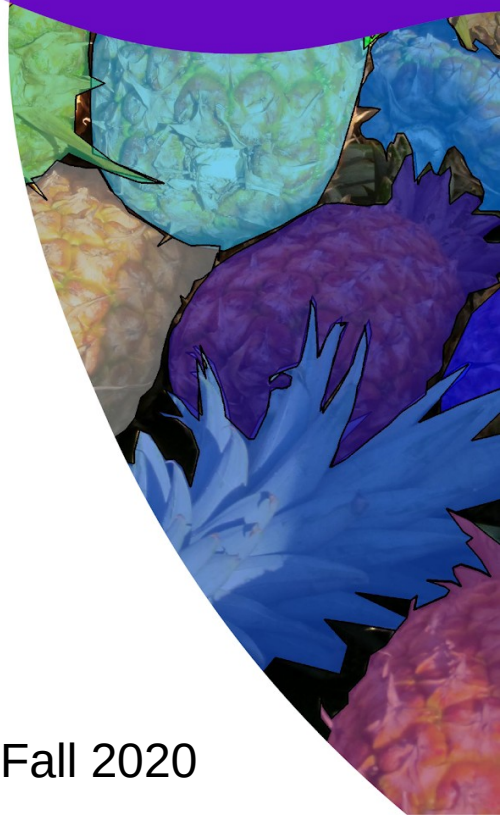
NEW YORK UNIVERSITY

Convolutional Networks

Yann LeCun
NYU - Courant Institute & Center for Data Science
Facebook AI Research

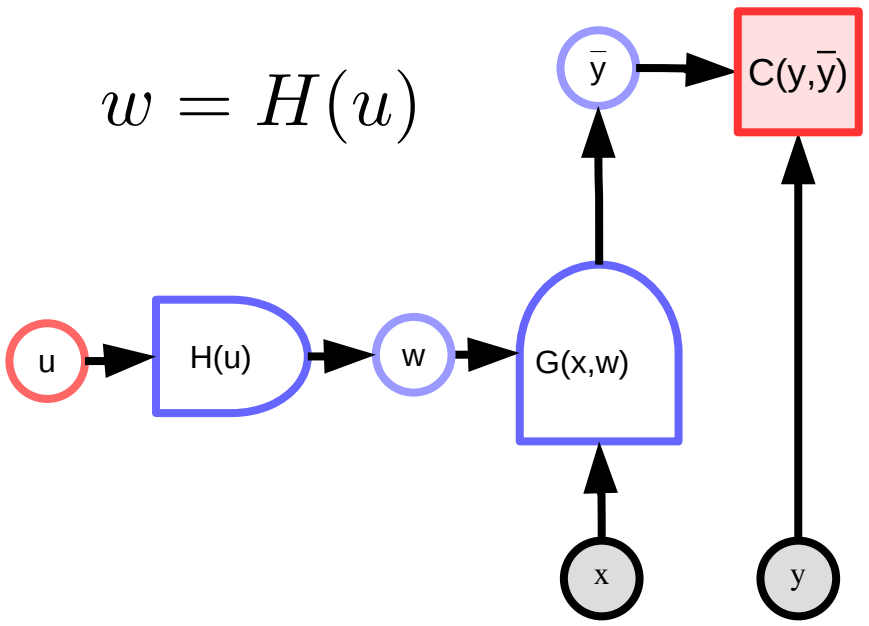


Deep Learning, NYU Fall 2020

Parameter transformations

- **When the parameter vector is the output of a function**

$$w = H(u)$$



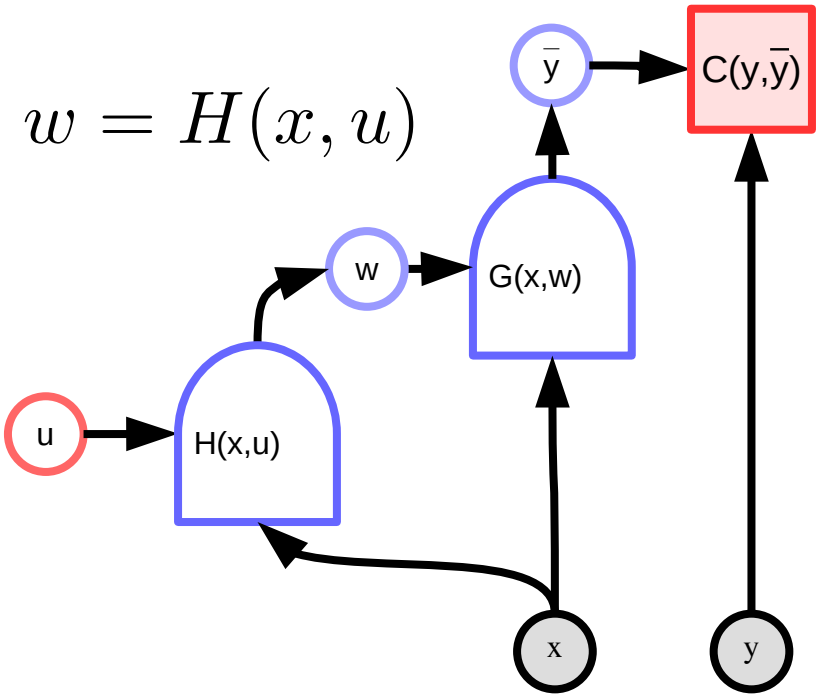
$$u \leftarrow u - \eta \frac{\partial H}{\partial u}^T \frac{\partial C}{\partial w}^T$$

$$w \leftarrow w - \eta \frac{\partial H}{\partial u} \frac{\partial H}{\partial u}^T \frac{\partial C}{\partial w}^T$$

$$[N_w \times N_u]\ [N_u \times N_w]\ [N_w \times 1]$$

“Hypernetwork”

$$w = H(x, u)$$



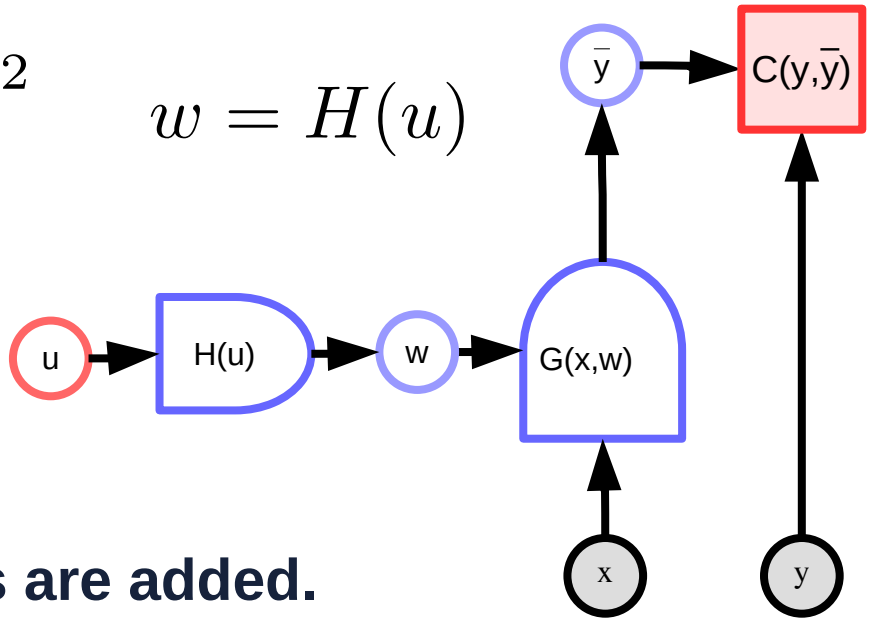
- **When the parameter vector is the output of another network H(x,u)**
- **The weights of network G(x,w) are dynamically configured by network H(x,u)**
- **The concept is very powerful**
  - The idea is very old

Simple parameter transform: weight sharing

- **Function H(u) replicates one component of u into multiple components of w**
  - $w_1, = w_2 = u_1 \quad w_3 = w_4 = u_2$

$$w = H(u)$$

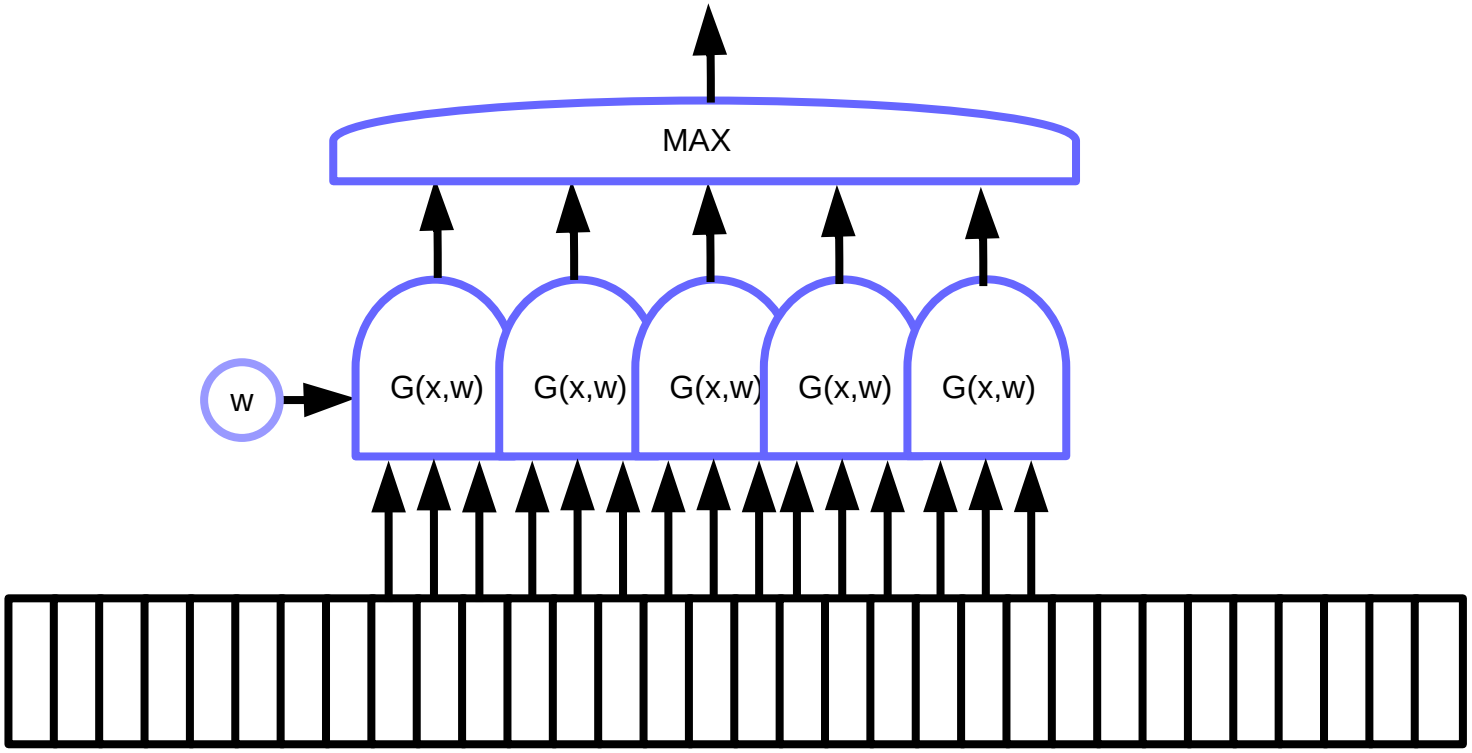
- **H is like a "Y" branch.**
  - Gradients are summed in the backprop



- **The gradients w.r.t. shared parameters are added.**

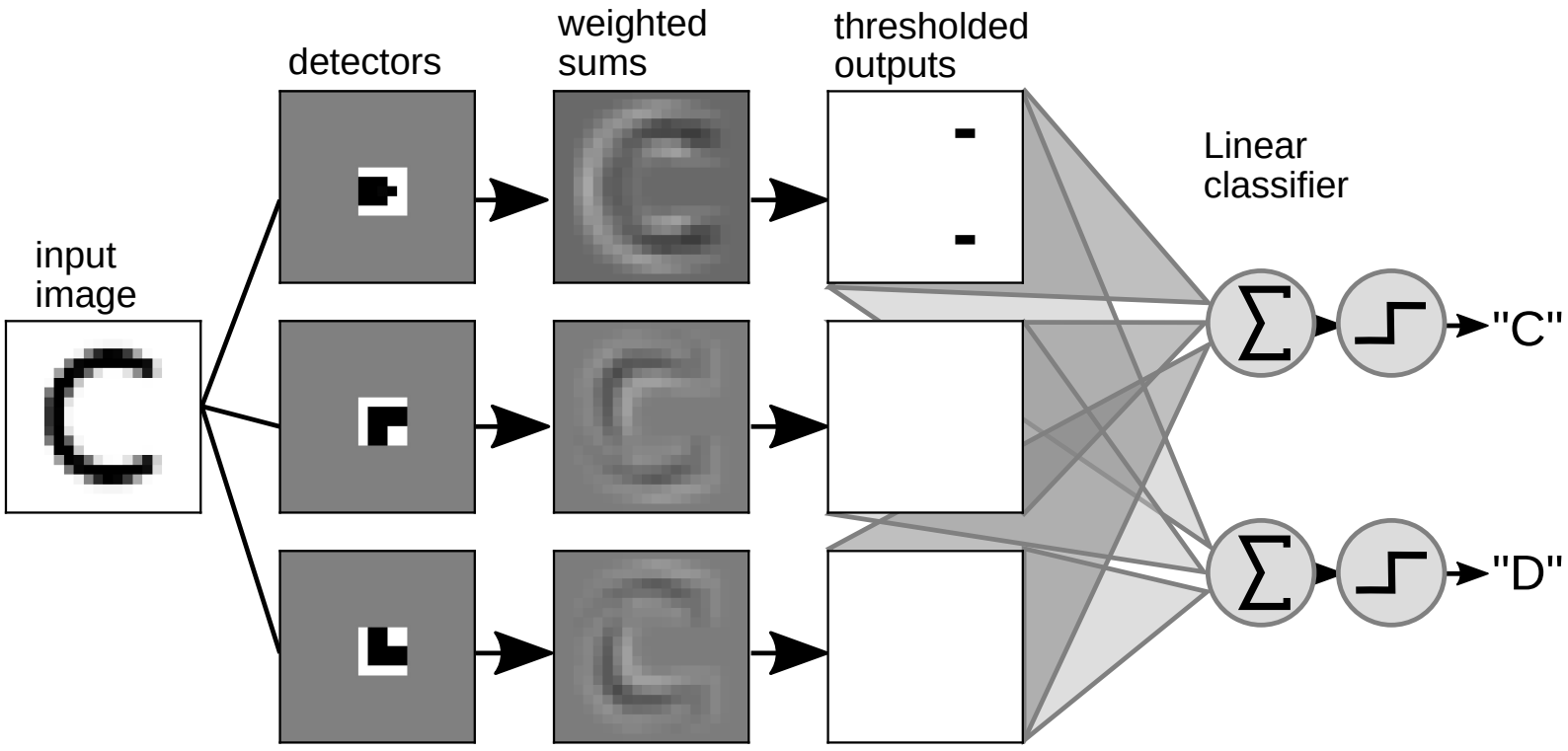
Shared Weights for Motif Detection

- **Detecting motifs anywhere on an input**



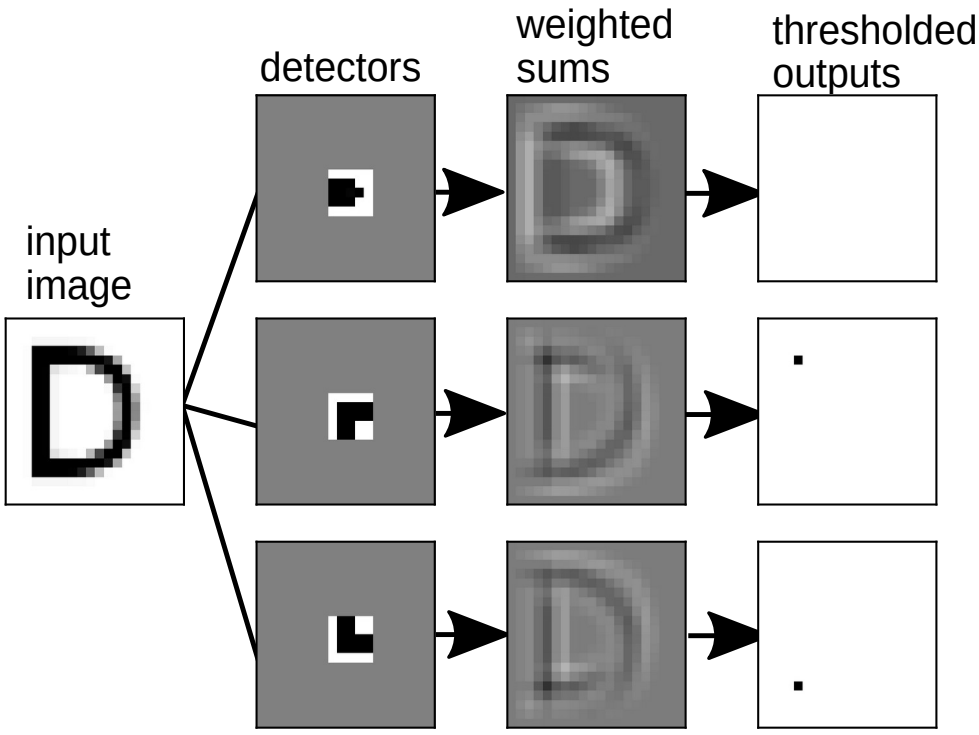
Detecting Motifs in Images

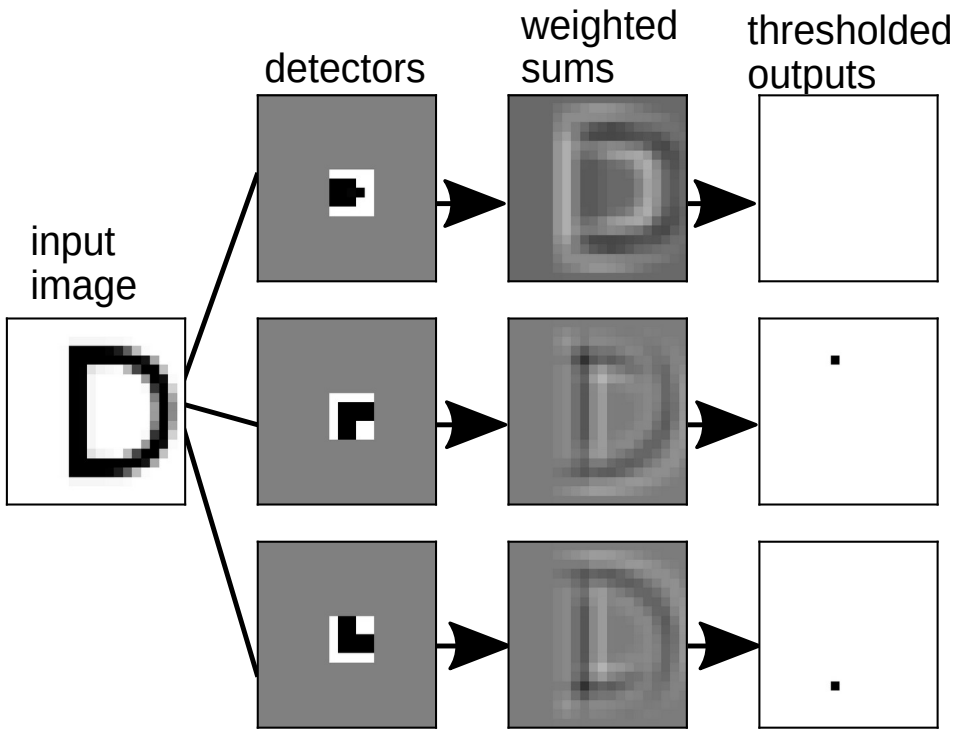
- **Swipe “templates” over the image to detect motifs**



Detecting Motifs in Images

- **Shift invariance**





Discrete Convolution (or cross-correlation)

