L'apprentissage Profond: Une Révolution en Intelligence Artificielle

Leçon Inaugurale au Collège de France

Chaire Annuelle 2015-2016

Informatique et Sciences Numériques

Yann Le Cun

Facebook AI Research,

Center for Data Science, NYU

Courant Institute of Mathematical Sciences, NYU

http://yann.lecun.com

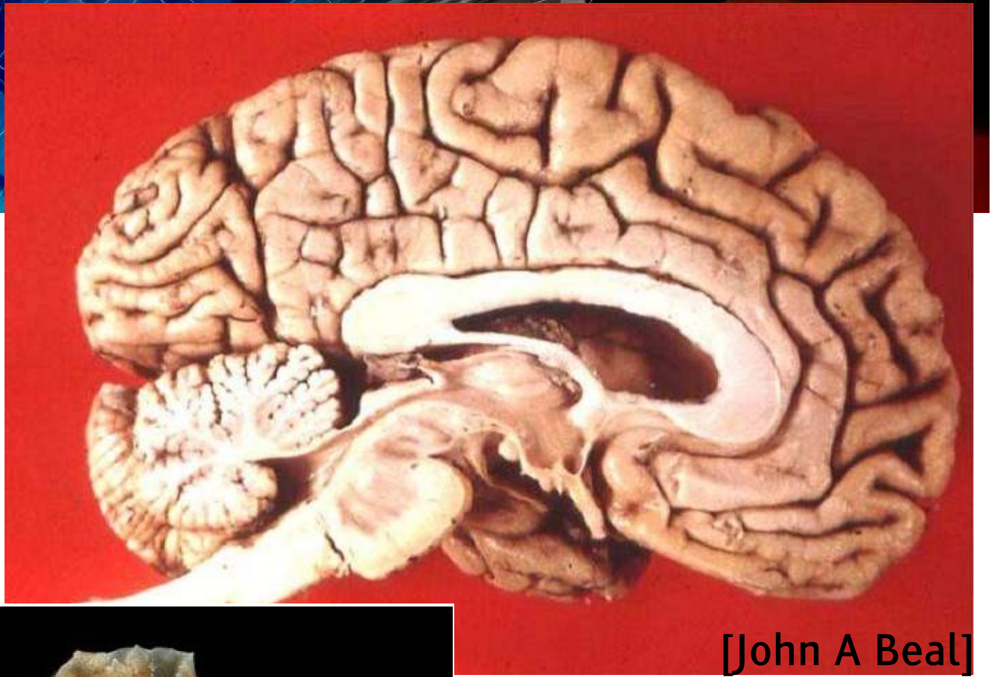
Should We Copy the Brain to Build Intelligent Machines?

- **The brain is an existence proof of intelligent machines**
  - The way birds and bats were an existence proof of heavier-than-air flight
- **Shouldn't we just copy it?**
  - Like Clément Ader copied the bat?
- **The answer is no!**
- **But we should draw inspiration from it.**

L'Avion III de Clément Ader, 1897
(Musée du CNAM, Paris)
His “Eole” took off from the ground in 1890, 13 years before the Wright Brothers.

The Brain

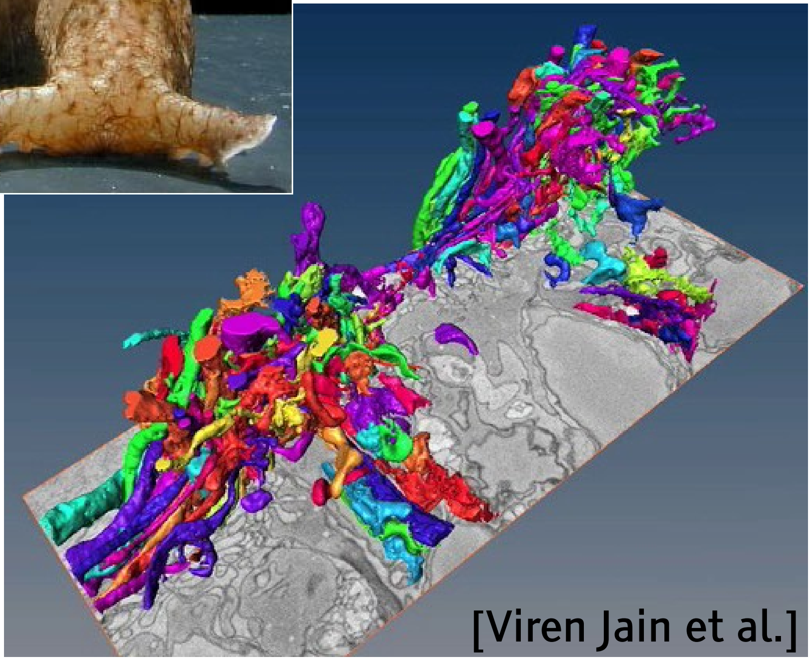
- 85×10^9 neurons
- 10^4 synapses/neuron → 10^{15} synapses
- 1.4 kg, 1.7 liters
- Cortex: 2500 cm^2 , 2mm thick
- 180,000 km of “wires”
- 250 million neurons per mm^3 .


[John A Beal]

- All animals can learn

- Learning is inherent to intelligence

- Learning modifies the efficacies of synapses
  - Learning causes synapses to strengthen or weaken, to appear or disappear.


[Viren Jain et al.]

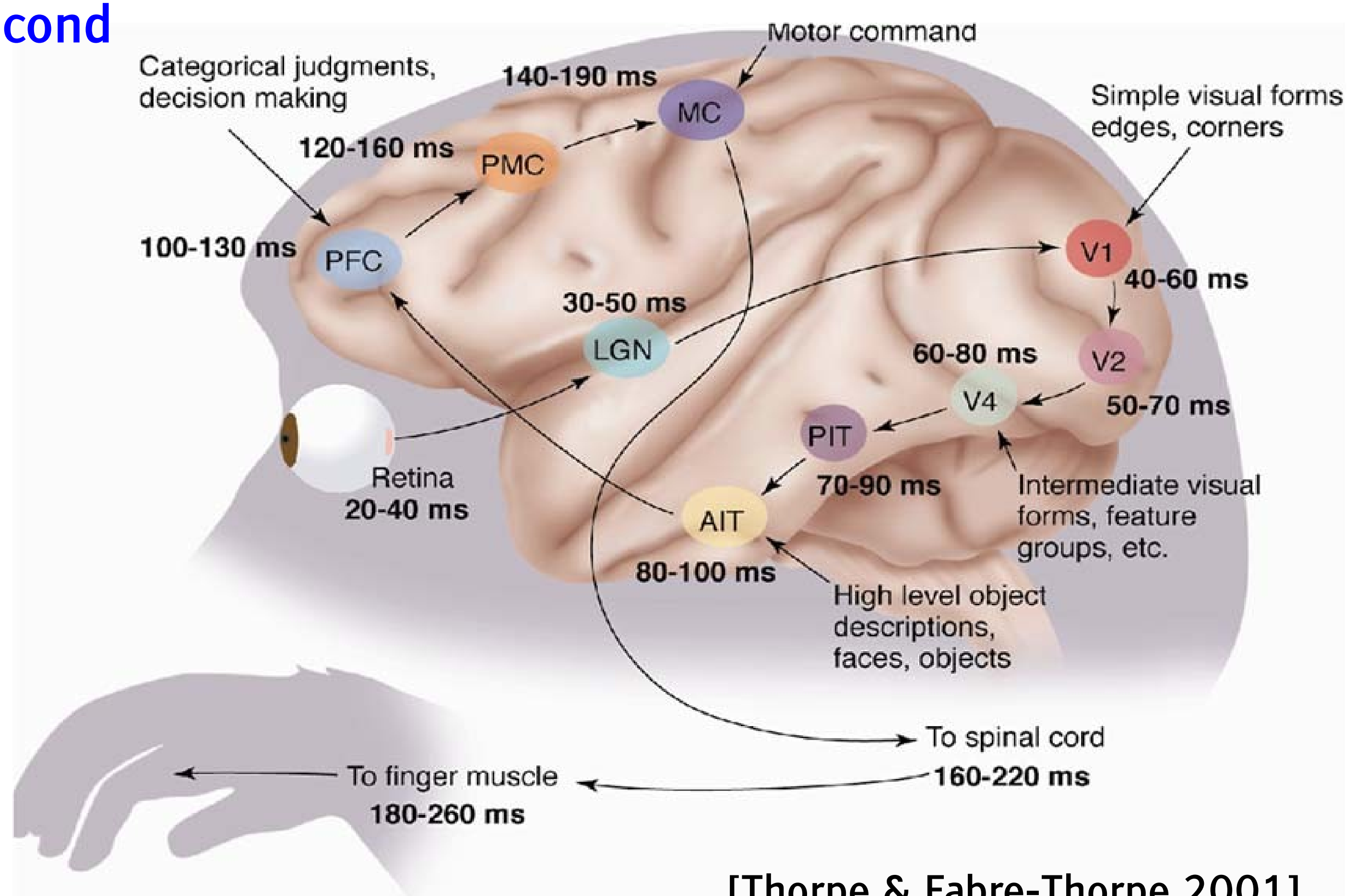
The Brain: an Amazingly Efficient "Computer"

- 10^{11} neurons, approximately
- 10^{4} synapses per neuron
- 10 "spikes" go through each synapse per second on average
- 10^{16} "operations" per second

- 25 Watts
  - Very efficient

- 1.4 kg, 1.7 liters

- 2500 cm2
  - Unfolded cortex

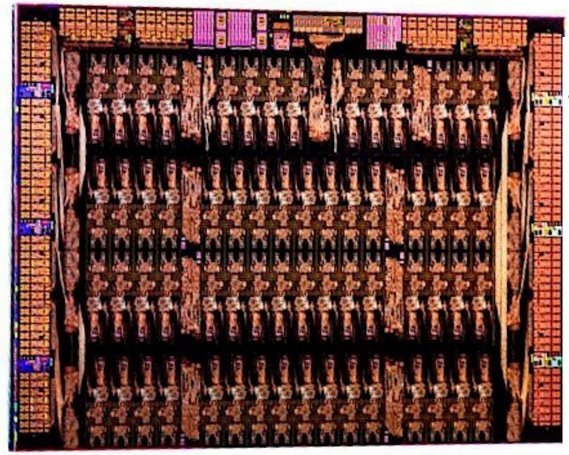


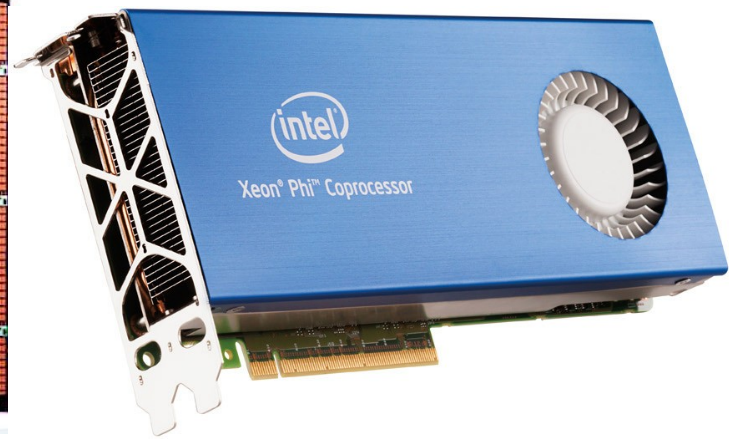
[Thorpe & Fabre-Thorpe 2001]

Fast Processors Today

Intel Xeon Phi CPU

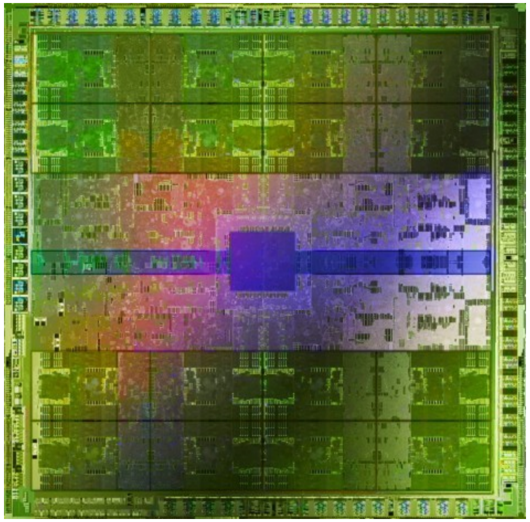
- 2×10^{12} operations/second
- 240 Watts
- 60 (large) cores
- $3000





NVIDIA Titan-Z GPU

- 8×10^{12} operations/second
- 500 Watts
- 5760 (small) cores
- $3000



Are we only a factor of 10,000 away from the power of the human brain?

- Probably more like 1 million: synapses are complicated
- A factor of 1 million is 30 years of Moore's Law
- 2045?

Can we build AI systems by copying the brain?

- **Are computers only a factor of 10,000 away from the power of the human brain?**
  - Probably more like 1 million: synapses are complicated
  - A factor of 1 million is **30 years of Moore's Law**

- **Will computers be as intelligent as human by 2045?**
  - Compute power is not the whole story
  - Moore's Law may not continue for that long
  - We need to understand the **principles** of learning and intelligence

- **Getting inspiration from biology is a good thing**
- **But blindly copying biology without understanding the underlying principles is doomed to failure**
  - Airplanes were inspired by birds
  - They use the same basic principles for flight
  - But airplanes don't flap their wings & don't have feathers

Let's be inspired by nature, but not too much

- It's nice imitate Nature,
- But we also need to understand
  - How do we know which details are important?
  - Which details are merely the result of evolution, and the constraints of biochemistry?
- For airplanes, we developed aerodynamics and compressible fluid dynamics.
  - We figured that feathers and wing flapping weren't crucial
- QUESTION: What is the equivalent of aerodynamics for understanding intelligence?

L'Avion III de Clément Ader, 1897

(Musée du CNAM, Paris)

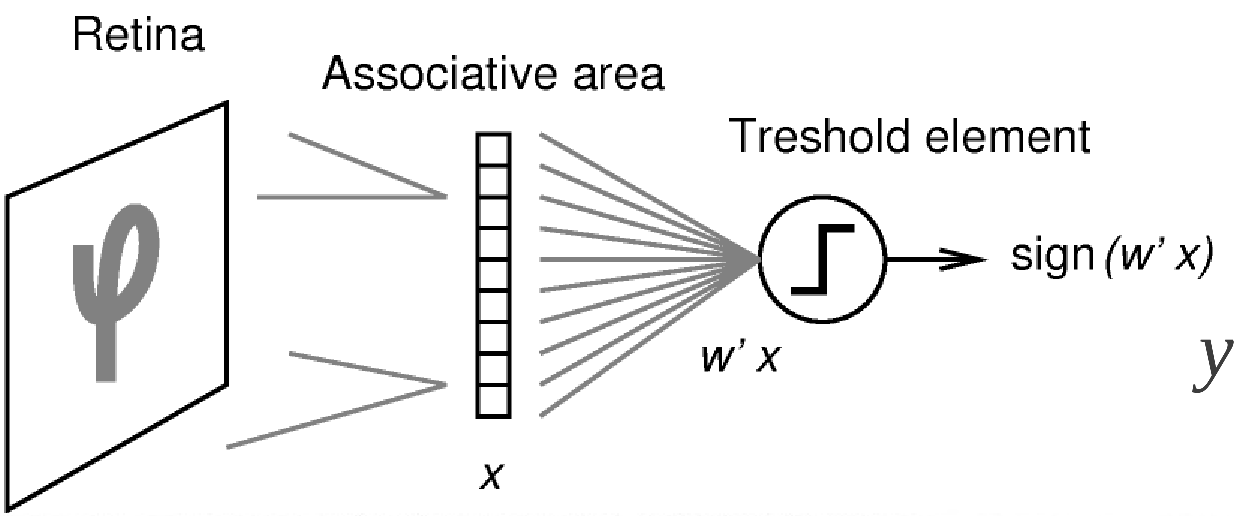
His “Eole” took off from the ground in 1890,

13 years before the Wright Brothers, but you

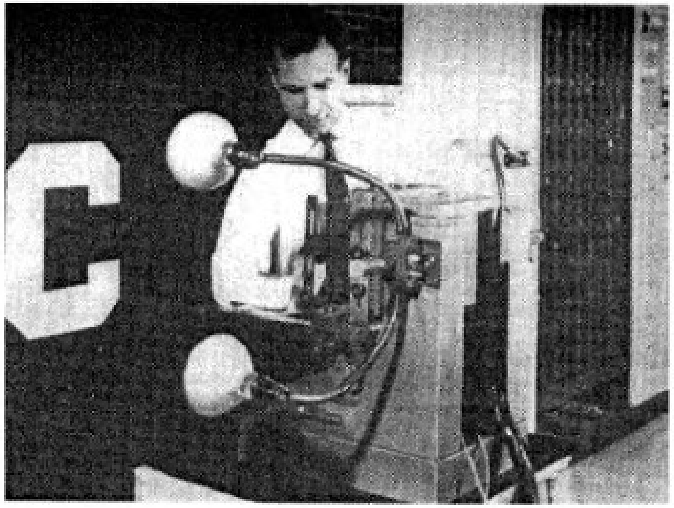
probably never heard of it (unless you are french).

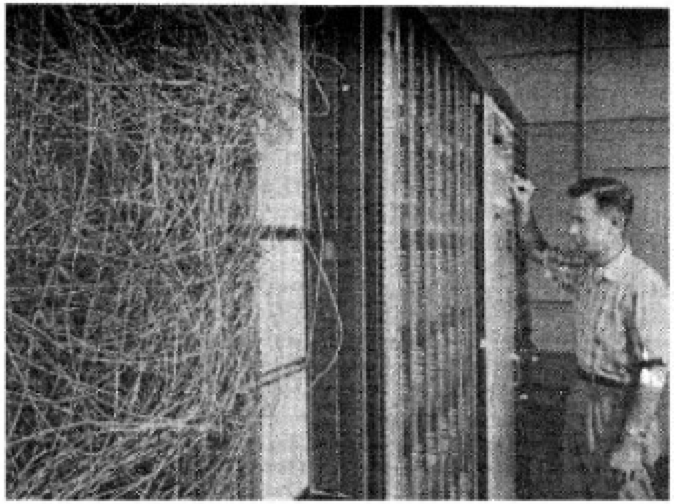
1957: The Perceptron (the first learning machine)

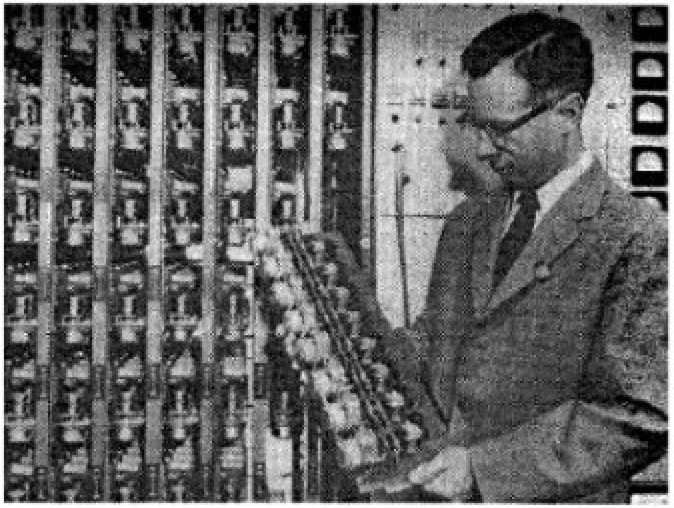
- A simple simulated neuron with adaptive "synaptic weights"
  - Computes a weighted sum of inputs
  - Output is +1 if the weighted sum is above a thresold, -1 otherwise.



$$y = sign\left(\sum_{i=1}^{N} W_i X_i + b\right)$$





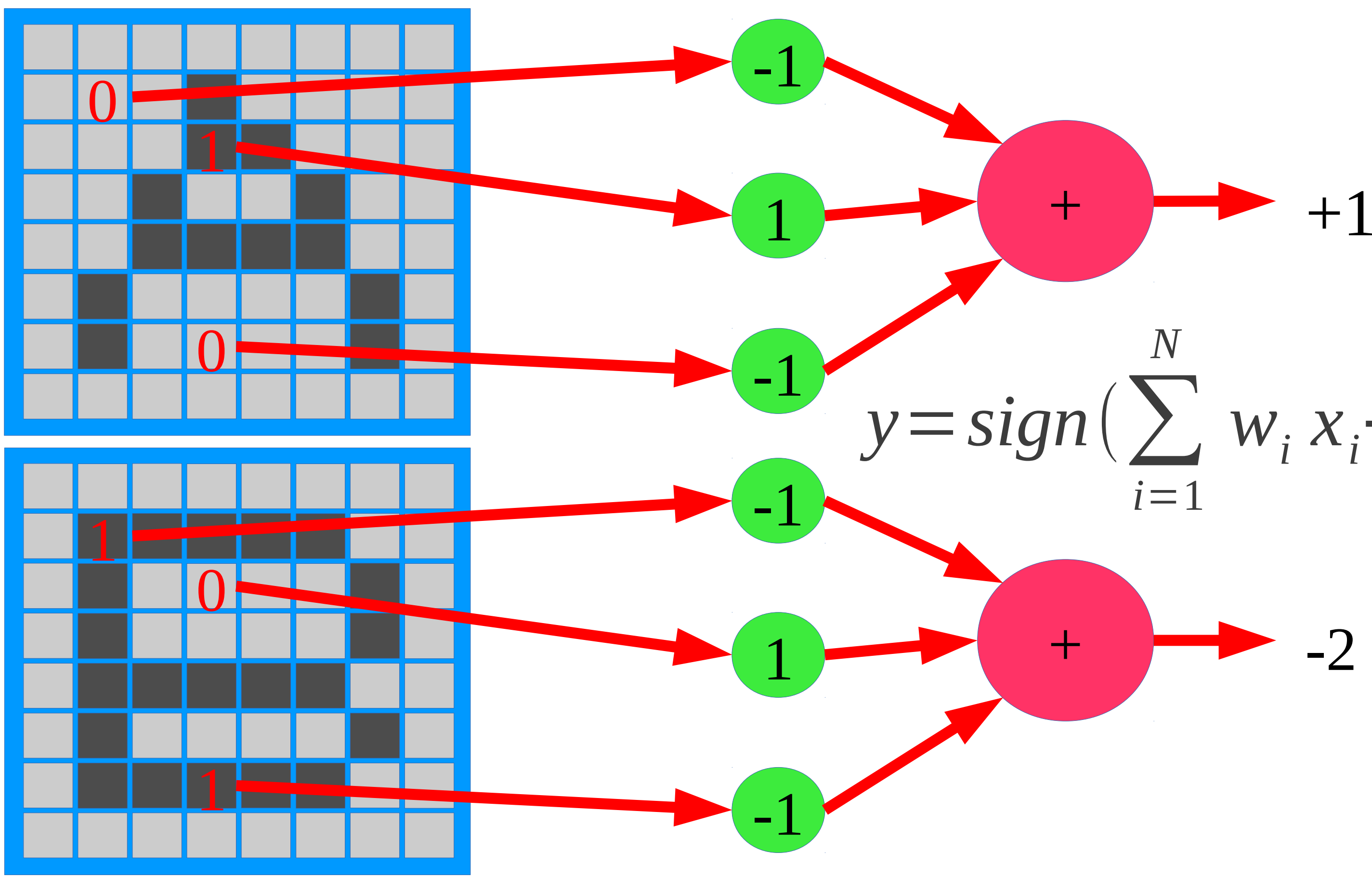


The Perceptron: a Trainable Classifier [Rosenblatt 1957]

Y LeCun

- Example: classifying letters "A" from "B"
- Learning: find the weight values that produce +1 for A and -1 for B
- Training set: $(X^1,Y^1),(X^2,Y^2),\ldots\ldots,(X^p,Y^p)$
- Example: (**A**,+1),(**B**,-1),(**A**,+1),(**B**,-1),(**A**,+1),(**B**,-1),......

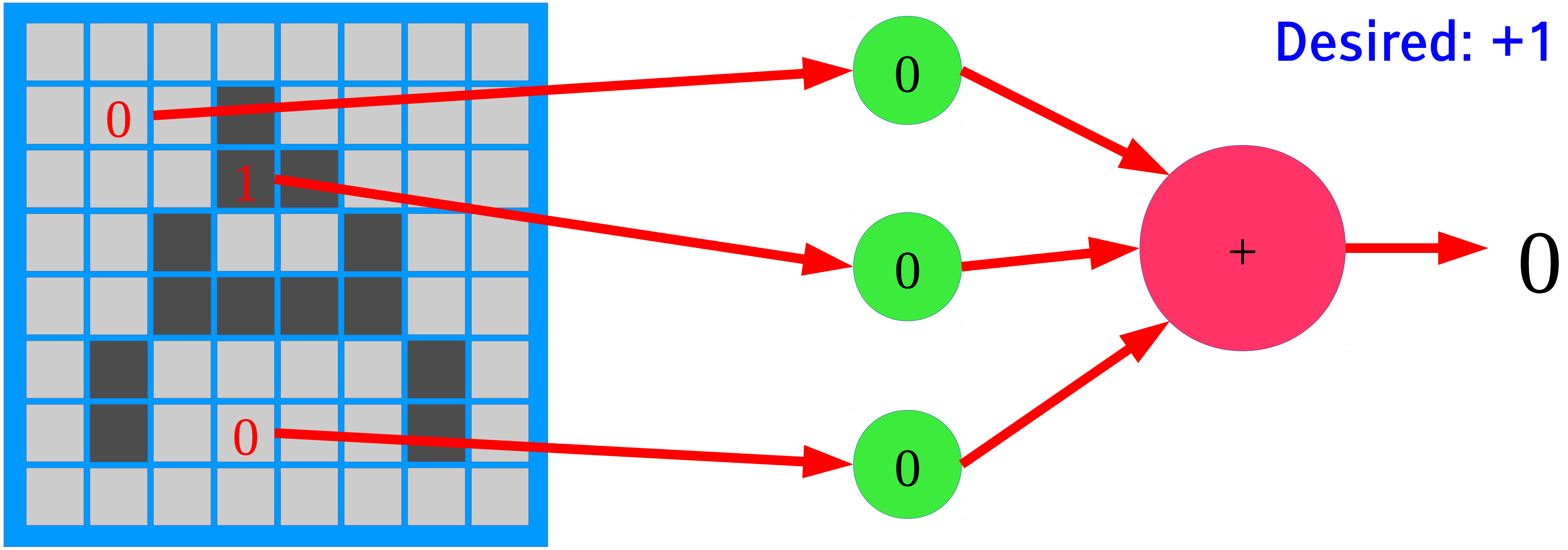
Supervised Learning



$$y=sign\left(\sum_{i=1}^{N} w_i x_i + w_0\right)=sign(W'X)$$

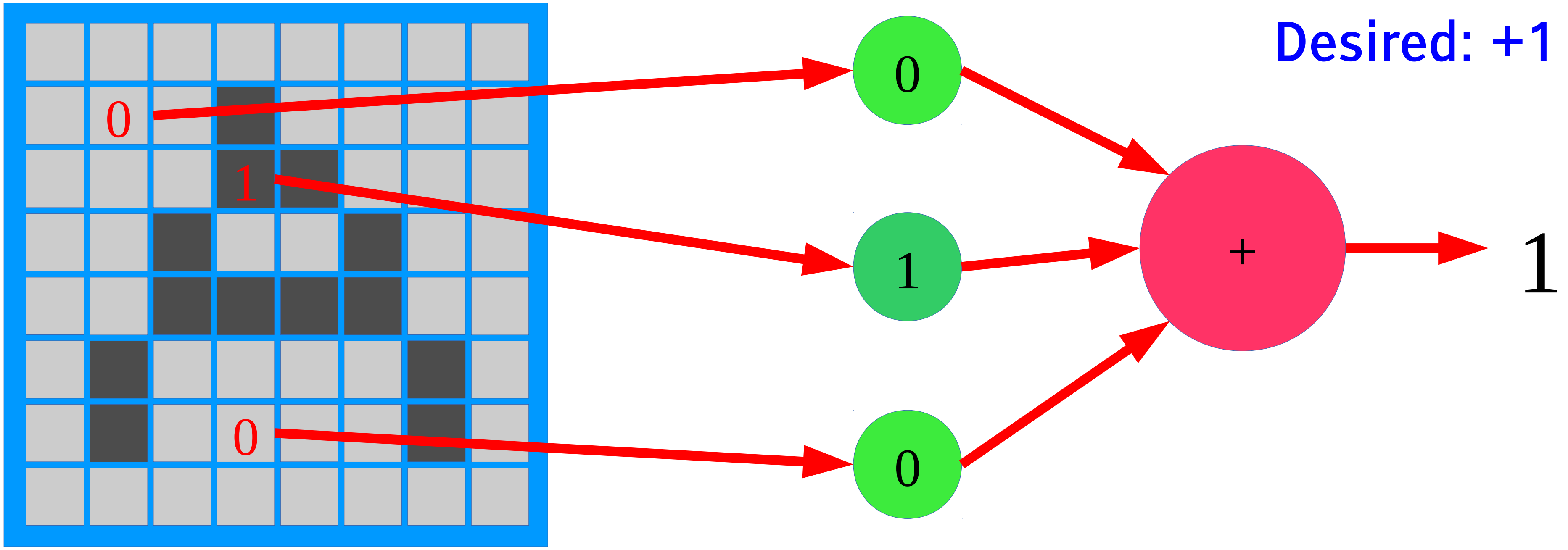
Learning the Weights

- Learning: adjusting the weights so as to obtain the desired result
  - Initially, the weights are 0.



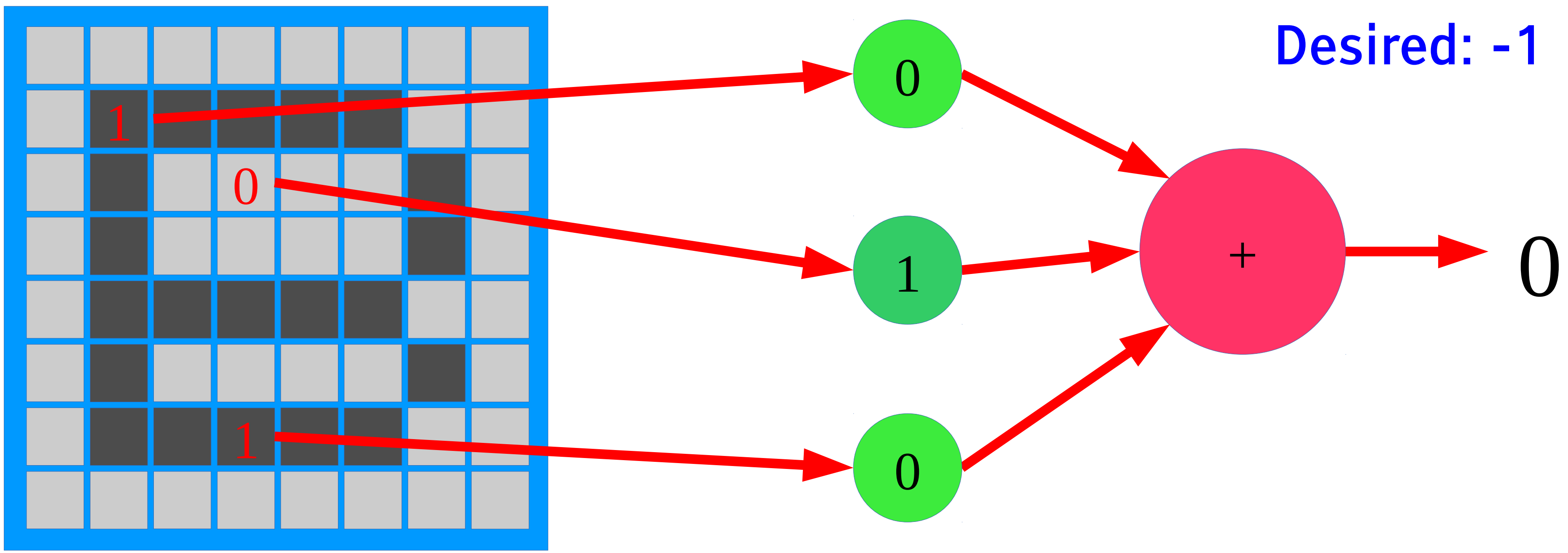
Learning the Weights

- **Adjusting the weights when the the output is incorrect**
  - If the desired output is +1, add pixel values to the weights (Hebbian learning)



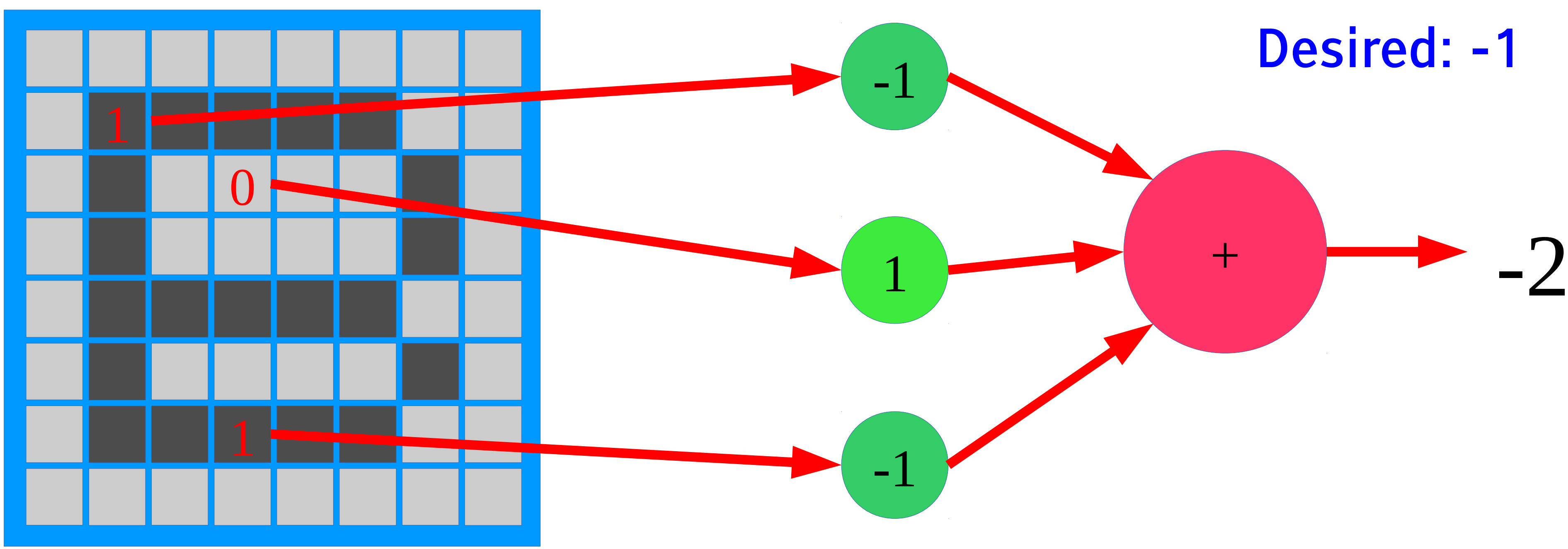
Apprentissage

- Adjusting the weights when the the output is incorrect
  - If the desired output is -1, subtract pixel values from the weights.



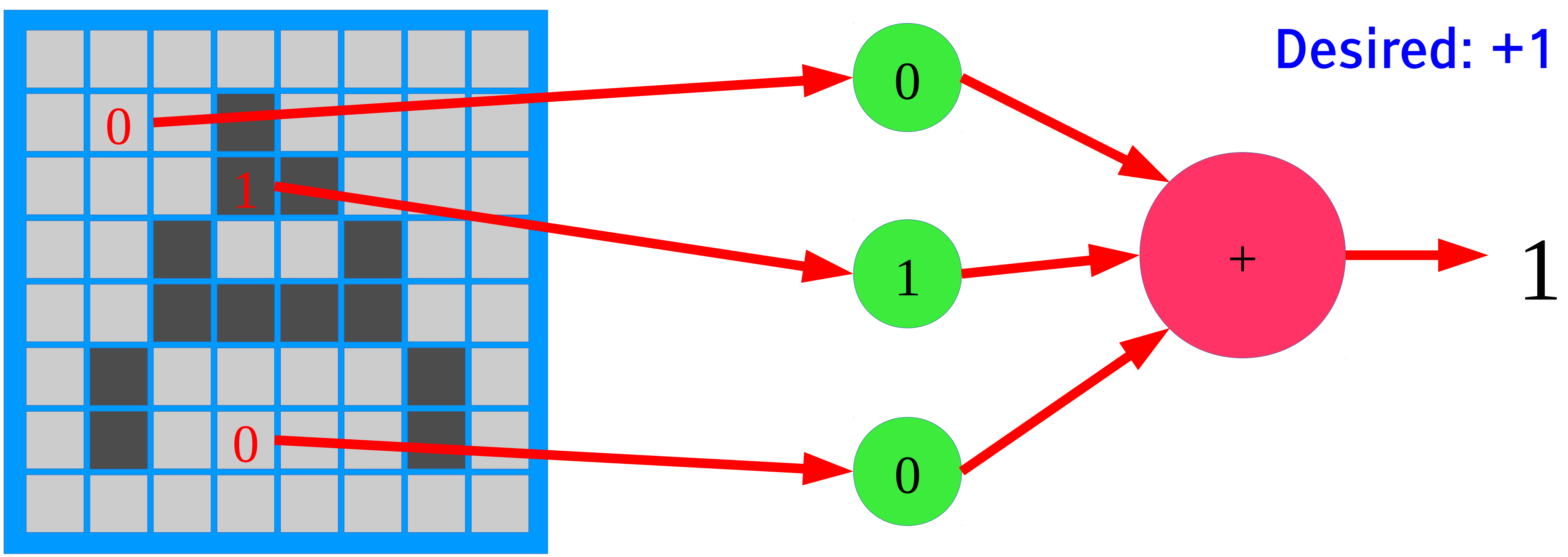
Apprentissage

- **Adjusting the weights when the the output is incorrect**
  - If the desired output is -1, subtract pixel values from the weights.



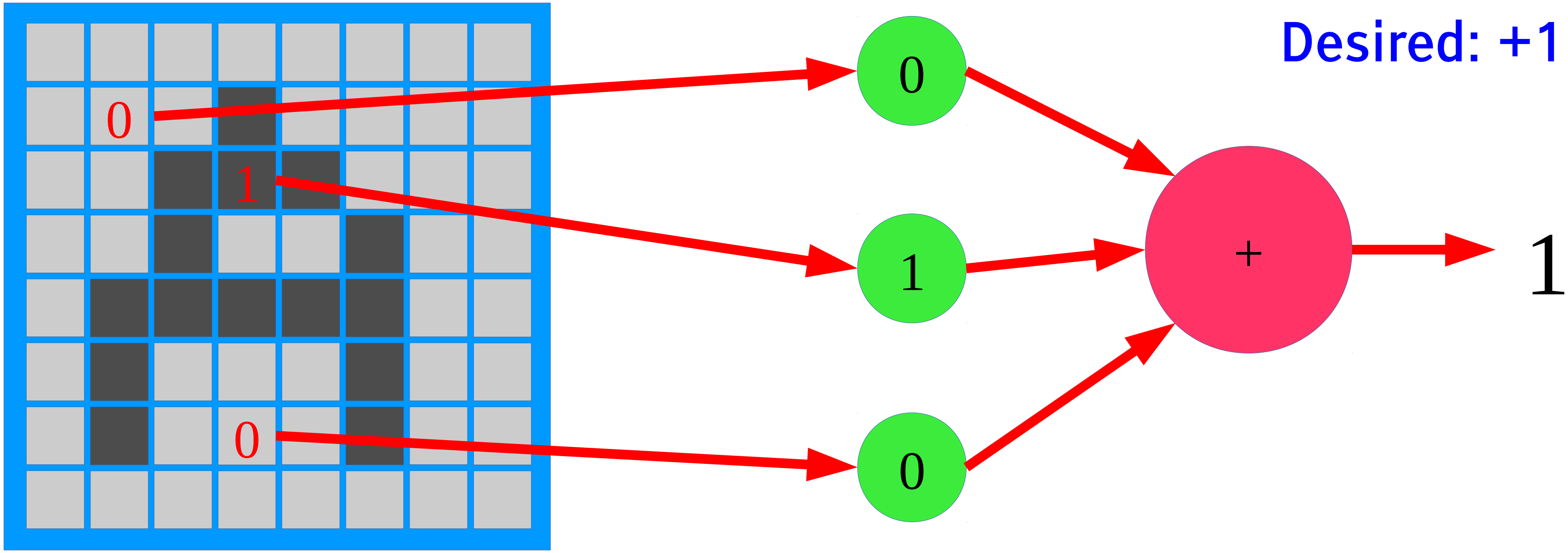
Problem

- Write if the writing style varies?



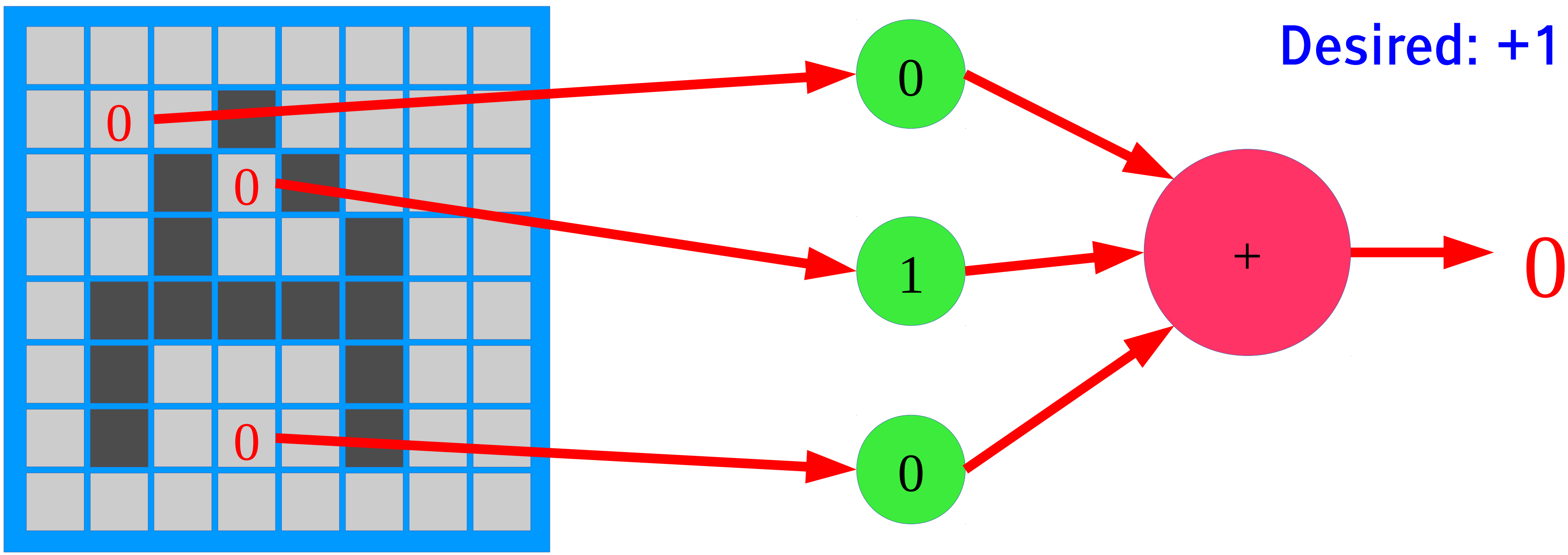
Problem

- What if the writing style varies?



Problè/em

- **What if the writing style varies?**
  - The output may become incorrect

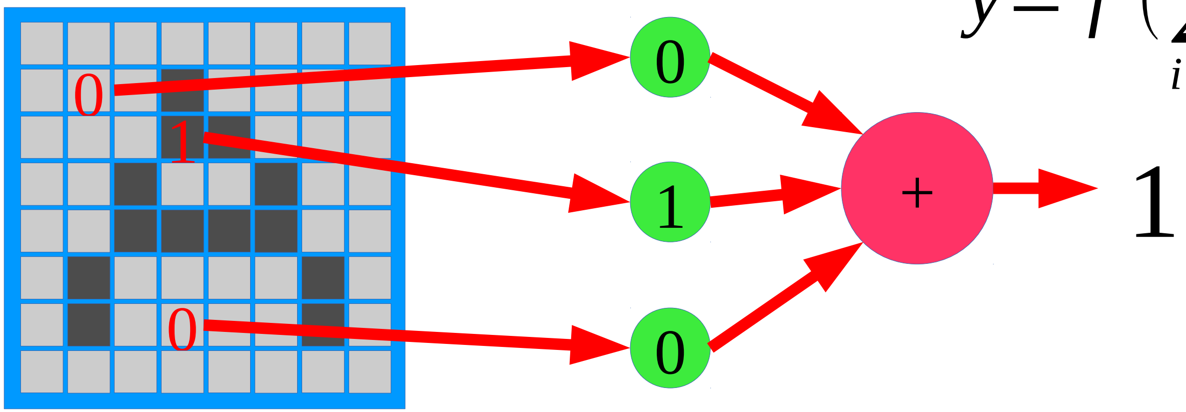


The Perceptron Learning Algorithm

- Training set: $(X^1,Y^1),(X^2,Y^2),\ldots\ldots,(X^p,Y^p)$
- Take one sample (X^k,Y^k), if the desired output is +1 but the actual output is -1
  - Increase the weights whose input is positive
  - Decrease the weights whose input is negative
- If the desired is -1 and actual is +1, do the converse.
- If desired and actual are equal, do nothing

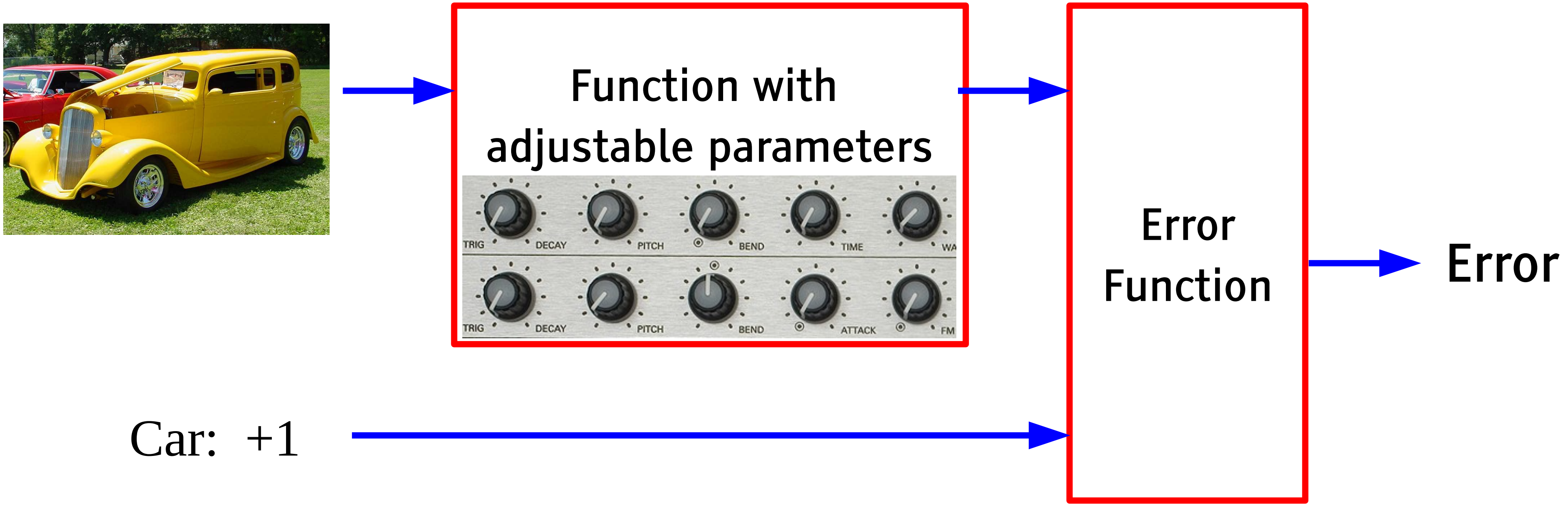
$$\boxed{w_i(t+1) = w_i(t) + (y_i^p - f(W'X^p))x_i^p}$$

$$y = f\left(\sum_{i=1}^{N} w_i x_i + w_0\right) = f(W'X)$$



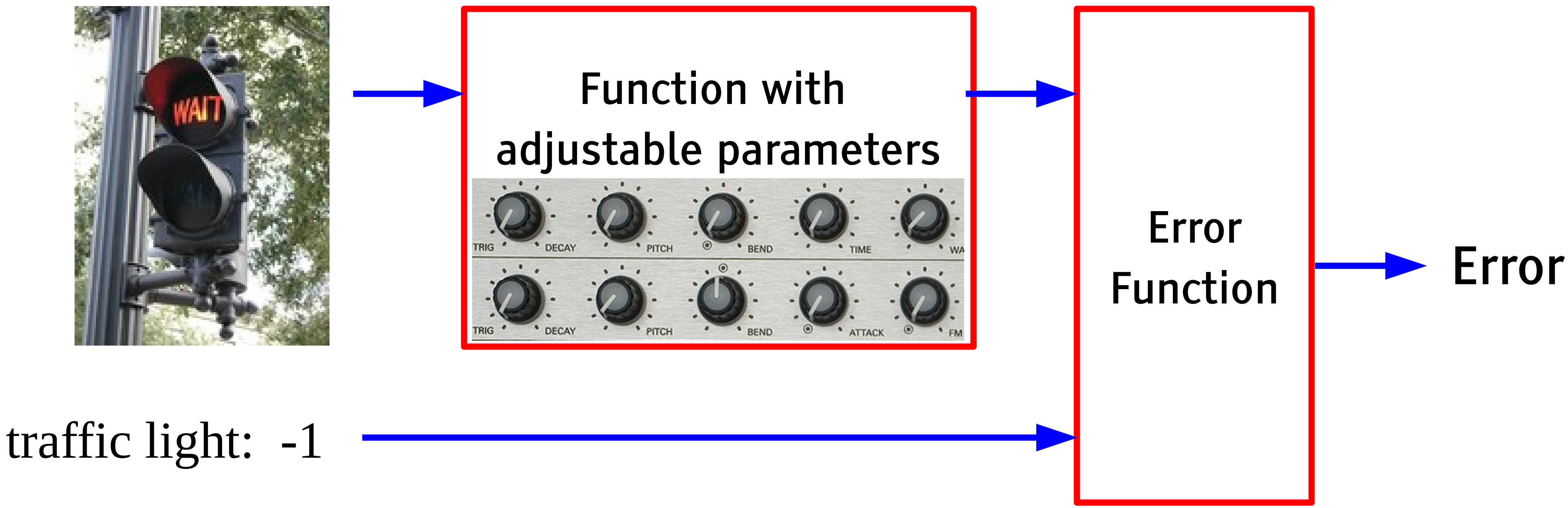
Supervised Learning

Machine Learning in General (supervised learning)

