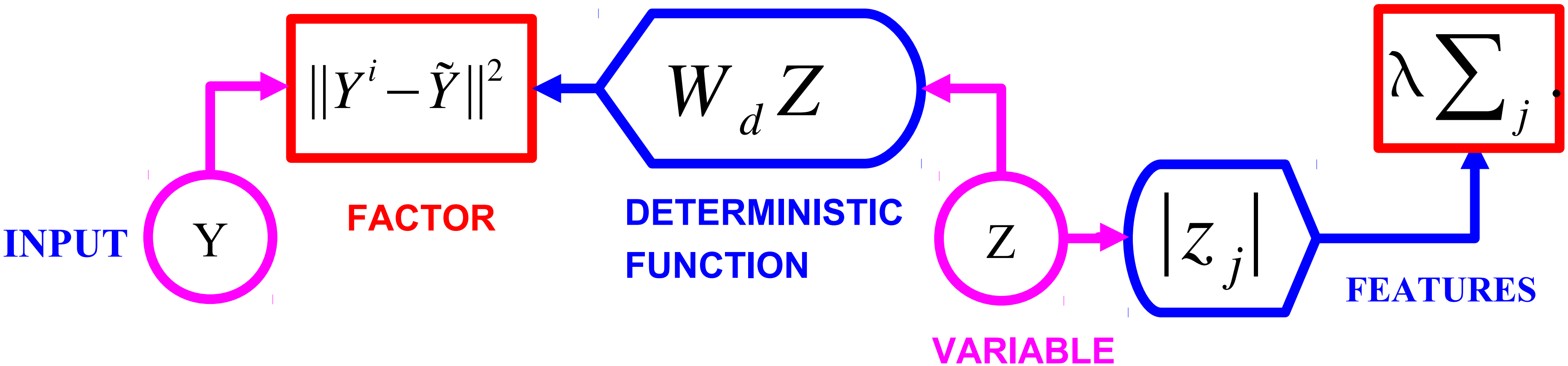
Dictionary Learning With Fast Approximate Inference: Sparse Auto-Encoders

Sparse Modeling: Sparse Coding + Dictionary Learning

[Olshausen & Field 1997]

- Sparse linear reconstruction
- Energy = reconstruction_error + code_prediction_error + code_sparsity

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

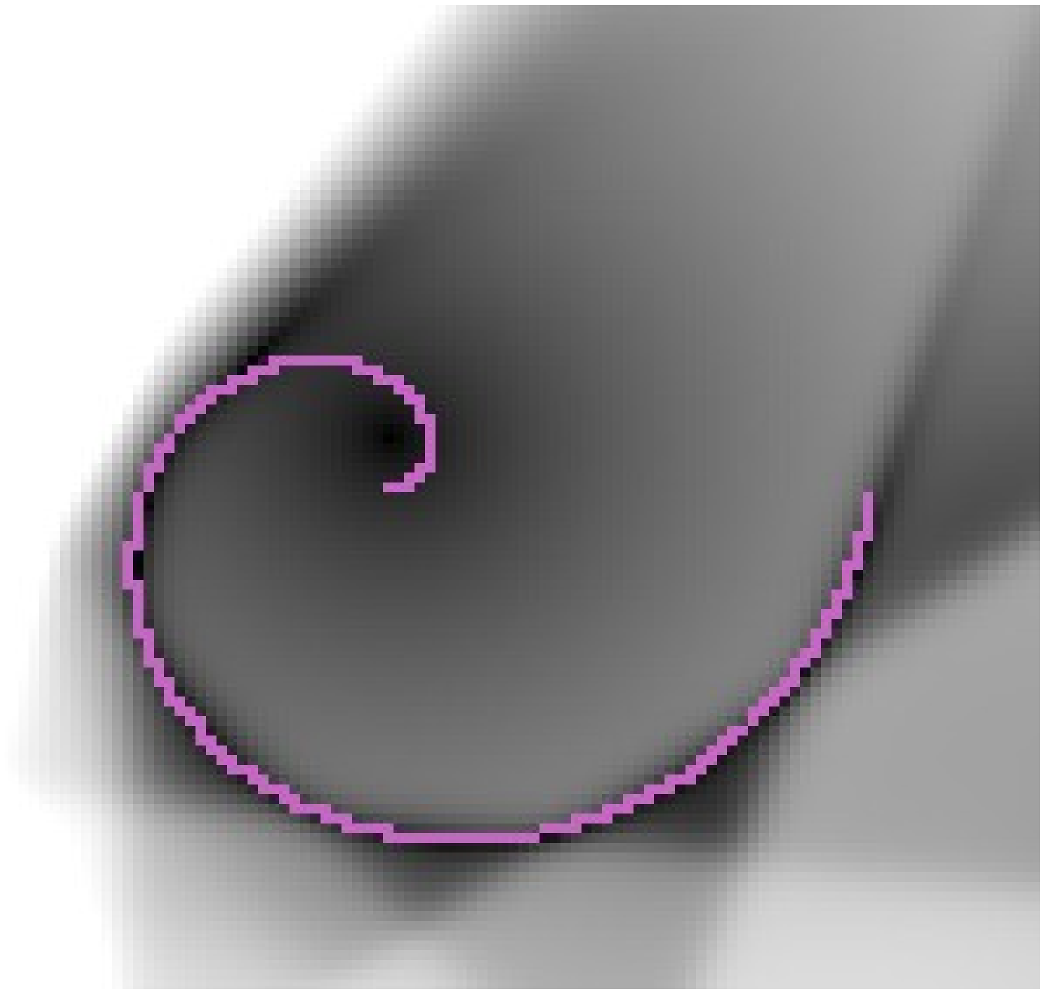


- Inference is expensive: ISTA/FISTA, CGIHT, coordinate descent....

$$Y \rightarrow \hat{Z} = argmin_Z E(Y, Z)$$

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

- Sparse coding, sparse auto-encoder, Predictive Saprse Decomposition



Learning to Perform Approximate Inference: Predictive Sparse Decomposition Sparse Auto-Encoders

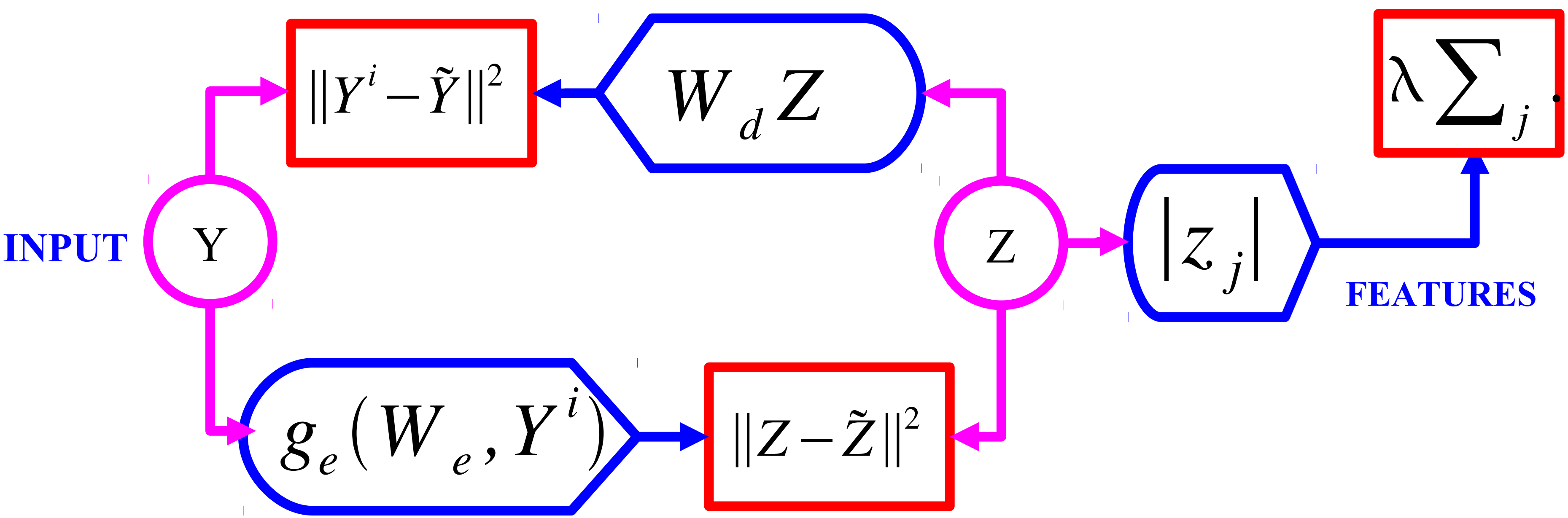
Sparse auto-encoder: Predictive Sparse Decomposition (PSD)

[Kavukcuoglu, Ranzato, LeCun, 2008 → arXiv:1010.3467],

- Prediction the optimal code with a trained encoder
- Energy = reconstruction_error + code_prediction_error + code_sparsity

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

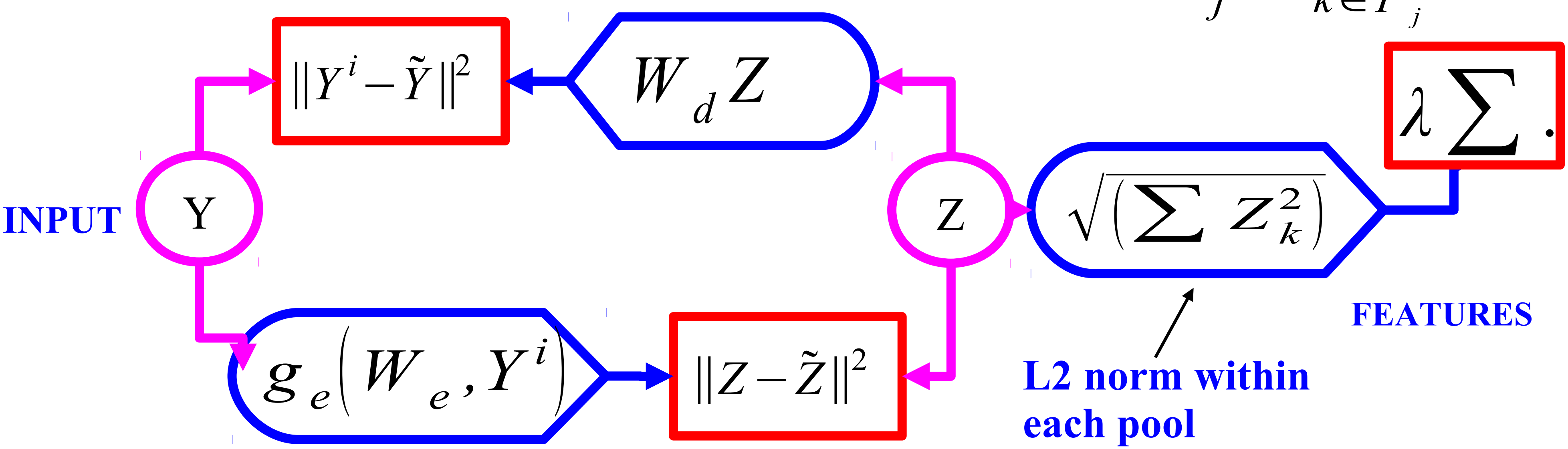
$$g_e(W_e, Y^i) = shrinkage(W_e Y^i)$$



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

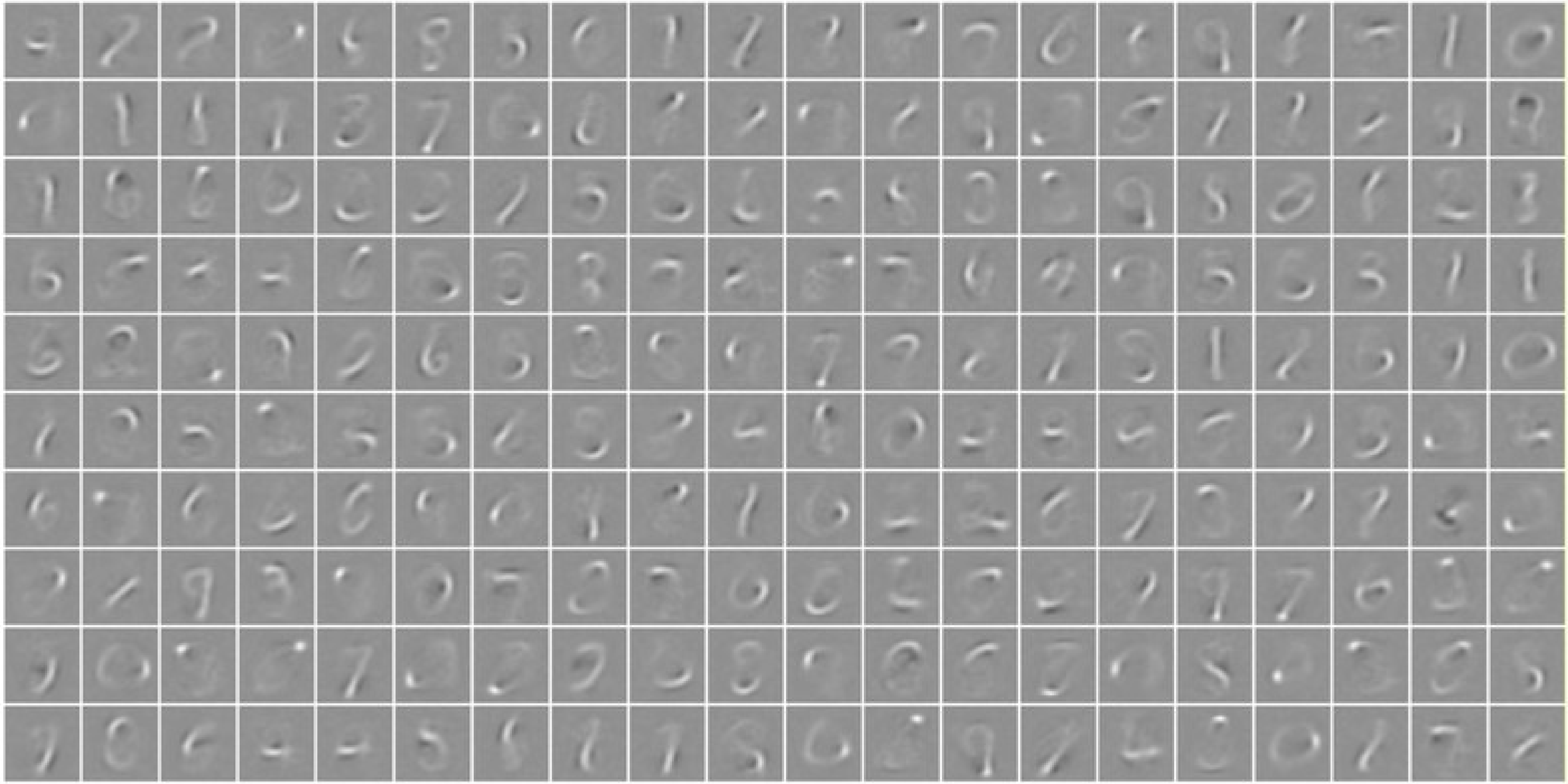
- **Encoder:** computes feature vector Z from input X
- **Decoder:** reconstructs input X from feature vector Z
- **Feature vector:** high dimensional and regularized (e.g. sparse)
- **Factor graph with energy function E(X,Z) with 3 terms:**
  - Linear decoding function and reconstruction error
  - Non-Linear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