- **Definition**
  - convolution

$$y_i = \sum_j w_j x_{i-j}$$

- **In practice**
  - Cross-correlation

$$y_i = \sum_j w_j x_{i+j}$$

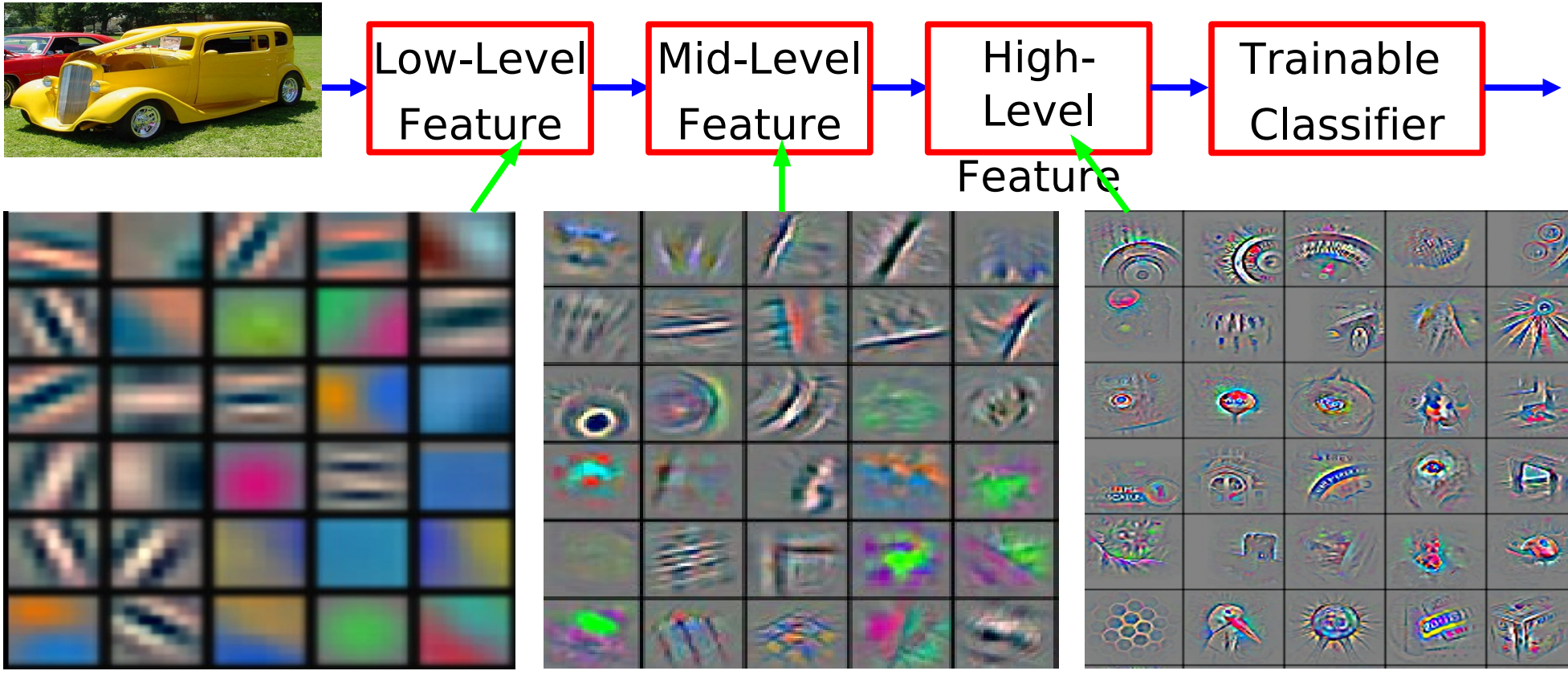
- **In 2D**

$$y_{ij} = \sum_{kl} w_{kl} x_{i+k,j+l}$$

Deep Learning = Learning Hierarchical Representations

Y. LeCun

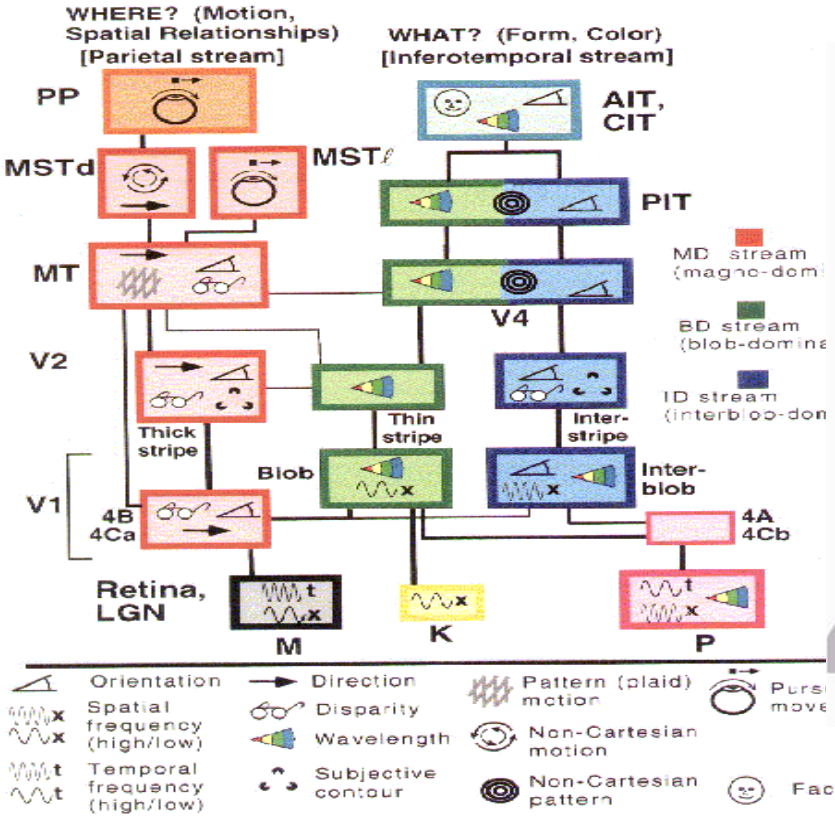
- It's **deep** if it has **more than one stage** of non-linear feature transformation

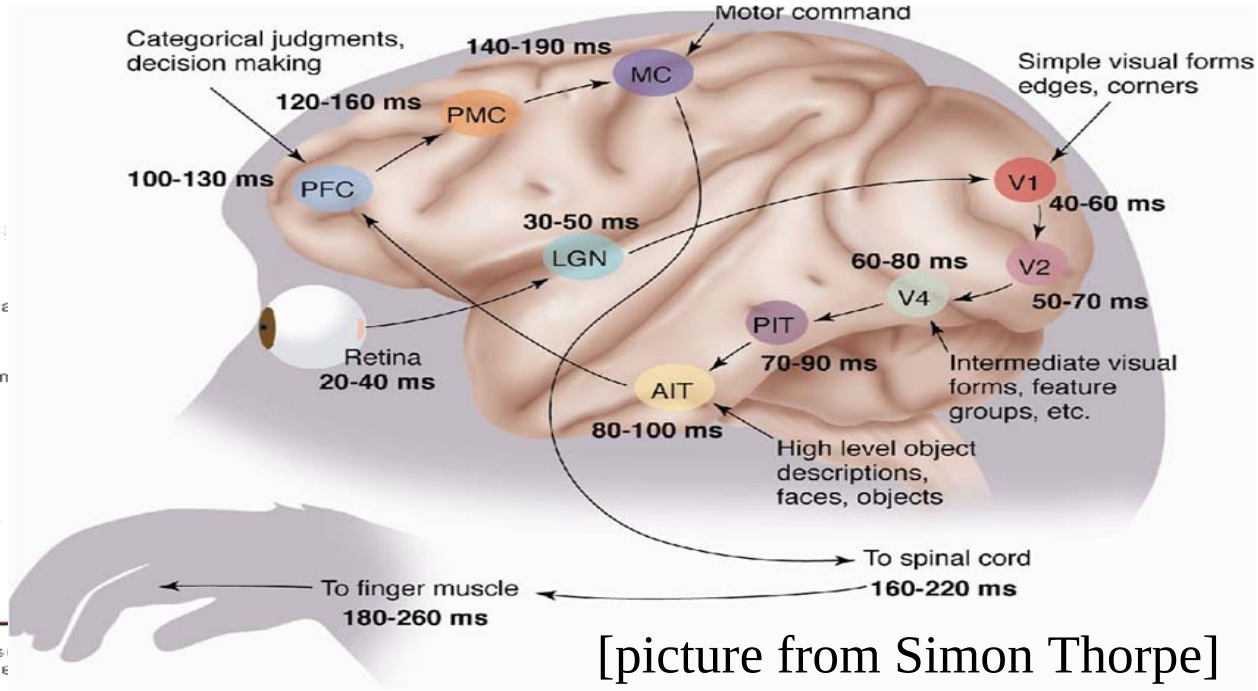


Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

How does the brain interprets images?

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





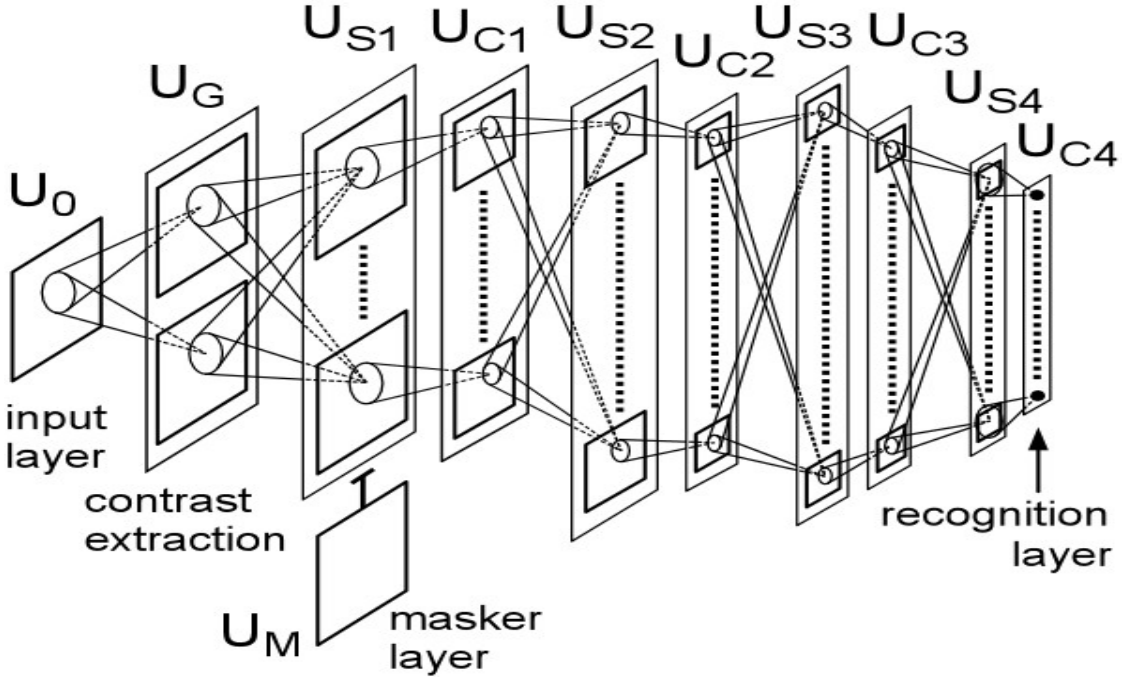
[picture from Simon Thorpe]

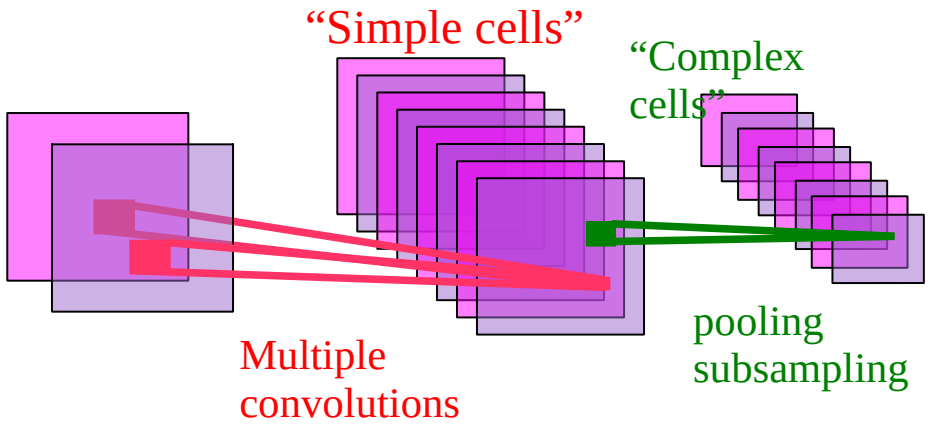
[Gallant & Van Essen]

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

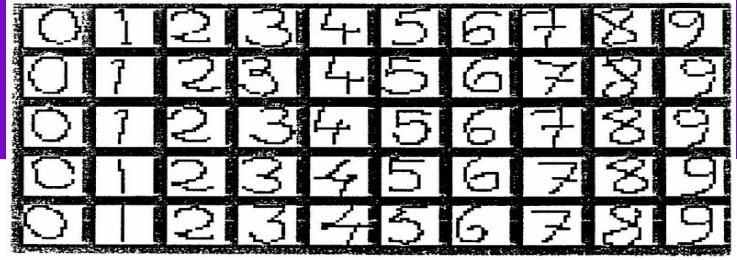
d.



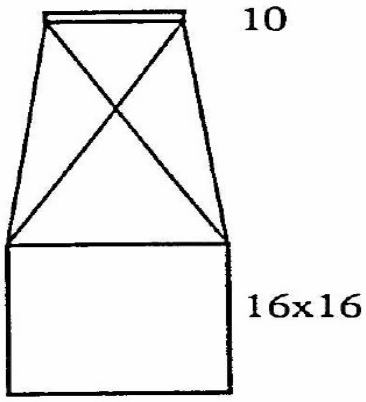


[Fukushima 1982][LeCun 1989, 1998],[Riesenhuber 1999].....

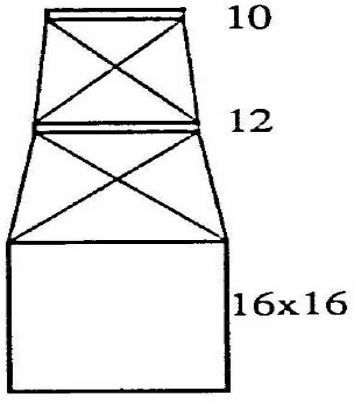
First ConvNets (U Toronto)[LeCun 88, 89]



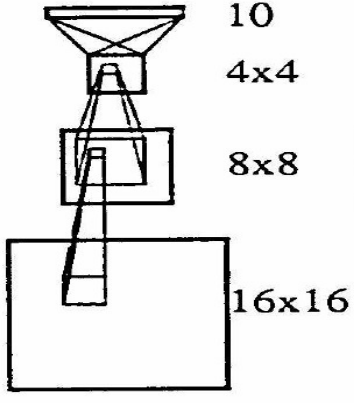
- **Trained with Backprop. 320 examples.**



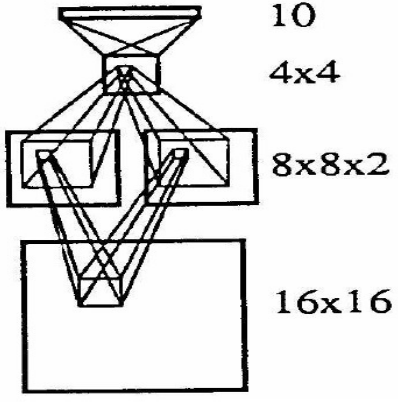
Single layer



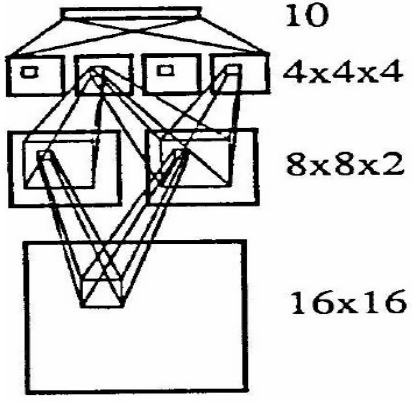
Two layers FC



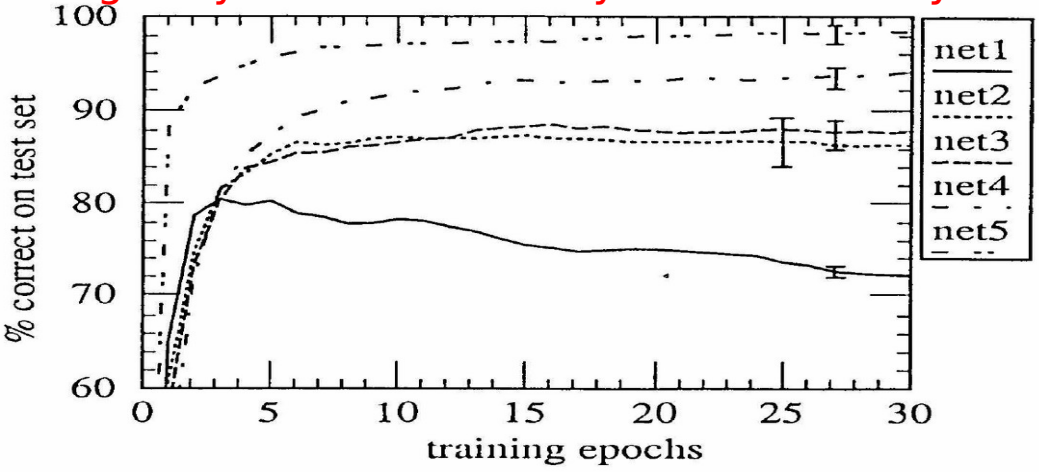
locally connected



Shared weights



Shared weights

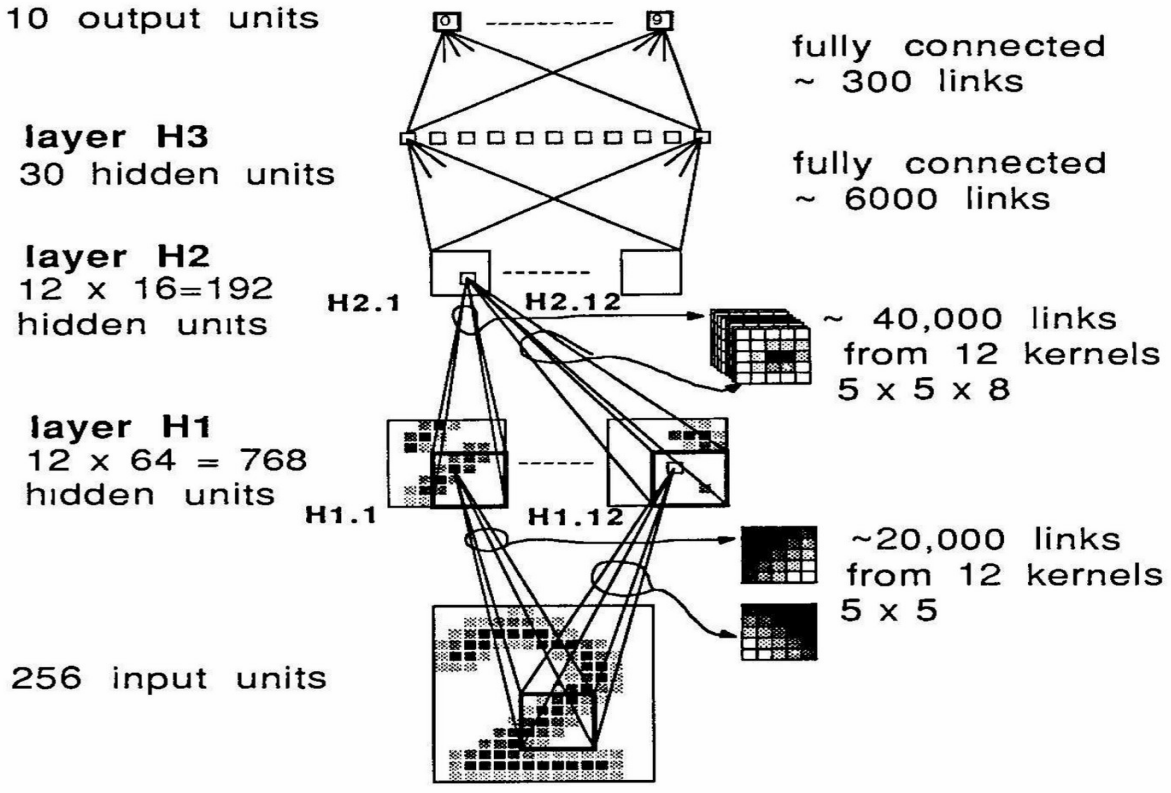


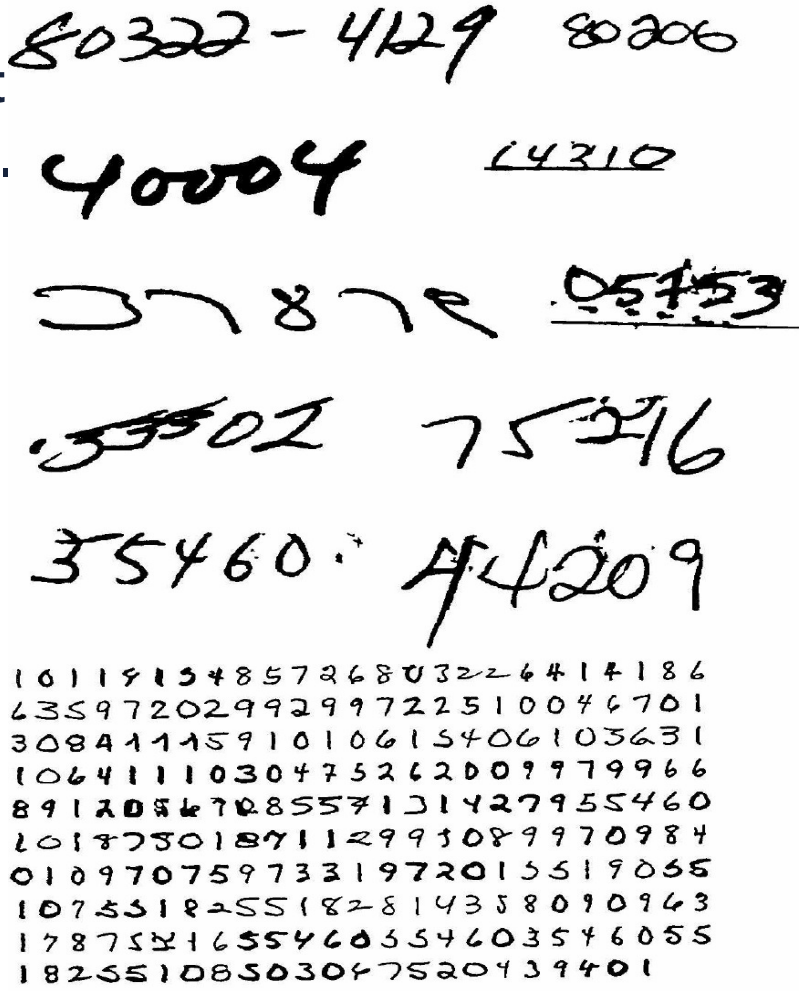
- Convolutions with stride (subsampling)
- No separate pooling layers

network architecture	links	weights	performance
single layer network	2570	2570	80 %
two layer network	3240	3240	87 %
locally connected	1226	1226	88.5 %
constrained network	2266	1132	94 %
constrained network 2	5194	1060	98.4 %