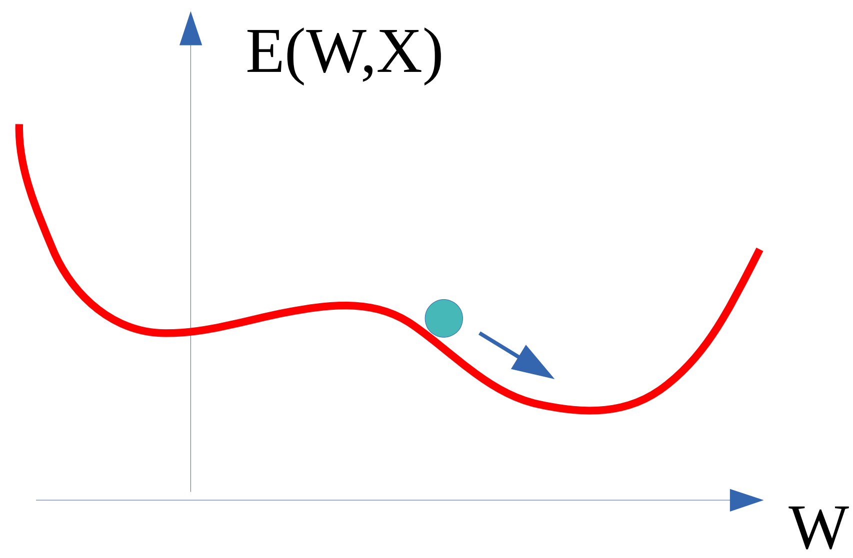


- Design a machine with adjustable knobs (like the weights in the Perceptron)
- Pick a training sample, run it through, and measure the error.
- Figure out in which direction to adjust the knobs so as to lower the error
- Repeat with all the training samples until the knobs stabilize

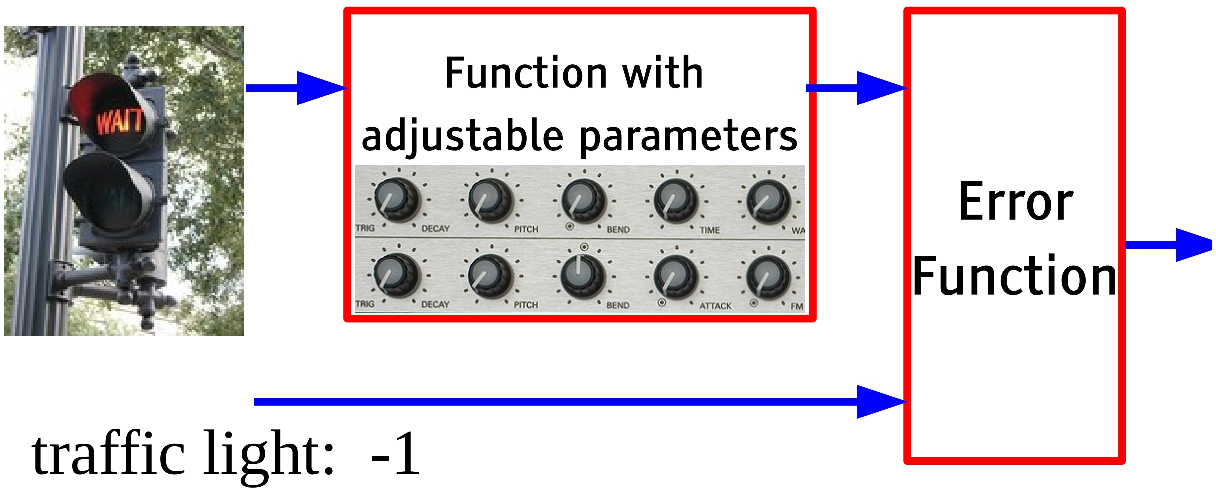
Machine Learning in General (supervised learning)

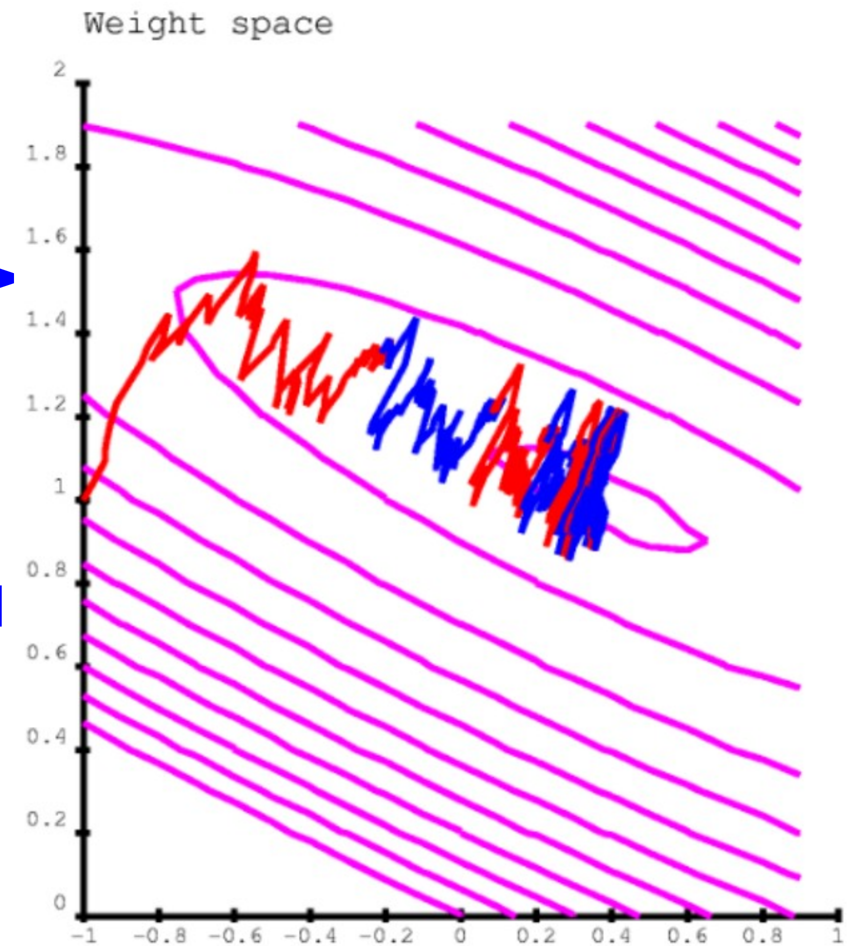


- Design a machine with adjustable knobs
- Pick a training sample, run it through
- Adjust the knobs so as to lower the error
- Repeat until the knobs stabilize



Machine Learning = Function Optimization





- It's like walking in the mountains in a fog and following the direction of steepest descent to reach the village in the valley
- But each sample gives us a noisy estimate of the direction. So our path is a bit random.

$$W_i \leftarrow W_i - \eta \frac{\partial E(W,X)}{\partial W_i}$$

- Stochastic Gradient Descent (SGD)

Generalization Ability: recognizing instances not seen during training

Function with adjustable parameters

After training:

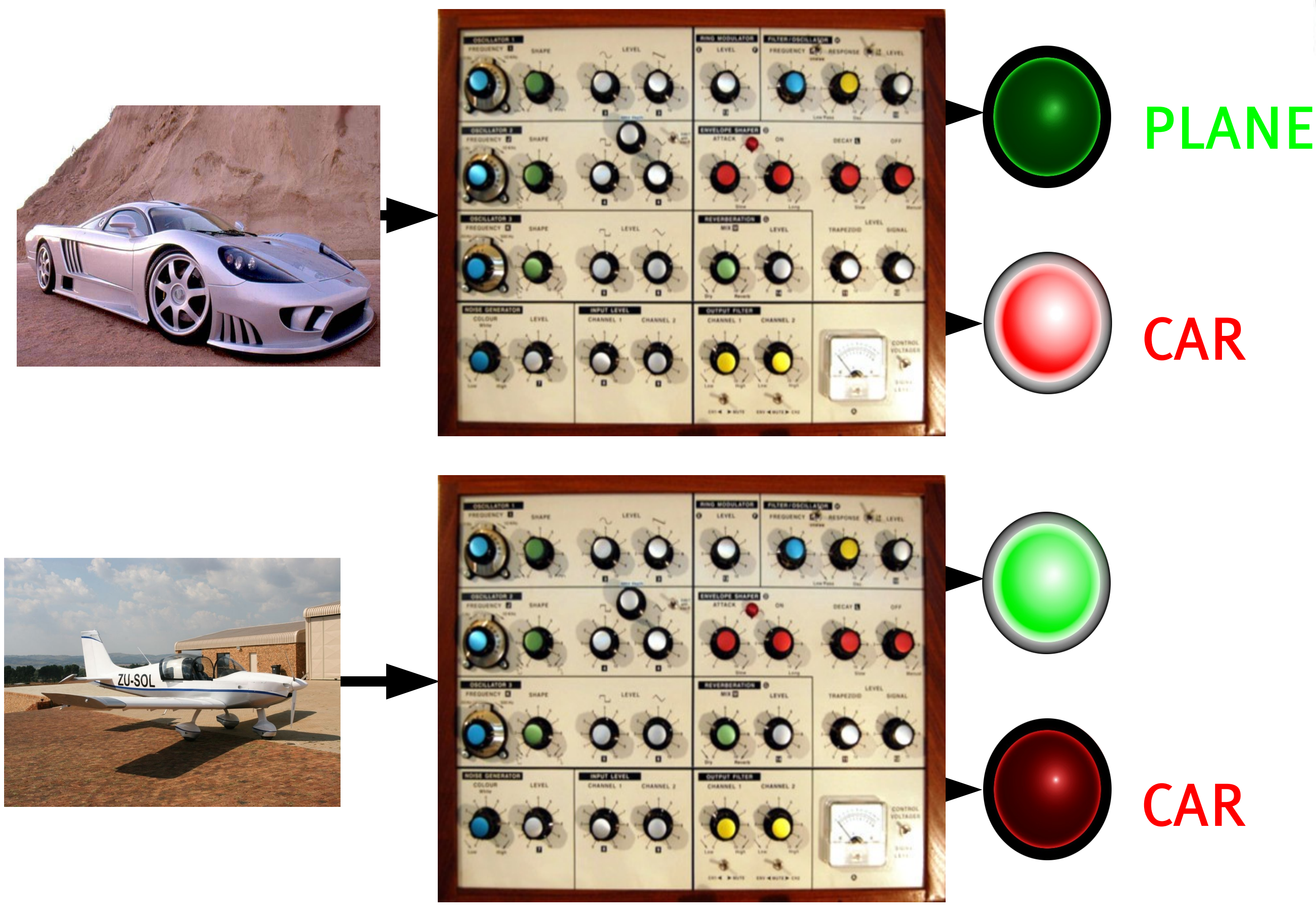
- Test the machine on samples it has never seen before.

Can you discover the rule?

- 0, 2, 4, 6, 8, 10, 12.......
- 3, 5, 2, 8, 1, 6, 7, 9, 12, 2, ...
- 5, 9, 2, 6, 5, 3, 5, 8, 9, ...

Supervised Learning

- We can train a machine on lots of examples of tables, chairs, dog, cars, and people
- But will it recognize table, chairs, dogs, cars, and people it has never seen before?



Large-Scale Machine Learning: the reality

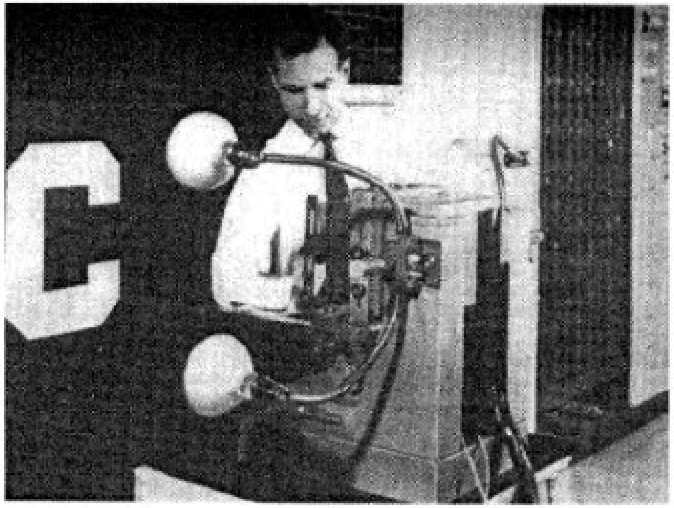
- Hundreds of millions of “knobs” (or weights)
- Thousands of categories
- Millions of training samples
- Recognizing each sample may take billions of operations
  - But these operations are simple multiplications and additions

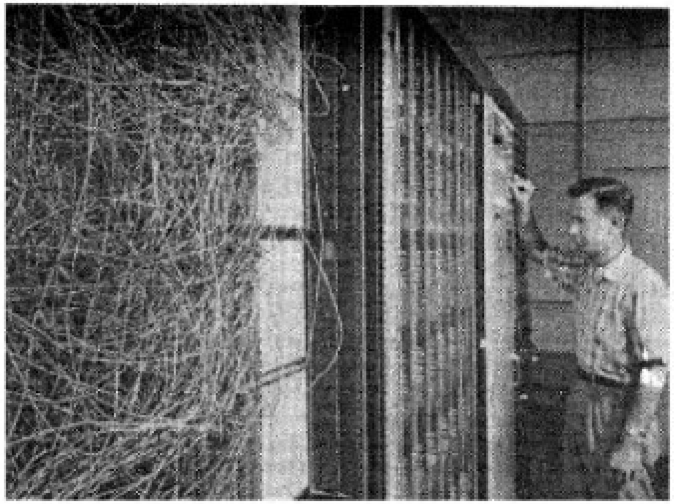
The Traditional Model of Pattern Recognition

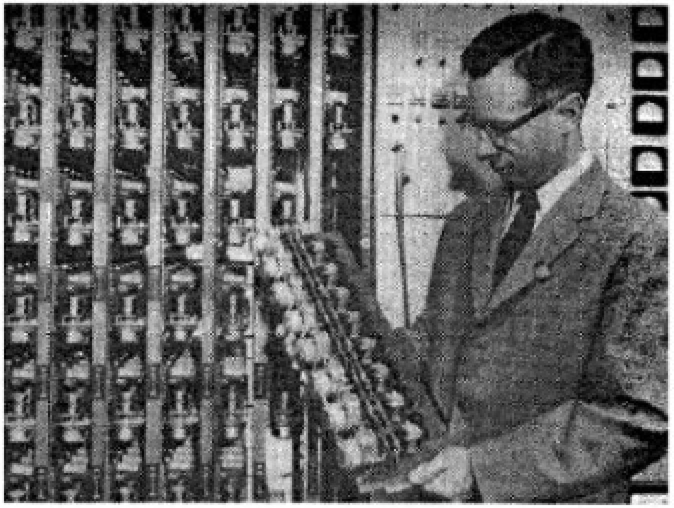
- **The traditional model of pattern recognition (since the late 50's)**
  - Fixed/engineered features (or fixed kernel) + trainable classifier

hand-crafted Feature Extractor → “Simple” Trainable Classifier →

- **Perceptron (Cornell University, 1957)**







Deep Learning = The Entire Machine is Trainable

Y LeCun

- **Traditional Pattern Recognition:** Fixed/Handcrafted Feature Extractor

Feature Extractor → Trainable Classifier →

- **Mainstream Modern Pattern Recognition:** Unsupervised mid-level features

Feature Extractor → Mid-Level Features → Trainable Classifier →

- **Deep Learning:** Representations are hierarchical and trained

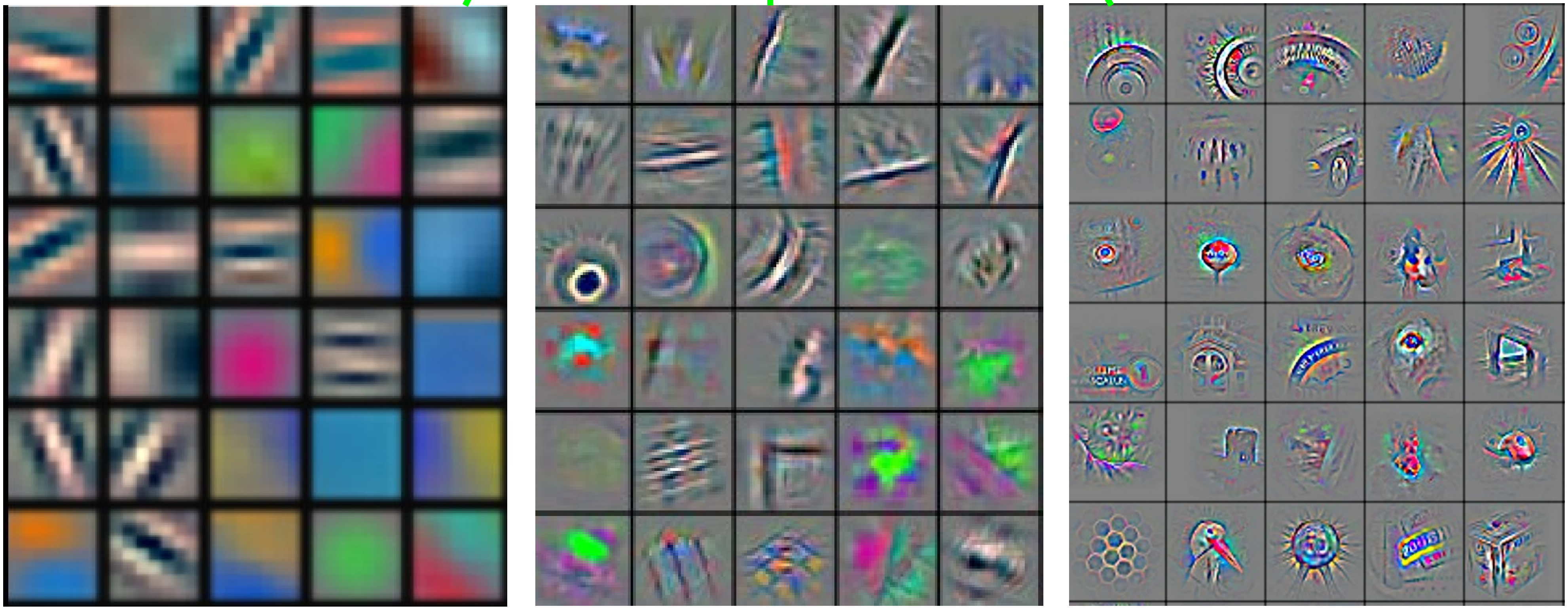
Low-Level Features → Mid-Level Features → High-Level Features → Trainable Classifier →

Deep Learning = Learning Hierarchical Representations

Y LeCun

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

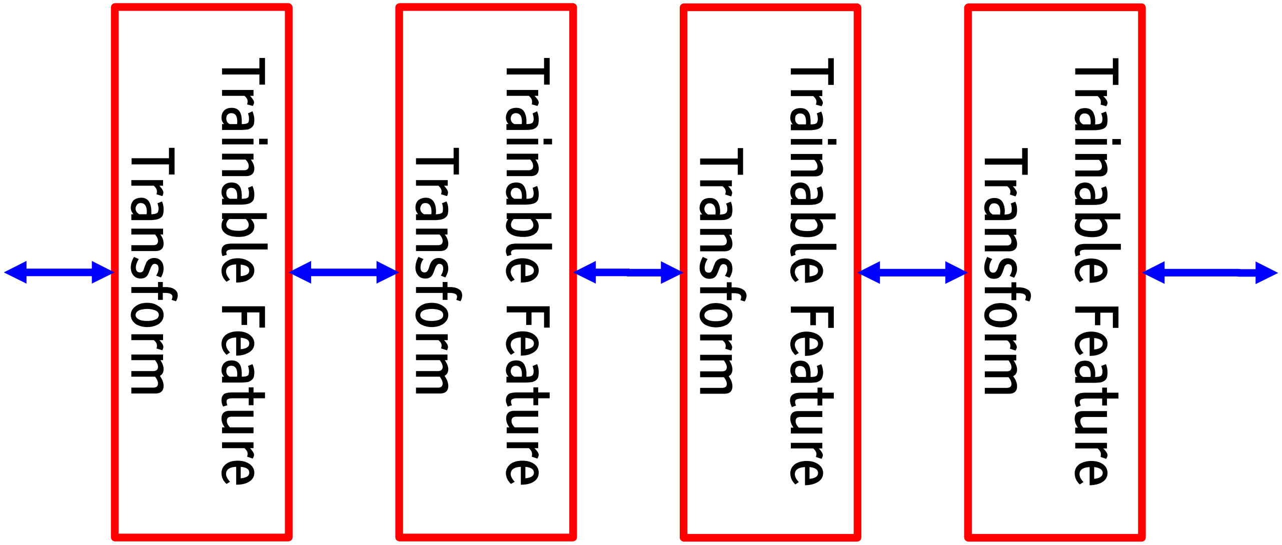
Low-Level Feature → Mid-Level Feature → High-Level Feature → Trainable Classifier →



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

Trainable Feature Hierarchy

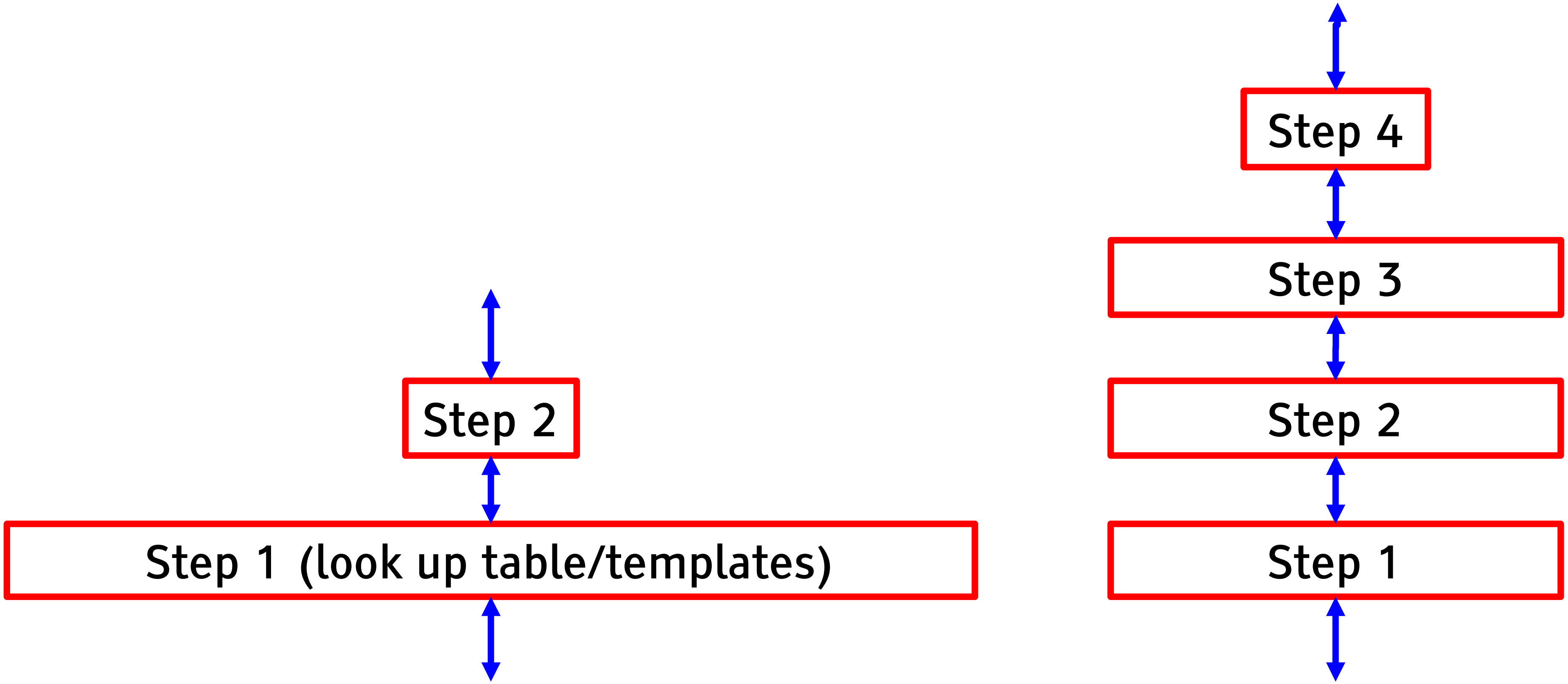
- **Hierarchy of representations with increasing level of abstraction**
- **Each stage is a kind of trainable feature transform**
- **Image recognition**
  - Pixel → edge → texton → motif → part → object
- **Text**
  - Character → word → word group → clause → sentence → story
- **Speech**
  - Sample → spectral band → sound → … → phone → phoneme → word



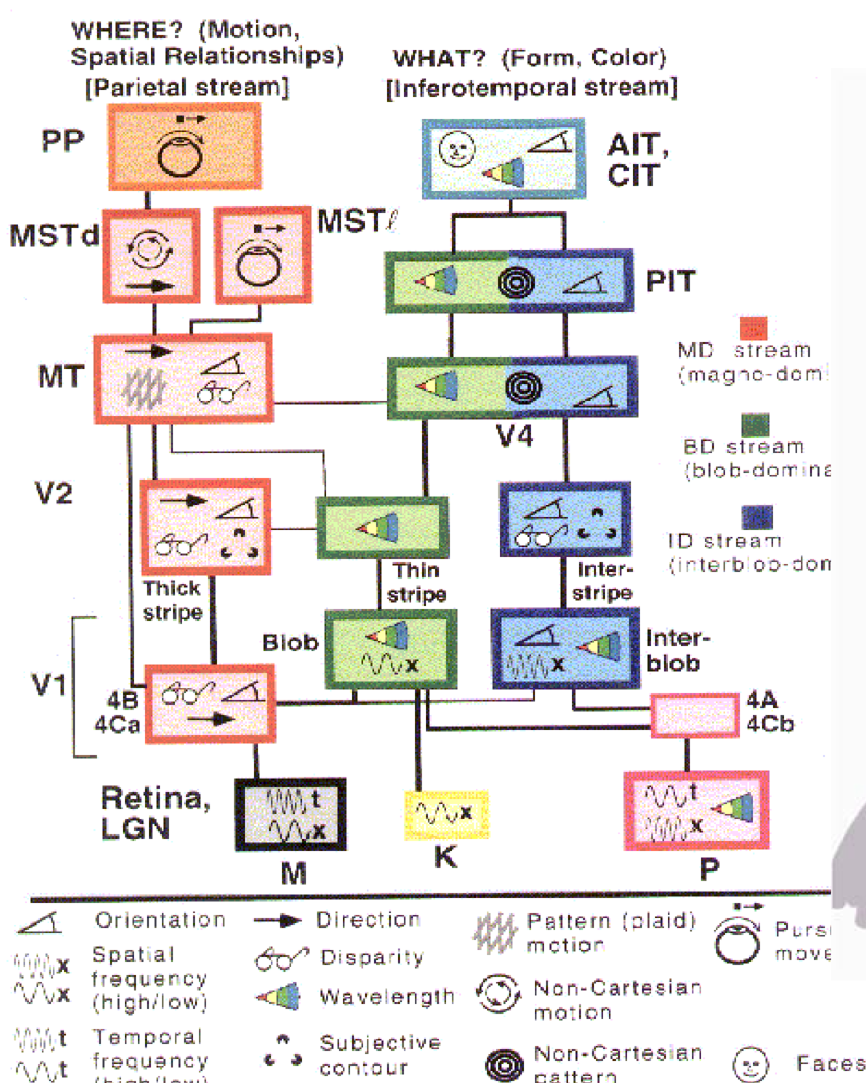
Shallow vs Deep == lookup table vs multi-step algorithm

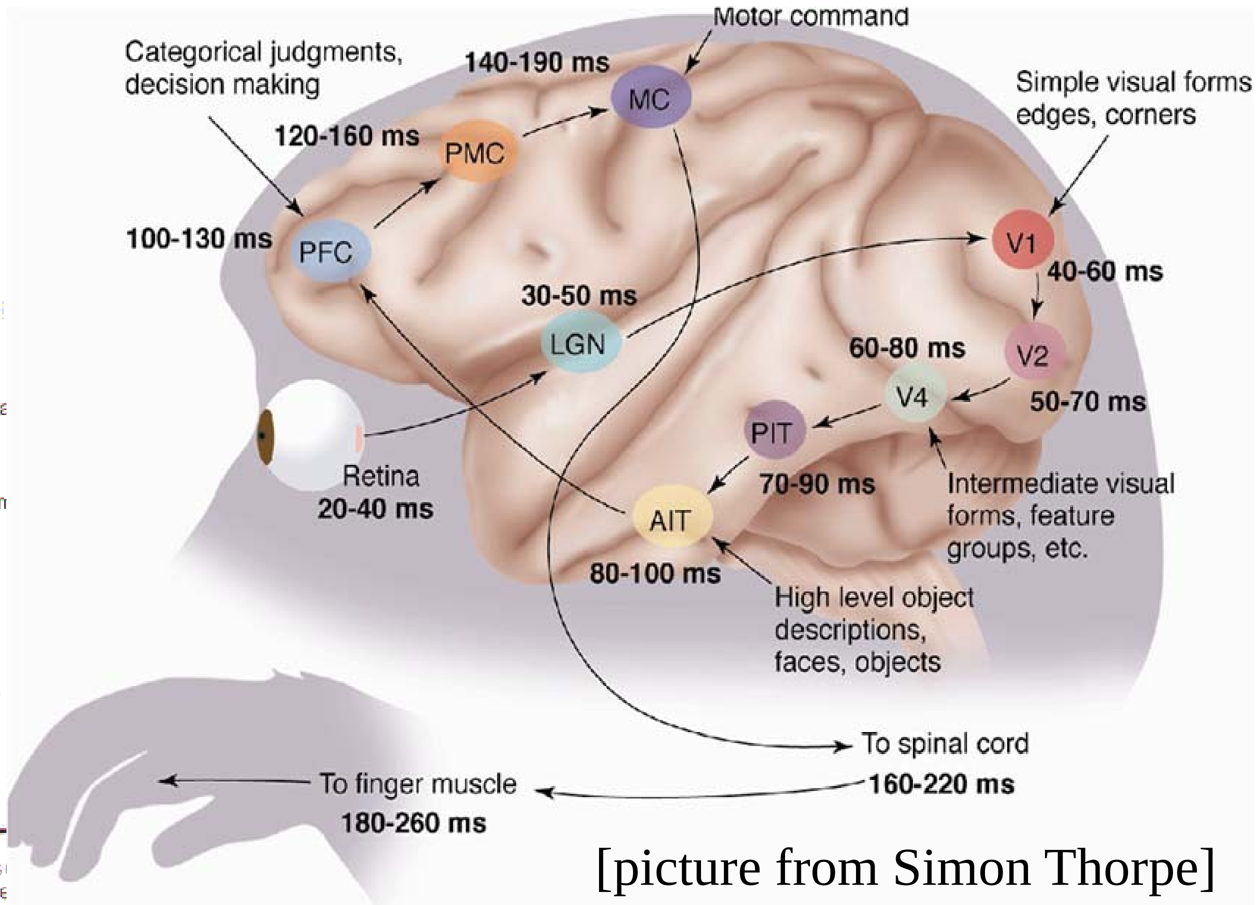
Y LeCun

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



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





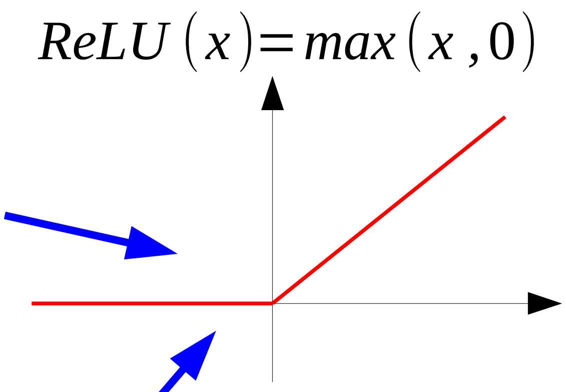
[picture from Simon Thorpe]

[Gallant & Van Essen]

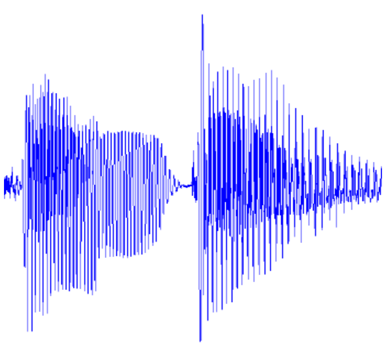
Multi-Layer Neural Networks

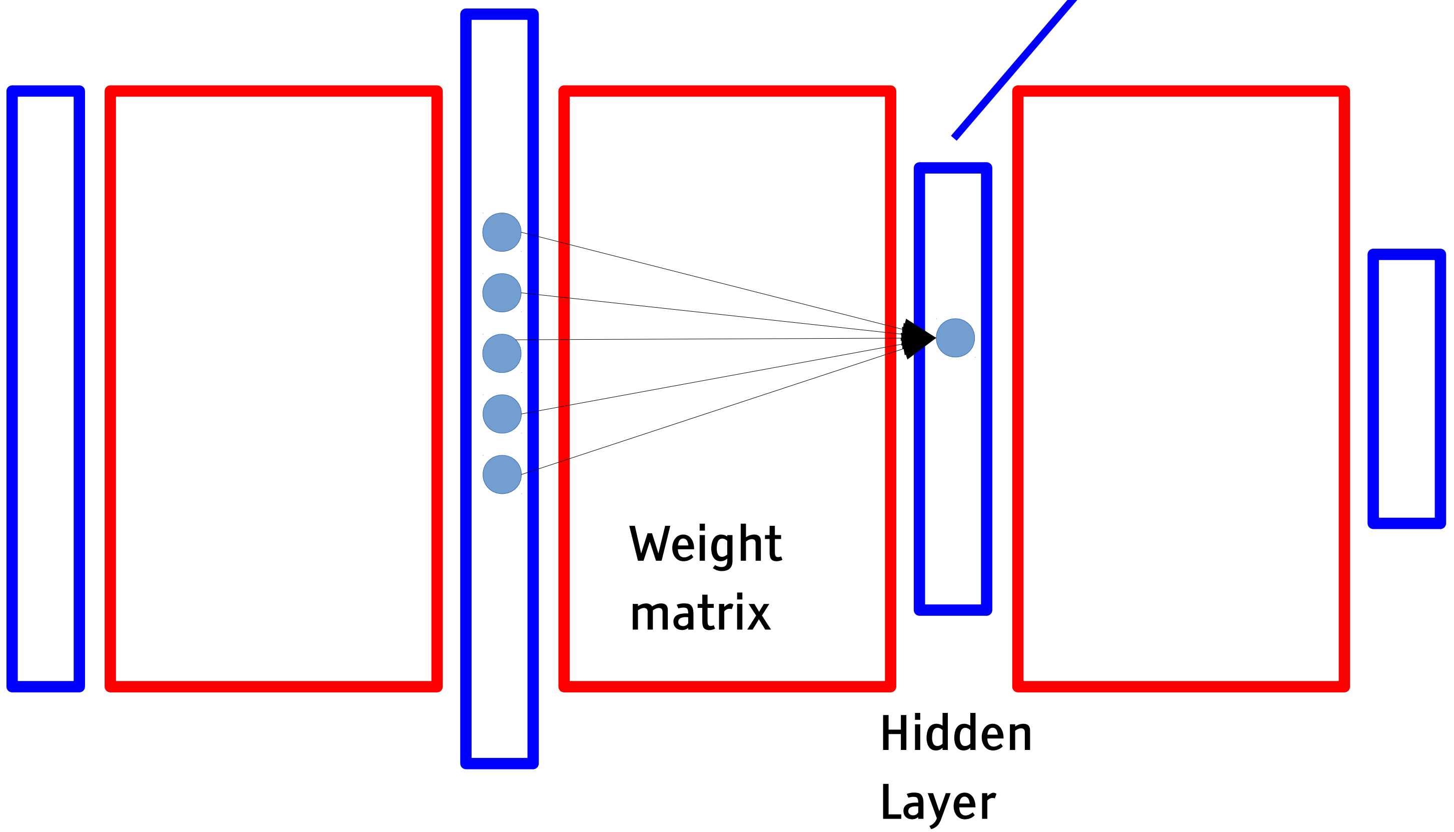
Multi-Layer Neural Nets

- Multiple Layers of simple units
- Each units computes a weighted sum of its inputs
- Weighted sum is passed through a non-linear function
- The learning algorithm changes the weights



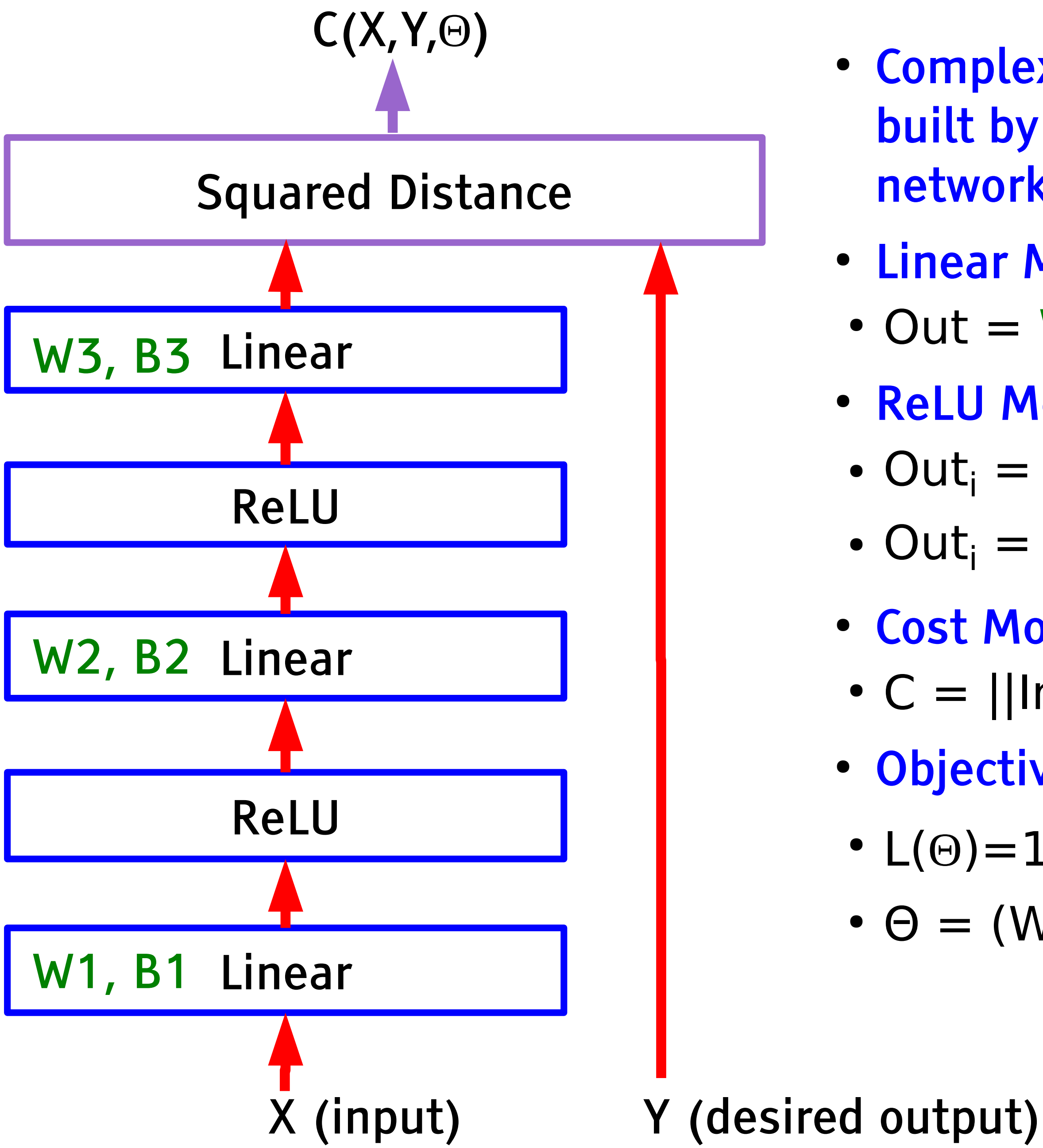
Ceci est une voiture





Typical Multilayer Neural Net Architecture

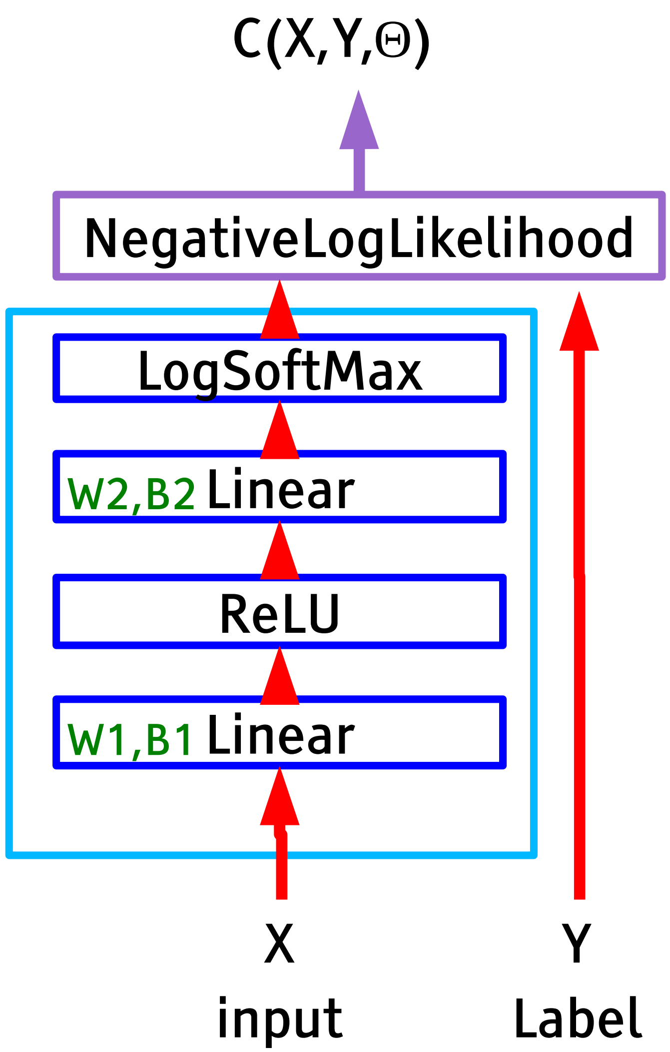
Y LeCun



- Complex learning machines can be built by assembling modules into networks
- Linear Module
  - Out = W.In+B
- ReLU Module (Rectified Linear Unit)
  - $Out_i = 0$ if $In_i < 0$
  - $Out_i = In_i$ otherwise
- Cost Module: Squared Distance
  - $C = ||In1 - In2||^2$
- Objective Function
  - $L(\Theta) = 1/p \sum_k C(X^k, Y^k, \Theta)$
  - $\Theta = (W1, B1, W2, B2, W3, B3)$

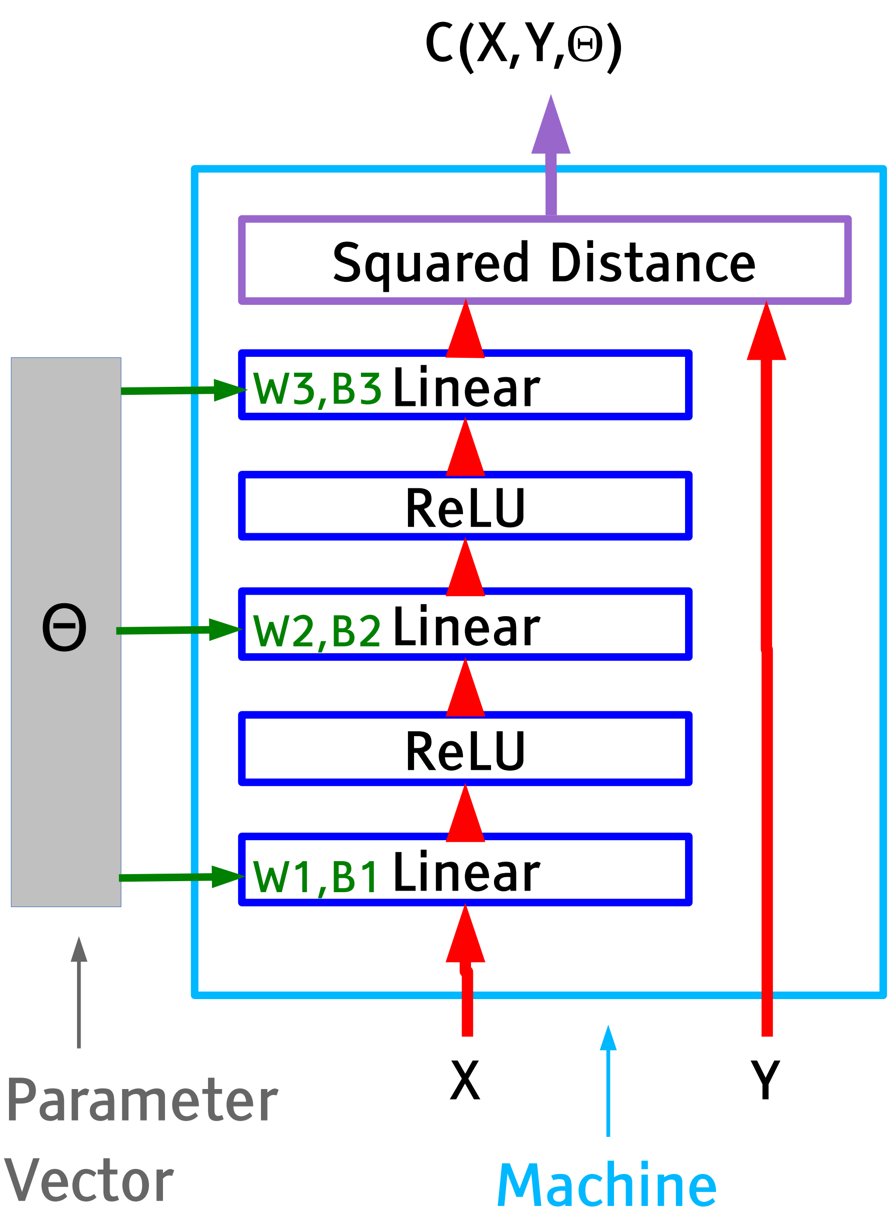
Building a Network by Assembling Modules

- All major deep learning frameworks use modules (inspired by SN/Lush, 1991)
  - Torch7, Theano, TensorFlow....



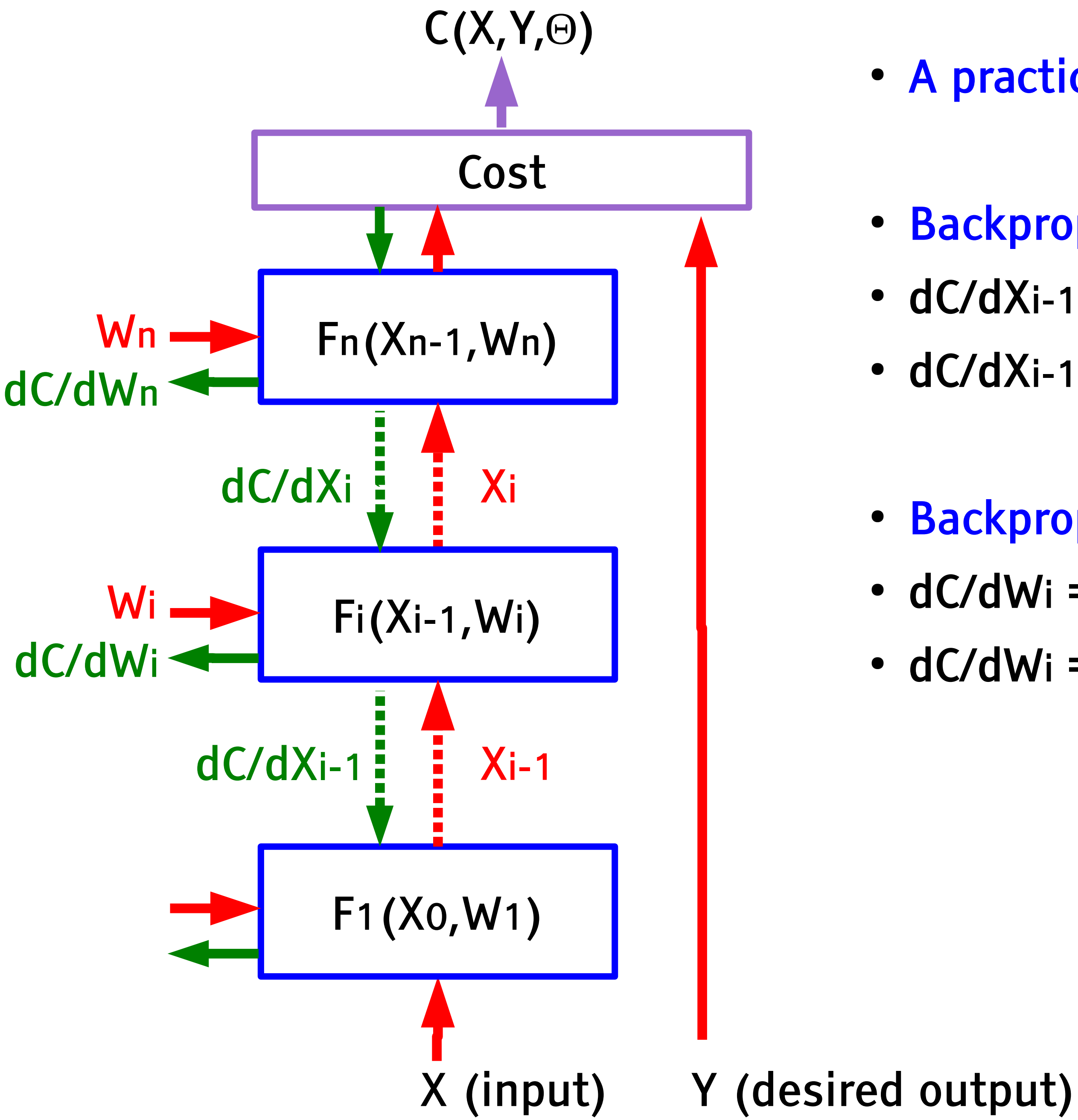

```
-- sizes
ninput = 28*28   -- e.g. for MNIST
nhidden1 = 1000
noutput = 10

-- network module
net = nn.Sequential()
net:add(nn.Linear(ninput, nhidden))
net:add(nn.Threshold())
net:add(nn.Linear(nhidden, noutput))
net:add(nn.LogSoftMax()))

-- cost module
cost = nn.ClassNLLCriterion()

-- get a training sample
input = trainingset.data[k]
target = trainingset.labels[k]

-- run through the model
output = net:forward(input)
c = cost:forward(output, target)
```

Training: Stochastic Gradient Descent (SGD)



- Objective Fn: average over samples
  - $L(\Theta) = 1/p \sum_k C(X^k, Y^k, \Theta)$
  - $\Theta = (W1, B1, W2, B2, W3, B3)$
- Stochastic Gradient Descent
  - $\Theta \leftarrow \Theta - \eta\, \partial C(X^k, Y^k, \Theta)/\partial\Theta$
  - $\Theta \leftarrow \Theta - \eta\, \Delta C(X^k, Y^k, \Theta)$
- Noisy estimate of the gradient
- In practice, we use a “minibatch”
  - $\Theta \leftarrow \Theta - \eta \sum_k \Delta C(X^k, Y^k, \Theta)$
  - Typical minibatch size:
  - 32 to 1024 samples
  - The smaller the better
- Why use minibatch then?
  - Because it goes faster on GPUs.