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



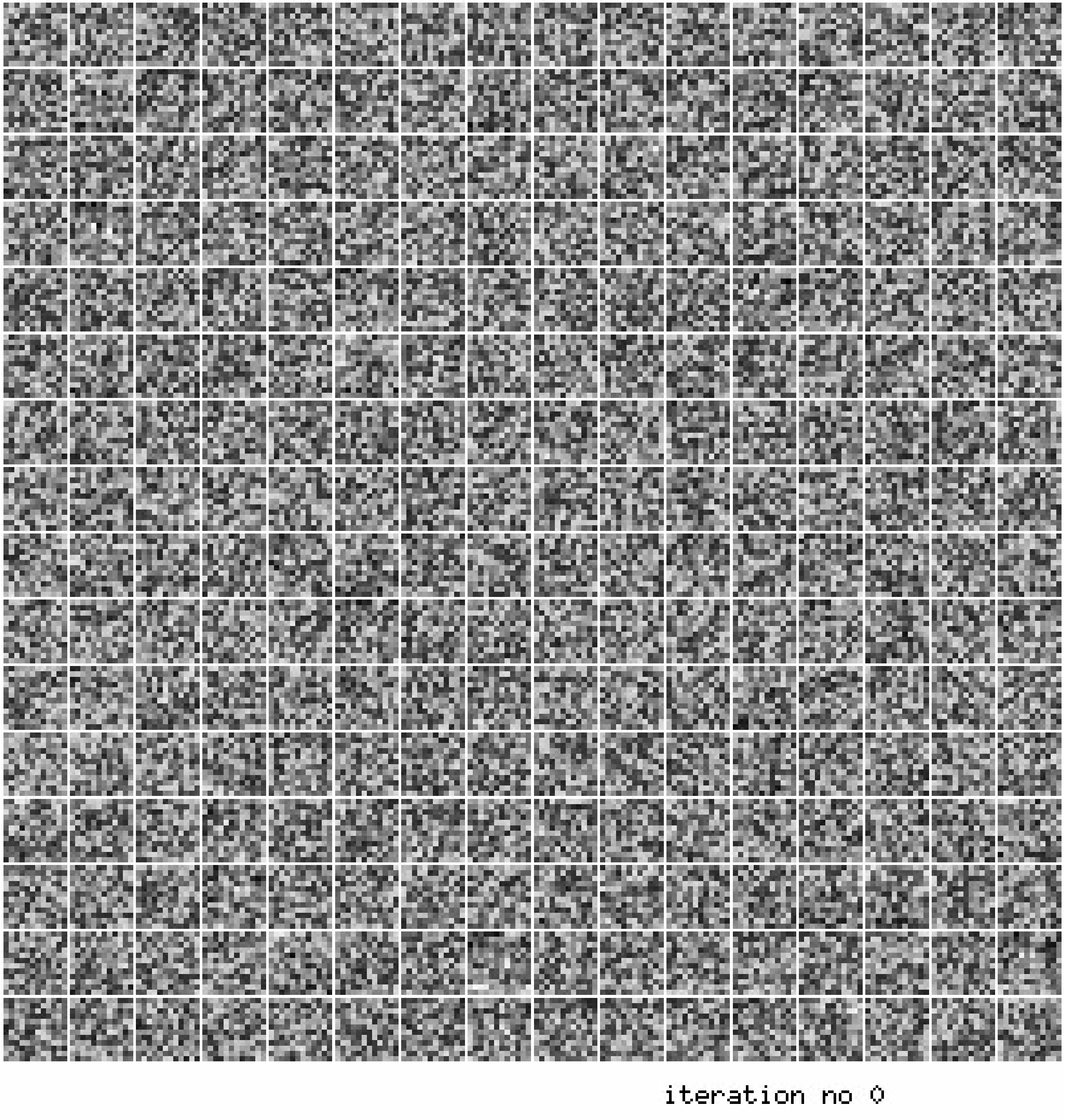
PSD: Basis Functions on MNIST

- Basis functions (and encoder matrix) are digit parts

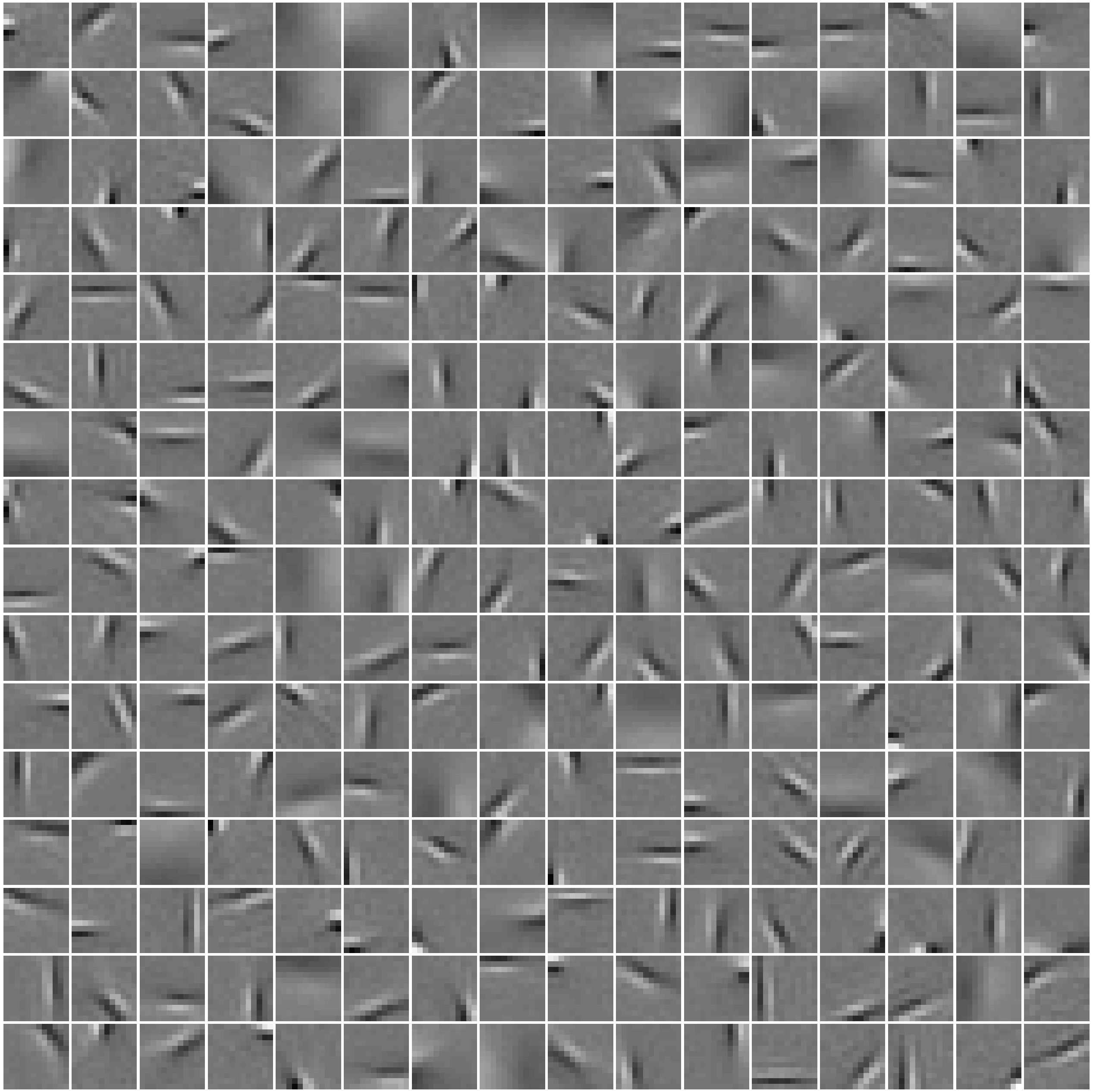


Predictive Sparse Decomposition (PSD): Training

- **Training on natural images patches.**
  - 12X12
  - 256 basis functions



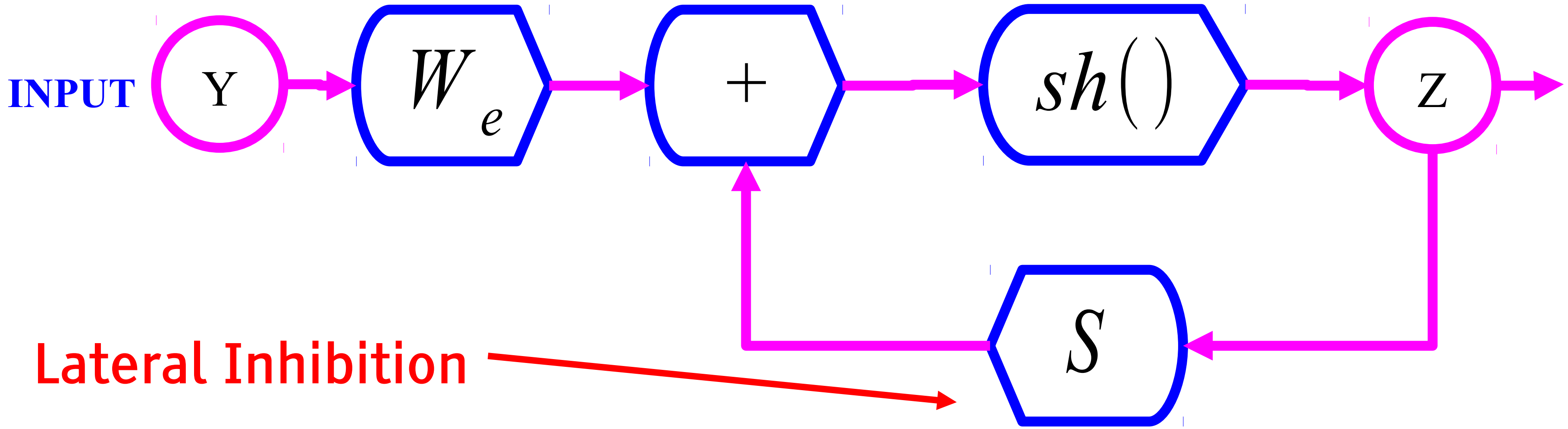
Learned Features on natural patches: V1-like receptive fields



Learning to Perform Approximate Inference
LISTA

Better Idea: Give the "right" structure to the encoder

- ISTA/FISTA: iterative algorithm that converges to optimal sparse code



$$Z(t+1) = \text{Shrinkage}_{\lambda/L}\left[Z(t) - \frac{1}{L}W_d^T(W_d Z(t) - Y)\right]$$

- ISTA/FISTA reparameterized:

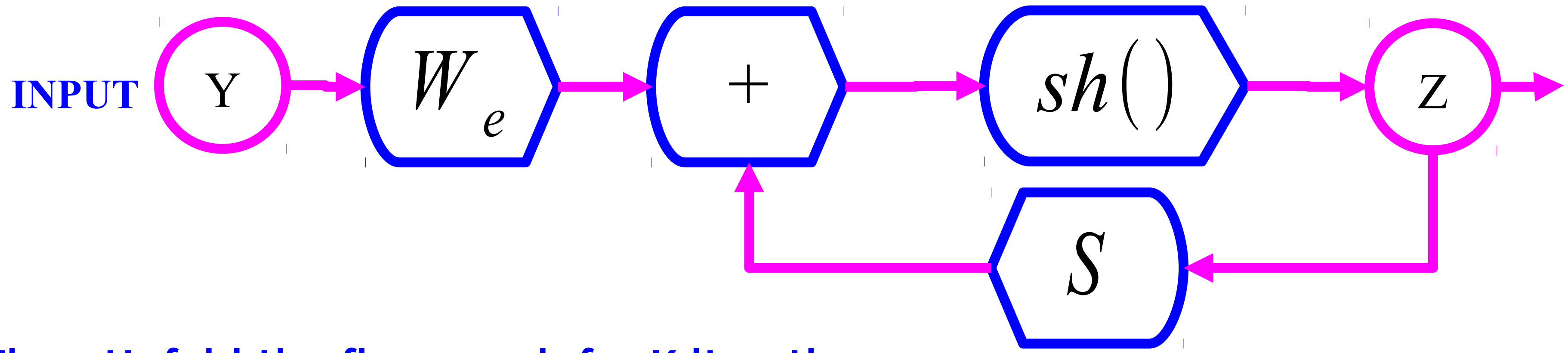
$$Z(t+1) = \text{Shrinkage}_{\lambda/L}\left[W_e^T Y + SZ(t)\right]; \quad W_e = \frac{1}{L}W_d; \quad S = I - \frac{1}{L}W_d^T W_d$$

- LISTA (Learned ISTA): learn the We and S matrices to get fast solutions

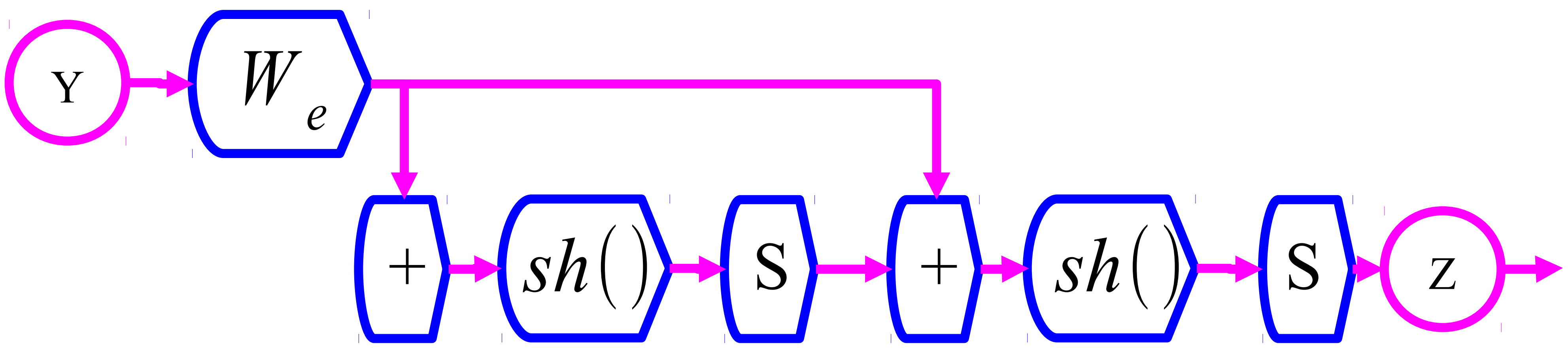
[Gregor & LeCun, ICML 2010], [Bronstein et al. ICML 2012], [Rolfe & LeCun ICLR 2013]

LISTA: Train We and S matrices to give a good approximation quickly

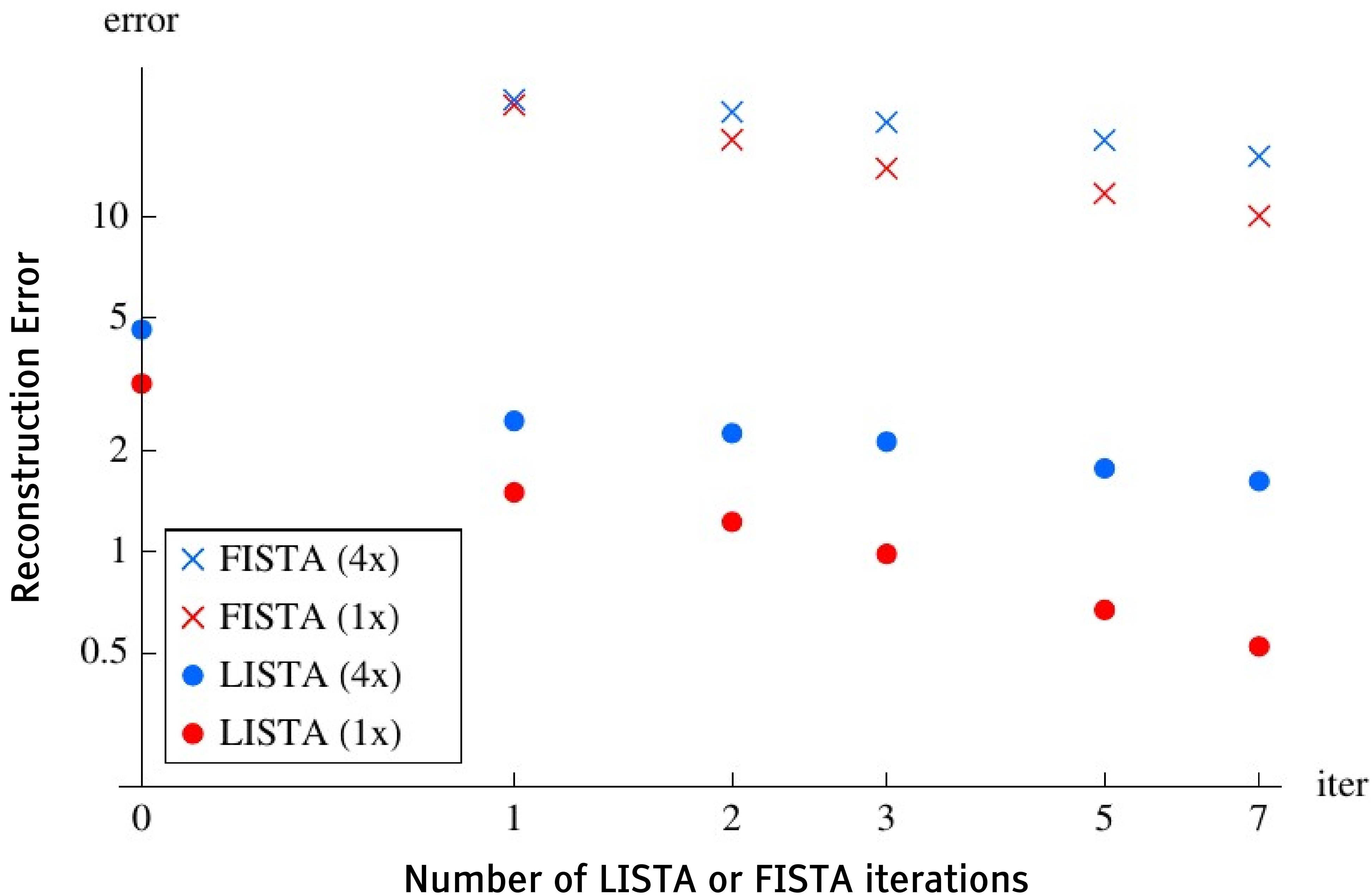
- Think of the FISTA flow graph as a recurrent neural net where We and S are trainable parameters



- Time-Unfold the flow graph for K iterations
- Learn the We and S matrices with "backprop-through-time"
- Get the best approximate solution within K iterations