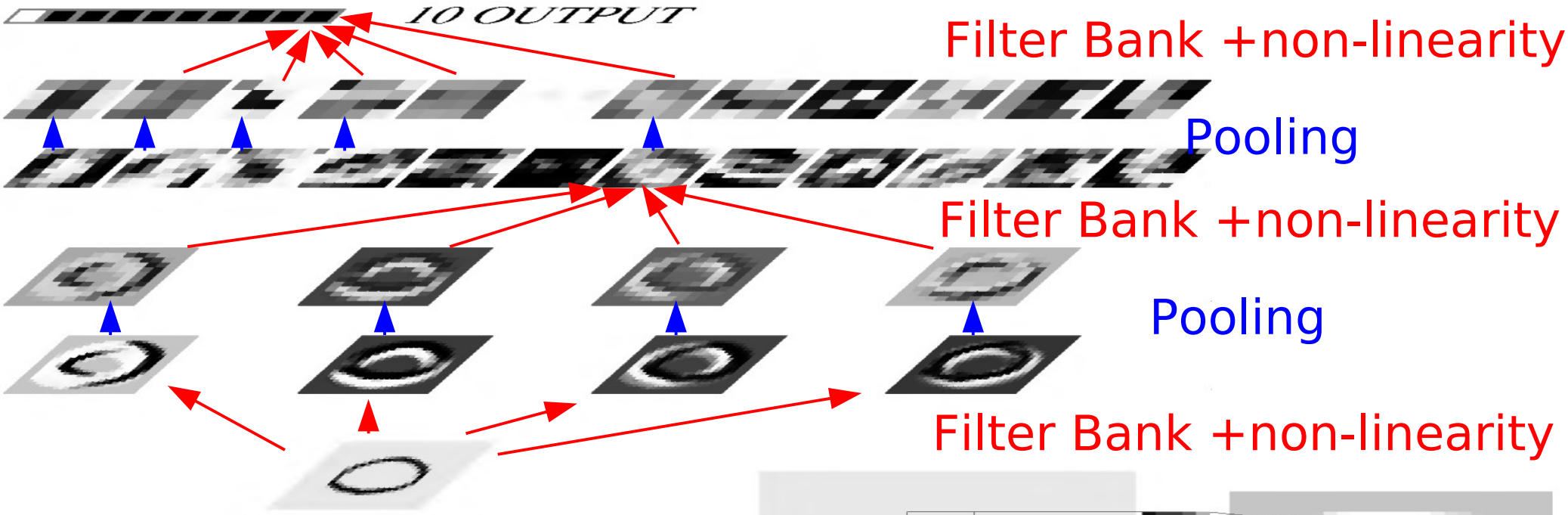
First “Real” ConvNets at Bell Labs [LeCun et al 89]

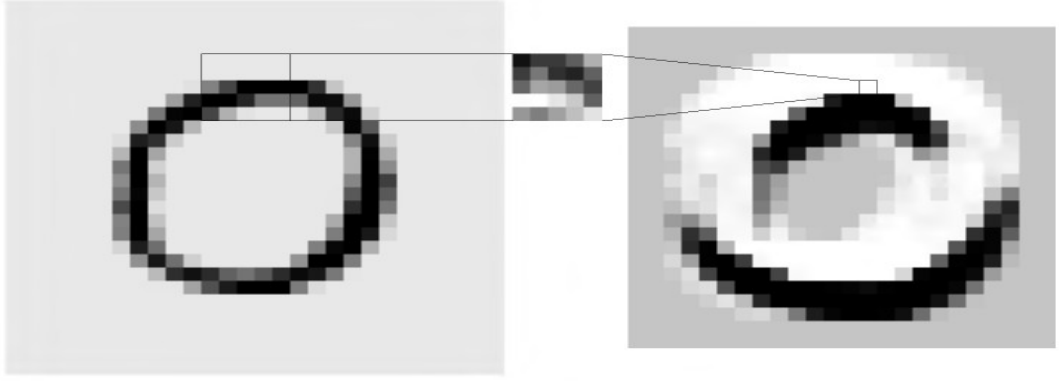
- **Trained with Backprop.**
- **USPS Zipcode digits: 7300 training, 2000 test**
- **Convolution with stride. No separate pooling.**





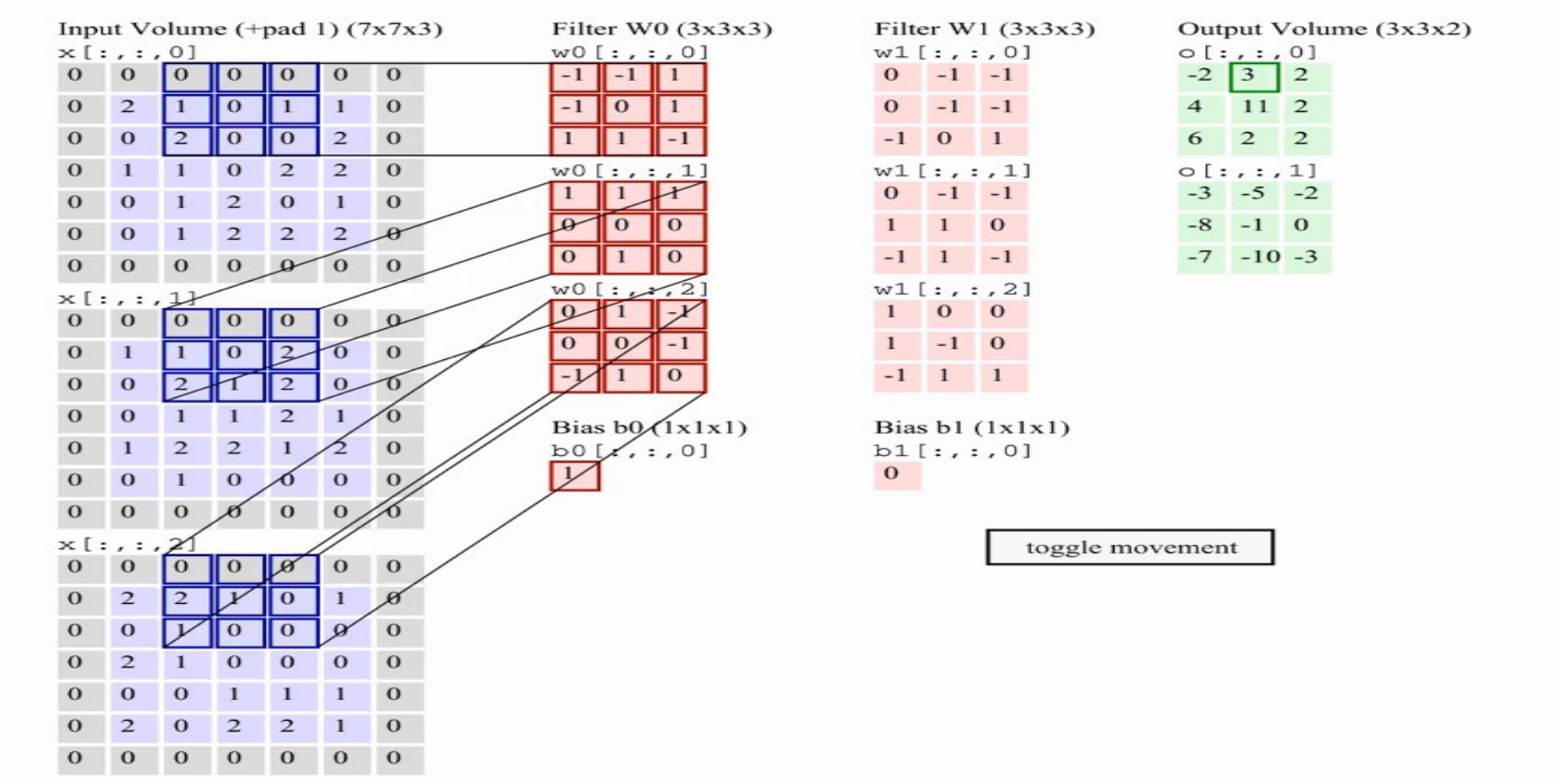
Convolutional Network Architecture





- **[LeCun et al. NIPS 1989]**

Multiple Convolutions

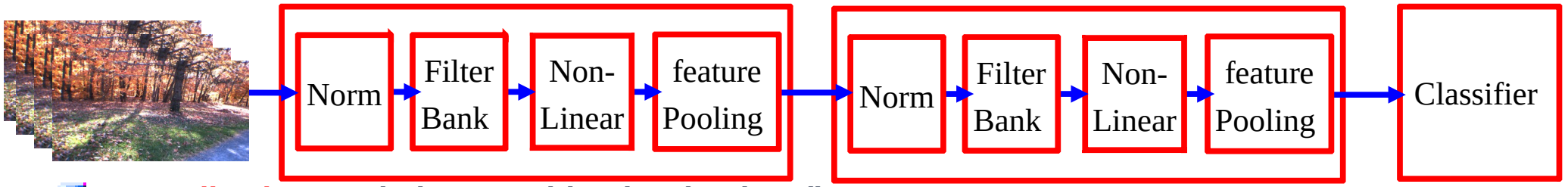


Animation: Andrej Karpathy http://cs231n.github.io/convolutional-networks/

Convolutional Network (vintage 1990)

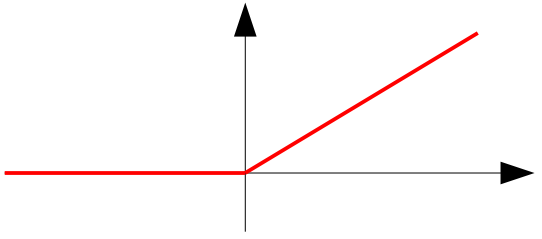
- **Filters-tanh → pooling → filters-tanh → pooling → filters-tanh**

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.

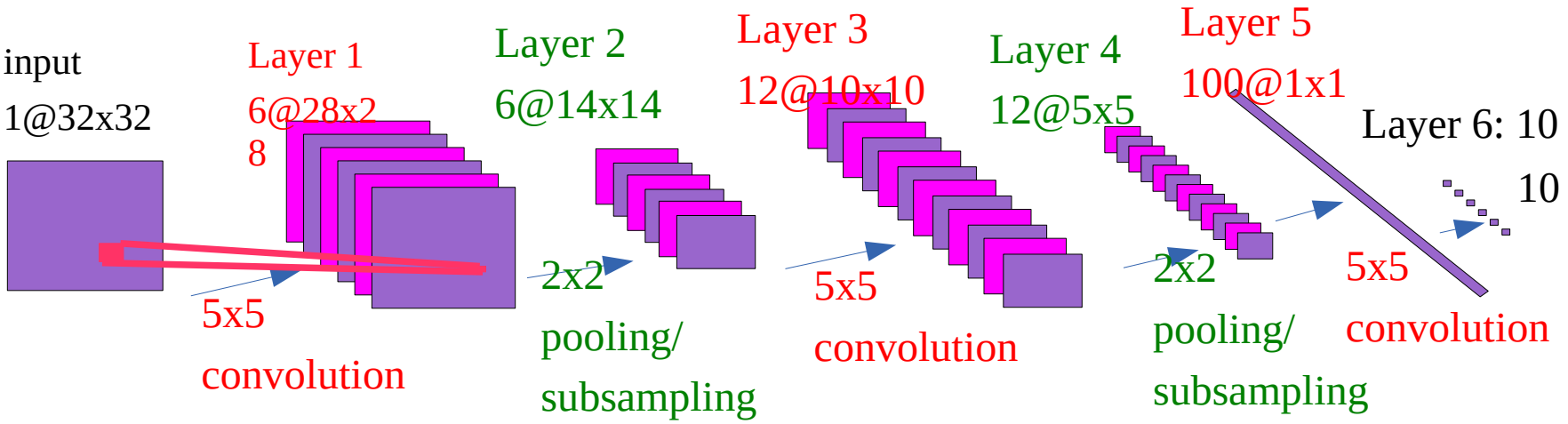
$$MAX: Max_i(X_i); \quad L_p: \sqrt[p]{X_i^p}; \quad PROB: \frac{1}{b}\log\left(\sum_i e^{bX_i}\right)$$

LeNet5

- **Simple ConvNet**
- **for MNIST**
- **[LeCun 1998]**

- **PyTorch code ------→**
  - (slightly different net)

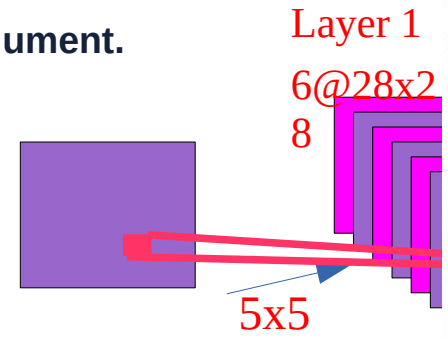
```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(1, 20, 5, 1)
        self.conv2 = nn.Conv2d(20, 50, 5, 1)
        self.fc1 = nn.Linear(4*4*50, 500)
        self.fc2 = nn.Linear(500, 10)

    def forward(self, x):
        x = F.relu(self.conv1(x))
        x = F.max_pool2d(x, 2, 2)
        x = F.relu(self.conv2(x))
        x = F.max_pool2d(x, 2, 2)
        x = x.view(-1, 4*4*50)
        x = F.relu(self.fc1(x))
        x = self.fc2(x)
        return F.log_softmax(x, dim=1)
```



LeNet5

- Simple ConvNet
- for MNIST
- [LeCun 1998]

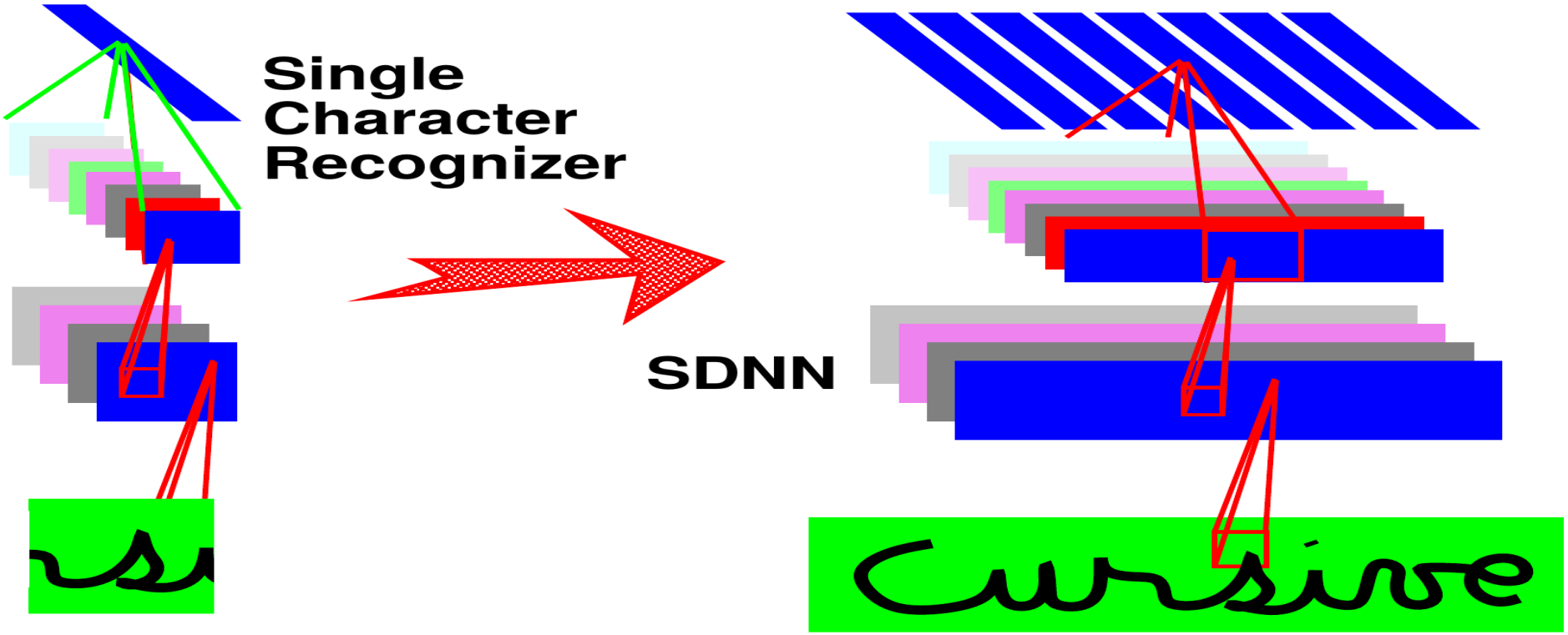
- PyTorch code --→ github.com/activatedgeek/LeNet-5
- nn.sequential() with ordered dictionary argument.



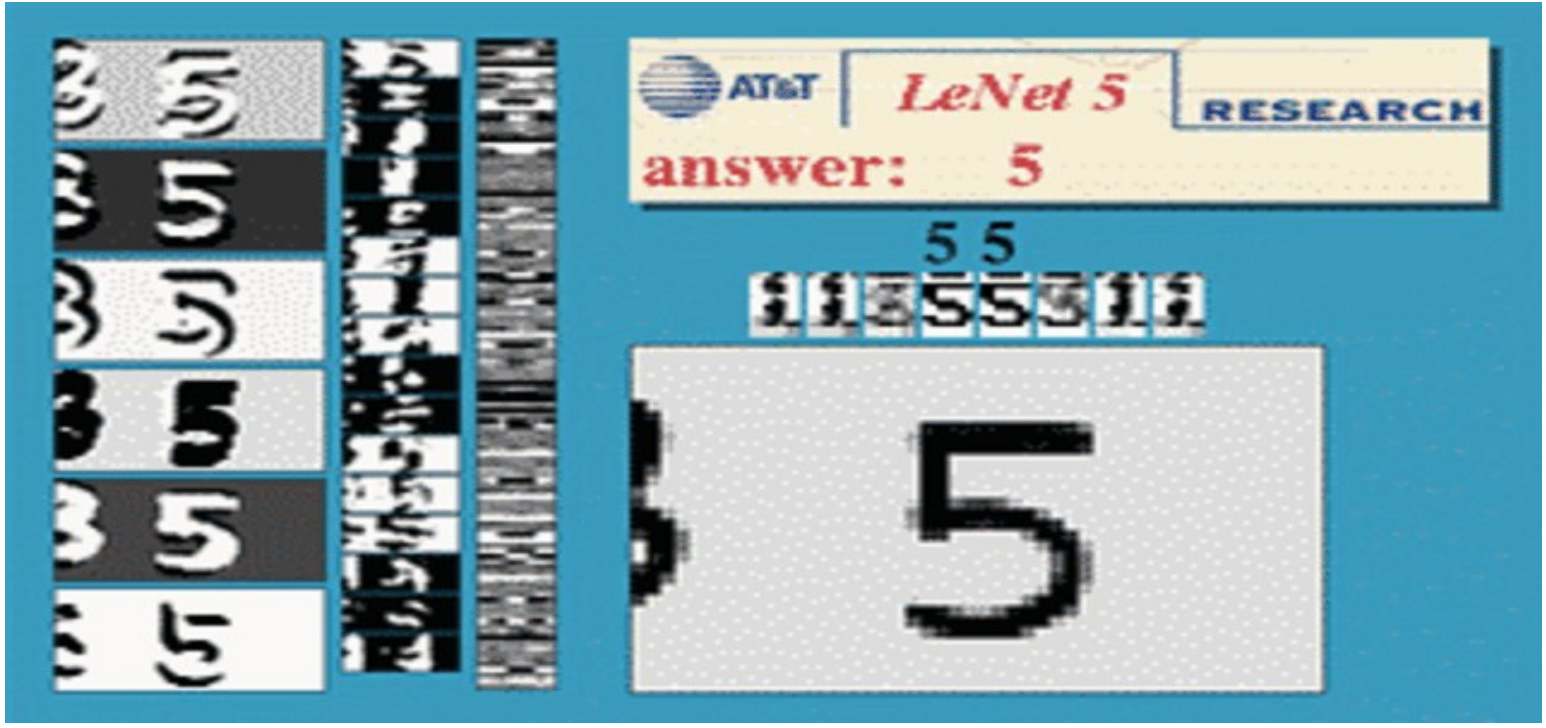

```
19      def __init__(self):
20          super(LeNet5, self).__init__()
21
22          self.convnet = nn.Sequential(OrderedDict([
23              ('c1', nn.Conv2d(1, 6, kernel_size=(5, 5))),
24              ('relu1', nn.ReLU()),
25              ('s2', nn.MaxPool2d(kernel_size=(2, 2), stride=2)),
26              ('c3', nn.Conv2d(6, 16, kernel_size=(5, 5))),
27              ('relu3', nn.ReLU()),
28              ('s4', nn.MaxPool2d(kernel_size=(2, 2), stride=2)),
29              ('c5', nn.Conv2d(16, 120, kernel_size=(5, 5))),
30              ('relu5', nn.ReLU())
31          ]))
32
33          self.fc = nn.Sequential(OrderedDict([
34              ('f6', nn.Linear(120, 84)),
35              ('relu6', nn.ReLU()),
36              ('f7', nn.Linear(84, 10)),
37              ('sig7', nn.LogSoftmax(dim=-1))
38          ]))
39
40      def forward(self, img):
41          output = self.convnet(img)
42          output = output.view(img.size(0), -1)
43          output = self.fc(output)
44          return output
```