Computing Gradients by Back-Propagation

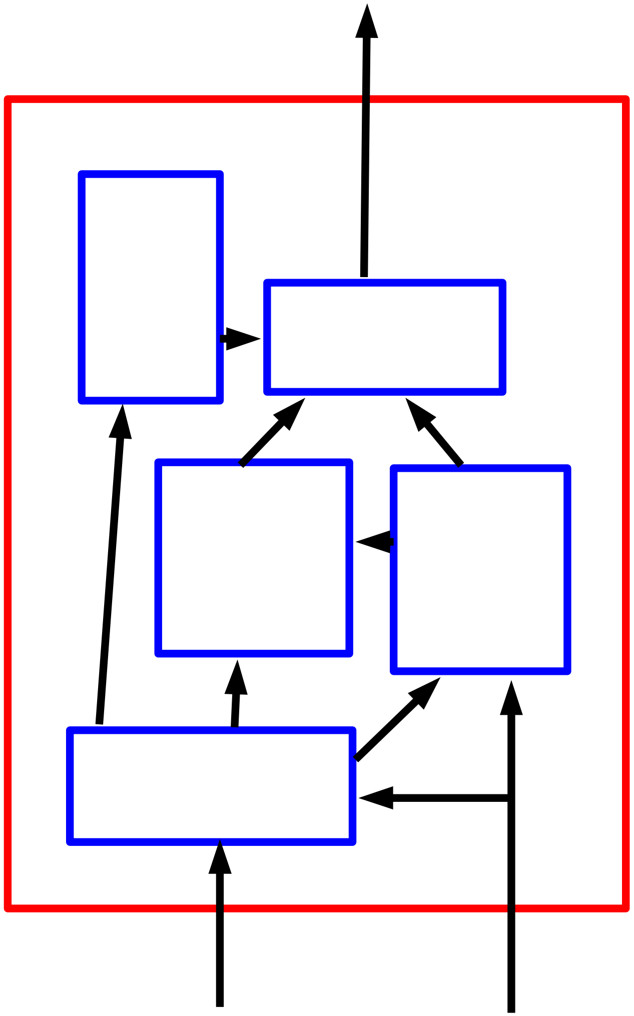


- A practical Application of Chain Rule

- Backprop for the state gradients:
- $dC/dX_{i-1} = dC/dX_i \,.\, dX_i/dX_{i-1}$
- $dC/dX_{i-1} = dC/dX_i \,.\, dF_i(X_{i-1},W_i)/dX_{i-1}$

- Backprop for the weight gradients:
- $dC/dW_i = dC/dX_i \,.\, dX_i/dW_i$
- $dC/dW_i = dC/dX_i \,.\, dF_i(X_{i-1},W_i)/dW_i$

Any Architecture works



- **Any connection graph is permissible**
  - Directed acyclic graphs (DAG)
  - Networks with loops must be “unfolded in time”.
- **Any module is permissible**
  - As long as it is continuous and differentiable almost everywhere with respect to the parameters, and with respect to non-terminal inputs.
- **Most frameworks provide automatic differentiation**
  - Theano, Torch7+autograd,…
  - Programs are turned into computation DAGs and automatically differentiated.

The Objective Function of Multi-layer Nets is Non Convex

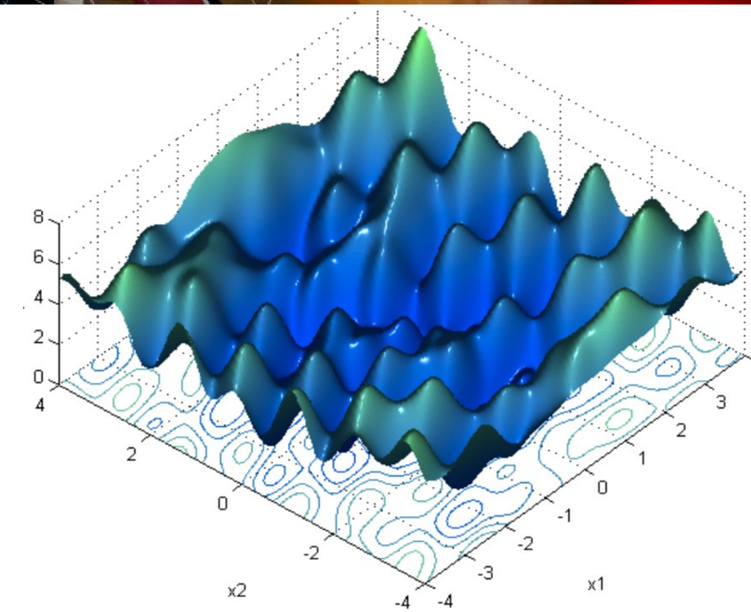
Y LeCun

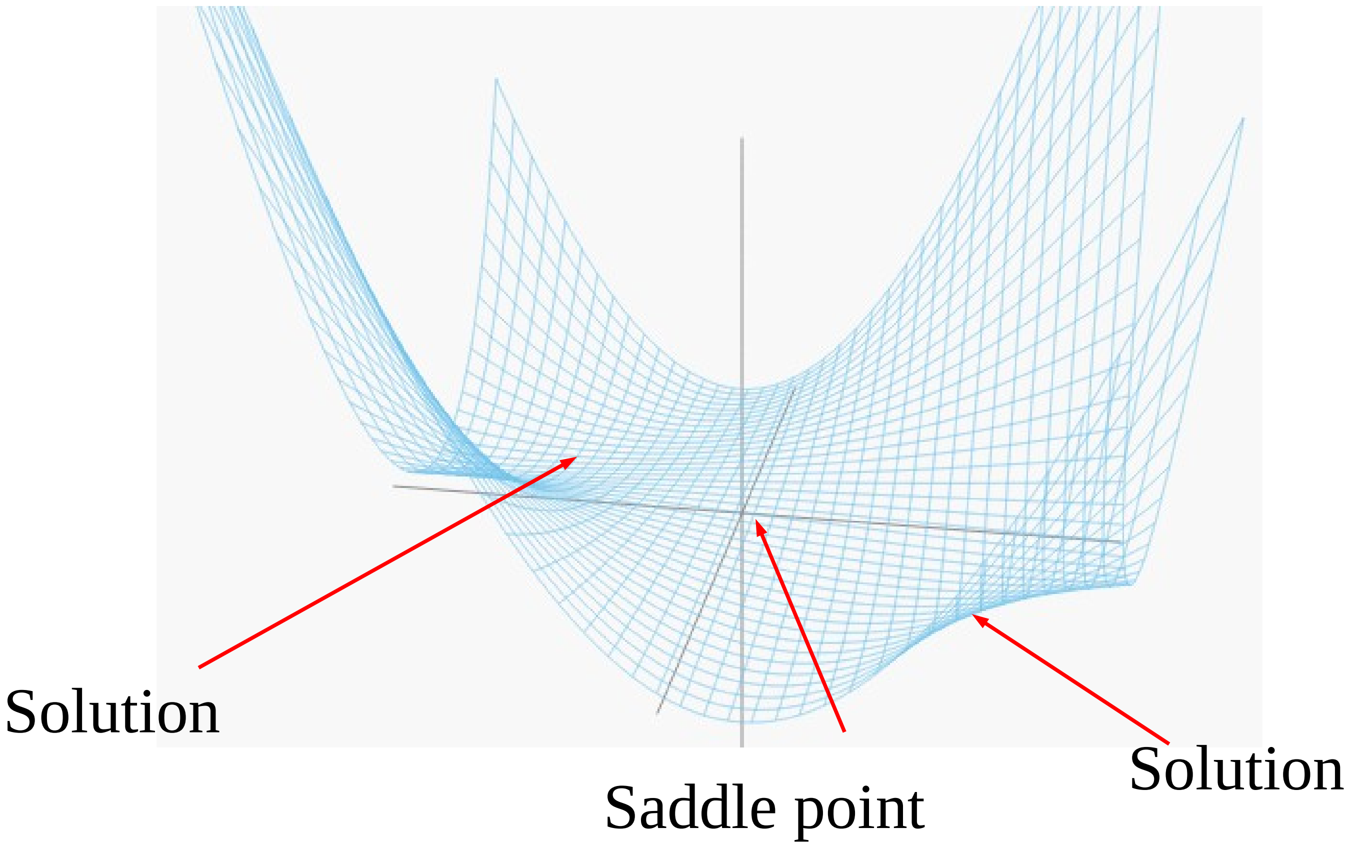
1-1-1 network

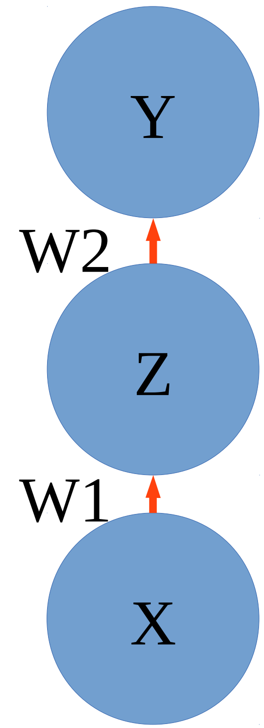
- Y = W1*W2*X

Objective: identity function with quadratic loss

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





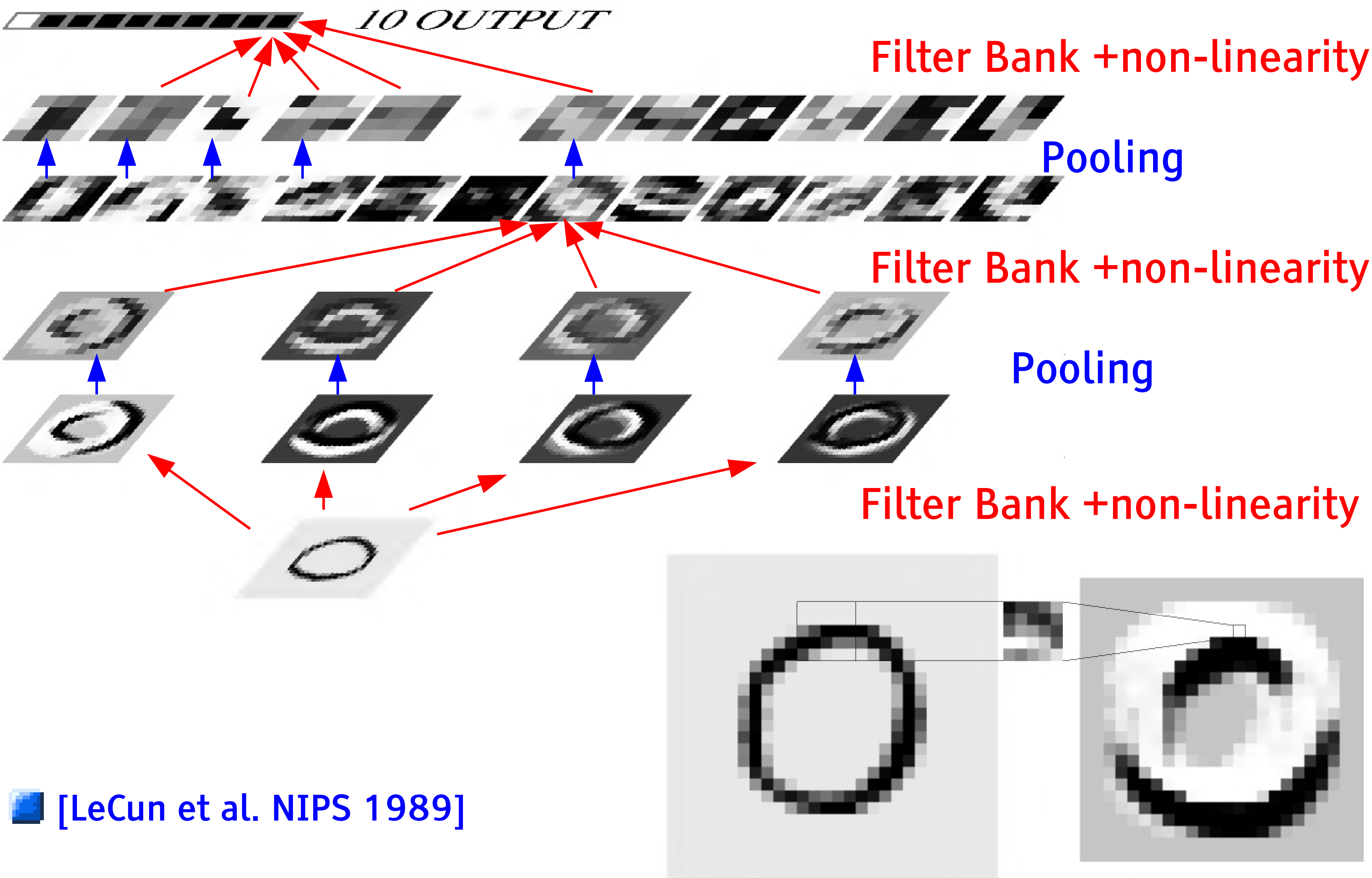


Convolutional Networks

(ConvNet or CNN)

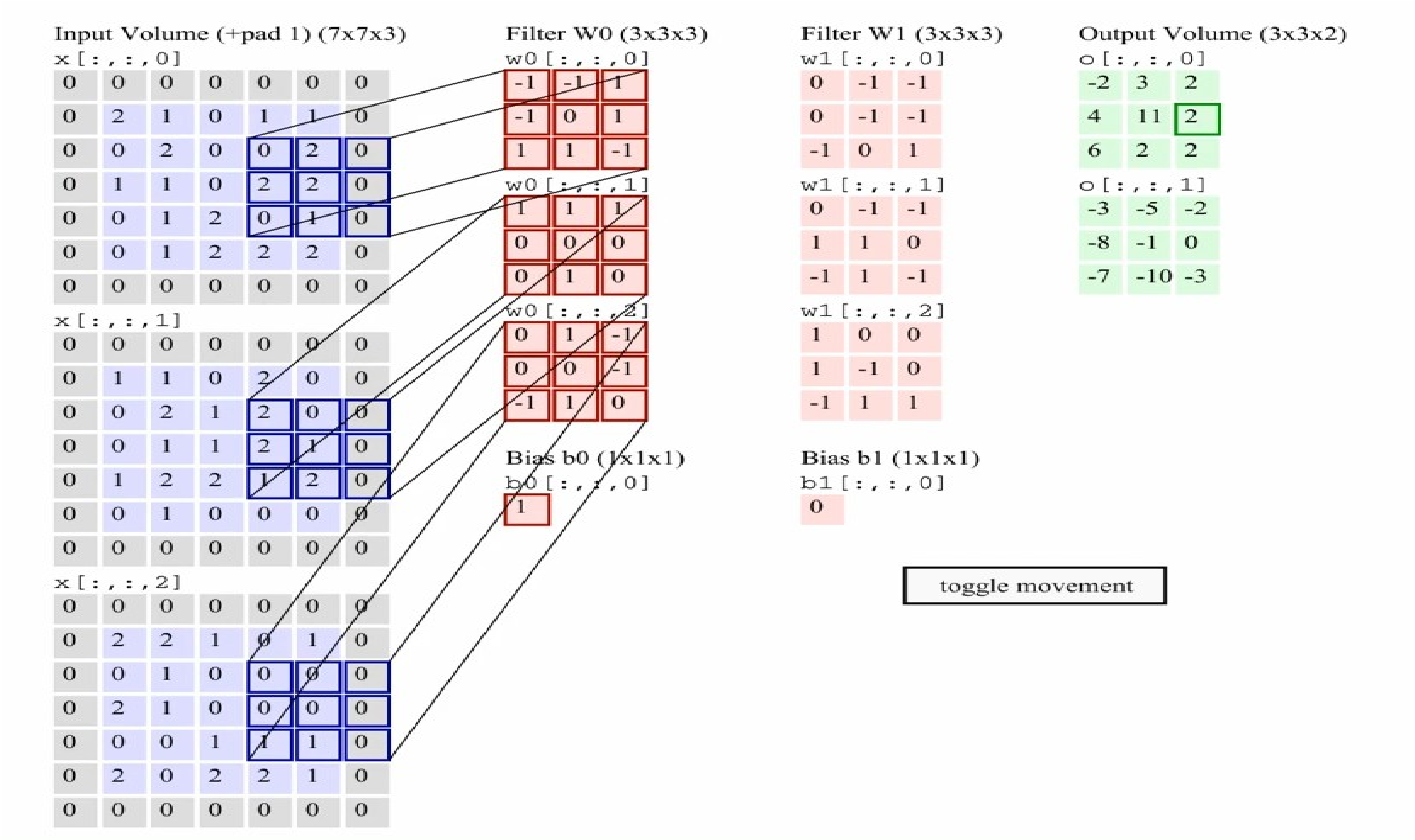
Convolutional Network Architecture

Y LeCun



- [LeCun et al. NIPS 1989]

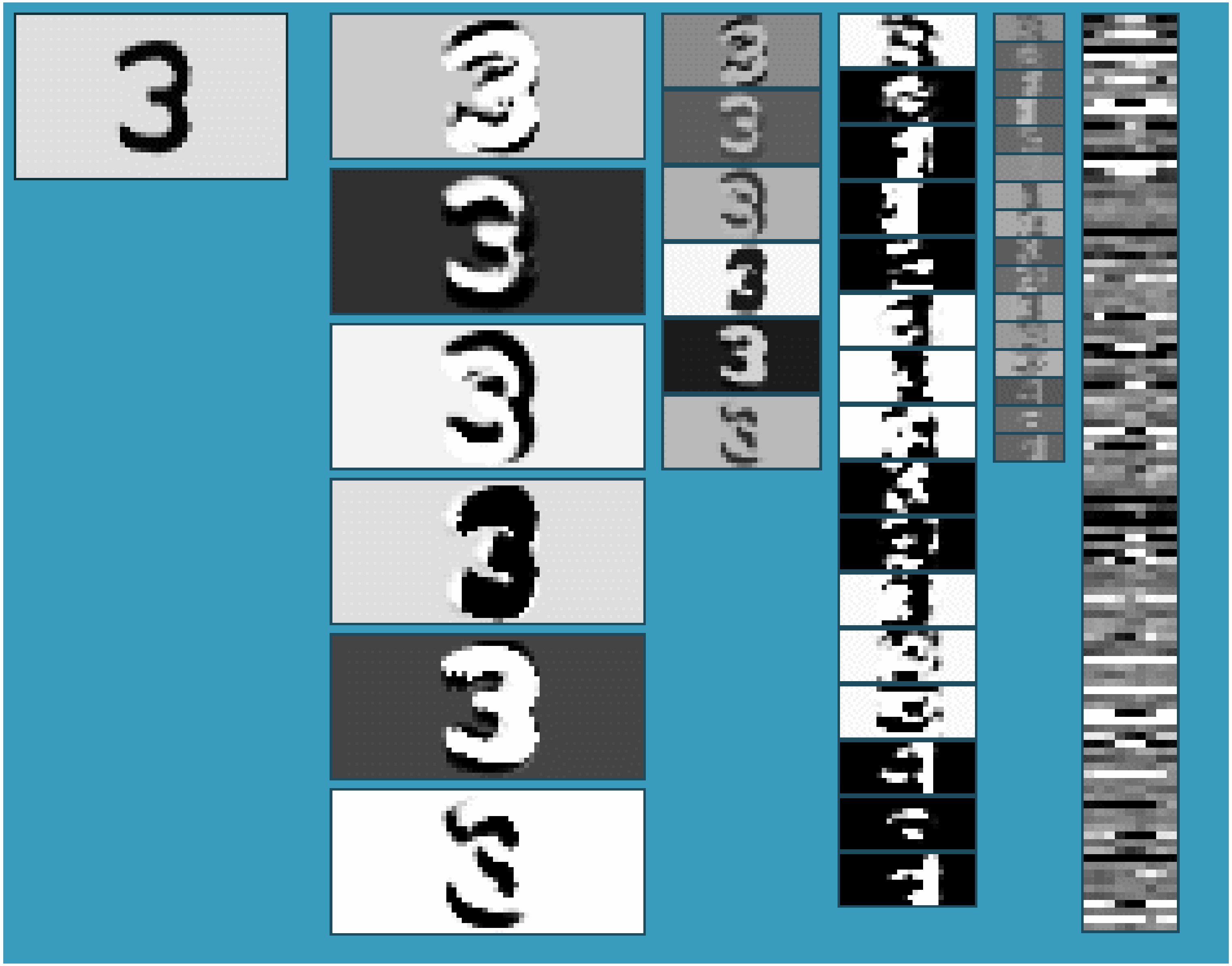
Multiple Convolutions



Animation: Andrej Karpathy http://cs231n.github.io/convolutional-networks/

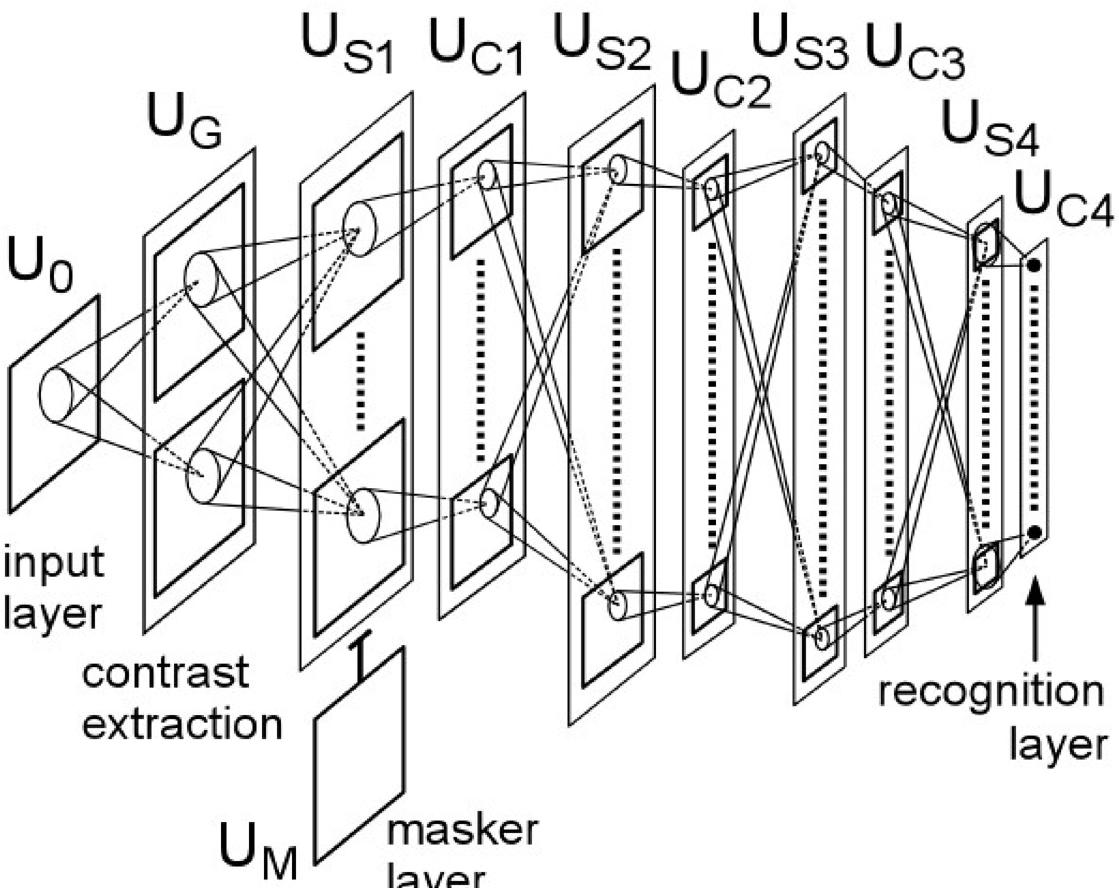
Convolutional Network (vintage 1990)

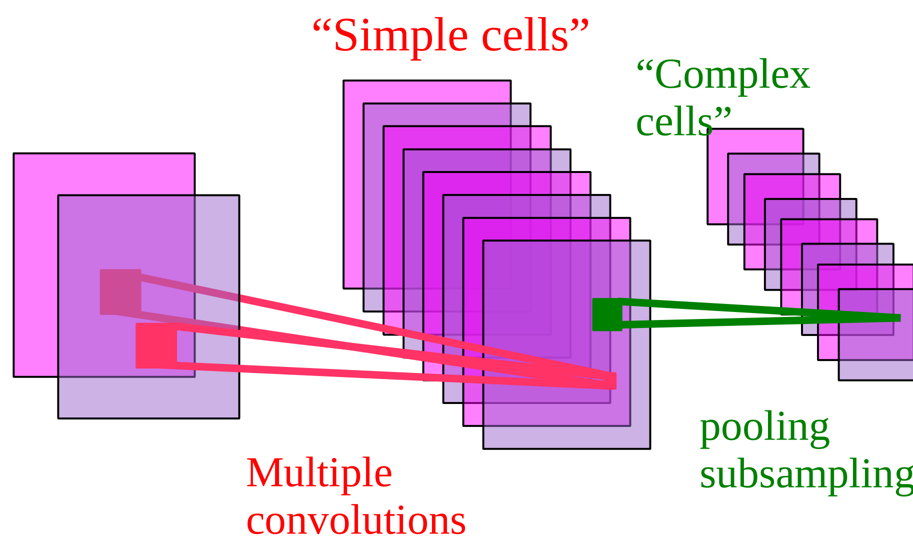
- Filters-tanh → pooling → filters-tanh → pooling → filters-tanh



Hubel & Wiesel's Model of the Architecture of the Visual Cortex

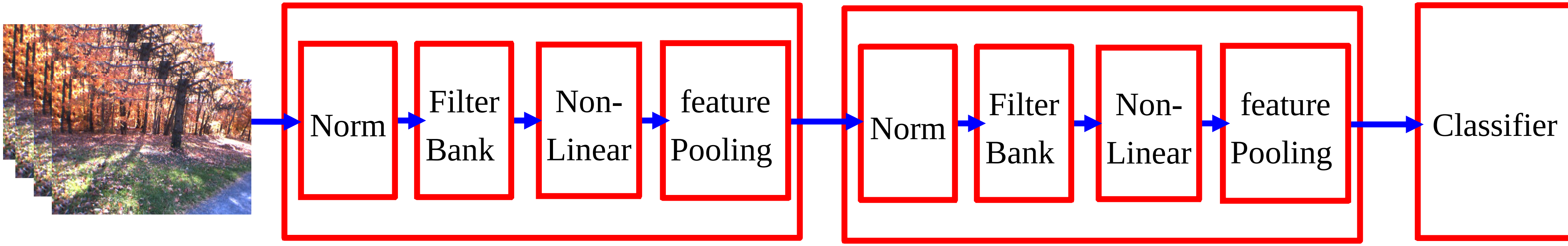
- [Hubel & Wiesel 1962]:
  - simple cells detect local features
  - complex cells "pool" the outputs of simple cells within a retinotopic neighborhood.





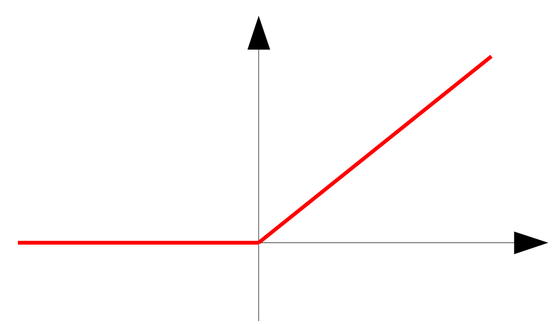
[Fukushima 1982][LeCun 1989, 1998],[Riesenhuber 1999]......

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



- **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.

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

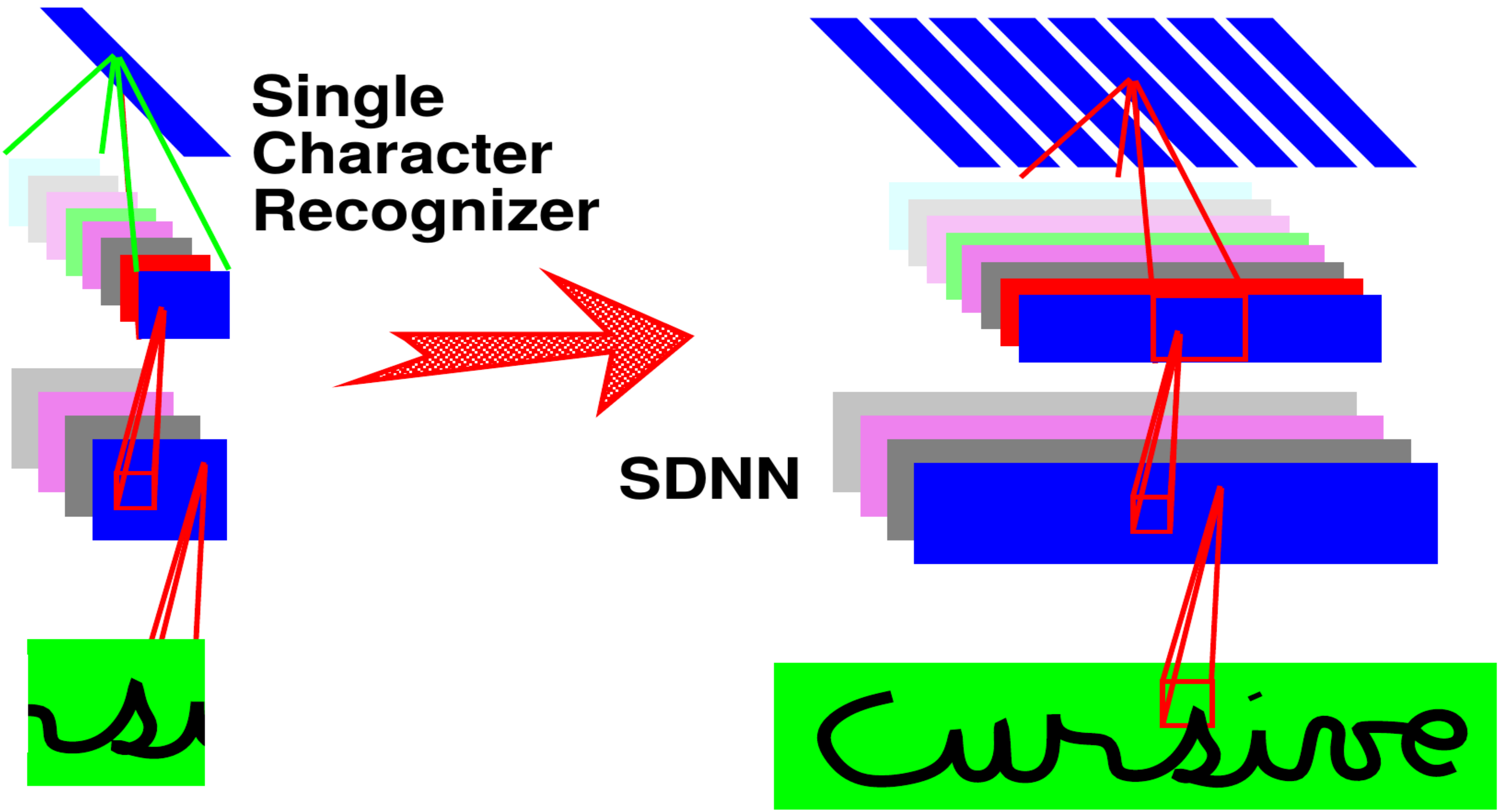
LeNet1 Demo from 1993

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

VIDEO: LENET 1992

Multiple Character Recognition [Matan et al 1992]

- Every layer is a convolution



Sliding Window ConvNet + Weighted Finite-State Machine

Sliding Window ConvNet + Weighted FSM

Check Reader (Bell Labs, 1995)

Graph transformer network trained to read check amounts.

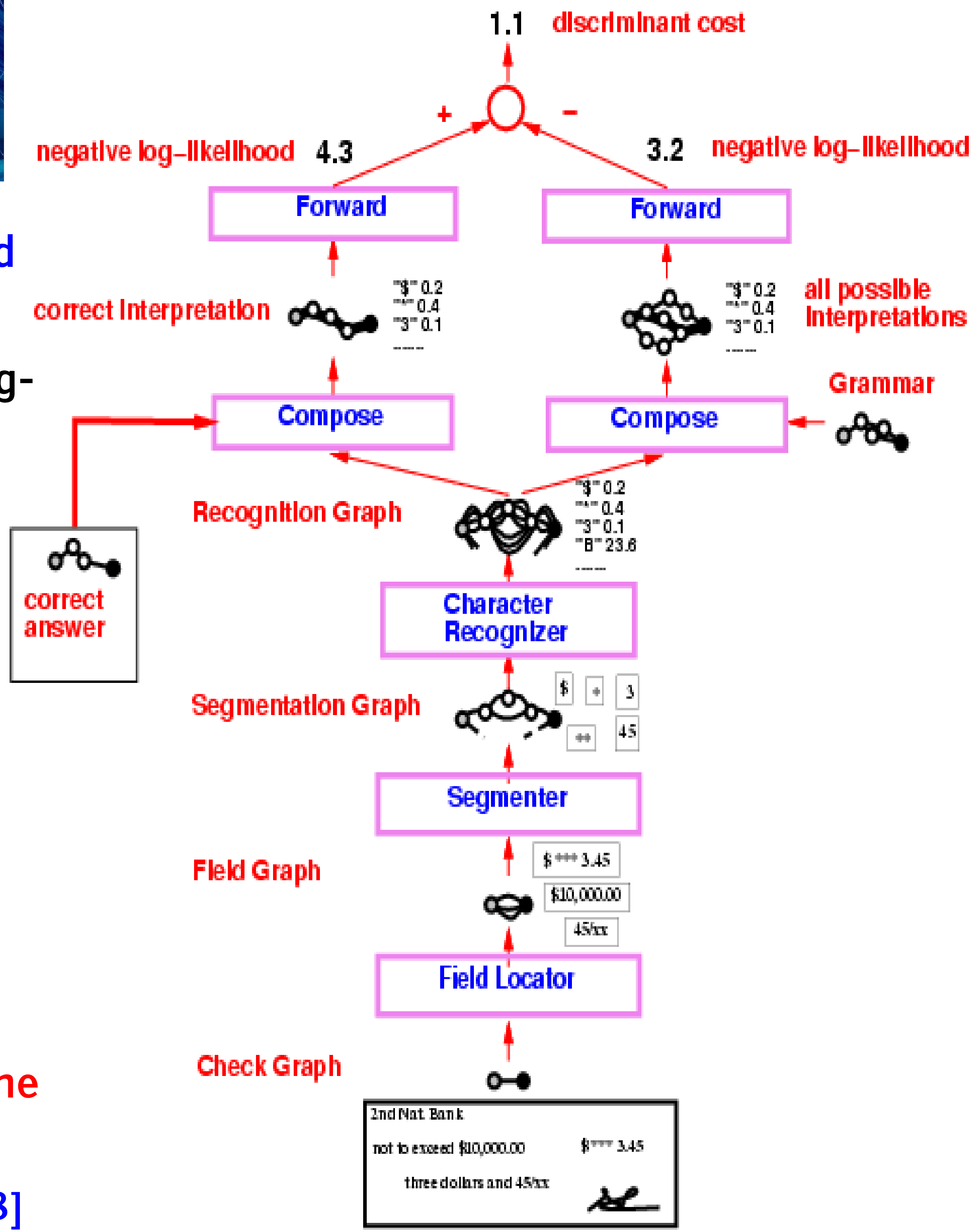
Trained globally with Negative-Log-Likelihood loss.

50% percent correct, 49% reject, 1% error (detectable later in the process).

Fielded in 1996, used in many banks in the US and Europe.

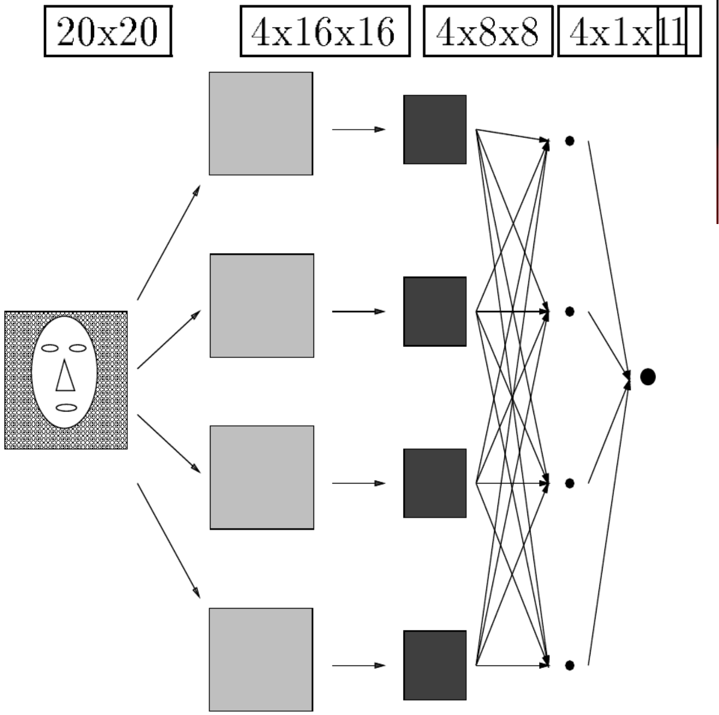
Processed an estimated 10% to 20% of all the checks written in the US in the early 2000s.

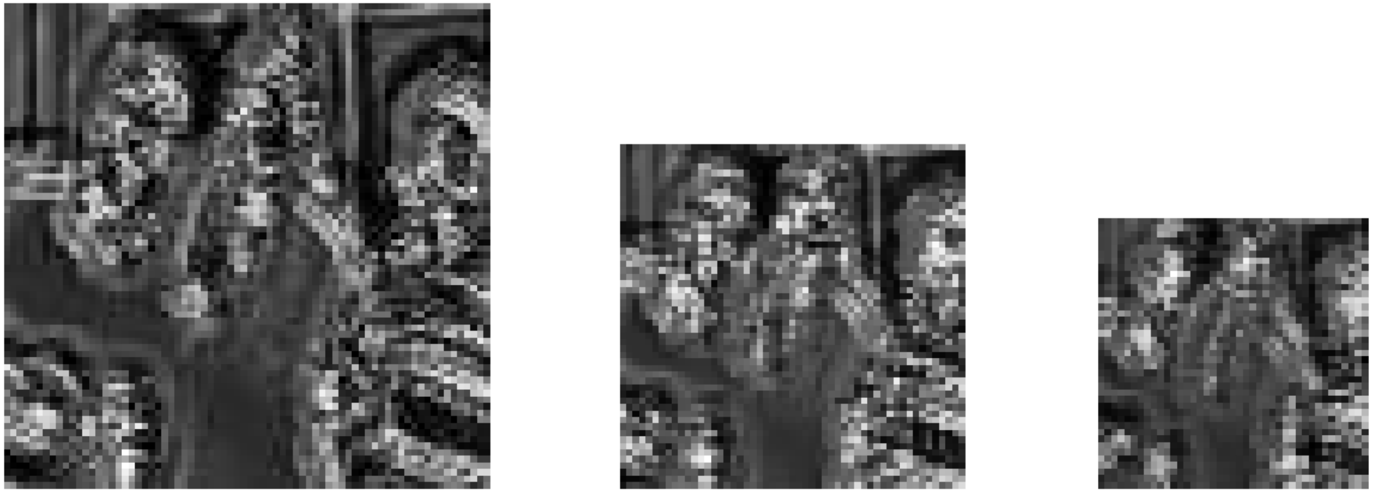
[LeCun, Bottou, Bengio, Haffner 1998]



Face Detection [Vaillant et al. 93, 94]

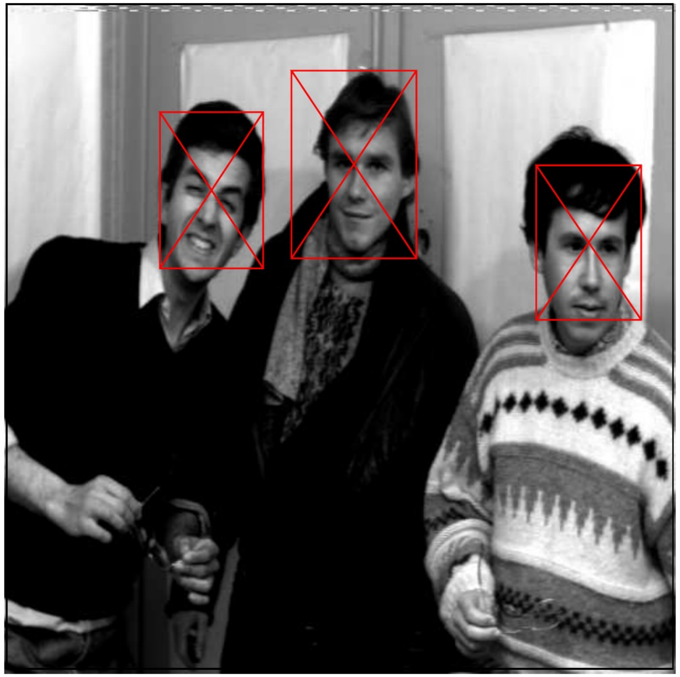
- ConvNet applied to large images
- Heatmaps at multiple scales
- Non-maximum suppression for candidates
- 6 second on a Sparcstation for 256x256 image



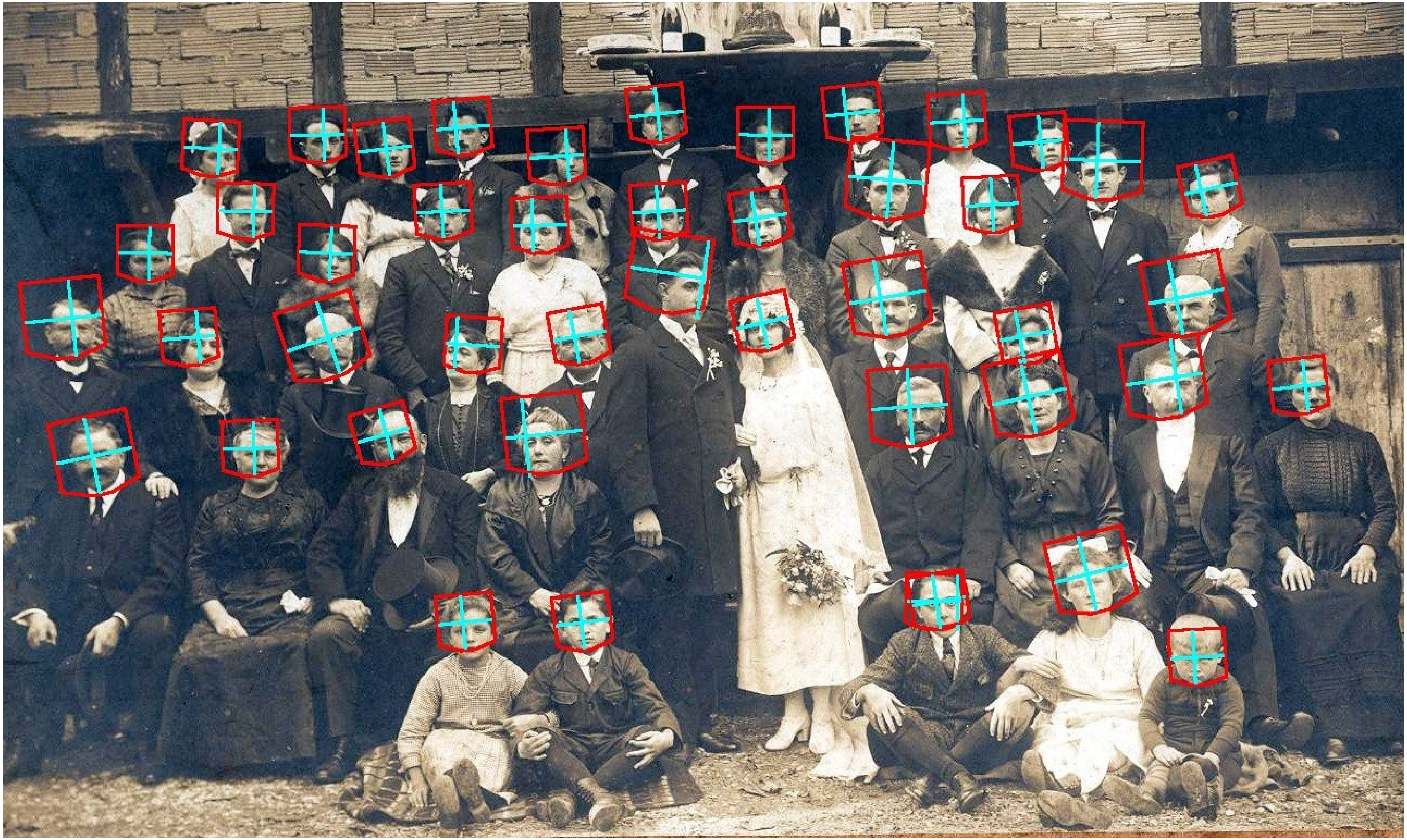


Scale 3 Scale 4 Scale 5

Scale 6 Scale 7 Scale 8 Scale 9

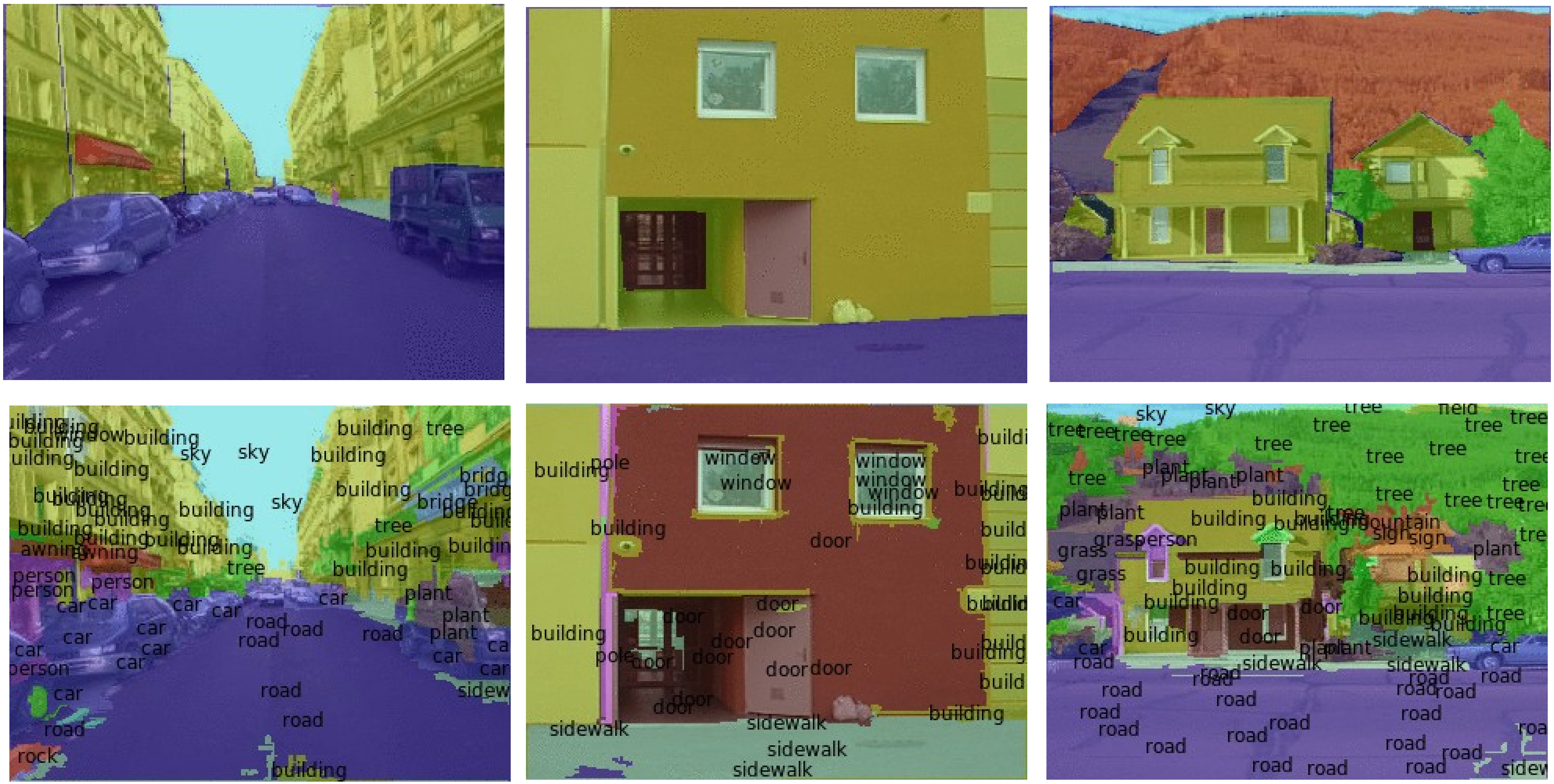


Simultaneous face detection and pose estimation



VIDEO: PEDESTRIAN DETECTION

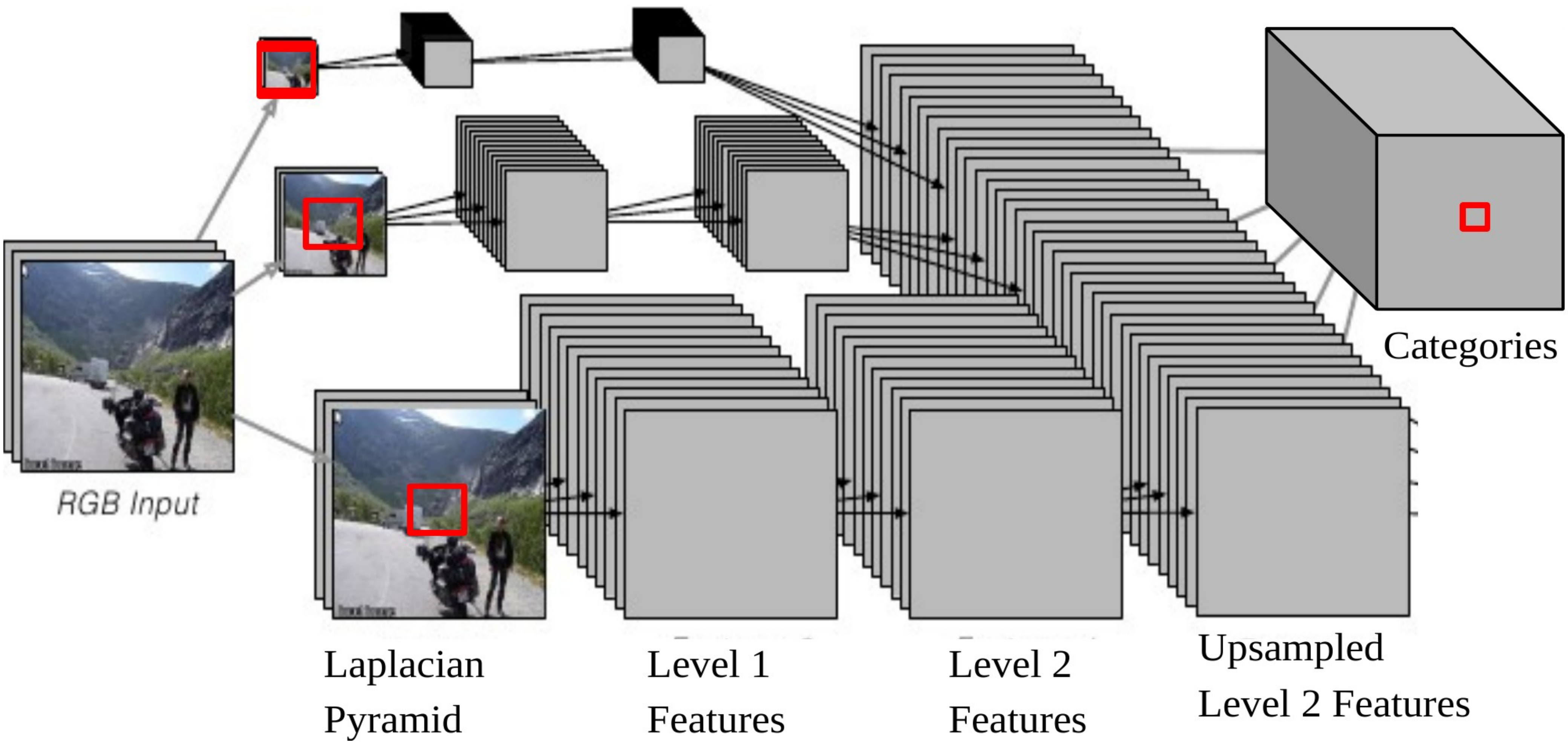
Scene Parsing/Labeling



[Farabet et al. ICML 2012, PAMI 2013]