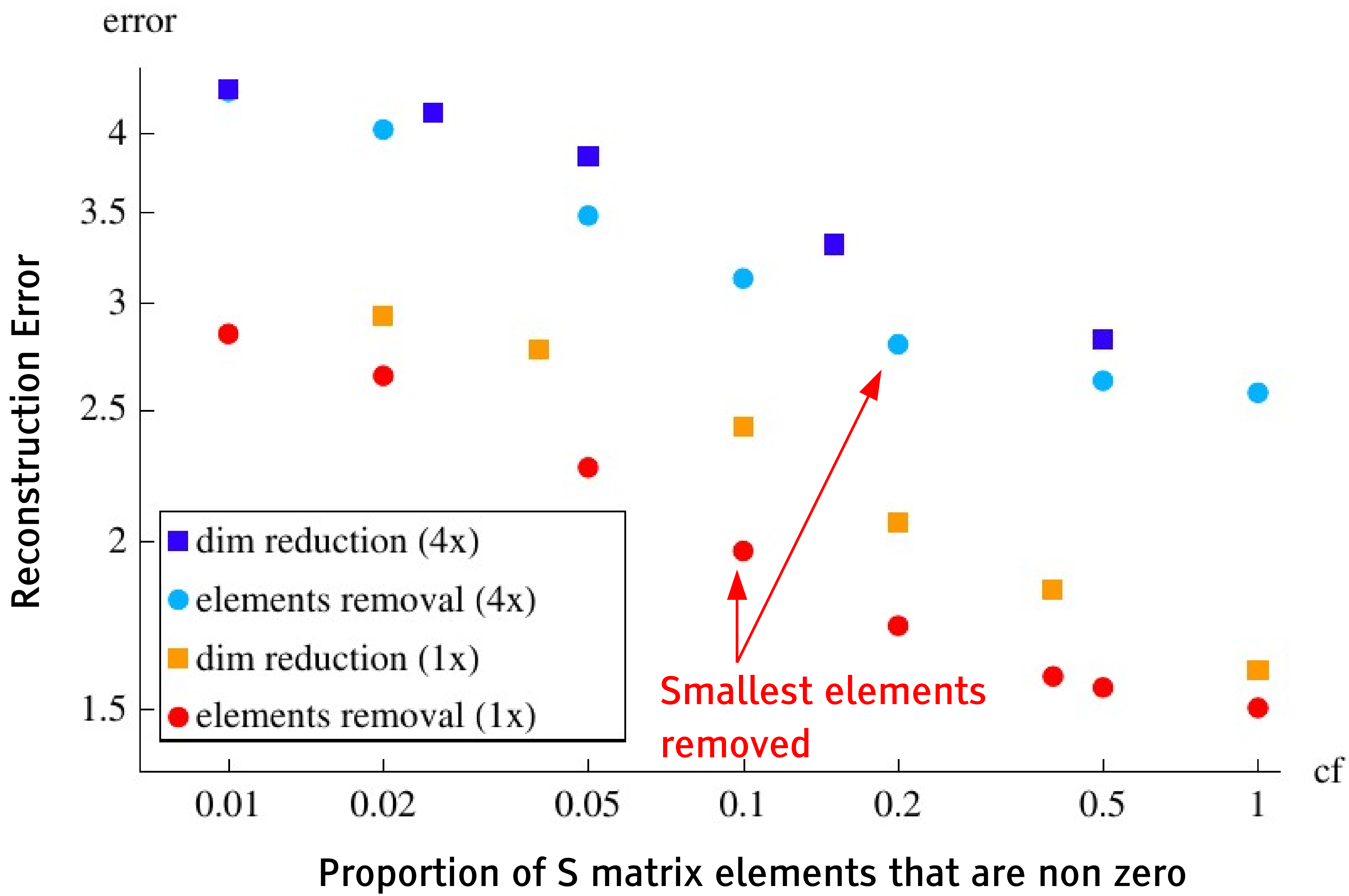


Learning ISTA (LISTA) vs ISTA/FISTA

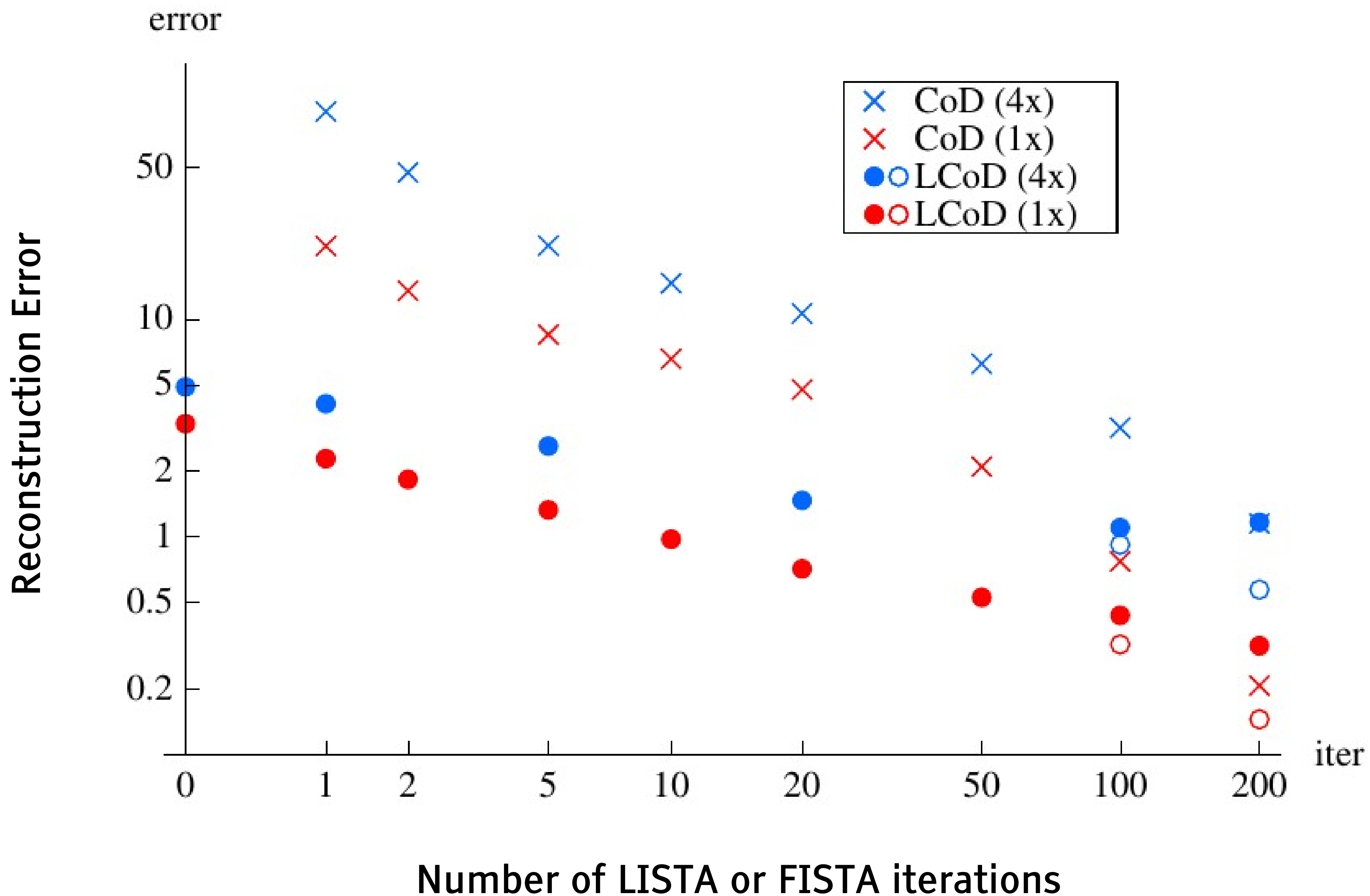


LISTA with partial mutual inhibition matrix

Y LeCun



Learning Coordinate Descent (LcoD): faster than LISTA

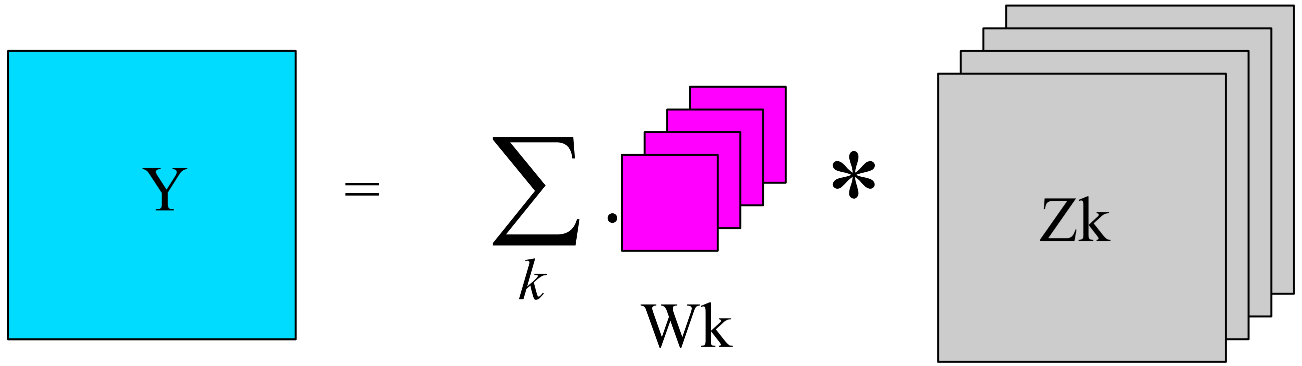


Convolutional Sparse Coding

- **Replace the dot products with dictionary element by convolutions.**
  - Input Y is a full image
  - Each code component Zk is a feature map (an image)
  - Each dictionary element is a convolution kernel

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

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



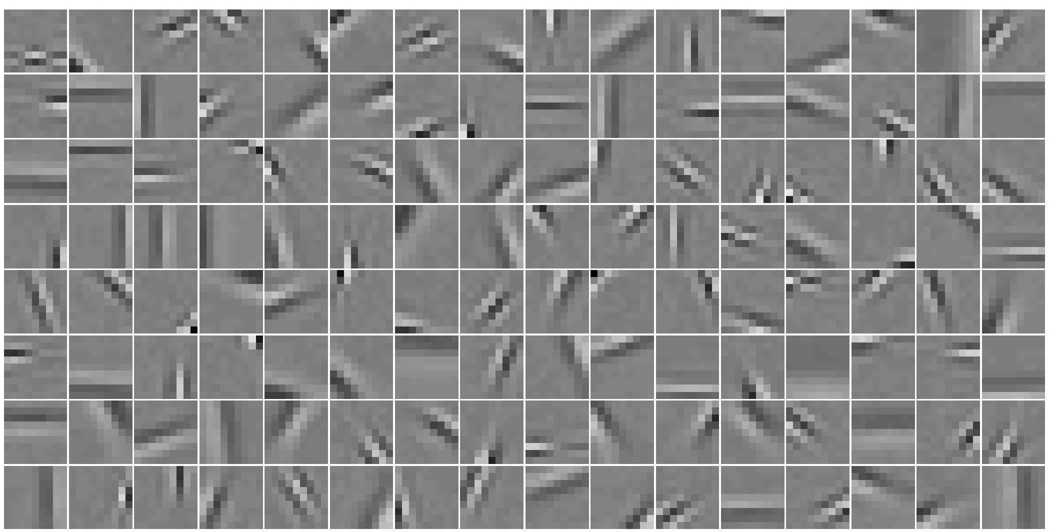
"deconvolutional networks" [Zeiler, Taylor, Fergus CVPR 2010]

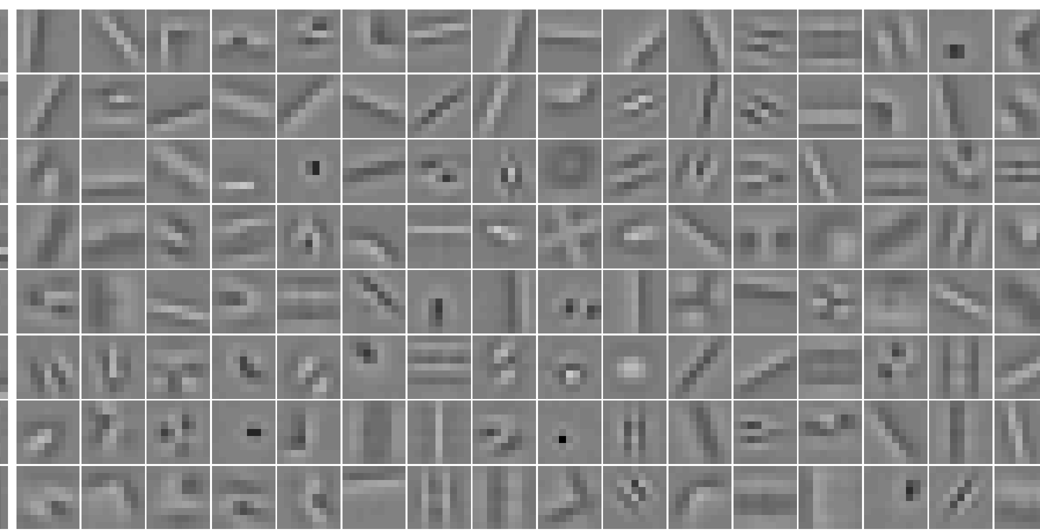
Convolutional PSD: Encoder with a soft sh() Function

Convolutional Formulation

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

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

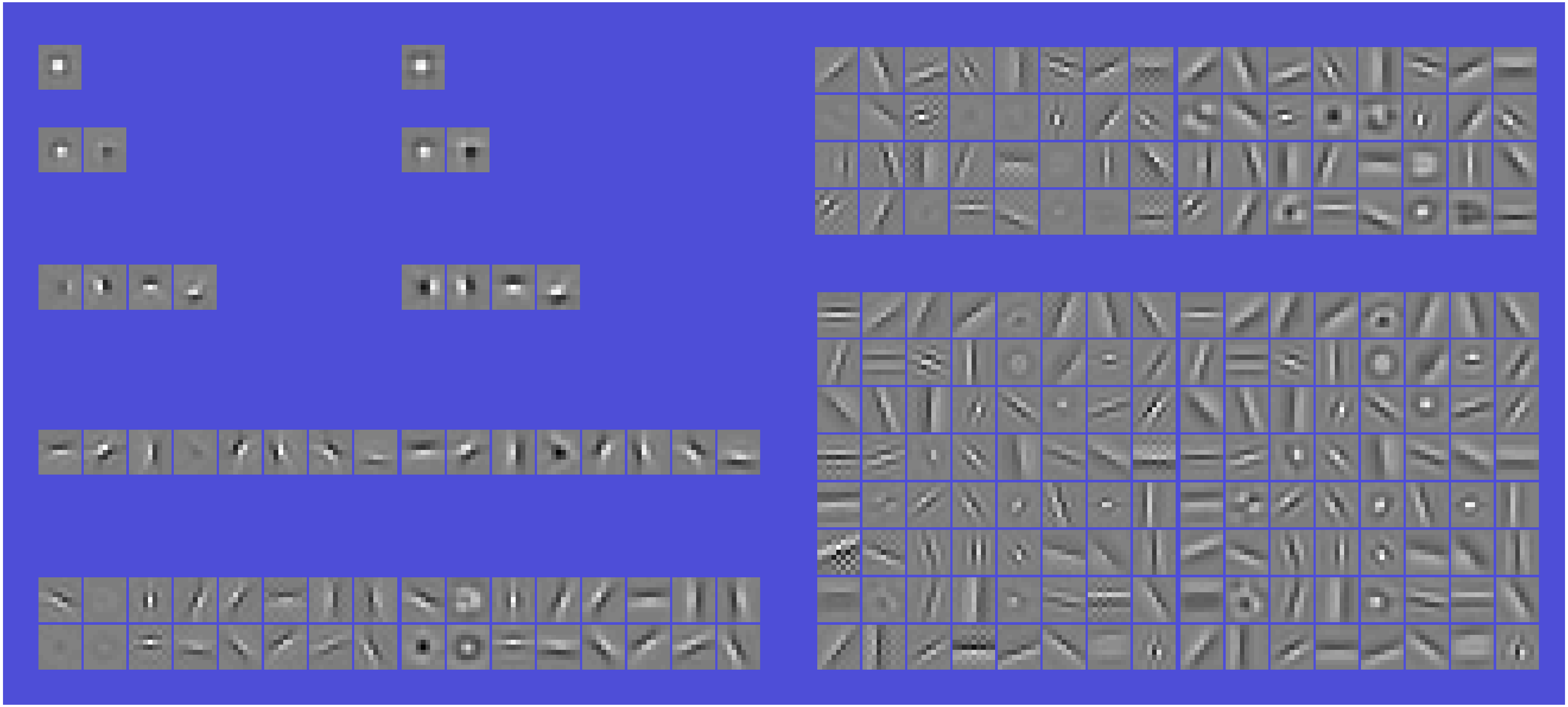




- ***PATCH*** based learning
- ***CONVOLUTIONAL*** learning

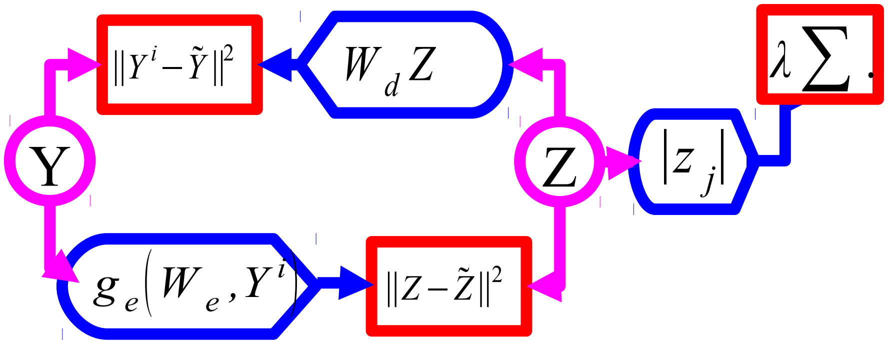
Convolutional Sparse Auto-Encoder on Natural Images