Multiple Character Recognition [Matan et al 1992]

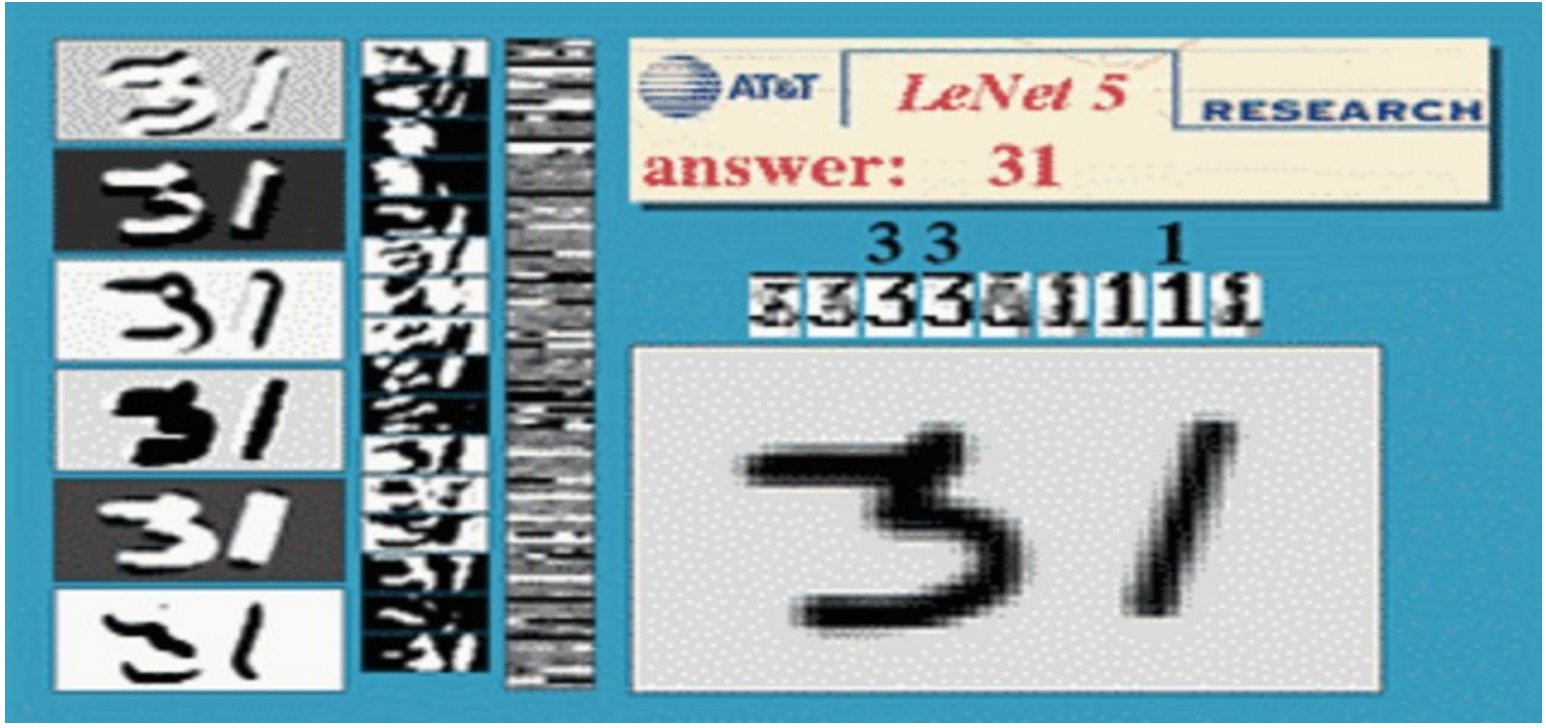
- Every layer is a convolution



Sliding Window ConvNet + Weighted Finite-State Machine



Sliding Window ConvNet + Weighted FSM



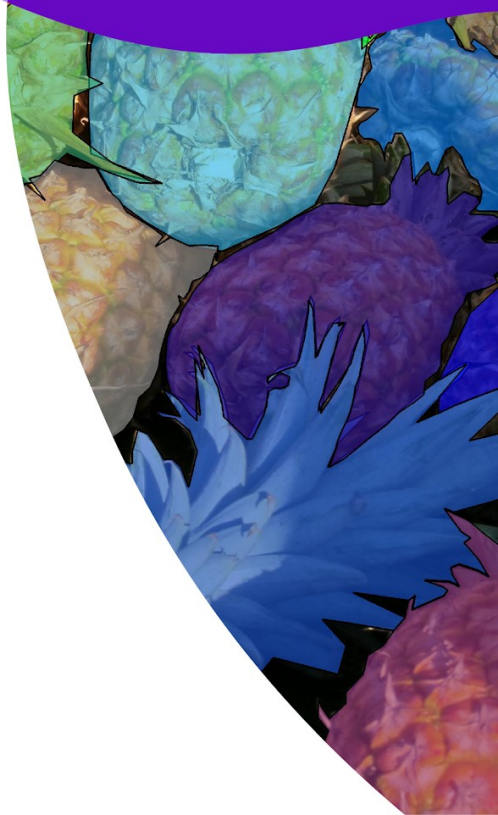
What are ConvNets Good For

- **Signals that comes to you in the form of (multidimensional) arrays.**
- **Signals that have strong local correlations**
- **Signals where features can appear anywhere**
- **Signals in which objects are invariant to translations and distortions.**
- **1D ConvNets: sequential signals, text**
  - Text, music, audio, speech, time series.
- **2D ConvNets: images, time-frequency representations (speech and audio)**
  - Object detection, localization, recognition
- **3D ConvNets: video, volumetric images, tomography images**
  - Video recognition / understanding
  - Biomedical image analysis
  - Hyperspectral image analysis

NEW YORK UNIVERSITY

What can Deep Architectures Do?

A preview and a bit of history



Convolutional Network (LeNet5, vintage 1990)

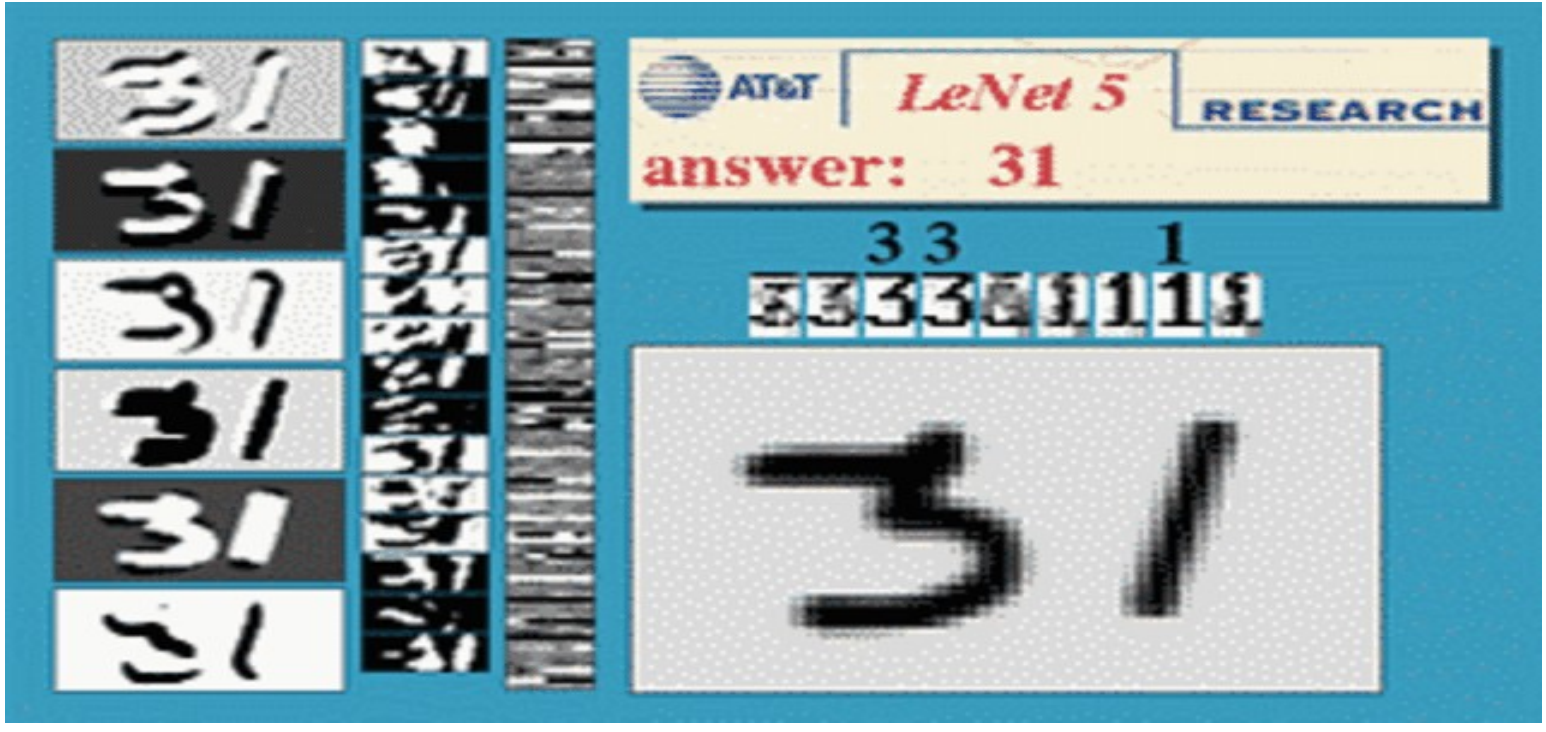
- **Filters-tanh → pooling → filters-tanh → pooling → filters-tanh**

LeNet character recognition demo 1992

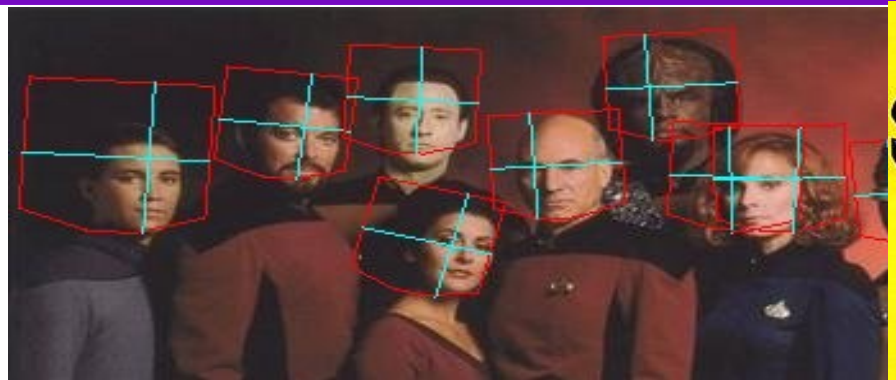
- **Running on an AT&T DSP32C (floating-point DSP, 20 MFLOPS)**

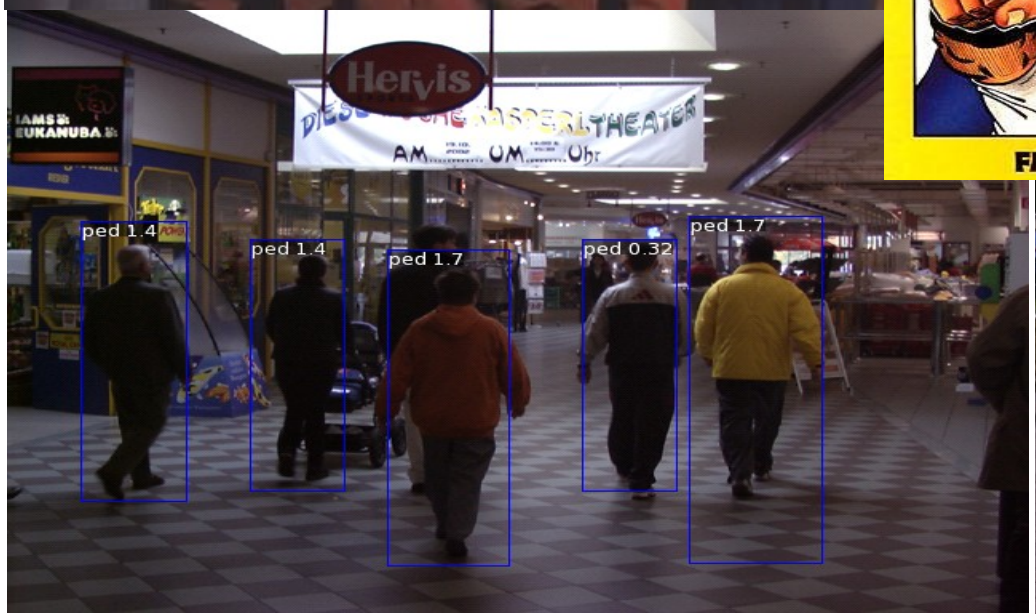
ConvNets can recognize multiple objects

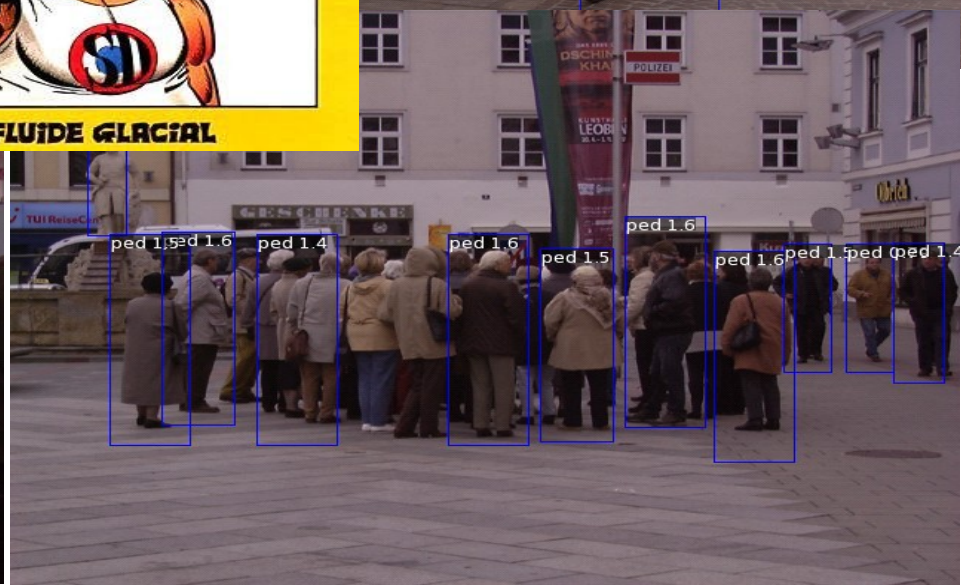
- **All layers are convolutional**
- **Networks performs simultaneous segmentation and recognition**
  - [LeCun, Bottou, Bengio, Haffner, Proc IEEE 1998]



Face & Pedestrian Detection with ConvNets (1993-2005)

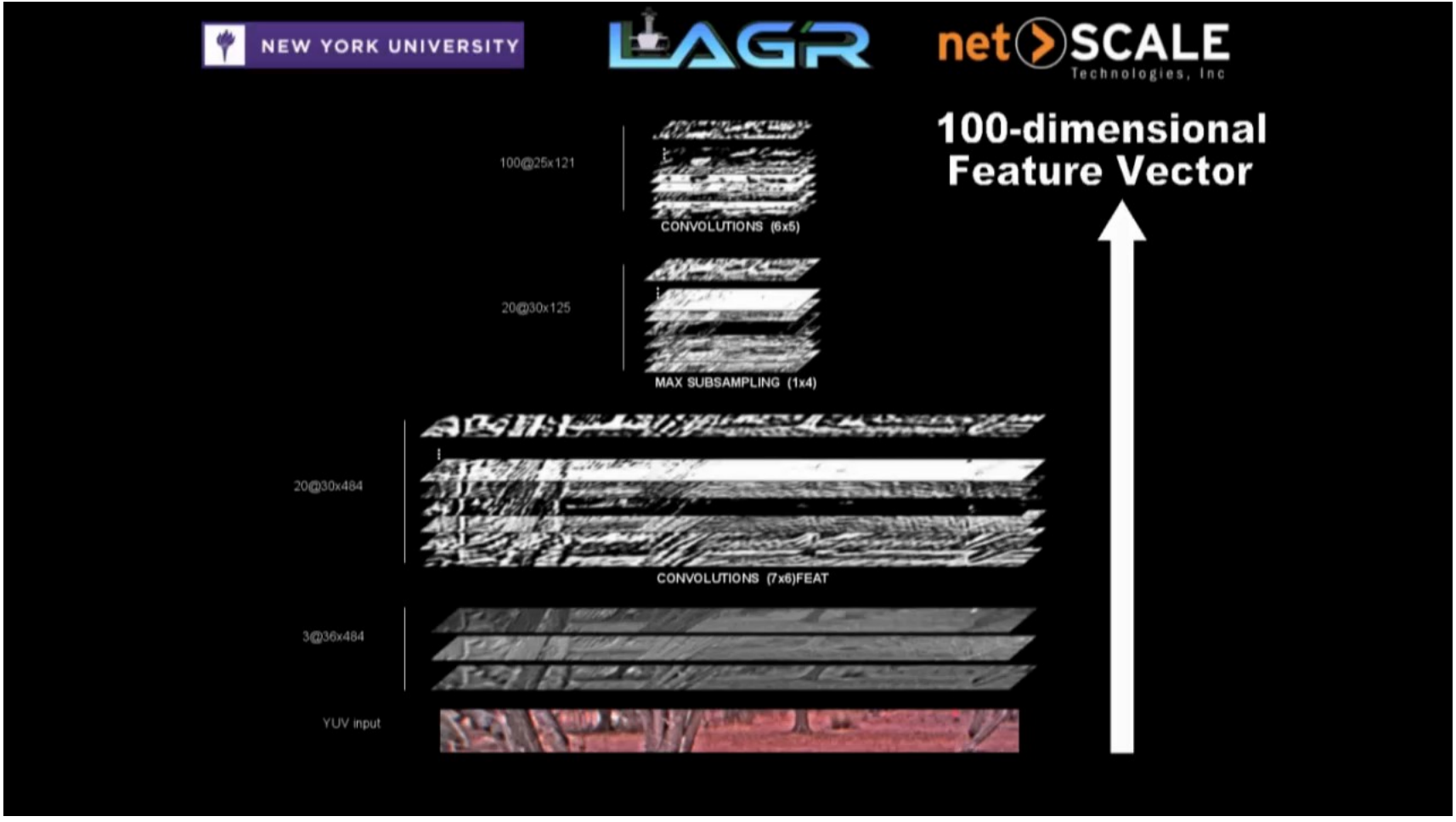






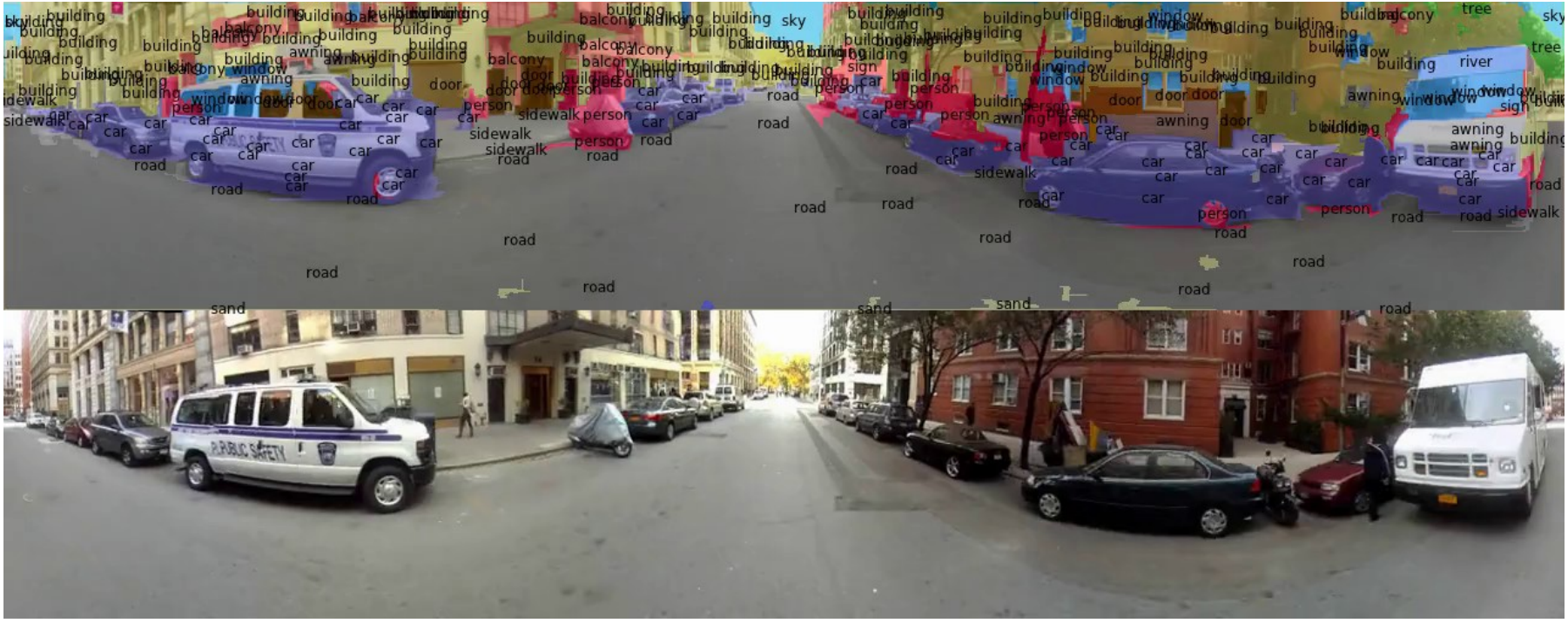
[Osadchy,Miller LeCun JMLR 2007],[Kavukcuoglu et al. NIPS 2010] [Sermanet et al. CVPR 2013]