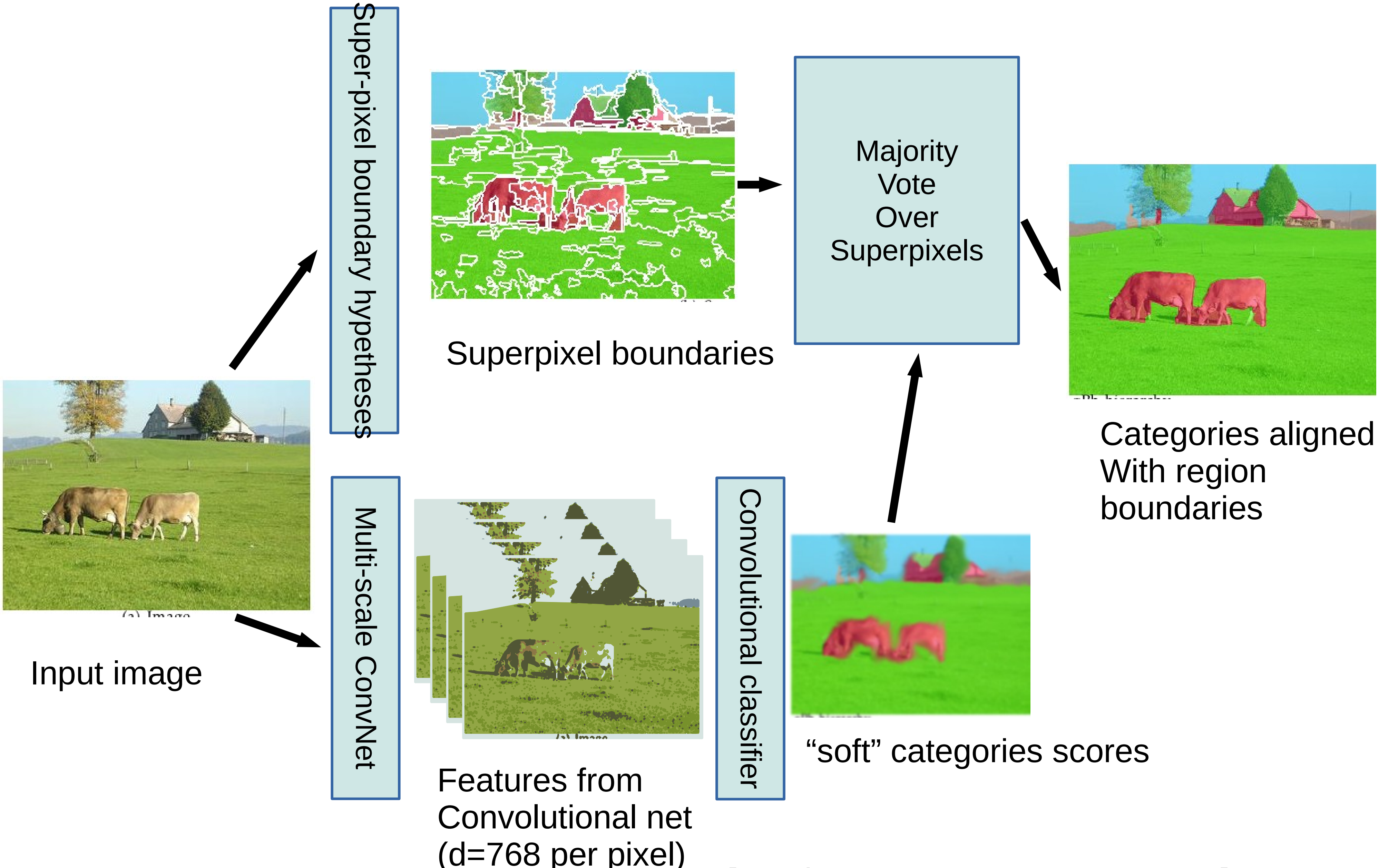
Scene Parsing/Labeling: Multiscale ConvNet Architecture

- **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



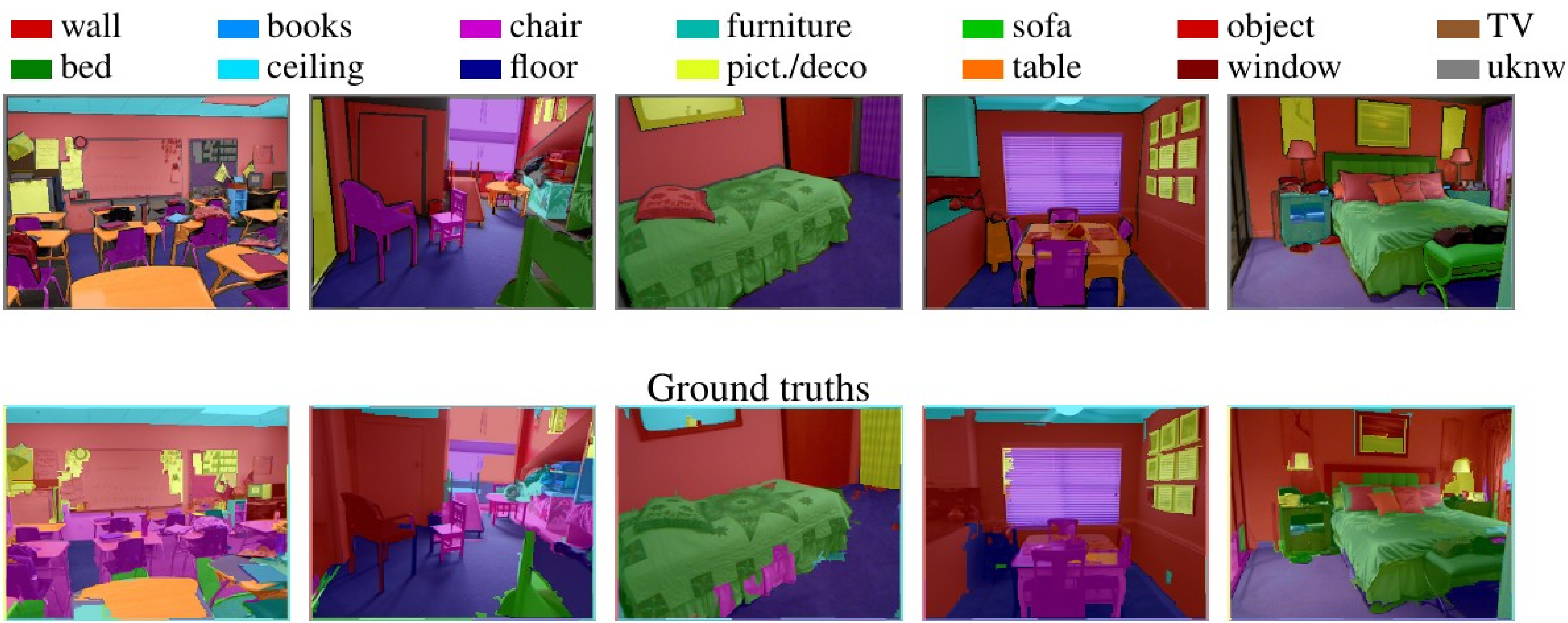
Method 1: majority over super-pixel regions

Y LeCun



[Farabet et al. IEEE T. PAMI 2013]

Scene Parsing/Labeling on RGB+Depth Images



Ground truths

Our results

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

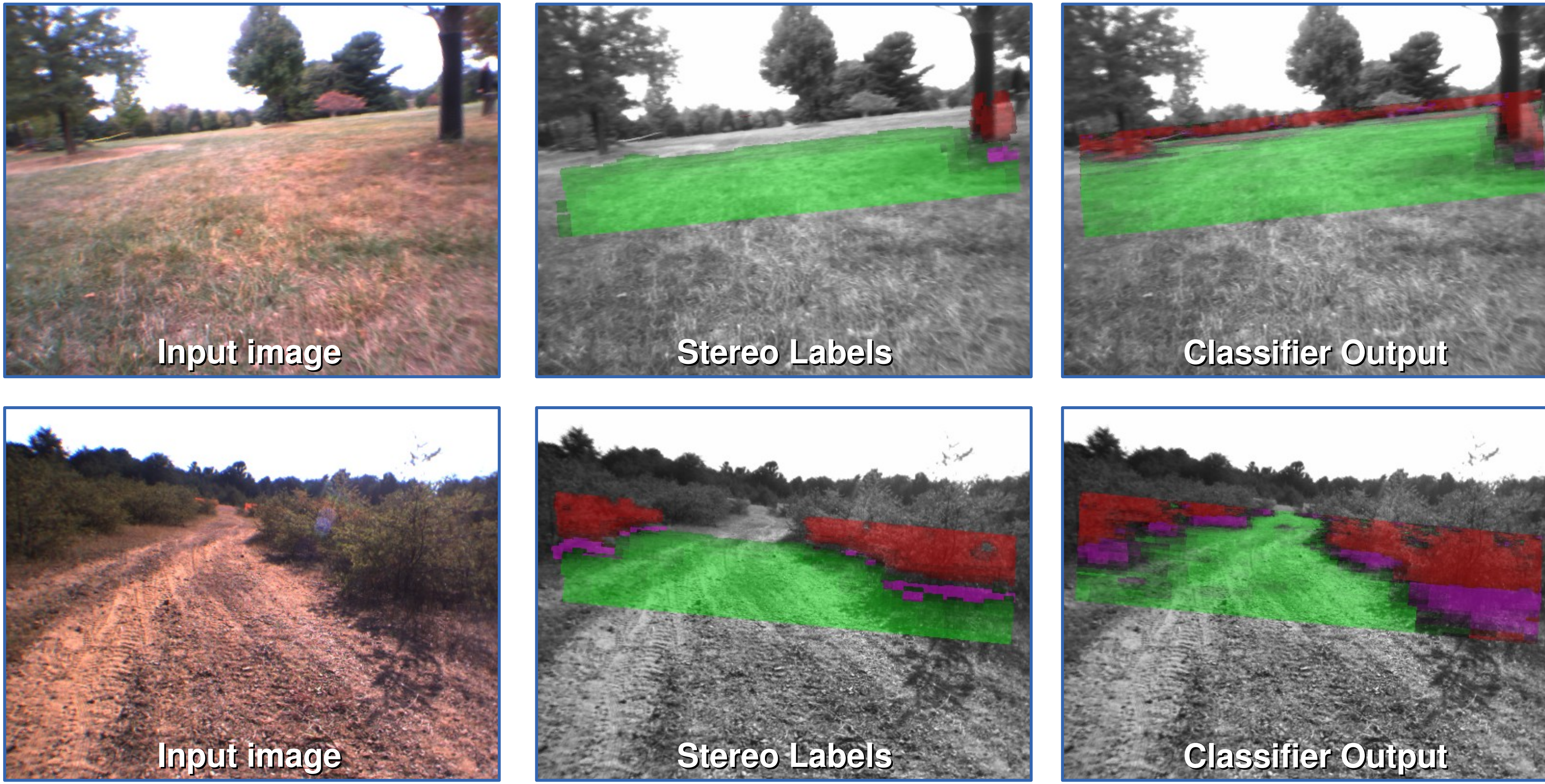
Scene Parsing/Labeling

- 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

VIDEO: SCENE PARSING

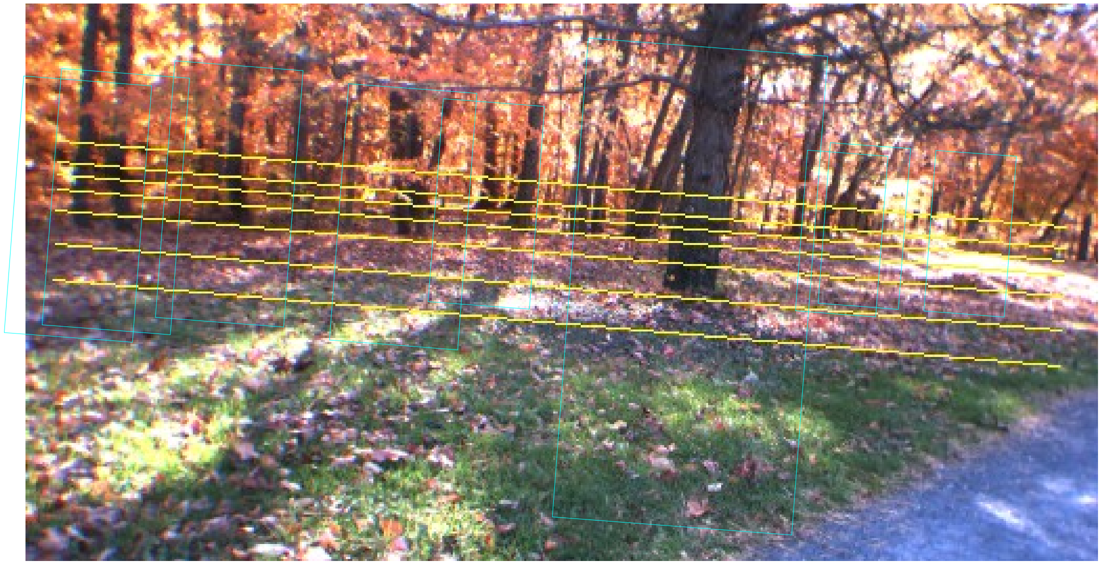
[Farabet et al. ICML 2012, PAMI 2013]

ConvNet for Long Range Adaptive Robot Vision (DARPA LAGR program 2005-2008)



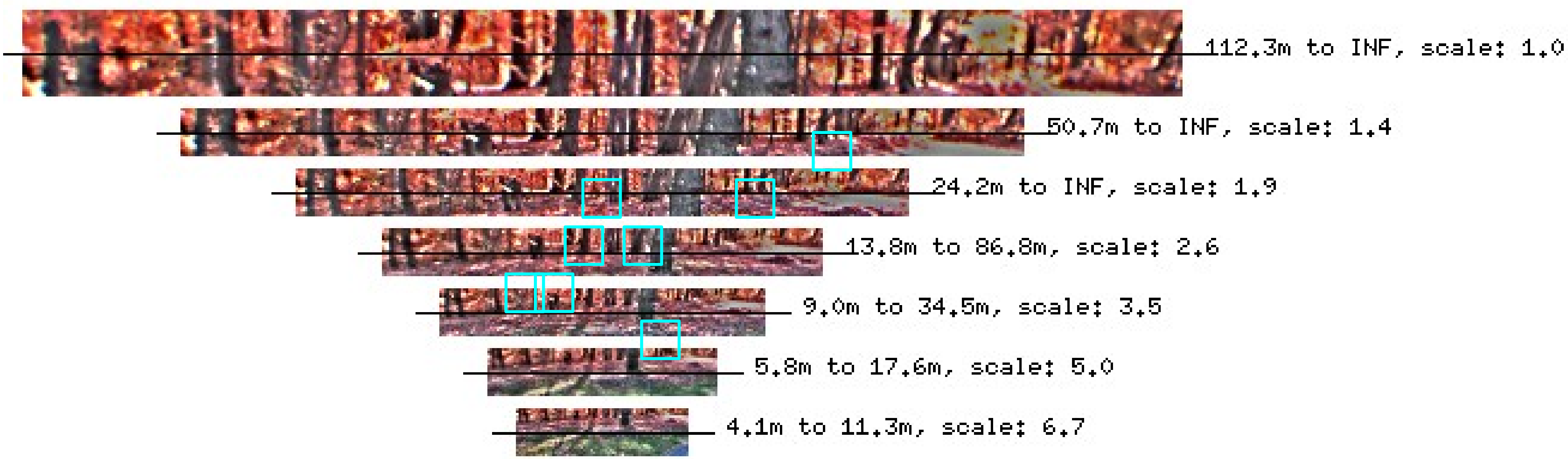
Long Range Vision with a Convolutional Net

Y LeCun



Pre-processing (125 ms)

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

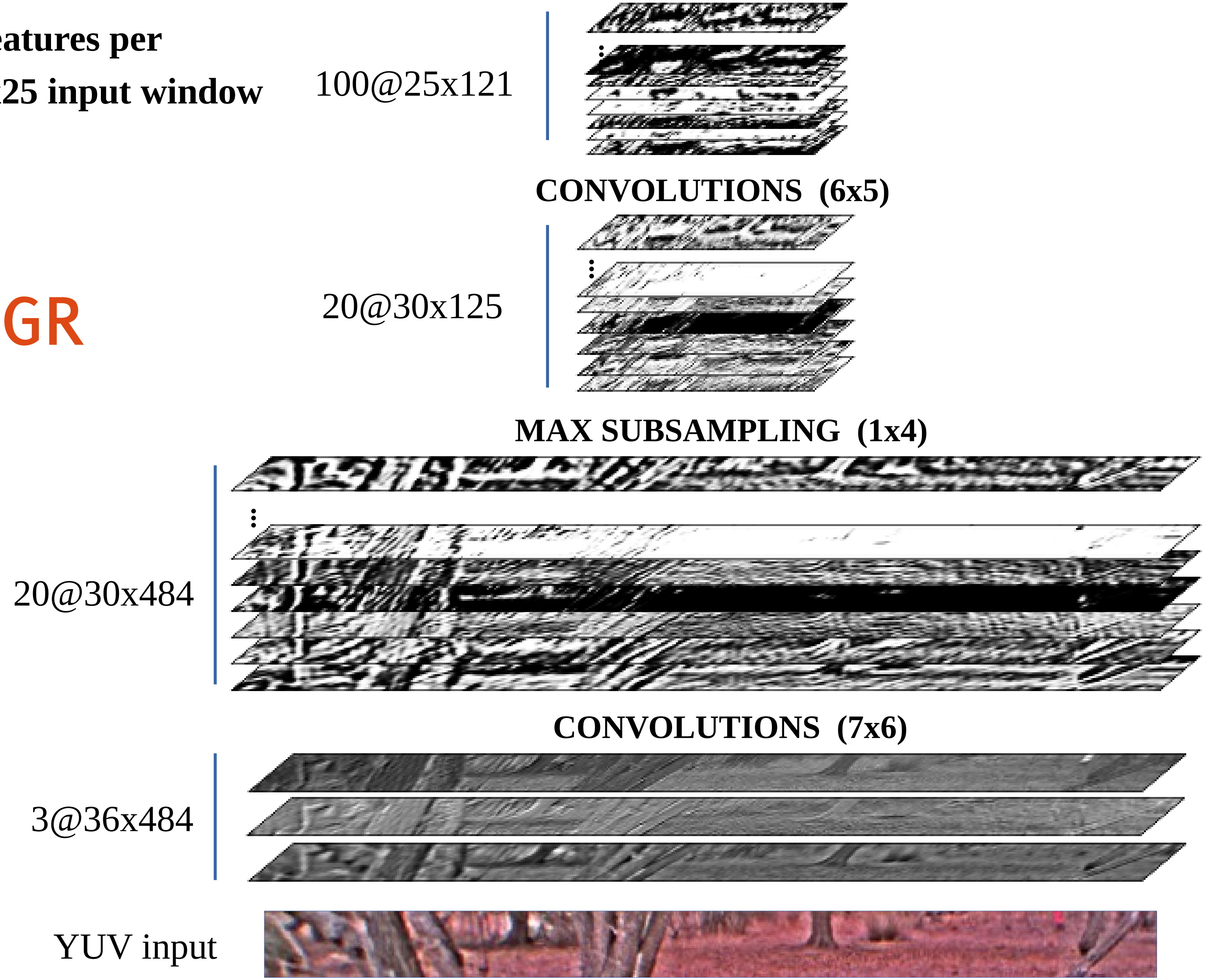


Convolutional Net Architecture

Y LeCun

100 features per
3x12x25 input window

VIDEO: LAGR

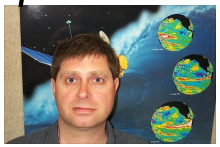


YUV image band
20-36 pixels tall,
36-500 pixels wide

Visual Object Recognition with Convolutional Nets

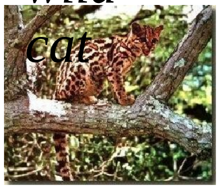
- **In the mid 2000s, ConvNets were getting decent results on object classification**
- **Dataset: “Caltech101”:**
  - 101 categories
  - 30 training samples per category
- **But the results were slightly worse than more “traditional” computer vision methods, because:**
  - 1. the datasets were too small
  - 2. the computers were too slow

face

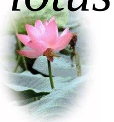


beaver

wild cat



lotus



an t

dollar

metronom e

w. chair

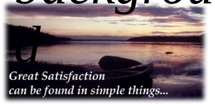
minare

cellphon

joshua t

cougar body

backgroun d



Then., two things happened...

- **The ImageNet dataset [Fei-Fei et al. 2012]**
  - 1.2 million training samples
  - 1000 categories

- **Fast & Programmable General-Purpose GPUs**
  - NVIDIA CUDA
  - Capable of over 1 trillion operations/second

Matchstick

Sea lion

Flute

Strawberry

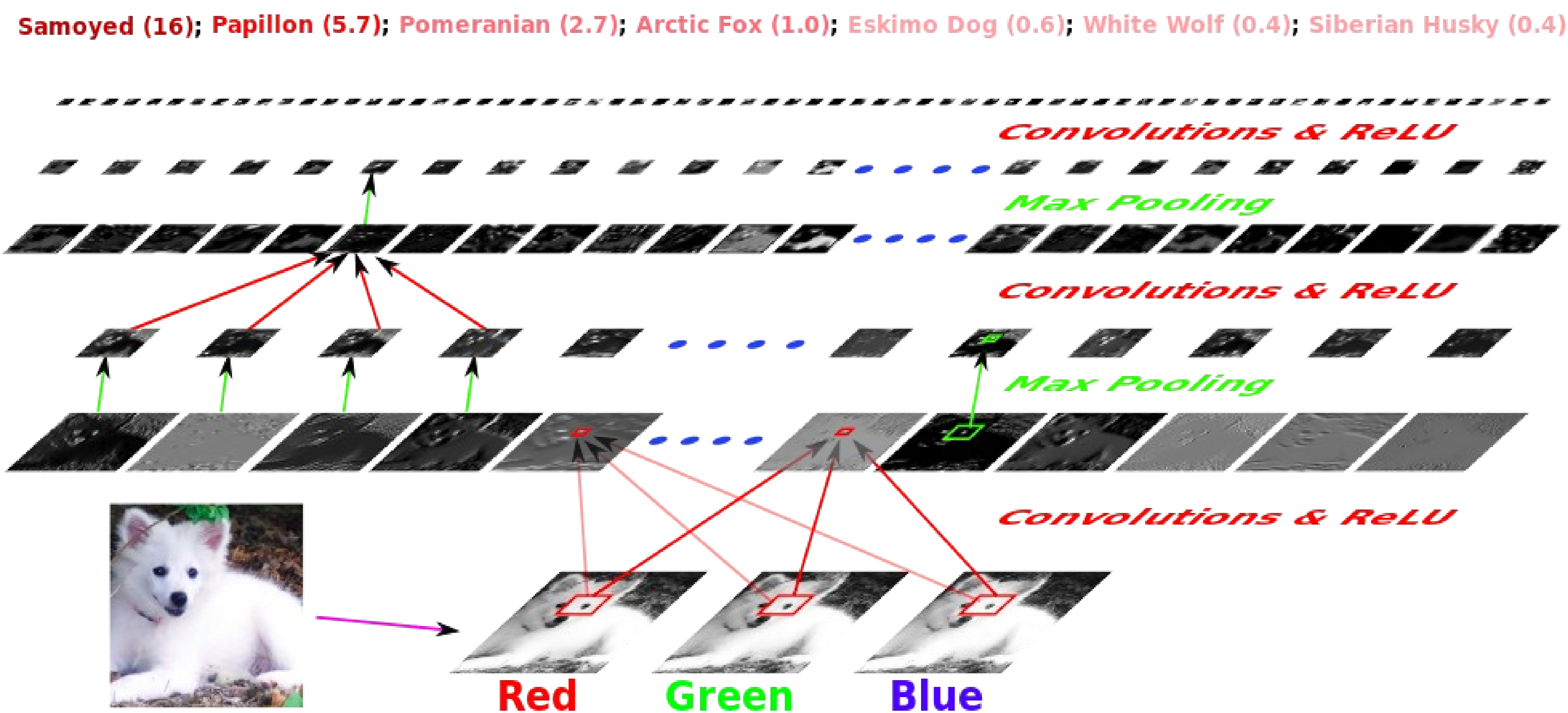
Bathing cap

Backpack

Racket

Very Deep ConvNet for Object Recognition

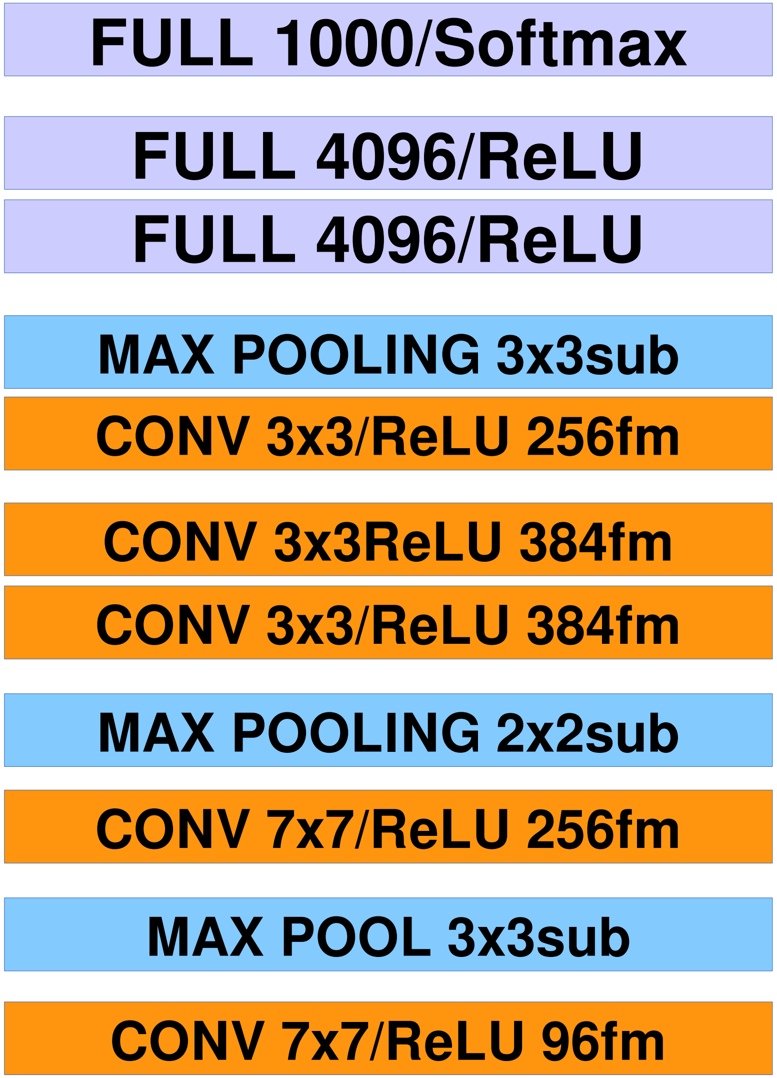
- **1 to 10 billion connections, 10 million to 1 billion parameters, 8 to 20 layers.**



Very Deep ConvNets Trained on GPU

- AlexNet [Krizhevski, Sutskever, Hinton 2012]
  - 15% top-5 error on ImageNet
- OverFeat [Sermanet et al. 2013]
  - 13.8%
- VGG Net [Simonyan, Zisserman 2014]
  - 7.3%
- GoogLeNet [Szegedy et al. 2014]
  - 6.6%
- ResNet [He et al. 2015]
  - 5.7%

- http://torch.ch
- https://github.com/torch/torch7/wiki/Cheatsheet



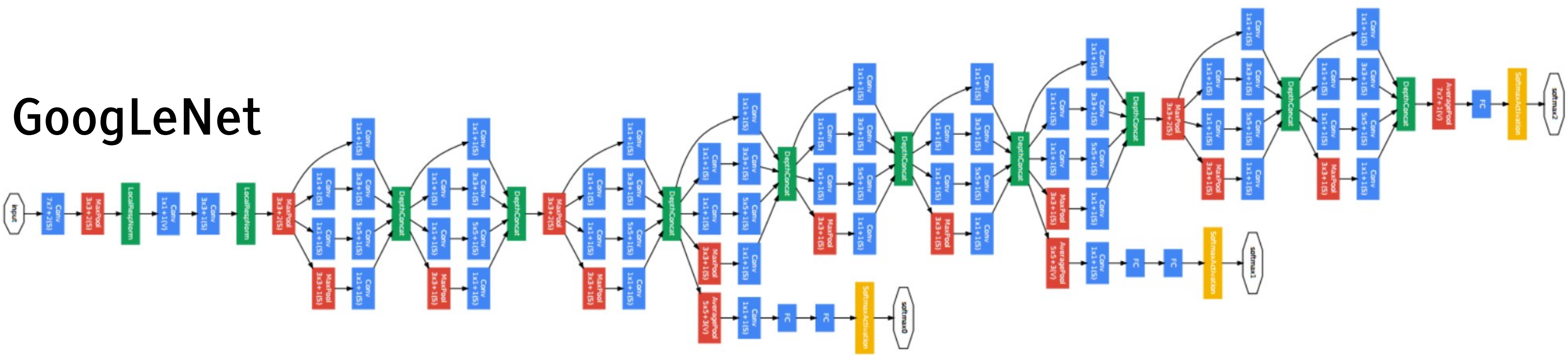
Very Deep ConvNet Architectures

Y LeCun

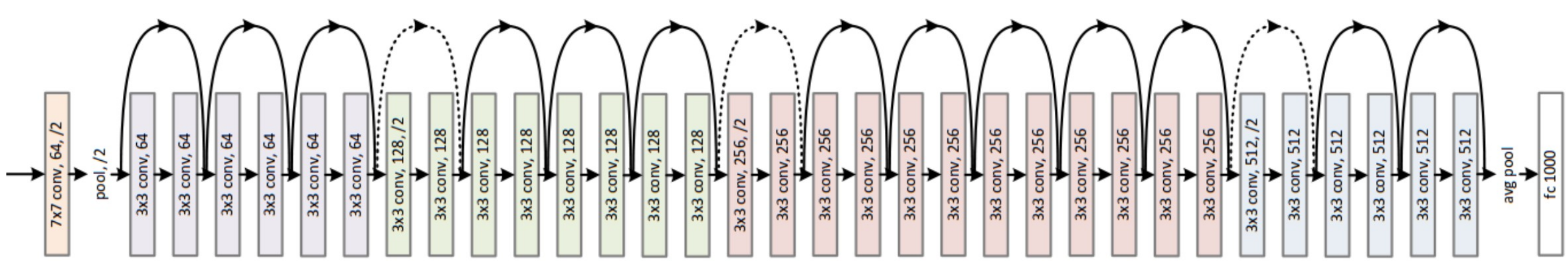
- Small kernels, not much subsampling (fractional subsampling).

VGG

GoogLeNet

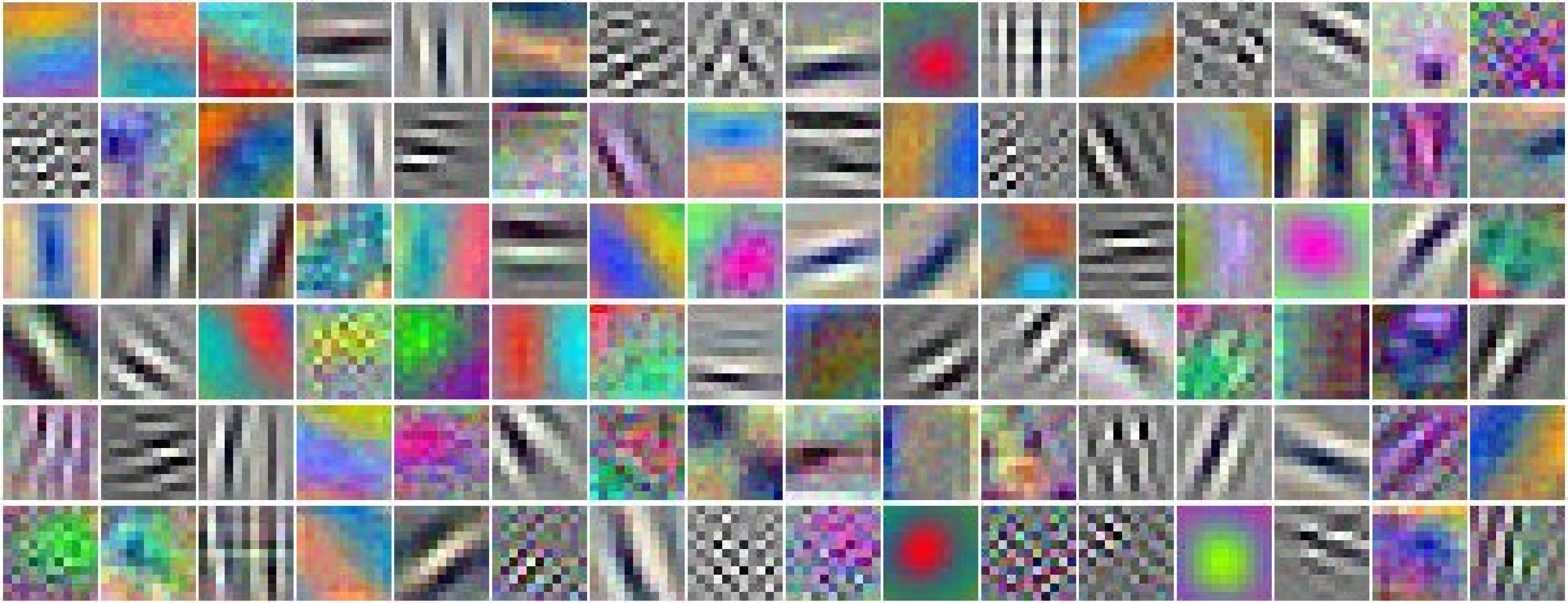


ResNet



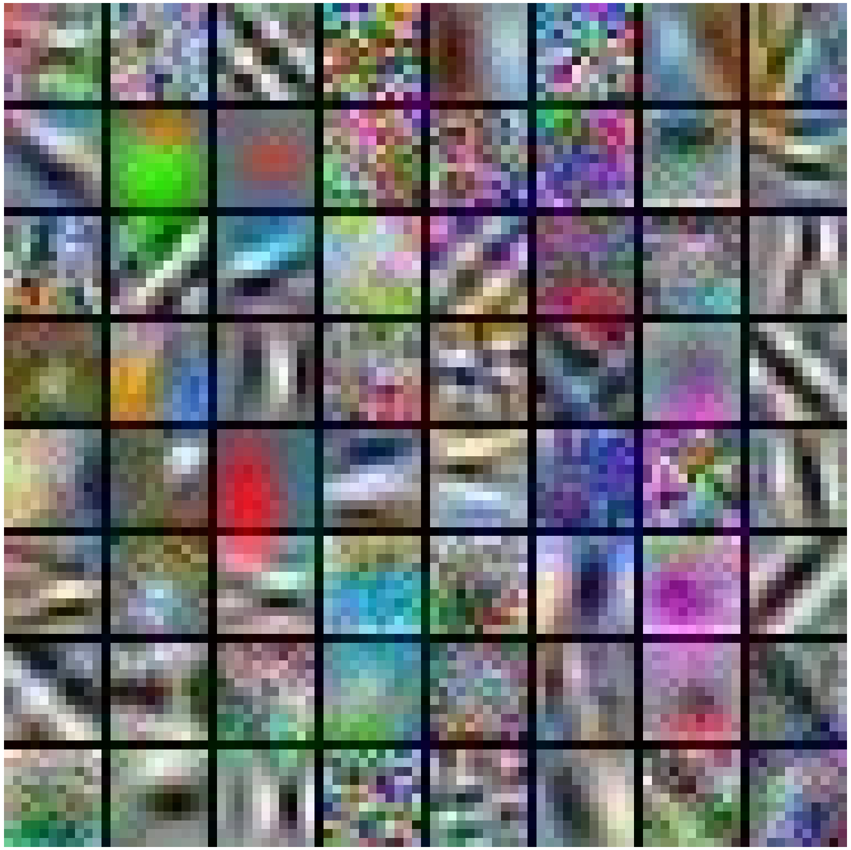
Kernels: Layer 1 (11x11)

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



Learning in Action

- **How the filters in the first layer learn**

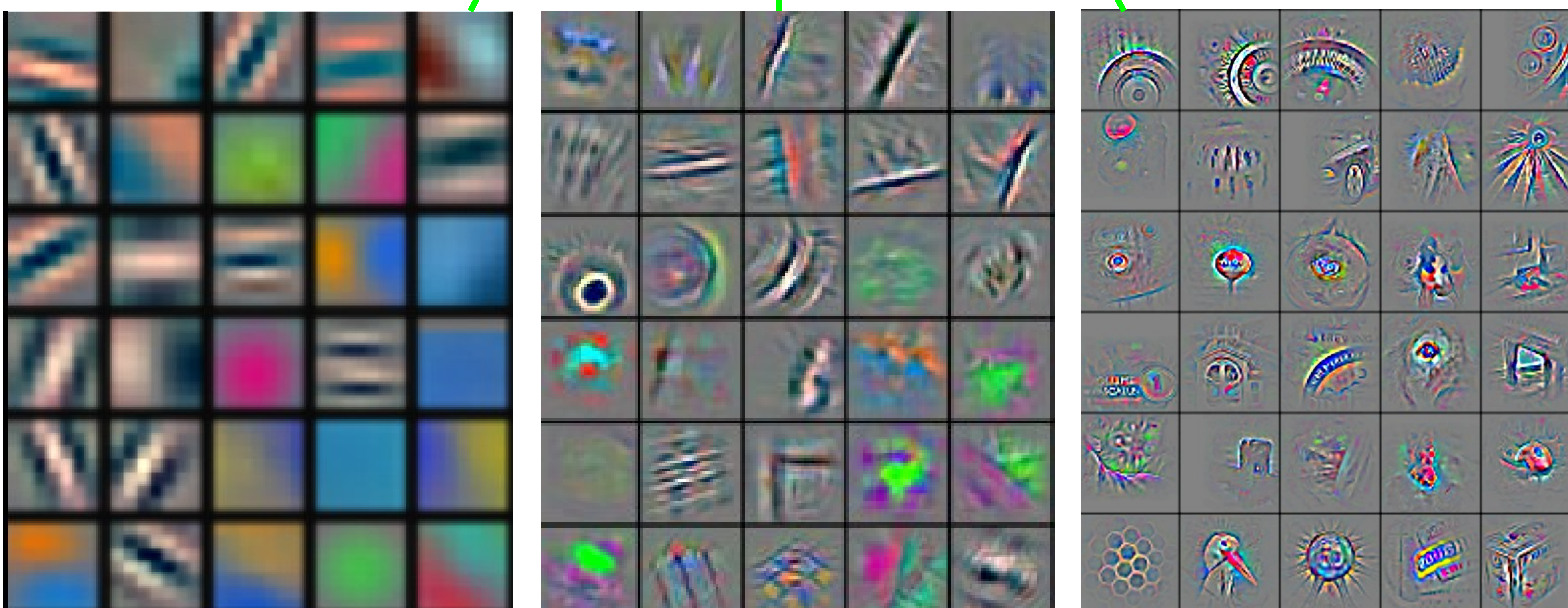


Deep Learning = Learning Hierarchical Representations

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]

ImageNet: Classification

Y LeCun

- **Give the name of the dominant object in the image**
- **Top-5 error rates: if correct class is not in top 5, count as error**
  - Black:ConvNet, Purple: no ConvNet

2012 Teams	%error
Supervision (Toronto)	15.3
ISI (Tokyo)	26.1
VGG (Oxford)	26.9
XRCE/INRIA	27.0
UvA (Amsterdam)	29.6
INRIA/LEAR	33.4

2013 Teams	%error
Clarifai (NYU spinoff)	11.7
NUS (singapore)	12.9
Zeiler-Fergus (NYU)	13.5
A. Howard	13.5
OverFeat (NYU)	14.1
UvA (Amsterdam)	14.2
Adobe	15.2
VGG (Oxford)	15.2
VGG (Oxford)	23.0

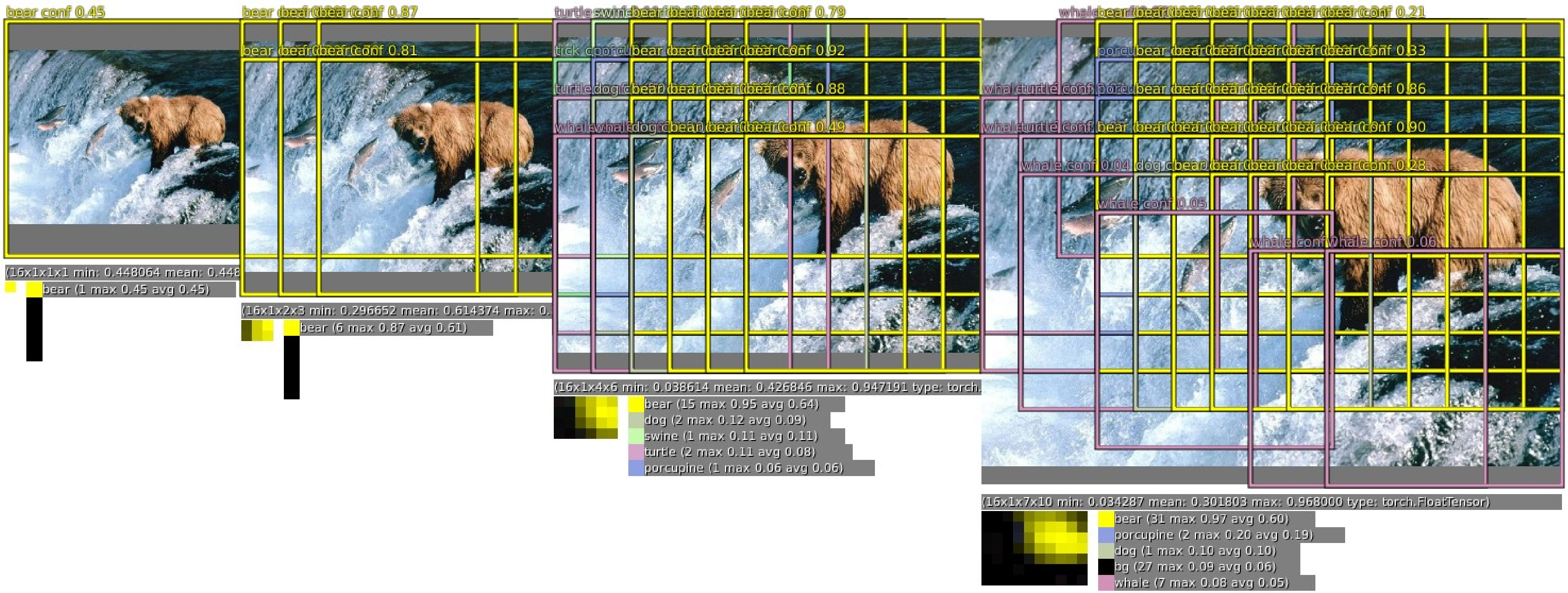
2014 Teams	%error
GoogLeNet	6.6
VGG (Oxford)	7.3
MSRA	8.0
A. Howard	8.1
DeeperVision	9.5
NUS-BST	9.7
TTIC-ECP	10.2
XYZ	11.2
UvA	12.1

Object Detection And Localization With ConvNets

Classification + Localization: multiscale sliding window

Y LeCun

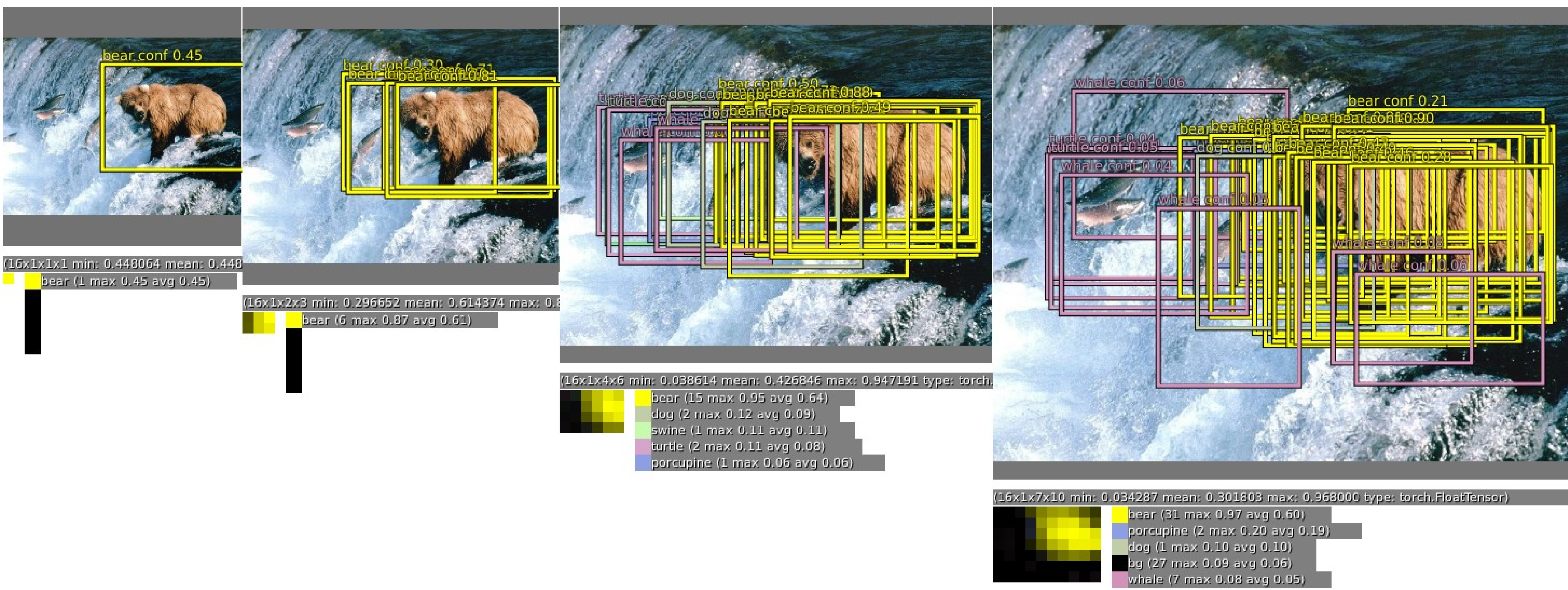
- **Apply convnet with a sliding window over the image at multiple scales**
- **Important note: it's very cheap to slide a convnet over an image**
  - Just compute the convolutions over the whole image and replicate the fully-connected layers



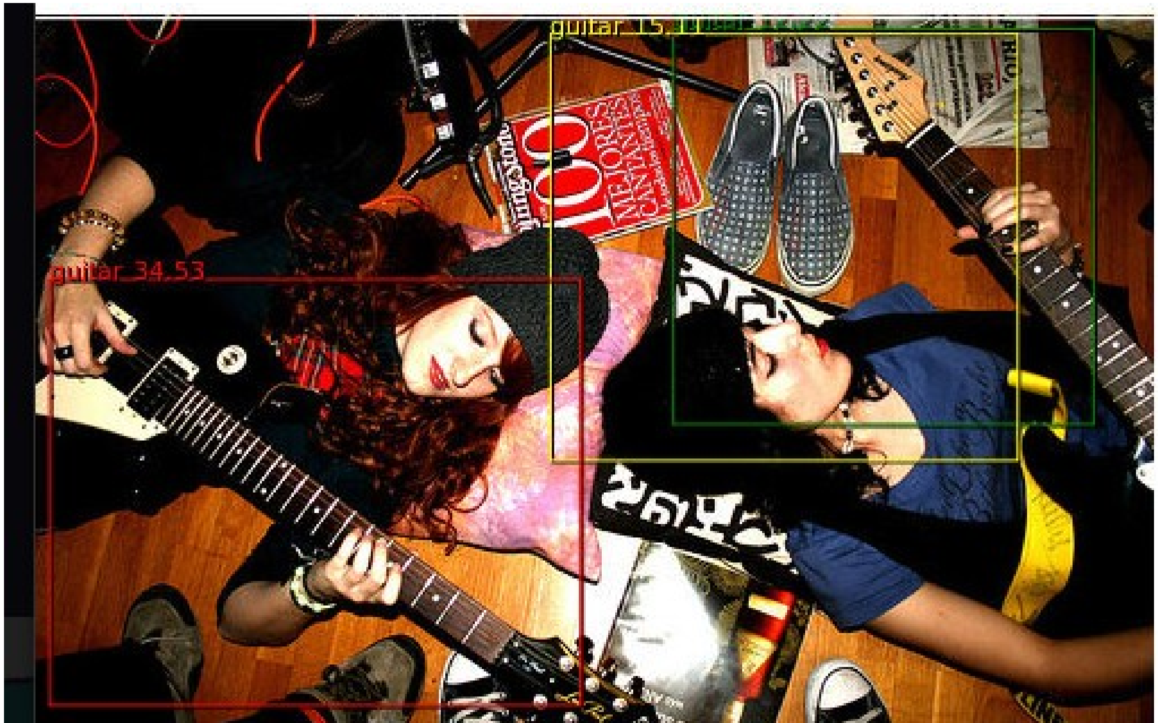
Classification + Localization: sliding window + bounding box regression

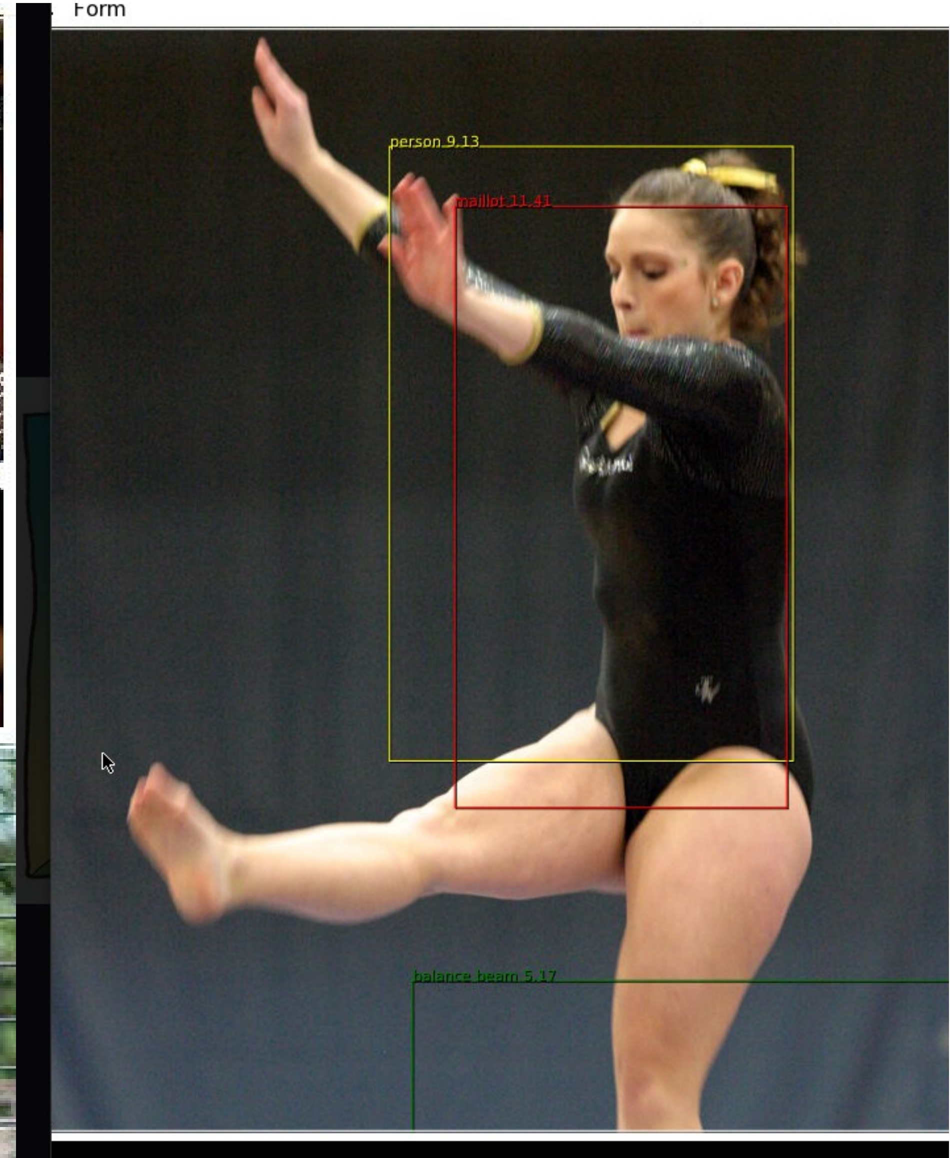
Y LeCun

- **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.