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



Using PSD to Train a Hierarchy of Features

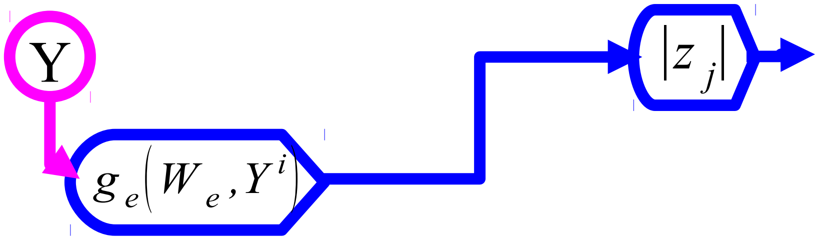
- Phase 1: train first layer using PSD



FEATURES

Using PSD to Train a Hierarchy of Features

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

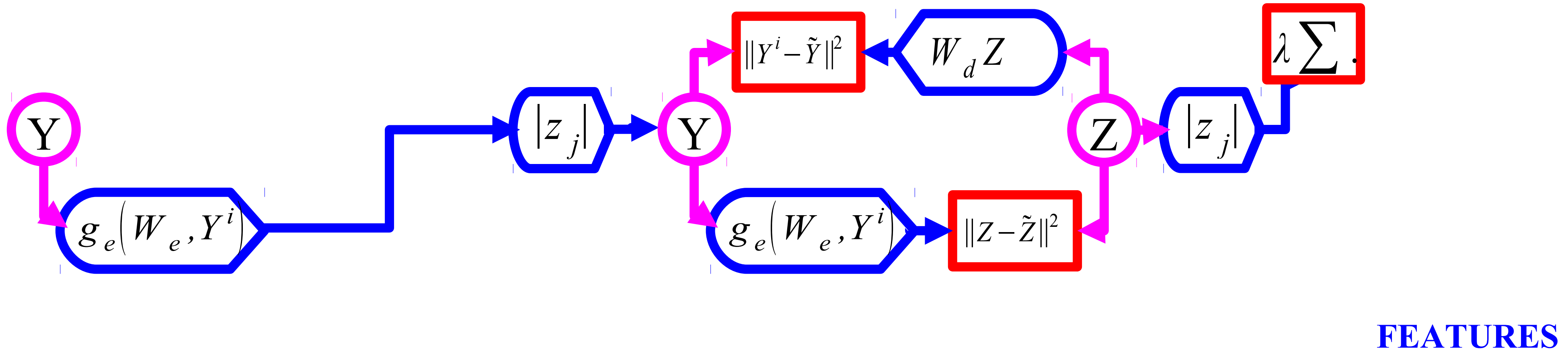


FEATURES

Using PSD to Train a Hierarchy of Features

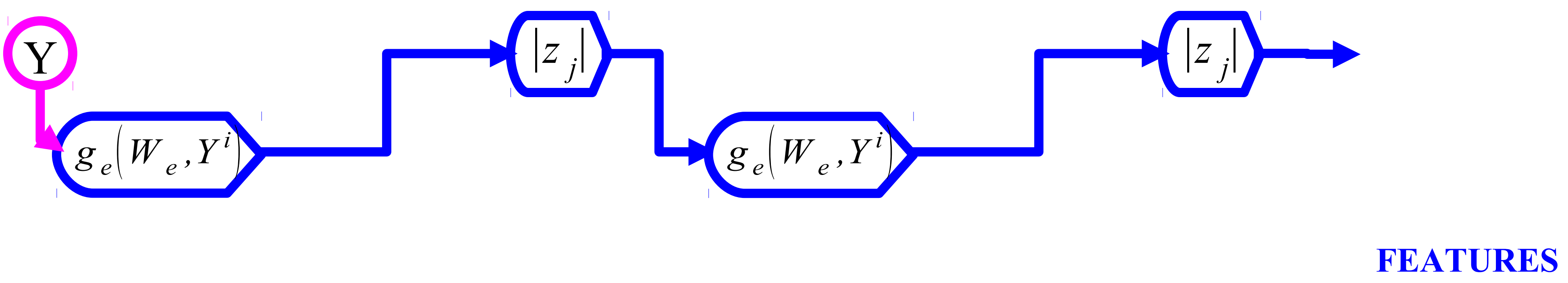
Y LeCun

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



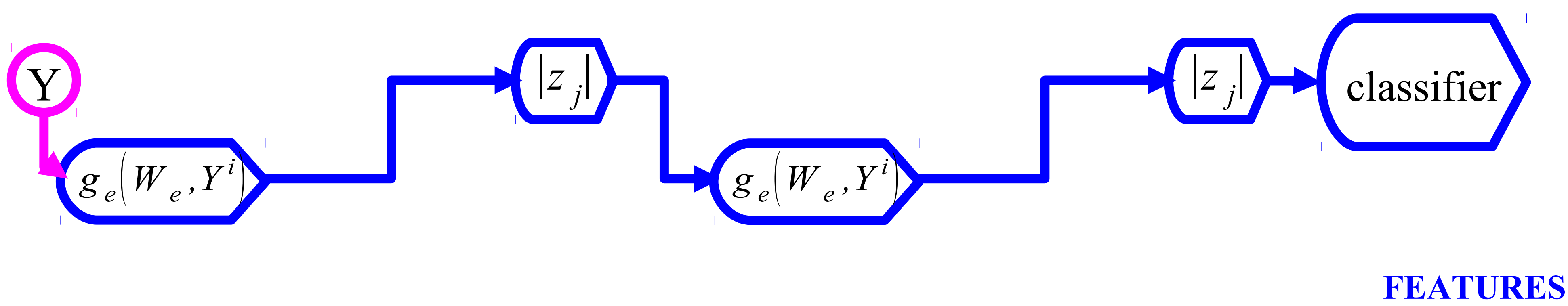
Using PSD to Train a Hierarchy of Features

- 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

- 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

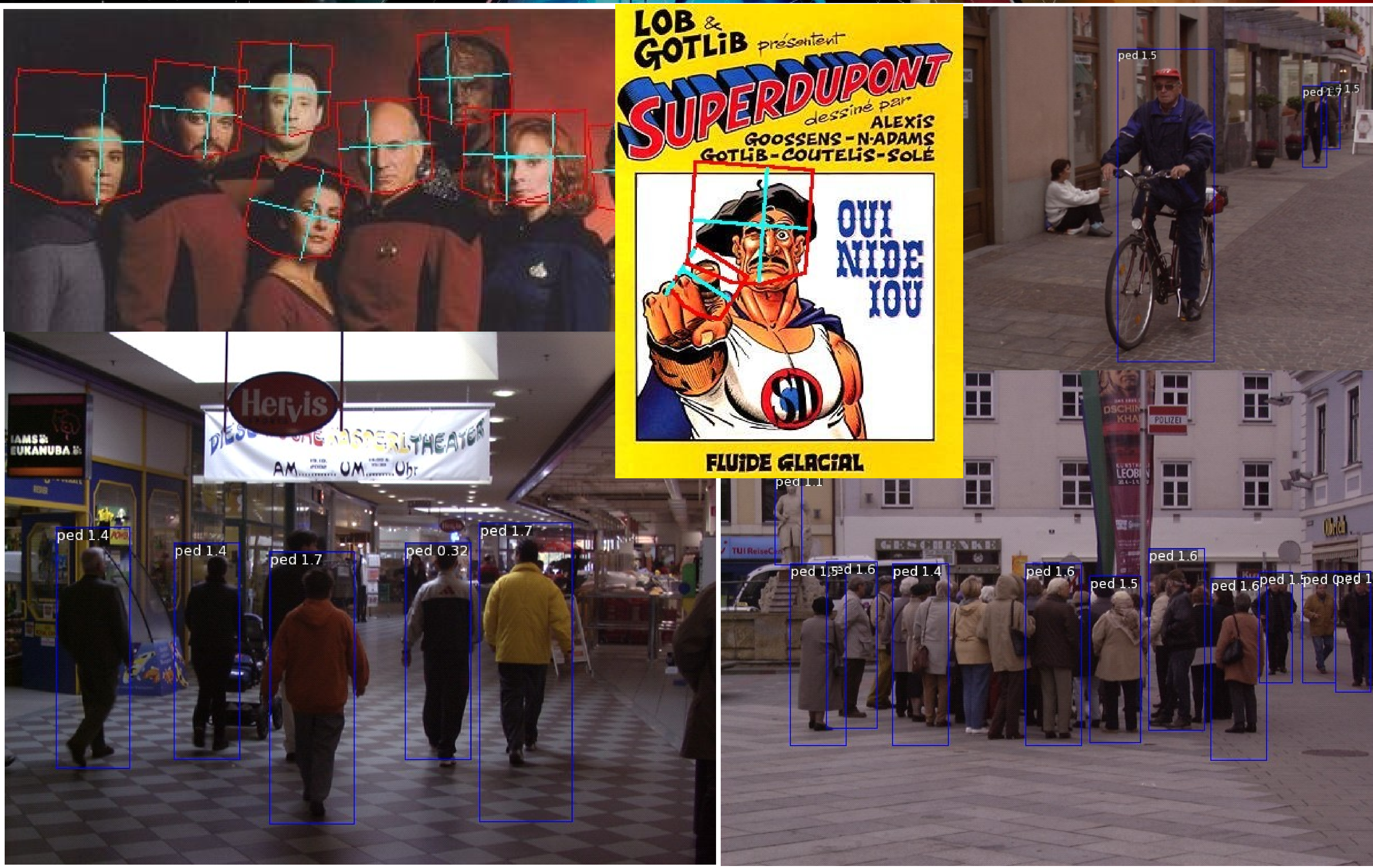


Unsupervised + Supervised

For

Pedestrian Detection

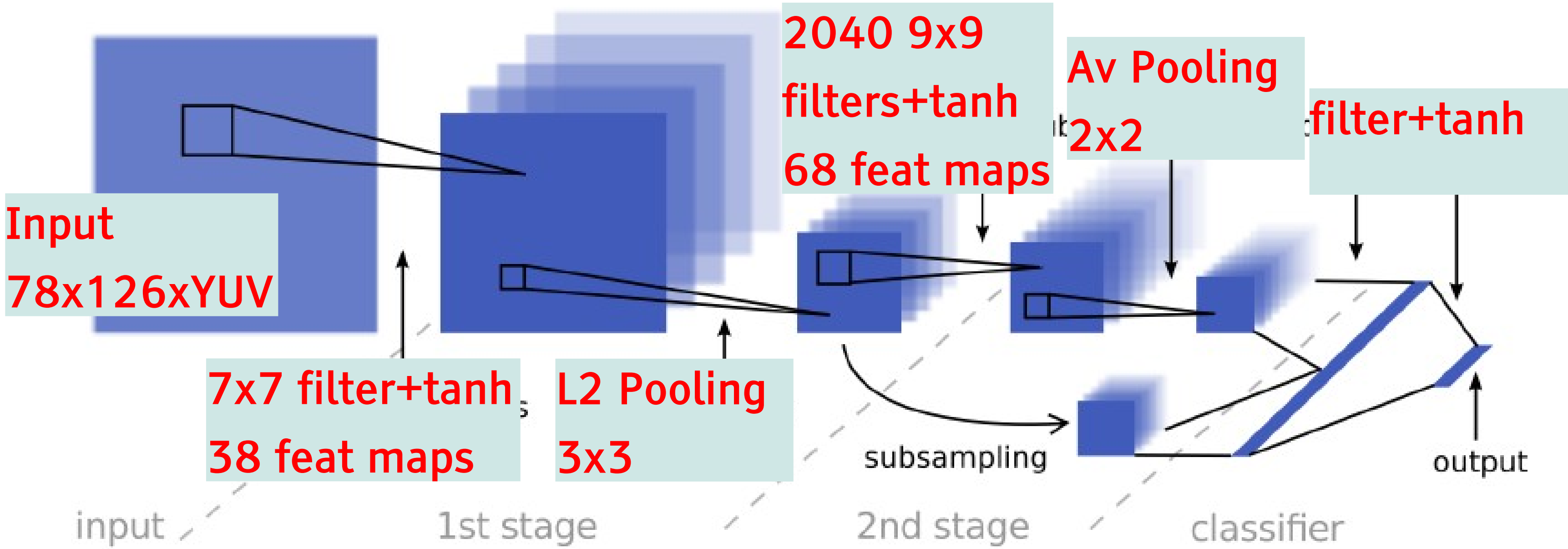
Pedestrian Detection, Face Detection



[Osadchy,Miller LeCun JMLR 2007],[Kavukcuoglu et al. NIPS 2010] [Sermanet et al. CVPR 2013]

ConvNet Architecture with Multi-Stage Features

- **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

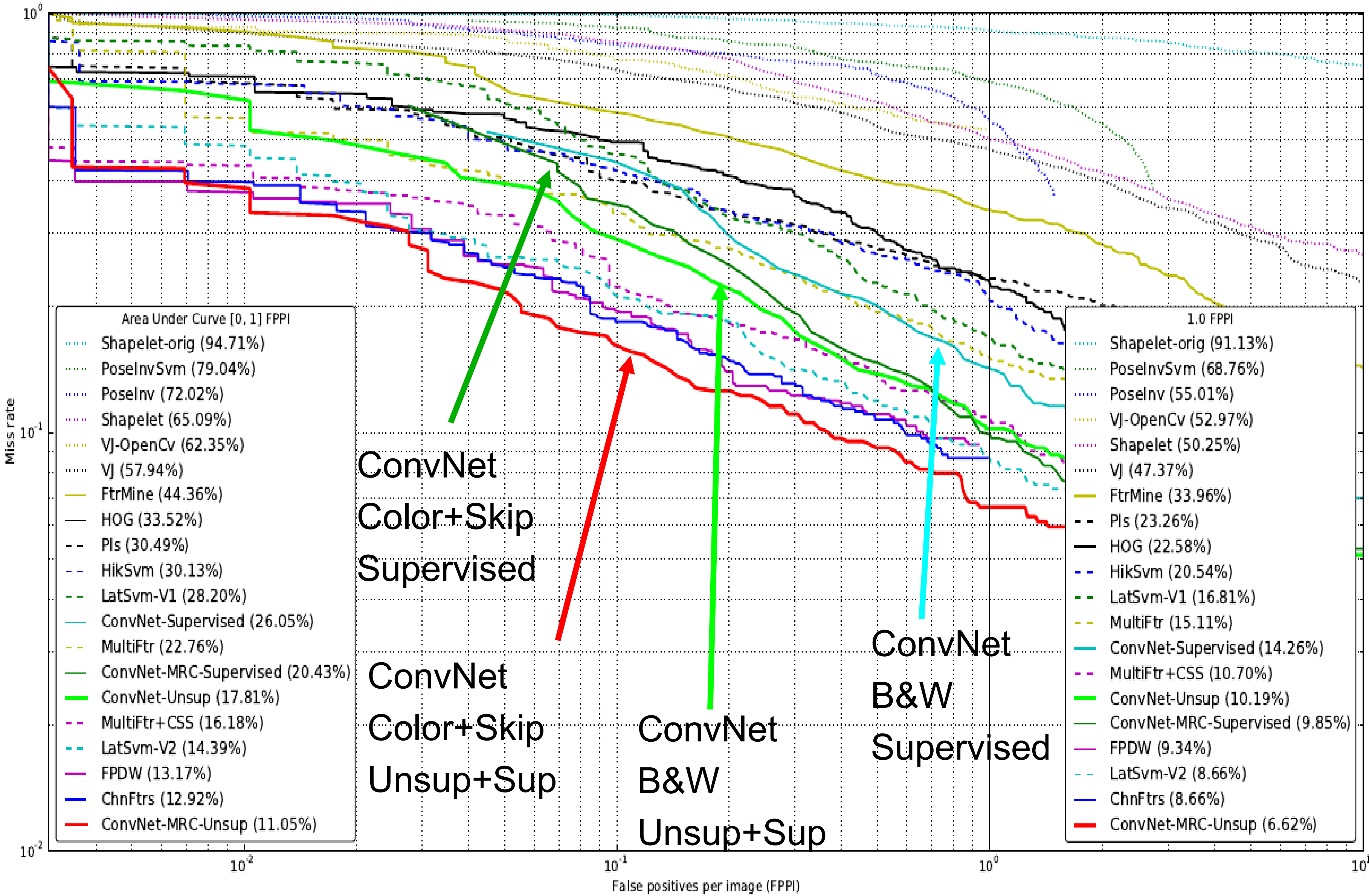


Task	Single-Stage features	Multi-Stage features	Improvement %
Pedestrians detection (INRIA)	14.26%	9.85%	31%
Traffic Signs classification (GTSRB) [33]	1.80%	0.83%	54%
House Numbers classification (SVHN) [32]	5.54%	5.36%	3.2%

[Sermanet, Chintala, LeCun CVPR 2013]

Pedestrian Detection: INRIA Dataset. Miss rate vs false positives

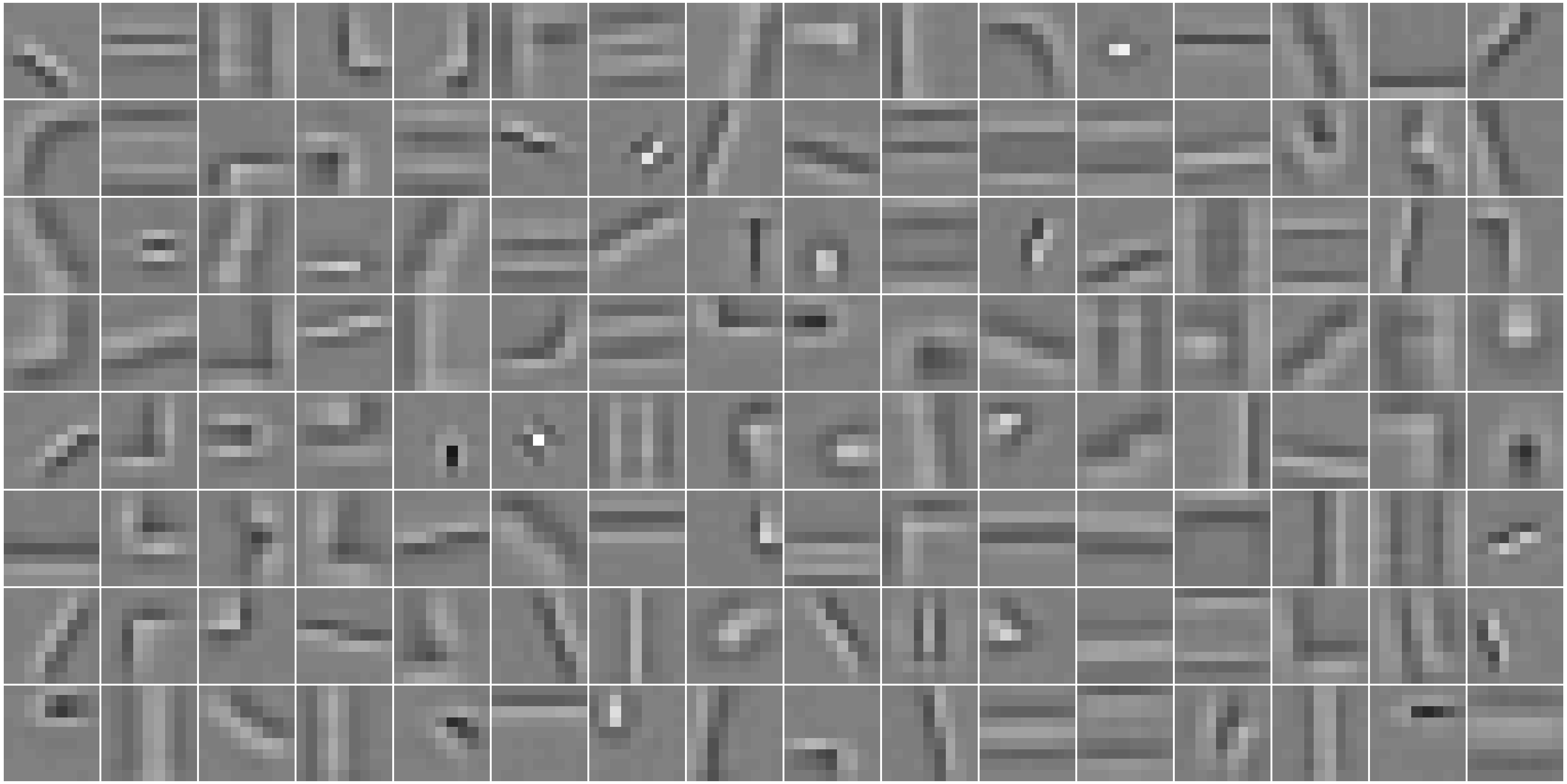
Y LeCun



[Kavukcuoglu et al. NIPS 2010] [Sermanet et al. ArXiv 2012]