Training a Robot to Drive Itself in Nature [Hadsell 2009]



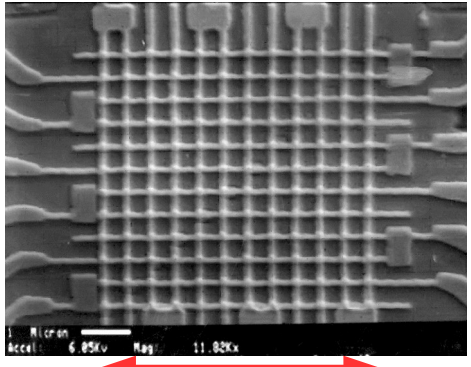
Semantic Segmentation with ConvNets [Farabet 2012]

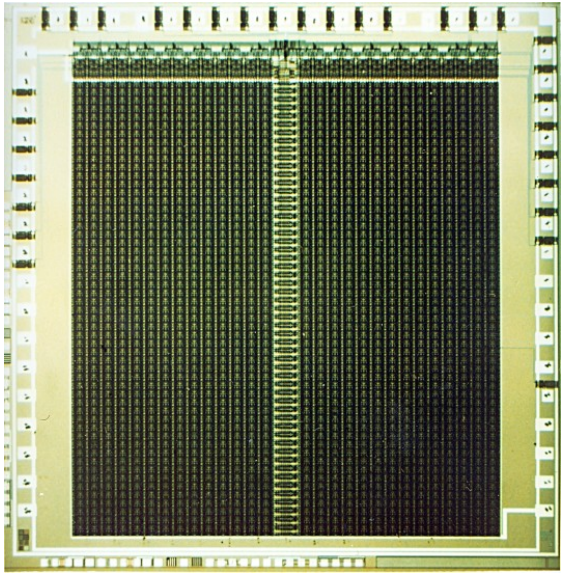
- **33 categories**

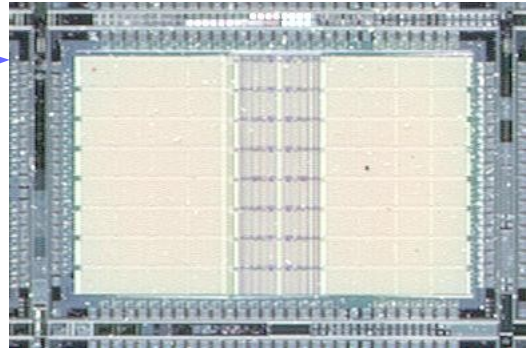


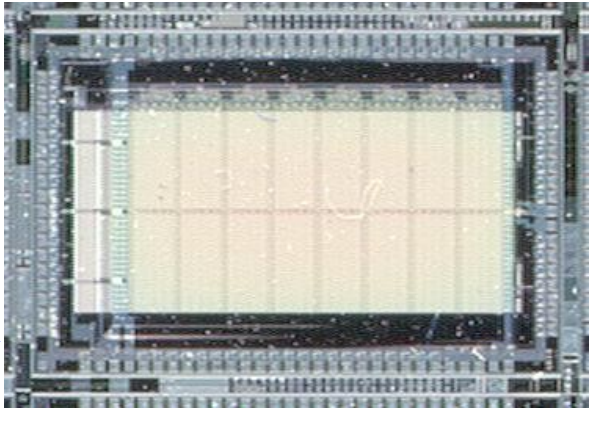
1986-1996 Neural Net Hardware at Bell Labs, Holmdel

- **1986: 12x12 resistor array**
  - Fixed resistor values
  - E-beam lithography: 6x6microns
- **1988: 54x54 neural net**
  - Programmable ternary weights
  - On-chip amplifiers and I/O
- **1991: Net32k: 256x128 net**
  - Programmable ternary weights
  - 320GOPS, 1-bit convolver.
- **1992: ANNA: 64x64 net**
  - ConvNet accelerator: 4GOPS
  - 6-bit weights, 3-bit activations



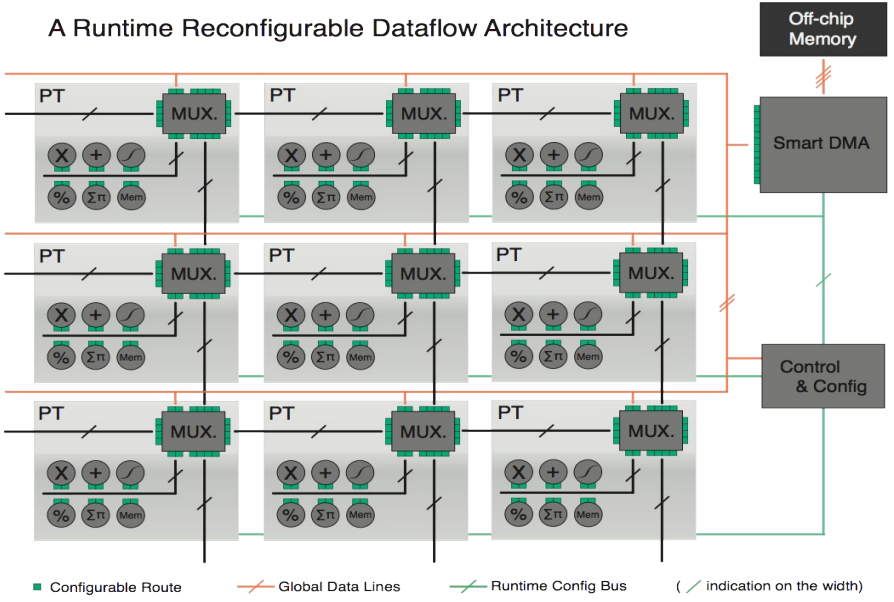




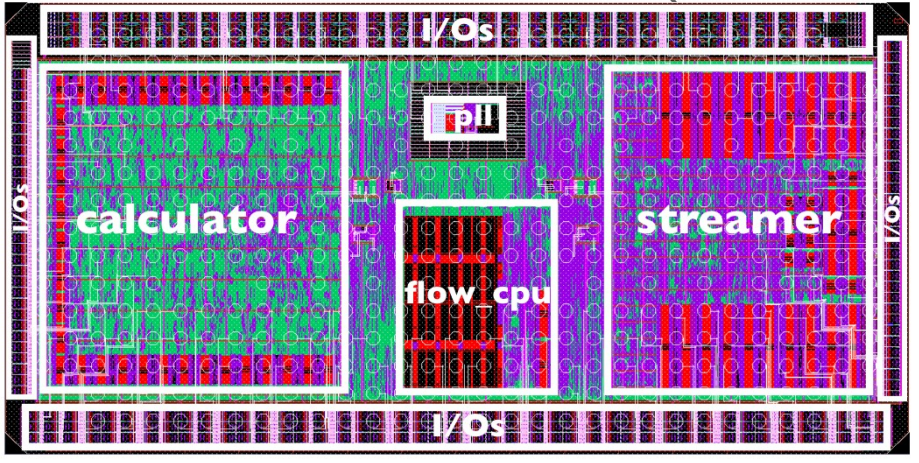


FPGA ConvNet Accelerator: NewFlow [Farabet 2011]

- **NeuFlow: Reconfigurable Dataflow architecture**
  - Implemented on Xilinx Virtex6 FPGA
  - 20 configurable tiles. 150GOPS, 10 Watts
  - Semantic Segmentation: 20 frames/sec at 320x240
  - **Exploits the structure of convolutions**

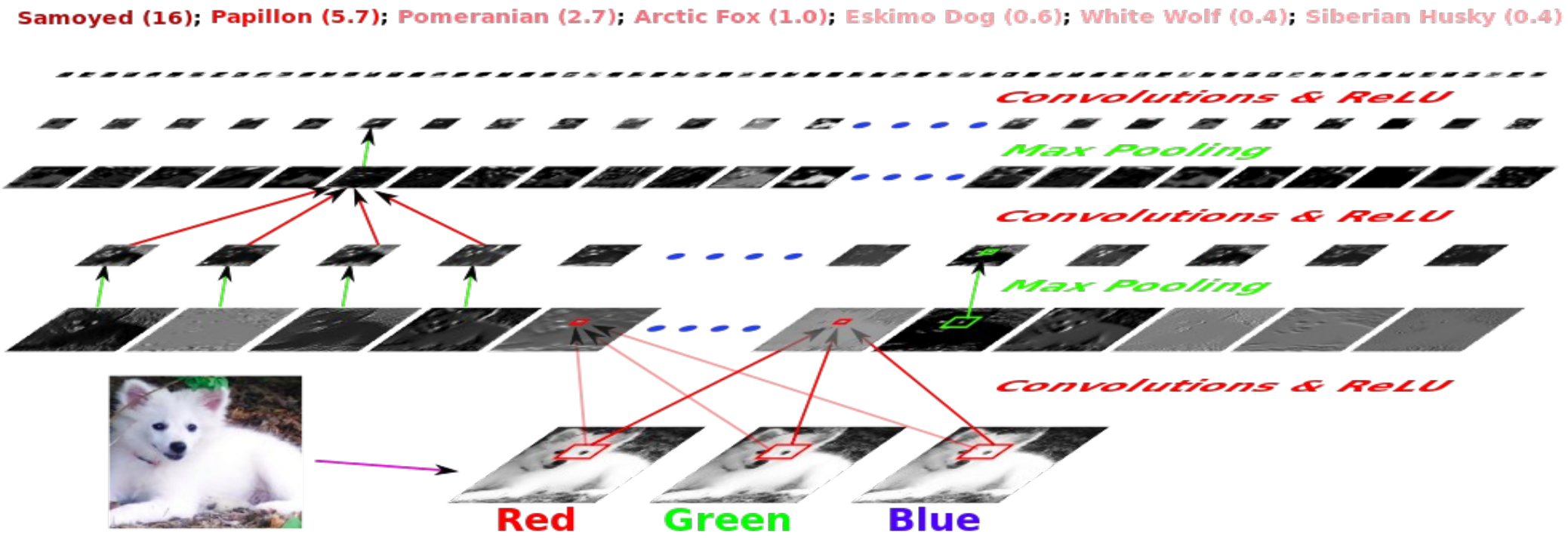


- **NeuFlow ASIC** [Pham 2012]
  - 150GOPS, 0.5 Watts (simulated)



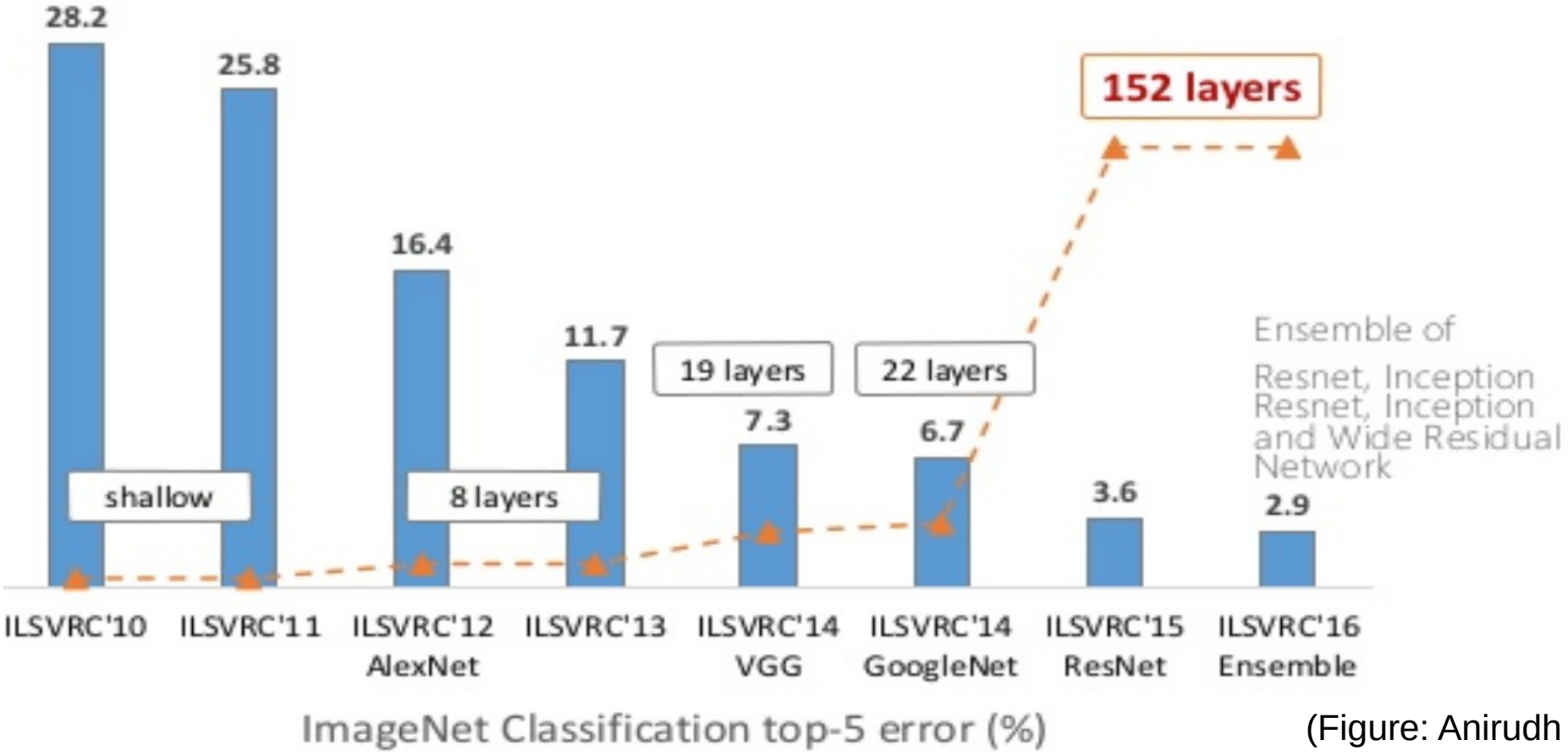
Deep ConvNets for Object Recognition (on GPU)

- AlexNet [Krizhevsky et al. NIPS 2012], OverFeat [Sermanet et al. 2013]
- 1 to 10 billion connections, 10 million to 1 billion parameters, 8 to 20 layers.



Error Rate on ImageNet

- **Depth inflation**

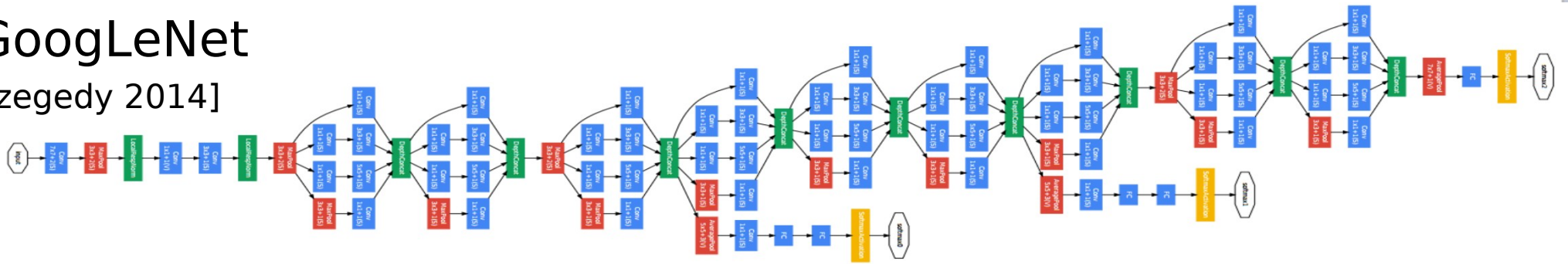


(Figure: Anirudh Koul)

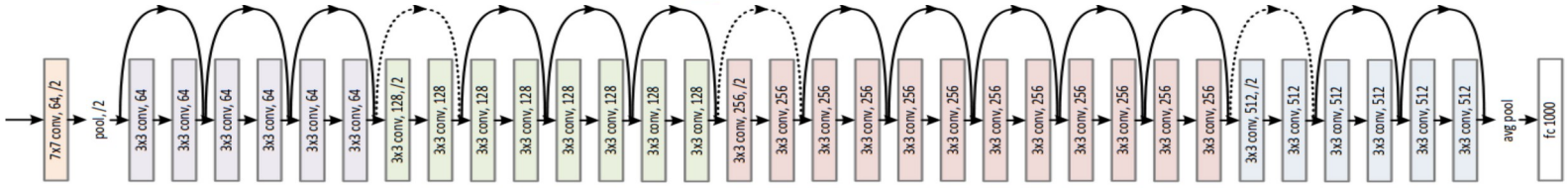
Deep ConvNets: depth inflation!