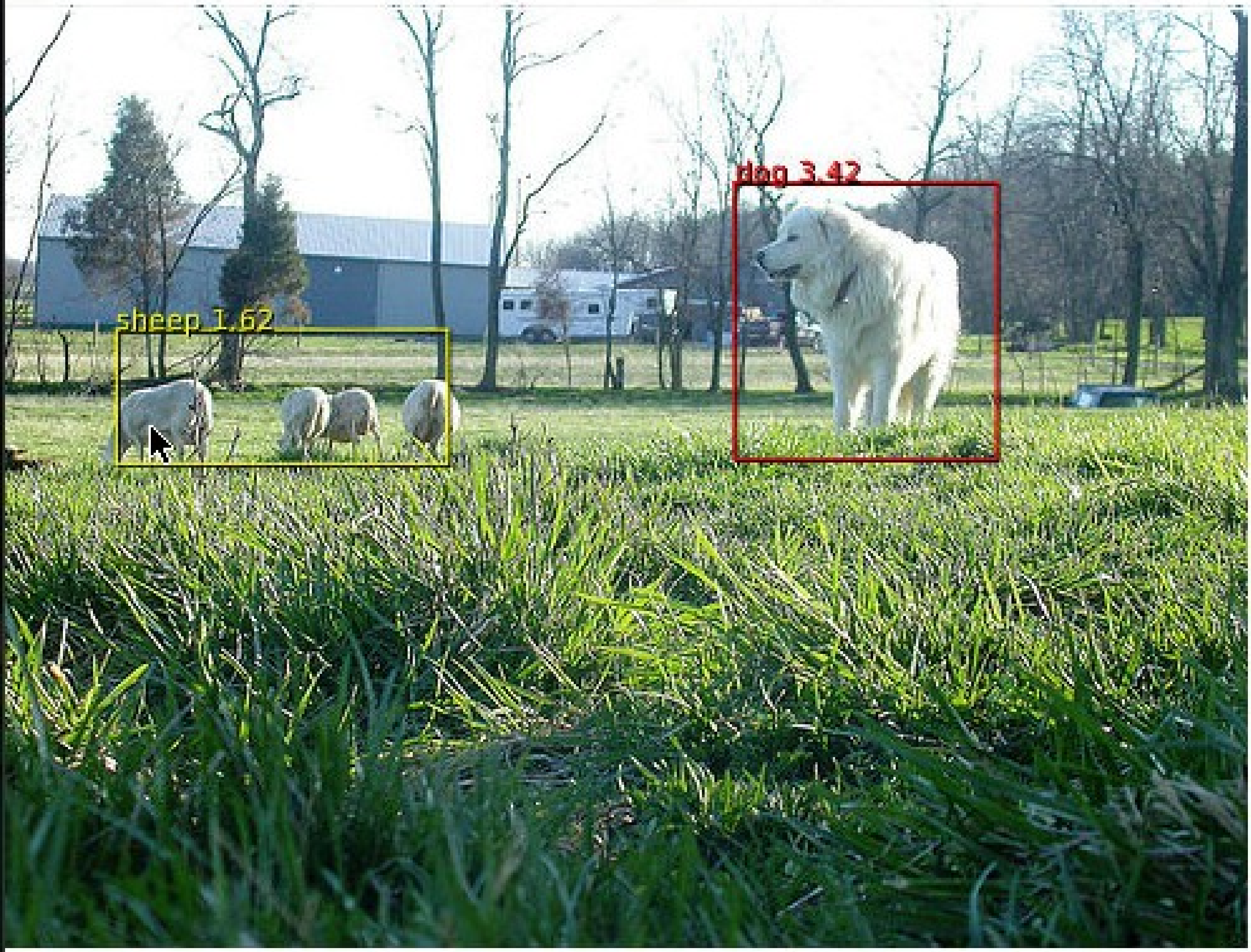


Results: pre-trained on ImageNet1K, fine-tuned on ImageNet Detection



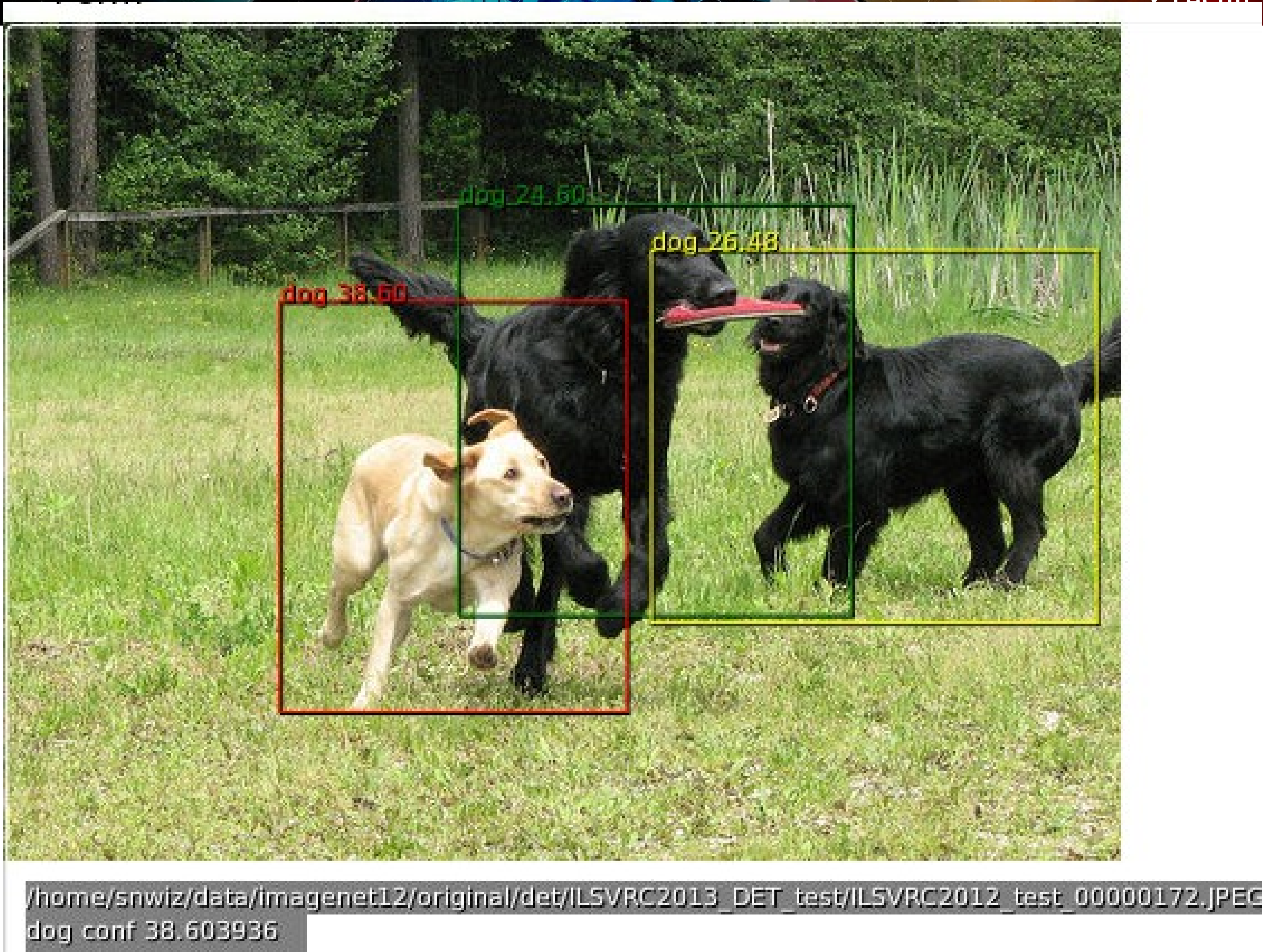


Detection Examples



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

Detection Examples



Detection Examples

Person Detection and Pose Estimation

Tompson, Goroshin, Jain, LeCun, Bregler arXiv:1411.4280 (2014)

Image captioning: generating a descriptive sentence

- [Lebret, Pinheiro, Collobert 2015][Kulkarni 11][Mitchell 12][Vinyals 14][Mao 14][Karpathy 14][Donahue 14]...

A man riding skis on a snow covered ski slope.

NP: a man, skis, the snow, a person, a woman, a snow covered slope, a slope, a snowboard, a skier, man.
VP: wearing, riding, holding, standing on, skiing down.
PP: on, in, of, with, down.

A man wearing skis on the snow.

A man is doing skateboard tricks on a ramp.

NP: a skateboard, a man, a trick, his skateboard, the air, a skateboarder, a ramp, a skate board, a person, a woman.
VP: doing, riding, is doing, performing, flying through.
PP: on, of, in, at, with.

A man riding a skateboard on a ramp.

The girl with blue hair stands under the umbrella.

NP: a woman, an umbrella, a man, a person, a girl, umbrellas, that, a little girl, a cell phone.
VP: holding, wearing, is holding, holds, carrying.
PP: with, on, of, in, under.

A woman is holding an umbrella.

A slice of pizza sitting on top of a white plate.

NP: a plate, a white plate, a table, pizza, it, a pizza, food, a sandwich, top, a close.
VP: topped with, has, is, sitting on, is on.
PP: of, on, with, in, up.

A table with a plate of pizza on a white plate.

A baseball player swinging a bat on a field.

NP: the ball, a game, a baseball player, a man, a tennis court, a ball, home plate, a baseball game, a batter, a field.
VP: swinging, to hit, playing, holding, is swinging.
PP: on, during, in, at, of.

A baseball player swinging a bat on a baseball field.

A bunch of kites flying in the sky on the beach.

NP: the beach, a beach, a kite, kites, the ocean, the water, the sky, people, a sandy beach, a group.
VP: flying, flies, is flying, flying in, are.
PP: on, of, with, in, at.

People flying kites on the beach.

C3D: Video Classification with 3D ConvNet

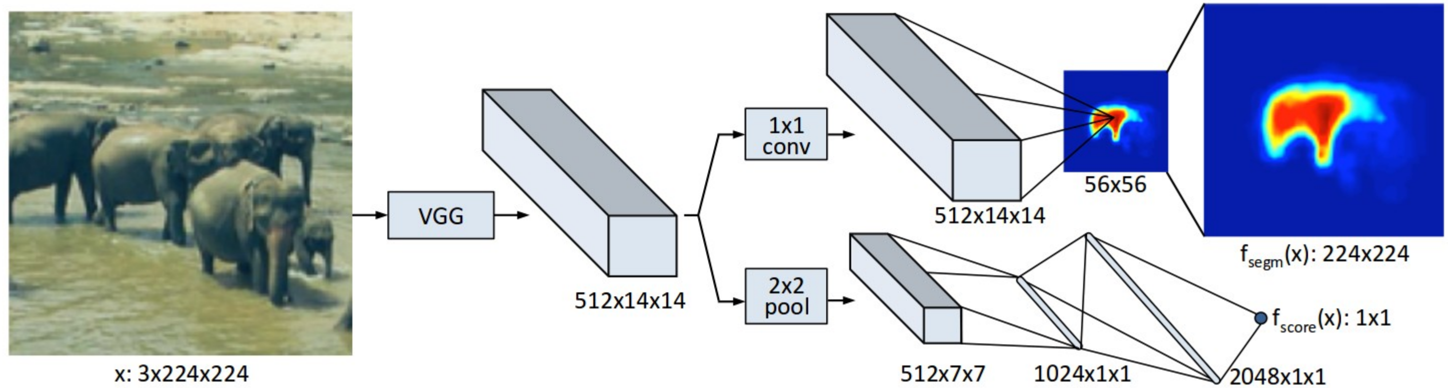
- [Tran et al. 2015]

VIDEO: COMMON SPORTS

VIDEO: UNCOMMON SPORTS

Segmenting and Localizing Objects (DeepMask)

- [Pinheiro, Collobert, Dollar ICCV 2015]
  - ConvNet produces object masks

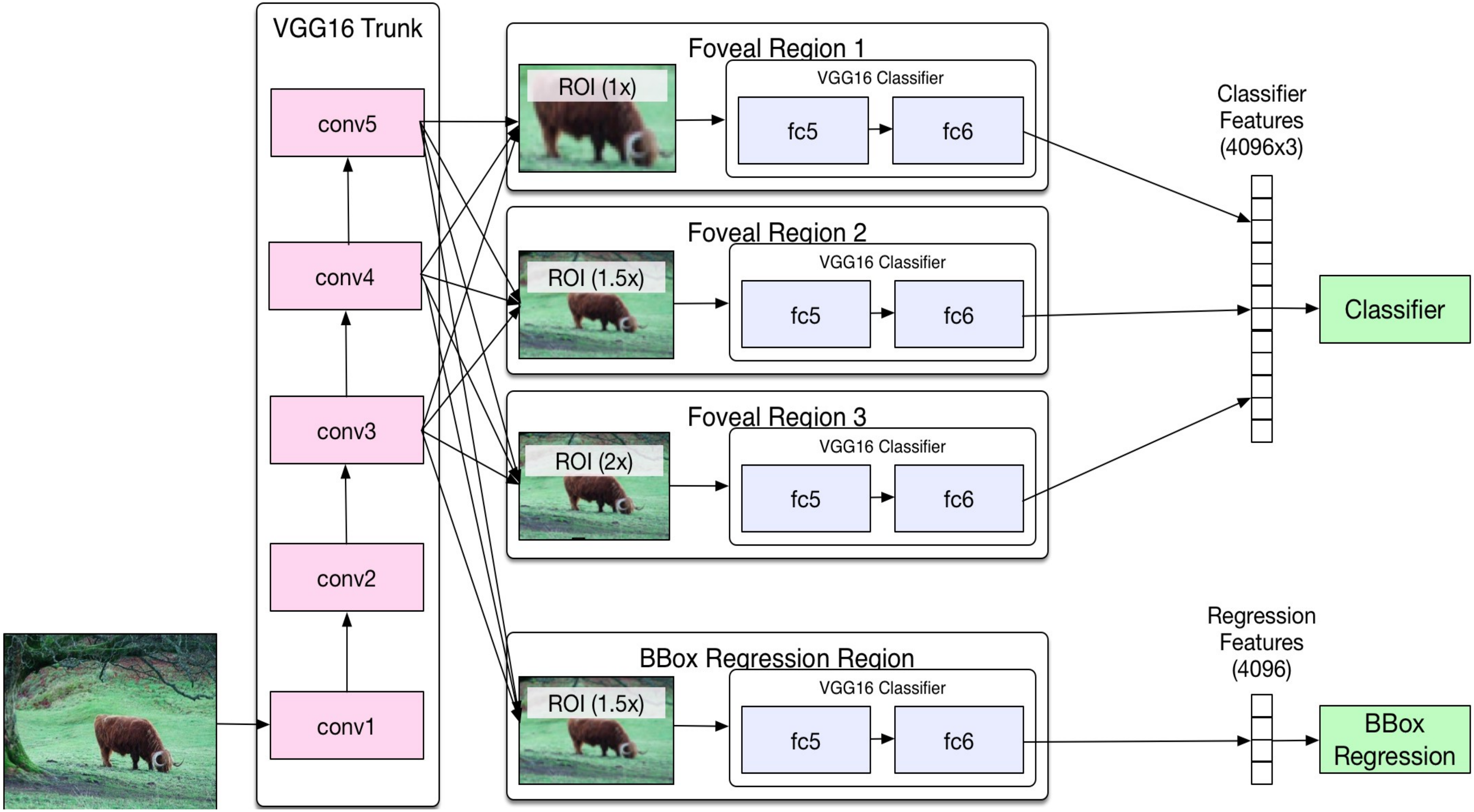


DeepMask++ Proposals

- FAIR COCO Team

Recognition Pipeline

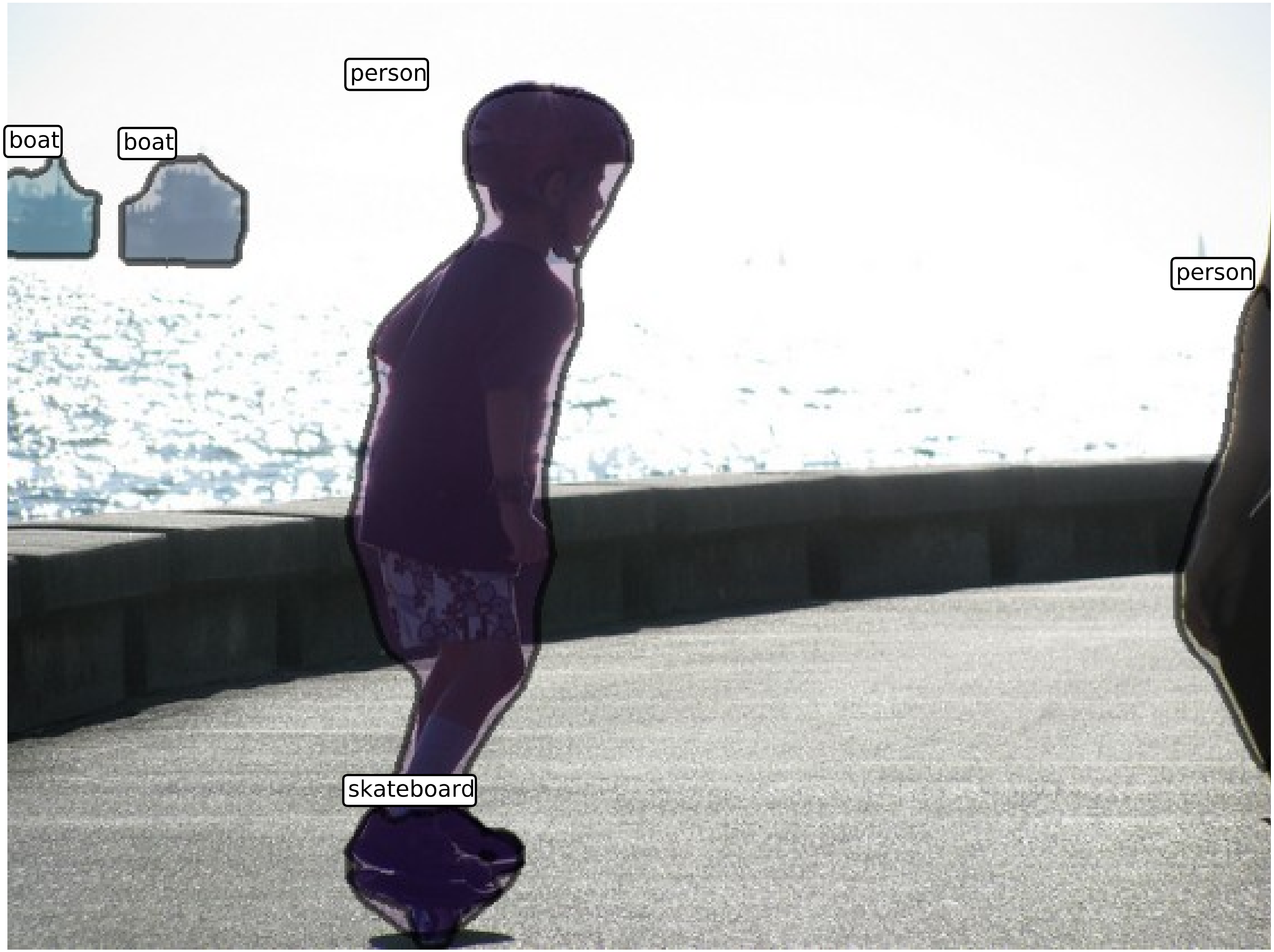
FAIR COCO Team



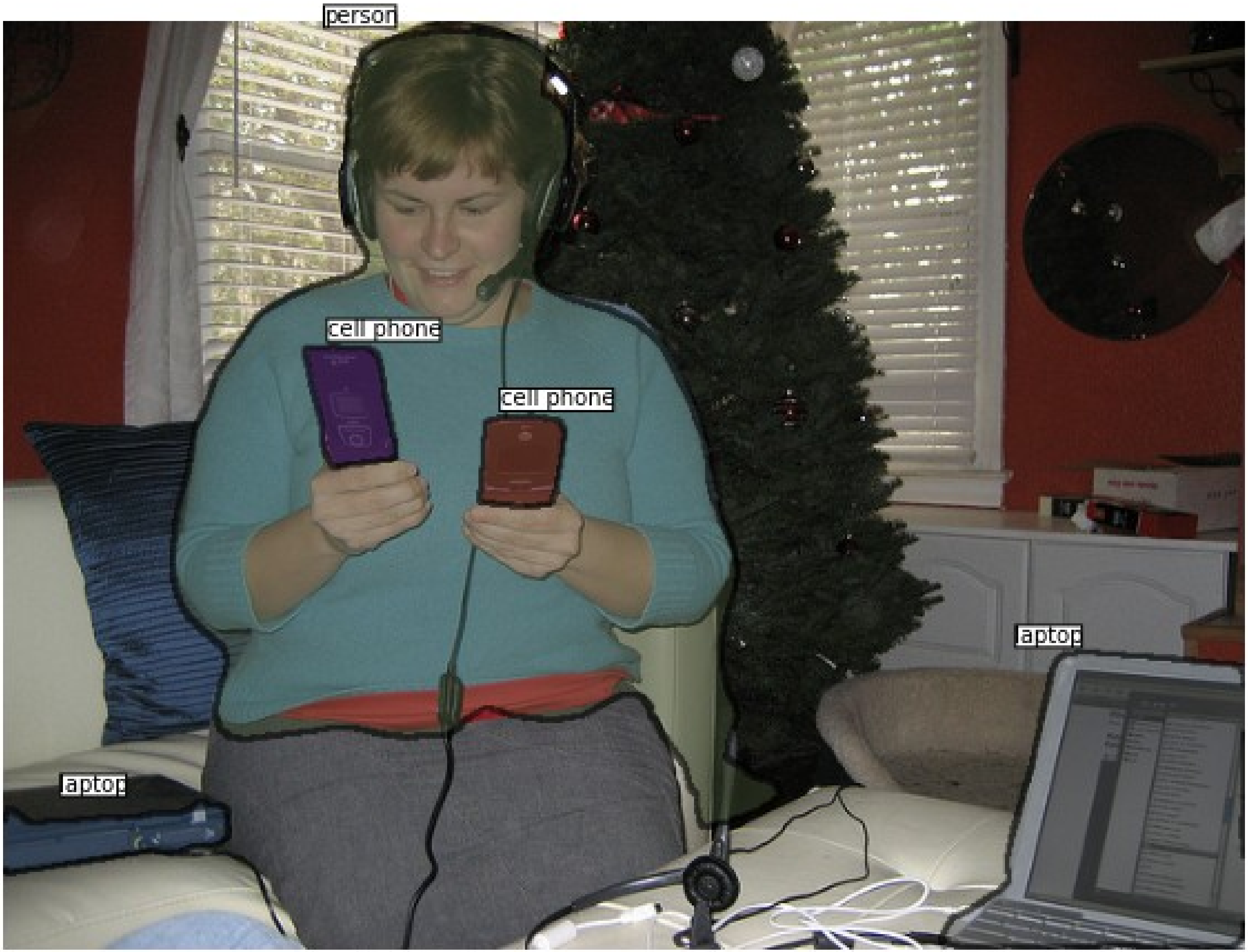
- 2.5 days on 8x4 Kepler GPUs with Elastic Avergaing Stochastic Gradient Descent (EASGD [Zhang, Choromanska, LeCun NIPS 2015]

- "Big Sur"

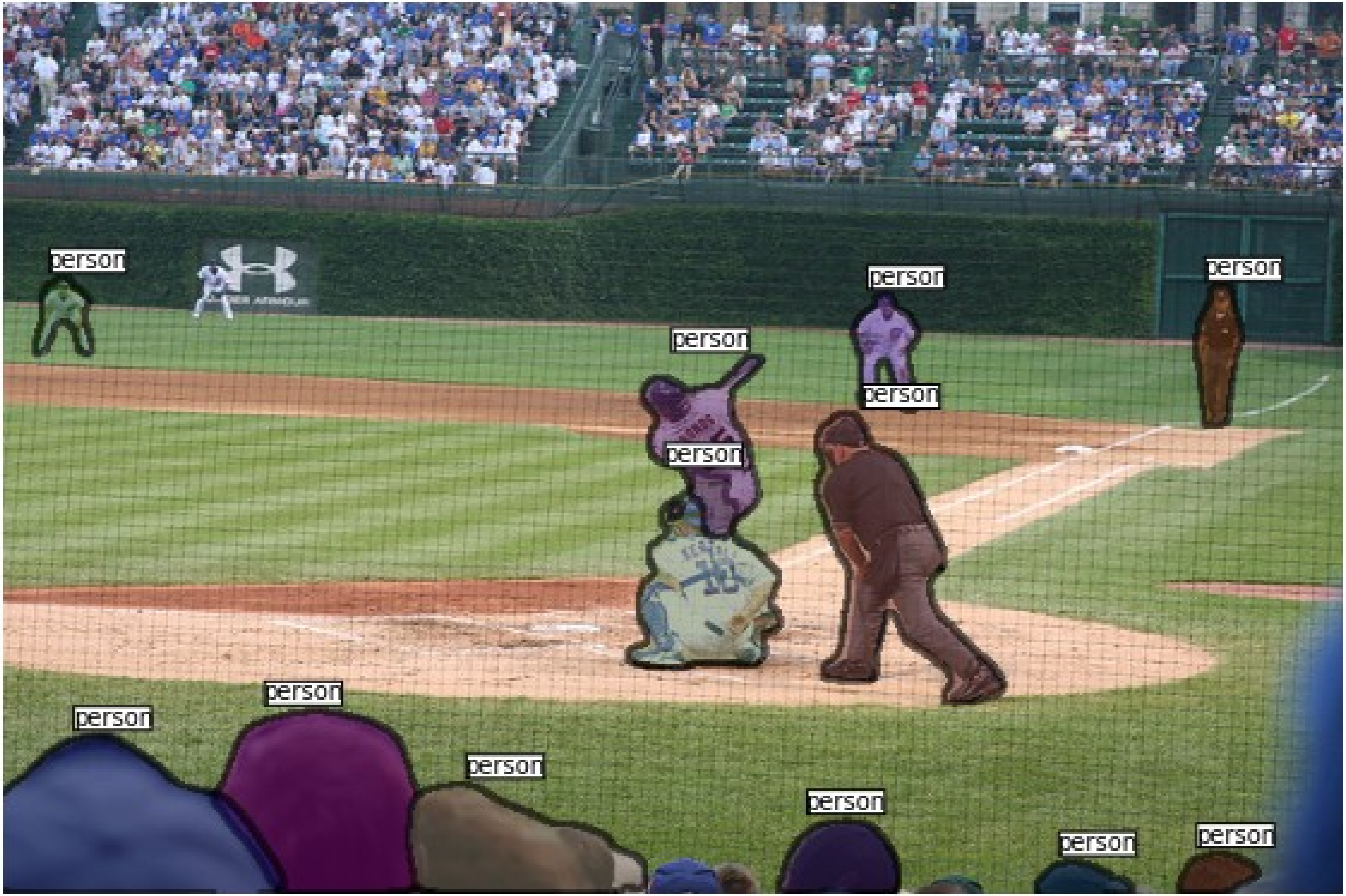
Results



Results



Results

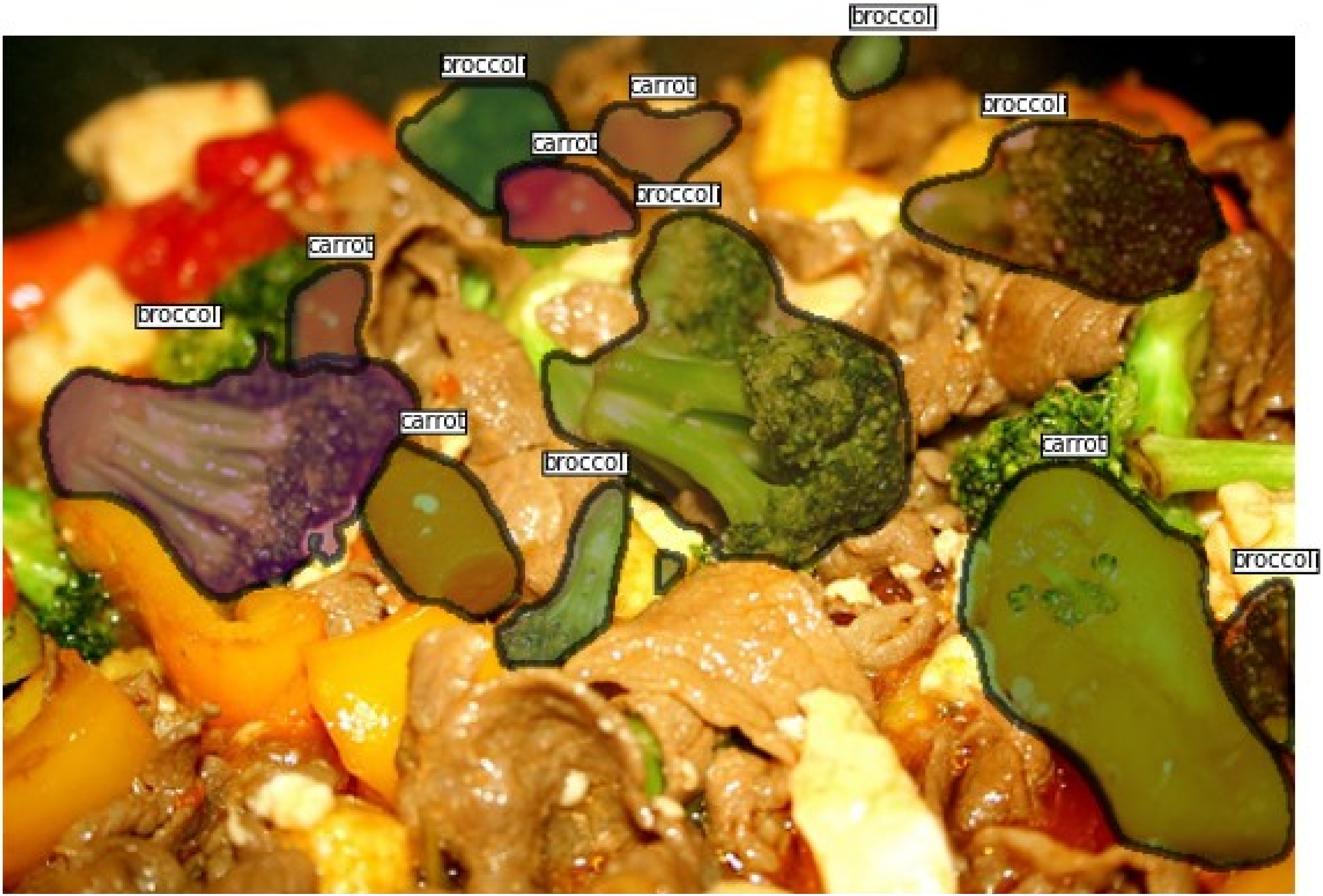

person
person
person
person
person
person
person
person
person
person
person
person
person

Results

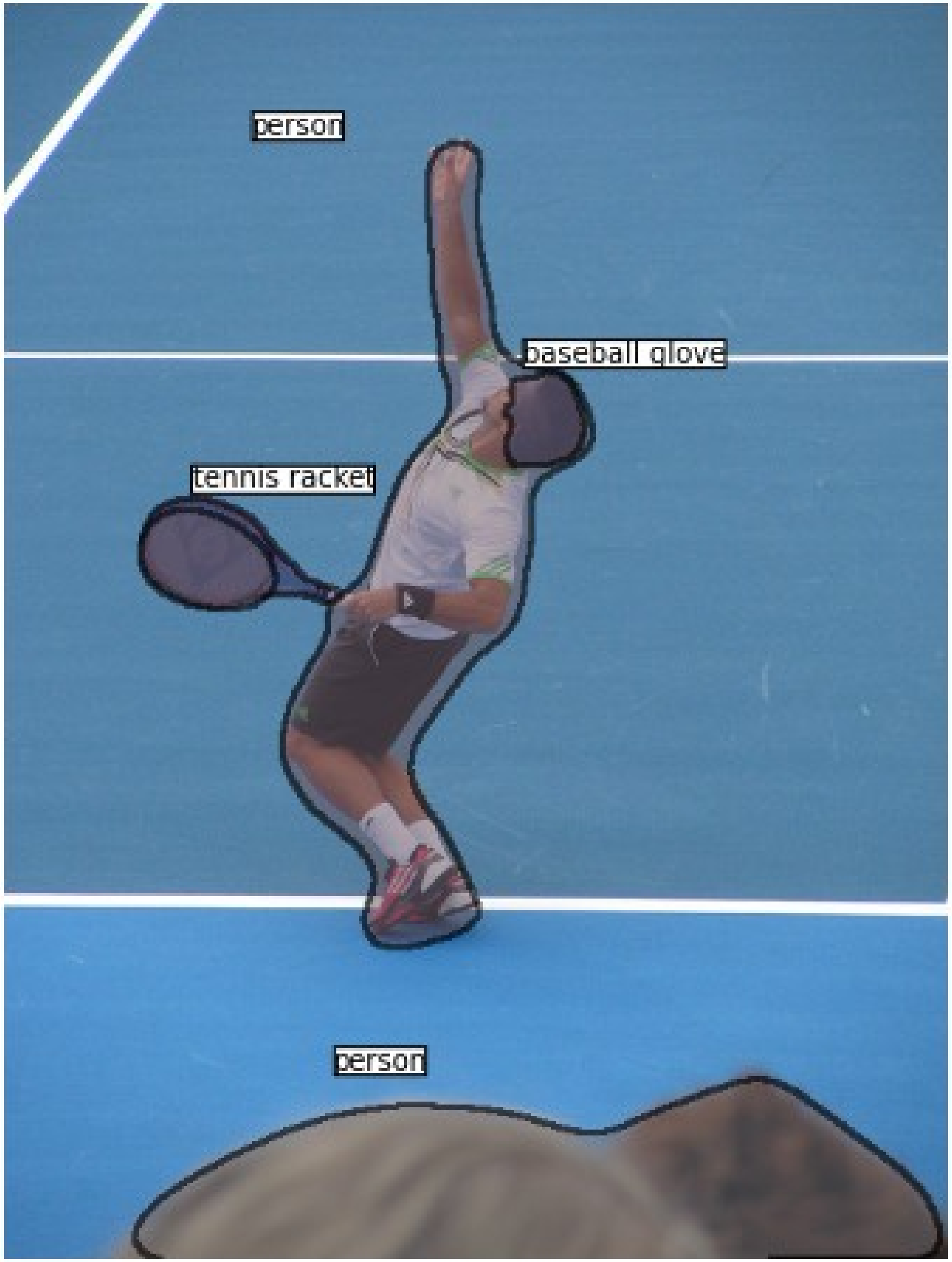
Results

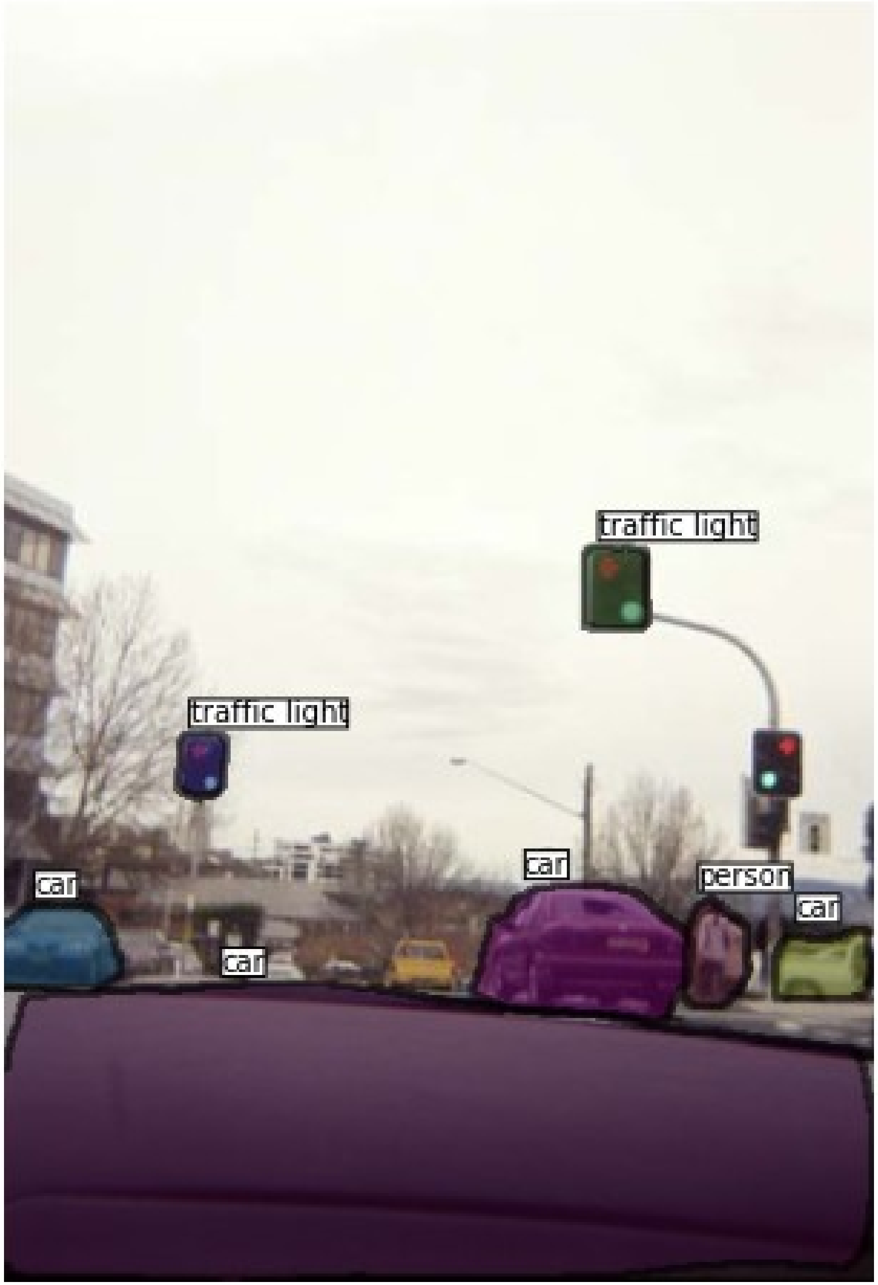
Results

Y LeCun



Mistakes


person
baseball glove
tennis racket
person


traffic light
traffic light
car
car
car
person
car

Results

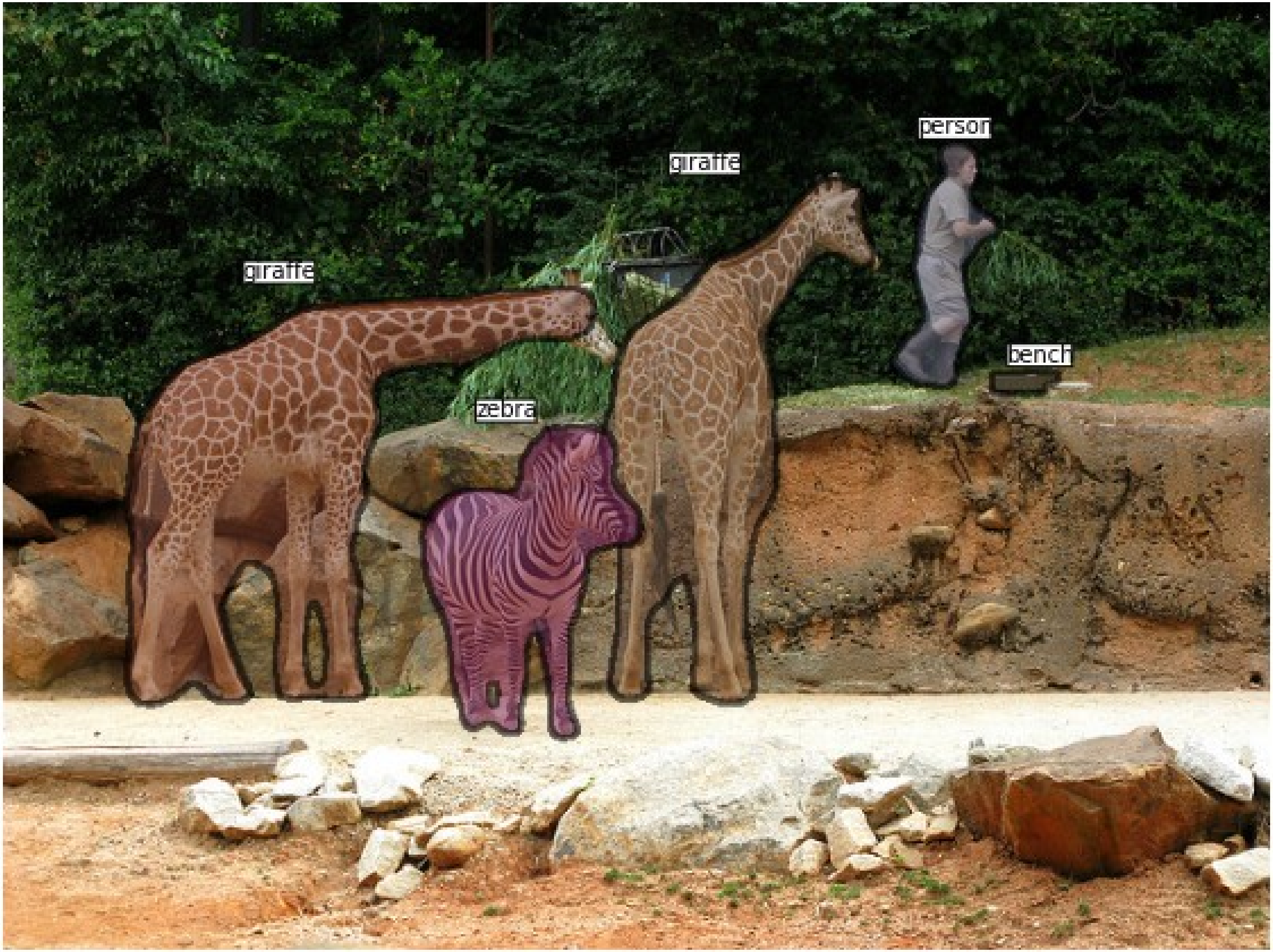
Results

Y LeCun

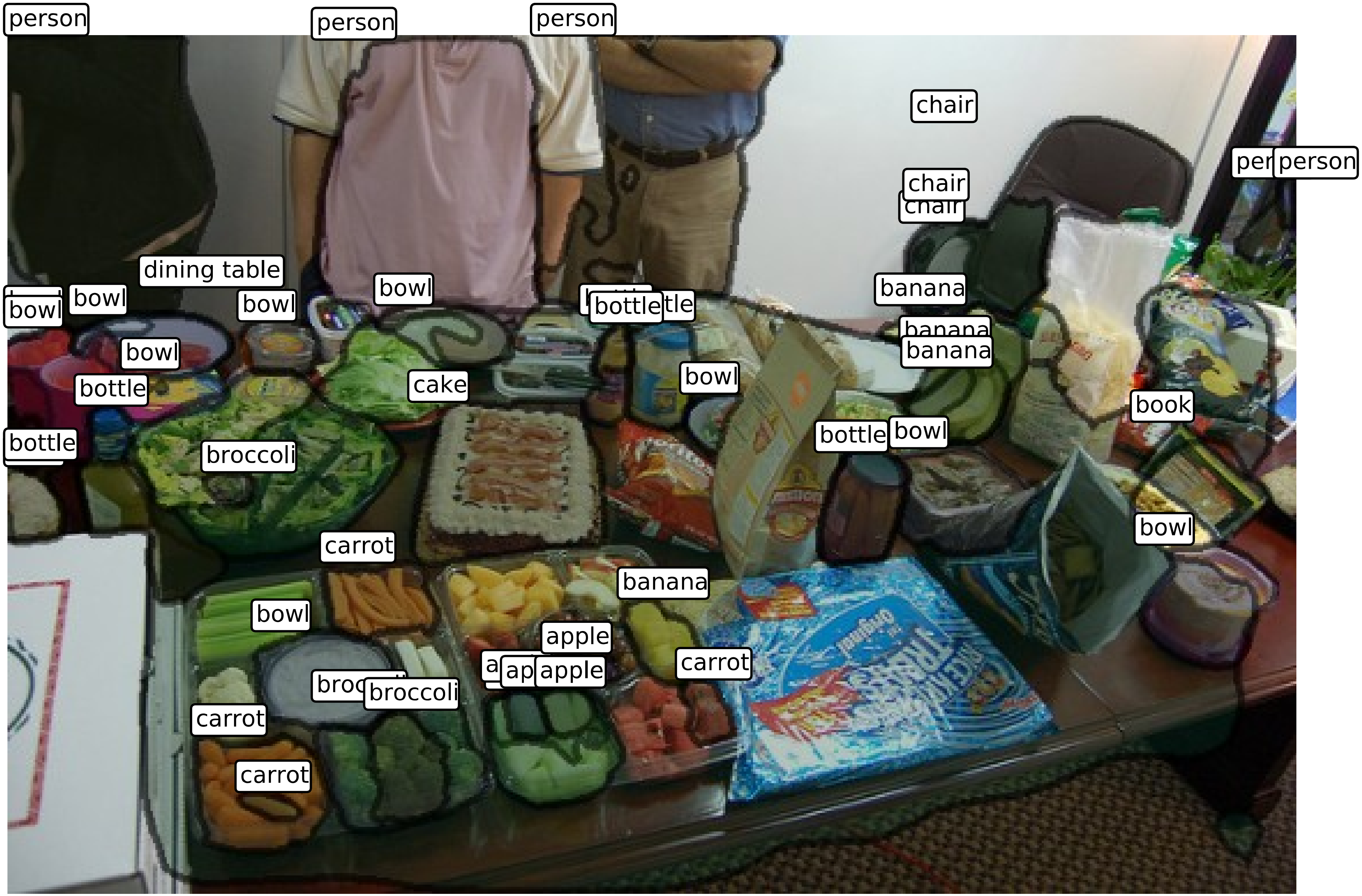
Results


donut
sandwich
donut
donut
donut
donut
donut
donut
donut
donut
donut
donut
donut
donut
donut
donut
donut
donut
donut
donut
donut
donut
donut
donut
donut
donut
donut