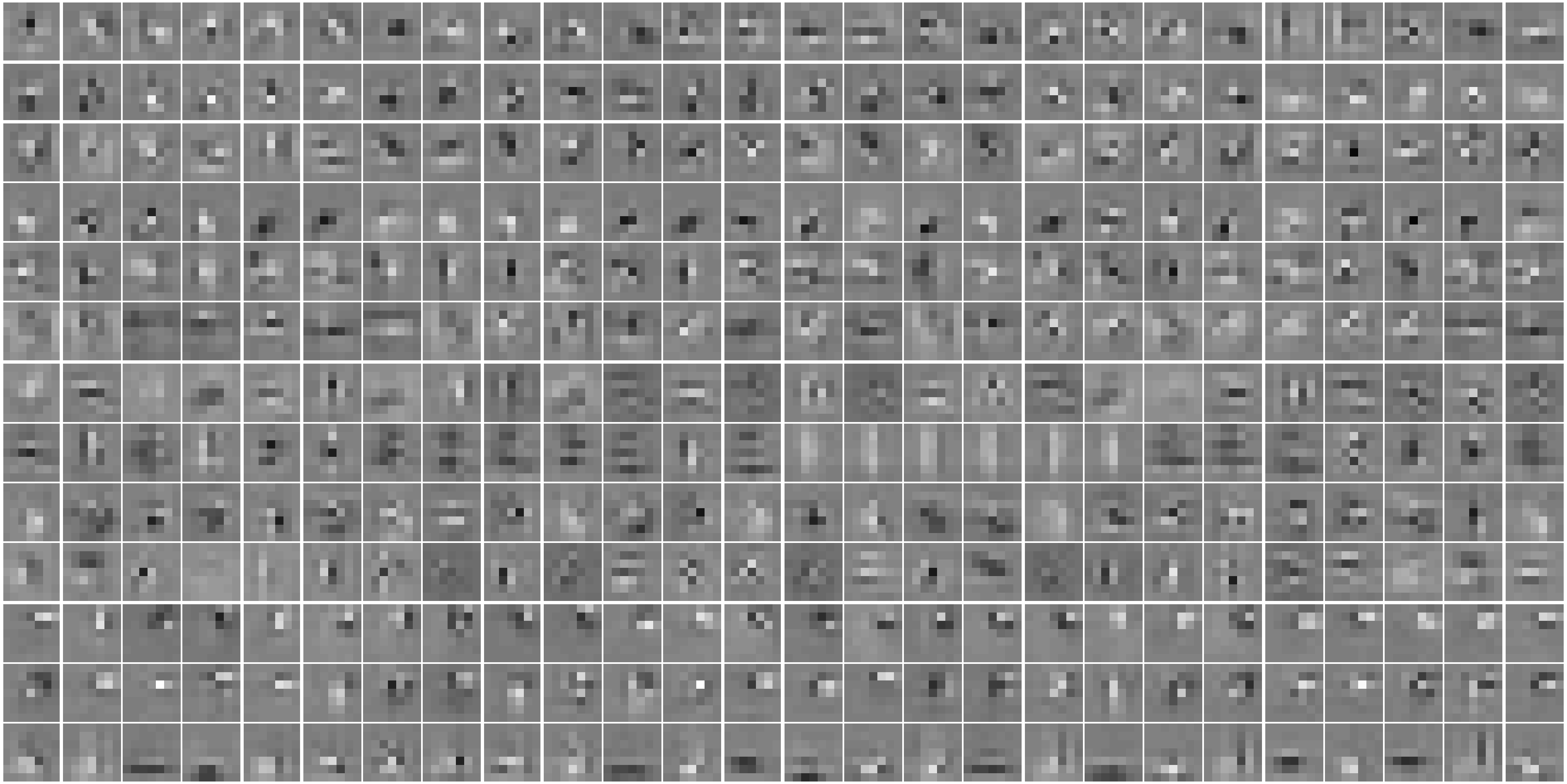
Unsupervised pre-training with convolutional PSD

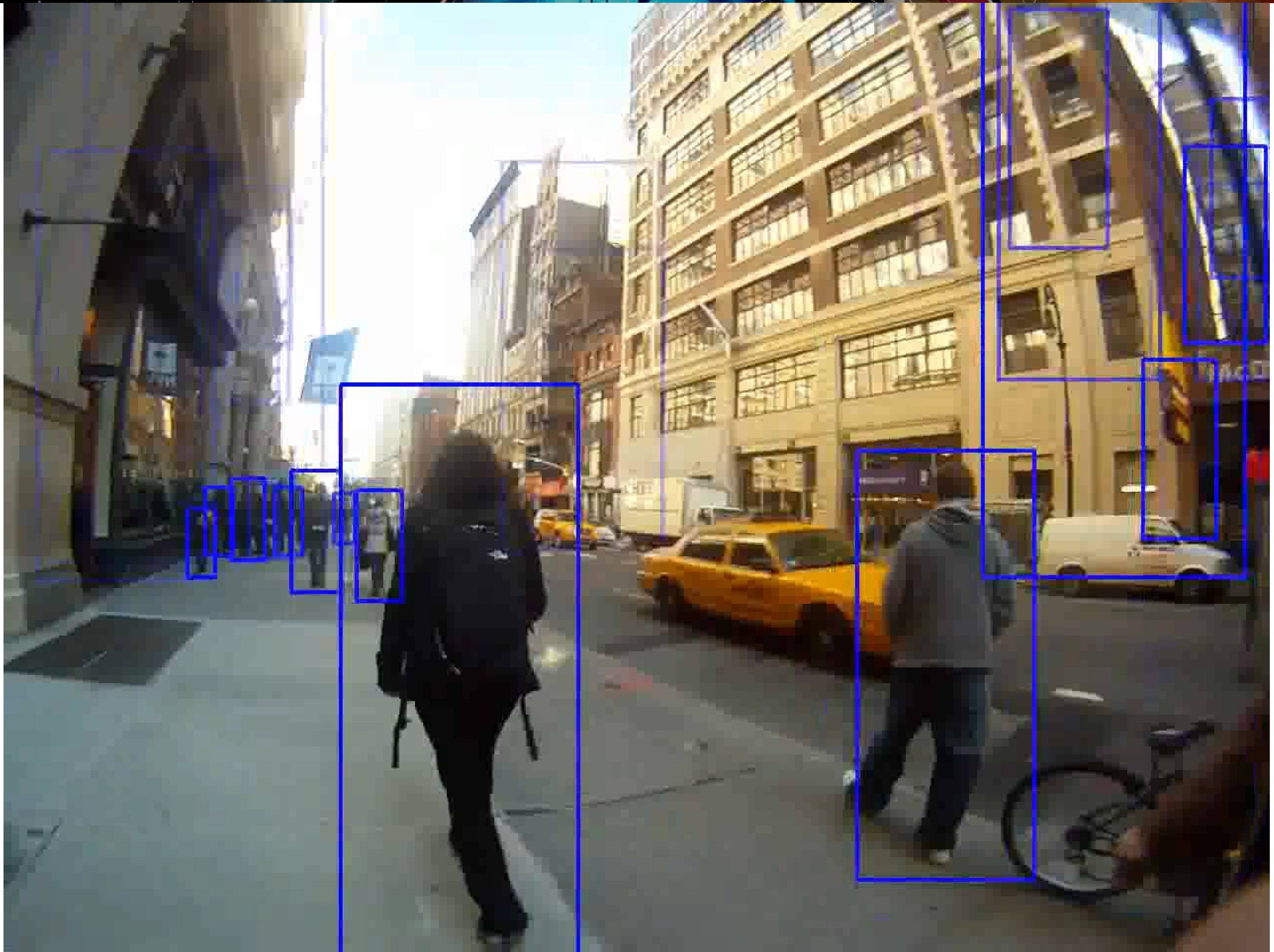
- 128 stage-1 filters on Y channel.
- Unsupervised training with convolutional predictive sparse decomposition



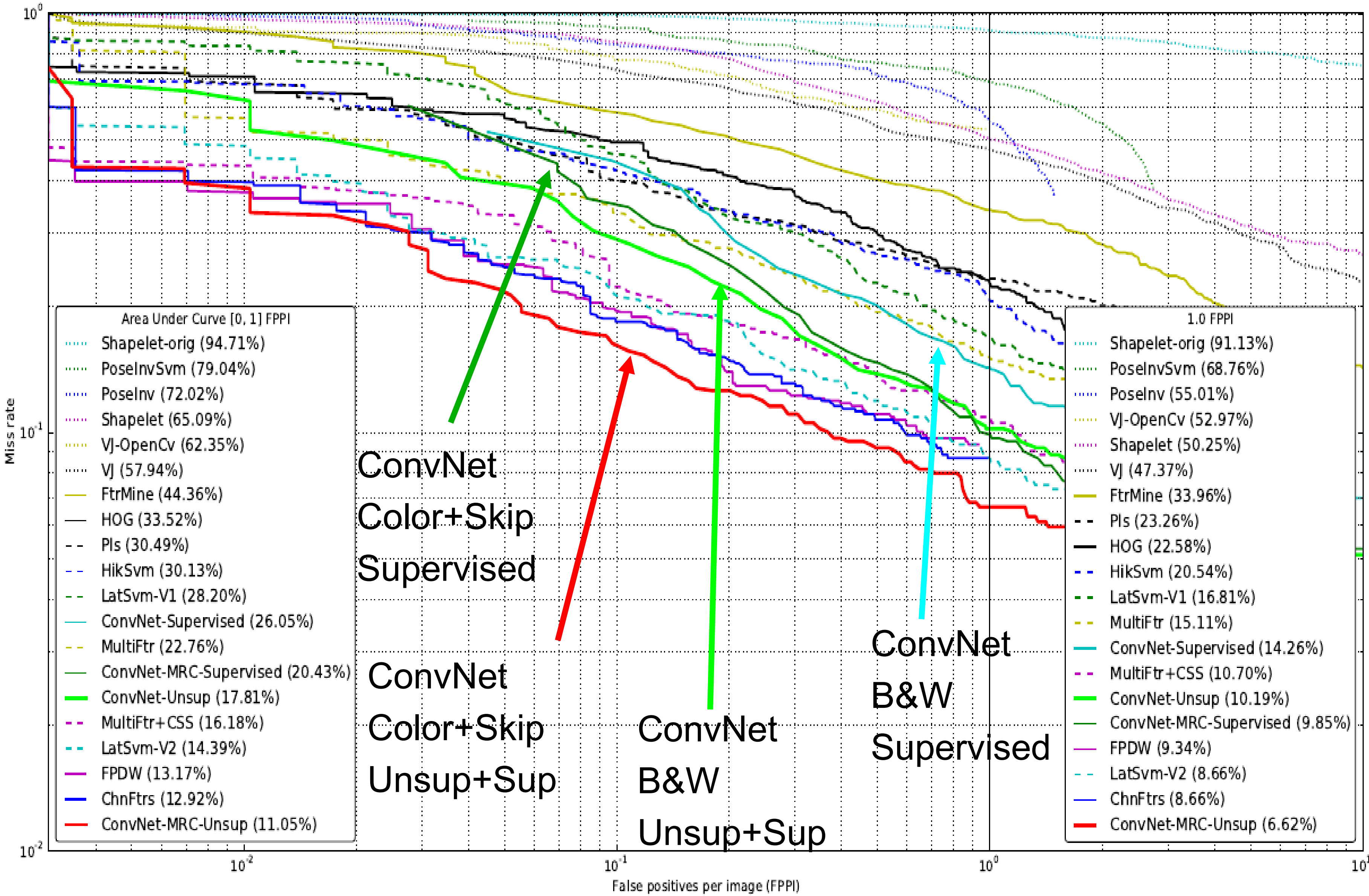
Unsupervised pre-training with convolutional PSD

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





Pedestrian Detection: INRIA Dataset. Miss rate vs false positives



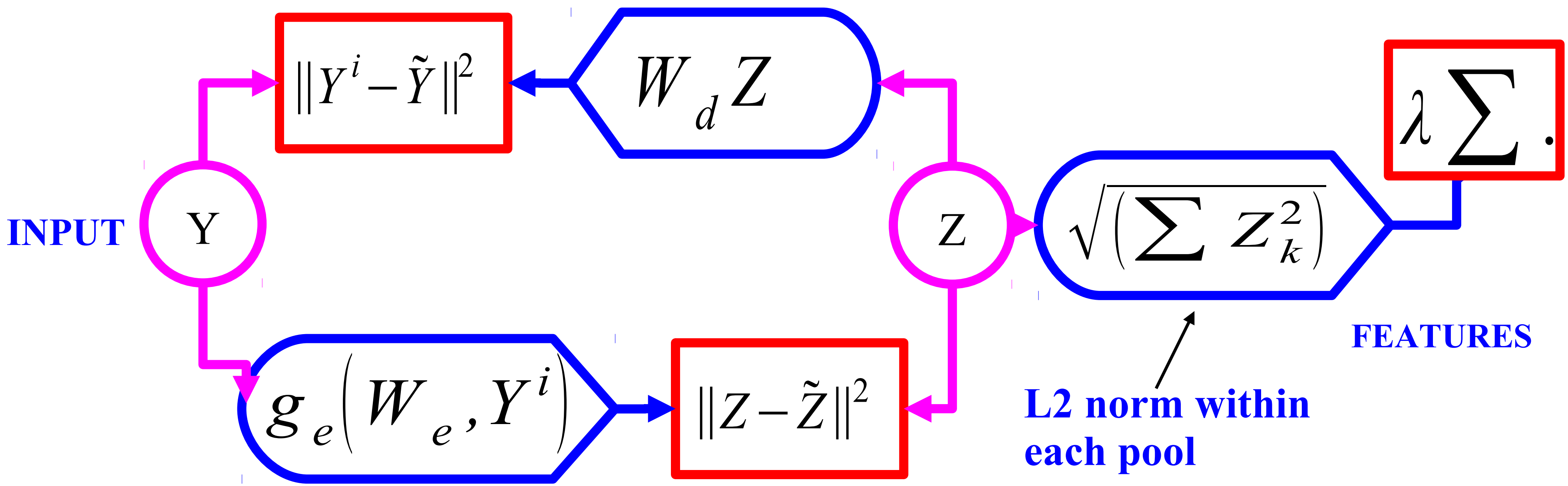
[Kavukcuoglu et al. NIPS 2010] [Sermanet et al. ArXiv 2012]

Unsupervised Learning: Invariant Features

Learning Invariant Features with L2 Group Sparsity

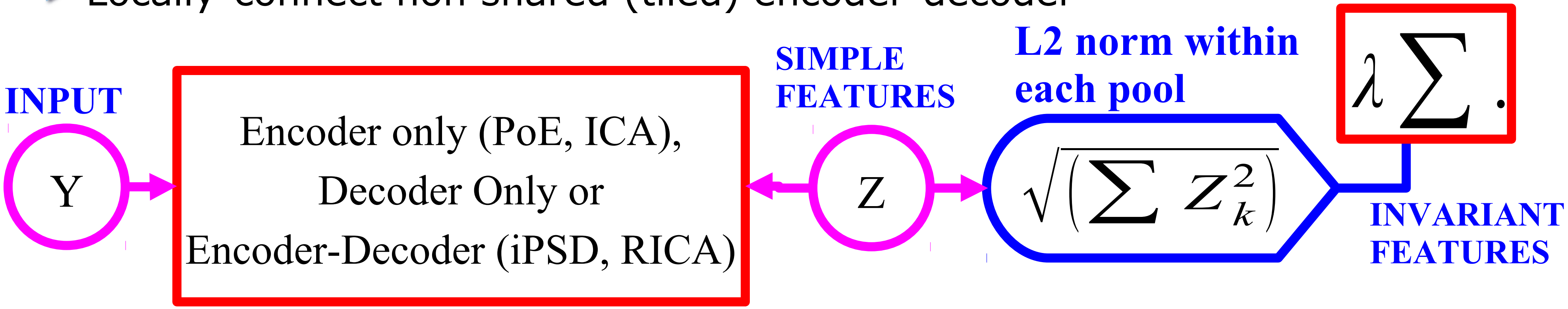
- Unsupervised PSD ignores the spatial pooling step.
- Could we devise a similar method that learns the pooling layer as well?
- Idea [Hyvarinen & Hoyer 2001]: group sparsity on pools of features
  - Minimum number of pools must be non-zero
  - Number of features that are on within a pool doesn't matter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

- **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.