VGG
[Simonyan 2013]

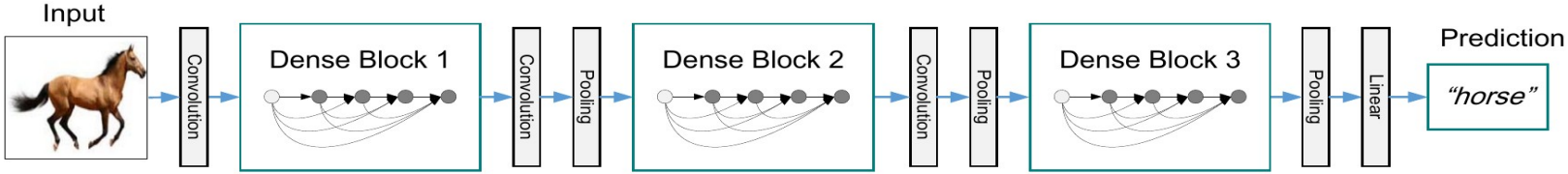
GoogLeNet
Szegedy 2014]



ResNet
[He et al. 2015]

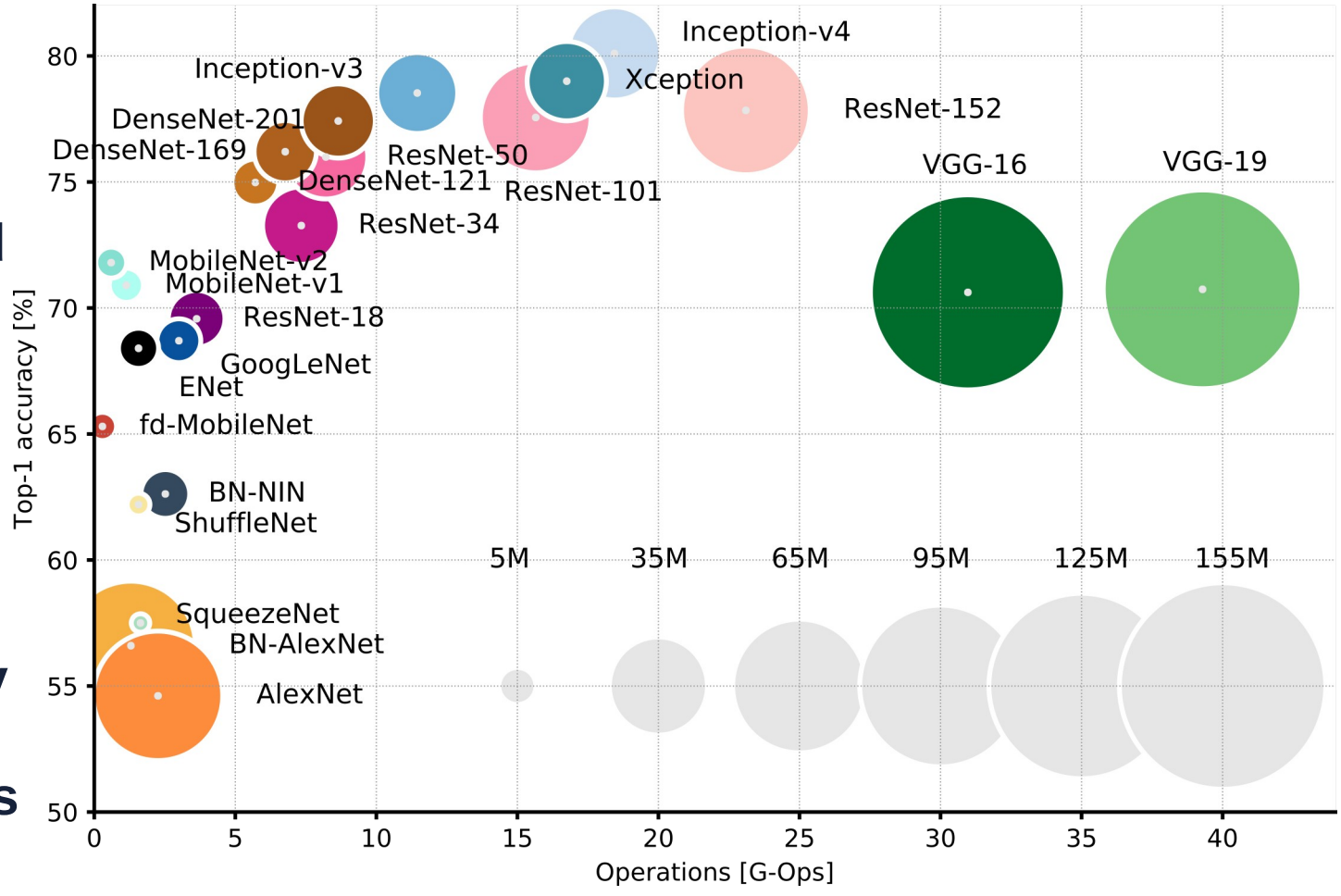


DenseNet
[Huang et al 2017]



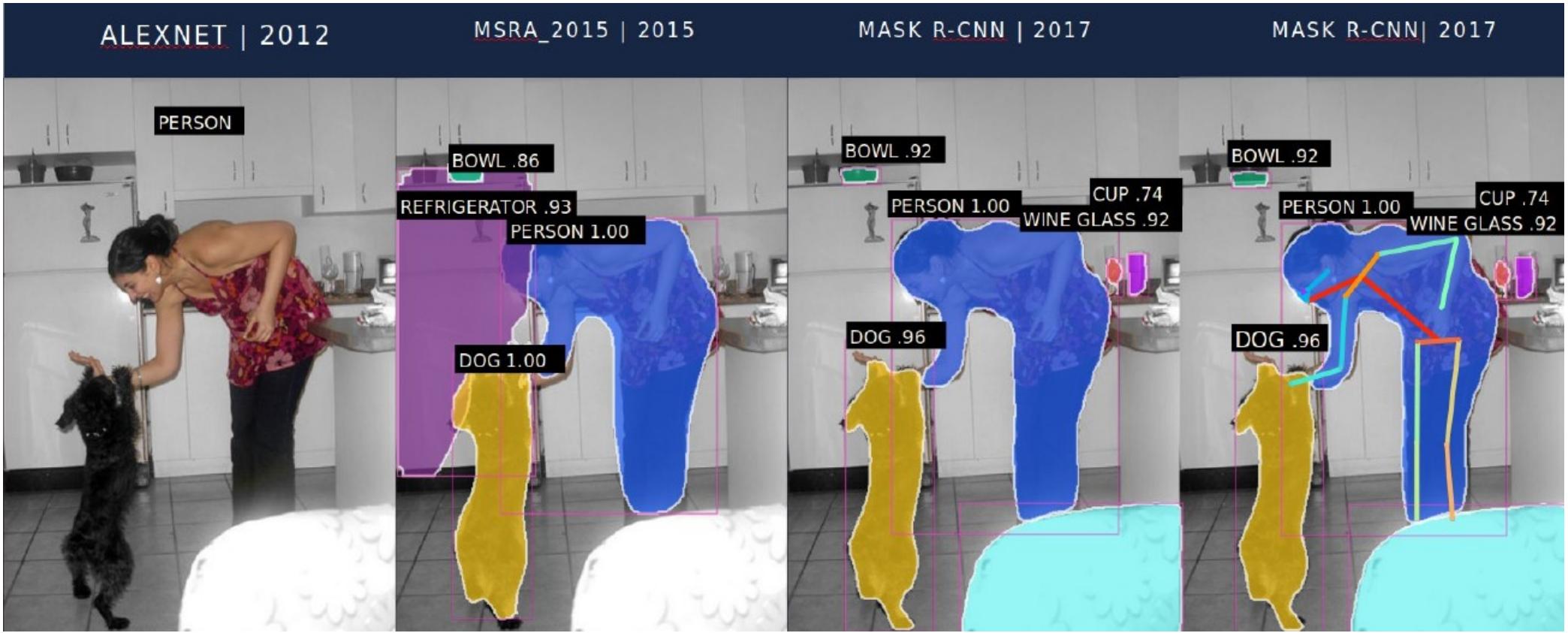
GOPS vs Accuracy on ImageNet vs #Parameters

- **[Canziani 2016]**
- **ResNet50 and ResNet100 are used routinely in production.**
- **Each of the few billions photos uploaded on Facebook every day goes through a handful of ConvNets within 2 seconds.**



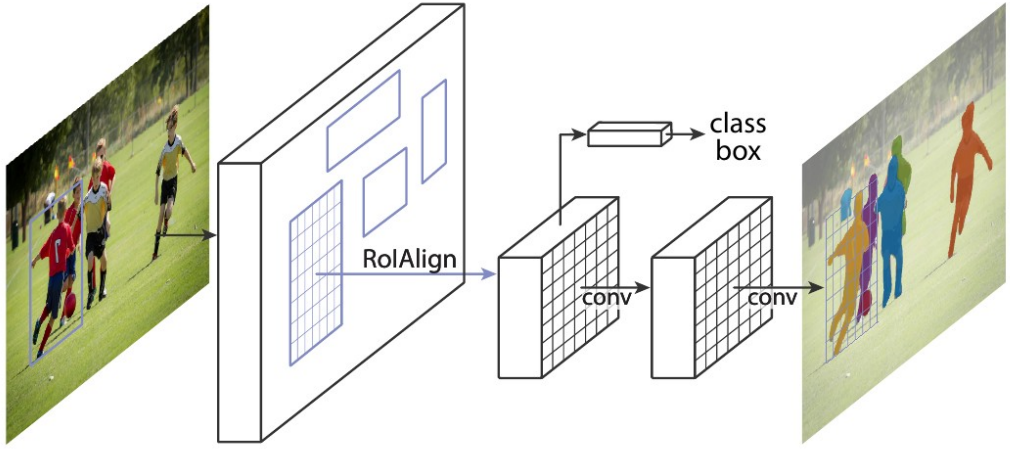
Progress in Computer Vision

- **[He 2017]**

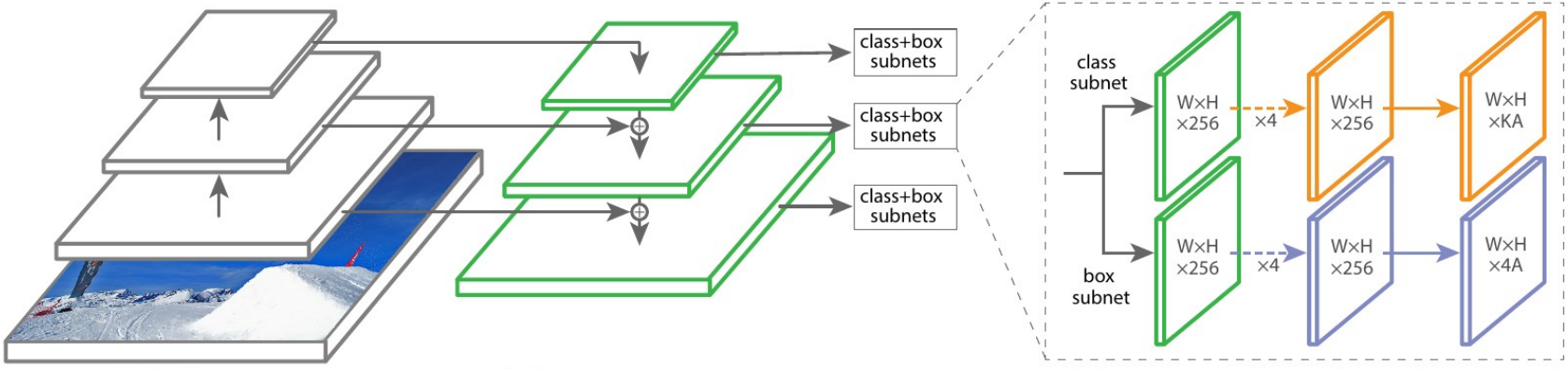


Mask-RCNN, RetinaNet, feature pyramid network

- **Mask-RCNN**
  - [He et al. arXiv:1703.06870]
  - ConvNet produces an object mask for each region of interest



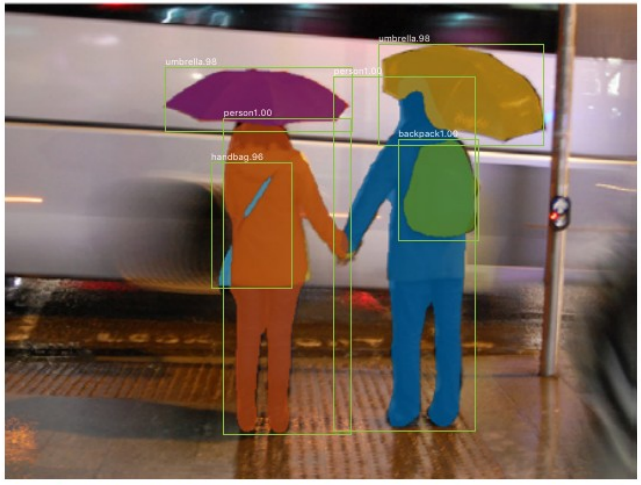
- **RetinaNet/FPN**
  - [Lin et al. ArXiv:1708.02002]
  - one-pass object detection

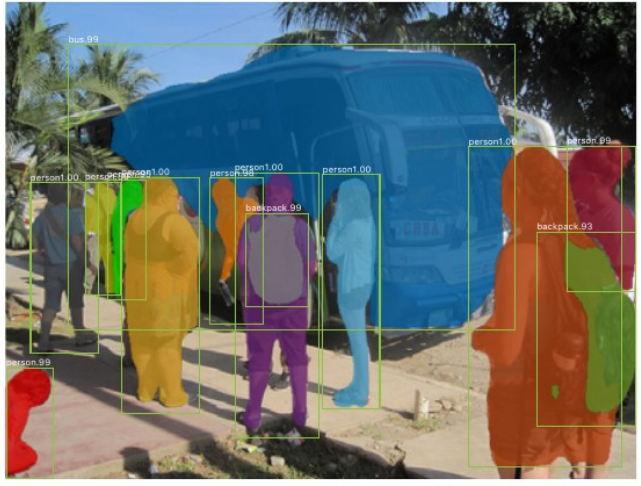


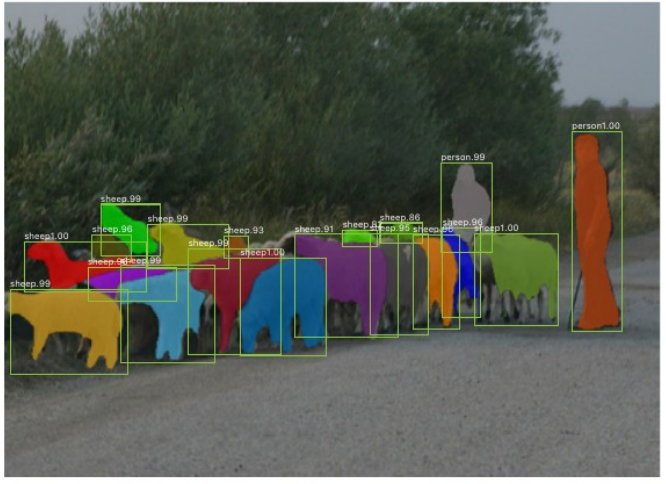
Mask-RCNN Results on COCO dataset

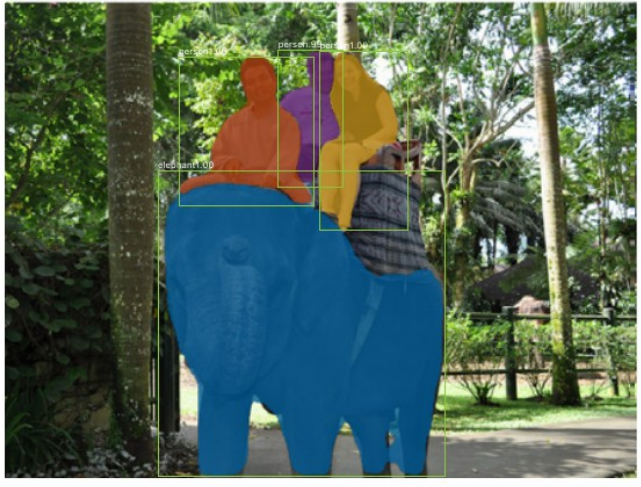
- **Individual objects are segmented.**

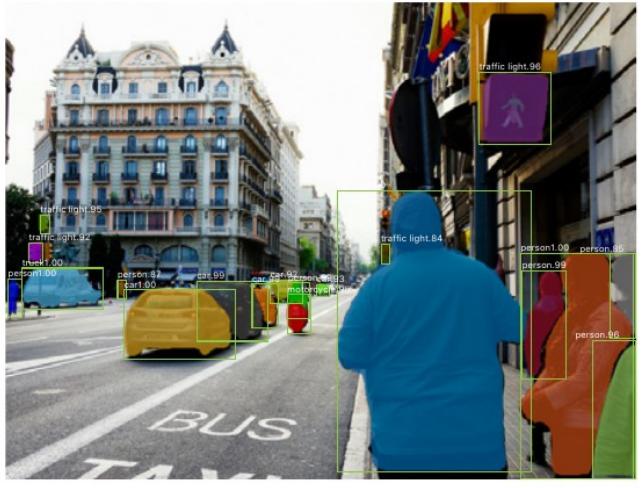
Mask R-CNN Results on COCO test set

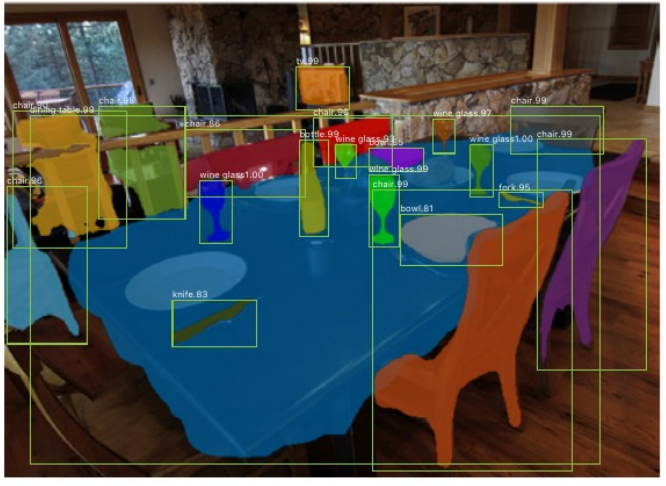






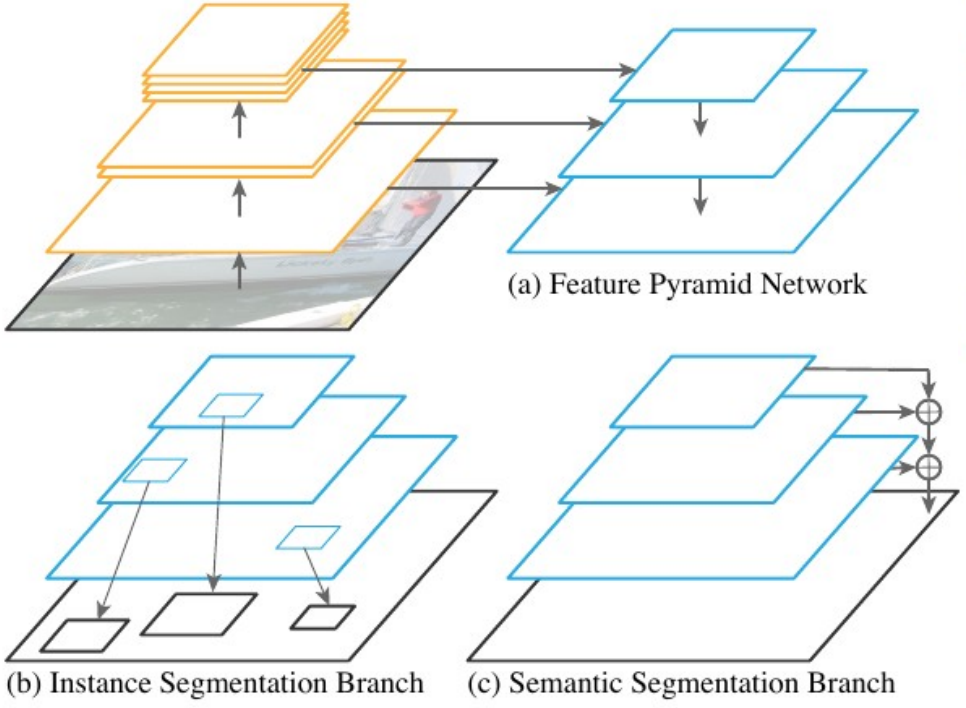






Panoptic Feature Pyramid Network

- **Segments and recognizes object instances and regions**
  - [Kirillov arXiv:1901.0244]

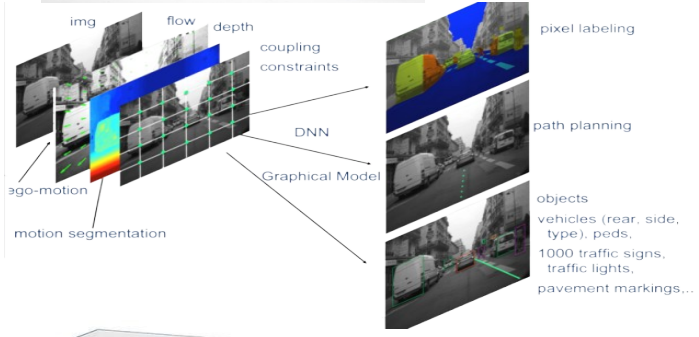


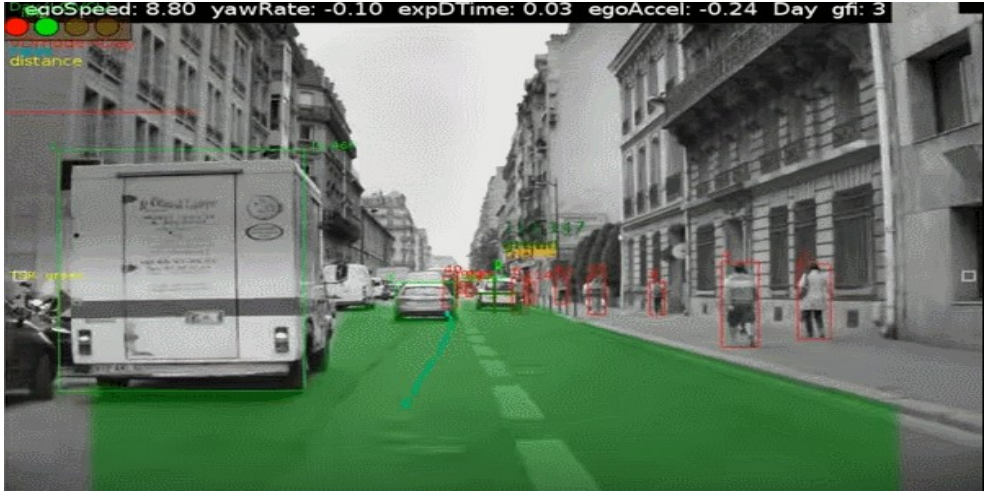
Detectron2 (FAIR) [Girshick 2019]

- **Panoptic instance segmentation, (dense) body pose estimation**
- **Open source: https://github.com/facebookresearch/detectron2**