Results

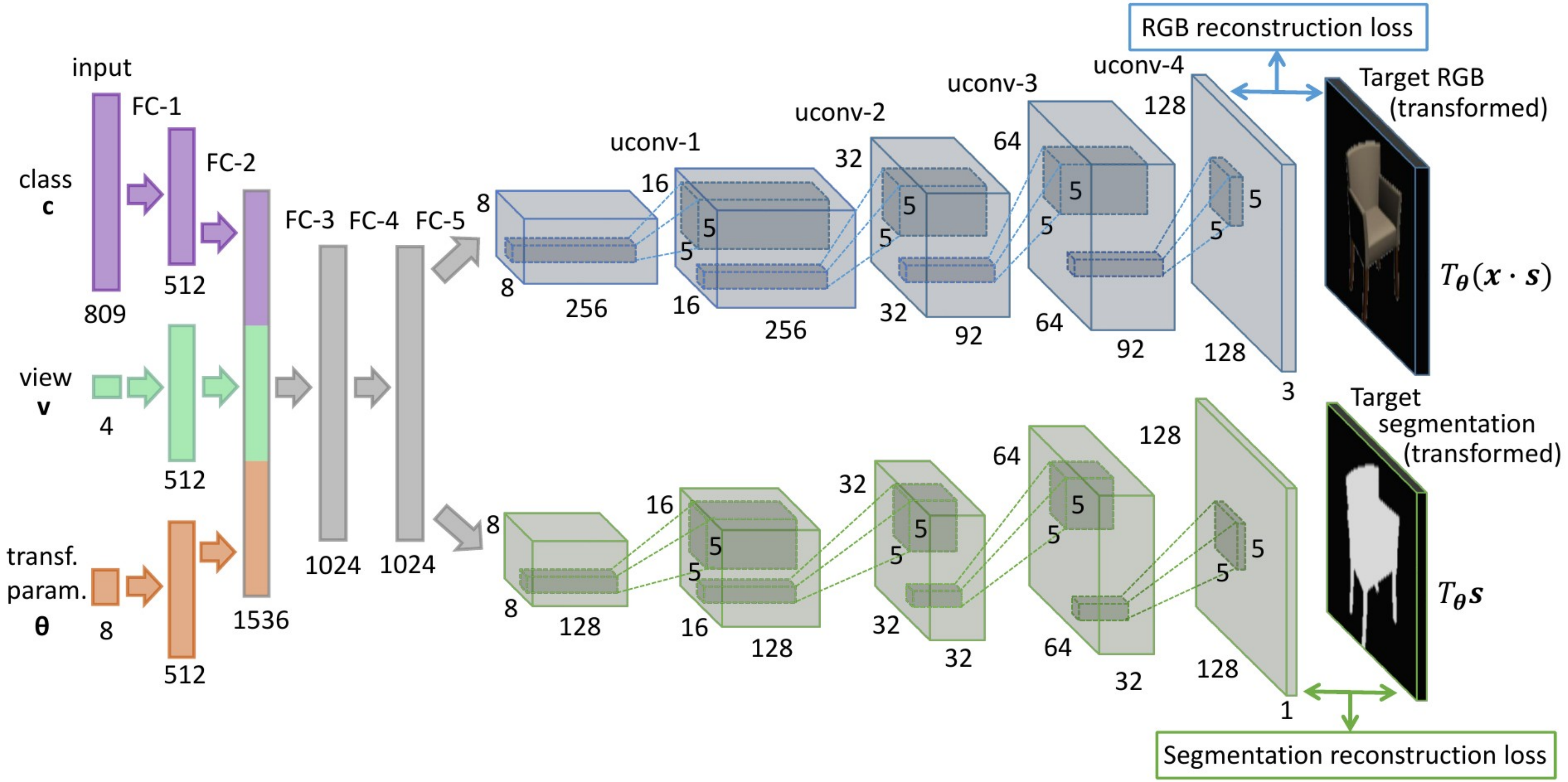


Results


person
person
person
chair
chair
per
person
dining table
bowl
bowl
bowl
bowl
bottle
bottle
banana
banana
bowl
bowl
cake
bowl
bottle
bottle
book
bowl
bottle
broccoli
bowl
carrot
banana
bowl
apple
apple
carrot
broccoli
carrot
carrot

Supervised ConvNets that Draw Pictures

- Using ConvNets to Produce Images
- [Dosovitskyi et al. Arxiv:1411:5928

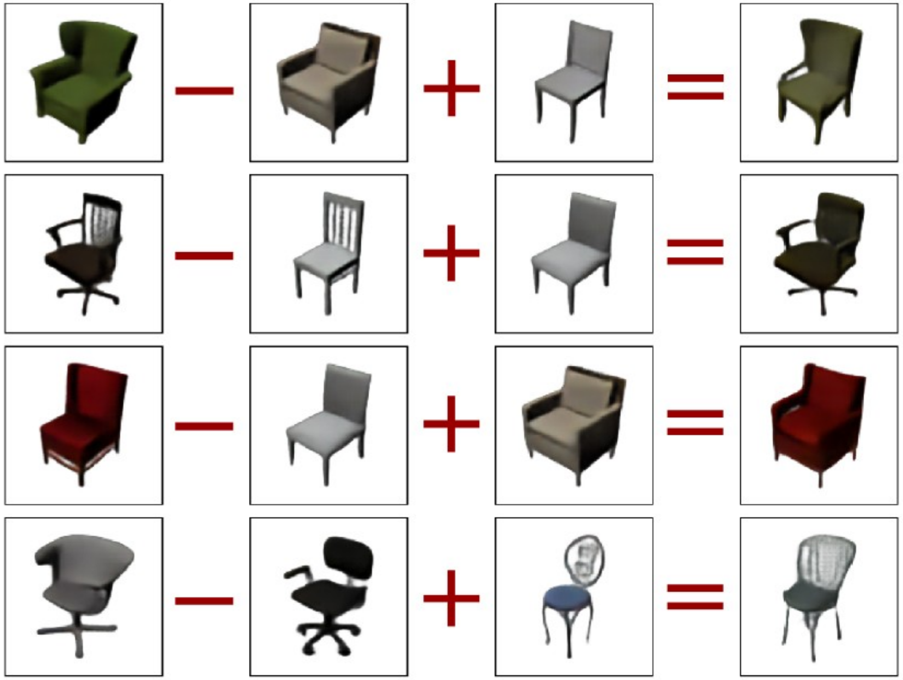


Supervised ConvNets that Draw Pictures

Y LeCun

- Generating Chairs

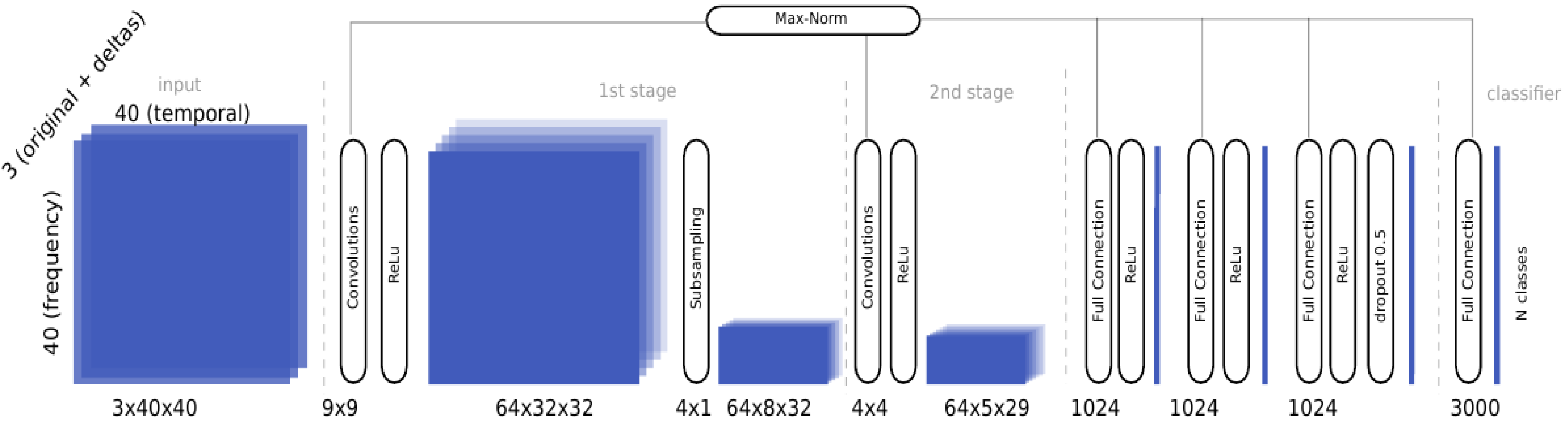
- Chair Arithmetic in Feature Space



Speech Recognition With ConvNets

Speech Recognition with Convolutional Nets (NYU/IBM)

Y LeCun

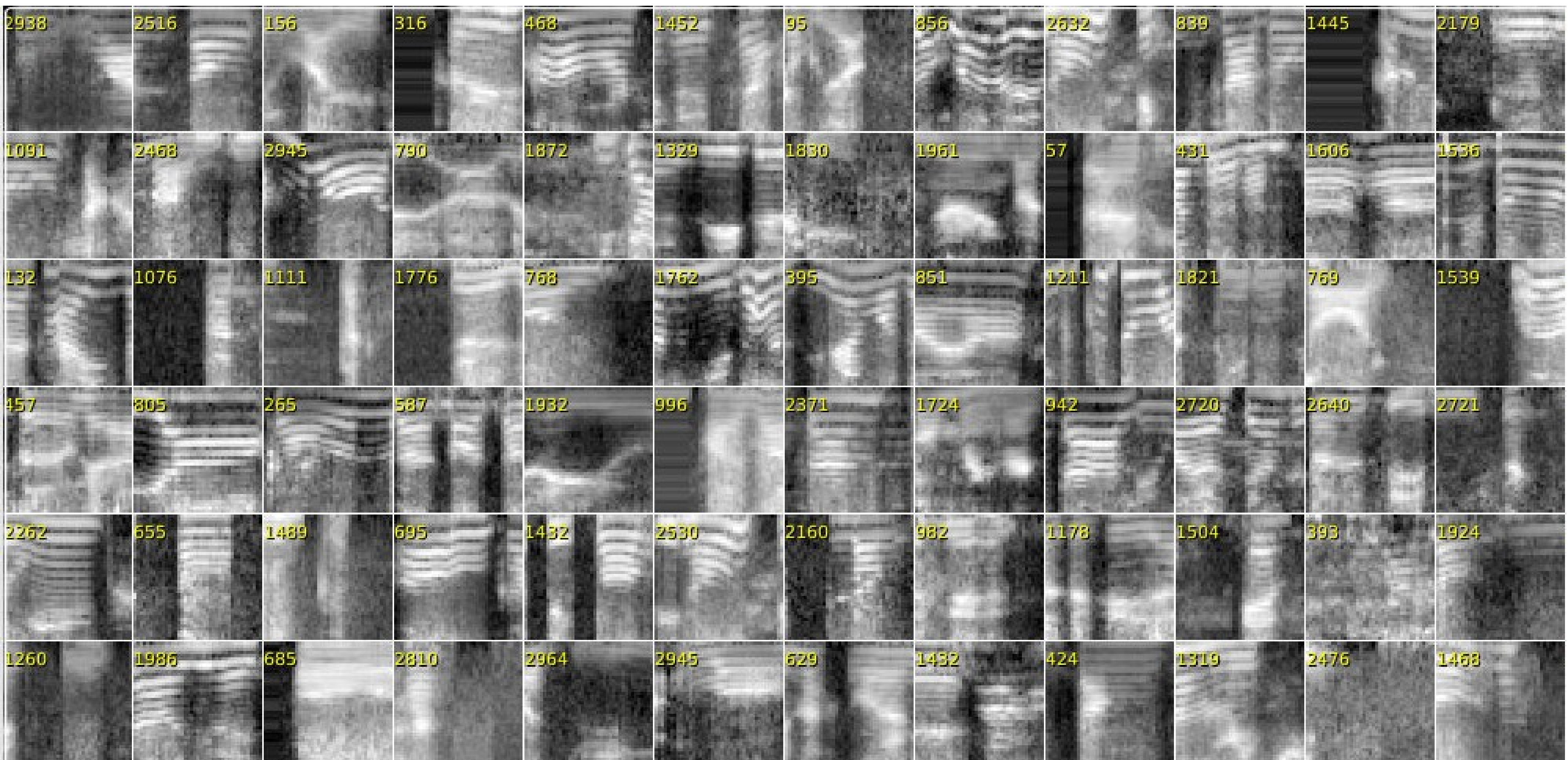


- Acoustic Model: ConvNet with 7 layers. 54.4 million parameters.
- Classifies acoustic signal into 3000 context-dependent subphones categories
- ReLU units + dropout for last layers
- Trained on GPU. 4 days of training

Speech Recognition with Convolutional Nets (NYU/IBM)

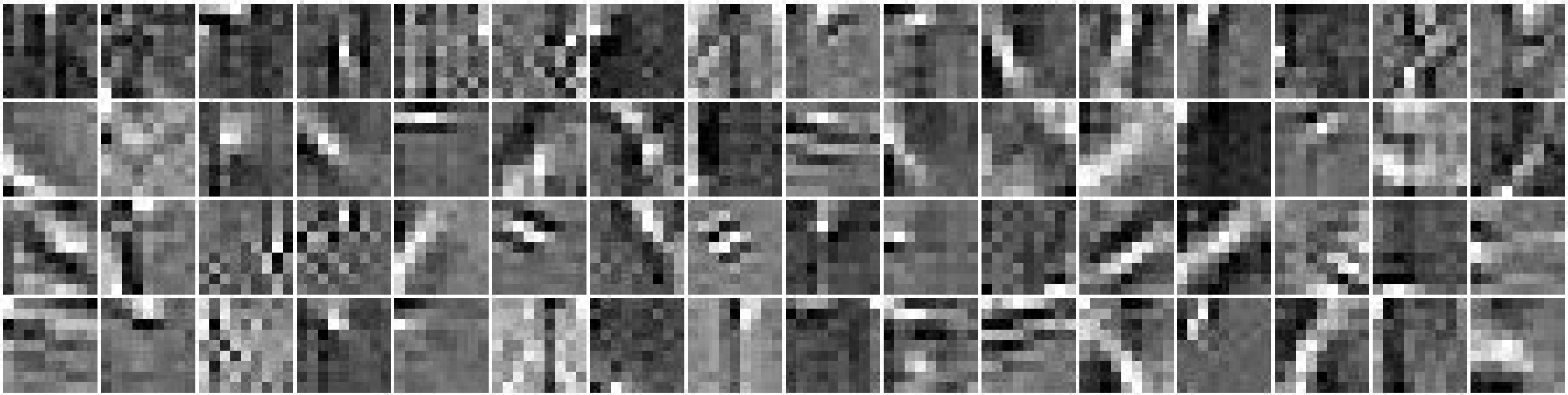
Y LeCun

- **Training samples.**
  - 40 MEL-frequency Cepstral Coefficients
  - Window: 40 frames, 10ms each



Speech Recognition with Convolutional Nets (NYU/IBM)

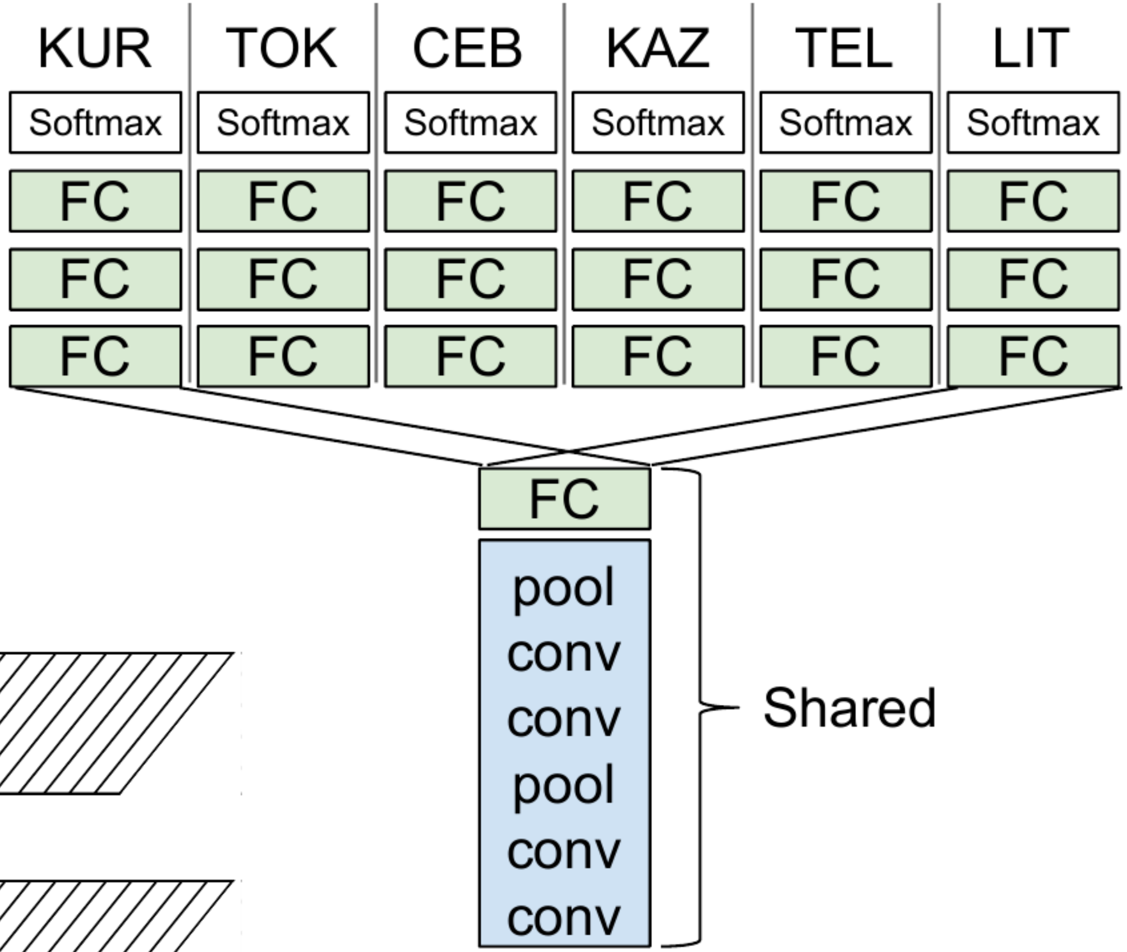
- **Convolution Kernels at Layer 1:**
  - 64 kernels of size 9x9

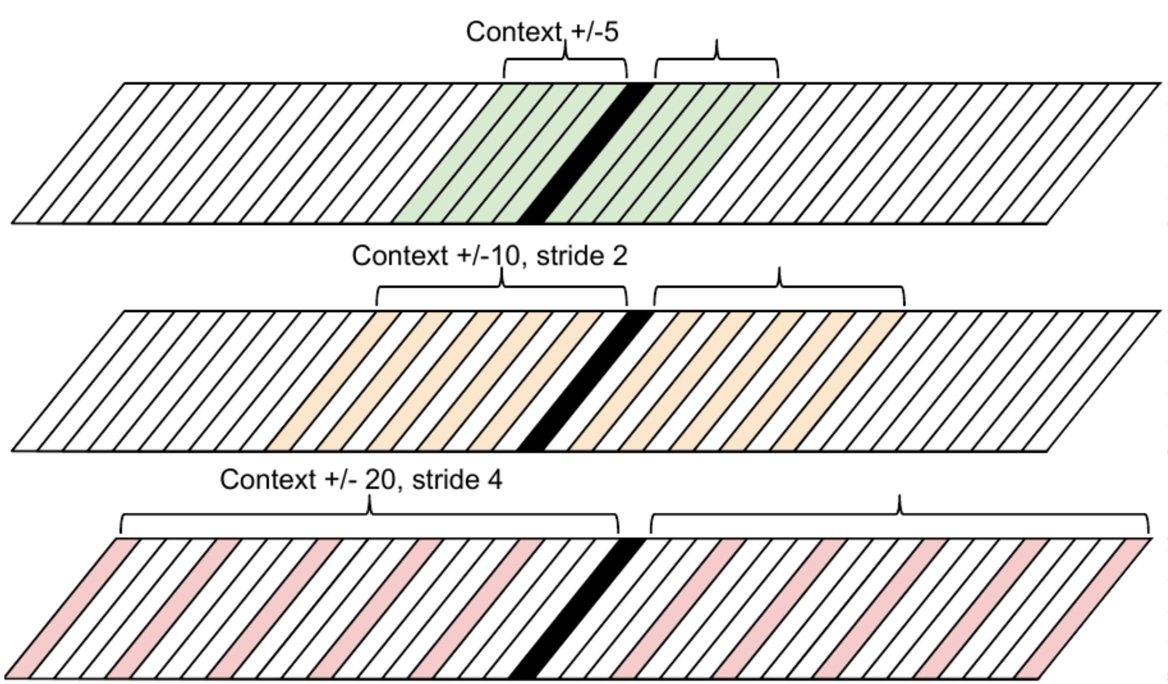


Speech Recognition with Convolutional Nets (NYU/IBM)

Y LeCun

- **Multilingual recognizer**
- **Multiscale input**
  - Large context window





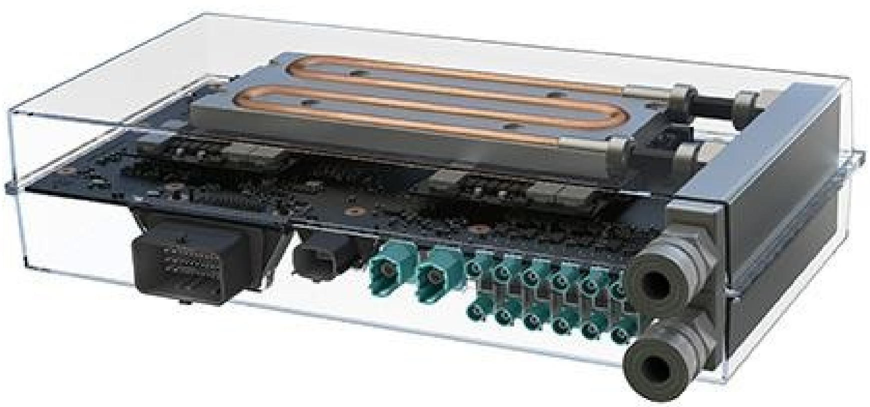
ConvNets are Everywhere (or soon will be)

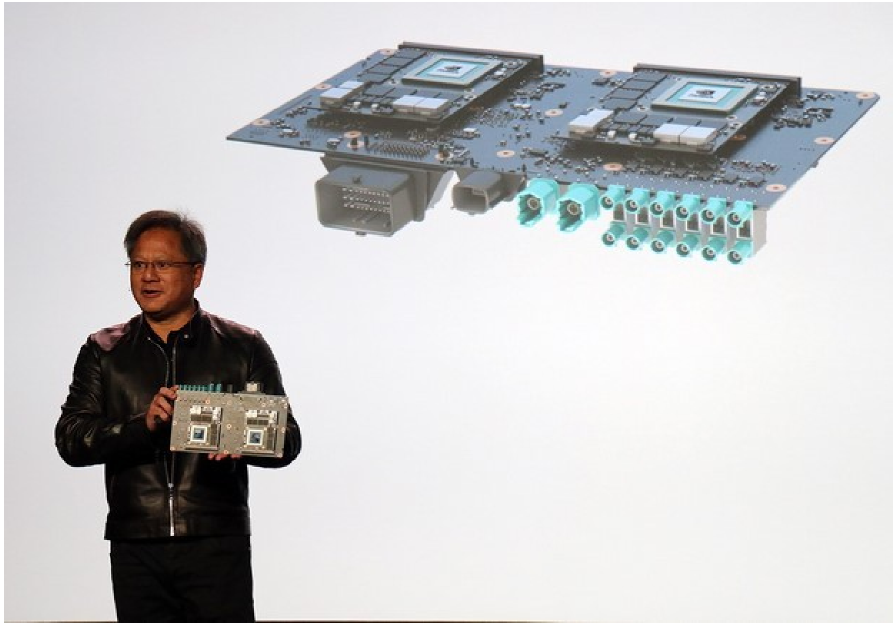
NVIDIA: ConvNet-Based Driver Assistance

Drive-PX2: Open Platform for Driver Assistance

Embedded Super-Computer: 42 TOPS

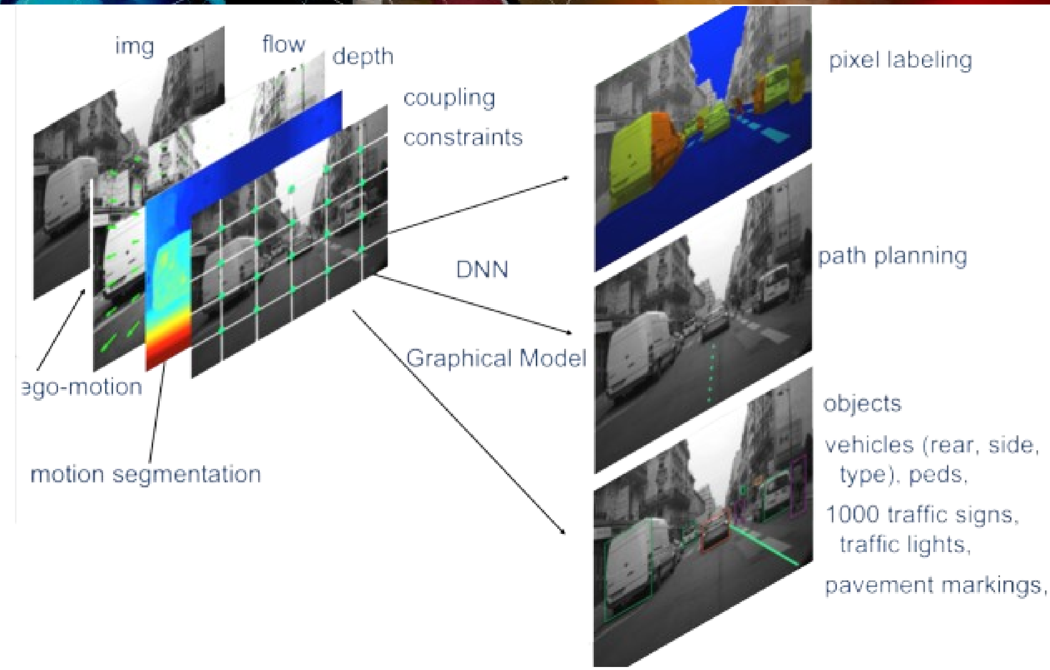
– (=150 Macbook Pros)

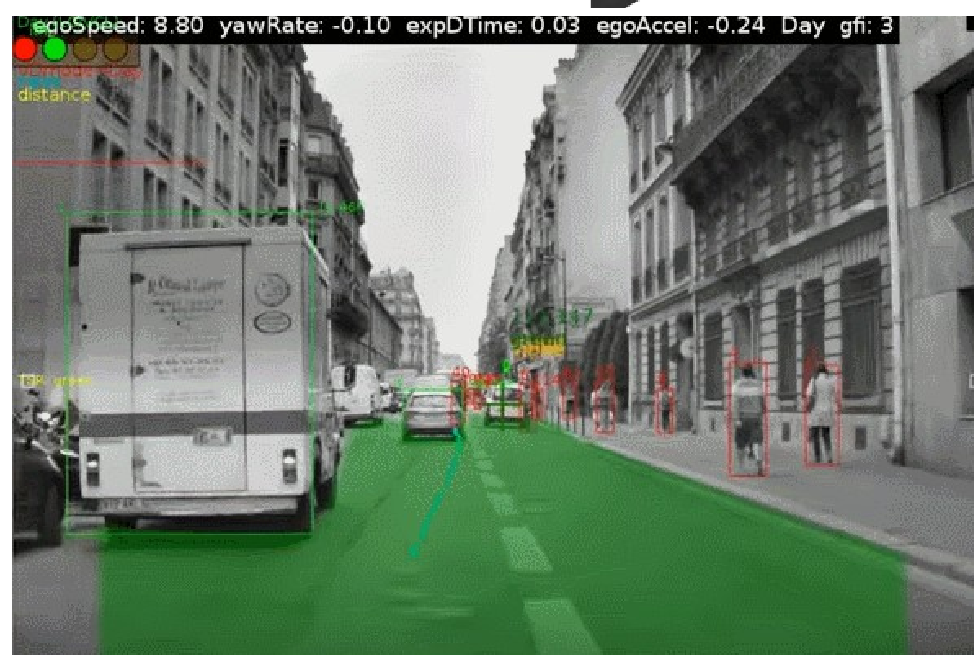




MobilEye: ConvNet-Based Driver Assistance

Deployed in the latest
Tesla Model S and Model X





ConvNet in Connectomics [Jain, Turaga, Seung 2007]

- 3D ConvNet

  Volumetric

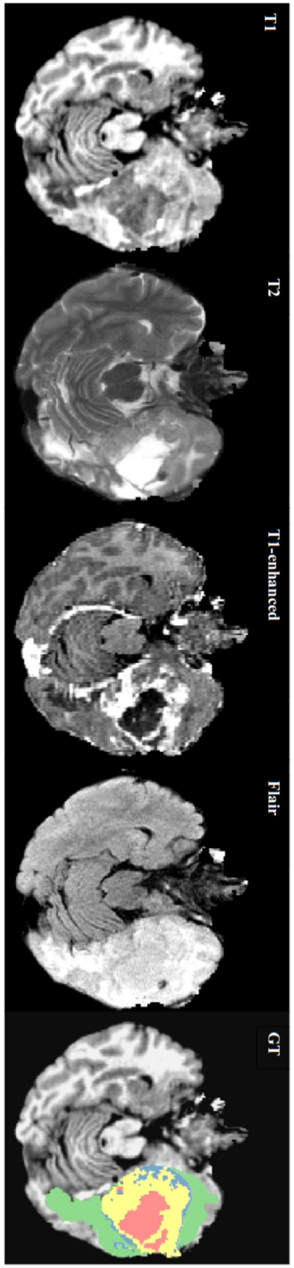
  Images

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

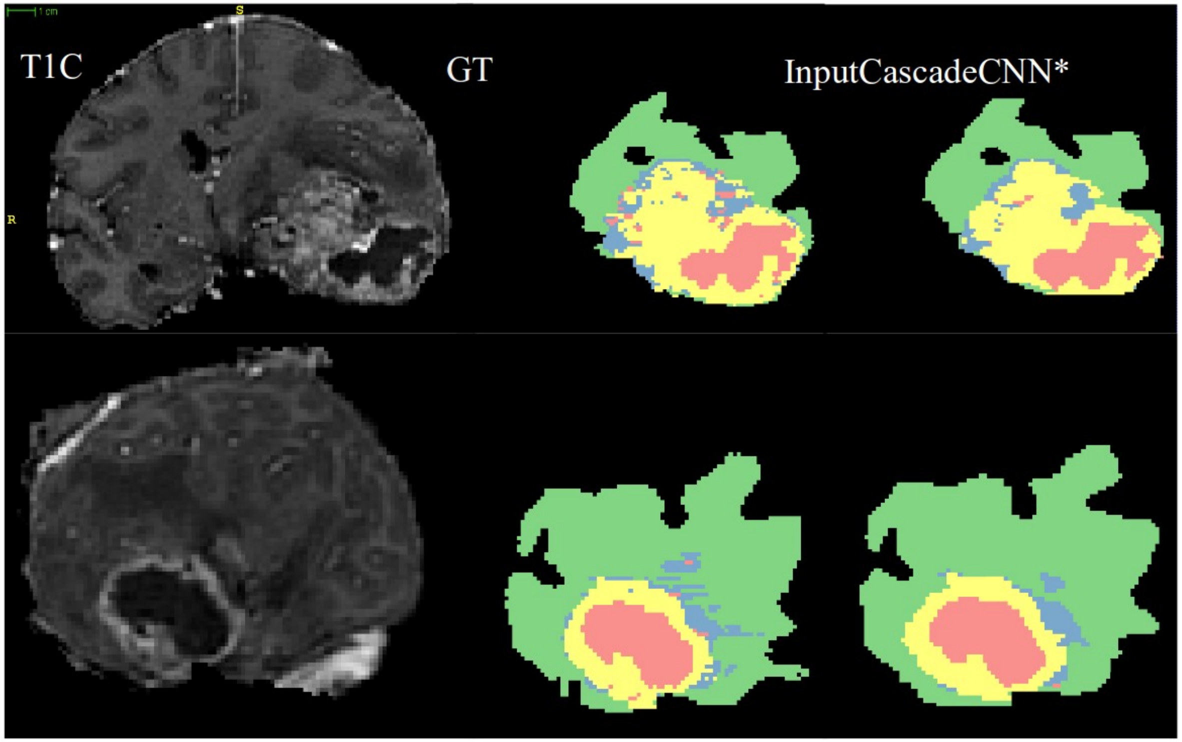
- Has become a standard method in connectomics

VIDEO

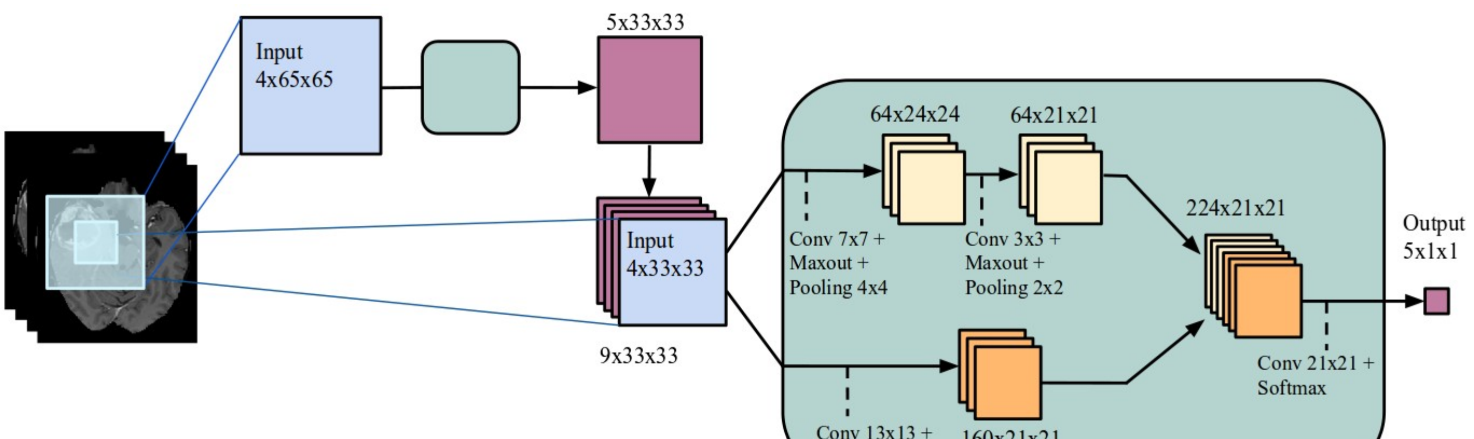
Brain Tumor Detection



- [Havaei et al. 2015]
  - Arxiv:1505.03540
- InputCascadeCNN architecture
  - 802,368 parameters
- Trained on 30 patients.
- State of the art results on BRAT2013



edema, enhanced tumor, necrosis, non-enhanced tumor.



Predicting DNA/RNA – Protein Binding with ConvNets

"Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning" by B Alipanahi, A Delong, M Weirauch, B Frey,

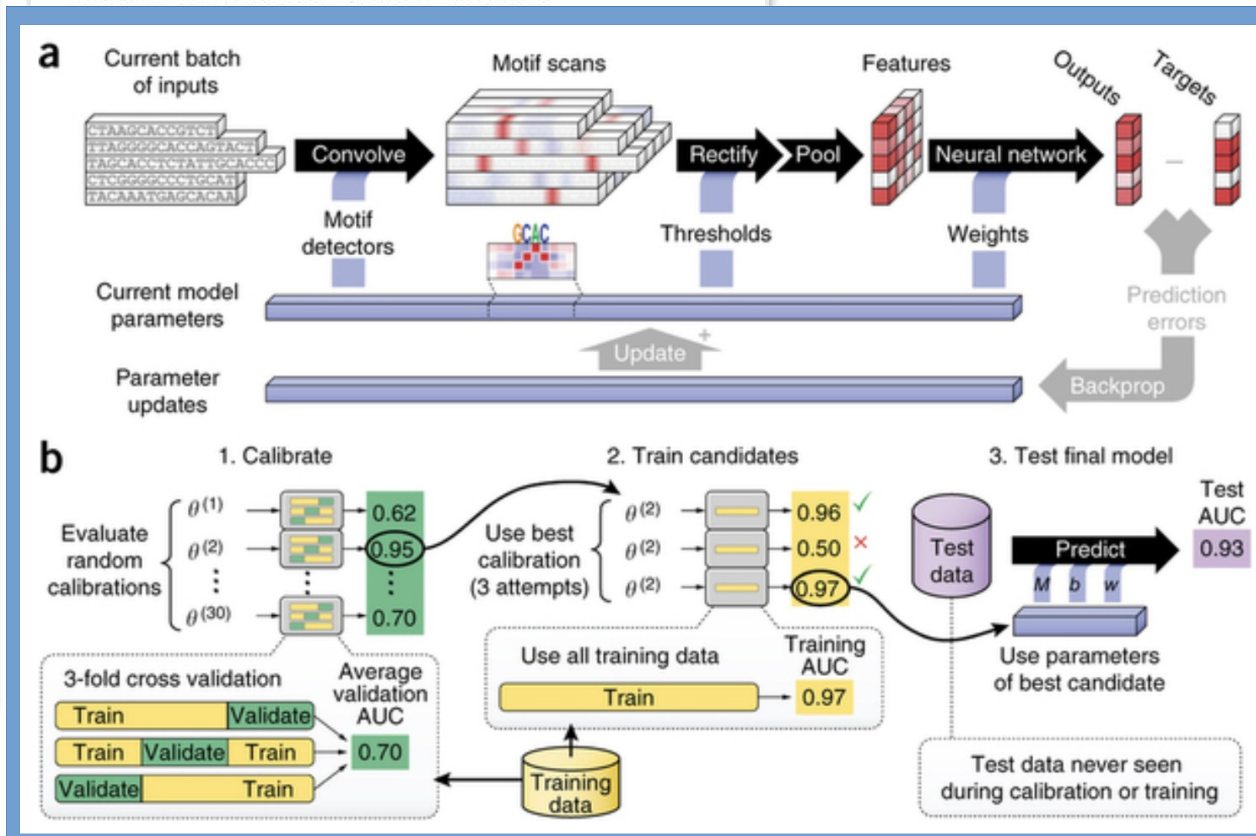
Nature Biotech, July 2015.


computational BIOLOGY

ANALYSIS

Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning

Babak Alipanahi, Andrew Delong, Matthew T Weirauch & Brendan J Frey



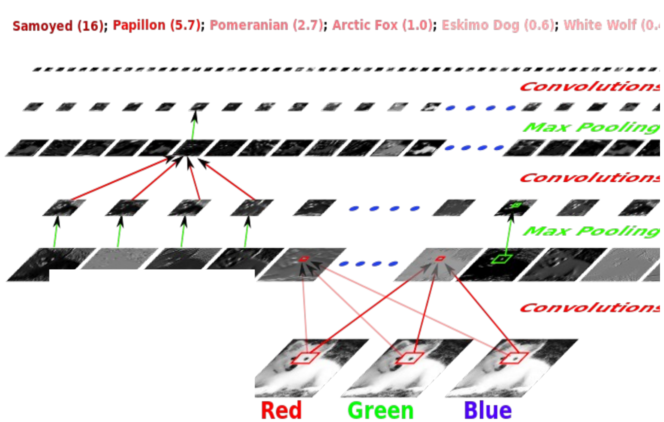
Deep Learning is Everywhere (ConvNets are Everywhere)

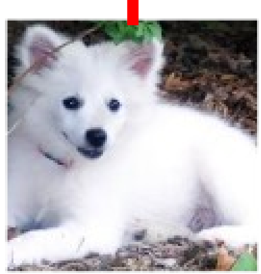
- **Lots of applications at Facebook, Google, Microsoft, Baidu, Twitter, IBM…**
  - Image recognition for photo collection search
  - Image/Video Content filtering: spam, nudity, violence.
  - Search, Newsfeed ranking
- **People upload 800 million photos on Facebook every day**
  - (2 billion photos per day if we count Instagram, Messenger and Whatsapp)
- **Each photo on Facebook goes through two ConvNets within 2 seconds**
  - One for image recognition/tagging
  - One for face recognition (not activated in Europe).
- **Soon ConvNets will really be everywhere:**
  - self-driving cars, medical imaging, augemnted reality, mobile devices, smart cameras, robots, toys…..

Embedding The World

Thought Vectors

[0.4, -1.3, 2.5, -0.7,…..]



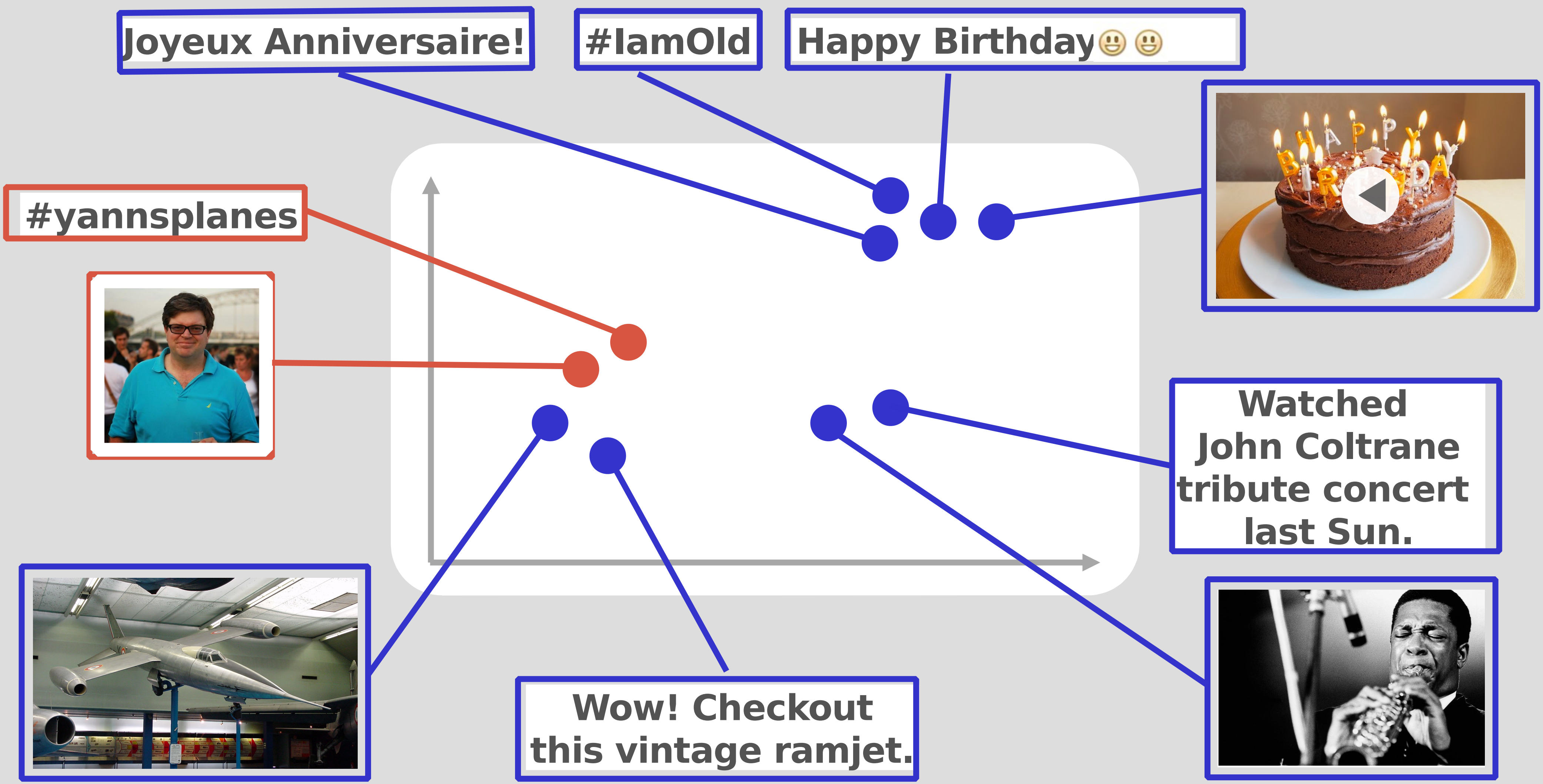


[0.2, -2.1, 0.4, -0.5,…..]

Recurrent
Neural
Net

“The neighbors' dog was a samoyed, which looks a lot like a Siberian husky”

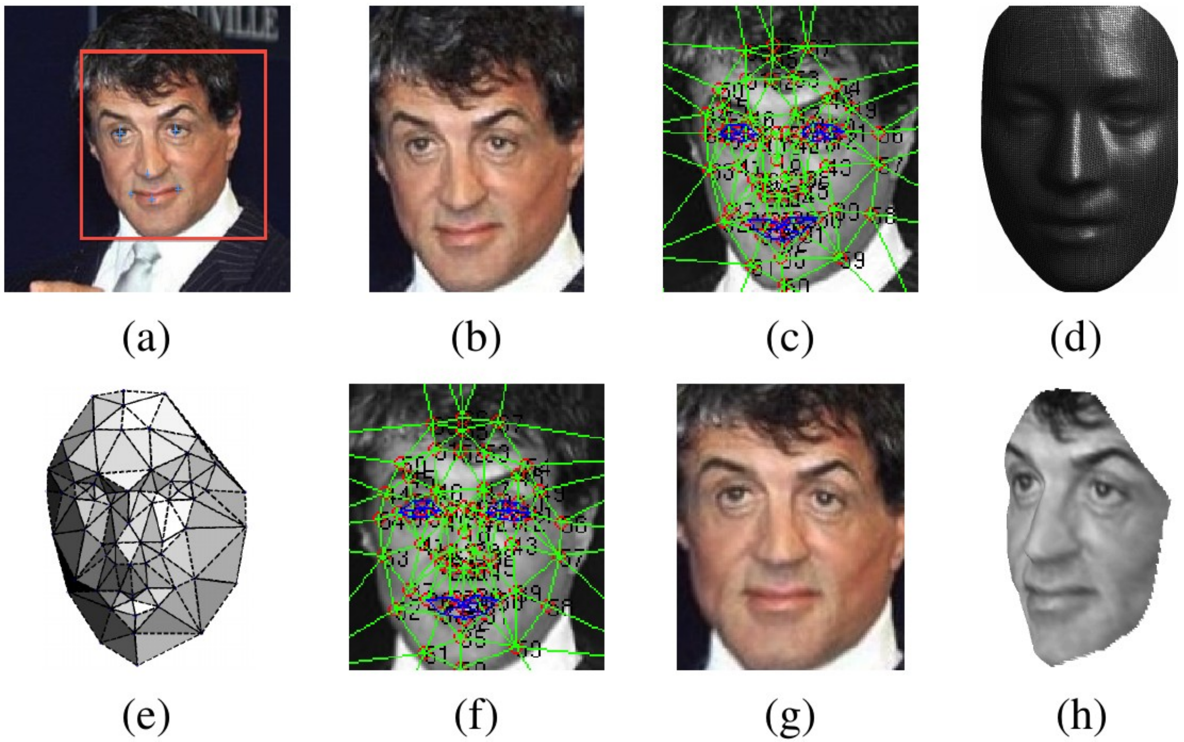
EMBEDDING THE WORLD

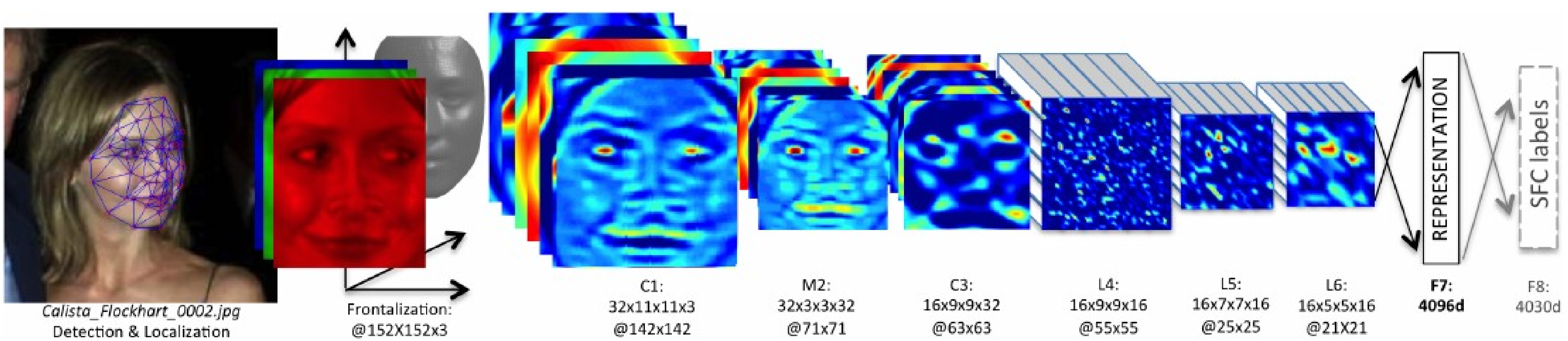


INSTAGRAM EMBEDDING VIDEO

Deep Face

- **[Taigman et al. CVPR 2014]**
  - Alignment
  - ConvNet
  - Metric Learning
- **Deployed at Facebook for Auto-tagging**
  - 800 million photos per day

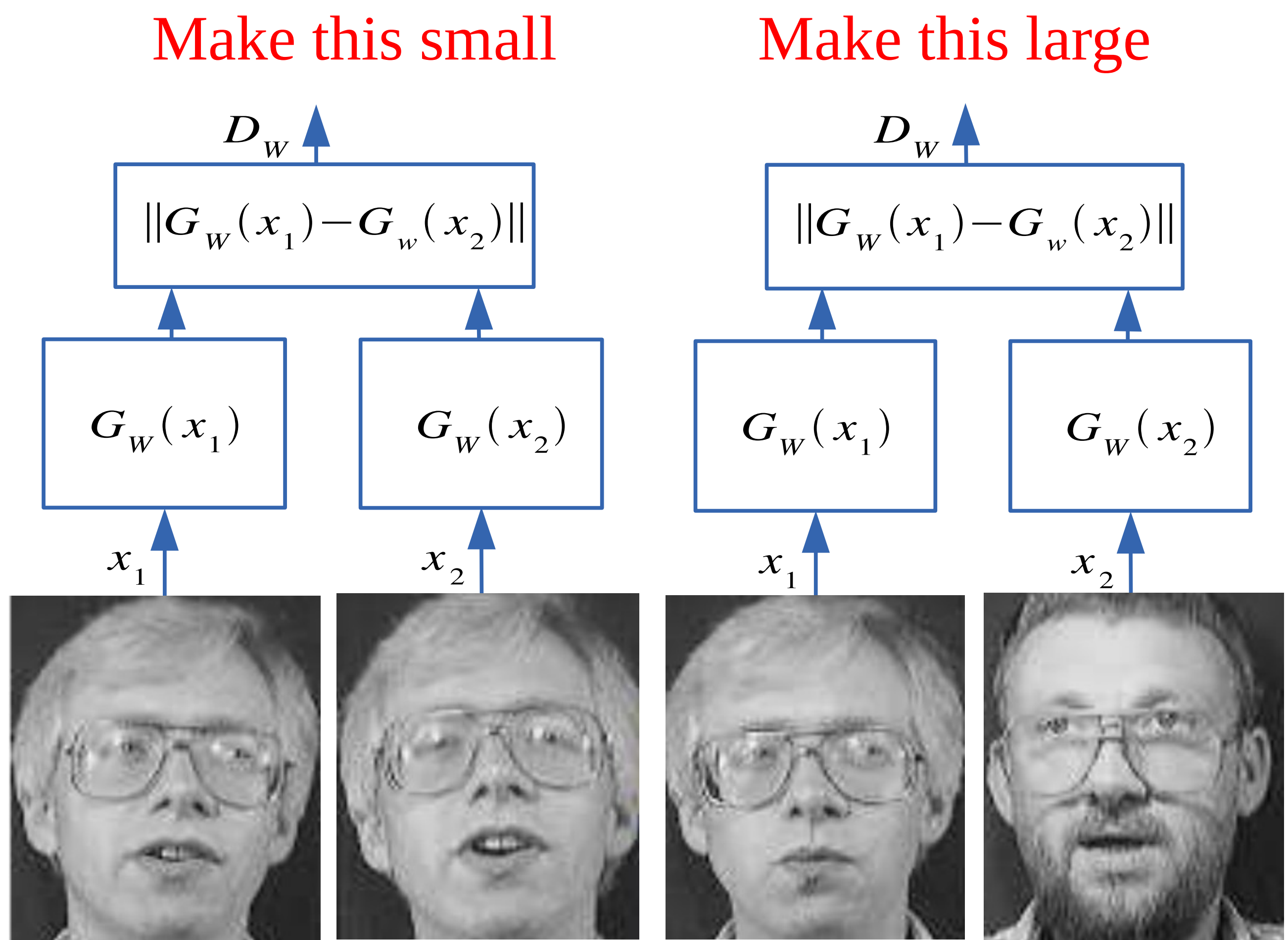




Metric Learning with a Siamese Architecture

Contrastive Objective Function

- Similar objects should produce outputs that are nearby
- Dissimilar objects should produce output that are far apart.
- DrLIM: Dimensionality Reduction by Learning and Invariant Mapping
- [Chopra et al. CVPR 2005]
- [Hadsell et al. CVPR 2006]