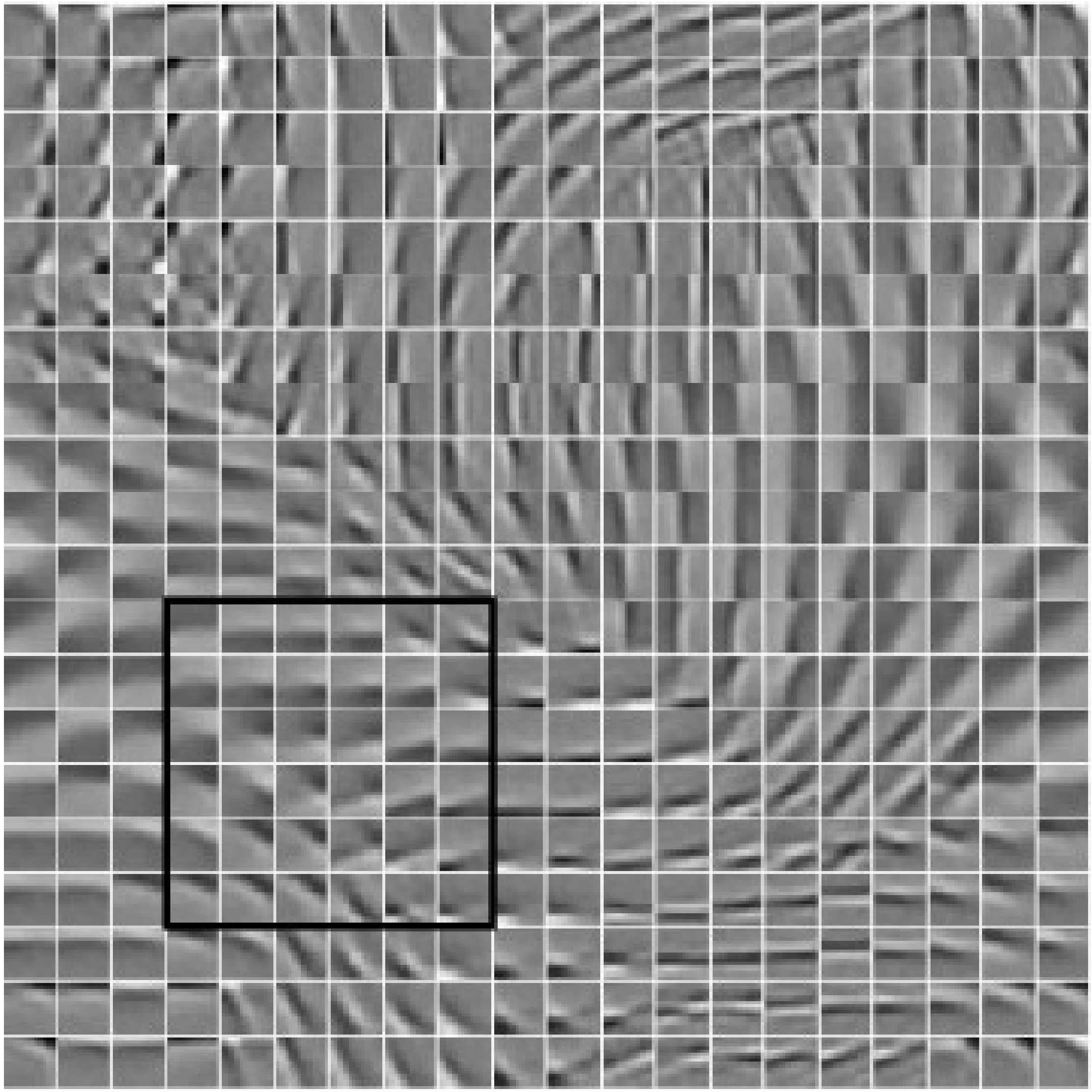


Image-level training, local filters but no weight sharing

- **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

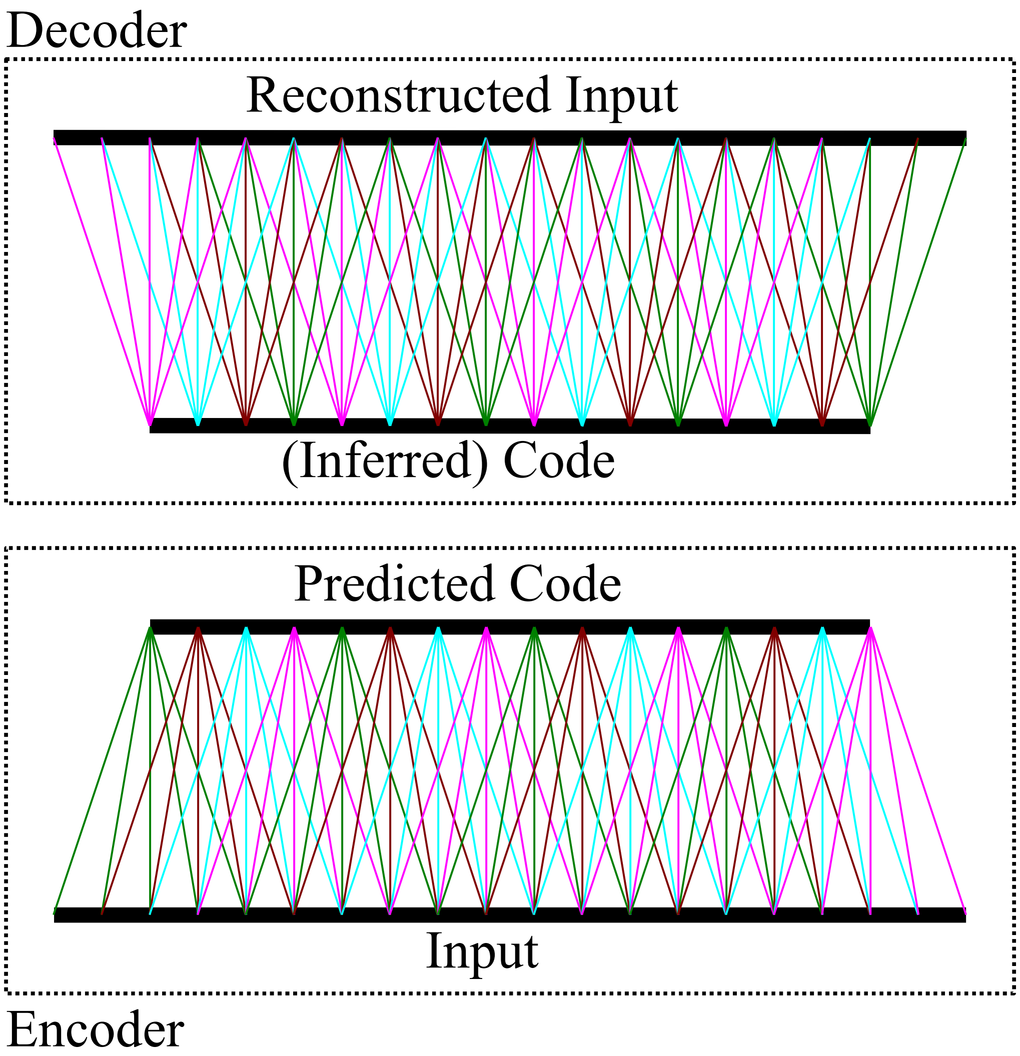
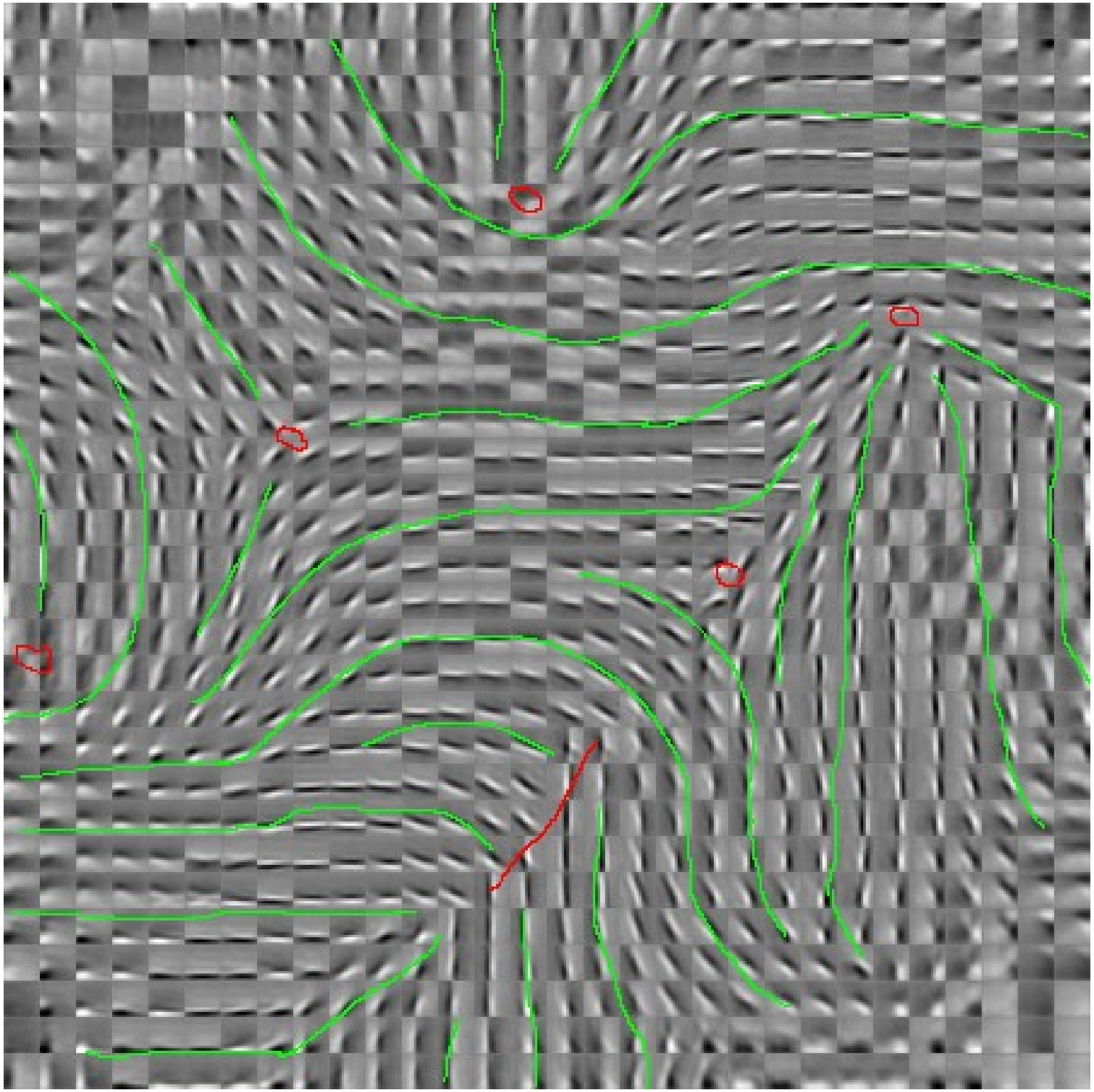
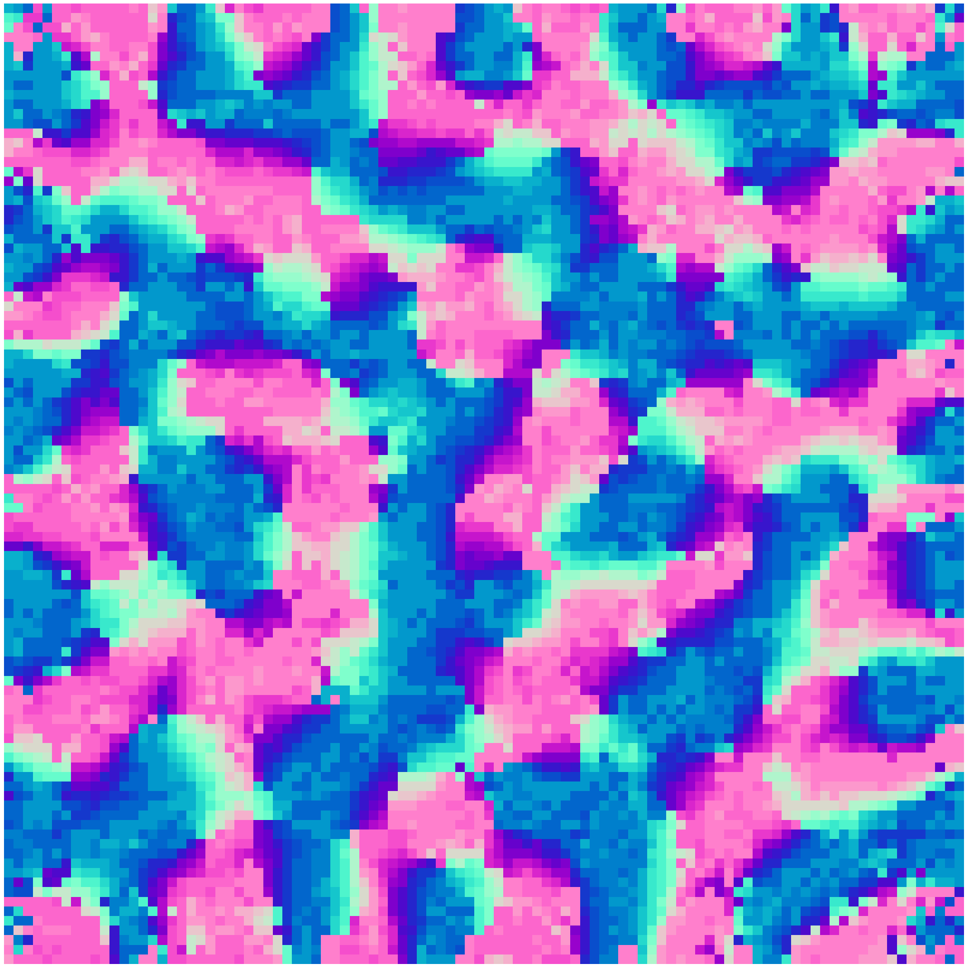