Driving Cars with Convolutional Nets

- **MobilEye (2015)**

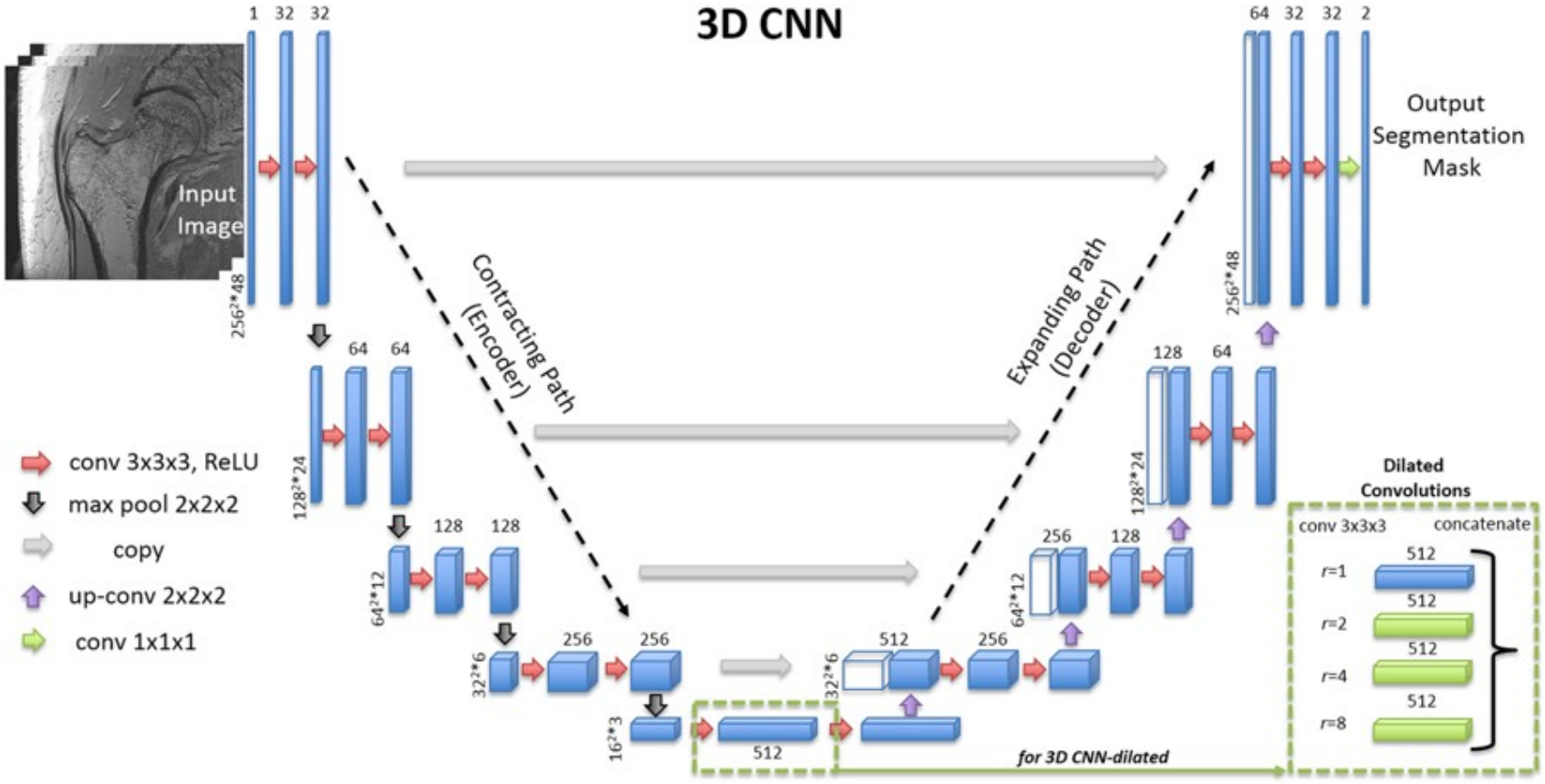




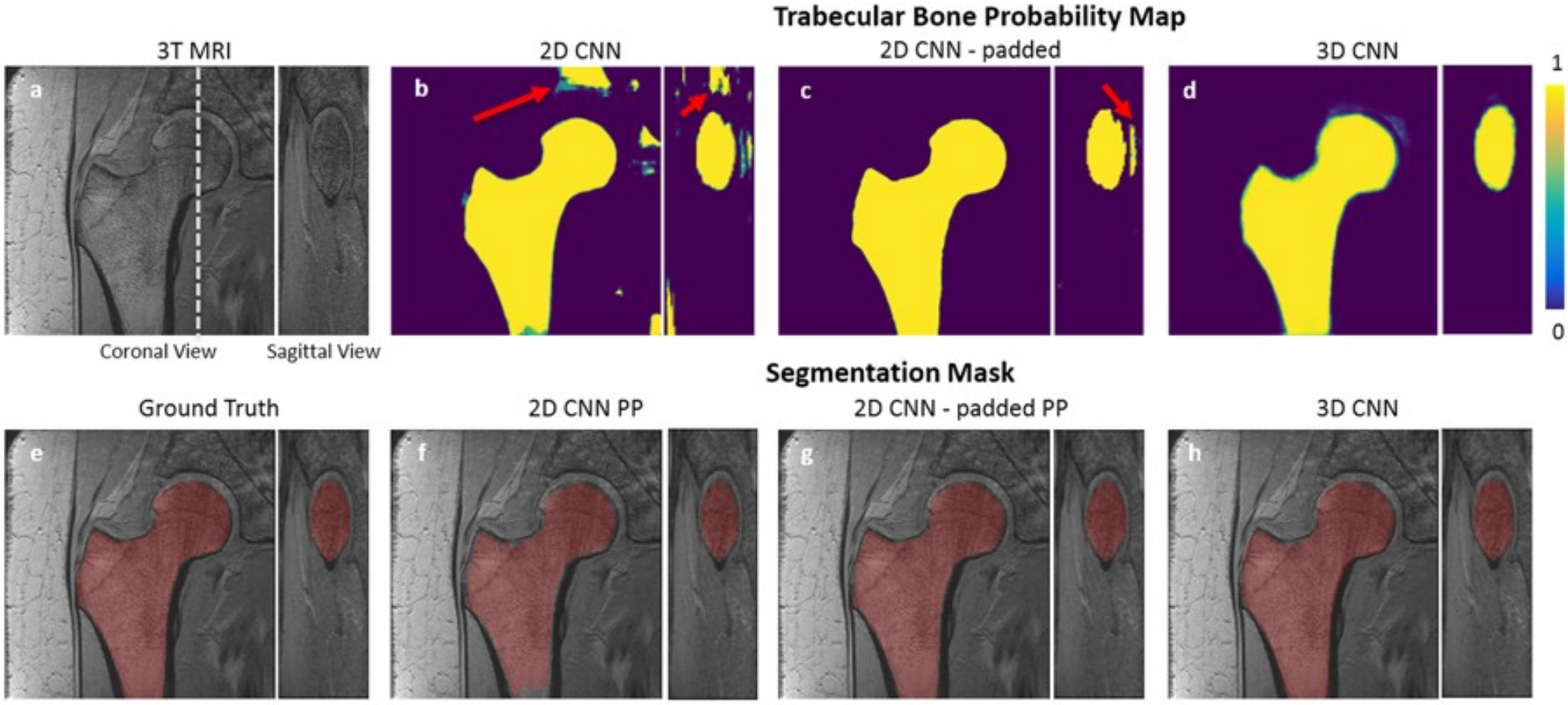
- **NVIDIA**

3D ConvNet for Medical Image Analysis (NYU)

- **Segmentation Femur from MR Images**
- **[Deniz et al. Nature 2018]**

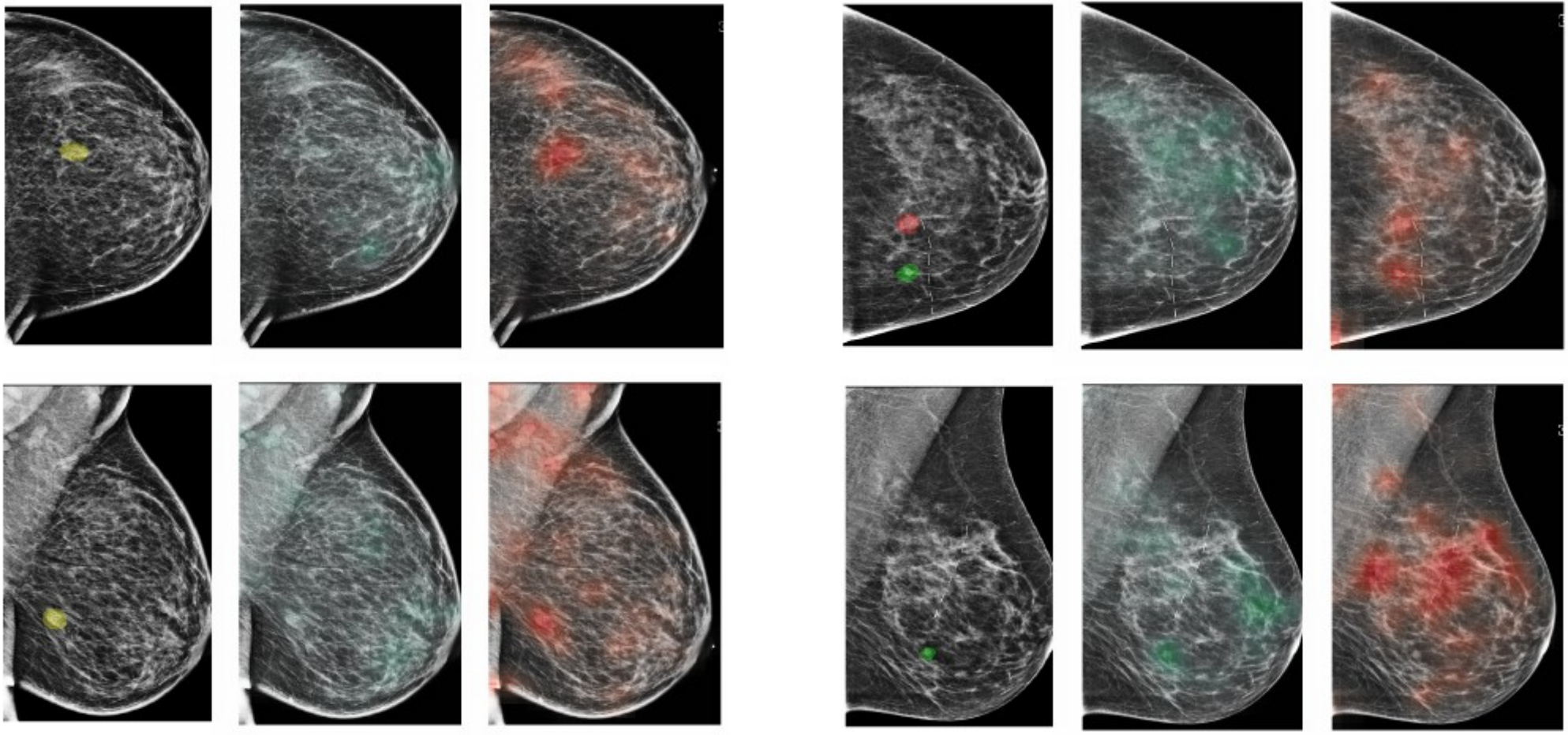


3D ConvNet for Medical Image Analysis (NYU)



Breast Cancer Detection (NYU)

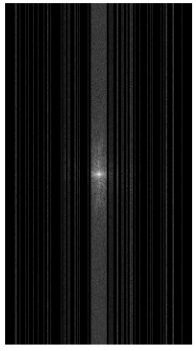
- **[Wu et al. ArXiv:1903.08297] https://github.com/nyukat/breast_cancer_classifier**



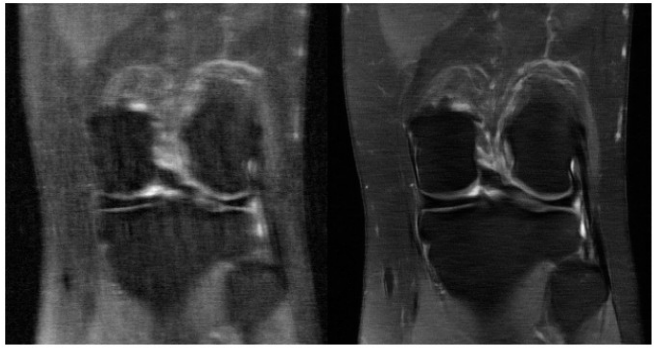
FastMRI (NYU+FAIR): 4x-8x speed up for MRI data acquisition

- MRI images subsampled (in k-space) by 4x and 8x
- [Zbontar et al. ArXiv:1811.08839]
- U-Net architecture
- 4-fold acceleration
- 8-fold acceleration
- K-space masks

(a) Cropped and vertically flipped reconstruction from fully sampled k-space data

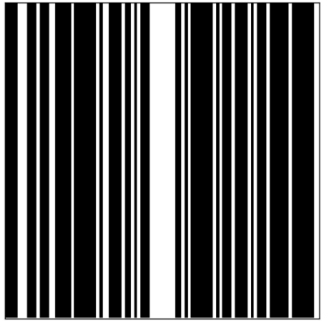


(b) Rectangular masked k-space

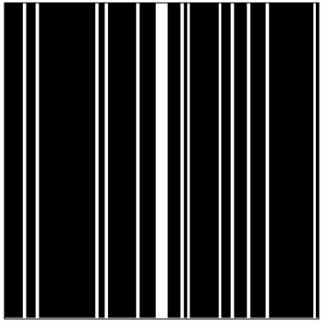


(c) Reconstruction via zero-filled IFFT

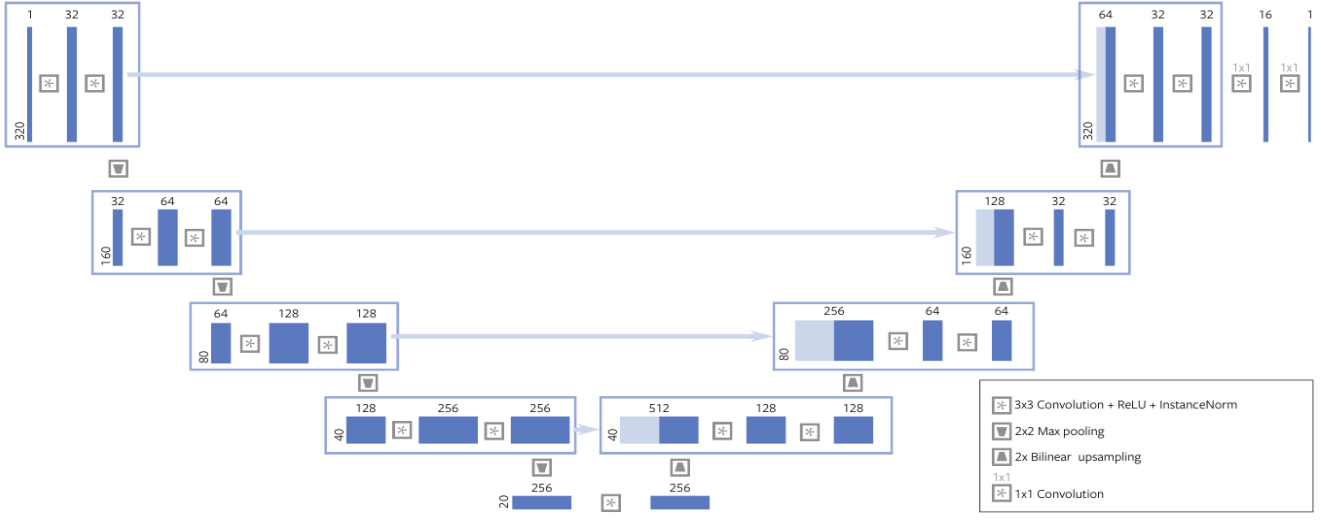
(d) Deep-learning baseline UNET reconstruction



(a) 4-fold acceleration



(b) 8-fold acceleration



ConvNets (and Deep Learning) in Physics

- **Approximate solutions of PDEs with a learned update**
  - Integration step of PDE solver: Z(t+1) = Z(t) + dt*G(Z(t))

    where is G() a translation-invariant local operator.

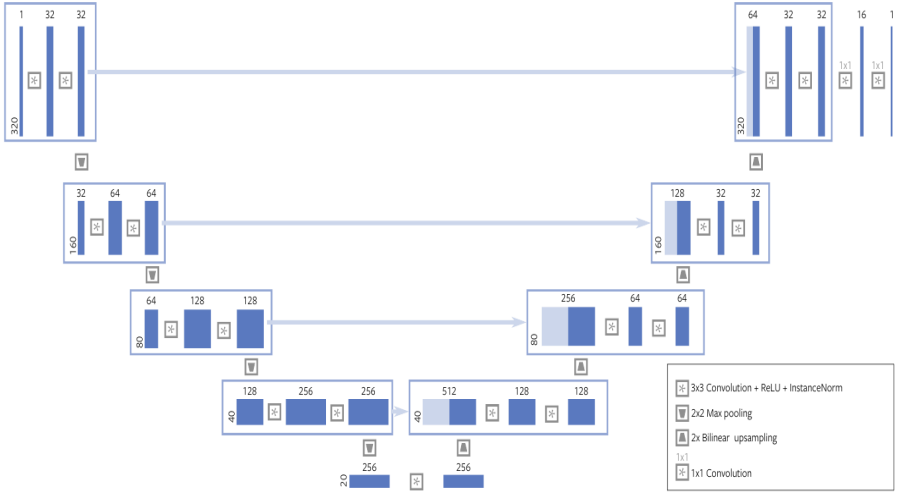
    Example: G(Z(t)) = V*f(W*Z(t)) conv->transfer_func->conv
- **High energy Physics**
  - Lattice QCD
- **Fluid Dynamics**
  - Prediction of aero/hydro-dynamical properties of solids
  - Shape refinement by gradient descent
- **Cosmology / Astrophysics**
  - Large-scale simulation of the early universe

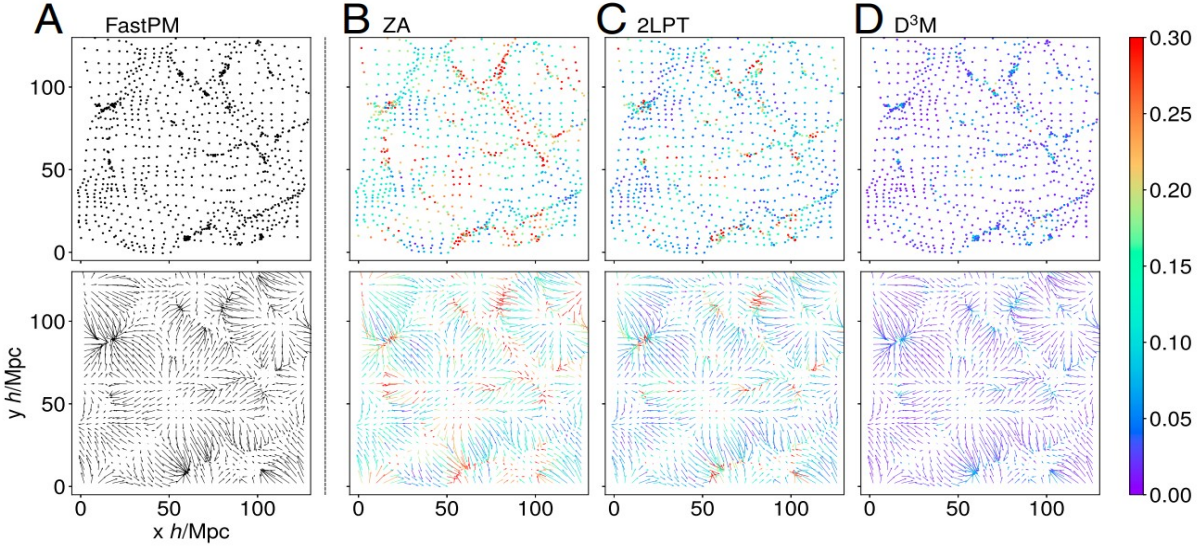
ConvNets in Astrophysics [He et al. PNAS 07/2019]

Learning to predict the cosmological structure formation

Siyu He[a,b,c,1], Yin Li[d,e,f], Yu Feng[d,e], Shirley Ho[a,b,c,d,e,1], Siamak Ravanbakhsh[g], Wei Chen[c], and Barnabás Póczos[h]