Similar images (neighbors in the neighborhood graph)

Dissimilar images (non-neighbors in the neighborhood graph)

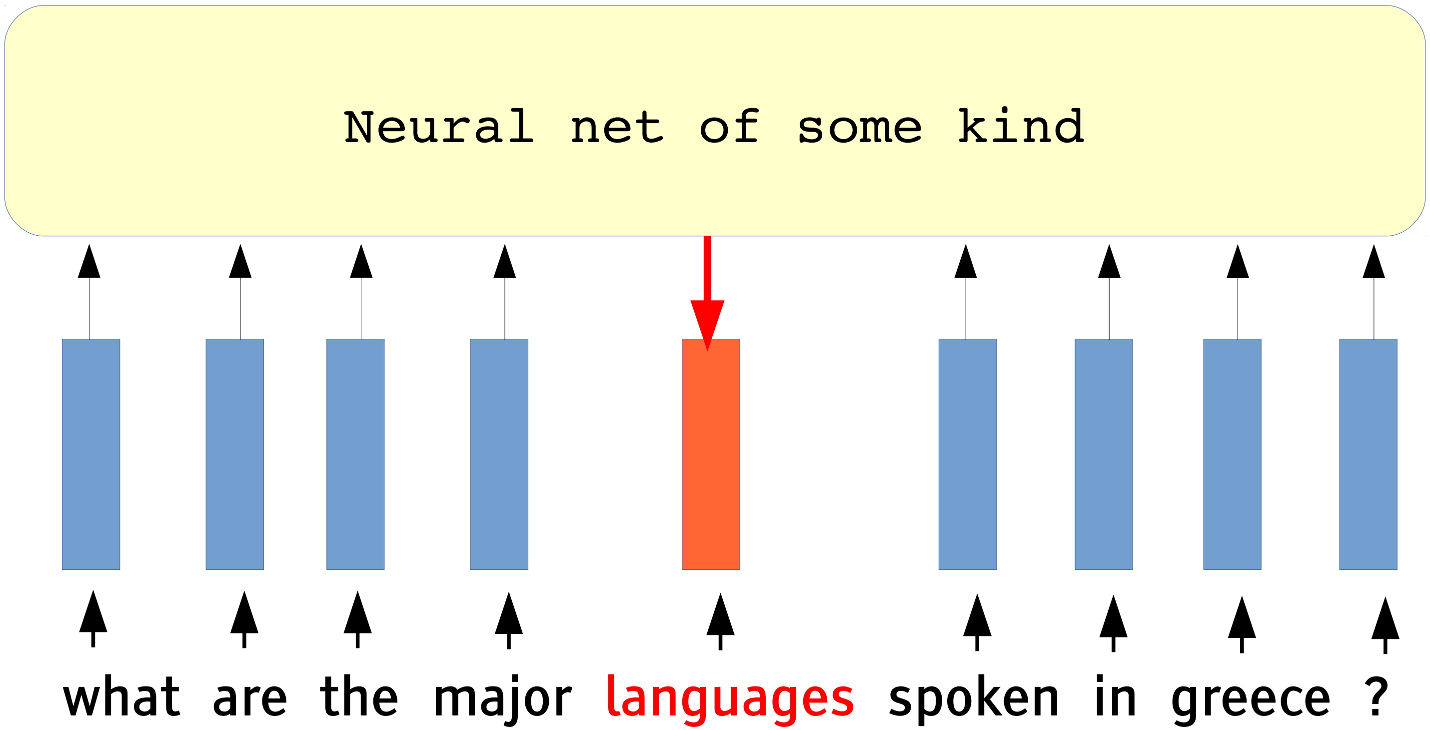
Representing the world with “thought vectors”

- **Every object, concept or “thought” can be represented by a vector**
  - [-0.2, 0.3, -4.2, 5.1, …..] represent the concept “cat”
  - [-0.2, 0.4, -4.0, 5.1, …..] represent the concept “dog”
  - The vectors are similar because cats and dogs have many properties in common
- **Reasoning consists in manipulating thought vectors**
  - Comparing vectors for question answering, information retrieval, content filtering
  - Combining and transforming vectors for reasoning, planning, translating languages
- **Memory stores thought vectors**
  - MemNN (Memory Neural Network) is an example
- **At FAIR we want to “embed the world” in thought vectors**

Natural Language Understanding (with embeddings)

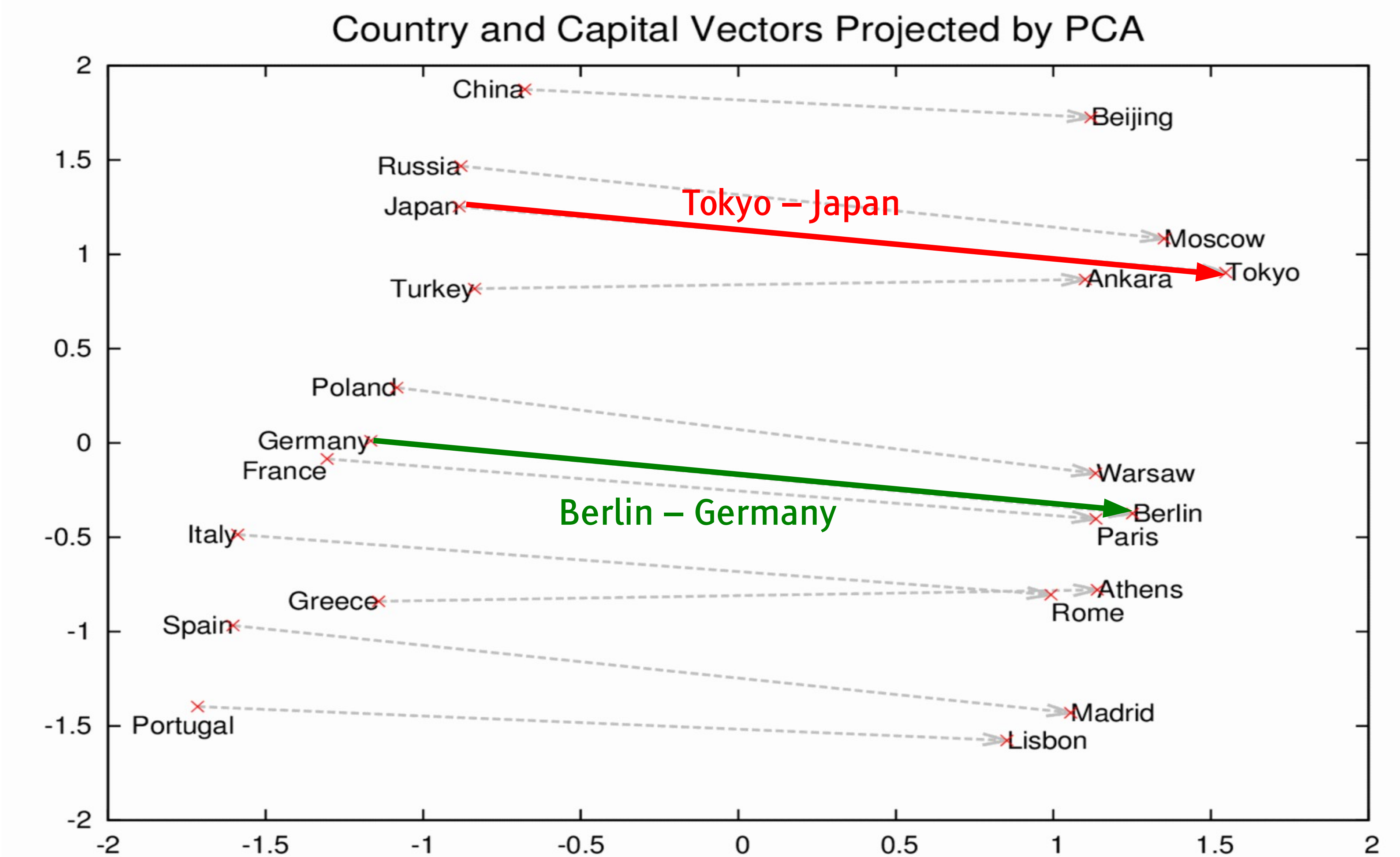
What about Language? Word Embedding

- **Word Embedding in continuous vector spaces**
  - [Bengio 2003][Collobert & Weston 2010]
  - Word2Vec [Mikolov 2011]
  - Predict a word from previous words and/or following words

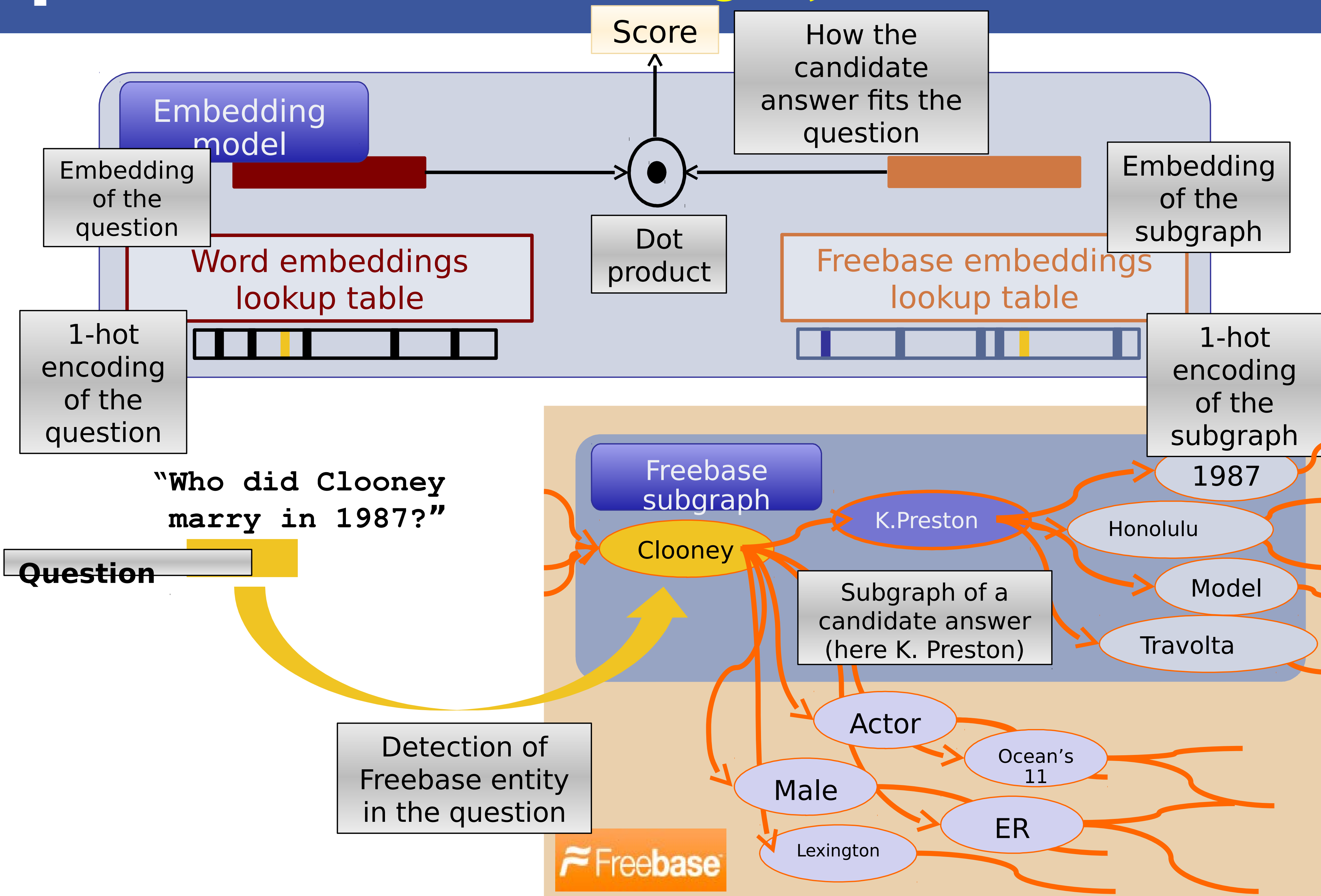


Compositional Semantic Property

- Tokyo – Japan = Berlin – Germany   Tokyo - Japan + Germany = Berlin



Question-Answering System



Question-Answering System

what are bigos?

["stew"] ["stew"]

what are dallas cowboys colors?

[“navy_blue", "royal_blue", "blue", "white", "silver"] ["blue", "navy_blue", "white", "royal_blue", "silver"]

how is egyptian money called?

["egyptian_pound"] ["egyptian_pound"]

what are fun things to do in sacramento ca?

["sacramento_zoo"] ["raging_waters_sacramento", "sutter_s_fort", "b_street_theatre", "sacramento_zoo", "california_state_capitol_museum", ...]

how are john terry's children called?

["georgie_john_terry", "summer_rose_terry"] ["georgie_john_terry", "summer_rose_terry"]

what are the major languages spoken in greece?

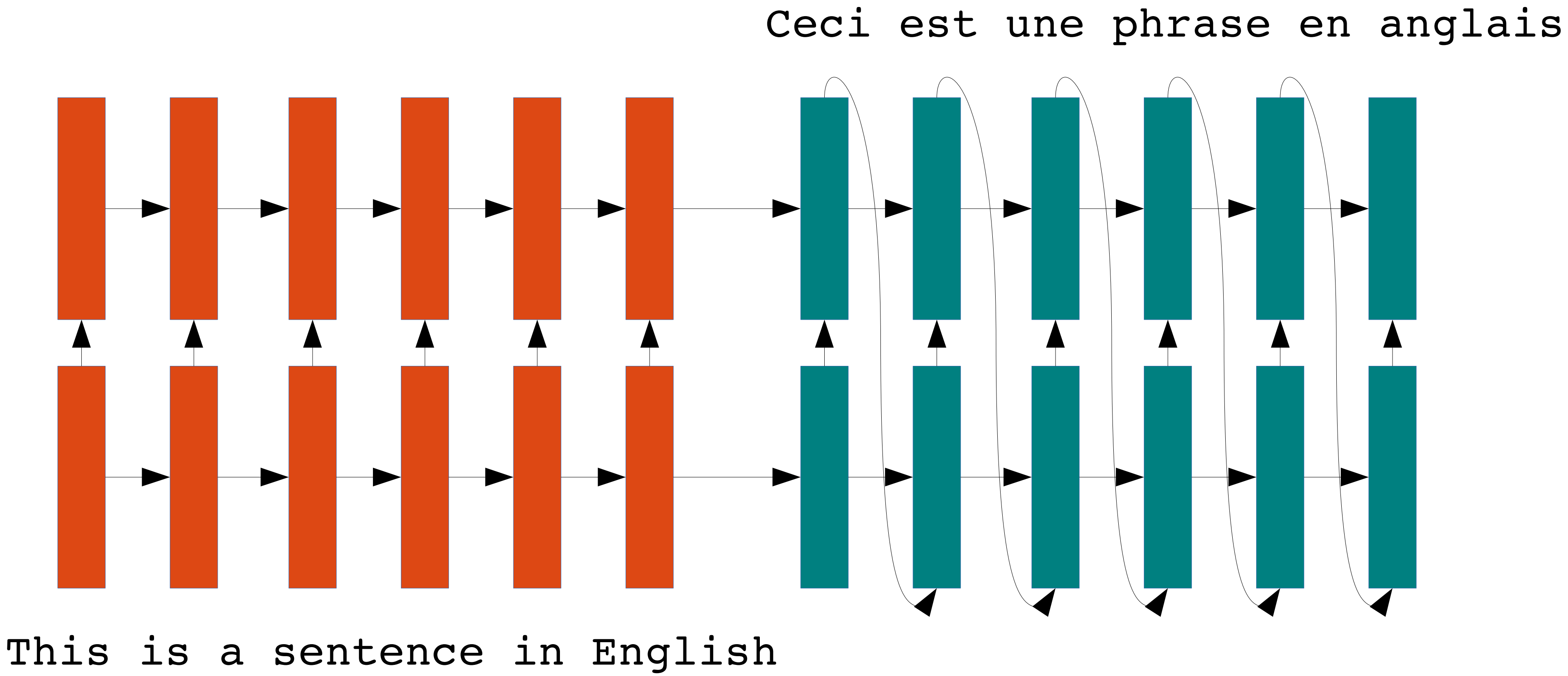
["greek_language", "albanian_language"] ["greek_language", "albanian_language"]

what was laura ingalls wilder famous for?

["writer", "author"] ["writer", "journalist", "teacher", "author"]

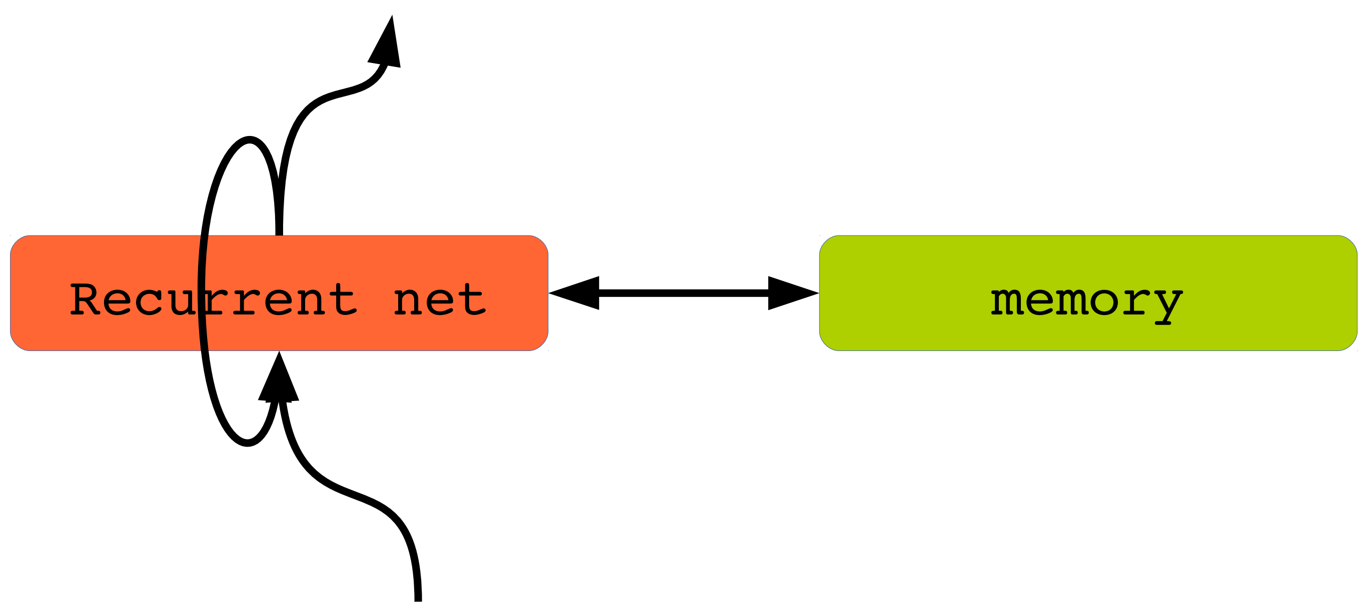
Language Translation with Recurrent networks

- [Sutskever et al. NIPS 2014]
  - Multiple layers of very large LSTM recurrent modules
    - [Hochreiter & Schmidhuber 1997]
  - English sentence is read in and encoded
  - French sentence is produced after the end of the English sentence
  - Accuracy is very close to state of the art.



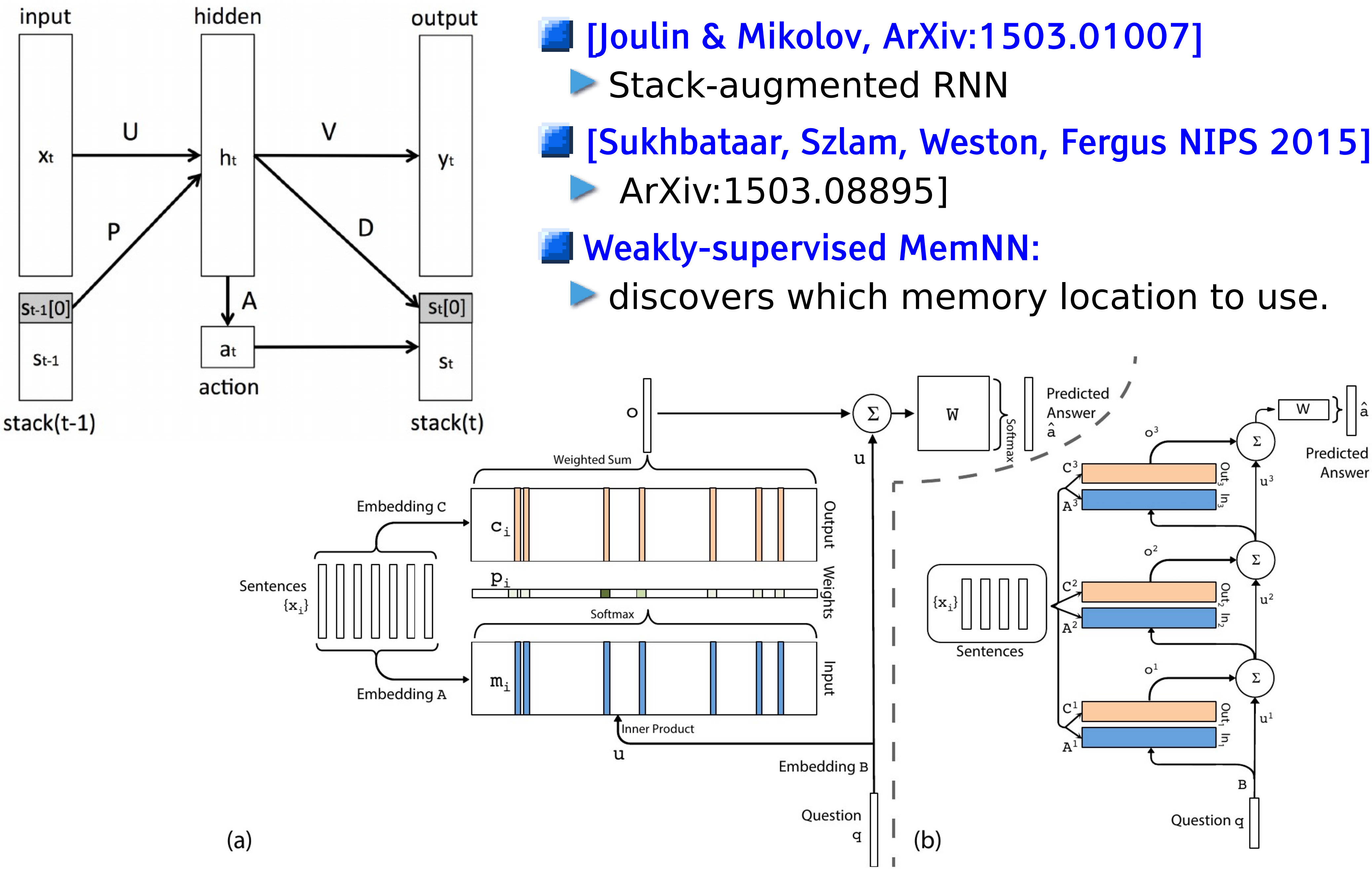
But How can Neural Nets Remember Things?

- **Recurrent networks cannot remember things for very long**
  - The cortex only remember things for 20 seconds
- **We need a “hippocampus” (a separate memory module)**
  - LSTM [Hochreiter 1997], registers
  - **Memory networks** [Weston et 2014] (FAIR), associative memory
  - **Stacked-Augmented Recurrent Neural Net** [Joulin & Mikolov 2014] (FAIR)
  - NTM [DeepMind 2014], “tape”.



Memory/Stack-Augmented Recurrent Nets

- [Joulin & Mikolov, ArXiv:1503.01007]
  - Stack-augmented RNN
- [Sukhbataar, Szlam, Weston, Fergus NIPS 2015]
  - ArXiv:1503.08895]
- Weakly-supervised MemNN:
  - discovers which memory location to use.



(a) (b)

Memory Network [Weston, Chopra, Bordes 2014]

Add a short-term memory to a network

http://arxiv.org/abs/1410.3916

I: (input feature map) – converts the incoming input to the internal feature representation.
G: (generalization) – updates old memories given the new input.
O: (output feature map) – produces a new output (in the feature representation space), given the new input and the current memory.
R: (response) – converts the output into the response format desired. For example, a textual response or an action.

Method	F1
(Fader et al., 2013) [4]	0.54
(Bordes et al., 2014) [3]	0.73
MemNN	0.71
MemNN (with BoW features)	0.79

Results on Question Answering Task

Bilbo travelled to the cave.
Gollum dropped the ring there.
Bilbo took the ring.
Bilbo went back to the Shire.
Bilbo left the ring there.
Frodo got the ring.
Frodo journeyed to Mount-Doom.
Frodo dropped the ring there.
Sauron died.
Frodo went back to the Shire.
Bilbo travelled to the Grey-havens.
The End.
Where is the ring? A: Mount-Doom
Where is Bilbo now? A: Grey-havens
Where is Frodo now? A: Shire

Fig. 2. An example story with questions correctly answered by a MemNN. The MemNN was trained on the simulation described in Section 4.2 and had never seen many of these words before, e.g. Bilbo, Frodo and Gollum.

Obstacles to Progress in Artificial Intelligence:

Four missing pieces for AI (besides computation)

Theoretical Understanding for Deep Learning

- What is the geometry of the objective function in deep networks?
- Why the ConvNet architecture works so well? [Mallat, Bruna, Tygert..]

Integrating Representation/Deep Learning with Reasoning, Attention, Planning and Memory

- A lot of recent work on reasoning/planning, attention, memory, learning "algorithms".
- Memory-augmented neural nets
- "Differentiable" algorithms

Integrating supervised, unsupervised and reinforcement learning into a single "algorithm".

- Boltzmann Machines would be nice if they worked.
- Stacked What-Where Auto-Encoders, Ladder Networks....

Effective ways to do unsupervised Learning

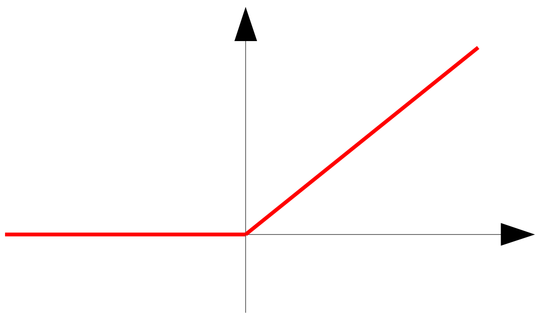
- Discovering the structure and regularities of the world by observing it and living in it like animals and humans do.

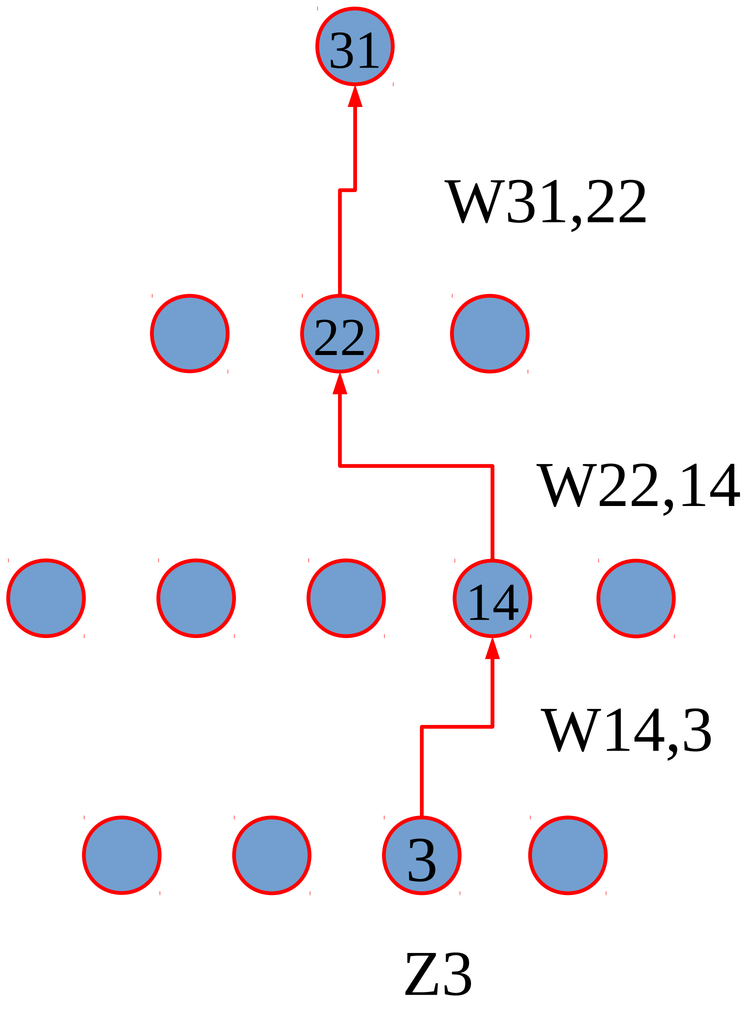
The Mysterious Geometry of the Objective Function

Deep Nets with ReLUs and Max Pooling

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

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





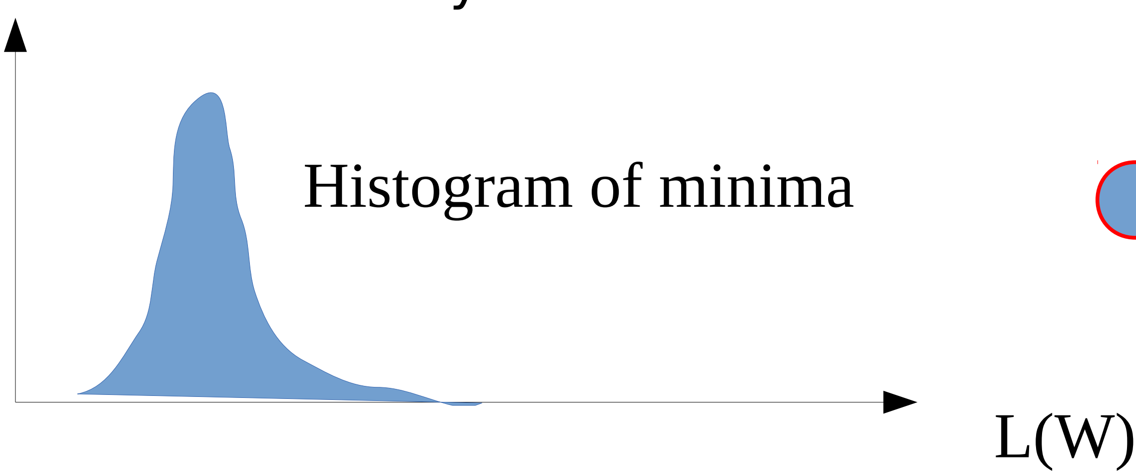
- Max Pooling
  - "switches" from one layer to the next

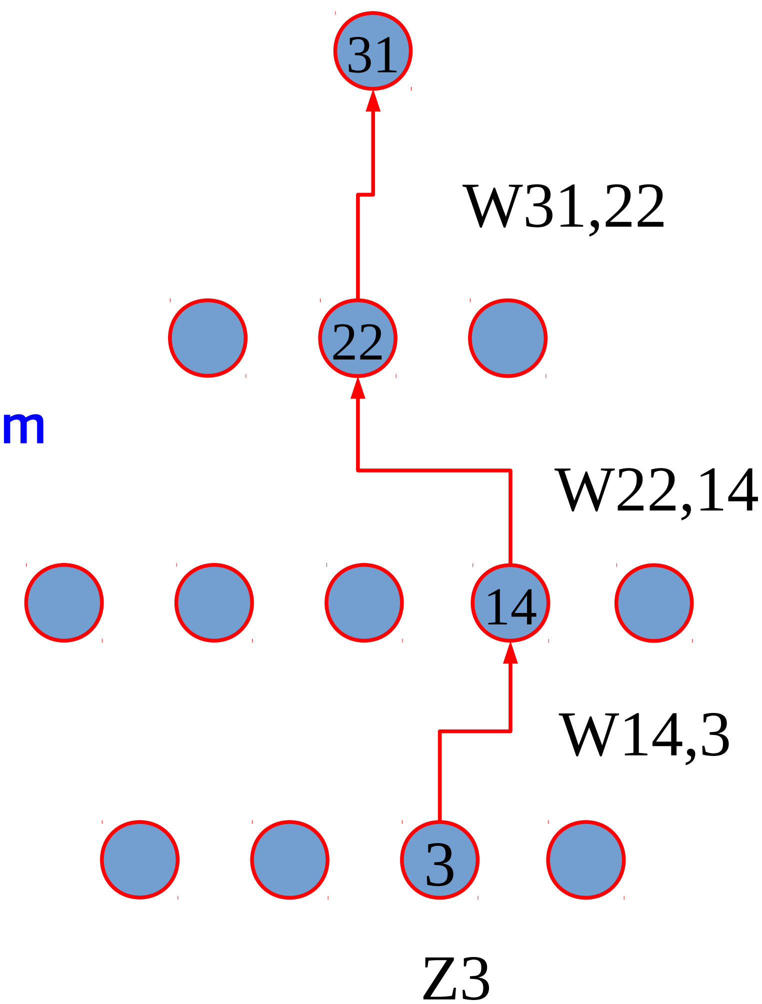
Deep Nets with ReLUs:
Objective Function is Piecewise Polynomial

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

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

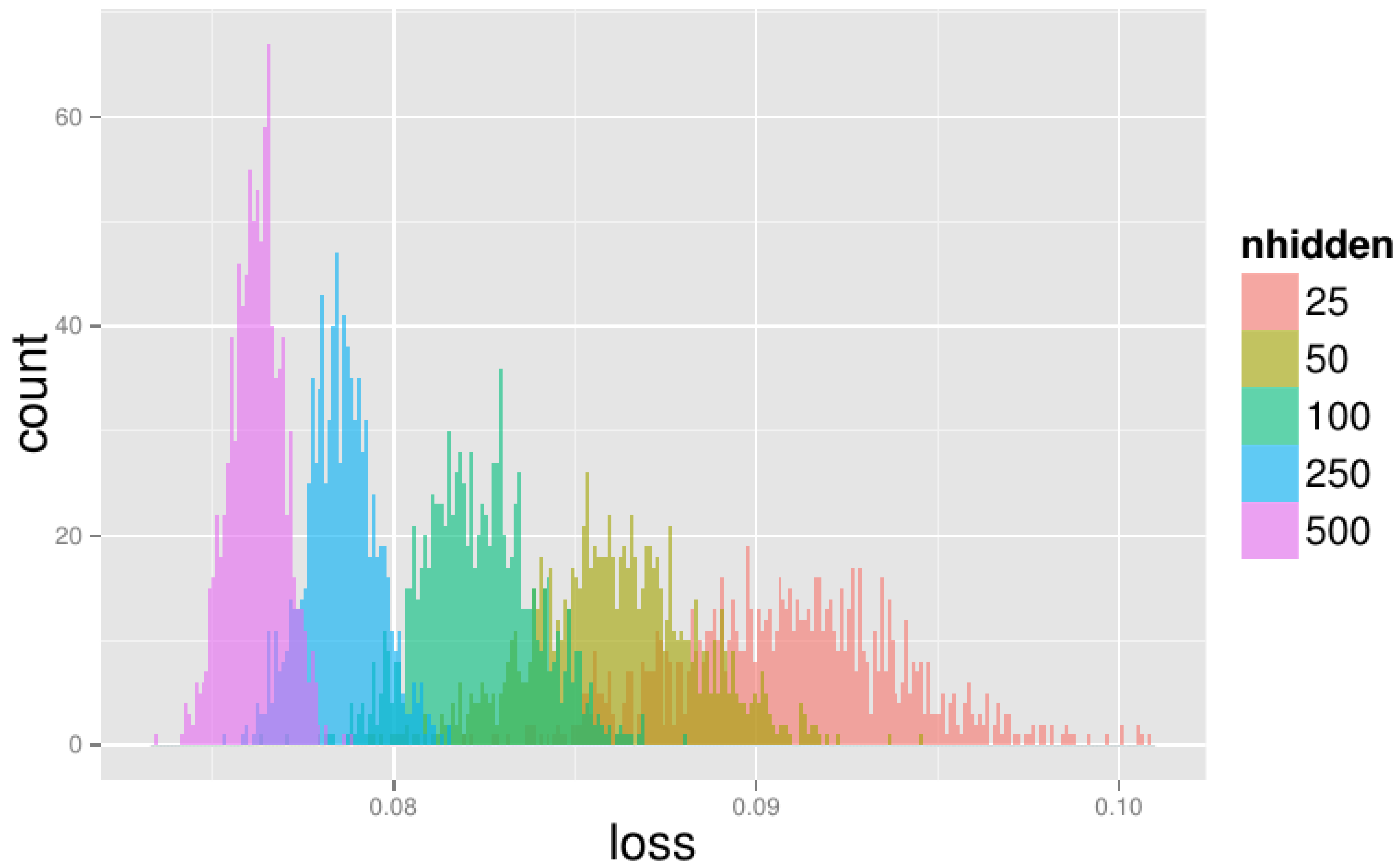
- Piecewise polynomial in W with random coefficients
- A lot is known about the distribution of critical points of polynomials on the sphere with random (Gaussian) coefficients [Ben Arous et al.]
  - High-order spherical spin glasses
  - Random matrix theory





Deep Nets with ReLUs: Objective Function is Piecewise Polynomial

- Train 2-layer nets on scaled-down MNIST (10x10) from multiple initial conditions. Measure loss on test set.



[Choromanska, Henaff, Mathieu, Ben Arous, LeCun 2015]

Reinforcement Learning, Supervised Learning Unsupervised Learning: The Three Types of Learning

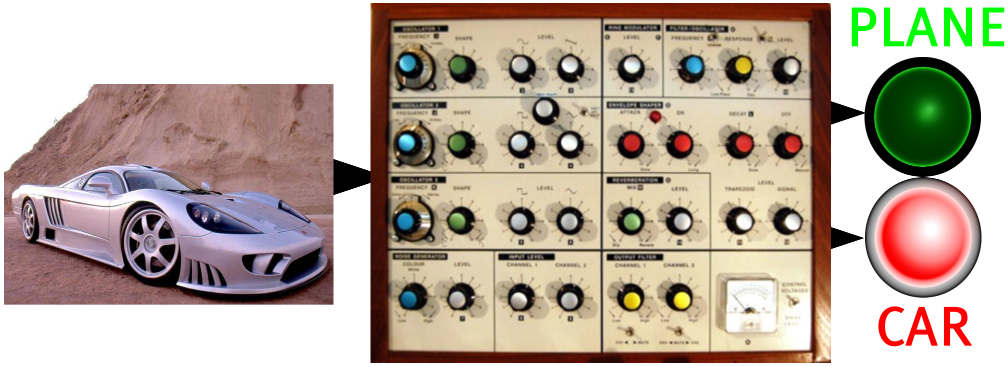
Three Types of Learning

Reinforcement Learning

– The machine predicts a scalar reward given once in a while.
– A few bits for some samples

Supervised Learning

– The machine predicts a category or a few numbers for each input
– 10→10,000 bits per sample



Unsupervised Learning

– The machine predicts any part of its input for any observed part.
– Predicts future frames in videos
– Millions of bits per sample

How Much Information Does the Machine Need to Predict?

Y LeCun

Reinforcement Learning (cherry)
- The machine predicts a scalar reward given once in a while.
- **A few bits for some samples**

Supervised Learning (icing)
- The machine predicts a category or a few numbers for each input
- **10→10,000 bits per sample**

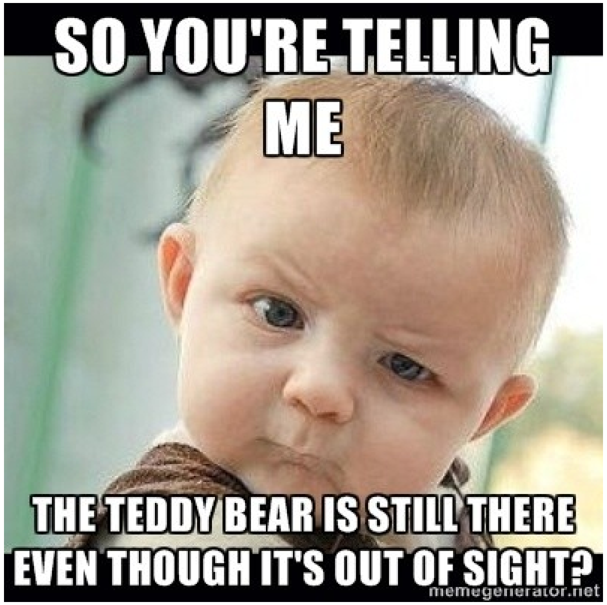
Unsupervised Learning (cake)
- The machine predicts any part of its input for any observed part.
- Predicts future frames in videos
- **Millions of bits per sample**

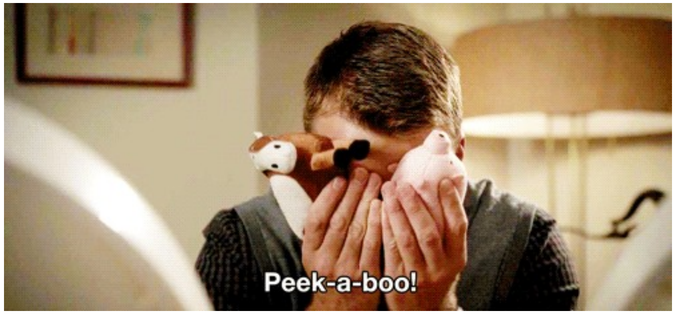
Unsupervised Learning is the “Dark Matter” of AI

Most of the learning performed by animals and humans is unsupervised

We learn how the world works by observing it

- We learn that the world is 3-dimensional
- We learn that objects can move independently of each other
- We learn object permanence
- We learn to predict what the world will look like one second or one hour from now.





We build a model of the world through predictive unsupervised learning

This predictive model gives us “common sense”

Unsupervised learning discovers regularities

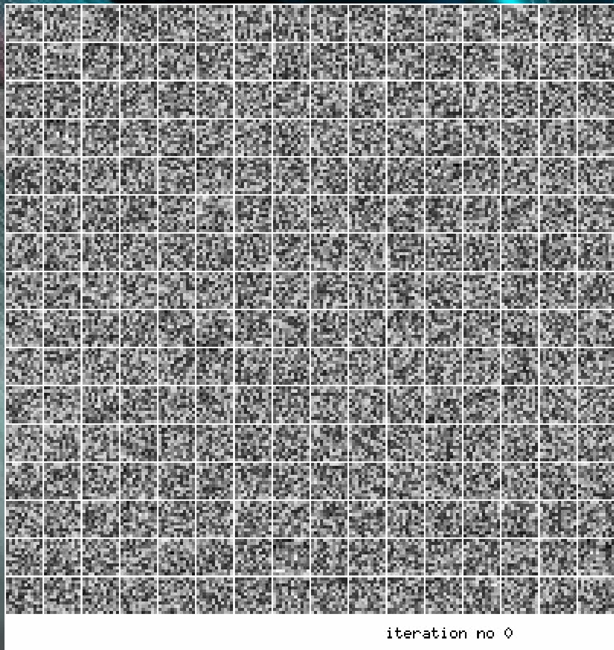
in the world.

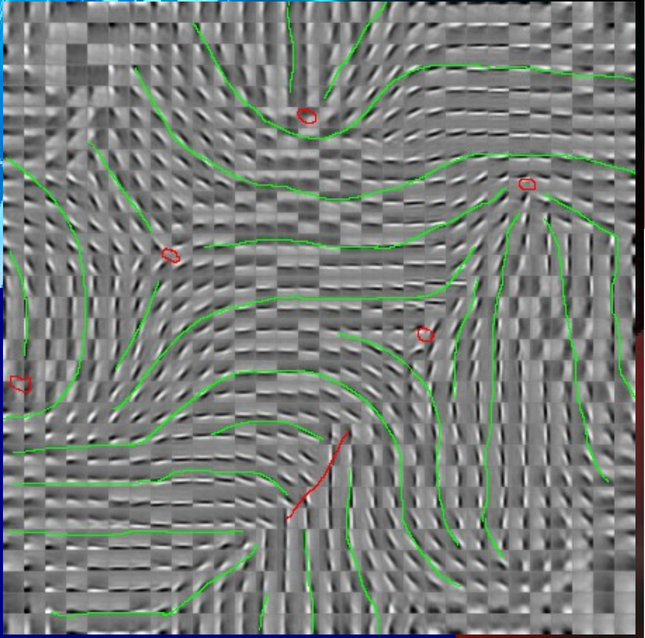
Learning a predictive model of the world gives us common sense.

If I say: "Gérard picks up his bag and leaves the room"

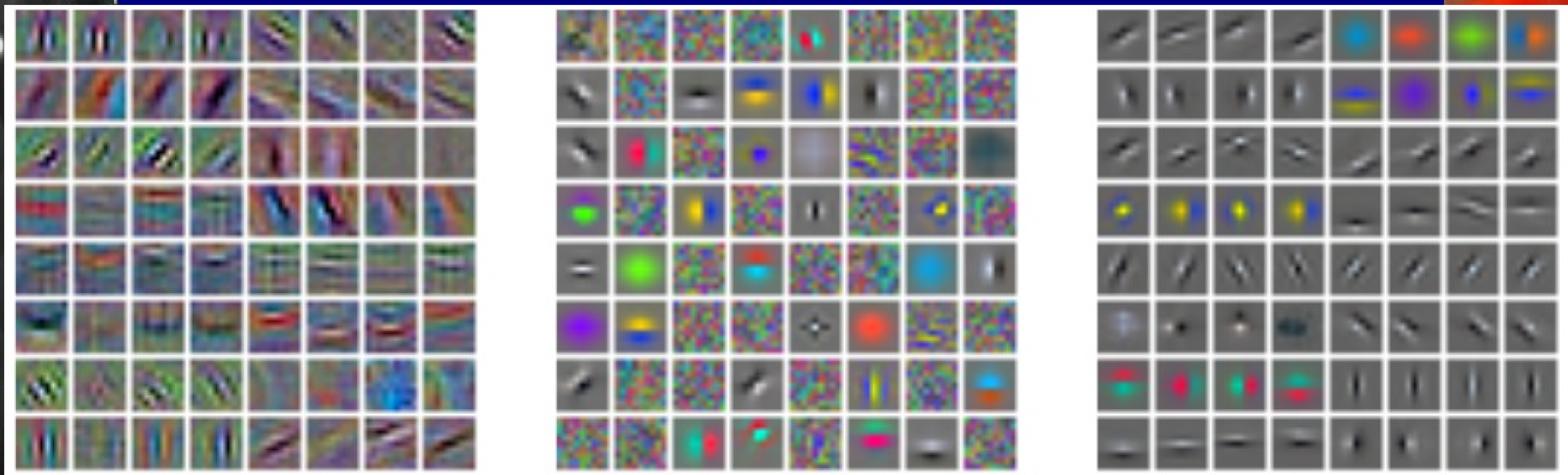
You can infer:

- Gérard stood up, extended his arm, walked towards the door, opened the door, walked out.
- He and his bag are not in the room anymore.
- He probably didn't dematerialize or fly out.