Image-level training, local filters but no weight sharing

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



Topographic Maps



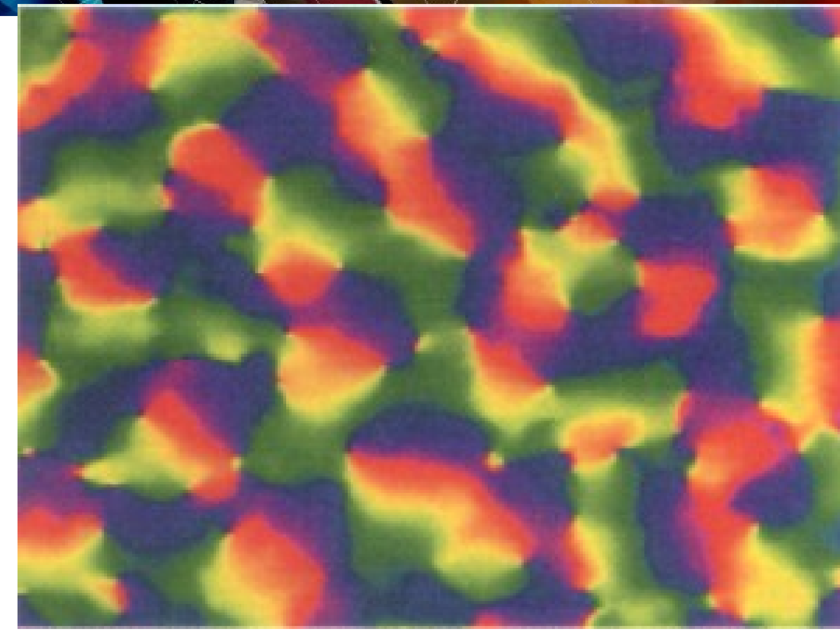
119x119 Image Input

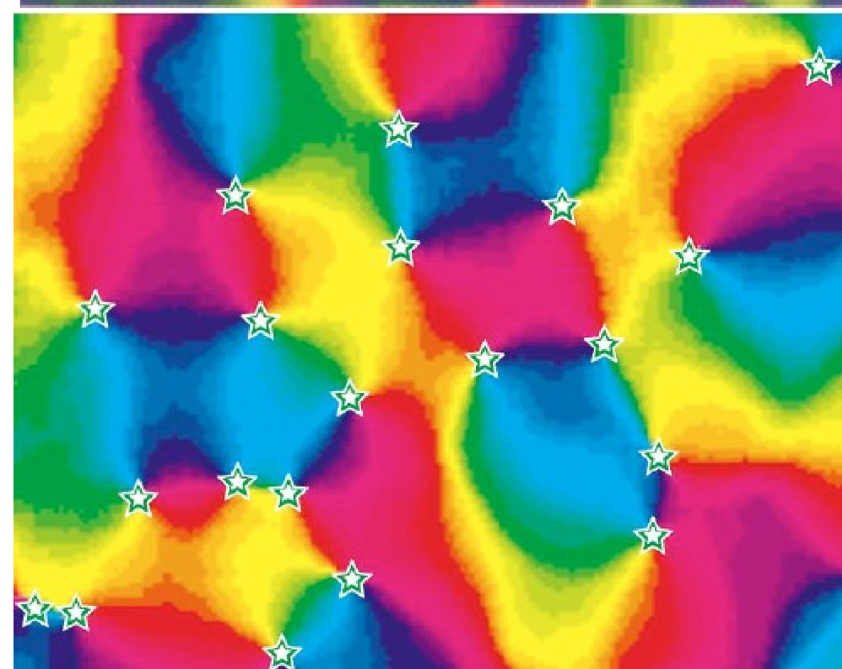
100x100 Code

20x20 Receptive field size

sigma=5

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

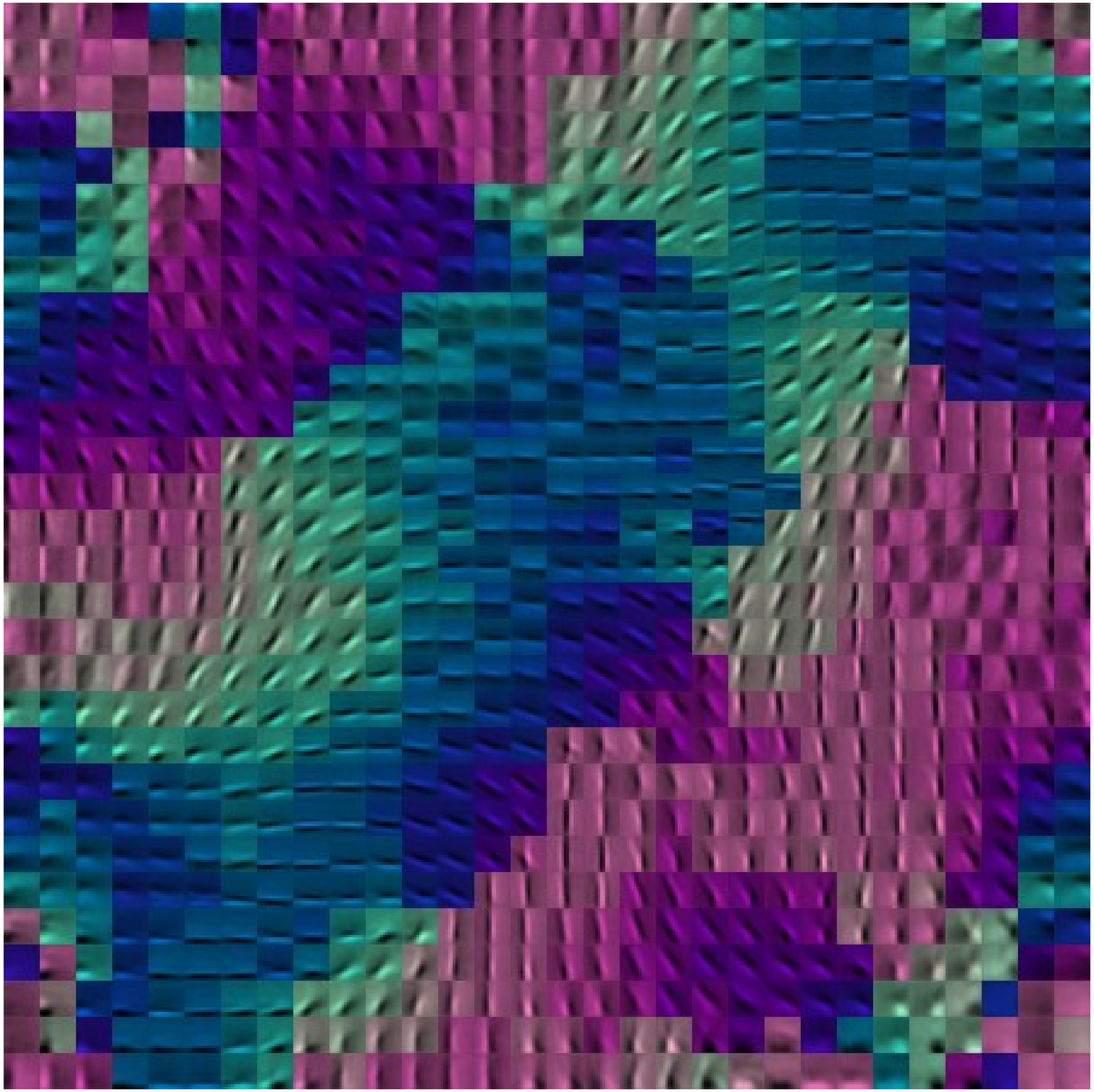




Michael C. Crair, et. al. The Journal of Neurophysiology Vol. 77 No. 6 June 1997, pp. 3381-3385 (**Cat**)

Image-level training, local filters but no weight sharing

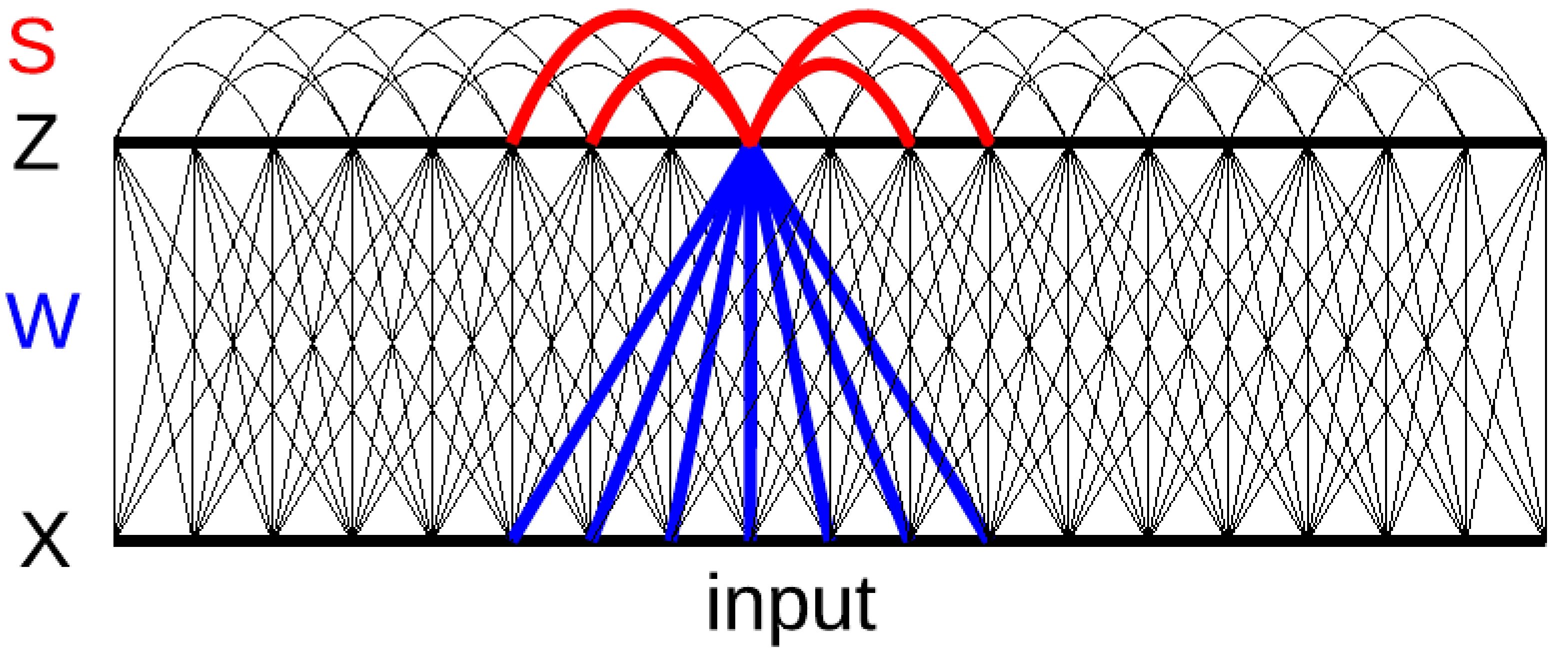
- Color indicates orientation (by fitting Gabors)



Invariant Features Lateral Inhibition

- Replace the L1 sparsity term by a lateral inhibition matrix
- Easy way to impose some structure on the sparsity

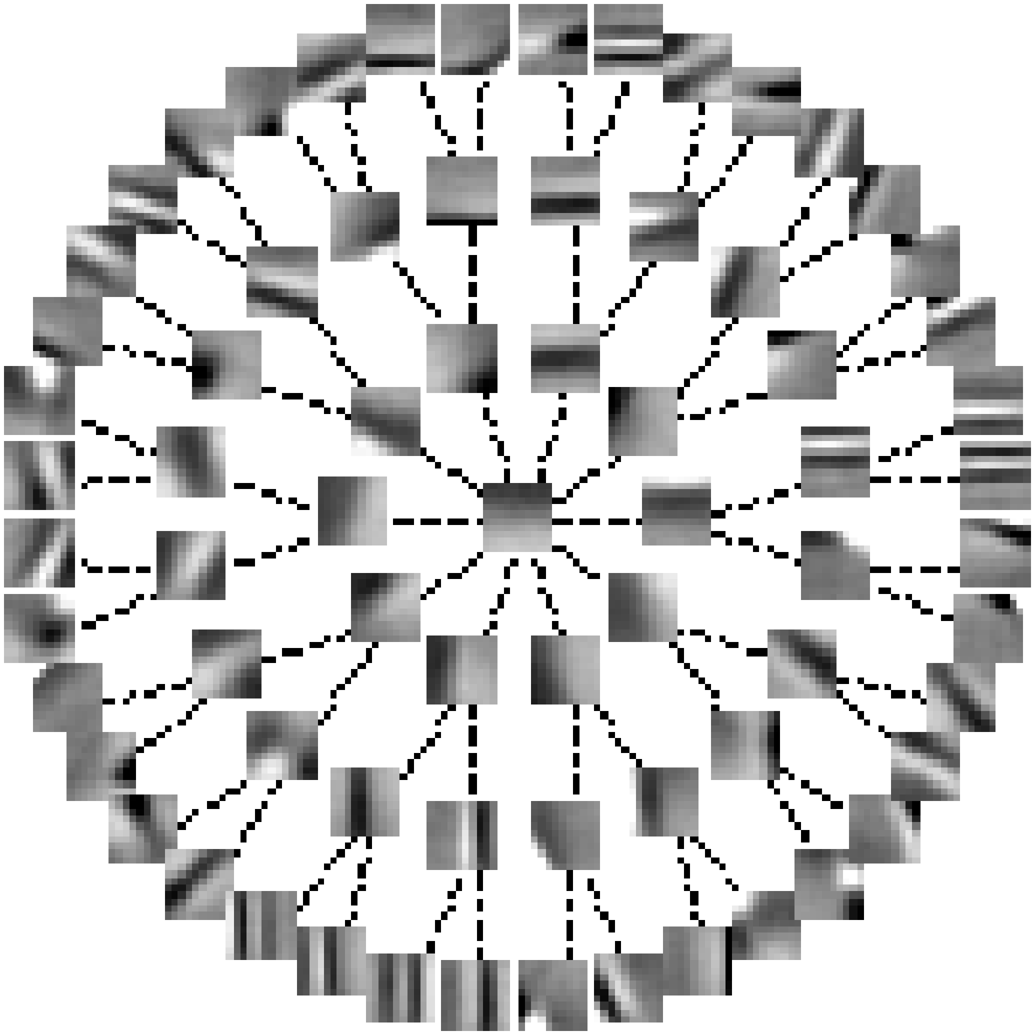
$$\min_{W,Z} \sum_{x \in X} ||Wz - x||^2 + |z|^T S|z|$$



[Gregor, Szlam, LeCun NIPS 2011]

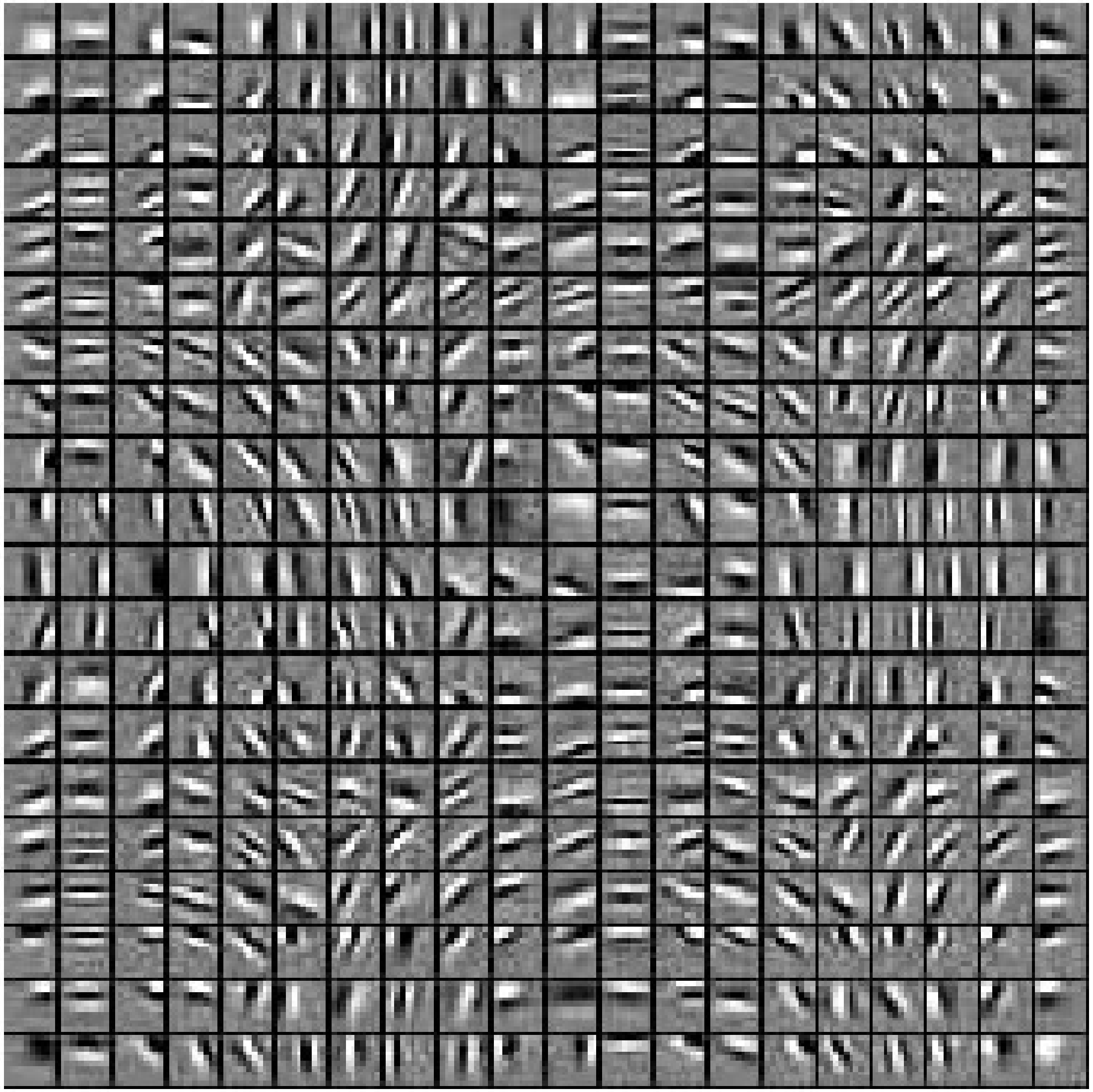
Invariant Features via Lateral Inhibition: Structured Sparsity

- Each edge in the tree indicates a zero in the S matrix (no mutual inhibition)
- Sij is larger if two neurons are far away in the tree



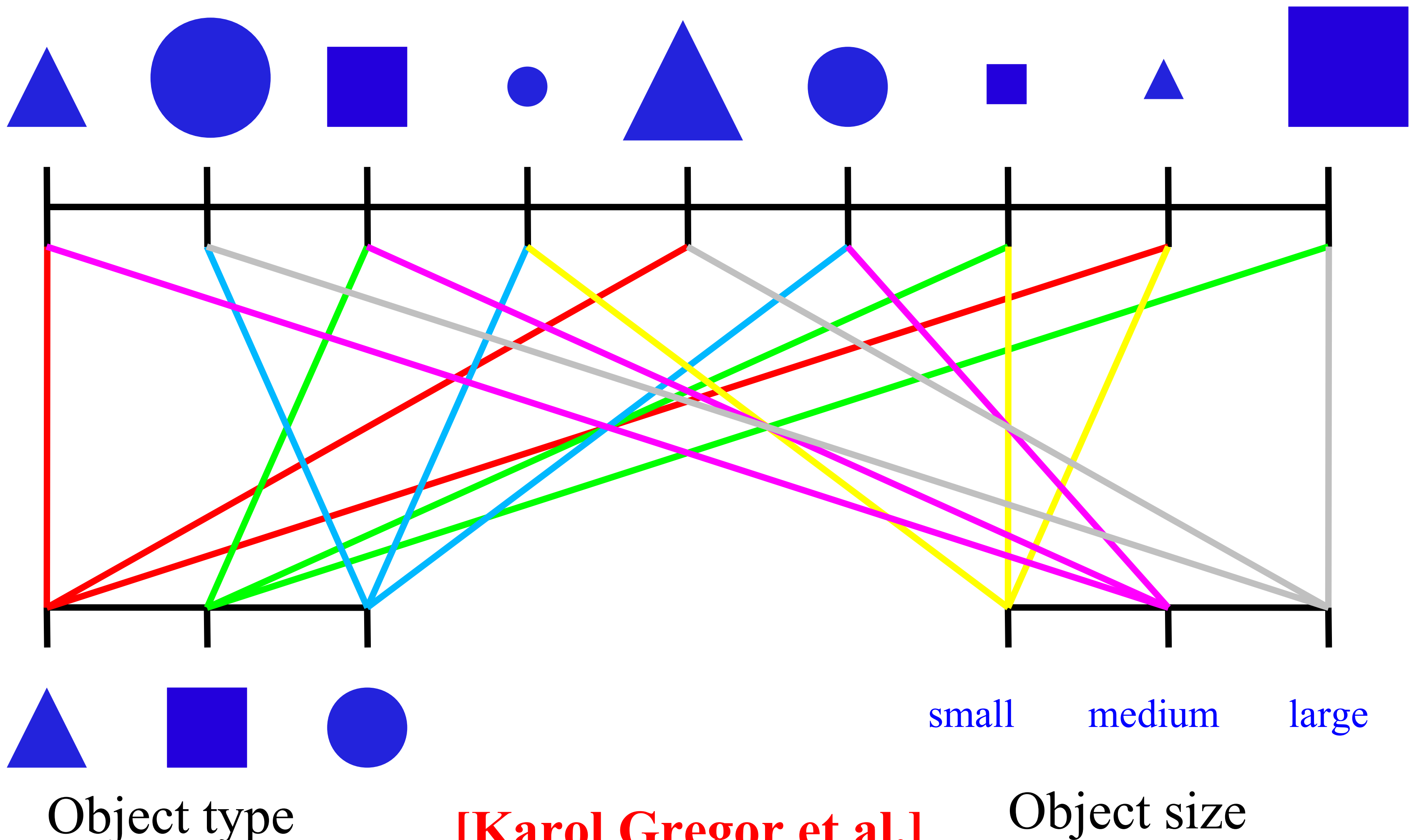
Invariant Features via Lateral Inhibition: Topographic Maps