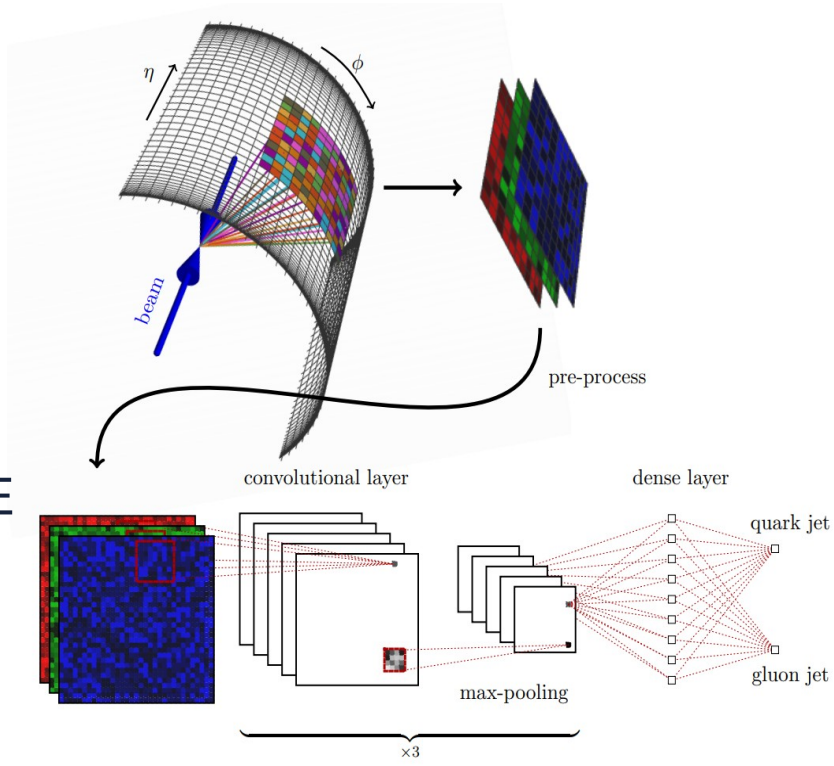
- 1. Train a coarse-grained 3D U-Net to approximate a fine-grained simulation on a small volume
- 2. Use it for a simulation on a large volume (the early universe)





ConvNets (and Deep Learning) in Physics

- **Material Science / Molecular dynamics**
  - Protein structure/function prediction
  - Prediction of material properties
- **High energy Physics**
  - Jet filtering / analysis
    - "Deep learning in color: towards automated quark/gluon jet discrimination", P Komiske, E Metodiev, M Schwartz, arXiv:1612.01551
- **Cosmology / Astrophysics**
  - Infering constants from observations
  - Statistical studies of galaxies,
  - Dark matter through gravitational lensing

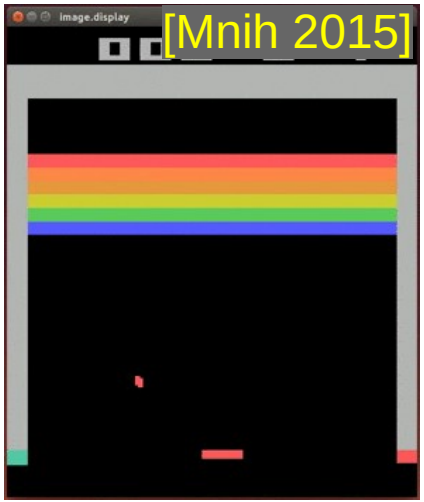


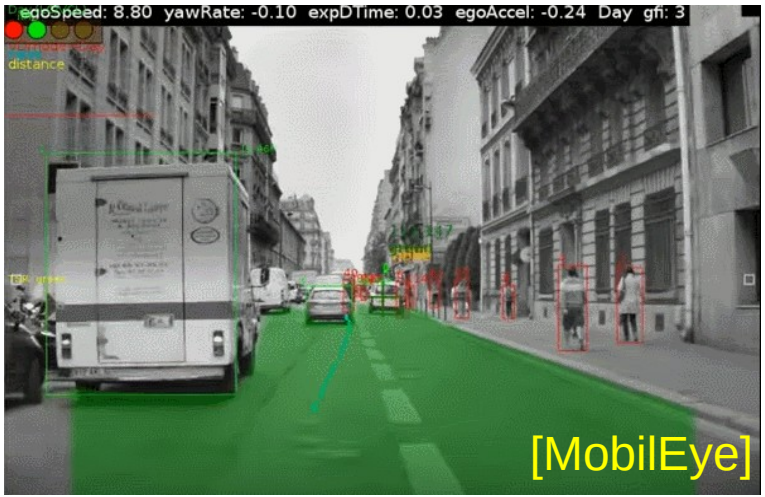
Applications of ConvNets

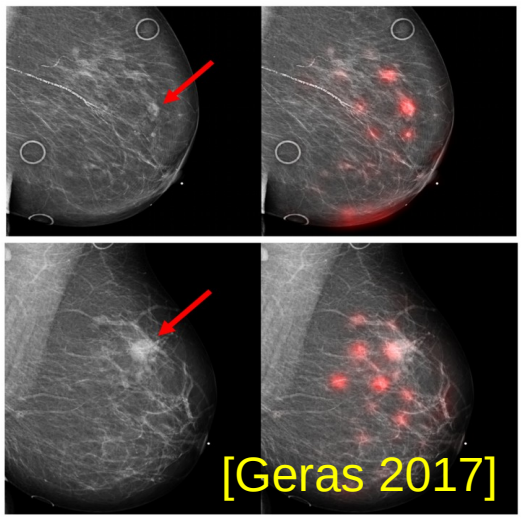
- **Self-driving cars, visual perception**
- **Medical signal and image analysis**
  - Radiology, dermatology, EEG/seizure prediction….
- **Bioinformatics/genomics**
- **Speech recognition**
- **Language translation**
- **Image restoration/manipulation/style transfer**
- **Robotics, manipulation**
- **Physics**
  - High-energy physics, astrophysics
- **New applications appear every day**
  - E.g. environmental protection,….

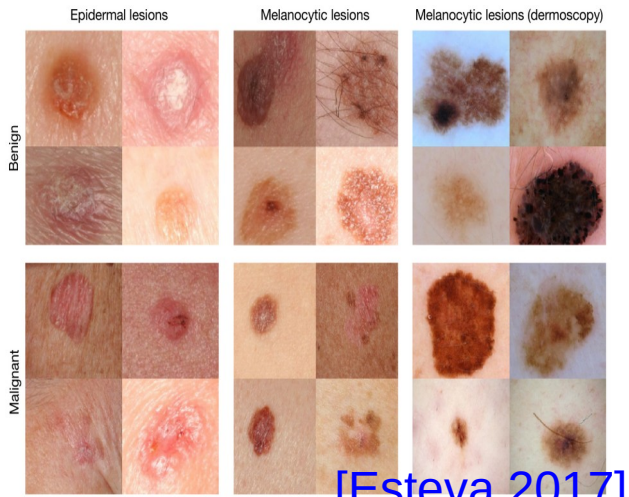
Applications of Deep Learning

- **Medical image analysis**
- **Self-driving cars**
- **Accessibility**
- **Face recognition**
- **Language translation**
- **Virtual assistants***
- **Content Understanding for:**
  - Filtering
  - Selection/ranking
  - Search
- **Games**
- **Security, anomaly detection**
- **Diagnosis, prediction**
- **Science!**

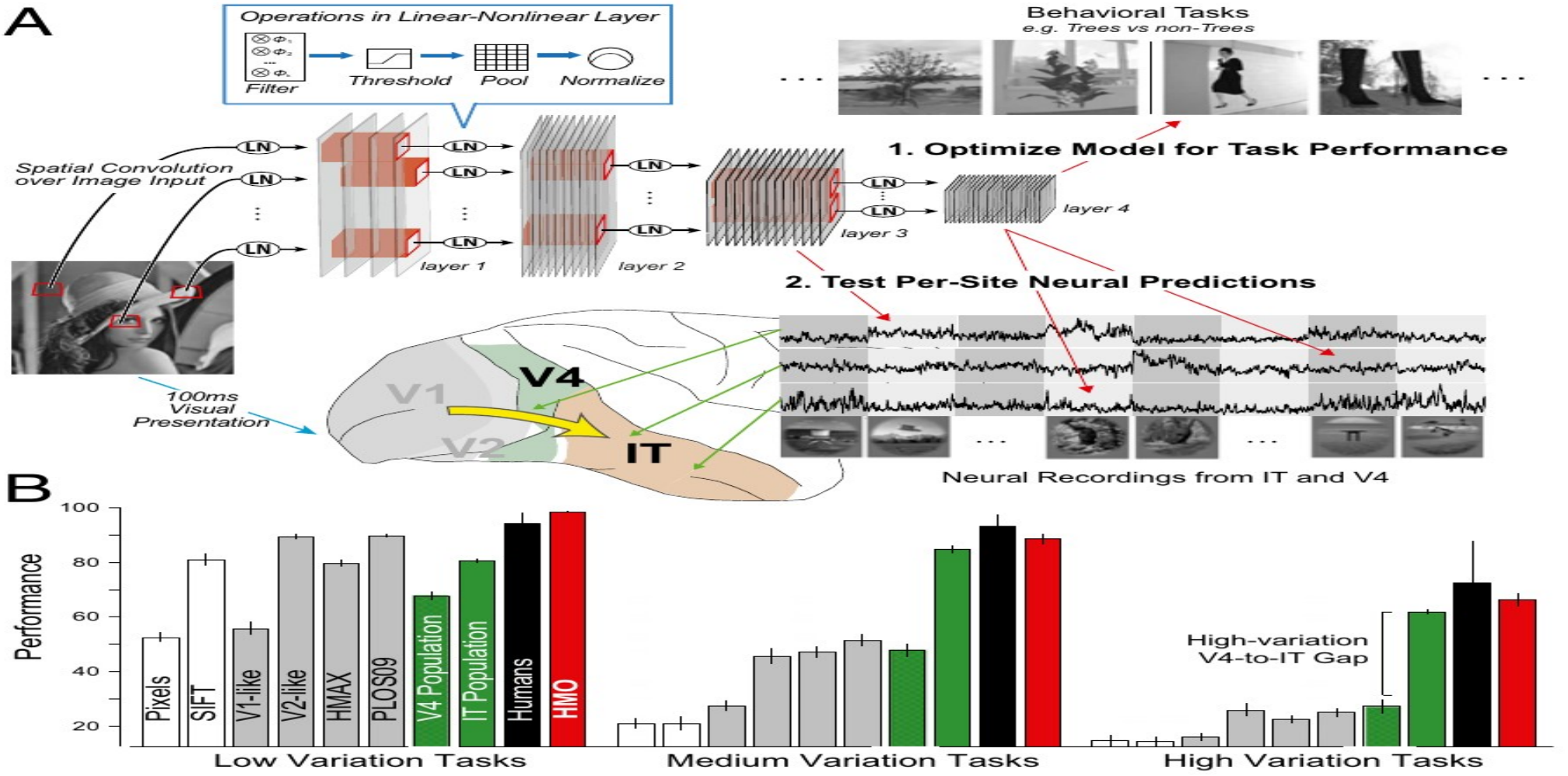








ConvNets & The Visual System [Yamins et al. PNAS 2014]



ConvNets as Models of the Visual System?

- **[Yamins & Di Carlo 2016]**

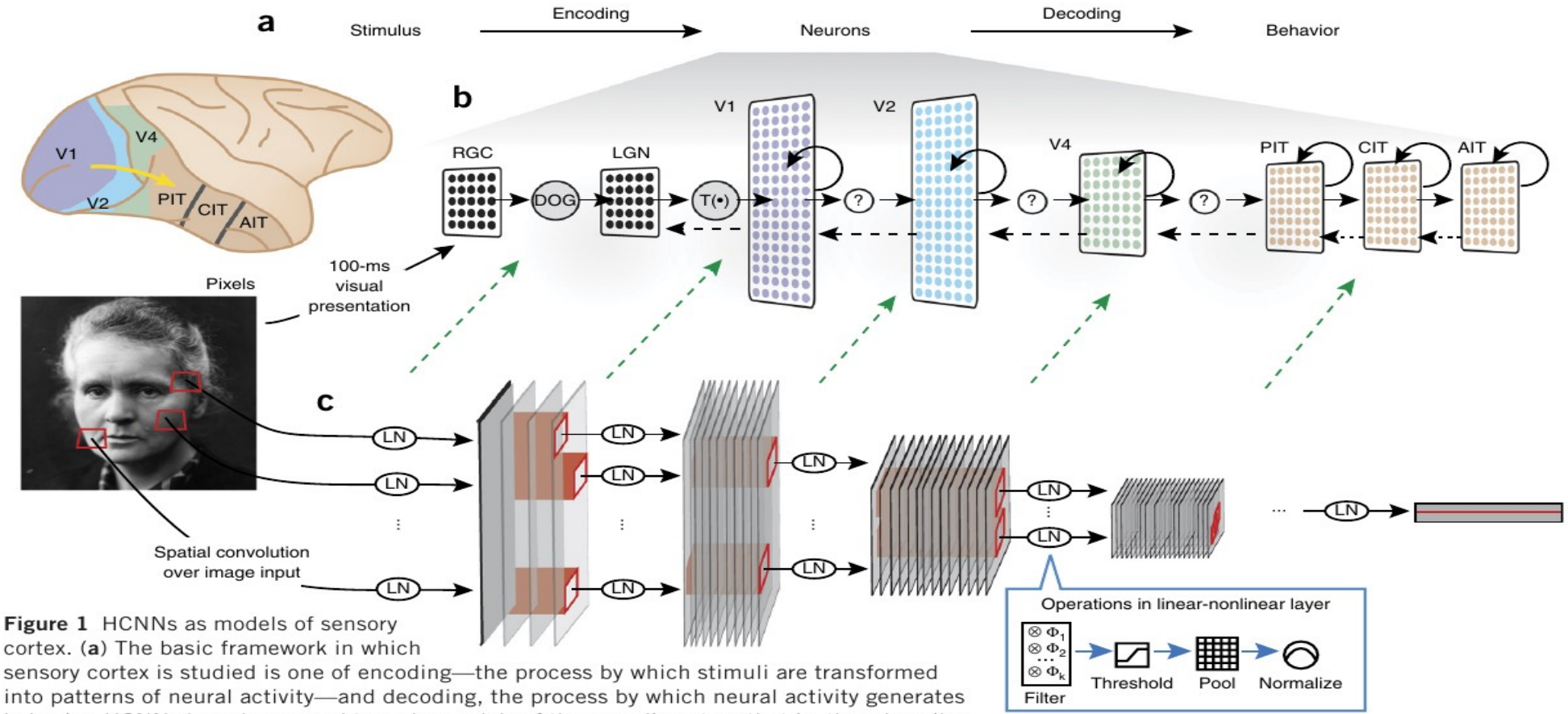
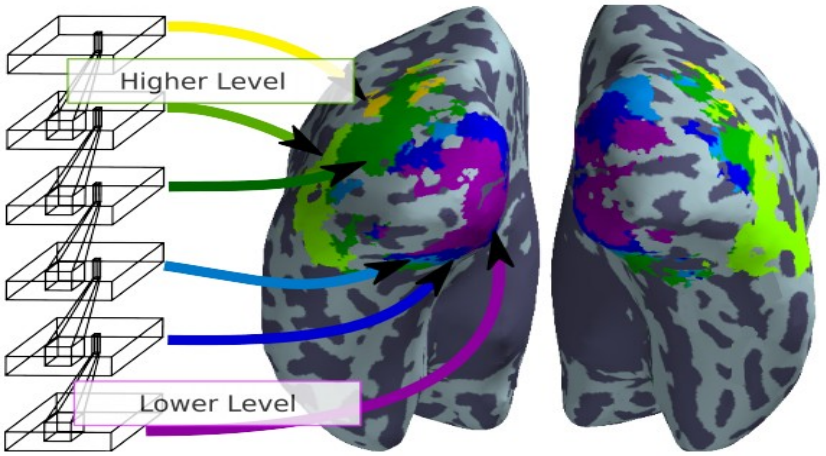


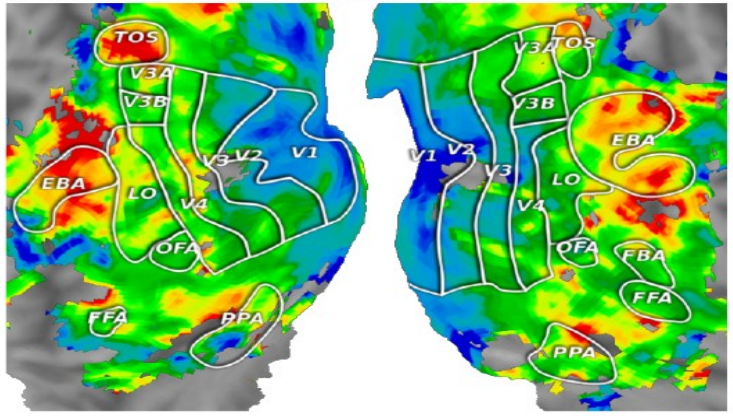
Figure 1 HCNNs as models of sensory cortex. (**a**) The basic framework in which sensory cortex is studied is one of encoding—the process by which stimuli are transformed into patterns of neural activity—and decoding, the process by which neural activity generates behavior. HCNNs have been used to make models of the encoding step; that is, they describe

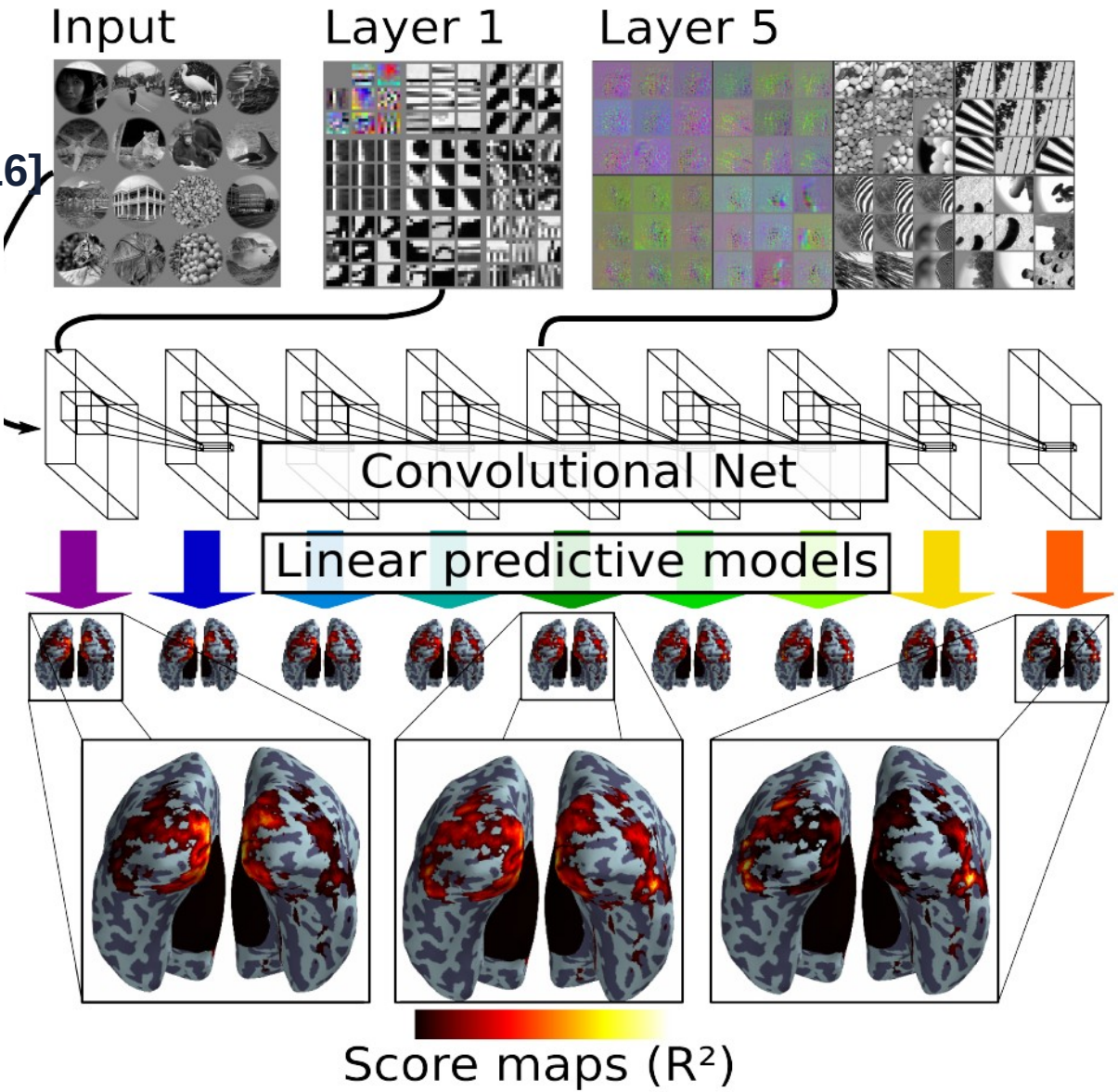
ConvNet models & fMRI

- **[Eickenberg et al. NeuroImage 2016]**



B Fingerprint summaries for Huth2012





Why does it work so well?

- **We can approximate any function with two layers**
  - Why do we need layers?
- **What is so special convolutional networks?**
  - Why do they work so well on natural signals?
- **The objective function are highly non-convex.**
  - Why doesn’t SGD get trapped in local minima?
- **The networks are widely over-parameterized.**
  - Why do they not overfit?

What current deep learning methods enable

- **What we can have**
  - Safer cars, autonomous cars
  - Better medical image analysis
  - Personalized medicine
  - Adequate language translation
  - Useful but stupid chatbots
  - Information search, retrieval, filtering
  - Numerous applications in energy, finance, manufacturing, environmental protection, commerce, law, artistic creation, games,…..
- **What we cannot have (yet)**
  - Machines with common sense
  - Intelligent personal assistants
  - “Smart” chatbots”
  - Household robots
  - Agile and dexterous robots
  - Artificial General Intelligence (AGI)