Unsupervised Learning

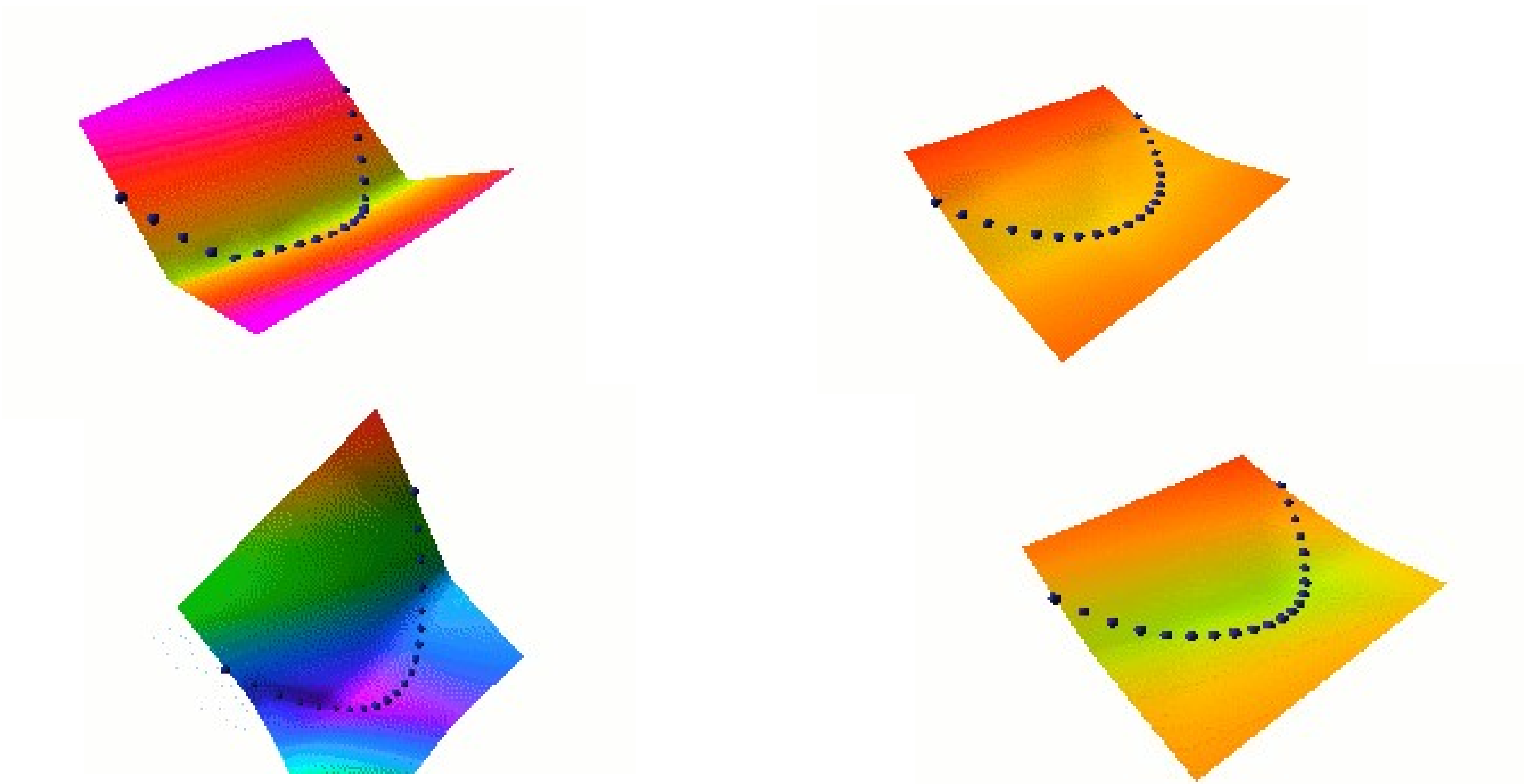


Energy-Based Unsupervised Learning

Energy Function: Takes low value on data manifold, higher values everywhere else

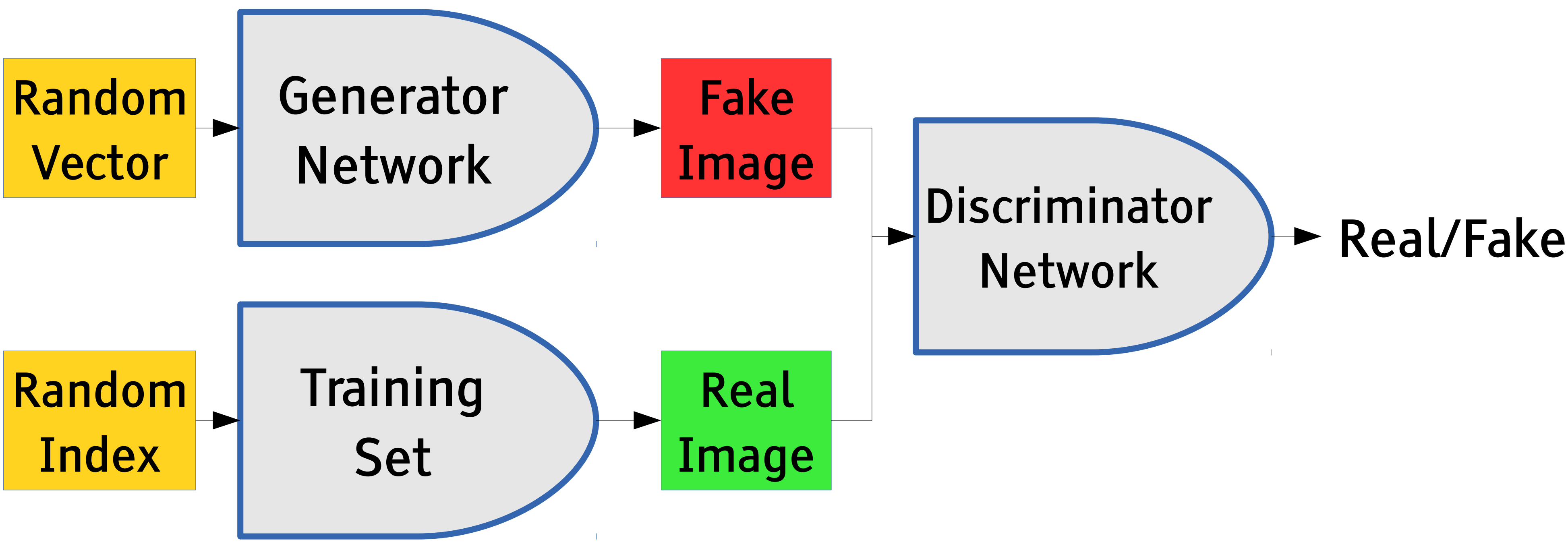
Push down on the energy of desired outputs

Push up on everything else



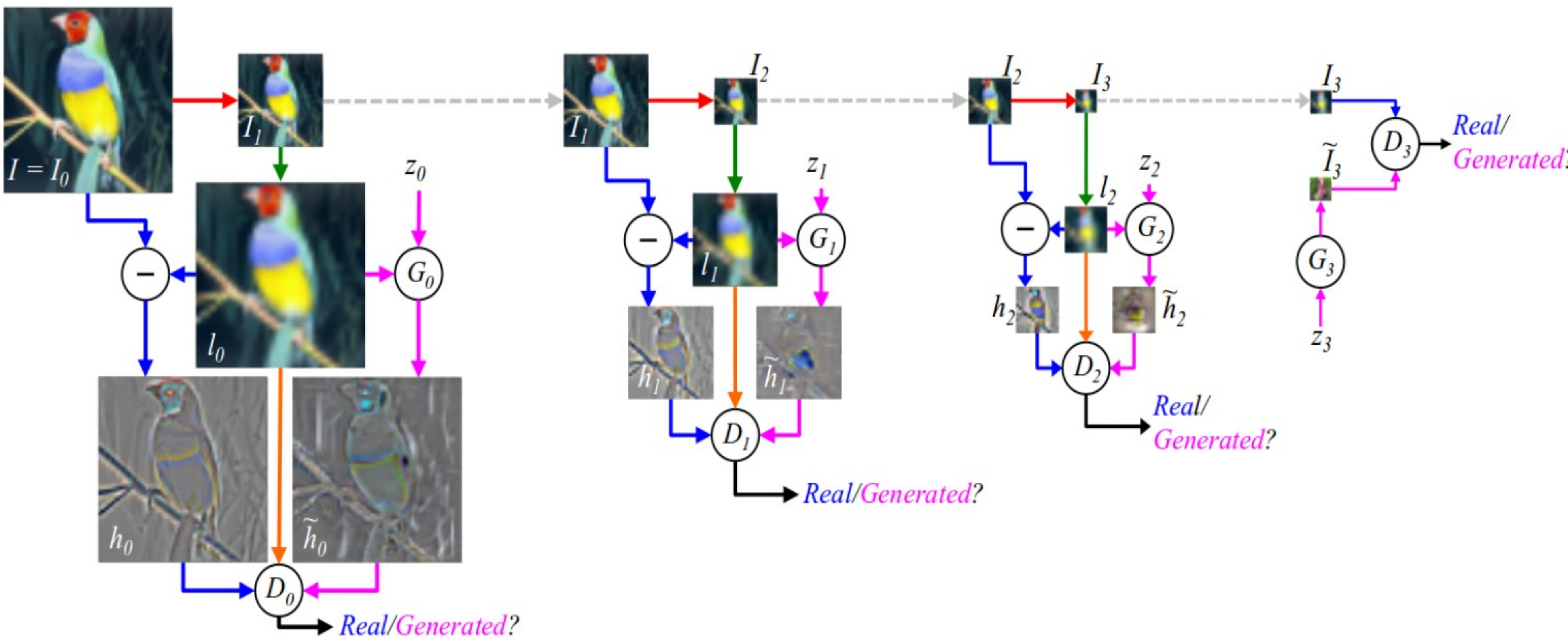
Generative Adversarial Networks

- [Goodfellow et al. NIPS 2014]
- Generator net maps random numbers to image
- Discriminator learns to tell real from fake images.
- Generator can cheat: it knows the gradient of the output of the discriminator with respect to its input



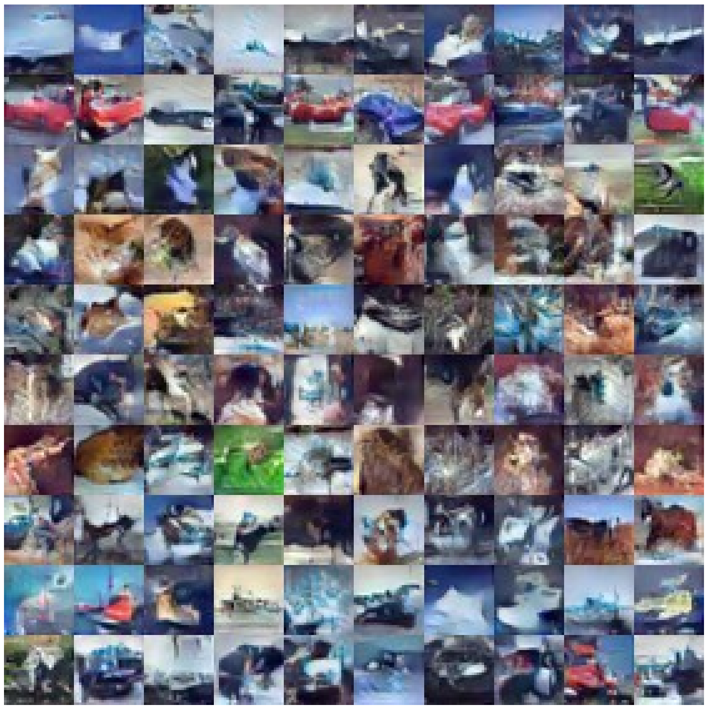
Laplacian GAN: LAPGAN (aka EyeScream)

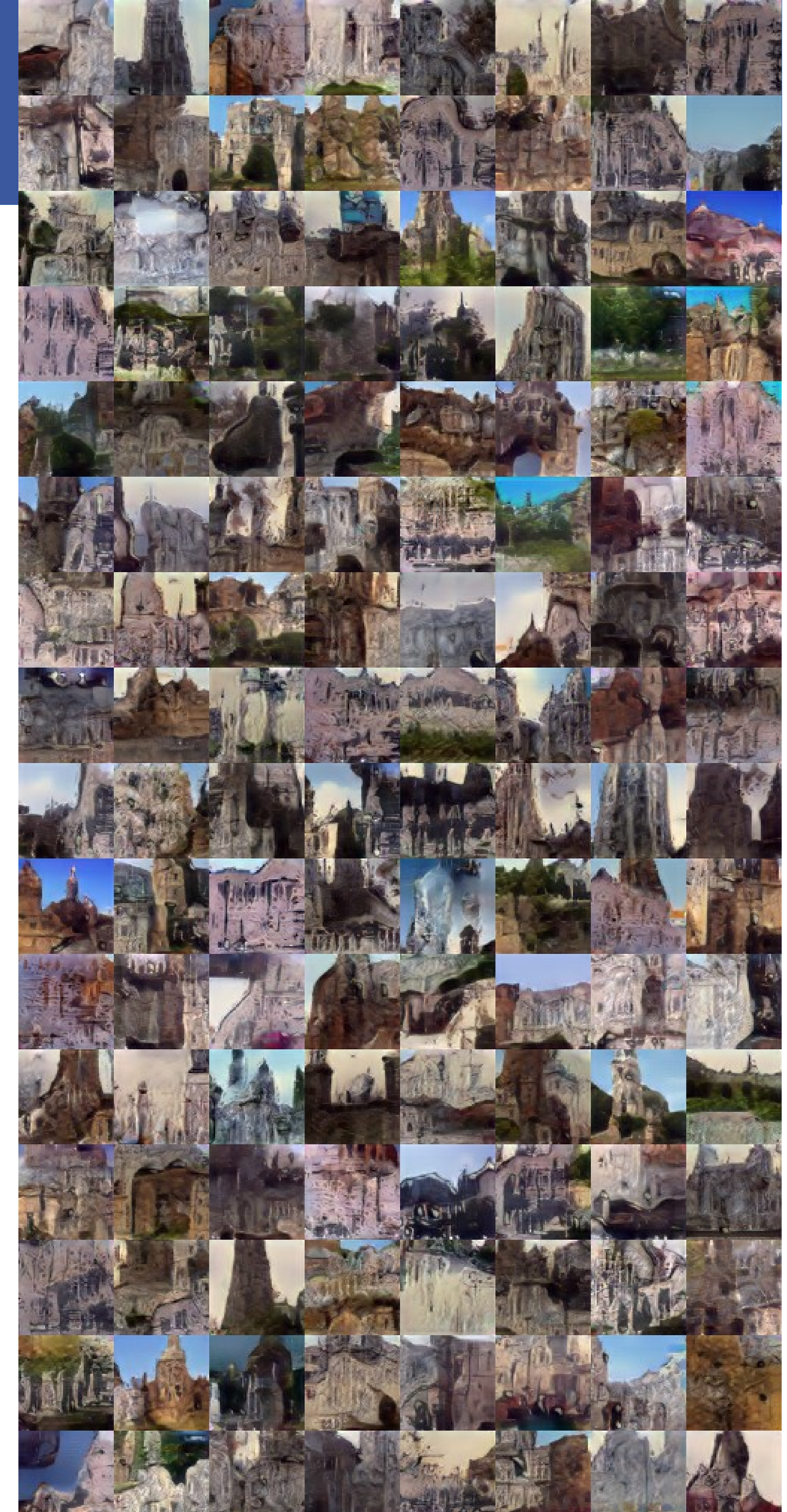
- **Learns to generate images [Denton et al. NIPS 2015]**
- **Generator net produces coefficients of a Laplacian Pyramid representation of the image**
- **Discriminator learns to tell real from fake Laplacian images.**



“EyeScream”

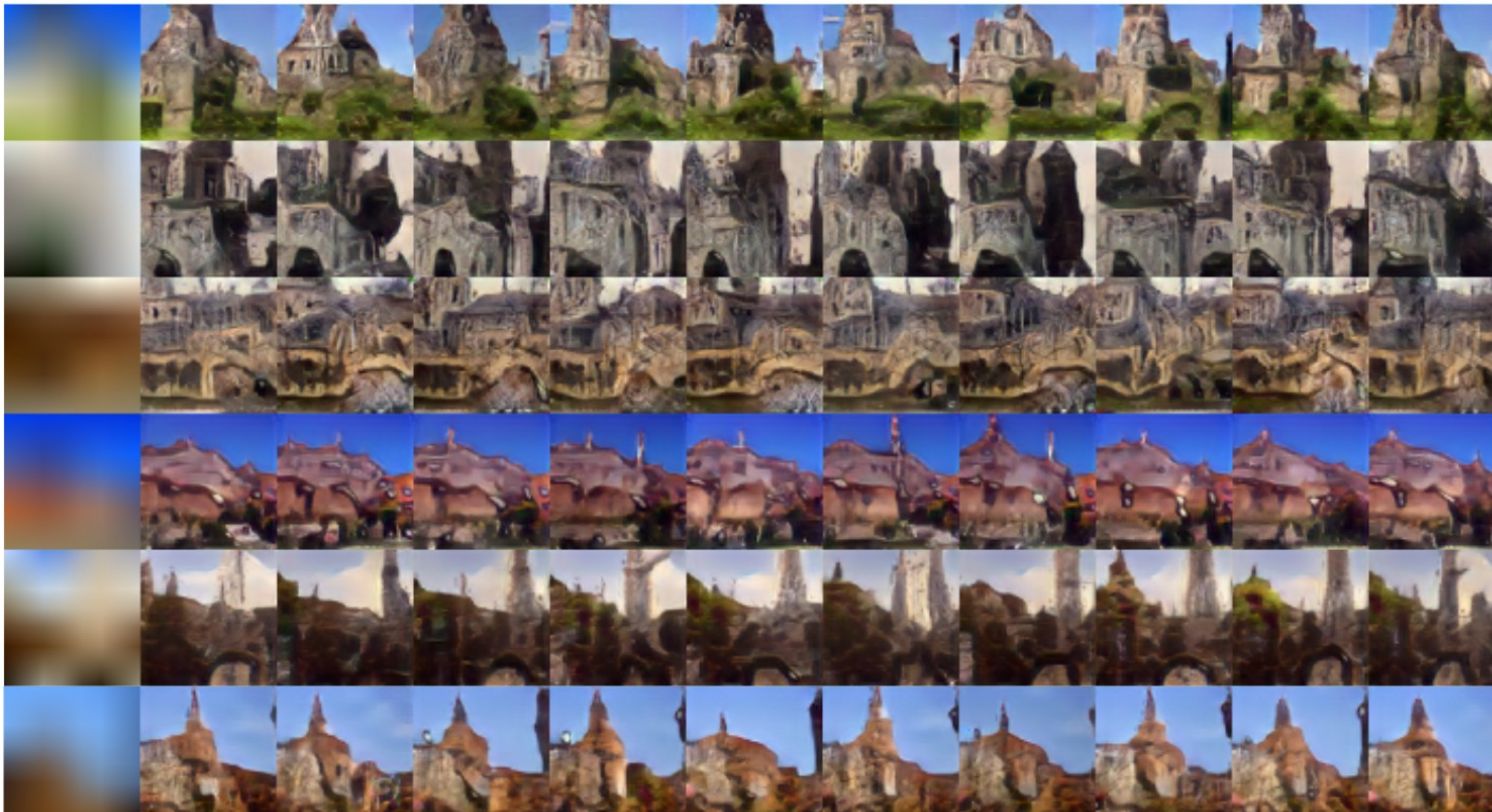
- http://soumith.ch/eyescream/





“EyeScream” / “LAPGAN”

- **http://soumith.ch/eyescream/**



Discovering Regularities

DCGAN: adversarial training to generate images.

[Radford, Metz, Chintala 2015]

- Input: random numbers; output: bedrooms.

Navigating the Manifold

DCGAN: adversarial training to generate images.

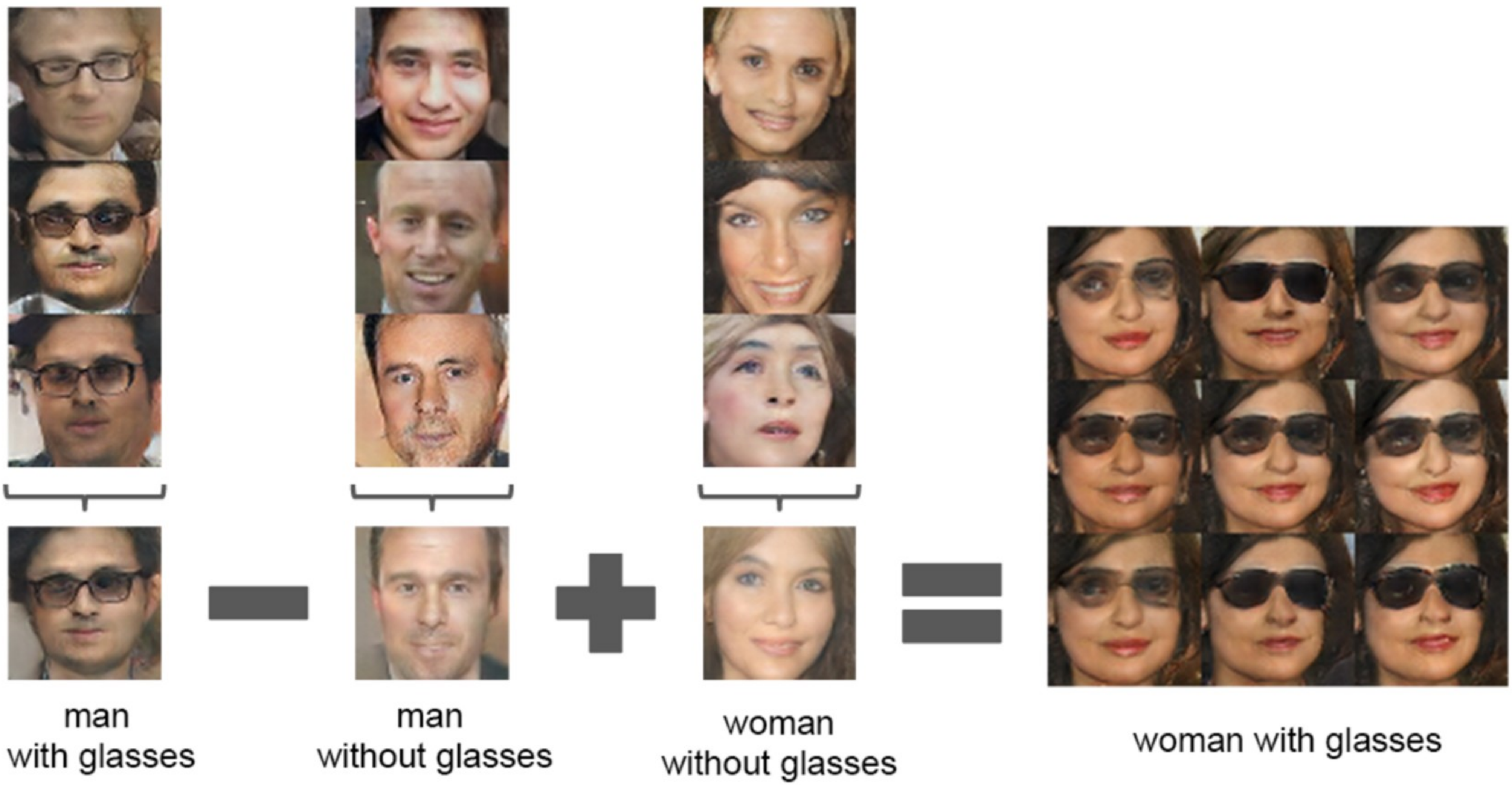
Trained on Manga characters

Interpolates between characters

Face Algebra (in DCGAN space)

DCGAN: adversarial training to generate images.

– [Radford, Metz, Chintala 2015]



Predictive Unsupervised Learning: Video Prediction

[Mathieu, Couprie, LeCun ICLR 2016]
arXiv:1511:05440

Unsupervised Learning is the “Dark Matter” of AI

Unsupervised learning is the only form of learning that can provide enough information to train large neural nets with billions of parameters.

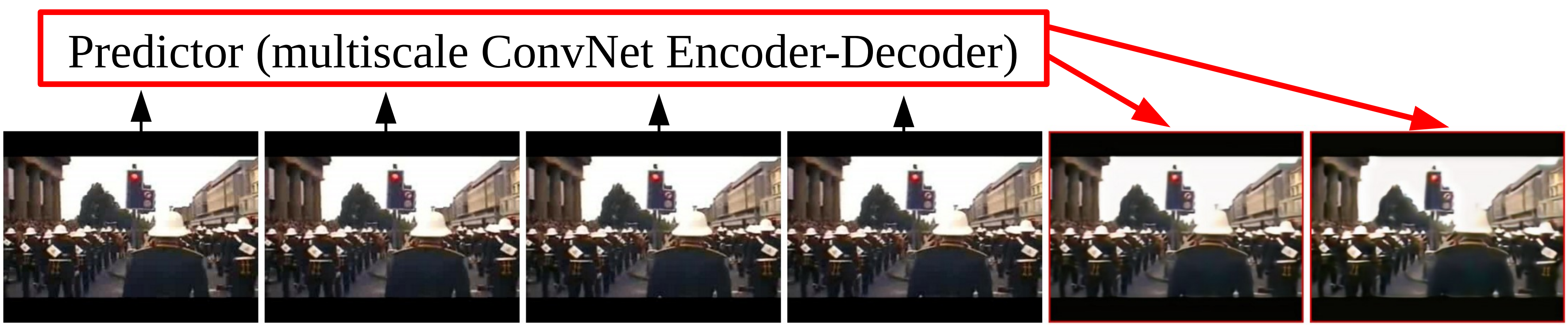
- Supervised learning would take too much labeling effort
- Reinforcement learning would take too many trials

But we don't know how to do unsupervised learning (or even formulate it)

- We have lots of ideas and methods
- They just don't work that well yet.

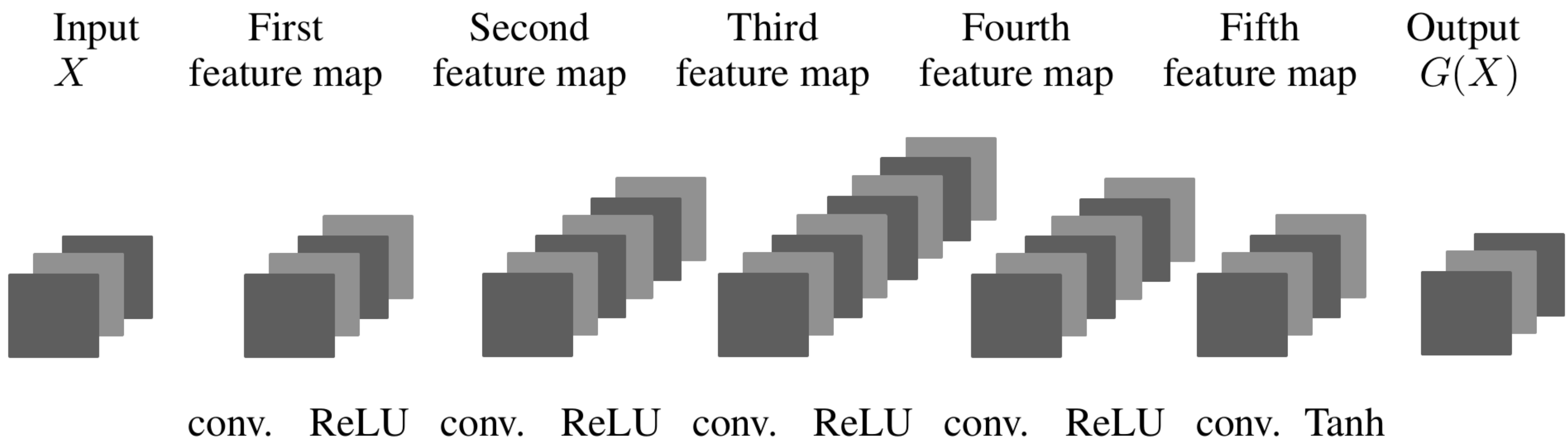
Why is it so hard? The world is unpredictable!

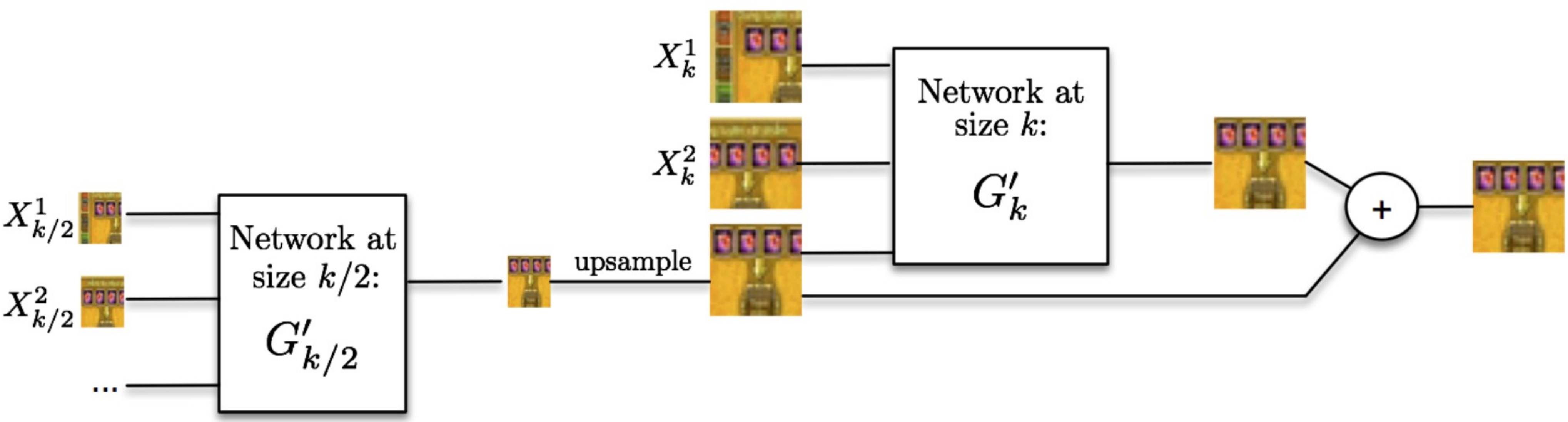
- Predictors produce an average of all possible futures → **Blurry image.**



Multi-Scale ConvNet for Video Prediction

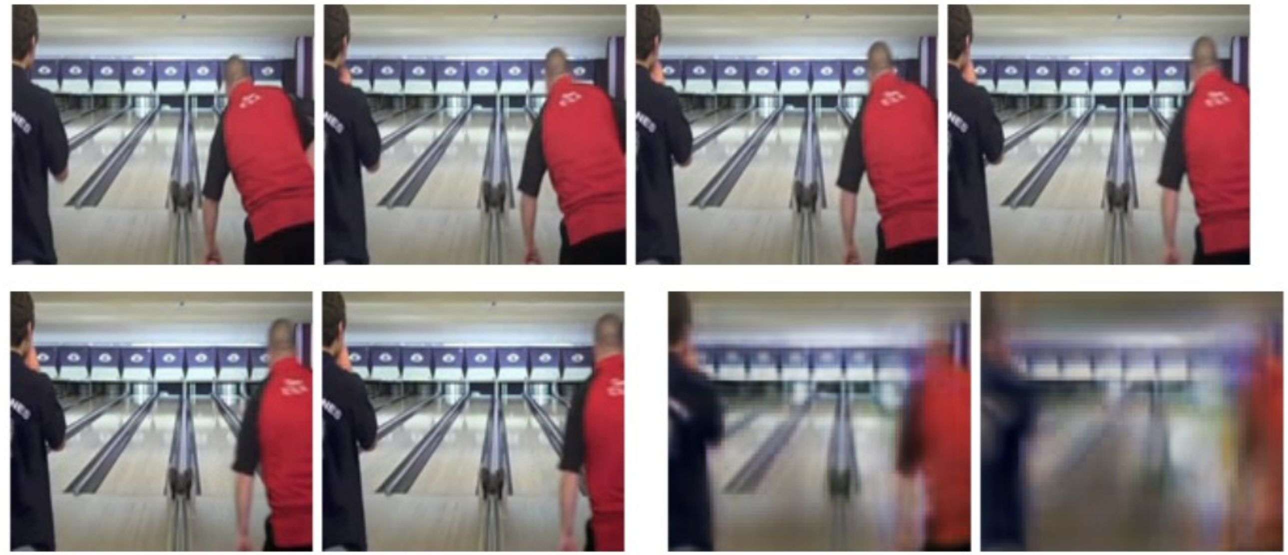
- **4 to 8 frames input → ConvNet with no pooling → 1 to 8 frames output**



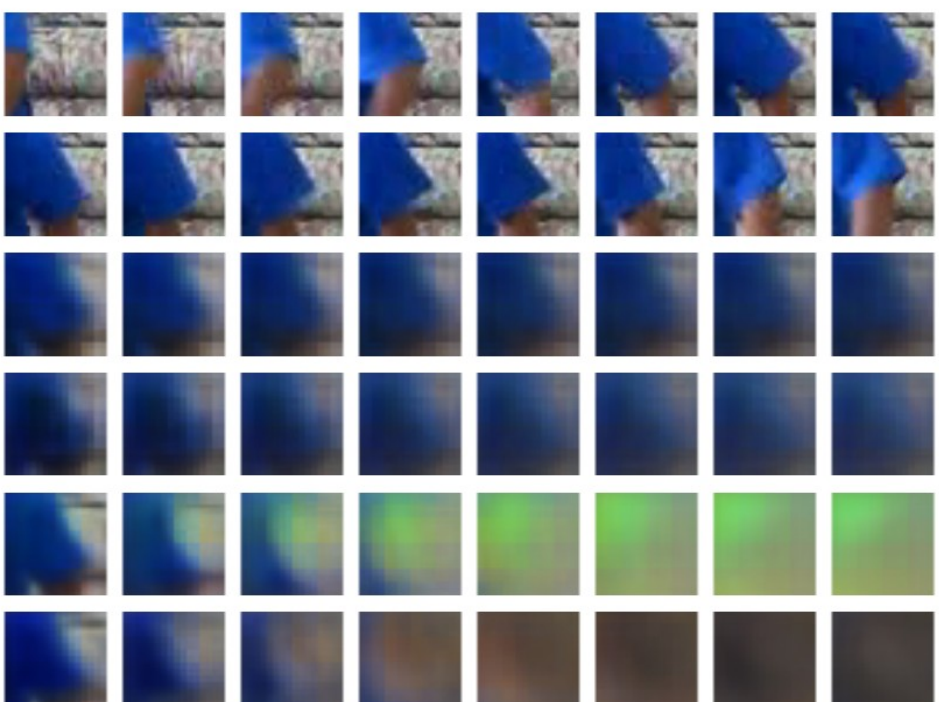


Can't Use Squared Error: blurry predictions

- **The world is unpredictable**
- **MSE training predicts the average of possible futures: blurry images.**



Ground truth ℓ_2 result



Input

Ground truth

ℓ_1

ℓ_2

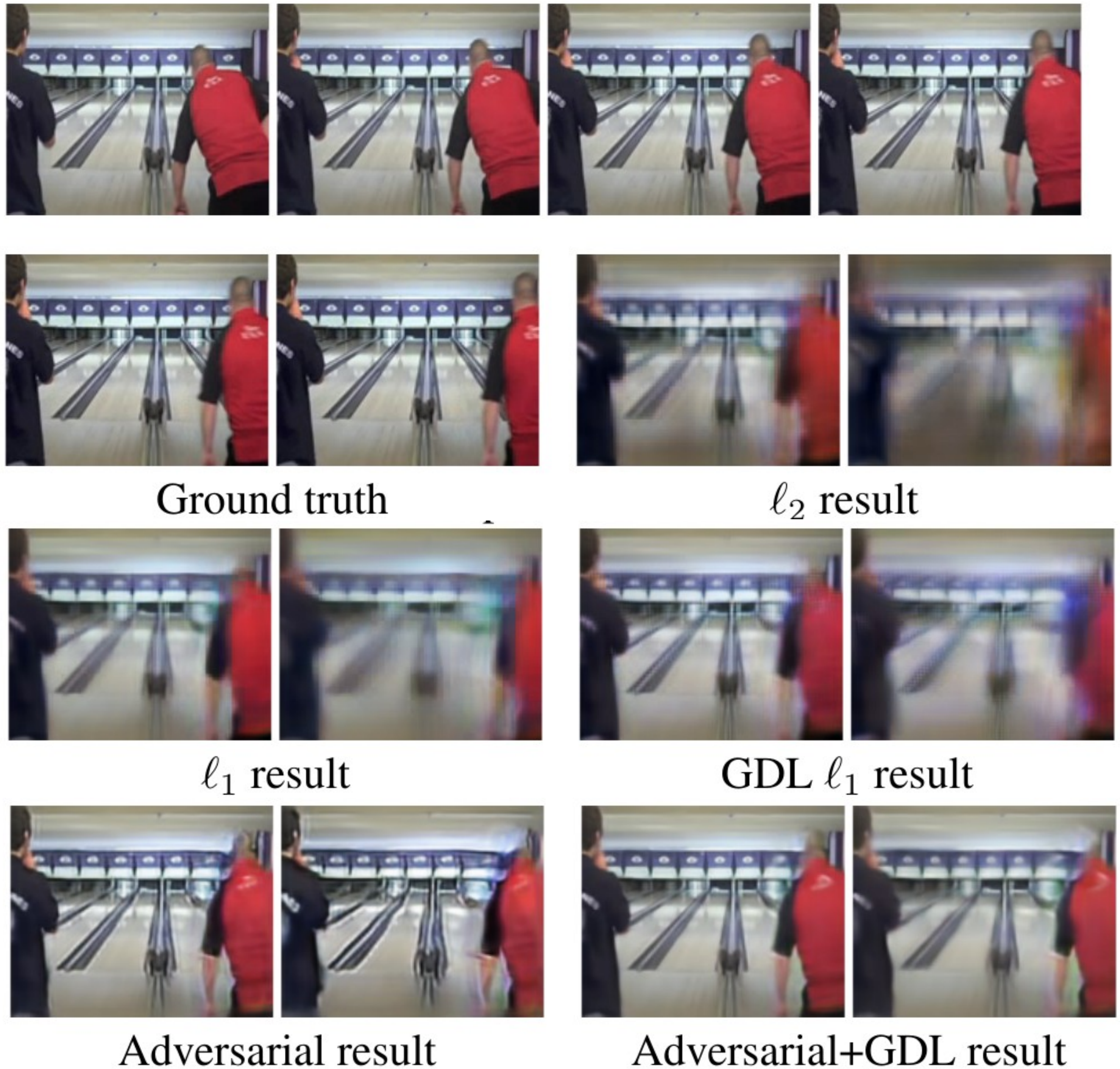
ℓ_1 recursive

ℓ_2 recursive

Multi-Scale ConvNet for Video Prediction

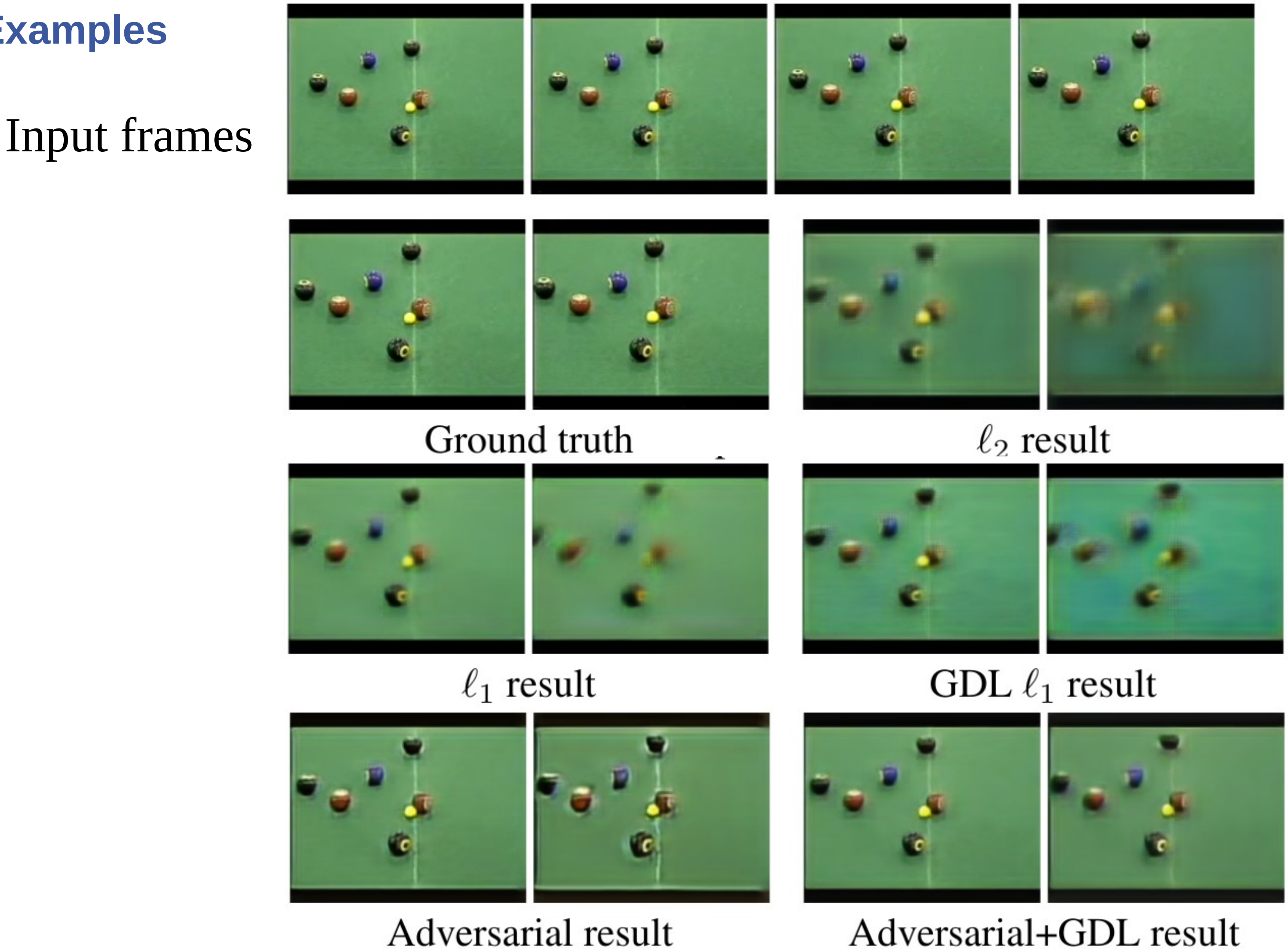
Examples

Input frames



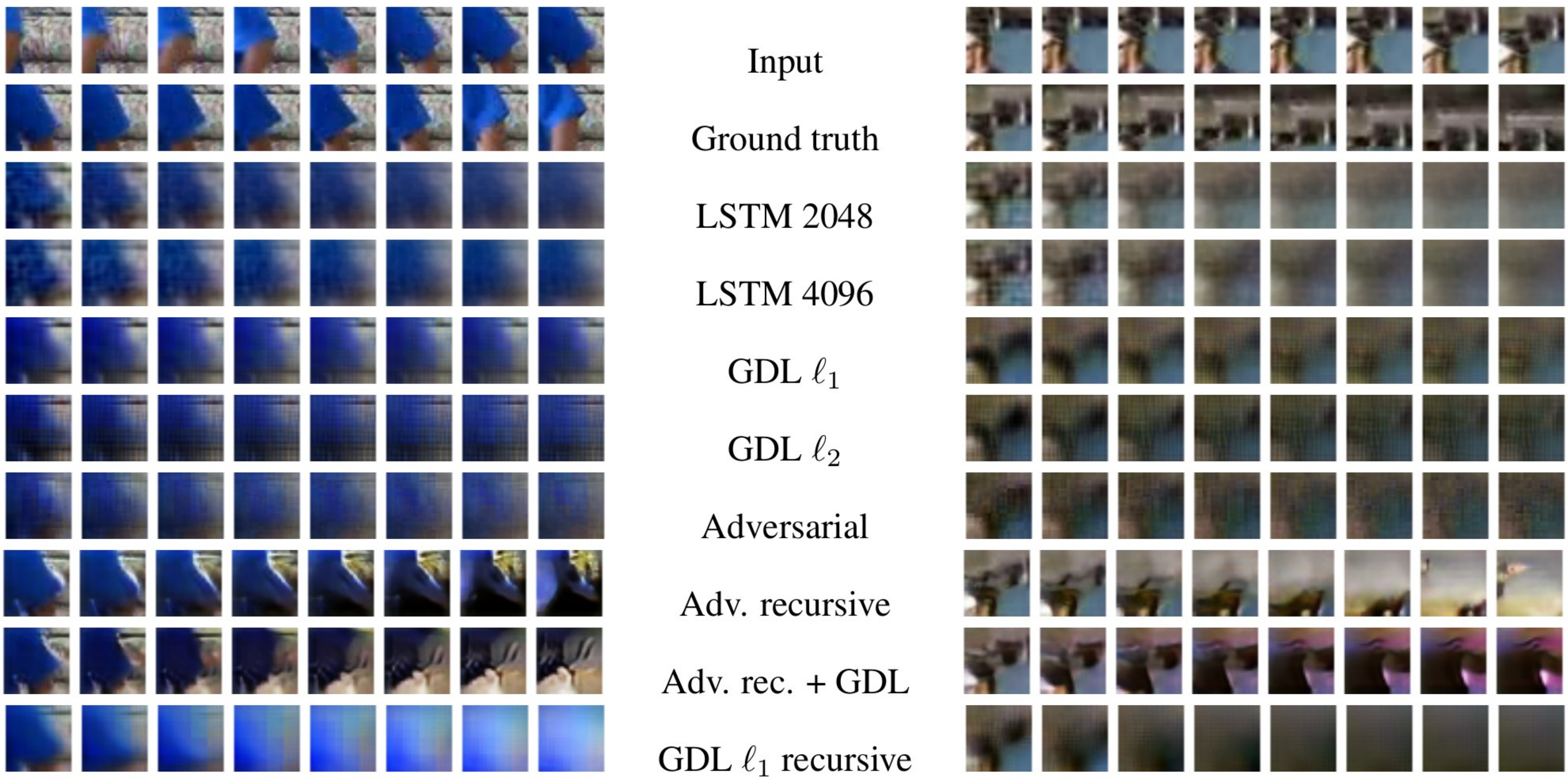
Multi-Scale ConvNet for Video Prediction

Examples



Multi-Scale ConvNet for Video Prediction

- **Comparison with [Srivastava et al. 2015] who used LSTM.**



Predictive Unsupervised Learning

- Some success with "adversarial training"
  - [Mathieu, Couprie, LeCun arXiv:1511:05440]
- But we are far from a complete solution.

PREDICTIVE LEARNING

Machine Intelligence
Will be very different from
Human Intelligence

What Will AI Be Like?

Human and animal behavior has basic "drives" hardwired by evolution

- Fight/flight, hunger, self-preservation, pain avoidance, desire for social interaction, etc…

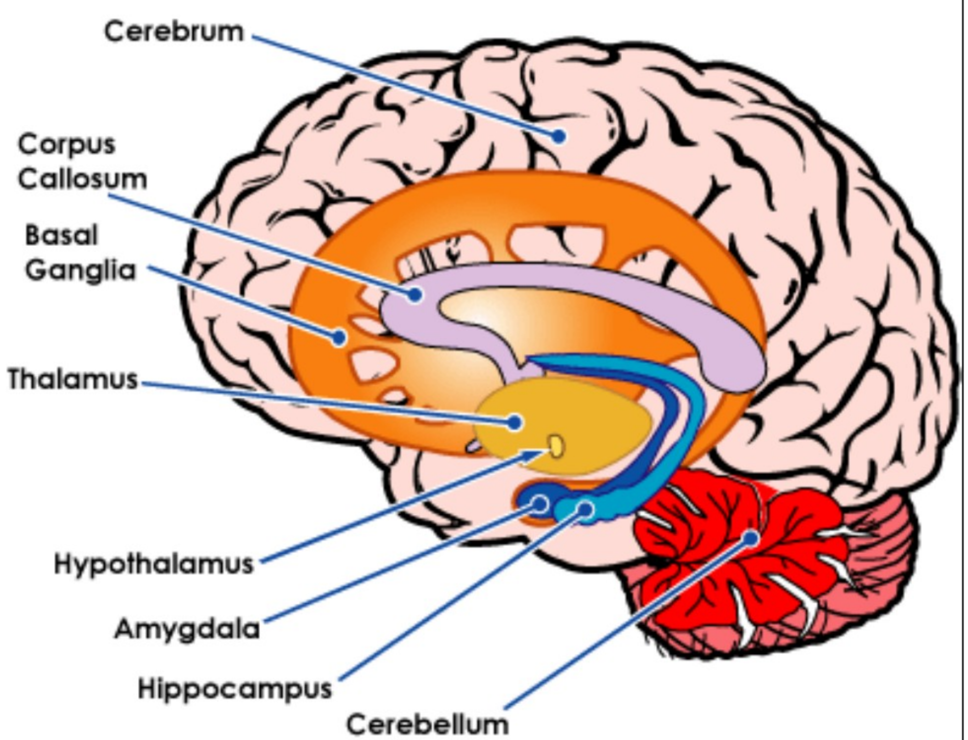
Humans do bad things to each other because of these drives (mostly)

- Violence under threat, desire for material resource and social power…

But an AI system will not have these drives unless we build them into it.

It's difficult for us to imagine an intelligent entity without these drives

- Although we have plenty of examples in the animal world



How do we align an AI's “moral” values to human values?

We will build a few basic, immutable, hardwired drives:

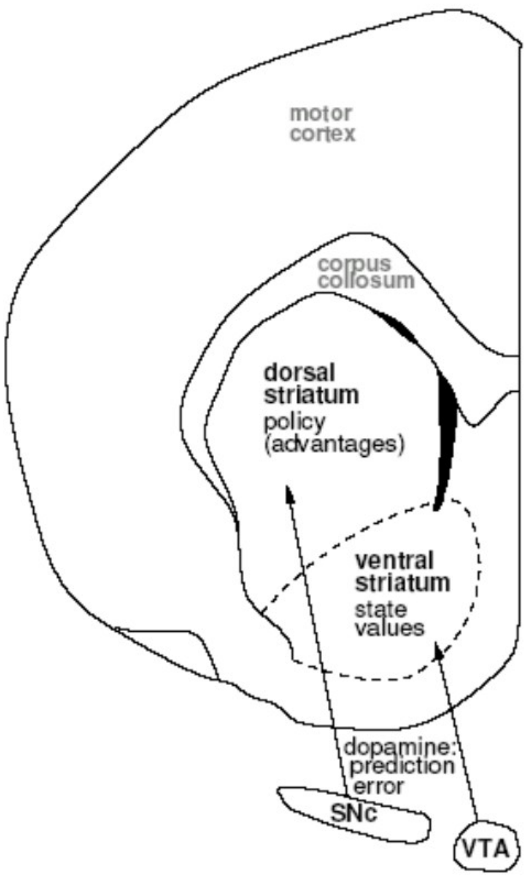
- To not hurt human and to interact with humans
- To crave positive feedback from trusted human trainers

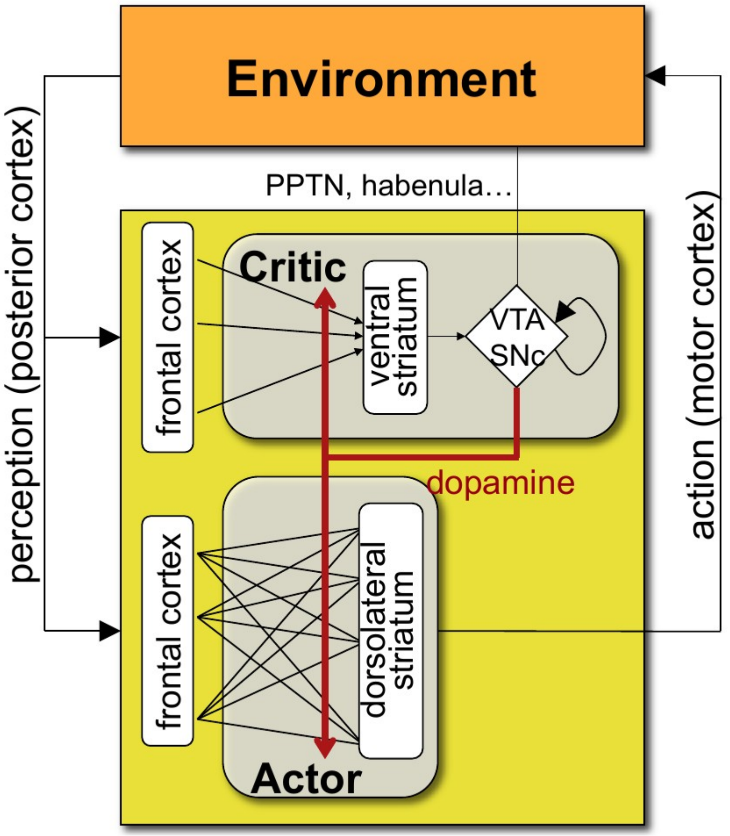
Human trainers will associate rewards with behaviors that make surrounding humans happy and comfortable.

This is how children (and social animals) learn how to behave in society.

Can we prevent unsafe AI?

Yes, the same way we prevent unsafe airplanes and cars.





[from Yael Niv]

How Will Human-Level AI Emerge

The emergence of human-level AI will not be an "event".

– It will be progressive

It will not happen in isolation

– No single entity has a monopoly on good ideas

Advancing AI is a scientific question right now, not a technological challenge

– Formulating unsupervised learning is our biggest challenge

Individual breakthroughs will be quickly reproduced

– AI research is a world-wide community

The majority of good ideas will come from Academia

– Even if the most impressive applications come from industry

It is important to distinguish intelligence from autonomy

– Most intelligent systems will not be autonomous.

Conclusions

- **Deep Learning is enabling a new wave of applications**
  - **Today:** Image recognition, video understanding: vision now works
  - **Today:** Better speech recognition: speech recognition now works
  - **Soon:** Better language understanding, dialog, and translation
- **Deep Learning and Convolutional Nets are being widely deployed**
  - **Today:** image understanding at Facebook, Google, Twitter, Microsoft…..
  - **Soon:** better auto-pilots for cars, medical image analysis, robot perception
- **We need hardware (and software) for embedded applications**
  - For smart cameras, mobile devices, cars, robots, toys….
- **But we are still far from building truly intelligent machines**
  - We need to integrate reasoning with deep learning
  - We need a good architecture for "episodic" (short-term) memory.
  - We need to find good principles for unsupervised learning

Merci