- **Non-zero values in S form a ring in a 2D topology**
  - Input patches are high-pass filtered



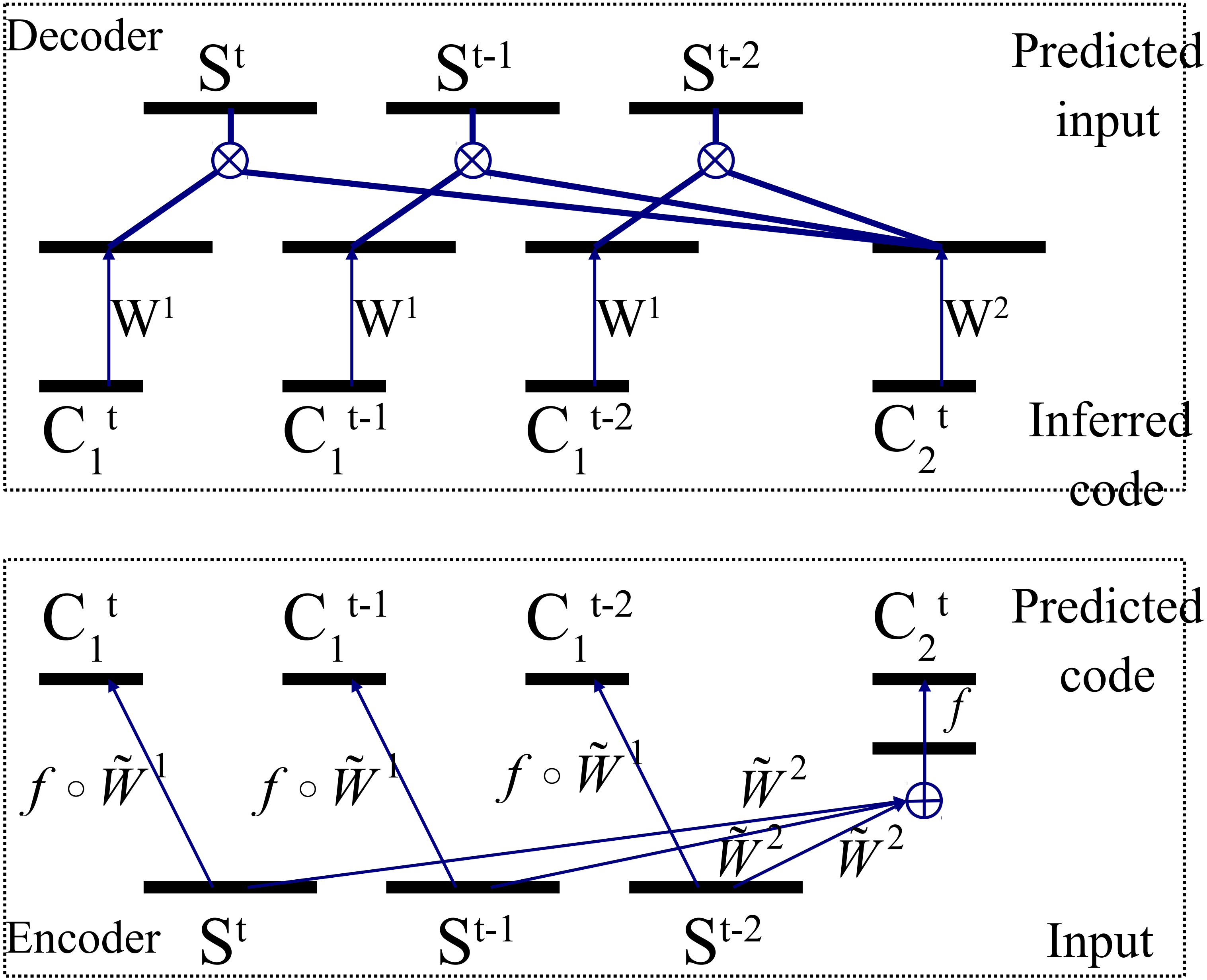
Invariant Features through Temporal Constancy

- **Object is cross-product of object type and instantiation parameters**
  - Mapping units [Hinton 1981], capsules [Hinton 2011]



[Karol Gregor et al.]

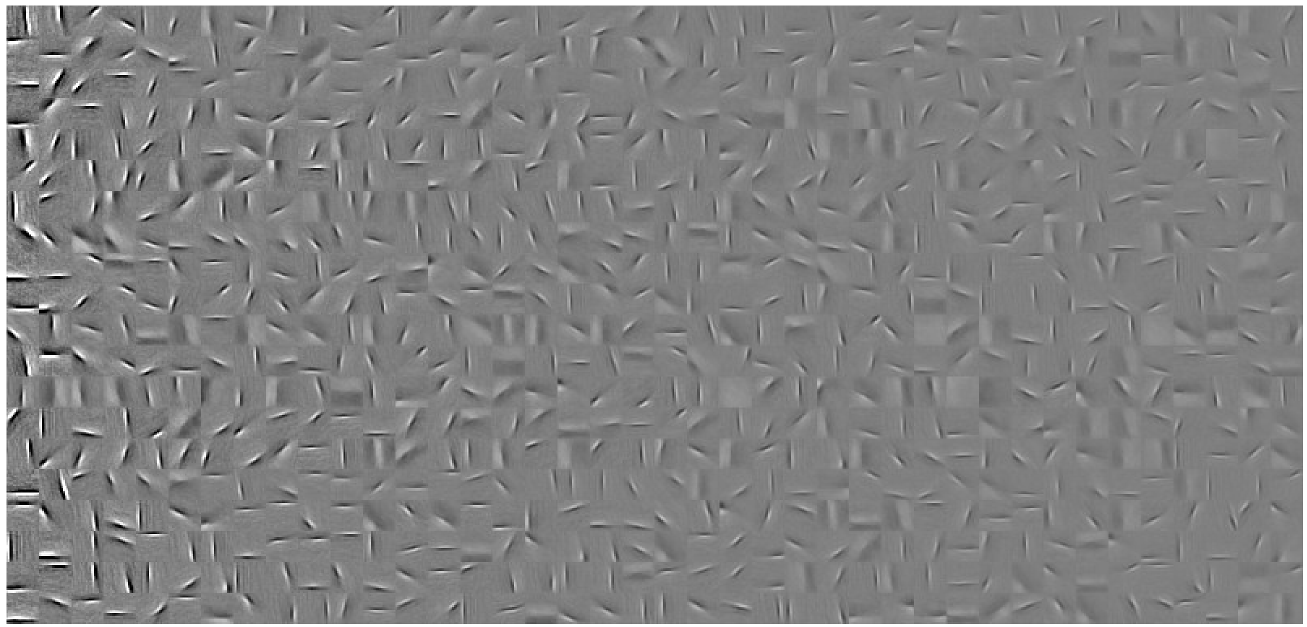
What-Where Auto-Encoder Architecture



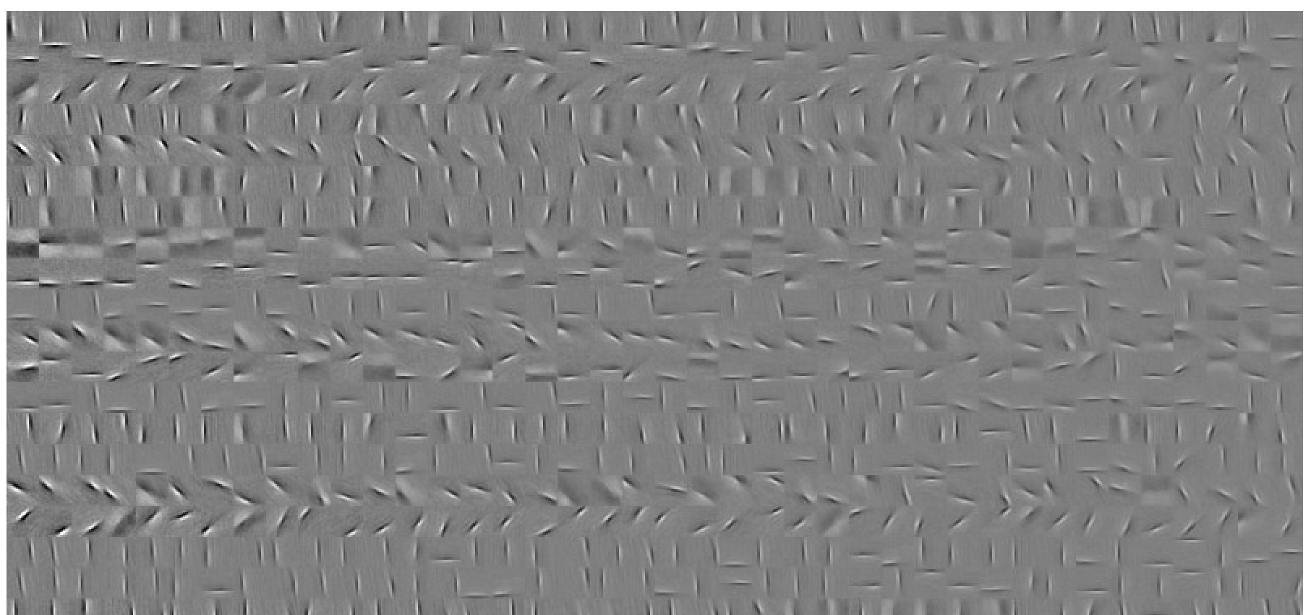
Low-Level Filters Connected to Each Complex Cell

Y LeCun

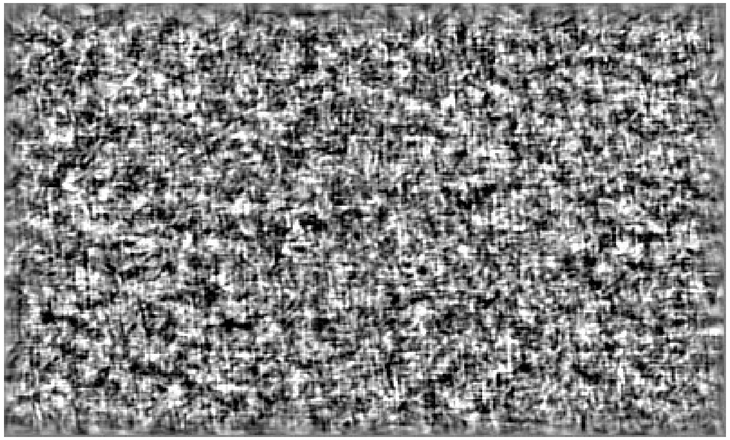
C1
(where)

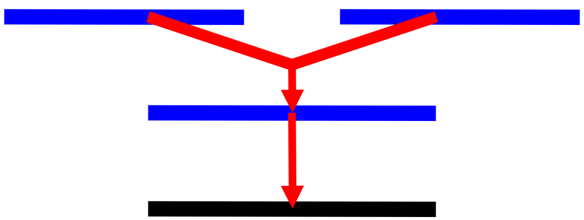


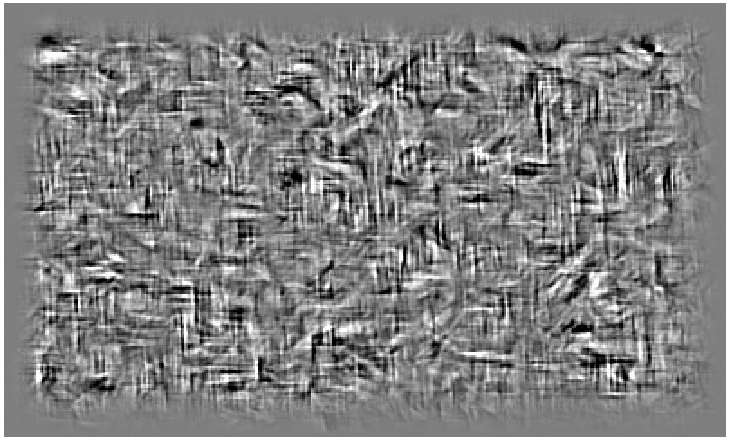
C2
(what)

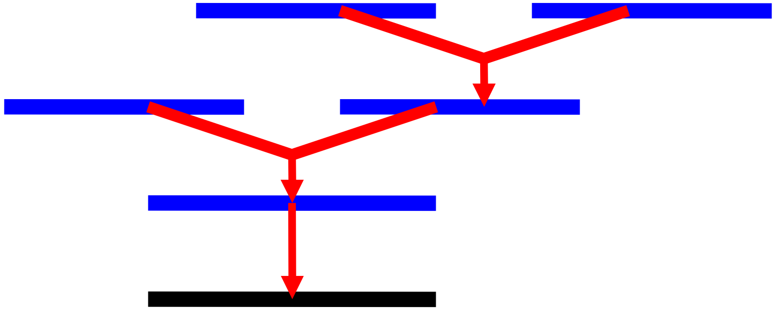


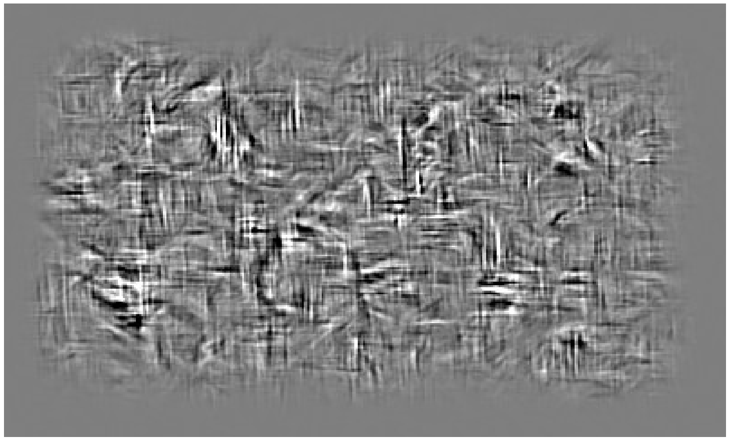
Generating images











Future Challenges

Future Challenges

- **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

- 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

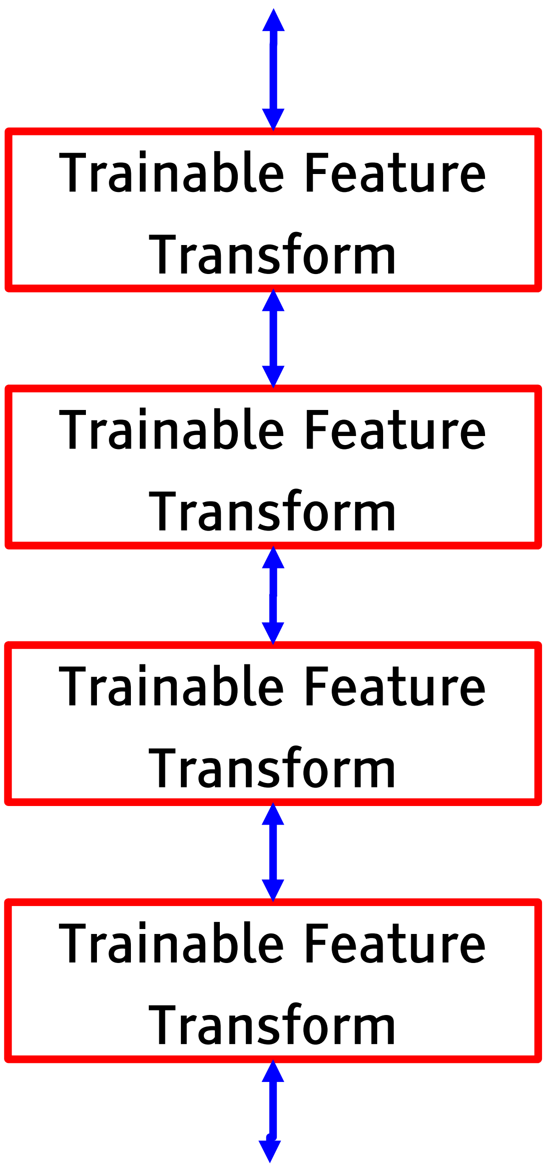
The Future: Unification
Feed-Forward & Feedback; Supervised & Unsupervised

- **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



The Graph of Deep Learning ↔ Sparse Modeling ↔ Neuroscience